ICT Convergence: Modeling issues

SAM 2004 Ottawa

Rolv Bræk, Jacqueline Floch
NTNU, SINTEF
The Question

Is it possible to identify fundamental properties of services we may use to determine the right/best service architectures, delivery platforms and service engineering methods?

... are the differences between approaches of the computing domain and the telecom domain just accidental or well justified?
Shaping forces

Telecom domain (real time domain)
- Telecom applications
- Telecom platforms

Computing domain
- Computing applications
- Computing platforms
The computing domain

- **Information processing by means of data and algorithms (or objects and methods).** Encapsulating data in objects and introducing classes with inheritance does not fundamentally change this.

- **Communication by invocation.**
The calling entity is blocked until control is returned from the called entity.

- **Asymmetrical, or client-server interactions.** Asymmetrical request-response types of communication dominate.

- **Concurrency as add-on.**
The telecommunication (RT) domain

- **Active objects with concurrent behavior.**
  Real objects, like users, behave concurrently and need to interact and to be served concurrently.

- **Communication by signaling.**
  Active objects need explicit communication mechanisms such as signal sending, or messaging to interact.

- **Symmetrical or peer-to-peer interactions.**
  Objects need to communicate on an equal basis, with few restrictions. Initiatives may be taken independently and simultaneously and lead to conflicts that must be resolved.
Two kinds of functionality:

- **Client-server (computing domain)**
  - One-way initiatives
  - A service as an interface
  - Communication by invocation
  - Restricted structure
  - Passive objects

- **Peer-to-peer (telecom and real-time)**
  - Multi-way initiatives
  - A service as a collaboration
  - Asynchronous communication
  - General structure
  - Active objects

... now meeting each other
Two modeling approaches

Computing domain:
- Passive objects
- Associations
- One-way interfaces and operations
- Client-server with one-way initiatives
- Communication by invocation
- Three-like structure

Telecom domain:
- Active objects
- Channels
- Two-way interfaces and protocols
- Peer-to-peer with multi-way initiatives
- Asynchronous communication by messaging
- General network structure
Two design approaches

- **Server oriented approach**
  - One interface - one service - multiple users
  - Class-operation focus

- **Agent oriented approach**
  - One interface - one user - multiple services (as roles)
  - Object-interaction focus
Convergence

P-P models ↔ C-S models

P-P applications ↔ C-S applications

P-P platforms ↔ C-S platforms
We believe the most general approach is Peer-to-peer with active objects and asynchronous communication by messaging,
because:

- It reflects real world domain and distributed platform issues
- It can support both peer-to-peer and client-server structures without restrictions
- It supports distribution transparency in a simple way and uses the basic mechanism for information transfer over networks
- It should therefore be at the core
For programming within a single address space and thread of computing (Active object).

When necessary to interface with legacy systems and APIs.

When convenient for application programming.

When speed can be gained (but remote interactions are bound to be slower)

How, then, can invocation and messaging be combined?
The SDL solution

- Remote procedure call
- Transform
- Add state
- Messages
- Add input
Active-active invocation

a) Direct invocation

b) "Invocation" by messaging

- add state
- add input

{sequential, guarded, concurrent}
Active-passive invocation

a) Direct invocation

b) Edge mediated

translate
messaging
add state

{sequential, guarded, concurrent}
Passive-active invocation

a) Direct invocation

b) Edge mediated

{sequential, guarded, concurrent}

translate

add input
Core invocation considerations

- Concurrency: are the parties concurrent or not?
- Initiative patterns: one-way or two-way?
- Communication structure: three-like or networked?
- Blocking delays: are they acceptable?
- Synchronisation delays: are they acceptable?

Suitable for:
- One logical thread of behaviour
- One-way initiatives
- Three-like structure
AMIGOS – meeting people

Calls, Chat, Multimedia conferences, Location awareness, Buddy lists
Sharing objects

e.g.: work-teams, classrooms, friends,...
Amigos: a mixed approach

Manage stateful behaviours and collaborations

Associate profiles with agents
Environment and edges
ActorFrame: towards a convergent framework

Functionality models using Role and Actor modeling (RAM)

P-P + C-S
ActorFrame
MIDP, JMS, RMI,

Asynchronous networks

Terminals, appliances

Application servers

P-P + C-S
EJB - ActorFrame
J2EE: JMS, WS, EJB, RMI, ...

PIM'}
RAM – Service and Actor Modelling

Collaborations, sequences

Role behaviours

Services

System

Agents

Actors

Application: PIM

Adaptation: PIM’

PIM’
RAM – implementation and deployment

System

Agents

Actors

Application: PIM

Adaptation: PIM'

PIM'

UML Actor Models

Java Frameworks

Java + XML

Dynamic deployment
Conclusions

- Yes, there are fundamental domain properties that shape applications and platforms
- Asynchronous communication at the application level needed as basis
- With invocation based communication as supplement
- Will application programmers accept it?
- Can they avoid it?
- Mixed modelling using UML is possible, with care
- An architectural framework helps
- Dynamic composition becoming more important