A New Compositional Adaptation Technique for Pervasive Computing

Cho-Li Wang (王卓立)
The Systems Research Group
Department of Computer Science
The University of Hong Kong

Oct. 26, 2005 at HHME2005, Kunmin

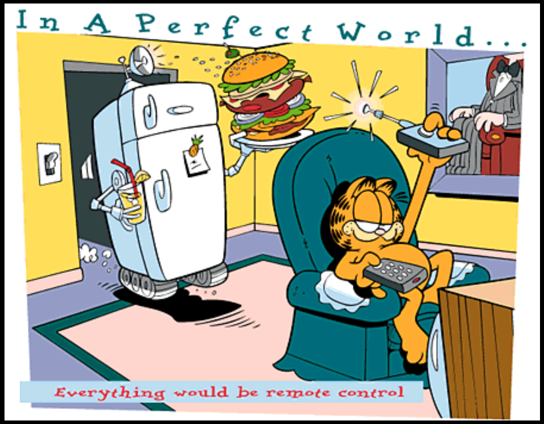
Outline

- Origin (a “system” viewpoint)
- Representative Projects
- Retrospection
- Future Blueprint
- Sparkle Legendary
- Conclusion

Origin

Mark Weiser's 1991

- *Pervasive Computing: An environment saturated with computing and communication capability, yet so gracefully integrated with users that it becomes a "technology that disappears".*
  -- Computing Anytime, Anywhere



Smart Meeting Room

Smart Office

Smart Home

What’s cool

- Extended scale of **space**
  - – The halo of computation feels free to move, and to span any number and type of devices
- Extended scale of **time**
  - – 7*24 hrs Schedule, Life-long
- Extended **application domain**
  - – Healthcare, daily activities, yet to imagine…

Dynamic Context

+

High Mobility

Representatives

- **Aura [CMU] :** ***Distraction-free computing***
  - User attention is costly
  - Abstraction: A “Task” = Multiple Applications.
  - *Task-driven Adaptation* : (re)configuring environment’s resources based on user intent and preferences before a task starts
- **Oxygen [MIT] :** ***Goal-oriented programming***
  - Goal is “beyond task” !
  - Core: *Goals* (policies) + *Pebbles* (implementation)
  - Higher-level adaptation : Revision of *goal tree*

Representatives

- **Gaia [UIUC] : *Programming over smart space***
  - An OS-like infrastructure to abstract the smart space
  - *Structural adaptation + 1st order predicates* (“&”, “||”, “-”..).
  - Intra- and Inter-space application-level mobility (same task)
- **One.world [UW] : *Programming for change***
  - Very dynamic environment
  - Virtualization of smart space (Virtual Machine)
  - “Environment” (application) = *Tuples* (data) + functional components
  - Control model : Discovery Service + Event-driven
  - Mobility support: very easy (checkpointing + migration)

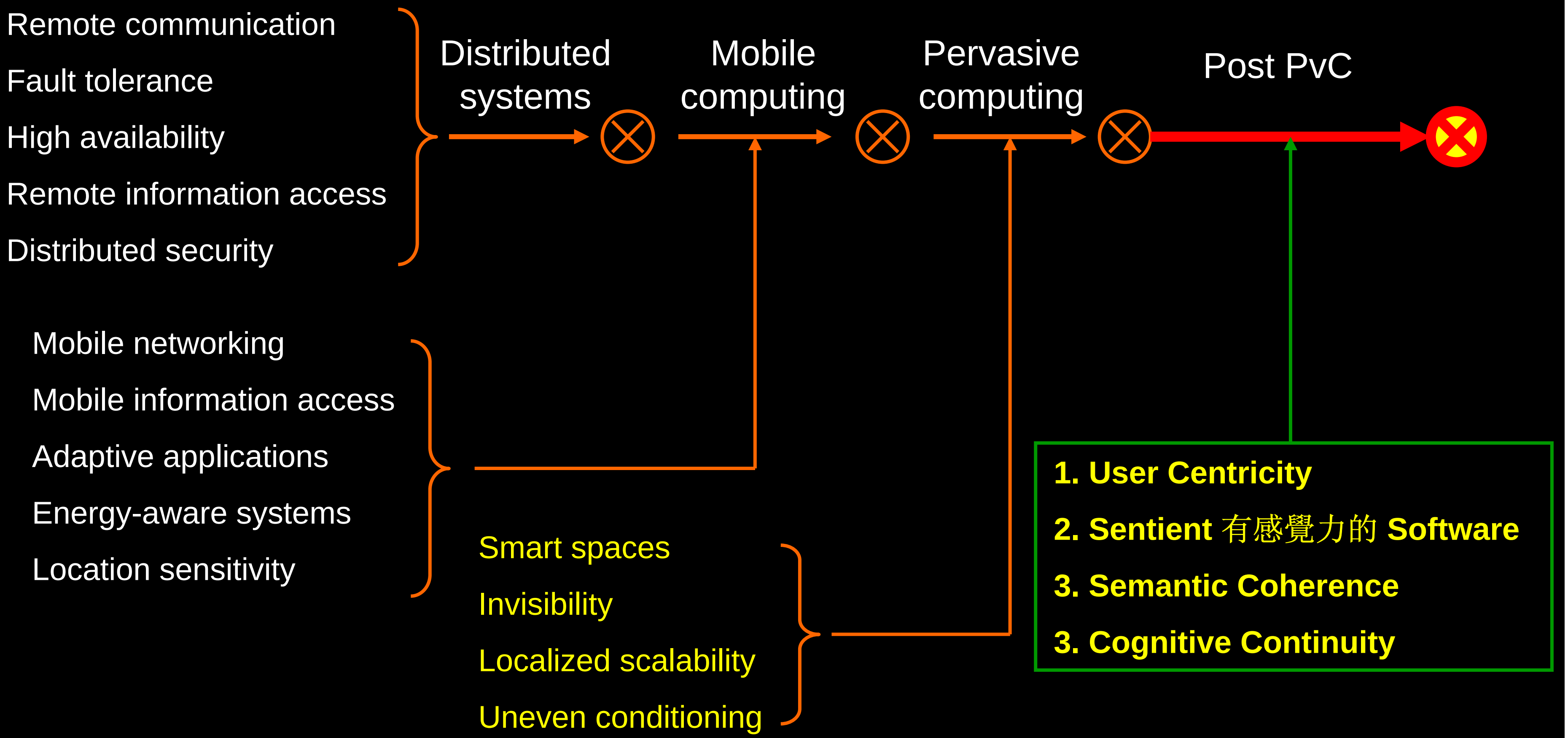
Retrospection

- Closed spaces
  - Small, familiar and well-controlled working space
  - Limited users, applications, context sources.
- Limited context usage
  - Limited context types (location, time, user identity, etc.). A small number of rules
  - Prediction of user intention is less exploited
  - Research in context acquisition/provision has just started
- Lack of sophisticated adaptation
  - Restricted adaptation strategies
  - Mostly deals with resources' changes
  - Decision is made in a straight-forward way

Retrospection

- Lack of flexible software architecture
  - The notion of “application” still remains
- Lack of high-level mobility support
  - User-perceivable continuity
- Lack of sparkling applications
  - Extension of legacy application
  - Utilization of office application
  - Where is the killer application?

The Pervasive Expedition



Remote communication

Fault tolerance

High availability

Remote information access

Distributed security

Distributed systems

Mobile computing

Pervasive computing

Post PvC

Mobile networking

Mobile information access

Adaptive applications

Energy-aware systems

Location sensitivity

Smart spaces

Invisibility

Localized scalability

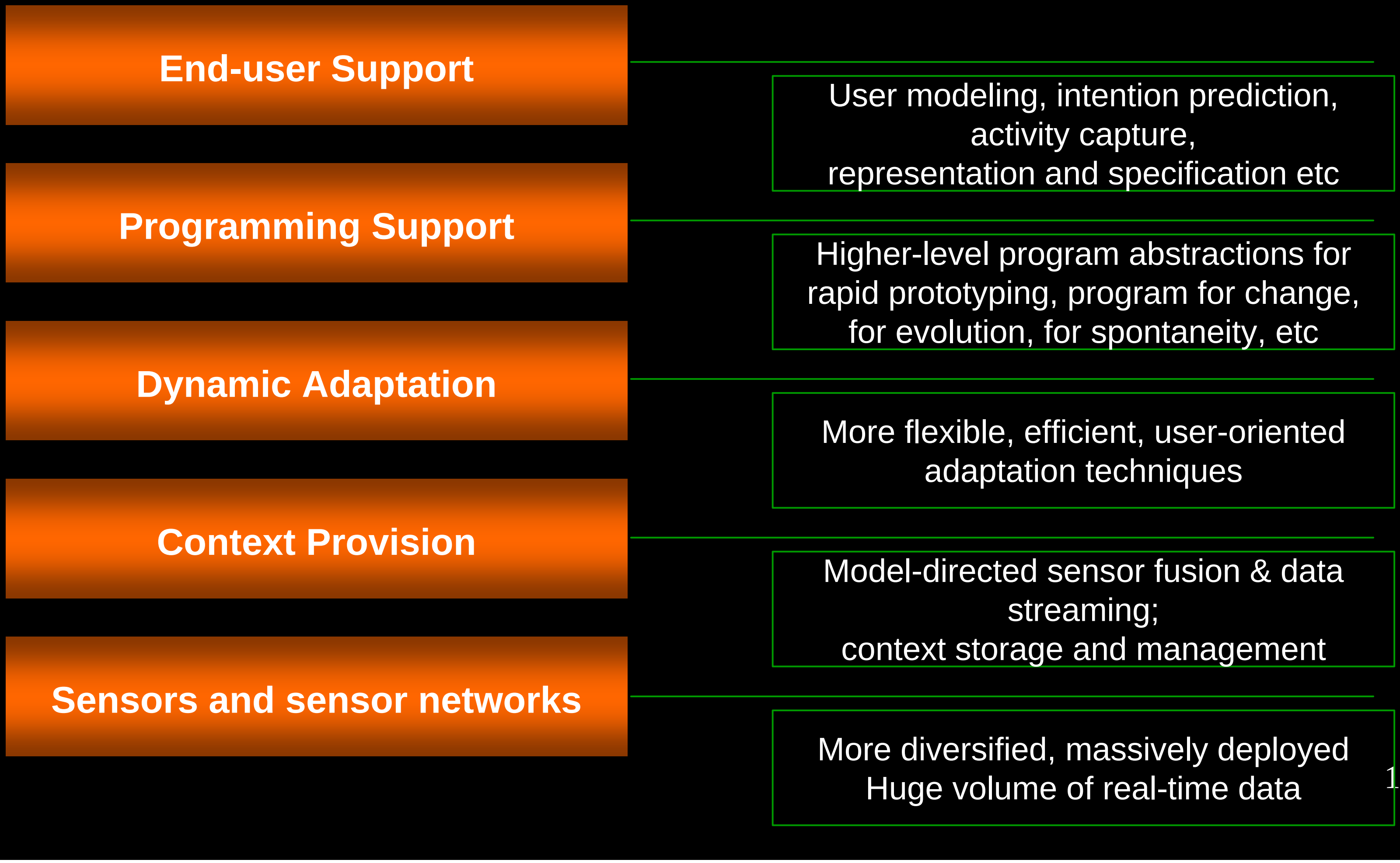
Uneven conditioning

1. **User Centricity**
2. **Sentient 有感覺力的 Software**
3. **Semantic Coherence**
3. **Cognitive Continuity**

M. Satyanarayanan [CMU, 1991]

Post-PvC Architecture

SPARKLE



End-user Support

User modeling, intention prediction, activity capture, representation and specification etc

Programming Support

Higher-level program abstractions for rapid prototyping, program for change, for evolution, for spontaneity, etc

Dynamic Adaptation

More flexible, efficient, user-oriented adaptation techniques

Context Provision

Model-directed sensor fusion & data streaming; context storage and management

Sensors and sensor networks

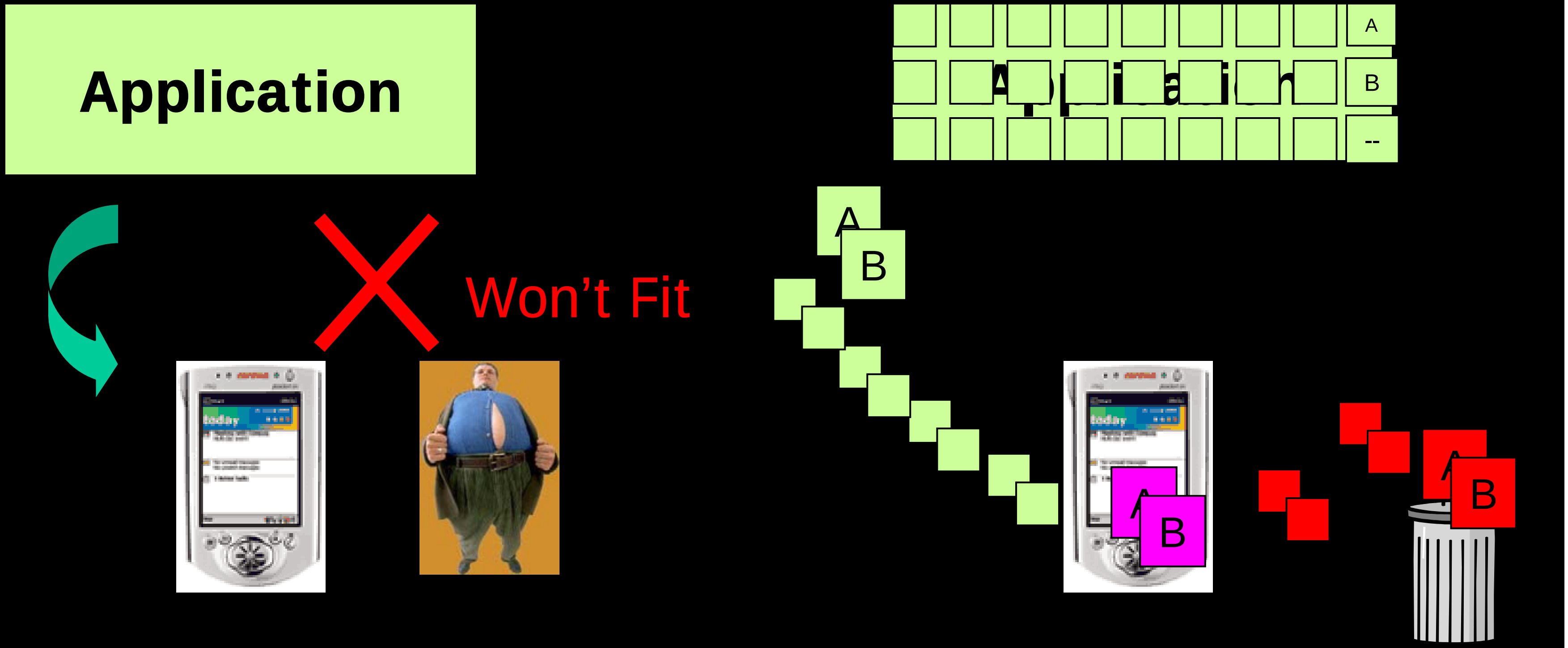
More diversified, massively deployed Huge volume of real-time data

Sparkle Legendary

- Sparkle I – Functionality adaptation
- Sparkle II – Semantic adaptation
- Sparkle III – Deep awareness

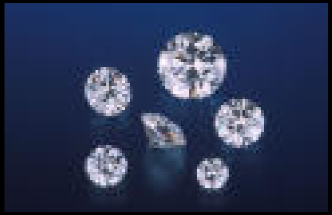
Sparkle I : Functionality adaptation

- Dynamic Software Architecture for Pervasive Computing – "Computing in Small"
- Resource-aware adaptation: Anytime, anywhere, <u>any device,</u> and <u>any application</u>



Applications distributed as monolithic blocks

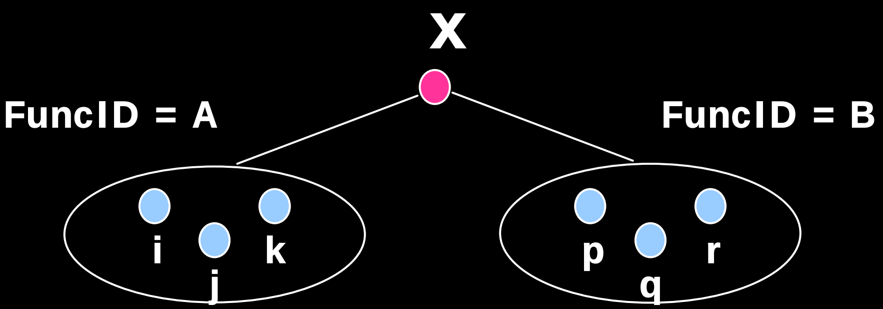
Facet Model



“Facet” = code (Java)+ shadow (XML)

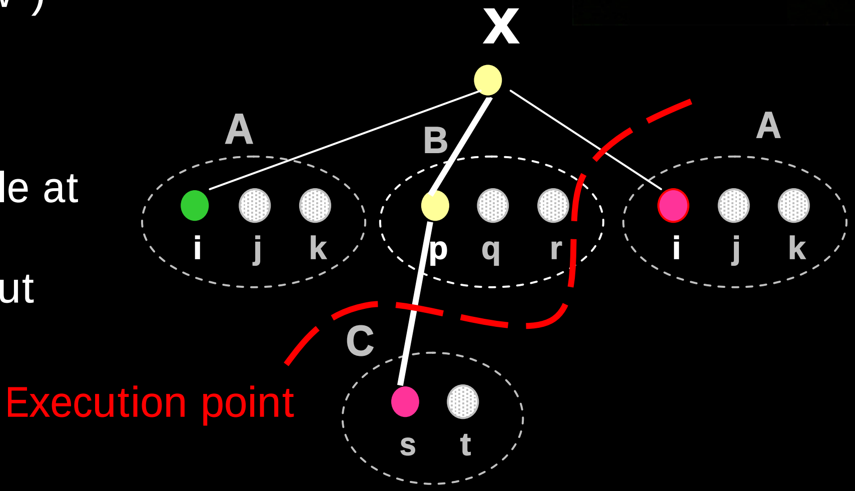
- Pure functional units (described by “Shadow”)
- Downloaded to client devices on demand
- Implement **single functionality**: single publicly callable method
- **Stateless**: Makes it throwable & replaceable at run-time

Shadow: specifies properties of the facet: Input and output specification, Resource requirements: memory, processing, etc



Facet Dependency Graph

- Facets may call upon other facets to achieve their functionality
- May have more than one facet fulfilling the functionality (e.g., i,j, k for A)



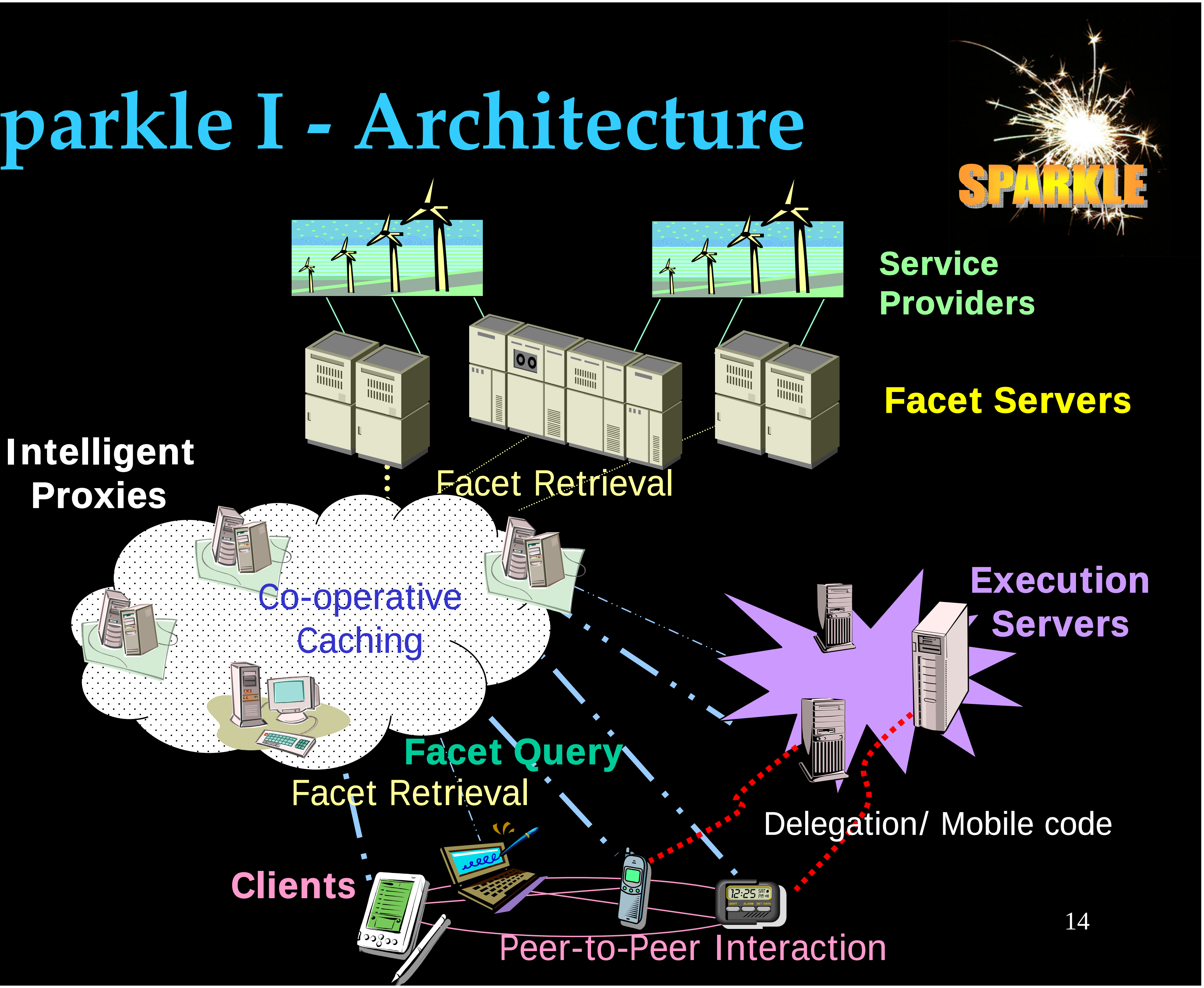
During execution, facets which are no longer active can be thrown

● Inactive Facet -already executed completely

● Facet which Has not yet been Brought in/loaded

● Active Facet - currently running

Sparkle I - Architecture

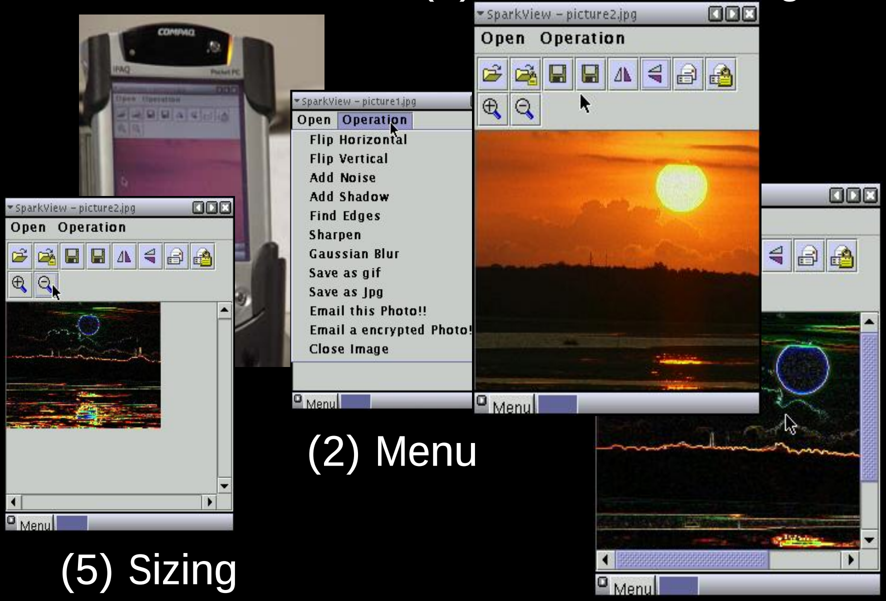


Sparkle I
(Image Viewer)

Developing a real-world application utilizing the facet model

(1) Image Viewer

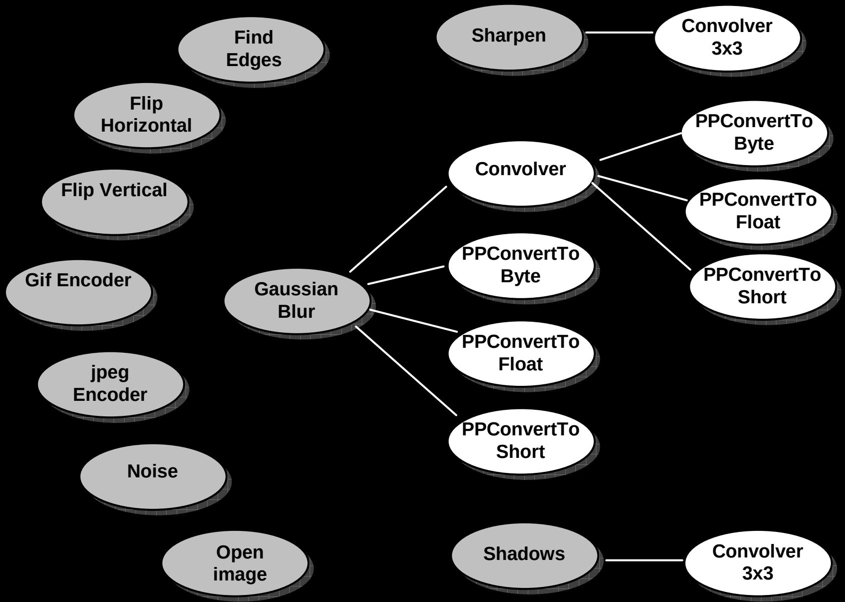
(3) Image rendering



(2) Menu

(5) Sizing

(4) Negative



Find Edges

Flip Horizontal

Flip Vertical

Gif Encoder

jpeg Encoder

Noise

Open image

Gaussian Blur

Convolver

PPConvertTo Byte

PPConvertTo Float

PPConvertTo Short

Sharpen — Convolver 3x3

Convolver — PPConvertTo Byte, PPConvertTo Float, PPConvertTo Short

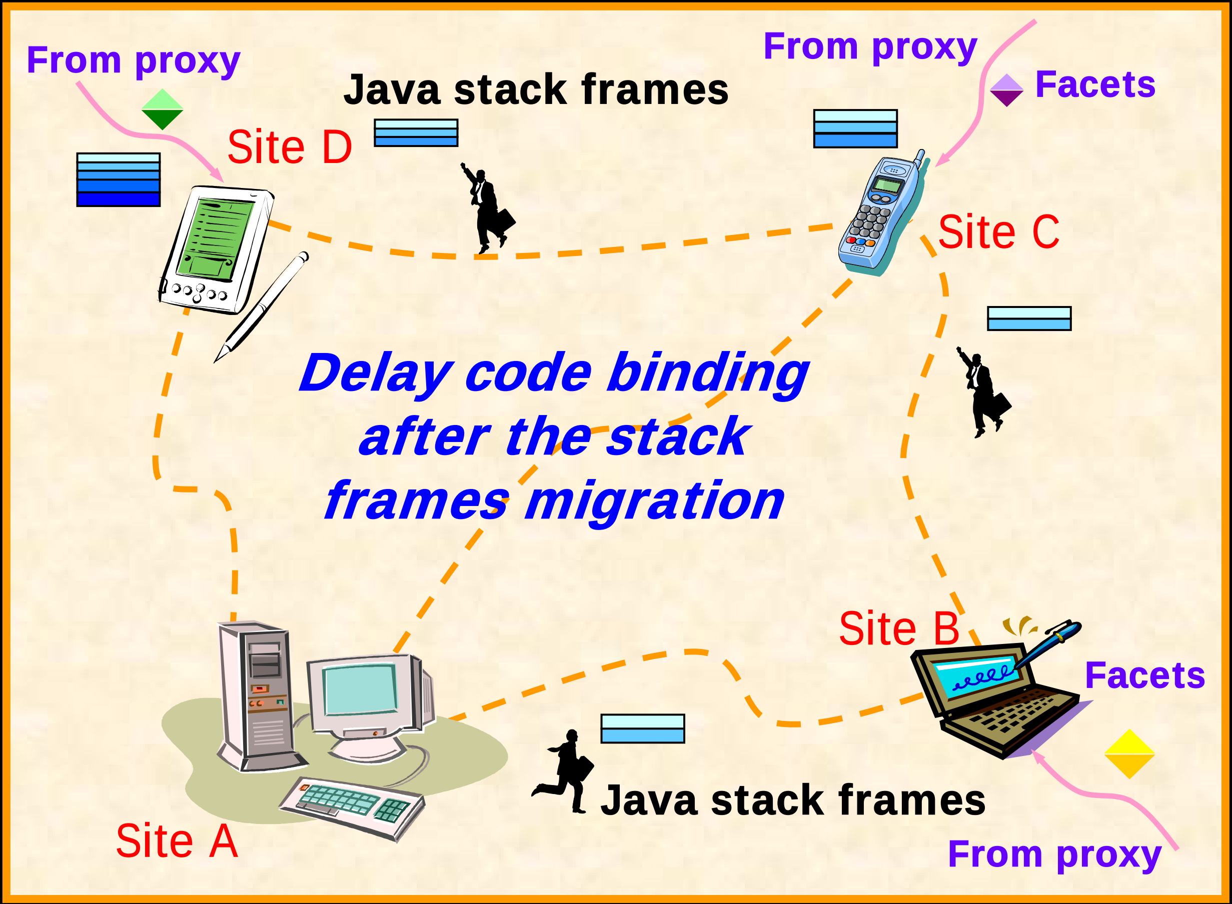
Shadows — Convolver 3x3

N. Belaramani, C.L. Wang and F. Lau, ``Dynamic Component Composition for Functionality Adaptation in Pervasive Environments,'' *The 9th Intl. Workshop on Future Trends of Distributed Computing Systems* (FTDCS2003), May 2003.

V. Kwan, F. Lau, and C.L. Wang, "Functionality Adaptation: A Context-Aware Service Code Adaptation for Pervasive Computing Environments", Web Intelligence 2003.

Sparkle I:
Mobility Support

Combining *code-on-demand* (COD) and *state-on-demand* (SOD) for distributed execution

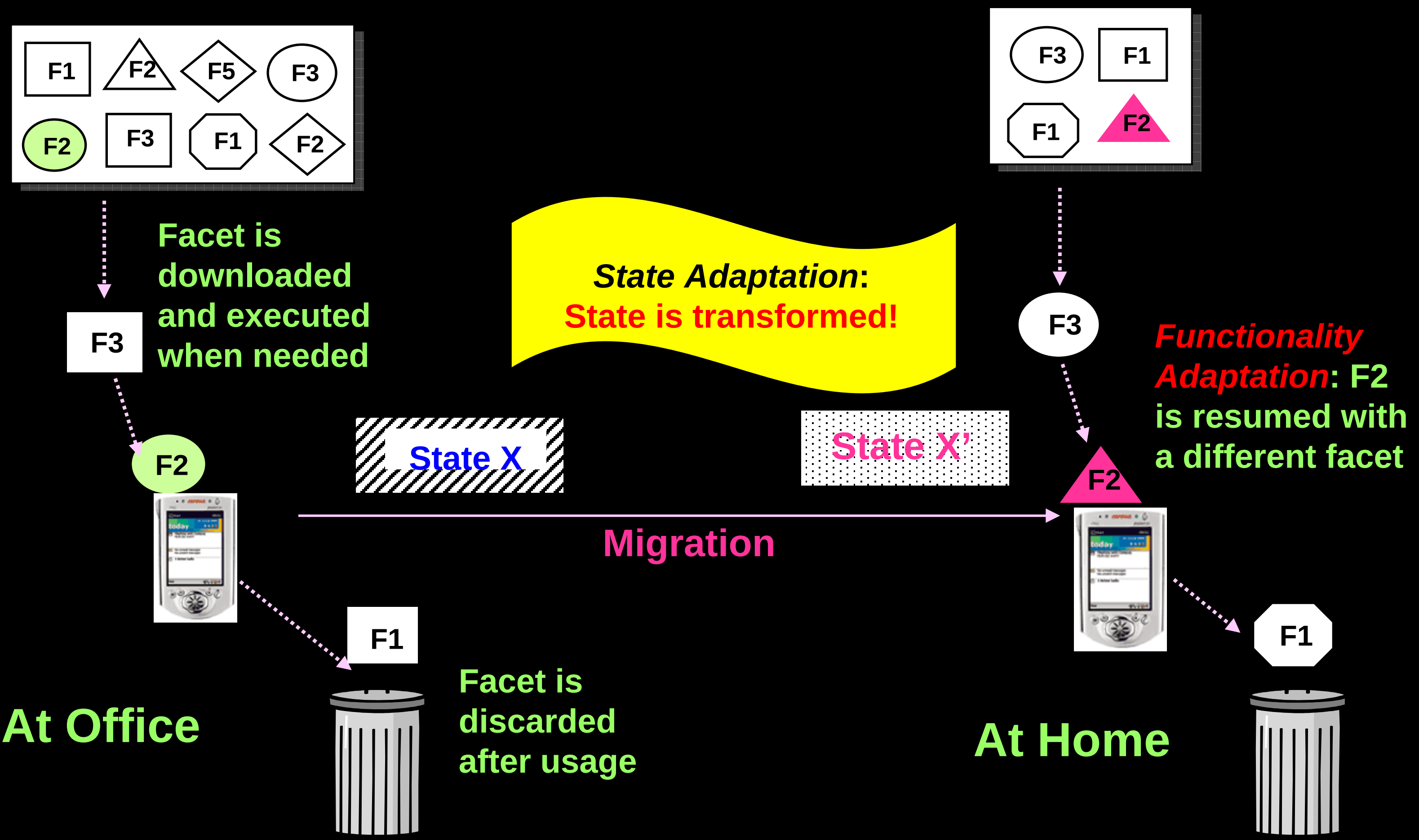


Delay code binding after the stack frames migration

Y. Chow, W. Zhu, C.L. Wang, F. Lau, ``The State-On-Demand Execution for Adaptive Component-based Mobile Agent Systems," ICPADS 2004.

Sparkle II :

Semantic adaptation for *Cognitive Continuity*



Facet is downloaded and executed when needed

***State Adaptation*:**
State is transformed!

***Functionality Adaptation*: F2 is resumed with a different facet**

Migration

Facet is discarded after usage

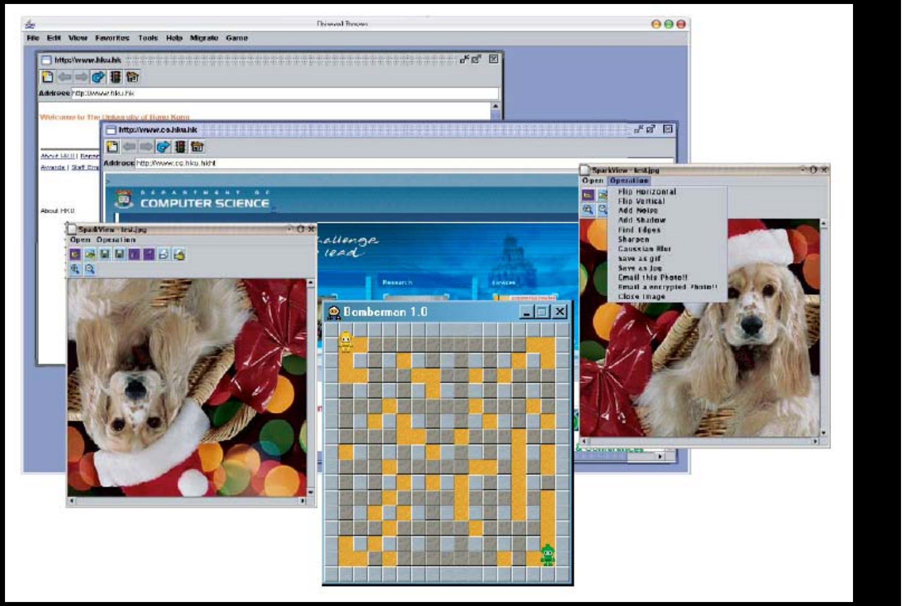
At Office

At Home

Sparkle II :

Universal Browser (UB)

- The UB targets “browsing whatever you want”. The special graphical user interface allows users to dynamically retrieve the functionalities they want, such as playing games, editing photos etc. State is captured, migrated, transformed, and application is then rebuilt.



Pauline P. L. Siu, C. L. Wang, and F. C. M. Lau, ``**Context-aware State Management for Ubiquitous Applications**," EUC2004

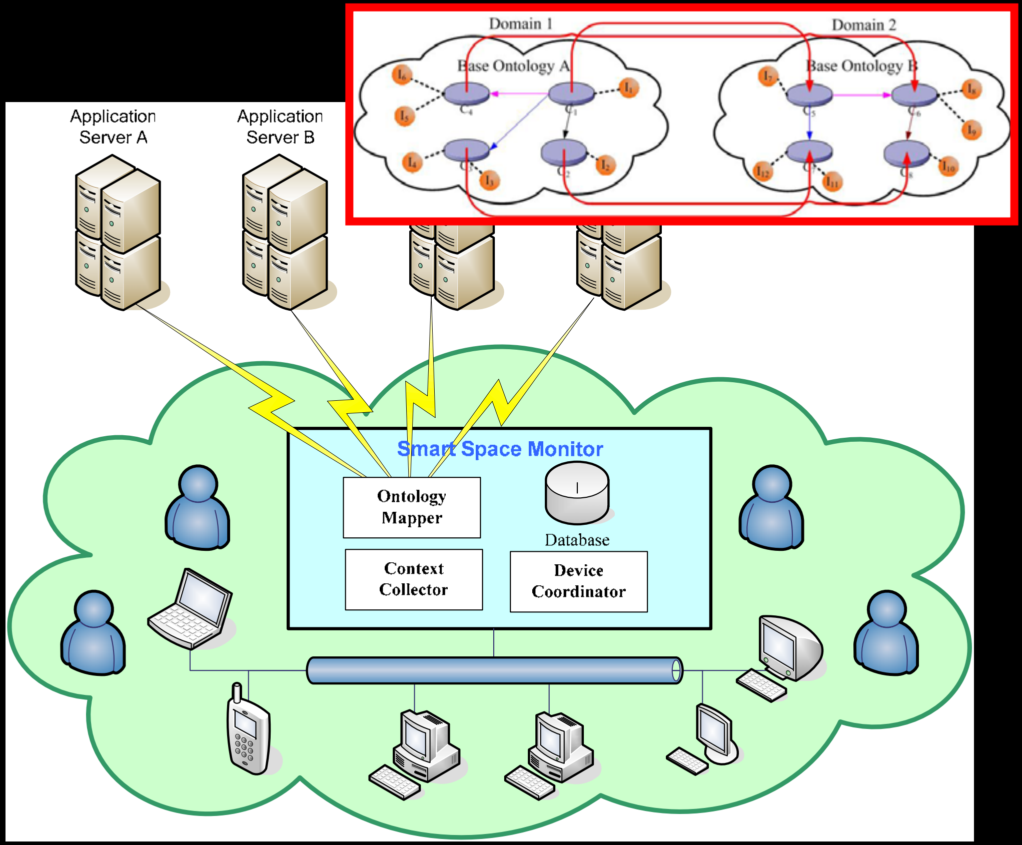
Sparkle III:
Towards Deep Awareness

- **Ontology-based Context Server**
  - Ontology mapping for ***Semantic Coherence***
- **Sentient software**
  - Smart Instant Messenger (ongoing)

Ontology-based Context Server

Ontology Mapping

- Domain ontology
  - Smart space context, resources, activities done.
  - One in each smart space
  - Static
- Application ontology
  - Device configuration, application parameters, service descriptions
- User ontology
  - High level: User identify, social status, user preferences/intentions
  - Very dynamic



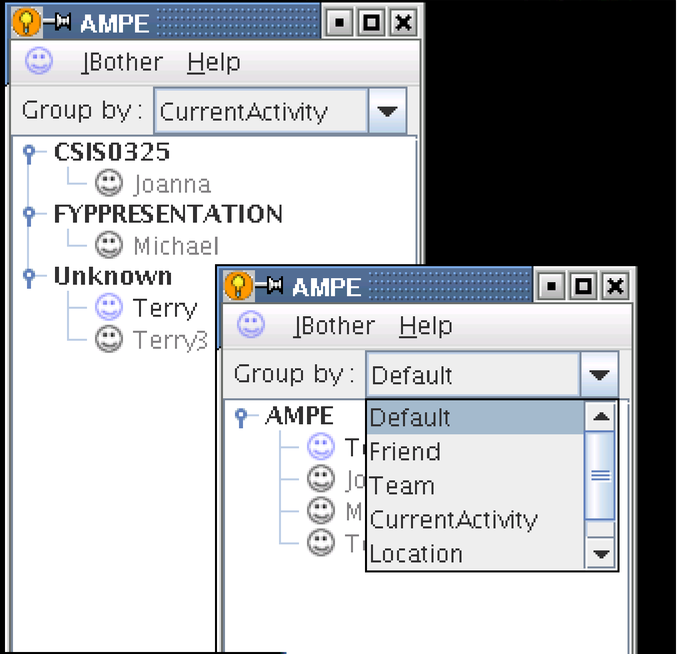
Laurel C. Y. Kong, C. L. Wang, and F. C. M. Lau, ``**Ontology Mapping in Pervasive Computing Environment**," EUC 2004.

Smart Instant Messenger

SPARKLE

- Main Features:
  - Dynamic buddy grouping: Creating groups base on the context information
  - Adaptive presence update: inference of current activities and prediction of future events
  - Resource buddy: Modeling each resource as buddy can allow quick access through the unified contact list interface

- A rule-based inference engine (Jena) for deriving useful context information



AMPE

JBother Help

Group by : CurrentActivity

CSIS0325
- Joanna

FYPPRESENTATION
- Michael

Unknown
- Terry
- Terry3

AMPE

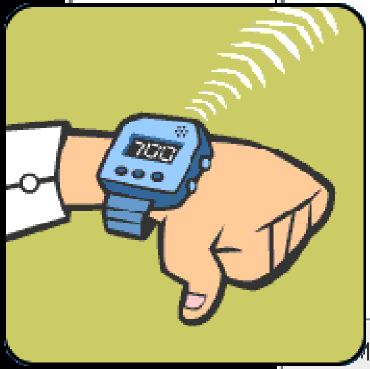
JBother Help

Group by : Default

AMPE
- T
- Jo
- M
- T

Default
Friend
Team
CurrentActivity
Location

My Location :



Conclusion (Our View)

- ``*Technology that disappears*” is hard to achieve, but
  - human intention is more predictable
  - A short step could make a great impact
  - Computer won’t disappear, but in what form ?

- ***Sentient software*** is hard to develop, but techniques are all there:
  - Aspect-oriented programming (AOP), reflection, runtime weaving, and various other adaptation techniques
  - Context Models (情境信息建模) : Call for a dynamic approach to context modeling: ***activity theory, situation theory, mental models*** could be useful
  - How to fit them in ?

More Sparkle references

- **Project URL: http://www.cs.hku.hk/~clwang/projects/sparkle.html**
- C.L. Wang , X.L. Zhang, N. Belaramani, P.L. Siu, Y. Chow, and F.C.M. Lau, **Software Infrastructure for Context-aware Mobile Computing**, to appear in Enabling Technologies for Wireless e-Business Applications, Springer.
- Francis C.M. Lau , Nalini Belaramani, Vivien W.M. Kwan, Pauline P.L. Siu, W.K. Wing, and C.L. Wang, ``**Code-on-demand and code adaptation for mobile computing**,'' to appear in Mobile Middleware, CRC Press, 2005.
- Nalini Moti Belaramani, Yuk Chow, Vivien Wai-Man Kwan, Cho-Li Wang, and Francis C.M. Lau, ``**A Component-based Software Architecture for Pervasive Computing**,'' Intelligent Virtual World: Technologies and Applications in Distributed Virtual Environments, chapter 10, pp. 191-212, World Scientific Publishing Co., Release: 07/31/2004.

Thanks !!

Acknowledge efforts from the Systems Research Group