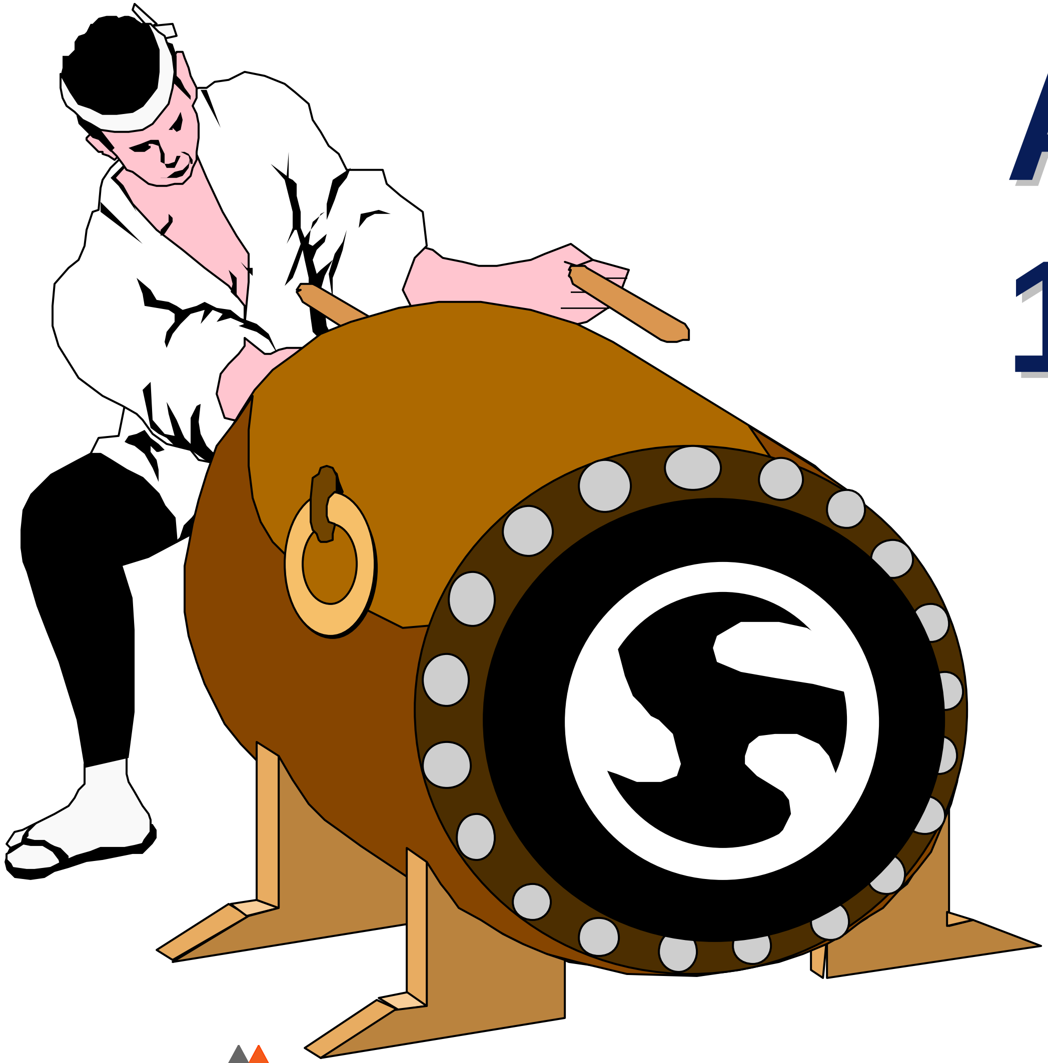


ANSA 1997

Brainstorm......

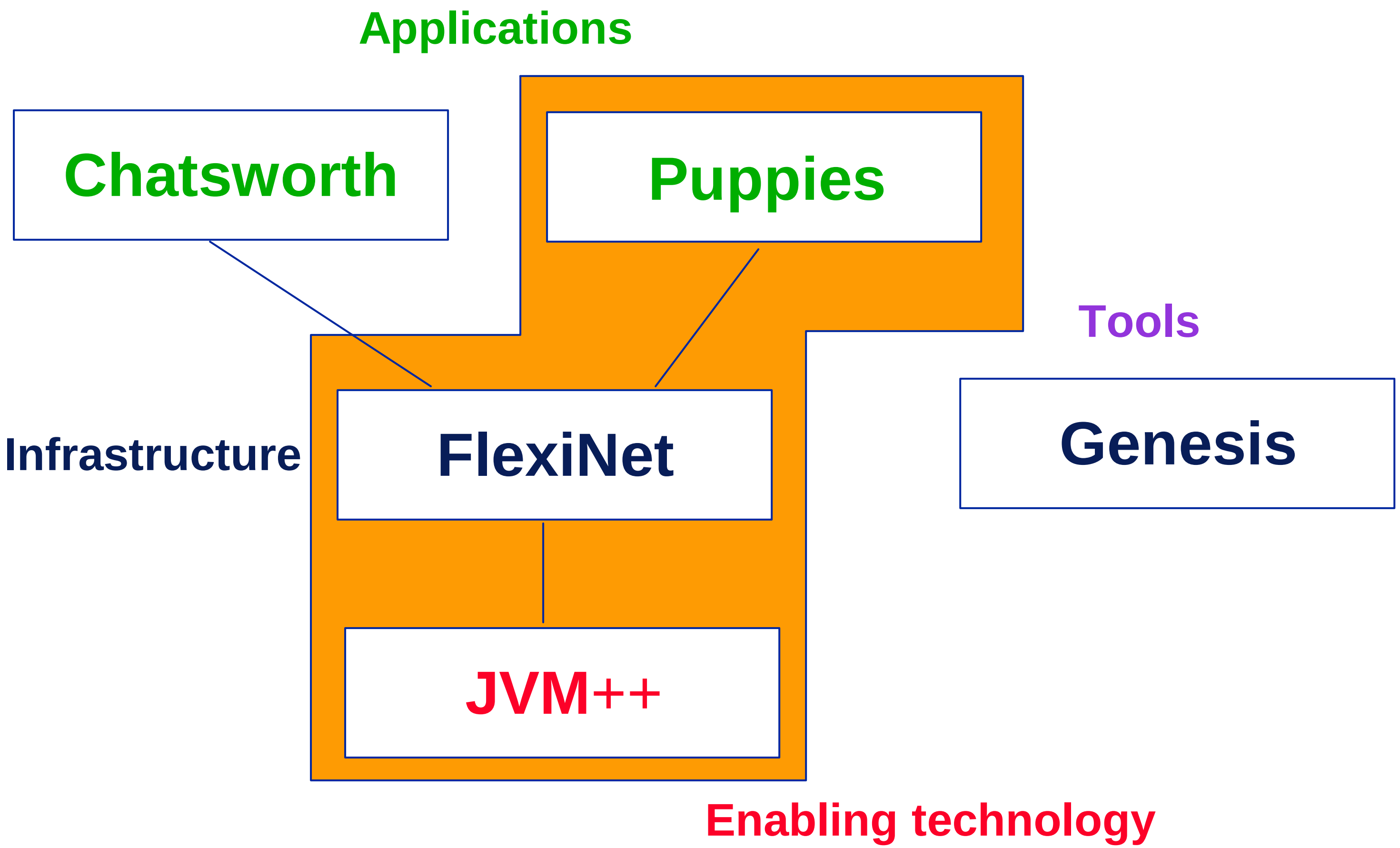
- **Reflective Java**
  - *dynamic infrastructure adaptation*
- **Chatsworth**
  - *networked game shows*
- **Genesis**
  - *design to deployment models*
- **Puppies**
  - *personal, mobile, information space*
- **FlexiNet**
  - *constantly evolving network*

Underpinning theme

- **Mobile code in global networks**
  - Java accelerates Network 2000 vision
  - revises key architectural assumptions
- ***Infrastructure independent code***
  - end of the monolithic OS
- ***Networks evolve constantly***
  - soft networks
- ***Network / service boundary dissolves***
  - the user is in charge

Revised Assumptions

- **WWW**
  - extensible, autonomous, active content, agents
- **Java**
  - small, clean, expressive, component-ware
- **Internet**
  - multicast, QoS, security, pricing
- **Telecoms**
  - broadband, federated, multi-service
- **Network computing**
  - ubiquitous, mobile, appliances



Applications

Chatsworth

Puppies

Tools

Infrastructure

FlexiNet

Genesis

JVM++

Enabling technology

Programme

- **Reflective Java** (Already started)
- **(Wrap up DIMMA)**

- **FlexiNet**
  - what comes after TINA
  - Telecoms, Internet convergence
- **Puppies**
  - what comes after WWW
  - “follow me” network desktop

(Underway)

What about.....

- **Chatsworth**
  - *needs a different consortium*
  - *alternative view to DAVIC*
  - *who wants to join in?*
- **Genesis**
  - *desirable*
  - *challenging*
  - *needs a context*
    - *APM's command and control practice*

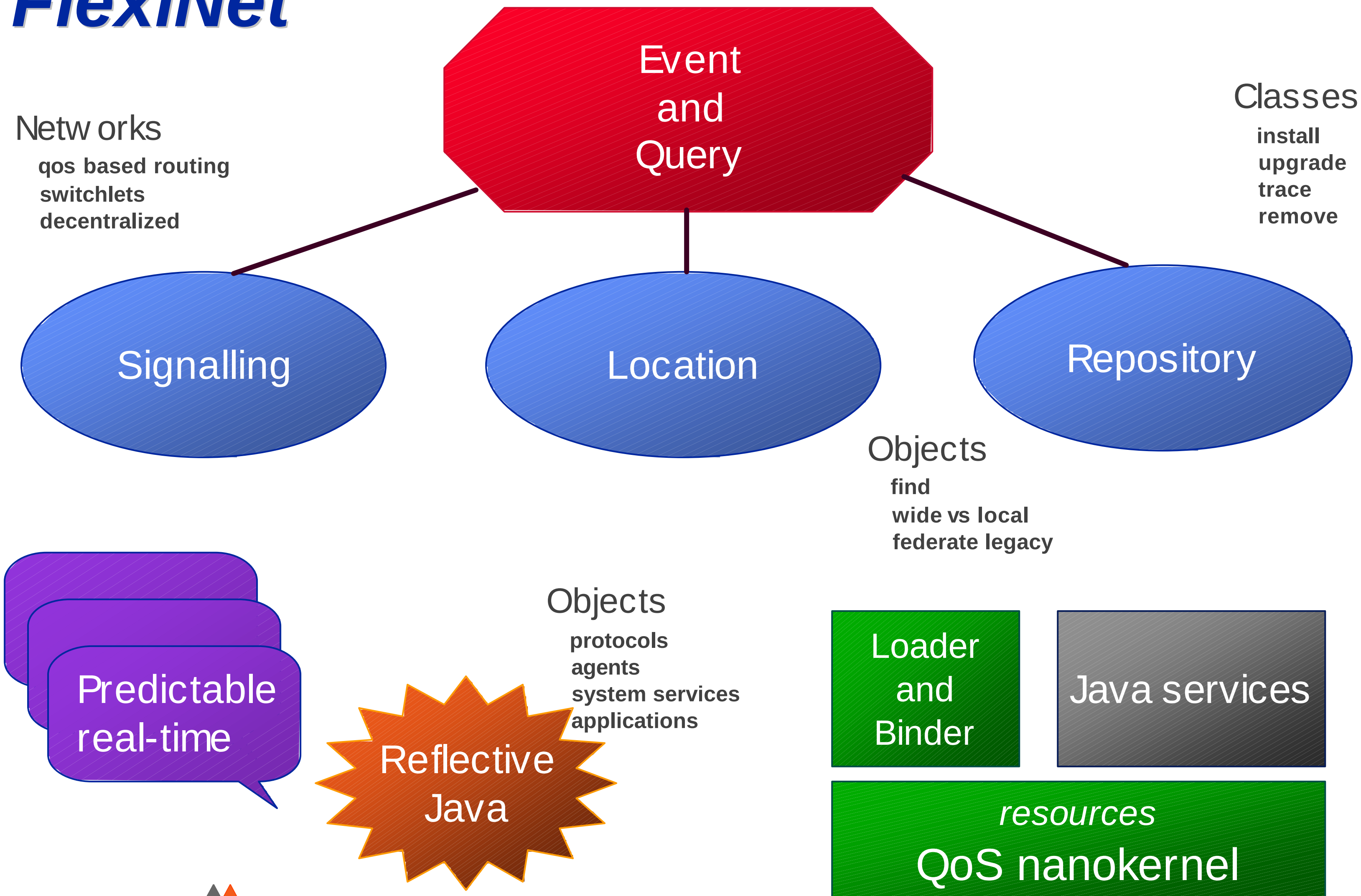
Reflective Java (Java++)

- **Configure non-functional capabilities of applications without changing source code**
  - Java = portability across h/w but not infrastructure
- ***MetaObjects*** **refine** ***virtual machine***
  - declarative *tools* to build *MetaClasses*
- ***Write an application once, run it anytime, anywhere, in any environment, with any "-ability"***
- ***Timescales***
  - Sep 96: core
  - Nov 96: persistence / transaction demo
  - Dec 96: consolidated release

FlexiNet

- **Can we build a completely soft, dynamically upgradeable network?**
  - *allow update of any component at any time*
  - *zero client administration*
  - *free choice of components*
  - *flexible configurations*
    - plug-in objects
    - co-existing different versions
    - federation
  - *automated management*

FlexiNet



Event and Query

Signalling

Location

Repository

Networks
- qos based routing
- switchlets
- decentralized

Classes
- install
- upgrade
- trace
- remove

Objects
- find
- wide vs local
- federate legacy

Predictable real-time

Reflective Java

Objects
- protocols
- agents
- system services
- applications

Loader and Binder

Java services

resources

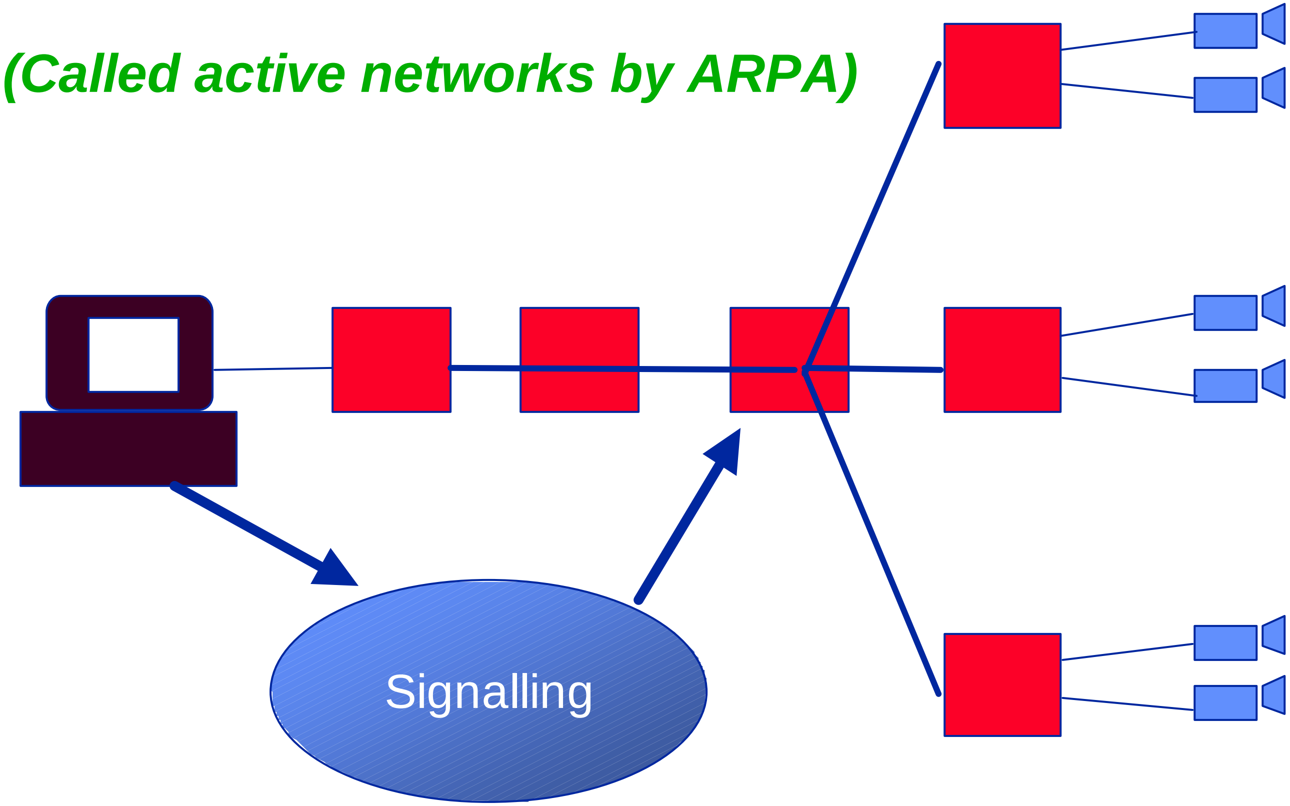
QoS nanokernel

Services

- **Event and database**
  - filter events
  - query state
  - fire on event + state
  - distributed ODMG model
- **Signalling**
  - outboard from switch
  - driven by QoS
  - user-defined
  - scaleable
  - federates
- **Repository**
  - applet server
  - software control centre
  - trading
  - adaptation
- **Location**
  - tracking mobile objects
  - decouple user names from system names
  - scaling issues
  - performance issues

User defined signalling example

(Called active networks by ARPA)



Infrastructure

- **Predictable Java**
  - visible VM resources
  - interface to kernel
  - synchronous Java
  - resource policy meta-objects
- **Platform**
  - minimum core
  - resources under application control
  - binder and loader to build infrastructure as required
  - Java services (e.g. JDBC for legacy interworking)

- Is Java suitable for distributed applications?
- What has to be pre-agreed?
- Does object mobility provide upgrade on the fly?
- How to add real-time predictability to Java?
- Costs/benefits of outboard connection management?
- How to name and track billions of mobile objects world-wide?
- How much management can be automated?

INPUTS

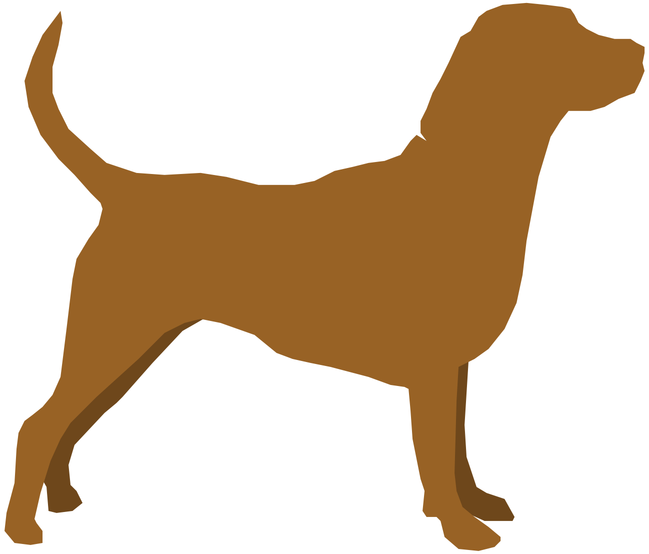
- **DCAN (CUCL, Nemesys)**
  - Outboard connection management
- **PEGASUS (CUCL, SICS, Glasgow, Twente)**
  - QoS-oriented nano-kernel
  - ATM & IP6 integration
- **OPERA (CUCL, ICL, Nortel)**
  - events and database
- **BROADCAST WG (Newcastle, INRIA, Bologna, EPFL, ...)**
  - Dependability
- **RETINA (Euro PNO labs, Siemens, HP, Alcatel)**
  - Yardstick for comparison
- **ARPA active networks (MIT)**
- **DIVA (Newcastle)**
  - reflection, dependability

Route

- **Resource management**
  - *experiment in DIMMA*
- **Binding**
  - *converge RETINA and DCAN in DIMMA*
- **Switching**
  - *leverage DCAN, simulate in Java*
- **Infrastructure**
  - *Java personality over DIMMA Object Nucleus*
  - *replace DON by Java framework*
  - *migrate to Nemesis kernel*
- **Events, Repository, Location**
  - *new!*

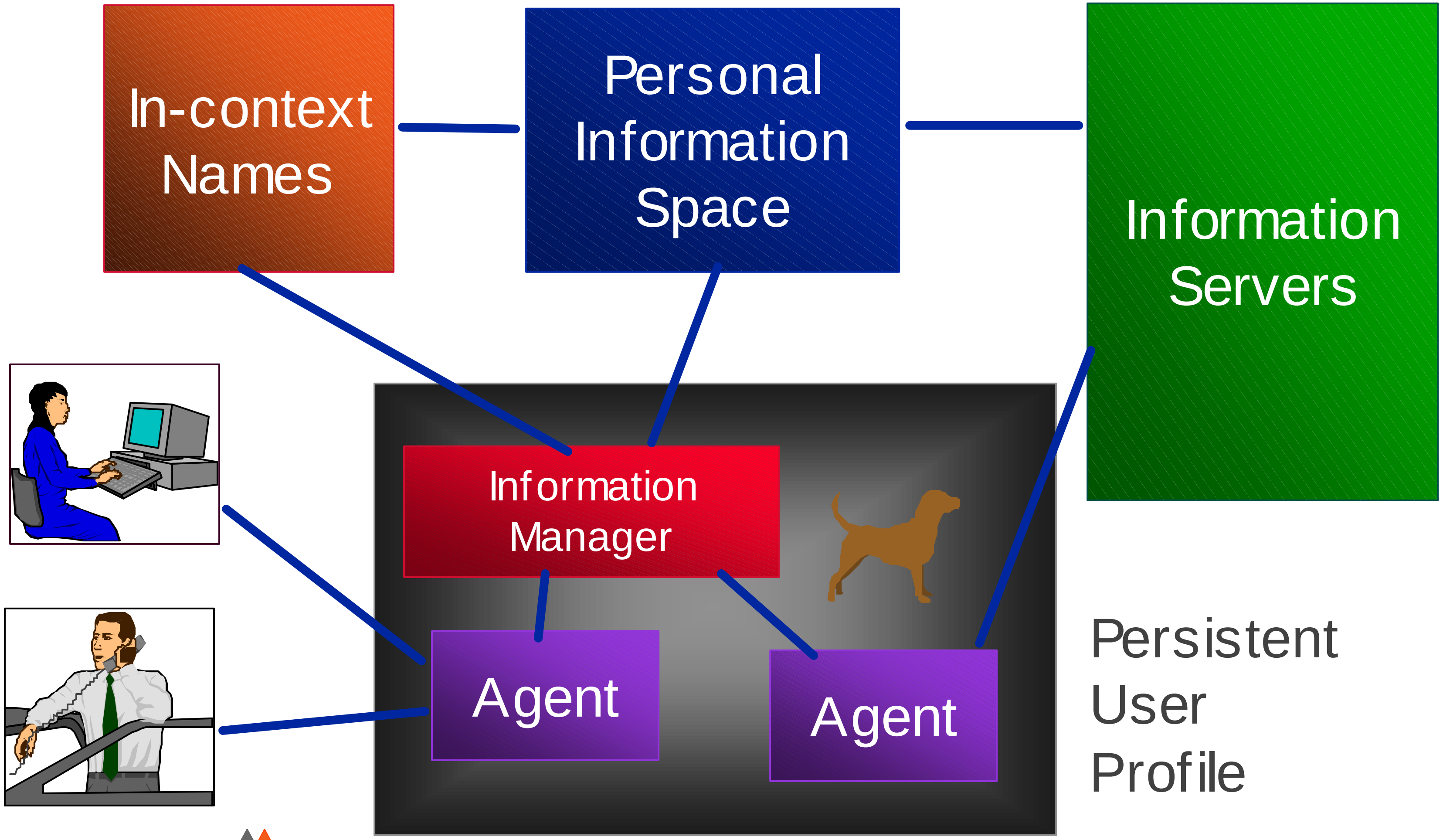
Puppies

- **A Personal information Space**
  - network home for user's stuff
  - managed for ease of use



- **Accessible by the mobile user**
  - "follow me" computing
  - information at hand
- **Accessible via different access points**
  - network computers, laptops
  - appliances
  - phones

Puppies



Persistent
User
Profile

Components

- In-context names
  - location independent
  - my names
- Persistent user profile
  - my links
  - agent policies
  - caching / moving policy
- Information manager
  - agent's access point
  - negotiates adaptations
- User agents
  - interface functions
  - searching / organizing
- Server agents
  - watching
  - reporting
  - archiving
- Information servers
  - WWW
  - mail
  - phone, fax

- What infrastructure do agents need?
- Can we do context-sensitive naming?
- How can we share puppies between people?
- How much access integration is possible?
- What indexing information do we need from information suppliers?
- What are effective caching strategies?
- Can we use asynchrony to mask low bandwidth?

INPUTS

- **Microcosm (Southampton)**
  - Link databases
- **Information Mesh (MIT)**
  - Object naming
- **Harvest (Colorado)**
  - Automatic meta-data distribution
- **IETF/W3C**
  - URNs, URLS
- **CODA (CMU)**
  - caching

Effort / Timescales (provisional)

	Q1	Q2	Q3	Q4	Q5....
1	Switching as a distributed application		Dependability / performance investigation		
2					
3					
4			Dynamic reconfiguration		Extension to IP active nets
5	Repository	Nanokernel			
6					
7	Location				Agents
8		Events			
9	Predictable Java			In-context names	
10					
11	PUP (email)	Voice adapter	Generic user profile and user manager		Adaptation techniques
12					

Wrap Up

- Plan
  - Reflective Java
  - Finalise DIMMA
  - Puppies
  - FlexiNet
- TC
  - Research questions, start points, approach
  - Customers
- MC
  - Priorities