3 – Some Computer Science Problems in Ubiquitous

- O que nós fala este paper?
Este paper fala-nos de um novo conceito que é o de computação ubiqua, indroduzido por Marc Weiser. São depois apresentados vários prototipos de hardware para implementar computação ubiqua. Por ultimo são apresentados vários desafios/areas em desenvolvimento deste grande mundo ubicomp.

- O que é computação ubiqua (ubicomp)?
A computação ubiqua é uma nova era que se aproxima, em que existiram muitos computadores embutidos em paredes, móveis, roupas, carros etc, partihados por cada um de nós.
O termo computação ubíqua foi primeiramente sugerido por Mark Weiser em 1988 para descrever a sua ideia de tornar os computadores onipresentes e invisíveis. Isto é, a tentativa de tirar o computador do caminho entre o utilizador e o seu trabalho. O objetivo é ir para além da "interface amigável" e longe da realidade virtual. Ao invés de usar ao máximo todas os canais de entrada e saída do corpo, como na realidade virtual, a ideia é permitir que o utilizador faça o seu trabalho com o auxílio de computadores sem nunca ter que se preocupar em trabalhar nos computadores. Simplesmente melhorar as interfaces fazem do obstáculo (seu computador) nada mais que um obstáculo mais fácil de usar.

- Explicação de *calm technology*

Uma importante transformação, necessária a esta onipresença digital, é o conceito de *calm technology*.

Uma *calm technology* deve ser capaz de aproveitar o centro e a periferia de nossa atenção. Por exemplo, ao conduzir, a nossa atenção está concentrada na estrada, no rádio, na conversa com o passageiro, mas não necessariamente no barulho do motor. No entanto, um barulho anormal produzido pelo motor é imediatamente percebido, mostrando que nós estávamos sintonizados com o barulho ao nosso redor (nossa periferia) e que somos capazes de mudar nosso foco de atenção para um objeto periférico rapidamente, tornando-o um objeto central.

Essa possibilidade de movimentação entre periferia e centro permite que, ao enriquecer nossa periferia, sejamos capazes de controlar muito mais coisas do que colocando-as todas no centro. Ou seja, a periferia é capaz de nos informar sem nos sobrecarregar.

- Para que servem os prototipos de hardware? Explique alguns desses prototipos
Os prototipos de hardware são dispositivos desenhados para suporte a aplicações ubiquas.

> **Tabs**
Pequeno dispositivo portátil de entrada de informação, também chamado de ParcTab, com tela sensível ao toque, e conectividade constante.

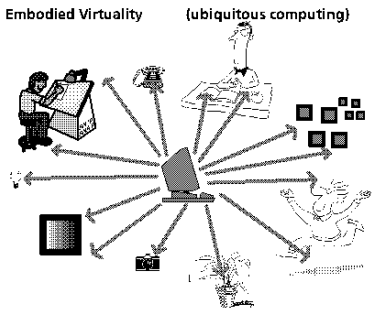
> **Pad**
Tem o mesmo tamanho que um notebook e possui caneta eletrônica e um microfone embutido. Foi projetado para permanecer fixo nos ambientes. Não é um dispositivo portátil.

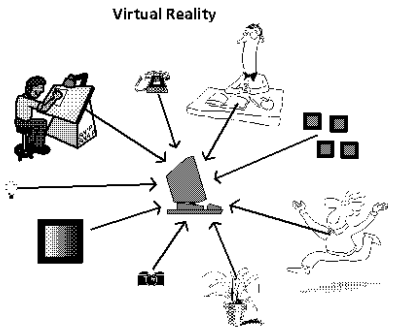
> **Board**
Telão sensível ao toque, grava os dados que são escritos através de uma caneta eletrônica.

> **Pen**
Dispositivo que simula uma caneta "escrevendo na tela", não é conectada (usa o infra-vermelho), não necessita encostar na tela. Deverá substituir completamente o teclado e o mouse.

- Faça a distinção entre Realidade Virtual e Computação Ubiqua

A Realidade Virtual tenta colocar o homem dentro do "mundo do computador" ao invés de colocar o computador no "mundo do homem". Ela força o utilizador a entrar num ambiente virtual (cyberspace). A Realidade Virtual controla os sentidos e emoções do homem, e ao tentar reproduzir o mundo no computador há uma grande perda de detalhe, que não pode ser melhorado a um custo razoável. A realidade virtual quer enganar o utilizador deixando o mundo físico para trás, ao contrário desta proposta que busca integrar o computador de maneira mais eficiente às actividades humanas.





- Apresente alguns desafios em computação ubiqua

● **Energia e consumo:** como diminuir o consumo de energia dos dispositivos por forma a terem uma maior disponibilidade?

Soluções: utiliza circuitos com menor tensão ou então utilizar circuitos assincronos que não utilizam relógio e funcionam para baixas tensões (ideias para dispositivos moveis).

● **Wireless**: a utilização de redes sem fios é uma necessidade da computação ubiqua. Mas tem de se resolver problemas: interferências (resultado de utilização das mesmas frequências); colisões (resolução com a utilização do protocolo de multiplo acesso MACA);

● **Localização**: O endereços IP utilizados actualmente não são apropriados para dispositivos móveis.

Solução: utilização de IPs virtuais (mecanismo de redundância), em que IP virtual fixo mapeia para IP real

● **Interacção**: Como interagir com dispositivos que tanto podem ser muito grandes como muito pequenos?

Soluções: Por voz, teclado, escrita e outros

● **Privacidade**: Com tantos dispositivos móveis é possível saber todos os movimentos de um indivíduo.

Conclusão

O Pervasive Computing está apenas em sua fase de pesquisa e sua implementação parece estar ainda distante, não por problemas das tecnologias isoladamente mas na sua integração, implantação e infraestrutura.

4 – Pervasive Computing: Vision and Challenges

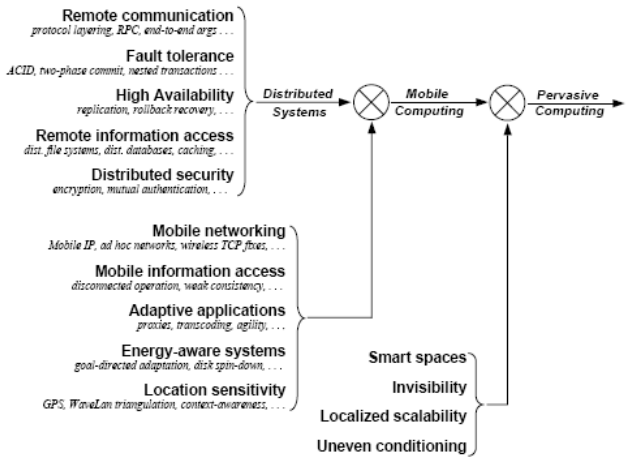
- O que nós fala este paper?

Este paper começa por falar na visão de Marc Weiser, que era bastante avançada para a altura e que proponha um novo tipo de computação, a **computação ubiqua**. A computação ubiqua também pode ser chamada de **computação pervasiva**.
A computação ubiqua como o nome indica, consiste em ominipresença, ou seja estar em todo lado e claro no contexto da computação consiste em sistemas capazes de tornar a interação dos utilizadores com os dispositivos de uma forma em que estes parecem totalmente invisiveis ao utilizador.

Na actualidade já existem tecnologias capazes de servir de suporte para esta visão, tanto na perpectiva fixa como móvel. O objectivo deste paper é perceber os vários dominios de investigação que podem ser feitos na computação móvel.

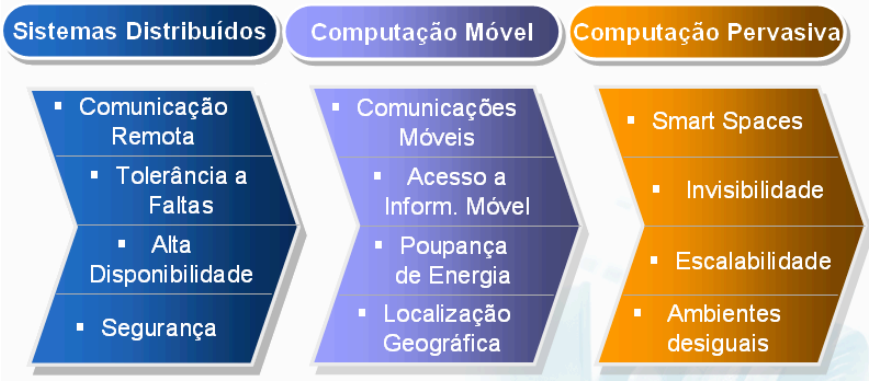
Este paper também nós dá 2 exemplos, que focam grandes detalhes dos sistemas ubiquos. Neste 2 exemplos é utilizada a arquitectura Aura, que é utilizada para desenvolver aplicações para computação ubiqua.

- Explique o enquadramento da computação pervasiva



Esta imagem mostra o enquadramento da computação pervasiva. A computação pervasiva lida com problemas de sistemas distribuidos e de computação móvel.

A evolução dos sistemas distribuidos e da computação movel, vêem portanto responder a muitas das necessidades da computação pervasiva.



Como podemos ver pela imagem, a computação pervasiva tem 4 Areas de investigação:
> **SmartSpaces**

Um active space, normalmente é um espaço fisico de tamanho limitado, como pro exemplo uma sala, em que existe uma infra-estrutura de software a correr por cima do mundo fisico. Um exemplo de um active space, seria uma sala em que existe um ajuste automático da temperatura e intesidade de luz, baseando-se no perfil electronico de um ocupante. A arquitectura de Middleware Gaia é utilizada para a implementação de active spaces

> **Invisibilidade**

Este aspecto tem a ver com o que o Marc Weiser disse, sobre o facto de as tecnologias de computação pervasiva terem de ser invisiveis para o utilizador, ou seja, aproximando-se da ideia de que tem de haver uma distracção minima por parte do utilizador, ao nivel do subconsciente, alertando apenas quando necessário.

> **Escalabilidade geográfica**

Este aspecto diz que deve-se **minimizar as interacções remotas**, para desta forma se poupar recursos, visto que as interacções só fazem sentido a nivel local.

> **Ambientes desiguais**

Os serviços podem variar de espaço para espaço, e como o utilizador é uma entidade movel, ao deslocar-se entre estes ambiente, surge com naturalidade o problema do novo ambiente não possuir os serviços com que estava a interagir no espaço anterior.

Desta forma a invisibilidade desaparece.

Uma solução para variação de ambientes seria o seguinte exemplo: mascarar a inexistência de cobertura wireless com capacidade de operação sem ligação.

- Explicação de um ambiente pervasivo com base na Arquitectura Aura
Cenário1- A Jane está na Gate 23 do aeroporto à espera do seu avião. Ela trabalhou num ficheiro e pretende envia-lo apartir da sua conecção wireless. Infelizmente a largura de banda é fraca porque há muitos utilizadores a navegar na net. O Aura observa a largura de banda corrente e sabe que a jane não tem tempo de enviar o ficheiro e apanhar o avião, por isso, consulta os serviços de voo e a rede do aeroporto, descobrindo que a largura de banda por wireless é excelente no gate 15, que só está a 3 minutos do sitio actual. O Aura envia um pop'up para o portatil da jane a avisar para que ela vá para a gate 15. No final, o Aura avisa quando estiver tudo transferido enquando a joana via TV CNN, e chegaria a tempo para a chamada na gate 23.

Neste exemplo, está claro a *proatividade* por parte do Aura que utiliza o comportamento da Jane para saber o que fazer. Também a obtenção de informação dos serviços presentes no active space por parte do Aura (Ex: congestionamento nas gates, tempos de voo, distância entre gates).

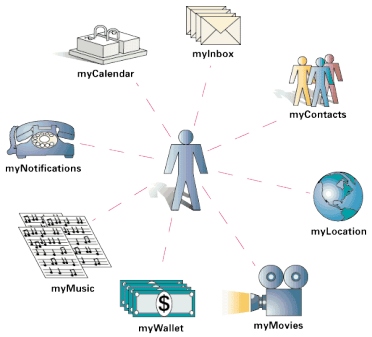
Senario2: O Fred está no escritorio a trabalhar na sua apresentação, que é dentro de 10 minutos, do outro lado do campus. O Fred não está pronto, mas leva o seu PDA wireless. O Aura transfere o estado do seu trabalho do PC para o PDA, e permite que ele faça alterações com comandos de voz durante o caminho. O Aura determina aonde é o local da apresentação consultando o calendário do Fred e de seguida contactando um serviço de localização. Depois o Aura faz o download do PDA para o projector e aquece o projector. O fred faz as ultimas actualizações antes de entrar na sala, e o Aura faz a actualização do ultimo trabalho do Fred para a apresentação. Durante a apresentação o Aura detecta que existe alguem que não é familiar e notifica o Fred para não mostrar um slide com informação importante para o negocio, e o Fred assim passa esse slide.

Neste exemplo, vimos a capacidade de execução movel por diversas plataformas (de um desktop para um PDA e de um PDA para um computador de projecção). Capacidade de execução de tarefas por voz. Mais uma vez está presenta a proactividade, quando o Aura tem a iniciativa de aquecer o projector e descarregar o trabalho do fred para o projector. O valor do active space é notavel, porque o Aura conseguiu saber aonde era o local da apresentação graças aos serviços de calendario online e de localização.

O grande desafio será integrar diferentes tecnologias num sistema como o Aura

"O todo é muito superior à soma das partes"

Já temos a tecnologia, só nós falta integra-la num sistema como o Aura



- Mencione vários desafios com os quais podemos deparar-nos em computação pervasiva

> Intenção do Utilizador

Para a proactividade ser eficiente, é absolutamente crucial que o sistema computacional consiga obter a intenção do utilizador. Senão não será possivel determinar que operações o sistema terá de realizar para ajudar o utilizador.

Exemplo: O utilizador está a ver um filme o SawIII apartir de uma conecção de rede por *streaming*, e de repente a largura de banda diminui drasticamente. O que fazer?

a) reduzir a fidelidade do filme

b) realizar uma pausa no video para tentar encontrar outra conecção

c) informar o utilizador que a sua tarefa já não pode ser realizada

R: Tudo depende da escolha do utilizador

Questões:
- A intenção do utilizador pode ser inferida, ou tem de ser explicitamente fornecida?
- Como pode a intenção do utilizador ser representada internamente? Quanto rica terá de ser essa informação para ser considerada útil? Quando e onde será actualizada?
- Até que nivel será melhor aceitar ou desprezar informação da intenção do utilizador?

> Cyber Foraging

Cyber foraging consiste em aumentar dinamicamente os recursos de um computador móvel wireless, à custa de uma infraestrutura de hardware wired. Deste conceito surgue o **surrogate** que é responsavel por prestar assistência à entidade móvel que necessite. Um surrogate pode fazer de gateway para acesso à Internet e também permite cache miss (explicado em maior detalhe noutros papers).

Exemplo: um computador móvel entra na vizinhaça, ele primeiro detecta os surrogates que existem e negocia com eles. A utilização de surrogates permite suportar a carga computacional elevada, para que os dispositivos móveis não consumam muitos recursos de energia e de memória.

Questões:
- Como é que descobrimos a presença de surrogates?
- Qual o mecanismo de descoberta de serviços que melhor se adapta? (Jini,UPnP, BlueTooth detecção de proximidade)
- Como se pode estabelecer um nivel apropriado de confiança com o surrogate?
- Como é feito o balanciamento de carga nos surrogates?
- Com que antecedência tem que ser instanciado um *surrogate*para para ser utilizável?

> Estrategias de adaptação

Quando existe um diferença substancial entre quem fornece e recebe os recursos, tem de ser tomadas estrategias. Um recurso pode ser: a largura de banda da ligação wireless, enegia, memoria, etc.... São 3 as estrategias:

1ª Os utilizadores podem configurar as suas aplicações para que estas reajam de determinada forma para determinado recurso (Ex: do Odyssey)

2ª O cliente pode perguntar ao ambiente para garantir um determinado nivel de recursos (Abordagem baseada em Qos).

3ª Acção correcta – um cliente faz uma determina acção e o fornecedor desse recurso responde com a devida acção. (Ex: quando o Aura avisa a Jane para caminhar para a gate15 para obter maior largura de banda).

Questões:
- Como escolher entre estratégias?
- Como honrar reserva de recursos?
- Como gerir a fidelidade das aplicações e serviços?
- É a estratégia de acção correctiva realista ou simplesmente demasiado intrusiva?

> Gestão de energia de alto nivel

Essencial para compensar o que a evolução das baterias não cobre hoje em dia

Questões:
- Pode toda esta *awareness*ser utilizada na gestão de energia?
- Quanto intrusiva pode ser?
- Podemos utilizar o "conhecimento da intenção"?

- Será possível utilizar *smartspaces*, *surrogates*e execução remota de modo a diminuir necessidades energéticas moveis? …

> Context Awareness

O sistema tem de ser context-aware, ou seja, tem de estar consciente do contexto do cliente para poder utilizar essa informação e realizar acções.

Questões:
>Qual a representação interna do contexto?
>Problemática acesso/validação/renovação de contexto
>Que serviços (mínimos) são necessários para nos permitir esta informação de contexto?
>Classificação de informação de contexto…mesmo o histórico?

Balanciamento de proactividade e transparência

Para que o dispositivo não seja nem incómodo nem inútil

Exemplo: Um utilizador que "não perceba" de informática e outro completamente à vontade com este "mundo"…

- Esperam comportamentos diferentes do mesmo dispositivo.

Questões:
- Até que ponto a Pró-Actividade ajuda?

Privacidade e confiança

A privacidade já é complicada em sistemas distribuidos e em computação movel, mas agrava-se ainda mais em computação prevasiva.

Questões:
- Gestão de privacidade?
- Gestão de confiança?
- Que técnicas de autenticação podem ser utilizadas?
- Como expressar toda a panóplia de informação de autorização?

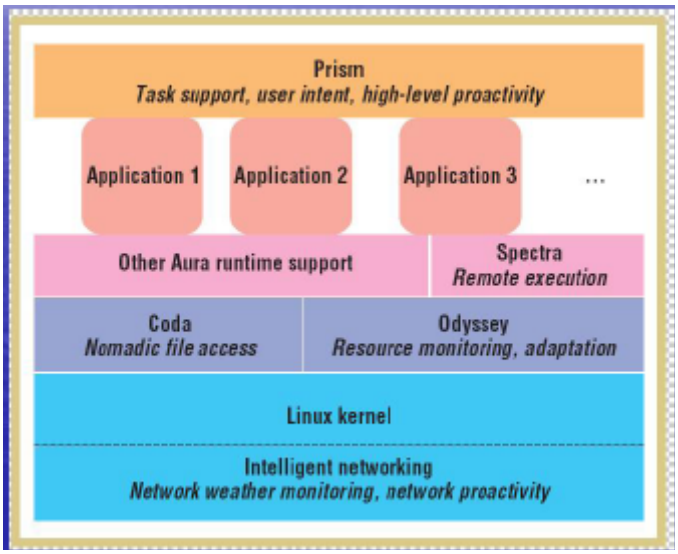
5 – Project Aura

- O que nos fala este paper?
Este paper fala-nos de uma arquitectura para desenvolver aplicações, como o Aura. Ao longo do paper é explicado como funciona cada componentes da sua arquitectura. Também são apresentados 2ªs implementações como o PhD e o IdeaLink.

- O que é o Aura?
É uma Arquitectura desenvolvida para desenvolver aplicações de computação pervasiva. Também tem como objectivo diminuir a distracção do utilizador. Caracteriza-se por dois conceitos fundamentais: self-tunnig e proactividade.

- Faça a distinção entre self-tunnig e proactividade
Self-tunning consiste na adaptação do sistema aos recursos existentes e desempenho a alcançar, enquanto que proactividade consiste em antecipar necessidades/requisitos do utilizador.

- Mostre e explique as camadas da Arquitectura do Aura



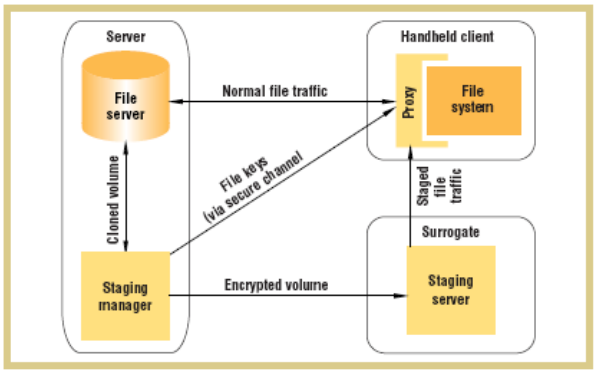
Prism: Camada responsável por capturar e gerir a intenção do utilizador.
Coda: Adaptação à largura de banda e permite desconexão (em computação móvel)
Odyssey:monitoriza e adapta de forma conciente a aplicação em causa.
Spectra: É um mecanismo de execução remota. Adapta-se ao contexto para melhor executar uma chamada remota.

- Que serviços é que o Aura fornece? Explique-os.
O Aura fornece os seguintes serviços:

Cyber foraging: amplifica a capacidade dos clientes : amplifica a capacidade dos clientes com recursos limitados

Como foi explicado no paper anterior, este serviço consiste na utilização de dispositivos que residem nas proximidades do cliente, ou seja, na sua rede, por forma a aumentar as capacidades do cliente (como memoria, capacidade de armazenamento e largura de banda). Estes dispositivos sao os surrogates, que vão permitir uma maior largura de banda para acesso ao exterior da rede (no caso de se querer aceder à Internet).

O Cyber foraging com uma arquitectura de **data staging** vai permitir resolver o problema do uso abusivo de cache por parte do cliente, ao armazenar a sua informação num surrogate. Basta pensarmos que: o cliente pode ter recursos limitados (como memoria) logo cache muito limitada; pode ocorrer desconexão e o clientes perder informação; ficheiro que não estão em cache podem tornar-se necessários.



componentes Laranja – representa o sistemas de ficheiros distribuido pre-existente.
componentes Amarelo – mostram as modificações para suportar dataStaging.
proxy – intercepta e redirecciona trafico do sistema de ficheiros

Funcionamento: se o trafico for para o surrogate, entao redirecciona esse trafico para o surrogate, caso contrário, redirecciona para o distante file server.

staging manager – permite a criação de snapshots do file server.

Funcionamento: o staging manager contacta o file server para criar uma snapshot, depois cifra e transmite essa imagem cifrada para o staging server.

As chaves dos ficheiros ou volume, sao enviadas por um canal seguro para o cliente.

Wireless Bandwidth Advisor: fornece informação sobre as condições da rede.
Este serviço tem como objectivo:
- sondar periodicamente os Acess Points
- obter informação sobre a rede
- prever a largura de banda disponivel no futuro

A tecnologia wireless está sujeita a muitos factores como sobrecargas, intereferências e ruido. Por isso, foram pensadas 3 soluções:

PPREV – prevê que os valores futuros são os mesmos que os observados recentemente. Este modelo trabalha melhor para celulas de alta utilização, com trafego bastante elevado.

AV – prevê os valores futuros com base numa média de amostras (por exemplo 10).

Arfima – prevê os valores futuros com uma média dinâ mica de valores passados (amostras de 10 por exemplos). Este modelo é bom em determinar padrões de trafego, por isso trabalha melhor em células de utilização baixas.

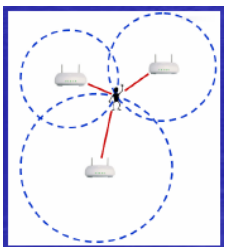
The WaveLAN-based people locator: localizar informação
Este serviço tem como objectivo:
- Desenvolver algo simples, pouco dispendioso, escalavel
- Alternativa válida ao GPS

Como solução, baseia-se na força do sinal e informações do Acess Points:

CMU-PM – Determina a localização através da medição da força do sinal de um computador em relação aos vários access points.
Compara a força do sinal medido com os valores de uma tabela.
Necessita de treino prévio do sistema

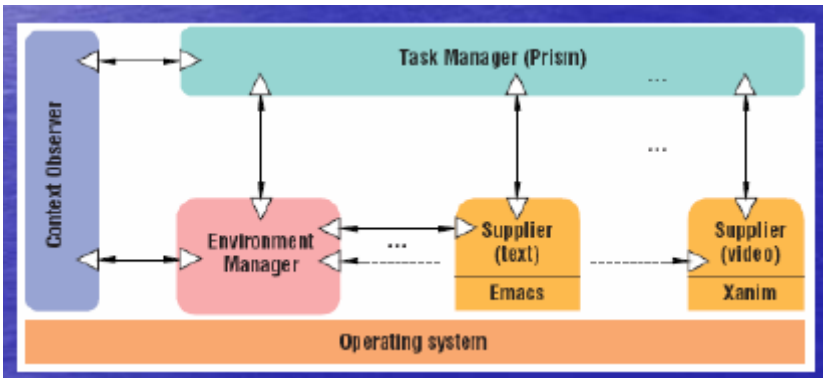
Confidence (%)	Strength (dBm)	Range (ft)
68.6	+/- 0.939	+/- 5
95.4	+/- 1.146	+/- 10
99.9	+/- 2.817	+/- 15

CMU-TMI – Necessita de saber a localização física de todas as AP's
Possuí uma função que converte o sinal na distância
Baseia-se nas observações efectuadas



Capturing user intent: adaptação do sistema às necessidades do utilizador.
- Permite que o utilizador ao trocar de ambiente continue a trabalhar nas mesmas tarefas
- O sistema fica continuamente a monitorizar o ambiente por forma a detectar possíveis mudanças de contexto
- Adaptar as tarefas a essas possíveis mudanças
- Obter a intenção Obter a intenção do utilizador
- Antecipar as necessidades do utilizador

A solução passa pela componente da arquitectura: **task layer (Prism)**

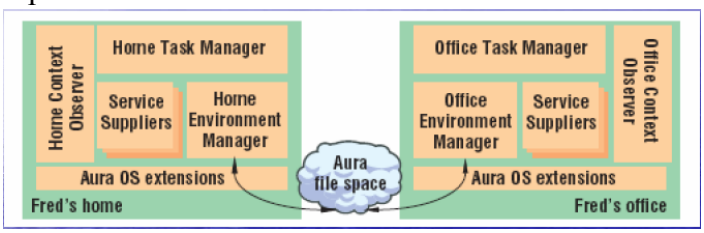


Context oberver: Notifica o Task Manager para possíveis alterações de ambiente.

Task Manager: Alerta o Environment Manager da alteração do contexto e adapta as tarefas ao novo ambiente (procura e configura aplicações)

Environment Manager: É responsavle por pausar os serviços.

Exemplo: casa -> escritorio



- Que aplicações conhece que tenham sido desenvolvidas para o Aura?

O IdeaLink e o PHD (Portatable Help Desk).

O IdelaLink é:

- Ambiente virtual para cooperação entre utilizadores móveis
- Tem como objectivo fornecer acesso fácil à informação
- Disponibiliza aos utilizadores um whiteboard por forma a partilhar ideias

O PHD é:

- Aplicação que permite consultar a localização e agenda de um utilizador
- Protótipo que apresenta mapas com indicações pessoas e recursos proximos.
- Possui interfaces de audio e video.

6 – Social Serendipity

- O que fala este paper?
Este paper fala-nos da utilização da computação ubiqua com o objectivo de socialização. Neste paper são descritas várias aplicações desenvolvidas para esse fim, como o BlueAware e o BlueDare. Tudo com base na arquitectura Serendipity, onde existe um servidor central que guarda as informações dos utilizadores, como os seus perfis e BTIDs. Na segunda parte do paper continua-se o mesmo tema mas já não com o Serendipity mas com outras abordagens como o BlueDating e outras aplicações. Neste paper estas aplicações são desenvolvidas para dispositivos que suportem a tecnologia bluetooth.

- O que é o BlueAware? Qual o seu modo de funcionamento?
BlueAware é uma aplicação MIDP2 projetada para funcionar passivamente em background de telefones com Bluetooth. O elemento tecnologico chave por detras desta aplicação social, reside no facto de telefones móveis com potencialidades ao nivel da rede local, tais como Bluetooth, transmitem continuamente um código original da sua identificação (BTID) que é recebido por outros dispositivos. No log de proximidade o BlueAware regista o BTIDs+timestamp dos BTIDs encontrados. Se um dispositivo for detectado que não esteja gravado recentemente no registro da proximidade, a aplicação emite automaticamente o BTID descoberto sobre a rede de GPRS ao Serendipity server. O scannig continuo dos BTIDs pode consumir muito de uma bateria mais velha do telefone móvel em aproximadamente 18 horas. Consequentemente BlueAware foi modificado para fazer o refrsh do ambiente em cada cinco minutos, fornecendo ao menos 36 horas do tempo à espera.

- Porque existe a necessidade de utilizar "timestamps"?
Porque como é natural que um utilizador com um telemovel com bluetooth possa deixar a area e a sua informação deixe de ser relevante.

- O que é o BlueDar? Qual o seu modo de funcionamento?
O BlueDar é uma variação do BlueAware. O BlueDar foi desenvolvido para ser colocado em locais publicos, para enviar via rede wireless 802.11.b todos os BTIDs encontrados para o servidor serendipity

- O que é o BlueDating? Indique vantagens e desvantagens.
É uma aplicação desenvolvida também para fins de socialização (1-1). A arquitectura utilizada pelo blueDating não incorpora um servidor central, por isso, armazena no seu dispositivo toda a informação dos utilizadores nas suas proximidades (BTIDs+timestamps e perfil). Esta aplicação tem vantagem de guardar a informação no seu dispositivo garantindo privacidade e não ter de efecturar comunicações de longa distância (por exemplo, no caso do blueaware notificar o servidor central de novos BTIDs). Tem desvantagens de os perfis serem calculados de uma forma simplista e logo o match de interesses entre pessoas ser mais falivel do que no serenditipy, porque tem menor capacidade de processamento.

- O que entende por BTShare?
São aplicações P2P que correm nos dispositivos moveis (N-N), em que os dados são partilhados e trocados entre utilizadores, via bluetooth.
Estudar resto do paper pelos Acetados de 2006

- O que entende por Shared Spaces? E por Public Spaces?
São arquitecturas P2P que estimular interacção/discussão entre pessoas (N-N), através de um sistema pervasivo que se integra no "seu" ambiente.
São arquitecturas cliente-servidor que tem como objectivo projectar conteudos em espaços publicos, em ue os utilizadores podem utilizar diversas tecnologias (SMS, MMS, WiFi) para enviar mensagens que são mostradas no ecrã, estimulando assim a discussão e logo a socialização entre pessoas.

- O que entendo por IMMS (Intelligente Multimedia Messaging System)?
É um sistema que permite comunicação do tipo 1-N, ou seja, possibilita que uma pessoa via interface web ou uso de SMS deixe mensagens nos dispositivos de forma a notificar as pessoas (Ex: do professor e os alunos).

- O que é o Smart Blog?
É uma aplicação (1-∞) que permite:
- Publicação instântanea, via internet, de novos artigos em blogs *(texto e imagem)*.
- Interface permite a publicação de artigos de uma forma rápida e simples.

7 – Intelligent file hoarding

- O que nos fala este paper?
Este paper fala-nos sobre uma nova abordagem em computação ubiqua, que consiste no conceito de **hoarding**. E ao longo do paper é feita uma listagem de possiveis abordagens.

- Qual é o principal objectivo de hoarding?
Evitar o maximo possivel os casos de cache miss enquando estivermos a trabalhar desconectados da rede.

- Que tecnicas existem na actualidade para fazer hoarding de ficheiros? Como funcionam?

Não faz nada – só guarda os ficheiros que estão a ser utilizados no momento de desconexão.

Utilizador fornece informação – o utizador especifica que ficheiros e directorias quer fazer hoarding num hoarding profile. Tarefa complicada de realizar, requirindo alguma experiencia, ainda por cima em especificar os ficheiros que são necessários.

Snapshot Spying – nesta abordagem o utilizador utiliza bookends para delimitar periodo da actividade. Monitorizar ficheiros utilizados, mas com uma simples execução não revele todos os ficheiro utilizados
Word -> apresentacao1.doc
Word -> apresentacao2.doc

Distância Semantica – esta abordagem é utilizada pelo SEER, que analisa as sequencias de acessos aos ficheiros. Tem a incapacidade de separar o acesso a ficheiros feitos por processos executados simultaneamente.

- O que é um working set?
Os ficheiros para os quais nós queremos para poder trabalhar.

- Explique o funcionamento do mecanismo de Hoarding proposto no paper?
O mecanismo de Hoarding explicado no paper é chamado de **transparent analytical spying**, que consiste nas seguintes caracteristicas:

1) Detecta automaticamente working sets para aplicações e data
   > Monitoriza-se continuamente o acesso a ficheiros por parte do utilizador e registando esta informação num log
2) Providencia generalized bookends para delimitar períodos de actividade
   > Primeiro, detectamos e anotamos os programas individuais executados, durante o período de spying delimitado
   > generalizamos o conteúdo dos programas através de sessões múltiplas
3) Apresenta pacotes convenientes ao utilizador através de uma interface gráfica (GUI - Graphical User Interface)
4) Deixa o utilizador "load the brief case"

8 – Automated hoarding

- O que nos fala este paper?
Este paper fala-nos sobre um sistema de hoarding como o SEER e o seu modo de funionamento. No final do paper são feitas alguns testes. Como explicado no paper anterior o Hoarding é importante para na ausência de disponibilidade de rede, os utilizadores poderem trabalhar *offline*, sem os obrigar a interromper o seu trabalho, conseguindo assim trabalhar sem necessidade de estar ligado à rede. Portanto o hoarding efectua o *caching* de ficheiros remotos antes da desconexão do dispositivo e saber que ficheiros hão-de ser guardados localmente em *cache*.

- SEER é um sistema de hoarding de previsão automatico, em que consiste?
Este sistema é baseado na ideia de que o sistema pode observar o comportamento do utilizador, fazendo inferências sobre as relações semânticas entre os ficheiros.

- Como é composto o SEER?
O SEER consiste em 2 grandes componentes: **observer** e **correlator**.
O observer observa o comportamento do utilizador e todos os acessos, classificando cada acesso de acordo com um tipo, convertendo os pathnames para um formato absoluto e por ultimo transfere esses resultados para o correlator.
O correlator avalia as referencias para os ficheiros, calculando a distância semântica entre os vários ficheiros. Estas distâncias semânticas dirigem um algoritmo de clustering que atrbui cada ficheiro a um ou mais projectos.

- Indique as forma de medir a distância semântica.

Medir a distância semântica entre ficheiros é uma operação complexa, porque não é fácil arranjar uma quantificação que seja ao mesmo tempo compreensivel e implementavel. Sendo assim terão listadas os segiuntes metodos:

Definição 1: *Temporal semantic distance*
"The temporal semantic distance between two fire references is equal to the elapsed clock time between the references."

Vantagens

- Simples e fácil de medir;
- Ficheiros referenciados ao mesmo tempo tendem a ser semanticamente relacionados

Desvantagens

- Disparidade entre escala temporal humana e a do computador
- Proximidade temporal entre ficheiros dependente da carga do sistema

Definição 2: *Sequence-based semantic distance*
"The sequence-based semantic distance between two file references is equal to the number of intervening references to other files"

Vantagens

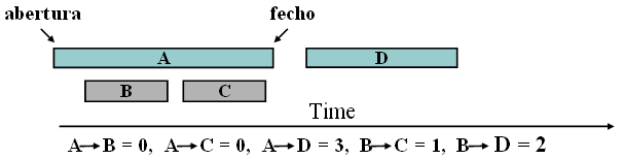
- Não depende de uma escala temporal;

Desvantagens

- Apenas tem em conta referencia a aberturas de ficheiros e não o tempo de vida dos mesmos

Definição 3: Lifetime *semantic distance*
"The lifetime semantic distance between an open of file A and an open of file B is defined as 0 if A has not been closed before B is opened and the number of intervening file opens (including the open of B) otherwise."

Já se tem em conta o tempo de vida de cada ficheiro



- Explicação do algoritmo de clustering

O algoritmo primeiro calcula os n visizinhos proximos para cada ponto, onde n é o parametro do algoritmo. Despois de 12 vizinhos proximos terem sido calculados, o algoritmo compara os vizinhos mais afastados listados para cada par de pontos. Se dois pontos tiverem mais do que n visinhos proximos em comum eles consideram que esses membros estao no mesmo cluster, e os seus cluster sao combinados.

From:	To: A	B	C	D	E	F	G
A		k_n	k_f				
B			k_n				
C				k_f			
D					k_n		
E							
F							k_n
G				k_n			

Table 2: Example relationships among seven files.

Primeira Fase **Segunda Fase**
{A,B,C} {D,E,F,G} {A,B,C,D} {D,E,F,G}

Portanto resumidamente o algoritmo realiza:
- comparação entre pares;
- agrupa os ficheiros em um ou mais projectos;
- faz os agrupamentos utilizando o critério de vizinhança;
- usa dois tipos de medidas: kn (near) e kf (far)

- Que problemas podem existir no Hoarding do SEER? Apresente exemplos.
Podem existir os problemas da seguinte ordem:

> **actividades irrelevantes**: são actividades que não contribuem como informação util para a realização do hoarding e no entanto têm de ser ignorados.

Exemplo: o utilizador usou o comando find numa determinada directoria. Esta operação tem de ser ignorada porque vai influenciar consideravelmente o tempo de referenciamento a ficheiros que não interessam prejudicando o calculo da distancia semantica entre ficheiros.

> **bibliotecas partilhadas**: a utilização das mesmas bibliotecas por determinados programas, podem causar relações incorrectas entre ficheiros. (Ex: 5 programas a utilizarem a biblioteca select.c).

> **ficheiros criticos**: podem haver ficheiros de boot que não são hoarded.

> **detectando hoard misses**: pode-se anotar de 0 a 4 um cache miss de um determinado ficheiro para no futuro ele poder ser incluido.

> **ficheiros e directorias temporarios**: devem ser ignorados

> **acessos simultaneos**: em sistemas multi-tasking, o SEER pode inferir relações entre ficheiros acedidos por diferentes processos incorrectamente.

9 – Obiwan

- O que nos fala este paper?
Este paper fala-nos sobre a ferramenta de Midleware OBIWAN, que permite facilitar a vida ao programador. Esta ferramenta permite ao utilizador focar-se na lógica da aplicação. São apresentadas as suas caracteristicas e componentes (por exemplo, garbage collector distribuido).

- O que é um agente móvel?
É uma composição de software e informação que tem a capacidade de se mover autonomamente de um computador para outro, e continuar a sua execução no computador de destino.

- Com que problemas é que os programadores se deparam? Soluções
Os programadores têm de ser preocupar com: replicação de objectos; uso abusivo de recursos por parte de agentes móveis; garbage collection distribuido; falta de flexibilidade na escolha do paradigma a utilizar.
Para lidar com estes problemas existe o OBIWAN.

- Explique as caracteristicas do OBIWAN
O OBIWAN tem as seguintes caracteristicas que se seguem:

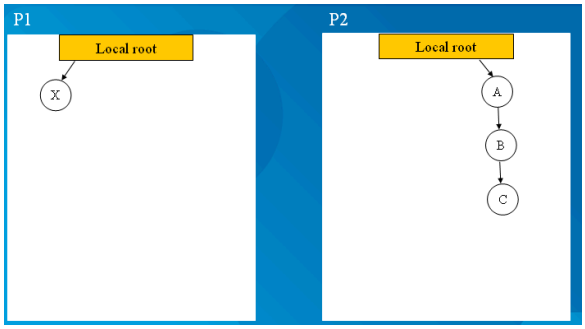
> **Flexibilidade na escolha do paradigma**
Permite aos programadores escolher entre RMI, replicação de objectos ou agentes móveis

> **Replicação Incremental**
Gestão de memória distribuída capaz de lidar com réplicas de objectos automaticamente

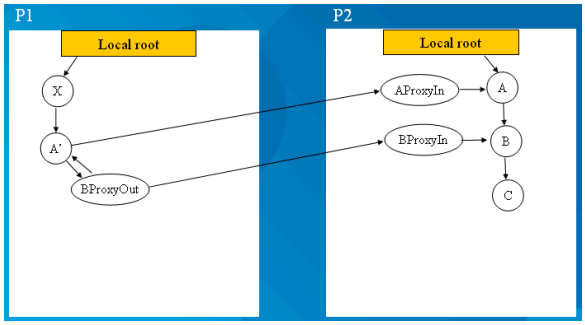
> Garbage Collection distribuído
Recolha automática de réplicas inutilizadas

> **Políticas de Segurança de Agentes Móveis**
Definição e cumprimento de políticas de segurança bem adaptadas às necessidades do agente

- Explicação do funcionamento da replicação incremental com o exemplo do paper

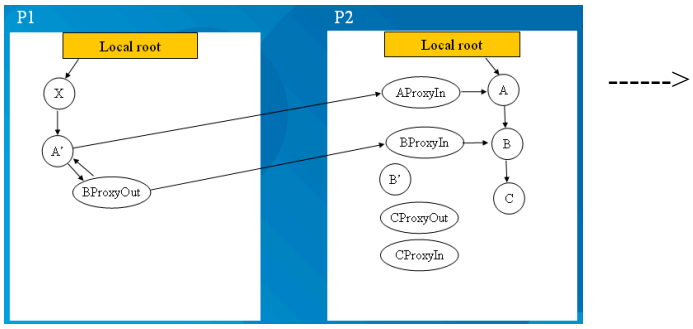


Inicialmente existem 2 processos (P1 e P2), que se encontram nas seguintes situações:
P1 – tem um grafo com o objecto X
P2 – tem um grafo com os objectos A, B e C

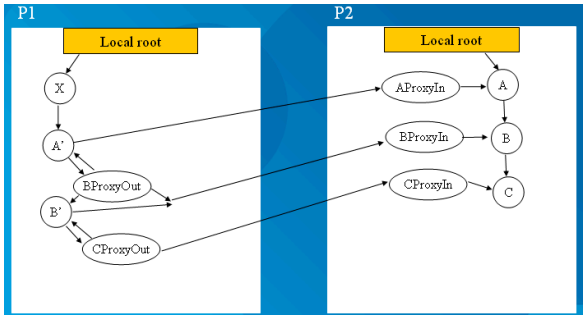


O objecto A é replicado do P2 para o P1, sendo o Objecto A' que está no processo P1
A' mantém uma referência para o objecto AproxyIn

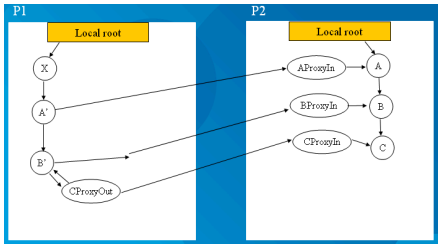
Como o objecto B ainda não foi replicado, então A' aponta para BProxyOut



------>

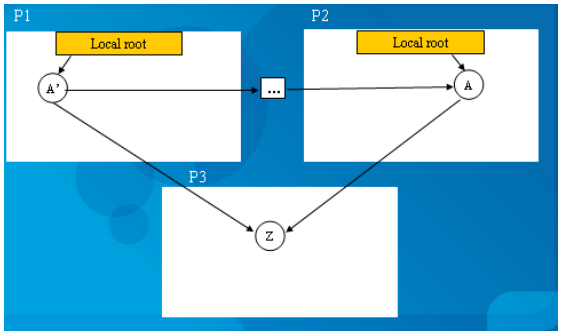


Seguindo a mesma lógica, o objecto B também é criado da mesma fora

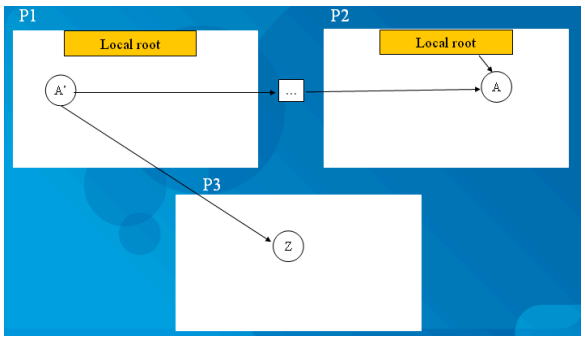


Ficando com uma referencia para o BproxyIn e com uma referencia para o Objecto CproxyOut do Objecto C que ainda não foi criado

- Explicação do Garbage colection distribuido



Consideremos que devido à execução de uma aplicação que o objecto A' fica inacessivel no P1 e que também no P2 A deixa de referenciar Z. Agora a grande questão é: deverá Z ser considerado lixo?



Só porque A' não é localmente acessivel e A não apontar para Z, não quer dizer que Z tenha de ser necessariamente lixo. Existe uma regra para consideração de um objecto como lixo: "Um objecto é considerado lixo se não for alcançavel por nenhuma das replicas do objecto de fonte que o referir" – regra da união (neste caso o A).

10 – Session Guarantees for Weakly Consistent Replicated Data

Ver os acetatos de 2006

11 – Data Staging on Untrusted Surrogates

- O que fala este paper?

Este paper fala sobre a segurança dos dados do cliente em surrogates. Os surrogates podem ser corruptos e mesmo assim o mecanismo de datastaging pode garantir a segurança dos dados do cliente. Por isso ao longo deste papel é explicado com maior detalhe em relação ao que se falou no Aura.

- O que faz o ClientProxy?

O ClientProxy é responsavel por interceptar trafego, se verificar-se grande latência na rede, ele encontra um surrogate que esteja presente na rede, com capacidade extra de armazenamento. O ClientProxy regista-se no surrogate e faz o staging dos ficheiros que utiliza mais para o surrogate.

Portanto o ClientProxy tem 3 tarefas a desempenhar:

1) Redireccionar pedidos ficheiros para o surrogate
2) Controlar que ficheiros são staged
3) Preservar consistência

Nota: O ClientProxy regista-se sempre no surrogate onde partilha uma chave de sessão.

- O que faz o Data Pump?

Proporciona o staging. O data pump situa-se normalmente num desktop perto do sistema de ficheiros. Quando um cliente desejar efectuar o staging de um ficheiro, ele envia uma mensagem para o pump via um canal seguro, em que nessa mensagem vai a informação do pathname do ficheiro e o IP do surrogate.O data pump obtem o ficheiro do sistema de ficheiros, gera uma chave simetrica, cifra o ficheiro e gera um hash do ficheiro. Depois coloca o ficheiro no surrogate e se teve sucesso então envia chave e o hash para para o ClientProxy, que guarda essa informação.

- Este modelo é seguro?

Não, porque o surrogate pode ser corrupto, e apesar de os ficheiros estarem cifrados e só o cliente possuir a chave, o surrogate pode usar metódo de brute-force para tentar determinar o seu conteudo. A solução para este caso seria um cliente descontente negar ao seu ClienteProxy o contacto com esse surrogate.

- Como sabe que o cliente que o ficheiro que obteu do surrogate não foi alterado?

Depois de decifrar esse ficheiro com a chave recebida pelo data pump e realizar o hash, se o hash estiver igual.

12 – Dynamic Binding in Mobile Applications

- O que nos fala este paper?

Este paper começa por falar-nos na grande complexidade que existe no desenvolvimento e integração de aplicações móveis em diferentes redes de acesso à Internet. Explicam que as soluções actuais do midleware não servem os interesses dos clientes móveis devido à frequente perda de conectividade à Internet. São portanto apresentados uma serie de desafios. Prentende-se portanto garantir a transparência na utilização de aplicações móveis, permitindo a sua adaptação a ambientes dinâmicos. Para tal é discutido o projecto Colomba ao longo do paper, e as várias politicas de biding que podem ser tomadas.

- Enumere alguns desafios que este paper refere

> O que fazer quando perder conectividade?
> Como a aplicação se comporta face à mudança de local?
> Como restabelecer a ligação aos recursos previamente estabelecidos?
> Se pudermos desenvolver uma aplicação sem nos preocuparmos com estes detalhes, teremos um software genérico.

>Terá que ser capaz de funcionar independentemente da forma de acesso à Internet e adaptada ao contexto local.

- O que é o Colomba (Context and Location-based Middleware for Binding Adaptation)?
É uma abordagem de Middleware que permite ao cliente movel o update automatico das referencias para os recursos que são precisos sempre que o cliente se mova, e seleccione dinamicamente as estrategias de binding mais adequadas. Portanto permite separar claramente a lógica aplicacional do processo de associação.

- Que estrategias são necessárias?
> Resource Movement
> Copy Movement
> Remote Reference
> Rebinding

- Que serviços de descoberta de serviços podem ser utilizados?
Podem ser uPNP, Bluetooth ou Jini

- O Colomba é location e context aware, quais as vantagens?
Consciência de Contexto: estar consciente do ambiente de utilização da aplicação. Logo, é possível discriminar os recursos que interessam aos utilizadores e seleccionar aqueles que são mais indicados para os dispositivos

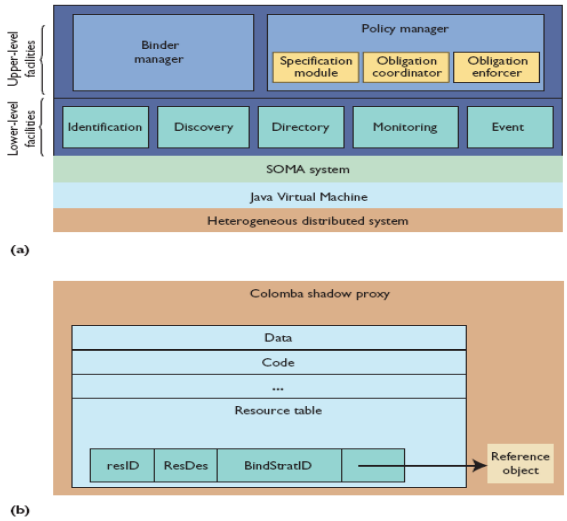
Consciência da Localização: estar consciente da localização física do dispositivo, ou do utilizador dentro de um espaço ou localidade servida por uma rede, quando conectados a ela – GeoIP, por exemplo.

- O que quer dizer MNS?
Mobile news service, ou seja, um serviço de informações.
- Definição para users e resouces
Users – são os que requisitam os serviços, dentro de um contexto heterogéneo (device) e móvel (location).

Resources – são a referência para a disponibilização de dados pelos serviços.

- Explique o são metadados
>Fonte de informação que permite caracterizar o utilizador e modelar o comportamento da aplicação.
>São utilizados em ambiente *runtime*, logo, a redefinição do comportamento é possível sem ter que reinicializar a aplicação.

- Explicação da Arquitectura do Colomba



O sistema mune cada dispositivo móvel com uma proxy (shadow proxy).

- Shadow proxies negoceiam serviços/ recursos de acordo com as necessidades do cliente (dispositivo ou utilizador).
- Mecanismo implementado através do uso de agentes móveis.
- Garante mobilidade , autonomia e assincronidade.
- O acesso das shadow proxies aos recursos é gerido pelo *binder manager.*
-

Binder Manager (BM) - Gere o acesso a recursos das shadow proxies. Ajusta dinâmicamente a estratégia de *bind* de acordo com a política especificada e ainda suporta as 4 estratégias de *binding* atrás referidas.

Policy Manager - Providencia informação para o *binder manager* poder ajustar dinâmicamente a estratégia de *binding* a usar.

Composto por 3 módulos:

- Specification module

  Explora ferramentas que permitem edição e actualização de políticas.

  Providencia maneiras de transformar políticas de alto nível em instruções de baixo nível interpretadas pela plataforma middleware.

  Todas as políticas armazenadas numa directoria específica.

- Obligation coordinator

  Recupera políticas e interpreta-as para analisar toda a informação relevante.

  Regista os eventos significativos e a informação relevante aos mesmos.

  Envia essa informação sobre os eventos às proxies "interessadas".

- Obligation enforcer

  Uma proxy notificada pelo *Obligation coordinator* delega o *Obligation Enforcer* para interpretar as alterações pedidas , extrair as acções necessárias e comandar o *binder manager* a proceder às alterações.

- Como funcionam as ShadowProxys?

> *Shadow proxy* é criada (para cumprir o *fetching* de recursos para um dispositivo / aplicação).
> Como não têm nenhum acesso directo a recursos , apenas referenciam o *binder manager*.
> Em resposta ao primeiro pedido , é-lhe dada um *resource descriptor* (modo de referenciar o recurso pedido).
> Quando a proxy é instanciada num novo local , o *B.M.* associa-a com uma tabela de recursos que contém todos os recursos disponíveis para a *proxy*.
> Verificada a validade da estratégia de *binding.*
> Caso a estratégia de *bind* tenha mudado , o *binder manager* instancia uma nova referência do objecto.
> Apenas com a chegada de um proxy a um novo local é que a estratégia de *binding* pode ser alterada.

- Explicação de 4 politicas programadas em Colomba

```
class MNSProxy extends ShadowProxy {
...
void init() {
... DataResource resourceID =
     COLOMBA.DataResource.get ("newspaper");
... )
void run() {
... for (;;) {
      if (isConnected==true) results =
         resourceID.query(search);
              visualizer(results);
              sleep(pollingInterval);
      } ... }
... }
```

Política de associação a recursos

Cada shadow proxy executa o metodo init() quando é criado para inicializar a referência para a informação do recurso chamado Newspaper. COLOMBA.DaraResouce.get() invoca o BM que adiciona uma nova entrada na tabela de recursos do ProxyID, inserindo o tipo de binding como default, referência remota e retorna o descriptor do resourceID para o Shadow Proxy aceder ao recurso. Enquanto o MT estiuver conectado à rede fixa (IsConnect), o ProxyID repete uma query especificada pelo utilizador e em que intervalos de tempo.

```
inst oblig Pol1 {
on DisconnectRequest (proxyID);
subject s = ProxyID;
target t1 = Binder, t2 = ProxyID;
do t2.showWaitingMessage() ||
     t1.setIsConnected(ProxyID,false) ->
     t1.setAgentbindingType(ProxyID,"resource
        movement") ->
     t1.updateReferenceObject(ProxyID) ->
     t2.removeWaitingMessage();
when MonitoringSystem.getFreeDiskSpace
    (ProxyID.getLocation())
     > ClientID.resourceSpaceSize();
}
```

Política de desconecção de recursos

Sem modificar a implementação do MNS Proxy, podemos adoptar o MNS para trabalhar em diferente cenários. Um será permitir as Shadow Proxys aceder a recursos necessários, mesmo em disconecção, ao especificar simplesmente a Pol1 como politica a usar. Quando o ProxyID necessitar de disconectar (DisconnectRequest_event), Pol1 comanda o BM (named Binder) para mover recursos necessários para o dispositivo Proxy e se o dispositivo tiver espaço suficiente em disco para os hospedar. O prototipo MNS necessita que o MT não esteja desconectado antes de completar a fase do movimento de recurso. Depois, o BM actualiza IsConnected a false e o tipo de binding como movement resource para cada recursos referenciado na tabela de recursos do ProxyID.

O metodo updateRerenceObject() do BM verifica o tipo de estrategia de binding para cada entrada na tabela de recursos do proxy e faz update às referencias dos

objectos devidamente. Se o recursos newpaper R1 se mover, por exemplo, o serviço de eventos notifica o resourceMovament para que o Shadow Proxy dispare o trigger para se realizar a Pol2.

```
inst oblig Pol2 {
on ResourceMovement(ProxyID,R1);
subject s = ProxyID;
target t = Binder;
do t.setAgentBindingType(ProxyID,R2,"copy
movement") ||
     t.updateReferenceObject(ProxyID,R2);
}
```

Política de reutilização de recursos

Na Pol2 o Shadow Proxy comanda o BM para mover uma cópia da informação do newpaper do resource R2 que está na mesma localização para onde o R1 migrou. Se o perfil de R2 especificar que R2 pode ser copiado e transferido, O BM adiciona uma nova linha para R2 na tabela de recursos do ProxyID (o metodo setAgentBindingType()) na Pol2 e força um copy movement ao fazer updates aos objectos referenciados (o metodo updateReferenceObject()).

```
inst oblig Pol3 {
on domainArrival(ProxyID,newNode);
subject s = ProxyID;
target t = Binder;
do t.setAgentBindingType(ProxyID,"rebinding") ||
     t.setIsConnected(ProxyID,true);
when MonitoringSystem.isDiscoveryAlive() == true;
}
```

Política de actualização de contexto

O Colomba facilita o desenvolvimento de contexto e localização ao especificar a Pol3. Quando o ProxyID entra numa nova localidade (DomainArrivalEvent), Pol3 força o BM para modificar tantp o tipo de estrategia de binding para rebinding de cada recurso MNS na tabela de recursos do ProxyID e IsConnected a true. No exemplo, colomba fornece à Pol3 accções que apenas o colomba discovery está a trabalhar no novo MT locality.

13 – Transactions Policies for Mobile Networks

- O que nos fala este paper?

Este paper fala-nos sobre o facto de as soluções transaccionais existentes não serem eficientes, levando a "aborts" desnecessários ou disponibilidade de dados reduzida. Portanto não existe flexibilide de adaptação a um largo conjunto de cenários (Ex: diferentes aplicações ou diferentes topologias de rede).

A solução abordada nesta paper, consiste no **MobileTrans**, onde é explicado como os programados podem configurar as suas politicas transaccionais de forma a modelar comportamentos das suas tranascções às suas preferencias. Desta forma as transcções podem adaptar-se em runtime aos problemas que possam surgir.

- O que é o MobileTrans? Vantagens

É um sistema transaccional baseado em objectos para redes móveis, que suporta a definição de politicas transaccionais. Os programadores podem configurar diversos aspectos relaticos ao comportamento das transacções, de forma simples.

Tem como vantagens de: as transacções comportarem-se como nós definimos; as transacções podem ser alteradas em runtime; possibilidade de adaptação a diversos cenários.

- Quais são as propriedades das transacções que o utilizador pode programar?

A configuração de políticas é feita declarativamente (ficheiro XML)/ programaticamente (Java ou C#), apenas configurando um conjunto de atributos que vão determinar o comportamento da transacção.

Consistency – Consiste em usar versões desactualizadas do objecto, se o original não estiver disponível

Fetching – Obter objectos necessários à transacção antes ou em tempo de execução

Delegation - Delegar a outro nó a responsabilidade de finalizar a transacção

Atomicity - Finalizar transacção sem todos os nós saberem isso (não estão disponíveis, não se interessam com isso)

Caching - Guardar os objectos adquiridos de nós remotos e os objectos alterados em transacções locais

Failure Handling - Determinar como reagir perante a não verificação de condições definidas previamente

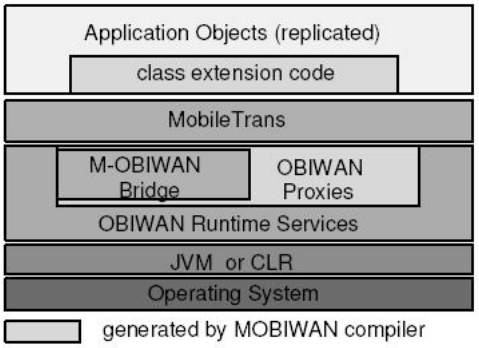
Attributes			
Name		**Value**	**Arguments**
consistency	.object	*required*	–
		replica	–
		dispensable	–
	.degree	*high*	–
		medium	–
		low	–
fetching	.object	*random*	depth
		node	depth, node
		randset	depth,{node}
	.mode	*ondemand*	–
		prefetch	{obj}
delegation	.coordinator	*random*	–
		node	node
		randset	{node}
	.responsibility	*local*	–
		foreign	–
atomicity	.object	*mandatory*	–
		tentative	–
	.degree	*high*	–
		low	–
caching	.read	*yes*	–
		no	–
	.write	*yes*	–
		no	–
failure	.consistency .fetching .delegation .atomicity .user.*	*abort*	–
		retry	attribute
		suspend .*timeout*	time, attribute
		suspend .*reshape*	attribute
		suspend .*user*	attribute

- Explicação da arquitectura do MobileTrans

O MobileTrans funciona sobre o **Mobiwan** que é uma extensão do Obiwan, para suportar as necessidades transaccionais do MobileTrans.

A arquitectura é composta por 5 partes:

- Runtime services – registo de objectos, serviço de nomes, descoberta e conexão a repositórios de objectos…
- Proxies – proxy-out/ proxy-in (comunicação entre nodes)
- Mobile-device bridges – web-services configurados em cada node, que são invocados pelos proxy-out.
- Open-compiler (obicomp) – geração de código para as classes proxy para a criação de réplicas de objectos.
- Class extension code



O controlo do acesso concorrente aos dados é baseado no algoritmo MVPV (multiversion parallel validation algorithm).

- composto por 2 fases:
  - Fetch phase – corresponde à fase leitura do MVPV
  - Commit phase – fases de validação e escrita do MVPV
- Distribuicão do algoritmo baseada no protocolo 2PC.

14 – Service Location Protocol

- O que nos fala este paper?
Este paper fala-nos de uma arquitectura para descoberta de serviços, que utiliza o protocolo IP. Chama-se **service location protocol (SLP)**, e todos os seus componentes (DirectoryAgent, UserAgent e ServiceAgent) são explicados e também as várias formas de funcionamento.

- Explique em que consiste cada um dos componentes deste protocolo
UserAgent (UA) – efectua a pesquisa dos serviços para o cliente
ServiceAgent (SA) – anuncia a localização e atributos dos serviços
DirectoryAgent (DA) – armazena as informações dos serviços

- Mencione as caracteristicas do service location protocol (SLP)
- Framework para descoberta automática de recursos em redes IP.
- Usa Multicast e DHCP para inicializar a framework
- É escalável desde pequenas LAN's até grandes redes
- É específico para Intranet
- Descobre servidores realmente disponíveis
- Acrescenta procura por características

- Porque é que o protocolo de descoberta de serviços (SLP) não é suportado na Internet?
Porque a infraestrutura da Internet não permite multicast.

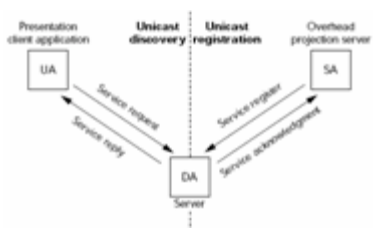
- Este Protocolo é mais rápico com ou sem DAs?
É mais rápido com DA's.

- Que informação contêm os serviços quando anunciados?
Um endereço URL do serviços, ou seja, o seu endereço IP, porto e o tipo de serviço descrito num determinado template.

- Que informação é transportada nos DA's advertisements?
A localização do proprio DA (endereço URL).

- Como funciona a arquitectura com e sem DAs?
A arquitectura tem a seguinte forma para cada um dos casos e tem o seguinte funcionamento:

Com DAs:

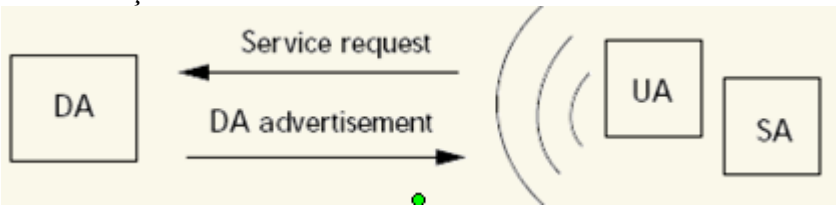
1) 

O SA regista o serviço no DA (unicast)
2) O UA pede a localização do serviço ao DA (unicast)

Sem DAs:

1) O UA pede o serviço usando *multicast*
2) O SA reponde usando *unicast*

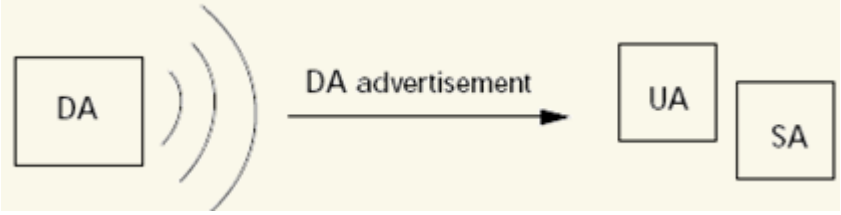
- Que tipos de localização dos DAs existem? Explique-os.
Existem 3 formas de localização dos DAs:
- Utilização de DHCP por parte dos UAs e os SAs
- Localização activa



1) O UA ou SA enviam um pedido de serviço em *multicast*
2) Um ou mais DAs respondem em *unicast* para o UA ou SA
3) O UA ou SA guarda a localização dos DAs que responderam
4)

Problemas
Se existirem muitos DAs um agente será **inundado** de respostas por parte dos DAs **sempre** que efectuar um pedido de serviço. Dai a importancia do algoritmo de convergencia multicast SLP.

- Localização passiva

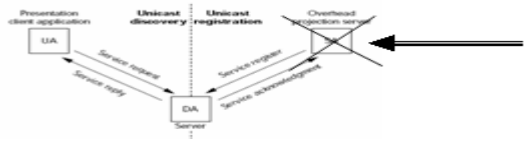


1) Os DAs enviam em *multicast* anúncios dos seus serviços
2) Os DAs continuam a enviar os anúncios repetidamente para que os novos UAs ou SAs os recebam

- Como funciona o algoritmo de convergência multicast de SLP?
Como o objectivo é determinar os DA's para o UA em causa, o UA vai realizando multicast (mensagem service request), e vai guardando a lista de todos DA's que respondam, e sempre voltar a efectuar multicast envia a informação dessa lista para os DA's que já responderam saberem que têm de se calar. No final ninguem responde é porque se tem uma lista com todos os DA's.

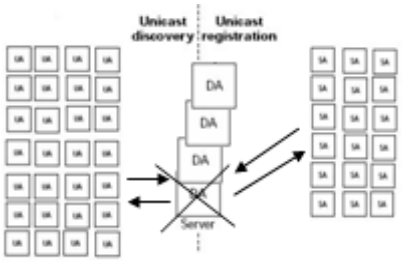
- Inumere os possiveis problemas que podem surgir
Os possiveis problemas que podem surgir são os seguintes:

- Falha no SA(desactualização de registos do DA)



Solução: tempo de vida dos registos dos SAs

- Falha no DA (ponto centralizado de falha).



Solução: existirem mais DAs para tolerância a faltas

- Como funcionam os Scopes?
O Scope é um mecanismo para aumentar a escalabilidade do SLPs. Um scope é uma string usada para agrupar recursos pela sua localização, rede ou categoria administrativa. Por deito os SA's e os UA's estao configurados com uma scope de "default". Assim os UA's só podem localizar os serviços nos scopes aos quais têm acesso. Por exemplo, um scope public e outro privado.

- A nivel de segurança existem preocupações a ter com os SA's?
Sim porque agente podemos ter assinaturas digitasis de que realmente comprovem que é o SA que eu quero contactar, mas nada me garante que o serviço faz aquilo que eu espero.

15 – An Architecture for Secure Service Discovery

- Do que fala este paper?

Este paper fala de uma arquitectura e implementação de um serviço seguro de descoberta de serviços (SDS). Os fornecedores dos serviços utilizam o SDS para avisar com complexas descrições dos serviços que se encontram a correr, enquanto os clientes usam o SDS para compor complexas querys para localizar esses serviços.

As descrições e queries destes serviços utilizam uma linguagem descritiva como o XML (Extensible Markup Language) para codificar caracteristicas importantes como custo, performance, localização, dispositivo ou serviço especifico.

O SDS fornce alta disponibilidade, tolerância a faltas, serviço incremental escalavel para localização de serviços na wide-area (nas emediações).

A segurança é um componente do nucleo do SDS, onde é necessário existir comunicações cifradas e com autenticação. Alem disso, o SDS usa uma lista de controlos de acesso hibrido e um sistema com capacidade de controlar os acessos à informação dos serviços.

- O que é o SDS?

O SDS é um serviço para procura de serviços. Este servidor tem um repositorio de informação sobre os serviços que estão presentes na rede, assim os clientes conseguem uma melhor eficacia na procura por serviços. Este servidor guarda dois tipos de informação:

1) descrições de serviços que estão disponiveis para execução nos recursos computacionais da rede
2) serviços que já estão em execução numa determinada localização

A descrição desses serviços e as querys são construidas em XML. Como explicado anteriormente.

- O que se quer dizer com Announcement-based Information Dissemination?
É um metodo que SDS tem de determinar falhas nos servidores (com serviços). Utiliza portanto uma abordagem de multicast periodicos. Como a informação pode ser alterada a qualquer momento, a cache revela-se como a melhor maneira e fiavel de guardar os dados.

- O que se quer dizer com Hierarchical Organization?
Como se tinha referido, o SDS é um mecanismo escalavel, em que os servidores SDS estão organizador segundo uma hierarquia. Isto resulta que um determinado SDS pertence a uma parte da rede (sub-rede), que podemos chama-lhe como um dominio, no qual só são redireccionados determinadas queries relacionado com esse servidor SDS nesse dominio. No caso de um servidor SDS ficar sobrecarregado, é criado um novo servidor SDS que sera criado como uma "child" e atribuida uma parte da rede, ficando com parte da carga.

- Que mecanismo de segurança são utilizados nesta arquitectura?
Uma abordagem seria cifrar toda a informação que é trocada entre entidades (entre clientes e servidores SDS e entre serviços e servidores SDS), mas num sistema desta natureza implicaria um overhead muito grande na operação de cifra.
Uma melhor abordagem seria cada componente ter um nome e um certificado de chave publica associado, por forma a autenticar-se peranto os restantes componentes.

- Explicação da arquitectura SDS

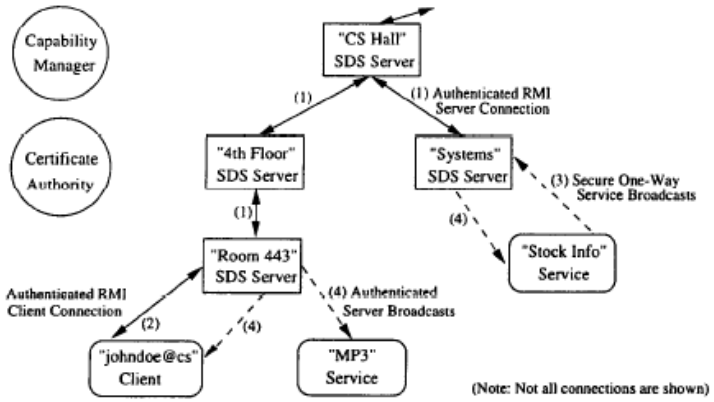


Figure 1: Components of the secure Service Discovery Service. Dashed lines correspond to periodic multicast communication between components, while solid lines correspond to one-time Java RMI connections.

Esta arquitectura como é ilustrado na imagem é composta por 5 componentes:
- Servidores SDS
- Serviços
- Gestores de capacidade
- Autoridades de certificação
- Clientes

- Como funciona o Gestor de Capacidades?
O SDS usa capacidades como mecanismo de controlo de acesso aos serviços, controlando os clientes que estão autorizados a aceder a um determinado serviço e descobrir a sua existência. Todos os serviços contactam o Gestor de Capacidades e depois de autenticação, especificam a lista de controlos de acesso (a lista dos clientes que podem aceder à descrição do serviço). O Gestor de Capacidades gera então as capacidades apropriadas e guarda-as para depois distribuir pelos clientes.

- Como funciona o AC (Autoridade de Certificação)?

O AC contem os certificados dos serviços, se um cliente quer contactar um determinado serviço, ele primeiro especifica o certificado de serviço e depois o AC devolve-lhe. De notar que não há necessidade de autenticação nesta interacção entre cliente-AC nesta accção porque ou o cliente tem a chave privada ou não, de pouco lhe serve.

- Importância dos sumários?

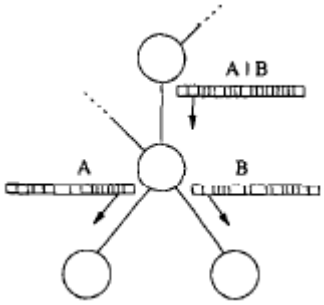
São uma sintese da informação dos serviços de nivel superior. Importante visto que quando surge um problema de escala não é viável cada SDS ter informação sobre todos os serviços.

- Em que consiste o conceito de Loss Agregation?

Para preveninr os servidores de níveis superiores da hierarquia de serem sobrecarregados com updates e trafico de querys, recorre-se a metodos de filtragem, para não se propagarem pela hierarquia. É ai que entra o papel do Loss Agregation, que é um processo de filtragem.

- Em que consistem os filtros de Bloom?

Seguindo uma abordagem de Loss aggregation e de query routing, esta abordagem é baseada no uso de hashing e hasing summarization via filtros de Bloom.



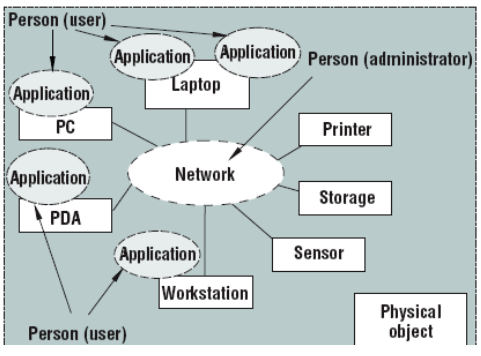
Os filtros de Bloom é um metodo de sumarização que usa um vector de bits, com bits colocados em posições correspondentes a um determinado valor de hash da informação. A propriedade chave dos filtros de Bloom é que fornecem uma sumarização hashed da informação, e colapsa esta informação numa tabela de tamanho fixo, em que existe uma probabilidade muito baixa de existirem falsos indexes. Esta abordagem nunca causa false misses.

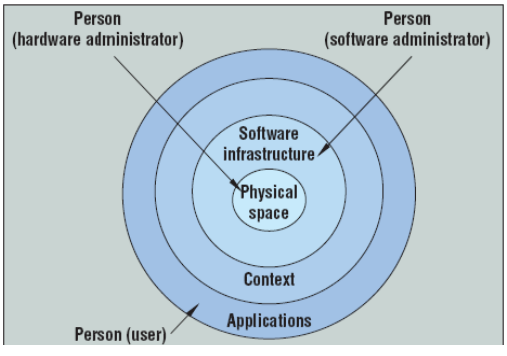
.

16 - A Middleware Infrastructure for Active Spaces

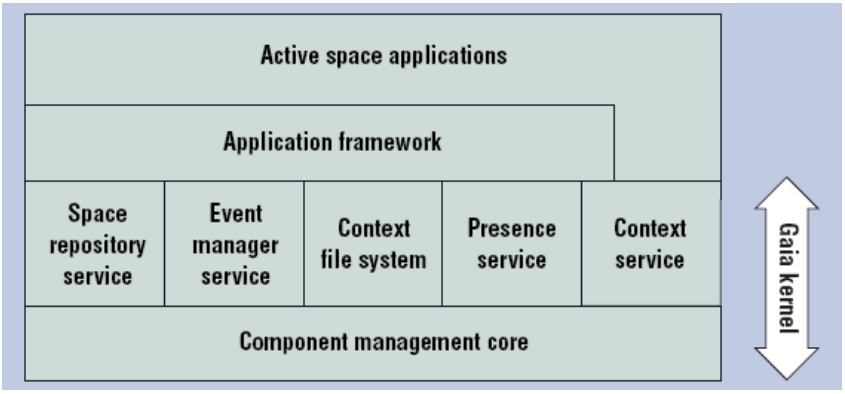
Gaia é um meta-sistema operativo,na medida em que corre por cima de todos os SO dos dispositivos presentes no active space (por exemplo, desktops, portateis, monitores com touchscreen, PDAs ou mesmo impressoras). O gaia distingue-se dos outros SO tradicionais pelo seu suporte a uma infraestrutura distribuida de middleware que permite a coordenação do software com os dispositivos na rede. Sendo uma das suas principais vantagens, a utilização de uma framework que permite ao utilizador a fácil programação de active spaces.

Um **fisical space** é caracterizado uma zona limitada, constituida por vários dispositivos heterogenios (ex: uns correm Windows, outros linux ou mesmo macintoch). Um **active space** é um fisical space mas com coordenação baseada em software com utilização de contexto.





O SO gaia tem uma determinada arquitectura, que é composta como mostra a seguinte figura:



Esta arquitectura pode ser dividida em 3 partes: **Gaia Kernel** que serve de suporte às aplicações, como vamos mostrar mais à frente, mas basicamente é responsavel por transmitir informações uteis no active space aos utilizadores; **Aplication framework**, que como o nome indica, vao ser as bibliotecas necessarias para a programação dos actives spaces. **Aplicações**, são no fundo os programas interactivos que vão correr nos dispositivos do Gaia de forma ubiqua.

O gaia kernel é composto por:

Componente management core (CMC) – esta camada vai ser responsável pela gestão dos componentes do Gaia (como serviço repositorio, gestão eventos, sistema de ficheiros de contexto, presença e contexto), como criação, destruição, actualização. Este componente é importante na medida em que o Gaia tem de suportar remotamente a gestão destes dispositivos no active space. Estes componentes devem ser uma unidade de sofware que estão a correr nos dispositivos.

Nota: O gaia utiliza uma comunicação baseada em CORBA, ou seja, comunicação por objectos remotos, mas também podia utilizar RMI.

Conjunto de serviços

- Event Manager – este serviço vai ser responsável pela gestão das comunicações entre os componentes. Para implementar esta comunicação, utiliza-se o modelo *canal*, *produtor* e *consumidor*. Os canais são utilizados como meio de comunicação, para os fornecedores comunicarem com os consumidores. Cada canal é utilizado para um evento distinto, no qual os componentes se registam, para serem comunicados. Um bom exemplo prático, seria um utilizador entrar no active space, e o serviço de presença detectar este, e enviar um evento para um determinado canal, de forma a notificar todo o sistema que esteja registado no canal.

Podem exitir várias fabricas de eventos, para uma melhor separação de tipos de eventos, em que uma fabrica, consiste num conjunto de canais. Como por exemplo, podemos ter uma fabrica de eventos para presença e outra fabrica para outro tipo.

O event Manager, tem portanto as seguintes funcionalidades: criar um tipo de canal (fabrica de evento); criar um um canal de eventos; etc...

- **Context service** – Este serviço tem como objectivo criar contextos de informação, para poderem ser utilizados pelas aplicações, de forma a criar uma interação mais dinâmica. A criação de um contexto pode ter como base o tempo, a localização do utilizador, a sua actividade (o que faz no momento) ou mesmo os mesmo uma actividade de grupo.

Para se recolher informação, de modo a criar os contextos, utilizam-se *context providers*, para fornecer contexto às aplicações. Estes context providers são sensores.

A linguagem utilizada para definir contexto, tem como base a seguinte estrutura:

Context(<ContextType>,<Suject>,<Relater>,<Object>)

O contextType é referente a um canal de eventos

O subject é a entidade com que o contexto diz respeito (ex: pessoa, lugar)

O relater tem a função de relacionar o Sujeito com o Objecto

Exemplos:

Context(temperatura,salaN1.1,is,98F)

Context(numero pessoas,salaN1.2, >,4)

Nota: semilhante à linguagen logica *Prolog*

Nota: podemos determinar informação de um contexto de nivel superior, com base em informação de nivel inferior, como o ToolKit Context agregation.

- **Presence service** – Este serviço, tal como o nome indica, tem a responsabilidade de determinar presenças de entidades no active space. Estas entidades podem ser pessoas, dispositivos ou aplicações. Como se pode ver temos de utilizar 2 tipos de localização, uma para as **entidades digitais** (aplicações) e outra para as **entidades fisicas** (pessoas e dispositivos).

A localização digital é feita com base em heartbeats periodicos, que nos permitem determinar se a aplicação esta activa. No caso desta entidade estar inactiva, o serviços de presença notifica o resto do active space, por um canal de comunicação. Bom para tolerância a faltas e aumento de disponibilidade.

A localização fisica é realizada com base em localizadores (tipo person trackers).

Nota: O sistema de detecção de presenças fisicas, actua como um proxy, porque procura proativamento por entidades.

Nota: O Gaia suporta a utilização de vários tipos de sensores (com drivers diferentes e algoritmos diferentes).

- **Space repository** – todas as entidades presentes no active space precisam de saber os seus recursos. Para tal existe o serviços de space repository, onde são guardadas as informações (nome, tipo e utilizador) de todos o software e entidades de hardware que estão no space. As aplicações acedem a este repositorio para obter informações (Por exemplo de um altifalante ou de um touchscreen).

A informação no space repository é descrita em XML.

Exemplo prático: Quando um recurso entra no active space (por exemplo um PDA), o space repository contacta este para obter a sua descrição em XML.

Nota: O space repository permite a independencia das aplicações em active spaces diferentes, já que podem ser incorporadas no active space como recurso, sem problemas.

- Context file system – neste sistema de ficheiros, as directorias estão associadas a um contexto. A um determinado caminho pode corresponder a um valor ou um tipo de contexto. Este tipo de estruturação do sistema de ficheiros, permite uma maior organização e perquisa de informação.

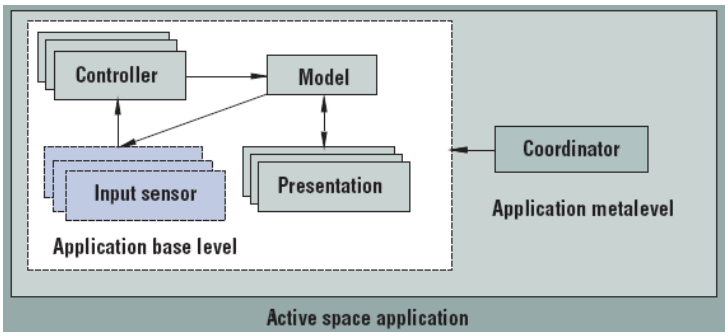
Exemplo: O joão entra no active space com o contexto "seminario" a sua aplicação carrega o ppt dos seminarios, mas se noutro dia ele entrar com outro contexto "futebol", a sua aplicação já vai carregar a sua informação para este contexto.

Portanto esta utilização de contexto conjuntamente com o context file system permite de forma proactiva, carregar automaticamente a informação do utilizador.

Como é que a aplicação sabe informação a ser utilizar? O CFS combina as propriedades do ambiente (localização, tempo, situação,...) com as propriedades especificas do utilizador.

Importantissimo: A arquitectura CFS é composta por servidor de mount e servidores de ficheiros. O servidor mount é responsavel por manter o namespace do active space. O namespace identifica a "zona" onde o cliente está. Quando um cliente muda de namespace, ou seja de active space, o servidor de ficheiros exporta a informação desse utilizador, para o seu corrente espaço local.

Aplication Framework

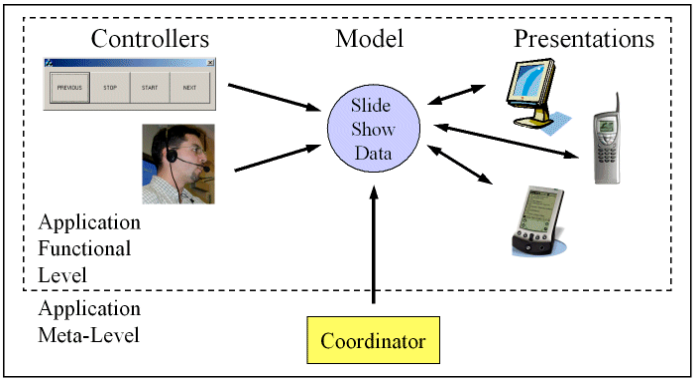


Com esta framework, as aplicações permitem aos utilizadores construirem ou adaptarem aplicações existentes aos active spaces.

Esta framework consiste numa:

- **infraestrutura para as aplicações**
- **mecanismo de mapping**
- **grupo de politicas**

- infraestrutura para as aplicações – Esta infraestrutura tem como modelo base o MVC (model+view+controller). O seguinte exemplo ilustra melhor:



- model - implementa a lógica da aplicação (o que vai fazer!!!)
- presentation – apresenta a informação do modelo
- controller – introduz alterações no modelo (o modelo automaticamente notifica todas as apresentações).
- coordenador – responsavel pela gestão destes 3 componentes

Nota: utiliza-se o termo presentation enves de view, para realçar que o modelo de informação, não só consegue externalizar como elemento visual, mas também como um elemento que pode ser percesionado pelo ser humano.

Nota: O coordenador pode modificar a estrutura da aplicação e monitorizar o estado dos diferentes componentes e reagir de acordo com as politicas definidas pelo utilizador e sistema.

Qualquer entidade no espaço pode interagir com o coordenador e alterar a estrutra da aplicação.

mecanismo de mapping

No active space existem muitos dispositivos heterogenios (como ecrãs plasma, PDAs ou PCs). Para adaptar a execução de uma aplicação em cada um deles é necessario um mecanismo de mapping.

O mecanismo de mapping define dois tipo de ficheiros que descrevemas as aplicações: *application generic description*(AGD) e

application customized desctiption(ACD).

Uma **AGD**, que é criada pelo criador da aplicação. Este ficheiro lista os componentes da aplicação, o numero de vezes minimo e maximo que pode ser instanciada e os requisitos de componentes (output de audio e Windows OS).

Exemplo: winamp (musica,WindowsOS)

Uma **ACD**, lista os componentes da aplicação, incluindo os seus nós de execução associados (escolhidos de acordo com os requisitos dos componentes) e parametros de inicialização.

Nota: Um mecanismo de especialização utiliza um AGD e um space repositorio sevice, para gerar ACD's. O mecanismo de mapping utiliza o espaço de repositorio para encontrar dispositivos e serviços que se respeitem os requisitos do AGD.

- grupo de politicas

A infraestrura da framework das aplicações e mecanismo de mapping fornece as ferramentas para contruir e instanciar aplicações. As aplicações da framework baseiam em politicas de adaptação e mobilidade.

O utilizadores também podem definir as suas proprias politicas ou utilizar as politicas por default da framework.

Gaia management tools

O gaia utiliza uma linguagem de script, para coordenar as entidades digitais que correm no active space.

- Scripting Language

O gaia utiliza uma linguagem de script de alto nivel, como o luaOrb, para programar e configurar o active space e para coordenar as entidades que ele contem.

Lua Orb é baseada na linguagem Lua, que simplifica as tarefas de gestão e configuração e permite rápidos testes de prototyping.Lua é rápido e não ocupa muita memoria.

A capacidade de LuaOrb comunicar com vários modelos de componentes, faz com que a interacção seja mais fácil com componentes do sistema.

Utiliza-se Lua para implementar o algoritmo de bootstrap.

- Algoritmo de bootstrap

O Gaia implementa um algoritmo de bootstrap que interpreta um ficheiro de configuração (Lua Script) e inicia os serviços do kernel.

17 – System Software for Ubiquitous Computing

- O que nos fala este paper?

Este paper começa por dar a importância dos conceitos de **integração fisica** e **interoperabilidade espontânea** para a computação ubiqua. Estes conceitos devem estar presentes para quem pretende desenvolver sofware ubiquo. A integração fisica é um sistema ubicomp que envolve alguma integração entre os nós computacionais e o mundo fisico (Ex: uma smart meeting room que sente a presença dos utilizadores, grava as suas ações e fornece serviços quando estes se sentam ou estão a falar em frente a um ecrã). A inteoperabilidade espontânea diz-nos que é um componente tem uma acção espontanea se consegue interaguir com um conjunto de componentes que conseguem modar ambos a entidade e funcionalidades ao longo do tempo e as suas circunstancias mudem, por isso é um requisito desejado em ubicomp.

Exemplos dados:

Jogo P2P – Utilizadores jogam a um jogo de piratas, com dispositivos portateis conectados por uma WLAN. Em alguns casos envolve-se integração fisica porque os dispositivos dos clientes têm sensores, tal como sensores de proximidade nos piratas. Adicionalmente, alguns jogos podem descobrir outros jogadores pela LAN e efectuar uma associação dinâmica entre os jogadores (interação espontânea).

A implementação da Integração Física e da Interoperatividade Espontânea resultam em desafios colocados aos desenhadores de sistemas ubíquos.

A Integração Física implica que os recursos disponíveis estão por vezes altamente limitados.

A Interoperatividade Espontânea significa que os recursos disponíveis são altamente dinâmicos mas que o sistema tem que ser funcional em qualquer lado no ambiente, com pouca ou nenhuma intervenção humana.

Dadas as tecnologias actuais existem poucas provas de que o software sozinho consiga resolver os desafios colocados pela computação ubíqua.

Desenhadores de sistemas devem então efectuar escolhas sobre o que é que o sistema não vai fazer, mas sim os humanos

A semântica Rubicon demarca a responsabilidade para tomada de decisões de alto nível entre o sistema e o utilizador.

O resto do paper fala nas investigações realizadas ao nivel de:

> **Descoberta** - Quando um dispositivo entra num ambiente, como é que se dá a descoberta mútua entre ele e outros serviços ou dispositivos e com quais é que deve interagir ?

Considerando um componente que entra num ambiente, para satisfazer a Interoperatividade Espontânea o componente tem que resolver três questões:

- Boostrapping – a necessidade de ter informação à priori do ambiente. Porque necessito à priori de endereços e parâmetros necessários à descoberta de serviços e integração na rede.
- Descoberta de Serviços - Se ambos os componentes não usarem o mesmo vocabulário, podem acontecer casos em que o serviço desejado não é encontrado, mas existe. Demasiada **especificação**, pode ser um problema.
  Exemplo: Pode existir um serviço que satisfaça o o pedido do PDA, mas por não estar dentro da sua lista de *todos-os-serviços-suportados*, simplesmente passa despercebido.
- Interacção – este é outro problema que se poe, porque depois de ter descoberto o serviço e ter o seu objecto, como posso interagir com este? Se não sei à partida que metodos tenho de invocar para utilizar este serviço.

> **Adaptação** - Quando perto de outros dispositivos heterogéneos, como podemos usar o nosso dispositivo para mostrar ou manipular dados ou interfaces de utilizador de outros dispositivos no ambiente ?

Adaptação na computação ubíqua é substancialmente mais difícil do que na computação móvel. Em vez de se adaptar um pequeno número fixo de tipos de conteúdo a uma variedade de tipos de dispositivo (*1-to-n)* tem que se adaptar o conteúdo a *n* tipos de dispositivos heterogéneos (*n-to-n)*.

> **Integração** - O que é que liga o software do dispositivo e o mundo físico e o que é que afecta essa ligação ?

Para integrar ambientes computacionais com o mundo físico é necessário:

- Interfaces de programação aplicacionais de baixo-nível que permitam ao software lidar com os sensores e actuadores físicos.

- Framework de software de alto-nível que permita às aplicações ter uma percepção e interagir com o seu ambiente, incluindo os sensores e actuadores físicos.

> **Framework de programação** - O que é que significa escrever um "Hello World" num ambiente de computação ubíqua: São a descoberta, adaptação e integração responsabilidade da camada aplicacional, de middleware-SO, de linguagem ou um conjunto dos três ?

Existe a necessidade de Midleware, por exemplo, como o Gaia.

> **Robustez** - Como se pode proteger um dispositivo de faltas momentâneas e falhas semelhantes que afectam um sistema de computação ubíqua ?

> **Segurança** - O que é se pode deixar um dispositivo ou um utilizador fazer ? Em quem confiar e como se dá a autenticação ? Como se podem minimizar ataques à privacidade ?

18 – Agile Application-aware adaption for mobility

- O que fala este paper?

Este paper fala sobre a necessecidade de existirem arquitecturas que permitam às aplicações que estão a correr, se adaptarem às variações da qualidade dos recursos disponiveis (ex: lagura de banda). O sistema Odyssey lida com estes problemas.

Um exemplo: Consideremos um tourista com um PDA que corre o Odyssey. O Odyssey vai monitorizando os recursos como a largura de banda, ciclos do CPU, nivel de bateria e interage com cada aplicação para melhor utilização delas. Por exemplo, quando um conectividade de banda larga é perdida devido a uma interferência de radio. Odyssey detecta a alteração e notifica as aplicações interessadas. Uma aplicação de video, poderia responder com passagem de frames, mostrando menos frames por minuto. Uma aplicação Web respondia, mostrando versões degradadas (Ex: jpg -> gif) de imagens grandes. Quando o utilizador emergi-se da zona de radio que causava interferencias, Odyssey detecta um subtancial melhoramento na largura de banda e notifica as aplicações, e elas revertem para o seu comportamento normal.
Embora o utilizador esteja consciente da alteração no comportamento da aplicação durante a caminhada, não tem de se preocupar com a inicialização da adaptação ou envolver-se nos seus detalhes. Ele pode delegar estas decisões para o Odyssey, confiante de que se adaptará às limitações.

- O que é o Odyssey?

O Odyssey é um sistema que permite às aplicações adaptarem-se às variações imprevisiveis da qualidade dos recursos disponiveis. Também é o primeiro sistema a considerar simultaneamente os problemas de adaptação para mobilidade, diversidade e concorrência de aplicações.

- O que é fidelidade? Que tipos de fidelidade existem?

Representa o grau de qualidade com que os dados são mostrados ao cliente.
Existem 2 tipos de fidelidade:

> **dimensão universal** – que é a consistência, ou seja, a manutenção da informação das aplicação quando a conectividade é fraca ou inexistente (Ex: Coda ou Bayou).

> **outras dimensões** – depende do tipo de informação em questão. Por exemplo, informação de video tem pelo menos duas dimensões: frame rate e qualidade de

imagem.

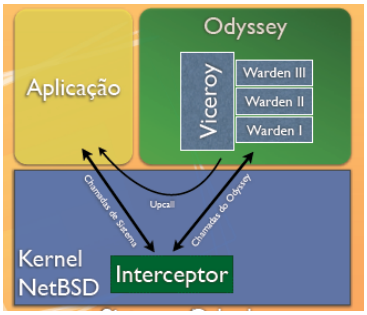
- O que é agilidade?
Tem a ver com a velocidade e exactidão com que um sistema móvel detecta e responde à mudança dos recursos disponíveis.
Nota: Um sistema pode ser muito mais sensível à mudança da largura de banda do que mudanças no nível de bateria.

- A diversidade de aplicações é suportada?
Sim, desde que deixem as aplicações determinarem a sua melhor fidelidade ao longo do tempo de acordo com os recursos disponiveis.

- O que concorrência? A sua importância?
A concorrência consiste na execução de multiplas aplicações independentes concorrentemente no dispositivo movel do cliente. Com a sua existência podemos ter aplicações a corre em background de bastante utilidade. Exemplo: uma aplicação que monitoriza preços em stock ou movimentos inimigos e alerta o utilizador quando necessário.

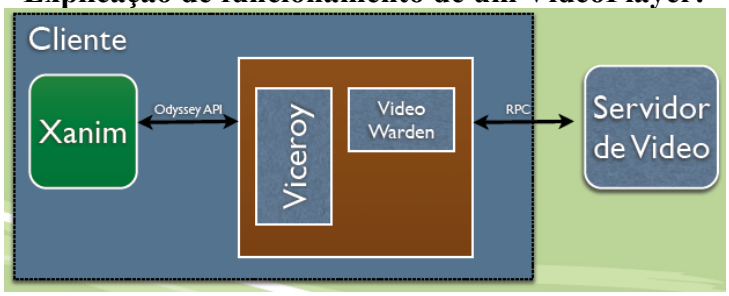
- Porque é que o Odyssey é application-aware (aplicação-ciente)?
O essencial deste modelo é uma parceria colaborativa entre o sistema e as aplicações individuais. O sistema monitoriza niveis de recurso, notifica aplicações das alterações relevantes. Cada Aplicação decide independentemente como melhor se deve adaptar quando notificada.

- Como funciona a arquitectura do Odyssey client?



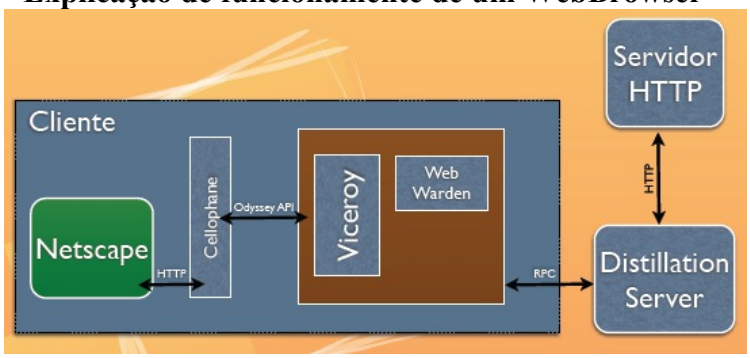
- O Viceroy e os Wardens foram implementados no userspace
- Todas as operações com os objectos do Odyssey são redireccionados para o viceroy por uma pequeno modulo in-kernel inteceptor. Todas as outras chamadas são entregadas directamente para o NetBSD (Odyssey foi implementado sobre uma base NetBSD).
- Os Wardens estão ligados ao Viceroy. A comunicação entre o viceroy e os wardens é feita por procedure call e estruturas de dados partilhadas.
- Os Wardens estão completamente responsaveis por comunicar com os servidores e fazer caching da informação proveninte deles quando apropriado. De notar que as aplicações não contactam os servers directamente.

- Explicação de funcionamento de um VideoPlayer:



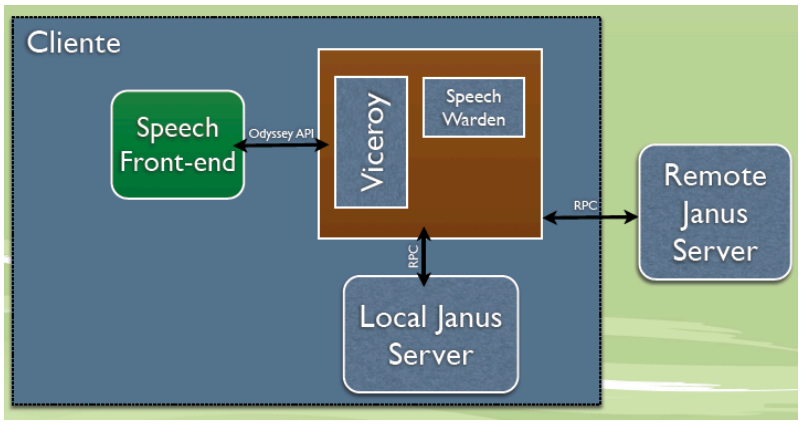
- Como mosta a figura, temos a parte cliente e servidor. Temos um warten para sarisfazer os pedidos do cliente e realizar o fetch da informação vinda do server.
- No servidor estão 3 tipos de tracks, cada um correspondente a um grau de fidelidade (uma com excelente qualidade, outra com menos qualidade e outra a preto e branco).
- O warden suporta 2 tstops: para ler o metadados do filme e buscar o frame particular do especifico track.
- Quando o videoplayer abre um filme, ele calcula os requisitos de largura de banda para cada track apartir do metadata do filme. O videoplayer começa o filme à qualidade máxima possivel e regista a correspondente tabela de tolerância com o Odyssey.
- Quando for notificado uma alteração significativa na largura de banda, o player determina o novo nivel de fidelidade e troca o correspondente track. Se o videoplayer trocar de uma fidelidade de baixa qualidade para um de maior qualidade, então o warden descarta os frames de baixa qualidade que foram pre-fetched.

- Explicação de funcionamente de um WebBrowser



- Todos os pedidos do netscape são redireccionados para um módulo do cliente que se chama cellophane. Juntos, Nescape e cellophane constituem uma unica aplicação, do ponto de vista do Odyssey. O cellophane faz uso da API do Odyssey e selecciona os niveis de fidelidade. O Netspace benificia da adaptação fornecidade pelo cellophane.
- O cellophane transforma pedidos HTTP do netscape em ficheiros de operações nos objectos web Odyssey. O warden do Odyssey encaminha estes pedidos via a conexão à rede movel para o distillation server. O deslillation server faz fetch dos objectos pedidos para o apropriado web server, deslita-os para o pedido nivel de fidelidade e envias os resultados para o warden. A informação é passada para o Netscape pelo cellophane. Estes passos são completamente transparentes para o Netspcape e para o webserver

- Explicação do funcionamento de um reconhecedor de voz



- O ponto de começo para esta implementação é um sistema para reconhecer voz, chamado de Janus. Foi criado um Warden para a fala.
- Quando a largura de banda for fraca ou nenhuma terá de ser utilizado o Local Janus Server.

- Feedback dos testes realizados ao Odyssey
Os testes demonstram que o protótipo faz um bom trabalho equilibrando performance e fidelidade e confirma a importância de ser utilizada gestão centralizada de recursos

19 – Mobile computing with Rover ToolKit

- O que nos fala este paper?
Este paper fala-nos sobre a ferramenta Rover que permite criar suporte para as aplicações móveis, tornando estas menos dependentes da conectividade e dos dados armazenados no server. É explicado os componentes desta arquitectura e o seu funcionamento e também efectuado testes a esta ferramenta.

- O que é o Rover?
Rover é um software toolKit que suporta a construção de aplicações **mobile-transparent** e **mobile-aware**.

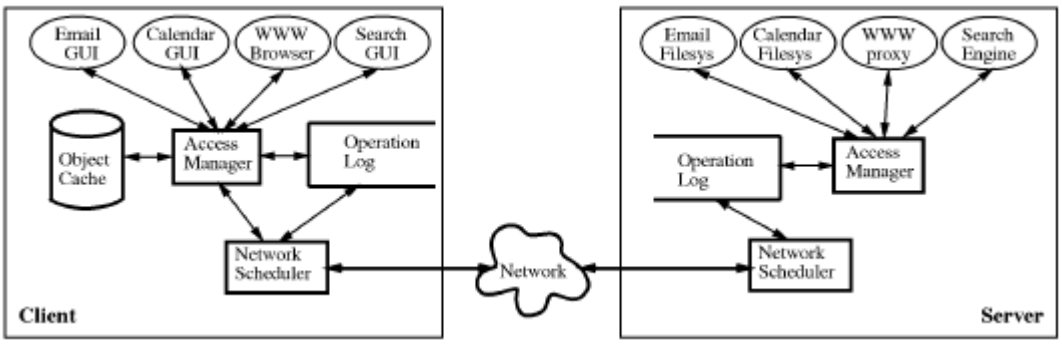
- Mobile-transparent VS mobile-aware
Uma abordagem **mobile-transparent** tem como objectivo tornar as aplicações existentes capazes de correr num ambiente movel sem alteração. Esta transparência é alcançada ao desenvolver-se proxies para um sistema de serviços que escondam as caracteristicas moveis do ambiente das aplicações.

- O que se ganha com a utilização do Rover ToolKit?
Usando as abstracçoes de programação e comunicação presentes no Rover ToolKit, aplicações aumentam a sua disponibilidade, concorrência, alocação eficiente de recursos, tolerância a faltas, consistência e adaptação.
Experiencias feitas com o Rover, mostam que o ToolKit requer pouca programação, permite correcta operação, aumento substancial de performance e reduz dramaticamente a utilização da rede.

- Como é feito suporte para a comunicação movel?
É baseada em duas ideias: RDOs e QRPCs.

- Explique o que é RDOs e QRPCs?
RDOs (relocatable dynamic object) são objectos (código e dados) que podem ser dinamicamente carregados de um servidor para um cliente ou vice-versa, reduzindo exigências de comunicação entre cliente e servidor.
QRPCs (queued remote procedure call) é um sistema de comunicação que permite às aplicações continuar a efectuar RPCs sem bloquear, mesmo quando o dispositivo está desconectado.

- Explicação do funcionamento da arquitectura do Rovel ToolKit



O Rover ToolKit consiste em 4 componentes chave: acess manager; object cache; operation log; network scheduler. Estas peças formam o "kernel" minimo do Rover.
- **Access Manager**
> Importa RDOs dos servidores para a cache dos clientes
> A pedido das aplicações, invoca métodos que alterem RDOs inserindo QRPCs no Operation Log
> Do lado do servidor, invoca as aplicações para que lidem com os *requests* (QRPCs) dos clientes
> Gere a cache do cliente
> Lida com recuperação de falhas
- Após uma falha, reconstrói a lista de QRPCs para serem reenviados.
- Semântica de entrega "At-Most-Once" com IDs únicos para os QRPCs.

- **Operation log** - mantém o armazenamento estável de QRPCs
- **Object Cache** - fornece armazenamento estável para cópias locais de objectos importados pelo acess manager.
- **Network schedular** é responsavel por examinar cada QRPC não enviado para determinar o seu destino e o protocolo de transporte a usar. Portanto:
> Envia os QRPCs existentes no Operation Log
> Usa heurísticas para agrupar QRPCs que estão destinados para o mesmo Servidor:
- Quando recebe um QRPC de uma aplicação interroga o Acess Manager para verificar se alguma aplicação está prestes a enviar um QRPC
- Se sim, atrasa o envio do primeiro e repete a verificação.
- Quando já não houver QRPCs "a chegar", ou quando o primeiro já estiver atrasado um limite máximo de tempo, agrupa os QRPCs e envia-os na mesma conexão.

- Que conclusões se tiram dos testes realizados com o Rover?

- A integração de RDOs e QRPCs proporciona uma base poderosa para construir aplicações móveis *transparent* e *aware*.
- É bastante fácil adaptar aplicações para usar as funcionalidades do Rover.
- As aplicações ficam muito menos dependentes da conectividade e dos dados armazenados no servidor.
- Verificam-se significativas melhorias na performance em redes lentas.

20 – Location Systems for Ubiquitous Computing

- O que fala este paper?
Este paper fala sobre tecnicas de localização que são bastante importantes em computação móvel, por exemplo a para localização de pessoas num active space.

- Que tipo de tecnlogias de localização existem?
Triangulação, Proximidade e Analise de cenarios.

- Explicação de localização fisica VS localização simbolica
A localização fisica dá uma posição real (Ex: GPS -> 47°39′17″ N).
A localização simbolica envolve ideias abstractas na posição dos objectos (Ex: Anfiteatro A4).

- Explicação de localização absoluta VS localização relativa
A localização absoluta usa uma grelha de referencias partilhada da localização dos objectos (Ex: Dois dispositivos GPS no mesmo local indicam as mesmas coordenadas).
Para a localização relatica cada objecto poderá ter o seu ponto de referencia
(Ex: Anfitreatro A4 encontra-se entre os anfiteatros A3 e A5).

- Que os locais de computação de localização que conhece? Explique-os
Existe a computação realizada no dispositivo, que assegura a privacidade (Ex: GPS) e existe a computação realizada na infra-estrutura, que não assegura a privacidade (Ex: RFID).

- Explicação de exactidão VS precisão
A exactidão refere-se à margem de erro da localização (Ex: 10m).
A precisão indica a frequência em que se consegue alcançar a exactidão (Ex: 95%)

Exemplo: os dispositivos de GPS mais baratos conseguem dar posições com uma exactidão de +-10 metros e uma precisão de 95% das medidas. Os dispositivos mais caros conseguem alcançar uma exactidão de 1 a 3 metros e uma precisão de 99%.

- Para avaliar a escalabilidade de um sistema o que é preciso fazer?
Saber a área de cobertura e o nº de dispositivos que o sistema localiza por intervalo de tempo.

- Explicação de reconhecimento em localização
A técnica usada para providenciar o reconhecimento de objectos é usar um Gobal Unique ID (GUID) associado aos objectos. Assim a infra-estrutura poderá aceder a uma base de dados externa e poder desta forma retirar informações sobre o objecto. Pode ainda combinar esta informação com o contexto do objecto e assim poder interpretar o

mesmo objecto de variadas maneiras consoante o contexto e que este está inserido.
Exemplo: soar um alarme em caso de roubo de um objecto

- Apresente os vários sistemas de localização que conhece?

- Active Badge
  - Técnica de proximidade por uso de infra-vermelhos
  - Baixa exactidão
  - Susceptível a interferências pela luz solar e luzes fluorescentes
  -
- Active Bat
  - Evolução do sistema Active Badge
  - Uso de triangulação por ultra-sons
  - Alta exactidão e precisão
  - Complexo de instalar

- RADAR
  - Baseado em IEEE 802.11
    - Utiliza a infra-estrutura 802.11 já instalada

  - Duas implementações
    - Triangulação
      - Exactidão de 4.3 metros com precisão de 50 %
    - Scene Analysis
      - Exactidão de 3 metros com precisão de 50 %
      - Susceptível a falhas perante mudanças no ambiente

- Smart Floor
  - Usa sensores de pressão
  - Sistema não intrusivo
  - Elevada exactidão e precisão
  - Custo elevado e baixa escalabilidade

- E911
  - Localização por uso da rede GSM
  - Usa a técnica da triangulação
  - Exactidão de 150-300 metros com precisão de 95%

20– An Application of context-aware file system

- O que fala este paper?
Este paper fala-nos da importância dos context file system em computação ubiqua, principalmente desenvolvido para active spaces. É feita também referência ao context file system do Gaia. Os context file systems baseiam-se na informação do contexto para organizar os dados aplicacionais.

- O que é um active Space?

É uma junção do mundo fisico e do mundo virtual, feita com base na coordenação do mundo fisico com base em software. Ambientes que consistem em salas inteligentes contendo uma panóplia de dispositivos.

Este espaço activo interage com os utilizadores de forma dinâmica.
Exemplo: Um utilizador com um PDA entra na sala. A sala detecta a presença do utilizador e os seus dados pessoais e preferencias ficam disponiveis no espaço. O utilizador seleciona a apresentação com o gestor de ficheiros do espaço e selecciona a condiguração pessoal que corre a apresentação em 2 ecras e coloca a componente de controle no PDA.

- O que é o contexto? Qual a sua importancia?

O contexto define a informação que é importante para uma actividade. A sua utilização facilita tarefas efectuadas em active spaces.

- Dê exemplos de como pode ser utilizado

1) disponibilizar automaticamente armazenamento pessoal às aplicações, em função da presença fisica.
2) organizar dados para simplificar a localização de dados importantes para as aplicações e para os utilizadores.
3) obter dados num formato baseado nas preferencias do utilizador ou nas caracteristicas do dispositivos.

- Explicação do funcionamento de um context file system

Merge dos dados – os dados de um utilizador são adicionados dinamicamente a /users/<username> sempre que ele é detectado no espaço.

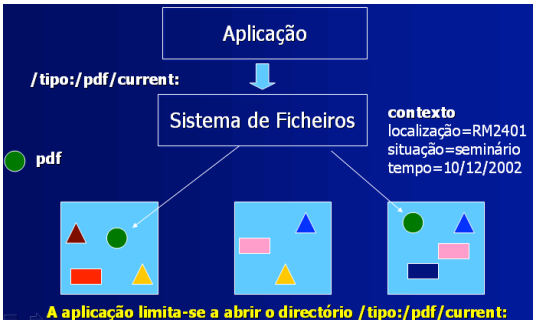
O sistema de ficheito opera em 2 modos:

Modo ficheiro – torna todos os dados que os utilizadores exportaram acessiveis no directorio do utilizador.

Modo contexto – a visibilidade dos dados é limitada para aquilo que é importante no contexto corrente.

Marcação dos dados – utilização de metadados para marcar ficheiros e directorios. As propriedades são definidas pelo utilizadores ou aplicações, por forma às aplicações organizarem a sua informação, enquanto que a informação de contexto é definida pelo ambiente. Para ficheiros especificos com metadados especificos, consiste em copia-los de modo de ficheiro para um directorio virtual em modo de contexto representando um numero variavel de atributos (localização, situação, tempo, dispositivo, etc...). Os caminhos são criados hierarquicamente virtal da forma [/<tipo:>/<valor>], por isso, dados com o mesmo par tipo/valorsão agregados nos mesmo directorio.

Exemplo: localização = =2401 && situação = = reunião
coloca os dados no directorio /localização:/2401/situação:/reuniao



Exemplo:

- O sistema criar um directorio de contexto por utilizador /user:/username.
- Neste exemplo a palavra current, indica ao sistema para procurar apenas os dados do contexto corrente.
- Os directorios em modo contexto sao um floresta e não uma arvore de directorios.

Neste exemplo o sistema selecciona apenas os dados que dizem respeiro ao contexto corrente, ou sejam, os pdfs que dizem respeito ao seminarios a ser apresentado.

Nota: Atributos – marcam os dados e resultam de um acordo entre aplicaçõe espaço.
Propriedades – são utilizadas por aplicações para especificar a informação em que estão interessadas.
Contexto – usado para determinar em que situações os dados são disponibilizados às aplicações.

Conversão de dados – dispositivos heterogeneos e preferências de utilizador são suportados atraves da utilização de tipos dinâmicos. Tipos dinâmicos são implementados atraves de modulos de conversão.

```
<CFS:Container>
    <CFS:File/>
    <CFS:Interface>GIFContainer</CFS:Interface>
    <CFS:Name>PowerPointContainer</CFS:Name>
    <CFS:Extension>.ppt</CFS:Extension>
    <CFS:Output>gif</CFS:Output>
</CFS:Container>
```

Fig. 4 An example XML module description for converting PowerPoint files to GIF objects

- O que entende por mount server?
Implementa a funcionalidade de base de dados para queries sobre propriedades de ficheiros e informação de contexto. Portanto o mount server:
> mantém uma tabela de referência dos dados disponiveis no espaço XML.

- O que entende por file server?
Implementam a funcionalidade de sistema de ficheiros e disponibilizam uma interface para acesso a ficheiros remotos.

21 – Reasoning about Uncertain Contexts in Pervasive

- O que nos fala este paper?
Fala-nos sobre a importância do contexto e como é que podemos definir contextos em computação movel. Como se sabe determinar o contexto é algo complexo, porque nunca sabemos ao certo se ele está certo, e isso dificulta a vida do programador. Por isso este paper fala-nos do modelo de incertezas que nos permite ultrapassar estes problemas, desenvolvendo-se assim linguagens de alto nivel que permitem uma correcta programação nesse sentido. São apresentados vários mecanismos, como **lógica probabilistica**, **fuzzy logic** e **redes bayseanas**, que podem ser integrados nesse modelo. Antes das explicações é importante saber que um modelo de incerteza é baseado em predicados (representam o contexto) e valores de confiança (representam as associações).

- O que é um predicado de contexto?
Um predicado de contexto representa um contexto e pode ter os seguintes formatos:
ContextType(<Subject>, <Object>) ou
ContextType(<Subject>, <verb>, <Object>)
Exemplos:
Context(temperatura,salaN1.1,is,98F)
Context(numero pessoas,salaN1.2, >,4)

- O que é um valor de confiança?
É um valor que é anexado aos predicados, entre [0,1].
Exemplo:
prob(location(Carol, in, room 3233)) = 0.5

- Quais são os vários modelos que conhece para definir contexto? Explique-os
Existem os modelos de lógica probabilistica, fuzzy logic e redes bayseanas.
Lógica probabilistica – mecanismo que permite a escrita de regras que dizem respeito à probabilidade dos eventos em termos das probabilidades de outro eventos relacionados.
Exemplo:
prob (X,Y,union, P) :- prob (X,Q), prob (Y,R), disjoint (X,Y), (P is Q+R)

Fuzzy Logic – é semelhante à lógica probabilistica. Em Fuzzy Logic, valores de confiança representam melhor graus de relações do que probabilidade. Fuzzy Logic é util na captura e representação de notações, como "trustWorthy" e "confidence" e realiza o raciocinio sobre eles. Nós podemos escrever regras involvendo vários predicados de contextos e fazer o raciocinio baseados nestas regras.
Exemplo:
IF height <= medium male THEN is_short IS agree somewhat
IF height >= medium male THEN is_tall IS agree somewhat

Redes bayseanas – são grafos em que os nós representam um determinado evento e as relações entre nós determinam as relações causais entre eles. Por isso, quando se calcula uma determinada probabilidade tem de se ter em conta que temos probabilidades condicionadas.

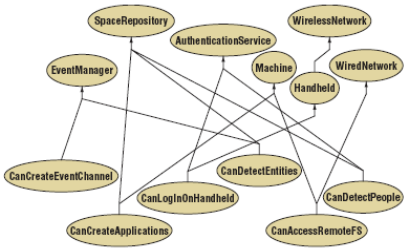
- Qual a importância das redes bayseanas?
As redes bayseanas são uma forma poderosa de lidar com incertezas, especialmente com relações causais entre vários eventos.
No exemplo do paper a variavel random activity, pode ter valores como "meeting", "presentation" e "idle" e outros mais, com diferentes probabilidades. Assumindo que treinamos a rede para a sala 2401 os predicados dos contextos activity(sala 2401, meeting); activity(sala 2401, presentation) e activity(sala 2401, idle) tem predicados diferentes.

- Explicação do código para controlo de acesso por parte de um utilizador

```
canAccess(P, display) :-
  confidenceLevel(authenticated(P), C), C > 0.7,
  Prob(activity(2401,cs 101 presentation), Y),
    Y > 0.8,
  possessRole(P, presenter)
```

O utilizador P pode aceder ao display se foi autenticado com uma confiança de pelo menos 0.7, a actividade na sala é de apresentação com uma probabilidade de pelo menos 0.8 e P está presente para a actividade de apresentação.
O predicado autenticated, está relacionado com base na informação do servidor de autenticação e o predicado activity é obtido do Activity Contect Provider (que utiliza a rede bayseana para determinar a probabilidade associada aquela activiodade).

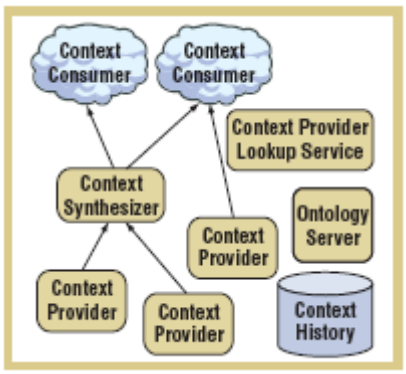
- Explicação da resolução de erros



As redes bayseanas também podem ser utilizadas para resolver problemas no ambiente. Com a analise de vários sintomas podemos saber se os serviços, aplicações, dispositivos ou outros recursos falharam. Por exemplo se um smart room não detectar presença de pessoas na sala, isso implica que o repositorio do espaço (que mantém num BD o estado actual da sala) ou o serviço de autenticação falharam com uma determinada probabilidade condicional.

Na imagem, esta rede bayseana consiste em nós como EventManager e AuthenticationService que representam o estado das diferentes entidades do gaia. Outros nós representam certos tipos de falhas. Por exemplo, o nó CanCreateEventChannel representa se é possivel ou não criar um canal.

- Como funciona a infraestrutura de contextos do gaia?



Context Consumer – são entidades (aplicações context-aware) que obtem diferentes tipos de contexto dos context providers ou context synthesizers
Context Provider – são sensores ou outra fonte de informação para a informação de contexto.
Context Synthesizer – obtem contextos de vários context providers, deduz-se contexto alto-nivel ou abstracto para simpres contextos, e depois fornece-se estes contextos a outros agentes.
Context Provider Lookup service – permiteos context providers de anunciar o que eles oferecem e os agente apropriados para encontrar os context providers apropriados.

Ontology Server – mantém as ontologias que descrevem os diferentes tipos de informação contextual.

Context History – permite aos agentes questionar por contextos antigos, que estão no logg de uma base de dados.

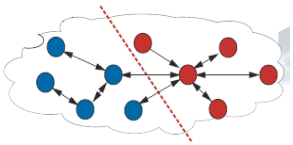
22 – Adaptive Offloading for pervasive computing

- O que fala este paper?
Fala-nos do desafio de possuir aplicações complexas, cujos recursos necessários estão limitados pelo dispositivo móvel (Ex: capacidade de mémoria, consumo de energia). Uma abordagem discutida neste paper é o Adaptive offloadding system, ou seja, um sistema de descarregamento adaptivo, que permite o partiticionamento dinâmico da aplicação e um eficiente descarregamento de partes de execução dessa aplicação para um surrogate nas emediacões da rede. Basicamente são replicados objectos para o surrogate, que são os menos usados por forma a não aumentar a latência com os RPCs para o surrogate.

- Qual a vantagem desta abordagem?
Se um utilizador quiser utilizar uma aplicação que necessite de muita memoria, e se tivermos um dispositivo movel limitado como um PDA, esta abordagem é altamente vantajosa. Quando a capacidade de memoria do dispositivo chegar à sua capacidade máxima então o sistema inicia um descarregamento. O sistema partiticiona os objectos do programa da aplicação em 2 grupos, descarregando alguns para um poderoso surrogate que se encontre perto para reduzir a utilização da memoria do dispositivo. A plataforma de descarregamento é transparente, transformando invocações de metidos para objectos descarregados em invocações remotas.

- Porque é que o metodo de brute force é mau?
Porque não queremos gartar tempo e dinheiro a desemvolver uma aplicação especifica para cada dispositivo para não ter estes problemas. Este offloading adaptativo resolve essa abordagem pedreira.

- Explicação das caracteristicas da plataforma distribuida de "offloading"
> Monitorização da Execução da Aplicação



- Uso de um grafo, mantido no *mobile device*, para caracterizar a execução de uma aplicação (AEG).
- Nó = Classe

- Aresta = representação das interacções e dependência entre duas classes

- Métricas de cada nó:
  - ■ Tamanho memoria - descreve quanta memoria o objecto da class Java ocupa
  - ■ Frequência de acesso- representa o numero de vezes qie outra class acede a outros metodos de outras classes ou campos.
  - ■Localização - descrever se o objecto da class Java reside no dispositivo movel ou no surrogate
  - ■*IsNative* – indica se um objecto da class pode migrar do dispositivo movel para o surrogate.

- Relações entre classes – dois campos:
  - ■ Interaction Frequency (entre 2 classes) - representa o numero de interacções entre estas dois objectos de classes.

- Bandwidth Requirement - representa o total de informação tranferido entre estes objectos.

> Monitorização dos Recursos

É necessário monitorizar os recursos do dispositivo movel, surrogate e da rede wireless. A memoria disponivel é medida através do espaço livre no Java Heap, e a largura de banda é estimada através da observação do trafego. Quando existem variações nestes valores, o mecanismo de inferência decide se activa ou não o descarregamento Passivo.

> Geração das várias possibilidades de particionamento da aplicação

É criado um grafo para descrever um plano de particionamento.

Cada aresta corresponde a um custo que reflecte o número de interacções entre classes e as suas dependências. Existem 2 grafos, PM(mobile) e PS (surrogate) que inicialmente estão vazios

> Plataforma de Suporte para Chamadas a Procedimentos Remotos (RPC) de uma forma transparente

Tem de ser utilizada uma maquina virtual modificada para suportar RPC de forma transparente.

- Quando provocar o "offloading"?

É utilizado o "Fuzzy Control Model", que inclui regras de decisão para especificar as regras de Offloading.

- Como escolher melhor o partiticionamento?

É escolhido o plano que minimiza o custo da partição.

23 – Automatic Partitioning for Prototyping

- Do que fala o paper?

Começa por falar na dificuldade que existe no desenvolvimento de aplicações ubicomp, porque existem muitas complexidades pelas quais os programadores têm de ultrapassar, como a **programação dos sistemas distribuidos**. As maiores dificuldades que os investigadores encontram na construção de aplicações ubicomp, estão mais pronunciadas durante a **investigação** e **desenvolvimento de um prototipo**. Para facilitar a prototipagem da aplicação neste dominio, o pessoal de desenvolvimento têm de ser capazes de modificar as estruturas de dados subjacentes às caracteristicas de distribuição com o menor de esforço. Infelizmente, os investigadores frequentemente não são experientes em sistemas distribuidos. Como resultado, as investigações ubicomp necessitam de uma solução simples, tecnicas automatizadas para suportar a rápida prototipagem desses dominios.

Assim entra a parte importante do paper, ou seja, explicação da tecnologia de prototipagem automatica da aplicação.

- O que é particionamento automatico da aplicação? Em que consiste?

É uma tecnologia para desenvolvimento de uma classe de aplicações ubiquas.
O particionamento automatico é um processo para adicionar capacidades distribuidas a uma aplicação centralizada, sem precisar de reescrever o código fonte da aplicação ou modificar existentes sistemas de runtime.

A ideia principal é simples: Uma ferramenta automatica recebe como input um programa normal e o utilizador fornece informação para o programa dos dados e do codigo. A ferramente reescreve o programa para que o código e a informação sejá dividida em partes que possam correr em localizações desejadas. Qualquer alteração da informação em alguma das partes em diferentes localizações, automaticamente se tornam comunicações remotas.

- O que é o J-Orchestra?
É uma ferramenta ideal para computação ubiqua, que permite uma rapida prototipagem dos projectos, capaz de tornar uma aplicação centralizada numa aplicação distribuida. O investigador apenas tem de se preocupar com a lógica da aplicação.

- Como funciona o algoritmo de analise do J-Orchestra?
A solução do J-Orchestra consiste neste algoritmo de analise, que tenta determinar heuristicamente qual a referencia escapa ao código não modificado (unmodifiable) e sofisticadamente rescreve o algoritmo que injecta código para transformar referencias directas em referencias indirectas em referencias directas e vice-versa em runtime.

- Como funciona o algoritmo de escrita?
Transforma referências directas em indirectas

- O que é o Kimura e o Kimura2?
O Kimura é um projecto ubicomp que explora e avalia a inclusão de displays para interfaces pessoa-máquina. O Kimura2 é uma simplificação do Kimura com a utilização do J-Orchestra.

Nota: Esta simplificação resultou numa arquitectura mais simples, com menos classes e código, permitindo uma manutenção e gestão mais fácil.

23 – Mobile device security using transient Authentication

- O que nos fala este paper?
Este paper fala-nos que de uma situação muito especifica, sobre o mecanismo de autenticação em dispositivos portateis. Os dispositivos portateis são dispositivos moveis que podem ser levado para qualquer lugar (Ex: casa, jardim, centro comercial) e como tal pode estar sujeitos a roubo ou à exposição de informação (Ex: O joão vai ao bar e entretanto alguem pode tentar roubar-lhe os seus trabalhos na sua ausencia). Existem vários mecanismos para autenticação, como passwords, SmartCards ou mesmo por impressão digital. Está certo podem garantir a protecção da informação, mas para computação ubiqua este processo de autenticação não é transparente ao utilizador e a sua pratica sistematica é cansativa para o utilizador (autenticar-se sempre que se tenha de ausentar). Por isso, a abordagem tem de ser feita de forma a permitir ao utilizador ter os seus dados acessíveis apenas quando está presente de uma forma transparente. É essa a abordagem do paper que fala que um utilizador utiliza um token (Ex: um relogio de pulso), que o permite autenticar perante o dispositivo. Este token utilizaria assim uma ligação wireless de curto alcance para comunicar com o dispositivo.

- Quais os 4 fundamentos base subjacente a este mecanismo?

Ligar capacidades de execução ao utilizador – O dispositivo móvel deve apenas executar operações sensíveis quando o utilizador está presente.

Não prejudicar – O mecanismo de segurança não pode ser intrusivo nem ter uma influência negativa na performance

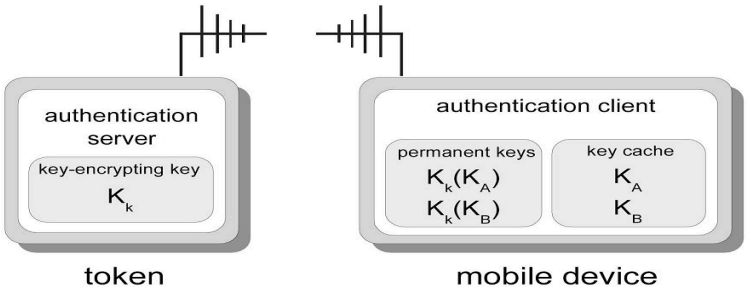
Guardar e Restaurar em tempo útil – Quando o utiilizador se afasta do dispositivo móvel este deve proteger-se de eventuais acções de terceiros (p.e. cifrando dados). Quando o utilizador se aproxima o sistema deve repôr o estado inicial.

Assegurar o consentimento explicito - O dispositivo móvel não deve tomar decisões de segurança sem consentimento do utilizador.

- Qual a diferença entre suspender o dispositivo e hibernar o dispositivo?

Quando suspendemos o dispositivo os contextos dos processos ficam guardados em memoria, para serem utilizados quando o utilizador estir presente.
Quando hibernamos o dispositivo os contextos dos processos são guardados em disco.

- Explicação da arquitectura



O token corre o processo de um autentication server e o dispositivo corre o processo de um autentication client. Estes dois processos comunicam via uma ligação wireless de curta distância. Esta comunicação está protegida por uma chave de sessão estabelecida. O token envia nonces periodicos de segundo a segundo, para indicar que o utilizador ainda está presente. Os nonces previnem replay attacks. Quando o autentication client no dispositivo deixar de receber um nonce esperado, ele declara que o utilizador está ausente e faz uma gestão da memoria. Quando o utilizador regressa o autentication client detecta a sua presença e restaura o seu estado original.

Existe aqui algumas considerações, se o alcançe de comunicação entre os dois for muito reduzido, o cliente pode virar-se na sua cadeira e os mecanismo de segurança serem activados desnecessariamente. Mas se o alcançe de comunicação entre os dois foi grande, o utilizador pode estar na sala, mas pode deixar o dispositivo aberto a ataques.

A comunicação entre o token e o dispositivo deve ser quebrada quando o dispositivo estiver mais de alguns metros de distancia.

Existem várias chaves no lado do cliente que são responsaveis por cifrar/decifrar os dados (ex: memoria, disco, etc...), mas que no nosso exemplo são as chaves K_A e K_B. Quando é activado os mecanismos de segurança as chaves K_A e K_B são cifradas com a chave K_k que é do token. As chaves K_A e K_B são apagadas quando o cliente se afasta do alcançe do dispositivo. Quando o cliente regressa, as chaves K_A e K_B são decifradas pelo cliente, mas as chaves cifradas de K_A e K_B não são apagadas (por isso só são criadas uma vez). Uma alternativa a este processo era utilizar-se um File system capaz de incorporar mecanismo de segurança nas suas directorias.

- Qual o problema gravissimo desta arquitectura?

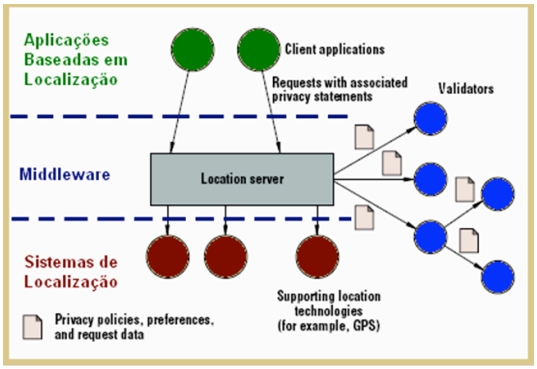
O utilizador pode perder o token

24 – Preserving Privacy in environments with Location-based applications

- O que fala este paper?
Este paper fala-nos na necessidade de existirem aplicações baseadas em localização que tenham em conta a privacidade da pessoas, relativamente à sua localização. Para este desafio é explicado um mecanismo que permite aos utilizadores controlar de forma automática a sua localização. Como tal é apresentado uma framework para solucionar este problema, que permita definir politicas para tratar a distribuição da sua localização.

- Um exemplo que ilustre esta necessidade?
Imaginemos que a Saly trabalha para a herbal-life, e que a empresa consegue localiza-la com base no seu telemovel. A empresa é contactada pelos clientes e no caso destes se encontrarem perto da zona da saly, eles telefonam à saly para se deslocar à casa desses clientes. Mas se a saly estiver fora do horario de trabalho ela não quer ser localizada. Imaginemo que a Saly vai a um restaurante, pede um hamburger e senta-se numa mesa, concerteza que a Saly quer ser localizada.

- Explicação da arquitectura do sistema



Nós assumimos a existência de um servidor de localização que responde às aplicações, que fazem querys sobre a localização dos utilizadores.
Estas querys podem ser de 3 tipos:
- **Localização utilizador** – pedidos para a localização especifica de utilizador(es), identificados por identificador unico.
- **Lista de utilizadores** – pedidos para ober as listas de utilizadores num localização especifica, expressa em termos geográficos ou atributos simbolicos.
- **Pedidos Assincronos** – pedidos por informação de eventos, tais como quando um utilizador entra ou sai de uma area especifica, ou quando existe uma relação de proximidade(Ex: o John estar a 100 metros da Jane).

Os **validators** são responsaveis por:
> Verificam a aceitabilidade das políticas de privacidade das aplicações
> Determinam se um pedido é aceite
> Podem chamar outros validators para auxiliarem a tomada de decisão
> Exemplo:
Confirmação do utilizador

25 – Mobile Agent Middleware for Mobile Computing

- O que nos fala este paper?
O paper começa-nos por falar da integração da Internet com as redes de telecomunicações, que assim conseguia assim globalmente tornar acessivel qualquer serviço. Fala-se entao da creste conectão de dispositivos moveis à Internet. E deste cenário definiram várias formas de **mobilidade**. Depois é apresentado uma framework SOMA é uma plataforma assente em Java e providencia uma infrastrutura de serviços

em camadas para desenhar, implementar e correr aplicações de Internet baseadas em Agentes Móveis.

- Que formas de mobilidade existem?
Mobilidade do utilizador – fornece ao utilizadore uma visão uniforme dos seus ambientes de trabalho preferidos, ou seja, o utilizador tem as suas preferências independentemente da sua localização na rede.
Mobilidade do terminal – permite aos dispositivos moverem-se transparentemente e conectarem-se a diferentes pontos, ou seja, o terminal do utilizador move-se e liga-se na rede de uma forma transparente

Mobilidade de Acesso a recursos – consiste no terminal, em movimento, ter a capacidade de aceder a serviços de localizações diferentes e a capacidade da rede de identificar e localizar esse mesmo terminal.

Notas: Manter o acesso a recursos requer soluções de "naming" que mantenham a informação acerca da disponibilidade e alocação de recursos e serviços.

Descoberta de Serviços – utiliza protocolos simples para obter informação acerca da localização da entidade. (Ex: UPnP, Jini e Service Location Protocol)

Directório de Serviços – organizam nomes e propriedades de entidades registadas de uma forma flexível e facilitam a navegação entre toda a informação registada. (Ex.: X.500 directory access protocol, LDAP)

Utilização de descoberta e directório de serviços para manter e actualizar ligações a recursos e serviços em cenários móveis ainda está no inicio.

- O que é um agente móvel?

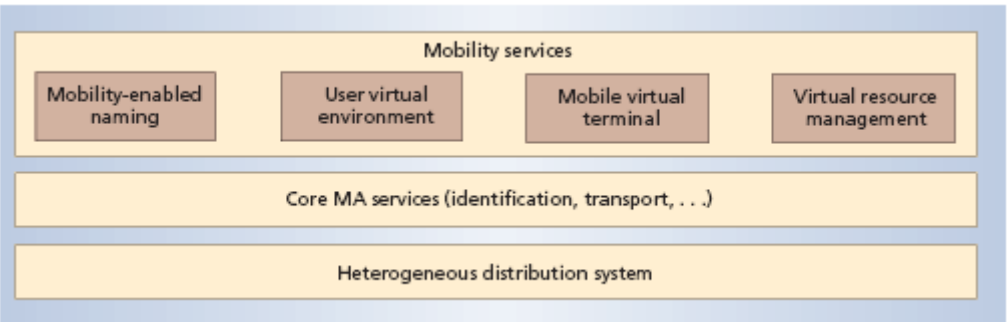
Um agente móvel é um processo que pode transportar o seu estado entre ambientes, deixando os seus dados intactos e ainda assim comportar-se de forma adequada ao novo ambiente.

O paradigma do Agente Móvel não necessita de conectividade contínua à rede pois as ligações são estabelecidas para colocar os agentes na rede.

Um Agente Móvel simplifica a personalização dinamica ao seguir os movimentos do utilizador e modelar serviços dependentes de preferências pessoais.

Nota: A investigação feita em Agentes Móveis centra-se na Mobilidade do Terminal

- Explicação da plataforma Middleware de Mobilidade Baseado em Agentes Móveis



Mobile naming services

> Atribuí um "Globally Unique Identifier" ("GUIDS") a cada Agente.

> Dados o GUID ser um identificador de baixo nível, os “naming services” atribuem um ou mais nomes a cada entidade com o intuito de facilitar a vida ao programador e utilizador final.

> Os “naming services” traduzem um nome a alto-nível num GUID, mas também mantém informação acerca da localização da entidade.

User virtual environment

> Este serviço permite ao utilizador ligar-se à Internet em diferentes locais e terminais, mantendo as suas configurações indicadas no respectivo perfil.

> Os utilizadores definem o seu perfil no primeiro registo e pode ser modificado a qualquer altura.

> O UVE replica e armazena os perfis em diversas localizações.

> Os perfis ficam globalmente disponíveis através do “Directory Service”.

> Quando um utilizador se desconecta, o UVE comunica ao middleware para este congelar o agente e este passa ao estado “descongelado” quando o utilizador se volta a conectar.

Mobile virtual terminal

> Suporta a migração de dispositivos móveis entre diferentes localizações, permitindo que o terminal continue a execução e preservando o estado das interacções com a rede.

> Requalifica as referências do terminal ao ligá-las a recursos equivalentes existentes na nova localização. Se esta ligação for impossível ou indesejada, o MVT mantém a referência aos actuais recursos.

> Suporta as operações não-interactivas efectuadas enquanto o terminal está desconectado.

> Explora a serialização do Agente Móvel para ordenar o estado da sessão do terminal num elemento de armazenamento estável e continuar a execução da informação recuperada.

Virtual resource management

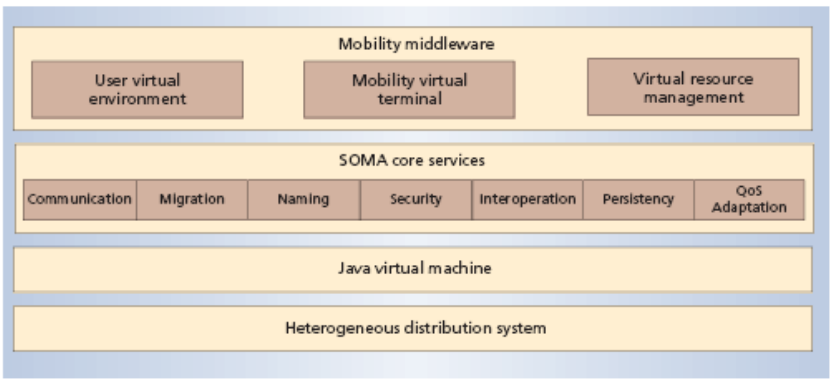
> Mantém informação acerca das propriedades e localização dos recursos e serviços disponíveis.

> Implementa as funções do lado do servidor para o estabelecimento de ligações dinâmicas entre terminais e recursos necessários.

> Simplifica a modificação e migração de recursos e serviços de sistema em tempo de execução, accionando operações complexas de gestão.

> Num cenário global, o VRM é solicitado para endereçar a heterogeneidade de recursos e serviços.

- Explicação da plataforma Middleware baseada em SOMA



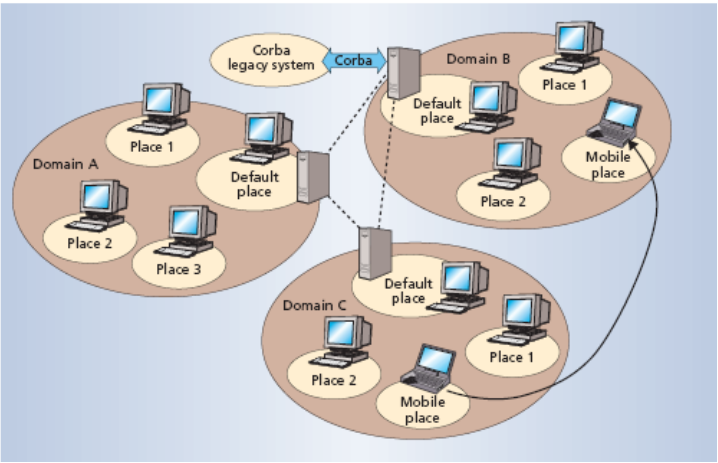
A framework SOMA é uma plataforma assente em Java e providencia uma infrastrutura de serviços em camadas para desenhar, implementar e correr aplicações de Internet baseadas em Agentes Móveis. O SOMA permite abstracção de localidade para descrever qualquer sistema interligado, desde intranet LANs até à Internet.

Cada nó é denominado de "place" que é o ambiente onde o agente se encontra em execução. Cada grupo de "places" é denominado "Domain", definindo assim o conceito de localidade. Em cada "domain" existe um "default place" que controla o roteamento e integração com componentes via Corba.

Permite a programadores e administradores de rede suspender a execução de agentes ao armazenarem o seu estado em disco (**serviço persistência**).

O serviço de "naming" do SOMA deriva de mecanismos de tratamento para localização de agentes e "places". O tratamento para qualquer agente ser seguido deve ser encontrado onde foi primeiramente criado.
Actualiza a "home" de cada agente na sua migração e coloca-as a quando da sua conecção ou desconecção.
Agentes e "places" SOMA têm o seu GUID independentemente da sua localização.
Integra um protocolo de descoberta e um "directory service" baseado em LDAP.

- Explicação do exemplo de aplicação do SOMA



Neste exemplo, um mobile place mirou do dominio C para o dominio B. O mobile place realça a abstracção da localização do place com funções especificas para reconfiguração aumotatica quando se alterare os dominios.

27 – The Jini Architecture for network-centric computing

Em que consiste o Jini?

-Sistema distribuído baseado no conceito de federação;
-Serviços descentralizados aos dispor de todos os utilizadores;
-Sistema escalável;
-JINI é JAVA-Centric(Desenvolvido em Java/requisitos só disponiveis em Java)
-Jini Enabler(Possibilidade do serviço migrar código para o cliente)

Nota: de ter em conta que o java tem propriedades altamente relevantes para a arquitectura do Jini.

1) compativel em redes heterogeneas
2) Mobile ObjectCode-Portabilidade;

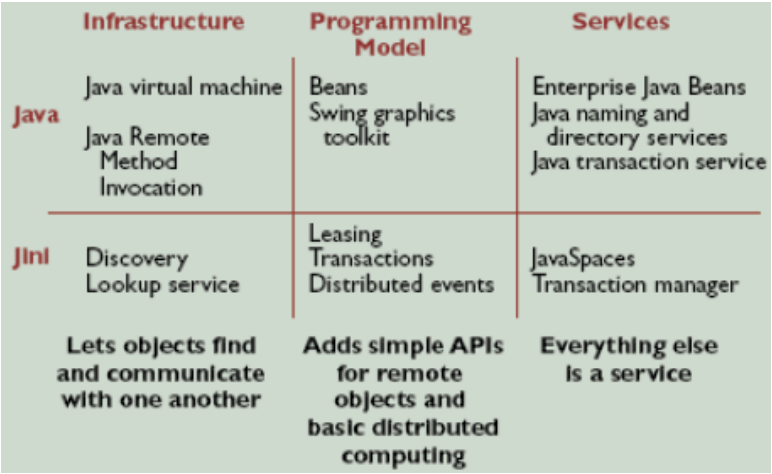
At a Glance	
Why use Jini Technology?	• Provides an environment for creating dynamically networked components, applications, and services that scale from the device to the enterprise • Offers an open development environment for creative collaboration through the Jini Community
Unique Qualities	• Code Mobility: Extends the Java programming model to the network; i.e., moves data and executables via a Java object over a network • Protocol agnostic: Provides the ultimate in design flexibility • Leasing: Enables network self-healing and self-configuration; i.e. improving fault tolerance
Unique Benefits	• Resiliency - Networks readily adapt to changes in the computing environment • Integration - Allows fast, easy incorporation of legacy, current, and future network components • Licensing - Jini network technology is available free of charge with an evergreen license

Problemas segurança resolvidos com bibliotecas Java
Permite efectuar load de código de num processo execução
3) Apenas 1 JVM na rede – devices sem JVM delegam funções para outra máquina.

Porque surgiu o JINI?
-Permitir aos programadores assumir algumas premissas como: redes fiáveis, redes homogéneas, rede seguras, etc;
-Tornar as redes distribuídas fáceis de gerir;
-Facilitar a inserção de novos serviços e *devices* na rede;

Principais Características e Objectivos do JINI
- Possibilitar a partilha transparente de serviços e recursos na rede;
- Fornecer aos utilizadores fácil acesso aos recursos e serviços da rede;
- Simplificar as tarefas de alteração e manutenção de serviços e *devices;*

Explicação da arquitectura do Jini



Infrastructure
-*DiscoveryProtocol*- Ligação de entidades àrede para descoberta de serviços;
-*LookupService*- Local onde estão registados os serviços e recursos de toda a rede;

ProgrammingModel
-Leasing – resolver problemas quanto à alocaçãode recursos em emsistemas distribuidos;
-Transactions- Suporta two-phase-commit;
-Distributedevents- Conjunto de interfaces para definir Modelos de Eventos Distribuidos

Explicação de Spontaneous Networking
-Característica garantida pela possibilidade de migrar código de modo seguro;
-Serviços e clientes podem entrar e sair da Federação a qualquer momento;
-Macro Tarefas
>Join da Federação;
>Registar Serviços;
>Pesquisar Serviços;

JoinFederation - Serviço envia pacote em multicaste espera que a entidade lookupservice responda. LookupService envia para o serviço o proxy local para o lookupservice.

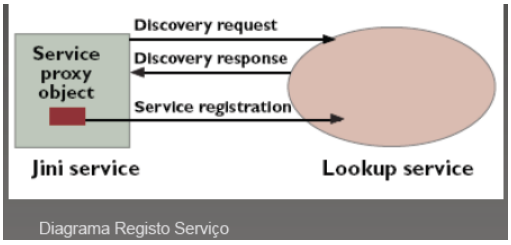
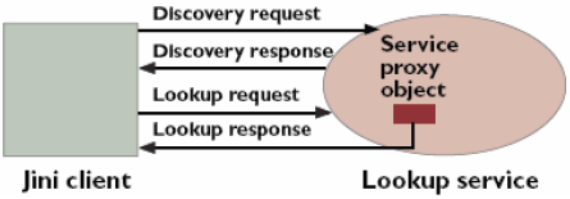


Diagrama Registo Serviço

Registo Serviço - Serviço envia seu proxy para os lookupserviceque responderam. Pode registar-se em vários.



Pesquisa Serviços - Clientes fazem multicastáprocura de respostas;>Pedem o serviço aos lookupservices

Jini permite a criação de Federações espontâneas, mas também permite que os serviços e clientes a deixem espontaneamente.
PROBLEMA: Pode ser destrutivo para os outros membros.
COMO RESOLVER? Leasing Model

Leasing Model
-Soluciona problema de quebra de ligação
-Cliente/Servidor pede determinado tempo de alocaçãode recursos;
- Tempo esgota:>Requisito nova alocação>Desliga-se

Explicação da federação Jini

- Constituida por:
  > Clientes;
  >Serviços:
  >LookupService
- Define regras de interacções entre nós de rede;
- Numero de regras mínima:
  >Quebra de ligação;
  >Interaçãoentre membros;
  >Ligação de membros árede;

Explicação de Jini Vs Disk Centric Computing

Disc-CentricComputing
> Obriga aplicações a serem compiladas individualmente para cada processador.

Jini
>Codigoétransferido entre máquinas em tempo de execução
>Código adapta-se somente a Java Virtual Machine

Quais as grandes vantagens do Jini?

- A arquitectura do Jini permite fazer upgrades à rede mantendo-a em funcionamento.
- Facilita ligação de dispositivos à rede

26 – Mobile agent security

- O que nos fala este paper?

Este paper fala-nos sobre medidas de segurança que tem de ser tomadas quando lidamos com sistemas de agentes e multi-agentes móveis.

- O que é um agente móvel?
Um agente móvel é uma classe particular de agentes que tem a habilidade de em runtime migrar de um host para outro aonde possa resumir a sua execução.
Esta tecnologia pode ser utilizada para reduzir o trafico da rede. Às vezes o agente também pode assumir o papel de outra entidade, como um humano ou uma organização.

- Explicação das caracteristicas dos agentes e multi-agentes importantes para segurança

situatedness – significa que o agente recebe input de um ambiente e que consegue executar acções que mudem o ambiente de alguama forma
autonomia – significa que um agente é capaz de actuar sem intervensão directa dos homens (ou outros agentes), e que tem controlo sobre as suas acções e estado interno.
flexivel – que se define com as seguintes propriedades:
- responsive: refere-se à abilidade de um agente perceba o seu ambiente e responder de forma oportuna a alterações que tenham ocorrido nele.
- proactivo: os agentes são capazes de exigir oportunismo, comportamento de sentido de objectivo e tomar a iniciativa quando apropriado.
- social: os agentes podem ser interactivos, quando apropriado, com outros agentes e humanos em ordema resolver os seus proprios problemas.

- **racionalidade** – é o assunto de que um agente não irá actuar de uma maneira que o previna de alcançar os seus objectivos e de tentar sempre alcaçar os seus objectivos.
- **veracidade** – nunca comunica falsa informação
- **benevolente** – o agente não pode ter conflictos que o forcem a transmitir falsa informação ou que afecte as suas acções de atinguir os seus objectivos.
- **mobilidade** – habilidade de um agente movel se deslocar entre redes e entre hosts para cumprir os seus objectivos.

Um multi-Agent é um sistema composto por multiplos agentes autonomos com as seguintes caracteristicas:
- cada agente não pode resolver o problema sozinho
- não existe nenhum sistema de controlo global
- a informação é descentralizada e a computação é assincrona

- Explicação das implicações de segurança

execução do agente – os agentes precisam de se executar em algum lado. Por isso, um agente deve ser impedido de lançar um ataque denial of service para consumir os recursos de um host.
autonomia – tem de se ter especial atençar, porque se é dado a autorização a uma agente para comprar ou vender coisas, ele não deverá ser possivel outra entidade forçar o agente a fazer commits em coisas que ele normalmente não fazia.
comunicação - para flexibilidade é importante, assim os agentes podem comunicar com outros agentes ou humanos. Por isso, estas comunicações têm de estar protegidas (confidencialidade, integridade dos dados, autenticação, disponibilidade e não repudiação).

mobilidade – os agentes necessitam de protecção de outros agentes e de outros hosts, da maneira como ele se executa. Igualmente, os hosts necessitam de se proteger de agentes. Mas o problemas mais dificil de resolver está em hosts para agentes. Visto que um agente está sobre o controlo da execução no host e em principio o host pode fazer o que quiser ao agente e ao seu codigo. Tipos de ataques: Observação codigo; manipulação de codigo; incorrecta execução de codigo; denial of execution; mascarar-se como outro host; manipular comunicação entre agentes.
racionalidade, veracidade e benevolencia – podem parecer a nivel de seguranças complicadas, mas o autor considera que são abstractas demais para as considerar praticas de seguranças. Mas podem ser tomadas medidas como logging do comportamento do agente para determinar se ele se tem portado bem. Porque não podemos assumir que o agente se vai portar sem bem e justo, ainda por mais em sistemas multi-agent.
identificação e autenticação – é sempre importante importante identificar agente para depois autentica-los, porque podem mesmo haver aplicações que necessitem de tal, por forma.
autorização e delegação – Autorização e delegação é um tema importante para sistemas de multi-agents. Não só para agentes que necessitam de ter os seus direitos para aceder à informação e outros recursos, em ordem de cumprir as suas tarefas, eles também podem agir em nome de uma pessoa, organização ou outros agentes, necessitando a transferencia de direitos de acesso entre diferentes entidades.

- Explicação de medidas de segurança
Protegendo agentes – Como foi referido o problema central está num host mal intencionado, são apresentadas soluções:

> acordos contractuais – utilização de contractos para determinar garantias na plataforma.

> hardware confiança – solução de alto custo, mas uma possibilidade seria a utilização de smartcards.

> nós de confiança – os agentes podem usar esses nós para migrar, visto confiarem neste ao contrario de um host desconhecido.

> agente coperativos – com a utilização de agentes cooperativos é quase impossivel corromper um agente porque os agentes trocam informação entre eles.

> rasto de execução – consegue-se determinar as modificações não autorizadas por uma agente, ao se registar todas as execuções desse agente na plataforma de agente.

> payload cifrado – utilização de cifra assimetrica, quando o agente pretender enviar resultados para o seu dono.

> geração de chave do ambiente – permite a um agente carregar informação cifrada do seu código e informação.

> assinatura indetectada – utilização de uma chave de assinatura ao agente, pode-se reduzir o dano causado por hosts mal intencionados

Protegendo plataforma agentes – Para proteger as plataformas de agentes de agentes mal intecionados existem as seguintes soluções:

> sandboxing e interpretação do código de forma segura – isola aplicações (no nosso caso agentes) em distinctos dominios forçados pelo software.

> Proof carrying code – necessita o autor do agente para formalmente provar que o agentes está de acordo com uma determinada plotica segurança.

> código assinado – ao fazer uma assinatura digital no agente (consegue-se alcaçar autenticidade e integridade).

> caminho histórias – permitir o host saber aonde o mobile agente se executou anteriormente.

Autoria:
Alunos de CM 2006/2007