Resilient Actors

A Runtime Partitioning Model for Pervasive Computing Services

Engineer Bainomugisha, Jorge Vallejos,
Éric Tanter*, Elisa Gonzalez Boix, Pascal Costanza,
Wolfgang De Meuter, Theo D'Hondt

Vrije Universiteit Brussel, Belgium

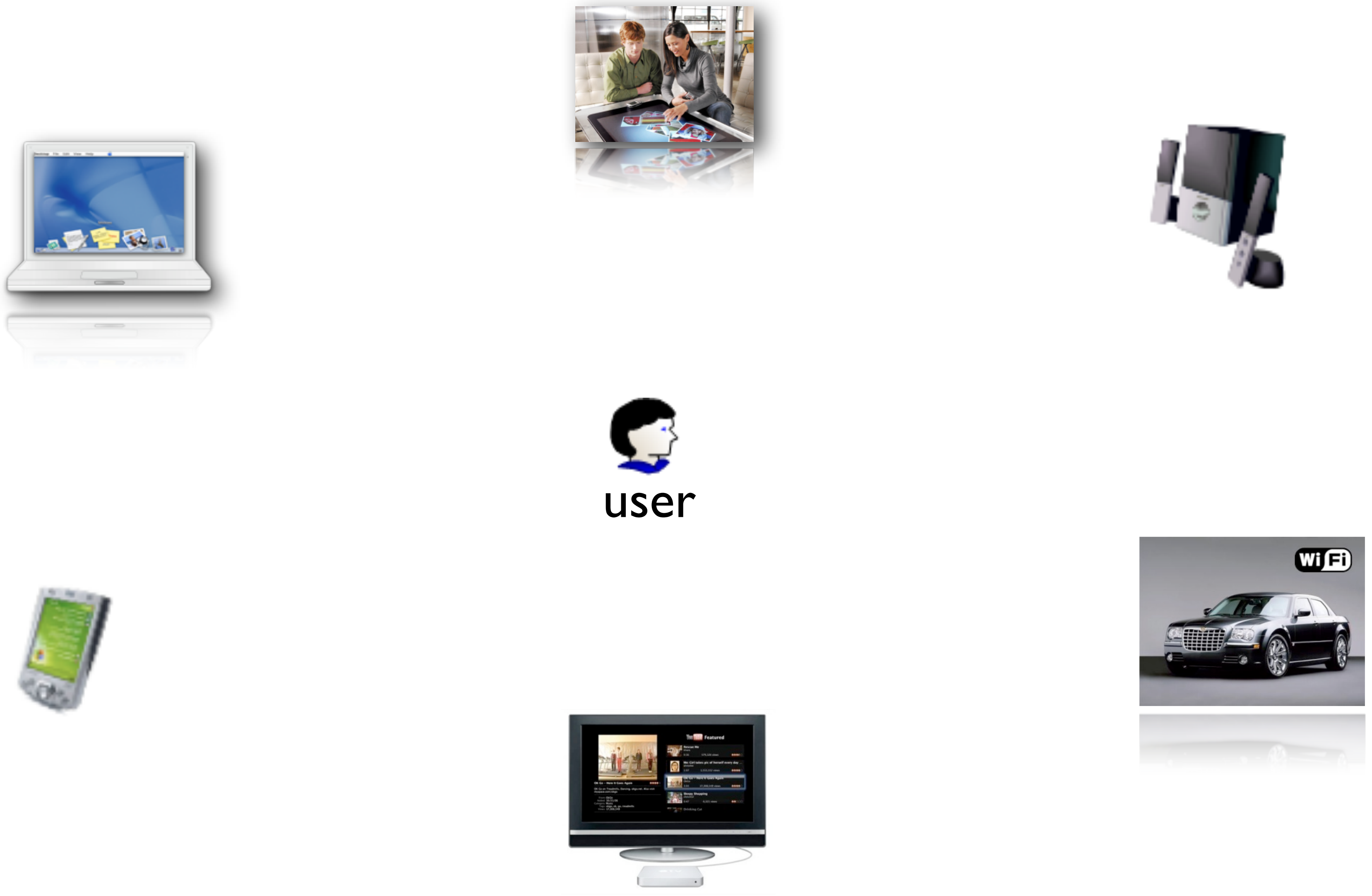
*University of Chile

July 14, 2009
International Conference on Pervasive Services (ICPS'09), London, UK

Pervasive Computing Environments

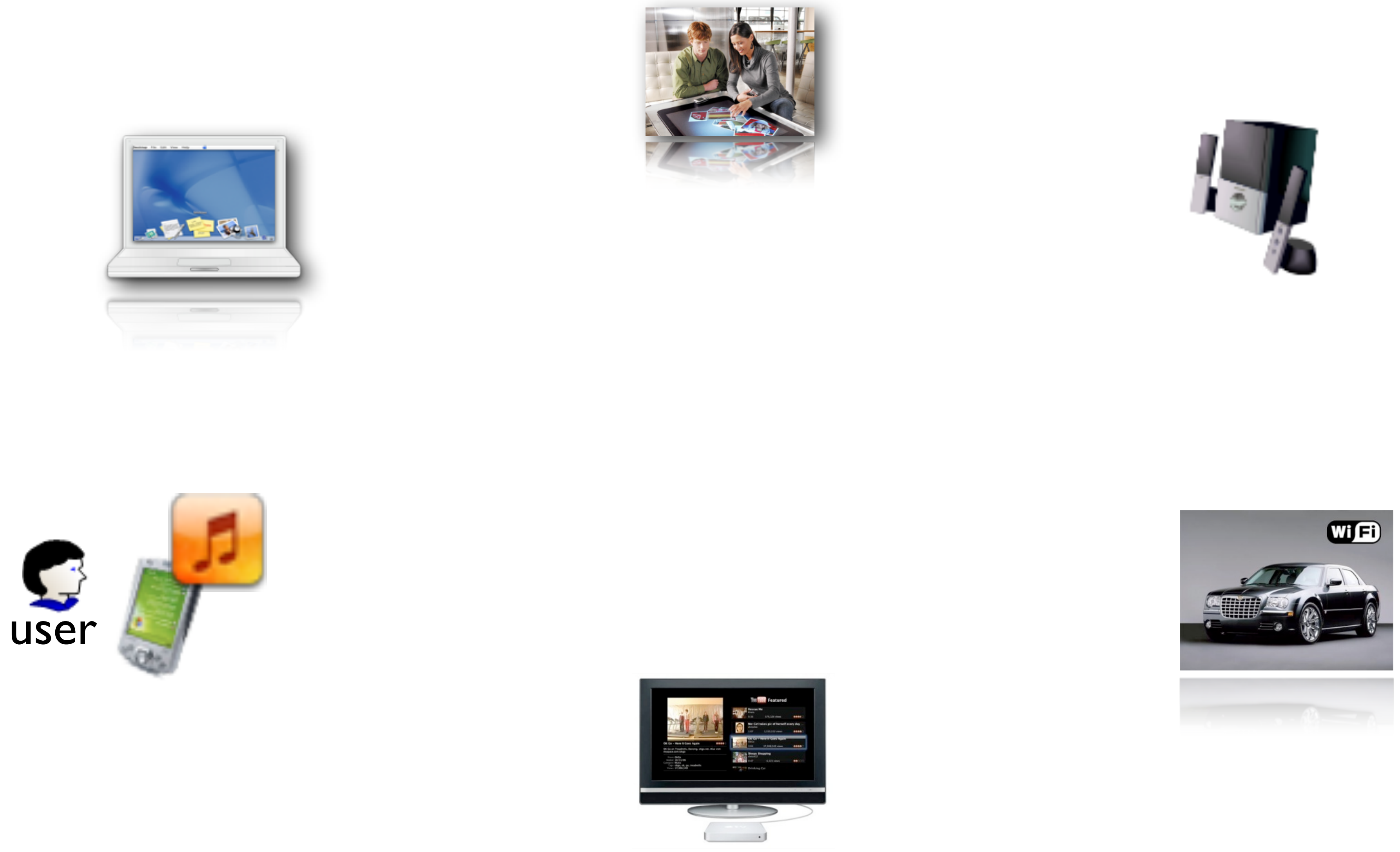
Computers integrated into everyday devices (Weiser, 1993)

Pervasive Computing Environments



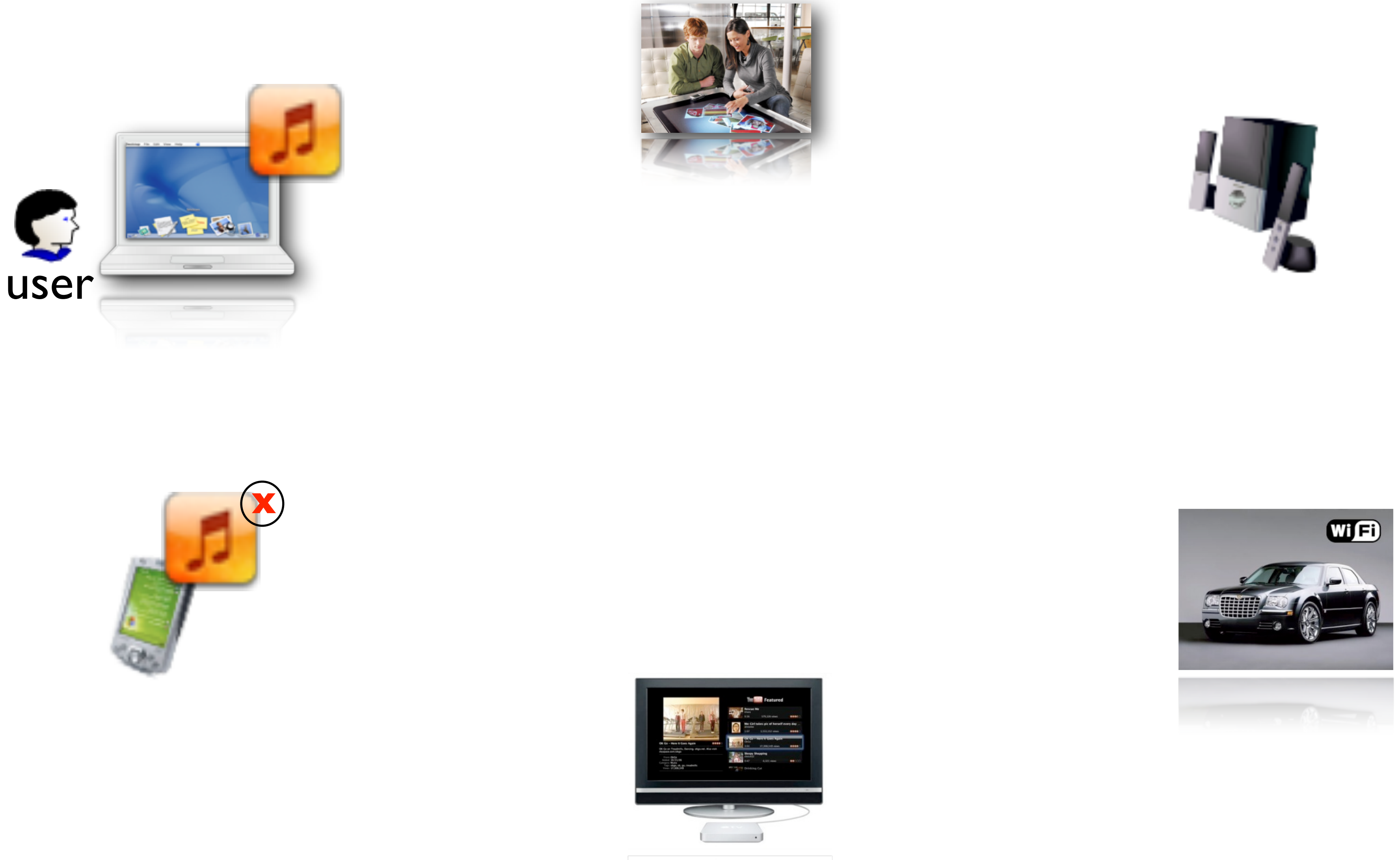
Computers integrated into everyday devices (Weiser, 1993)

Pervasive Computing Environments



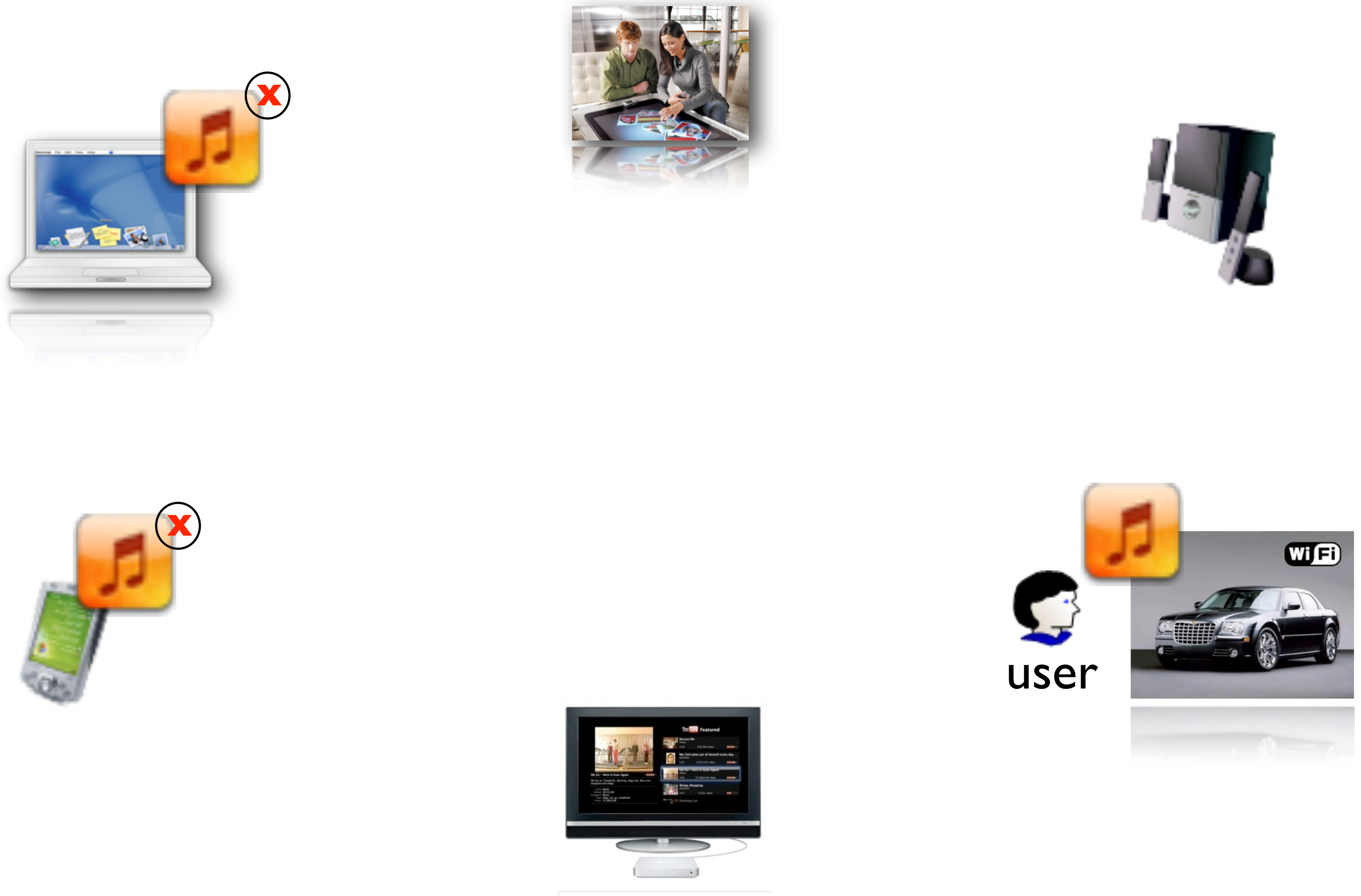
Computers integrated into everyday devices (Weiser, 1993)

Pervasive Computing Environments



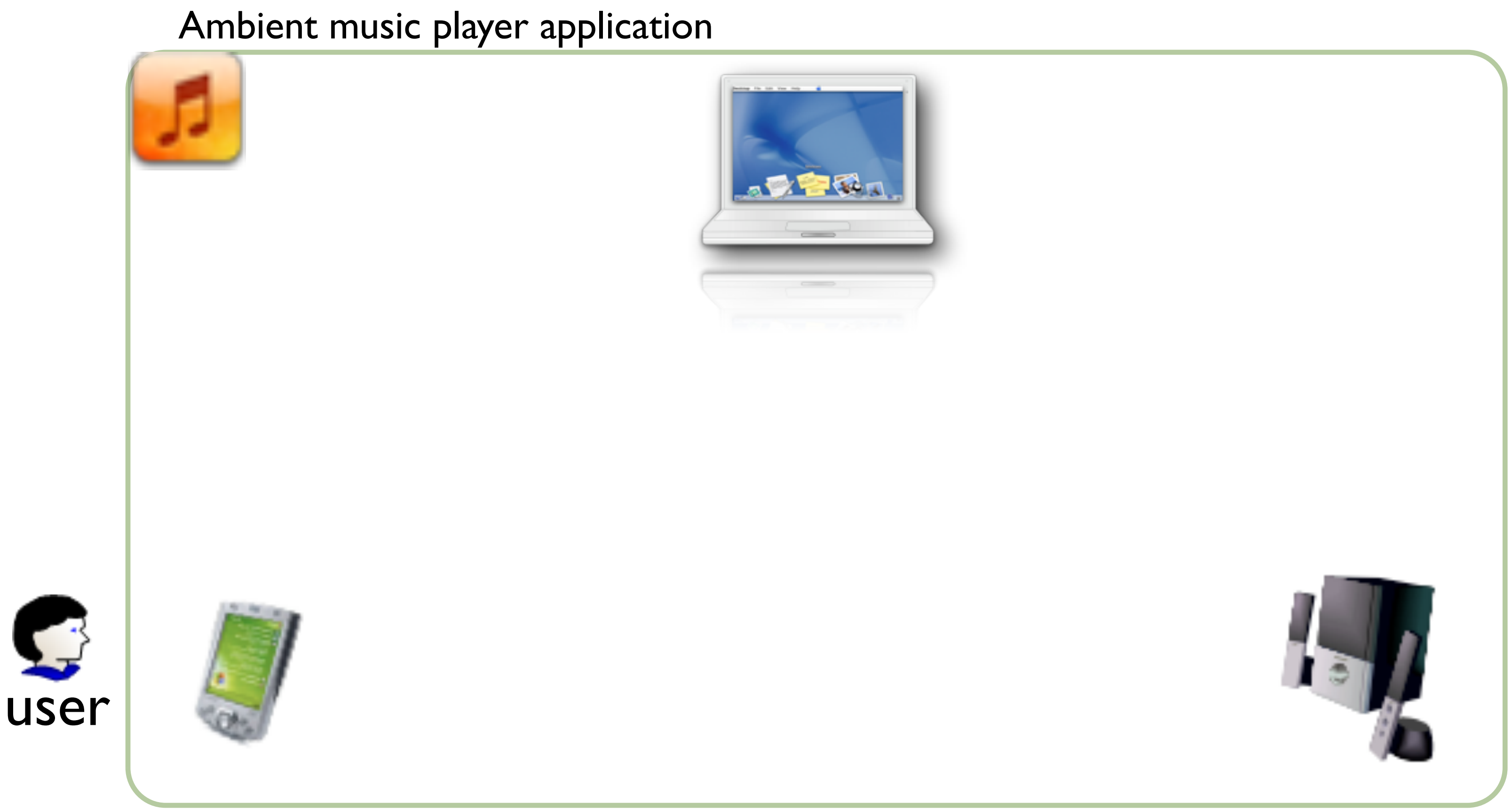
Close and start-up the application as the user roams

Pervasive Computing Environments



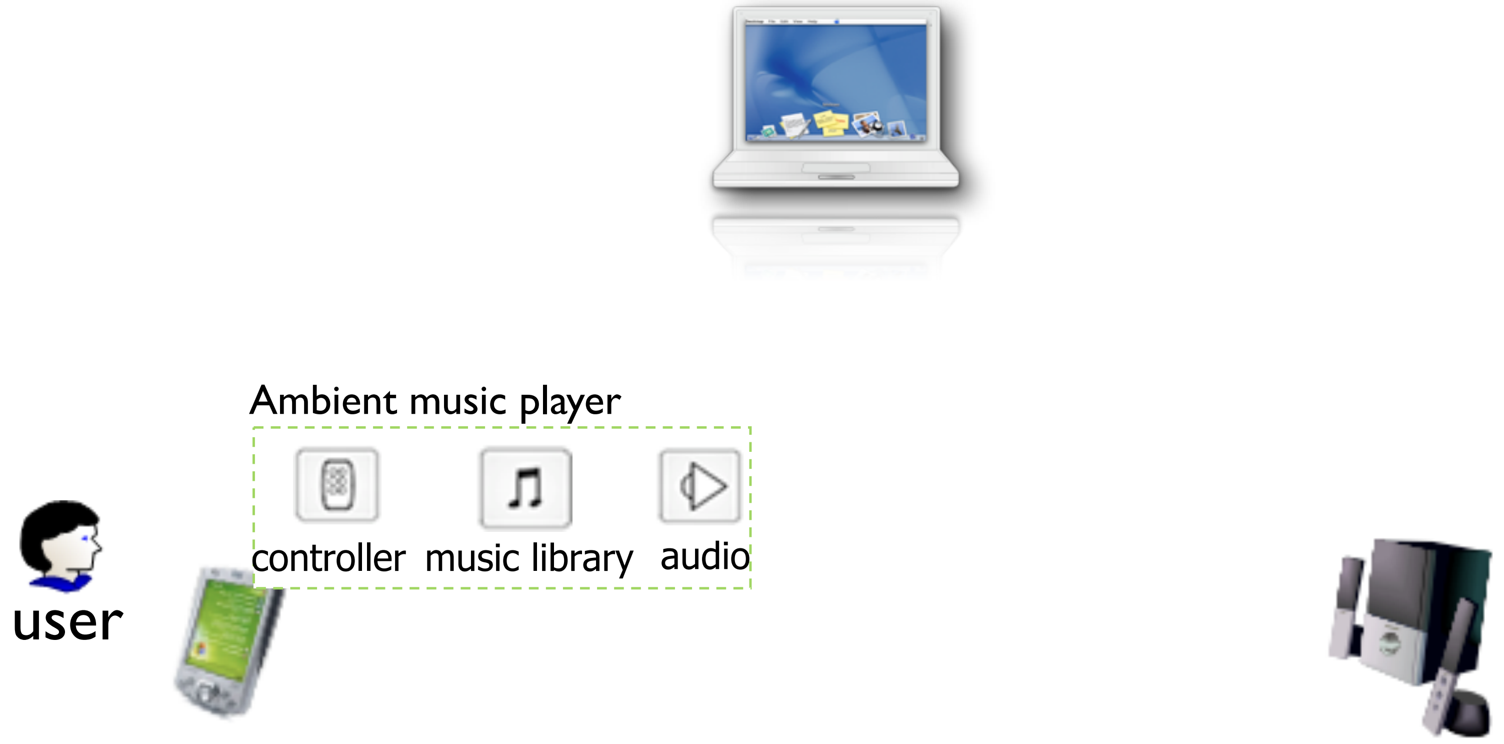
Close and start-up the application as the user roams

Partitioning of Pervasive Computing Services



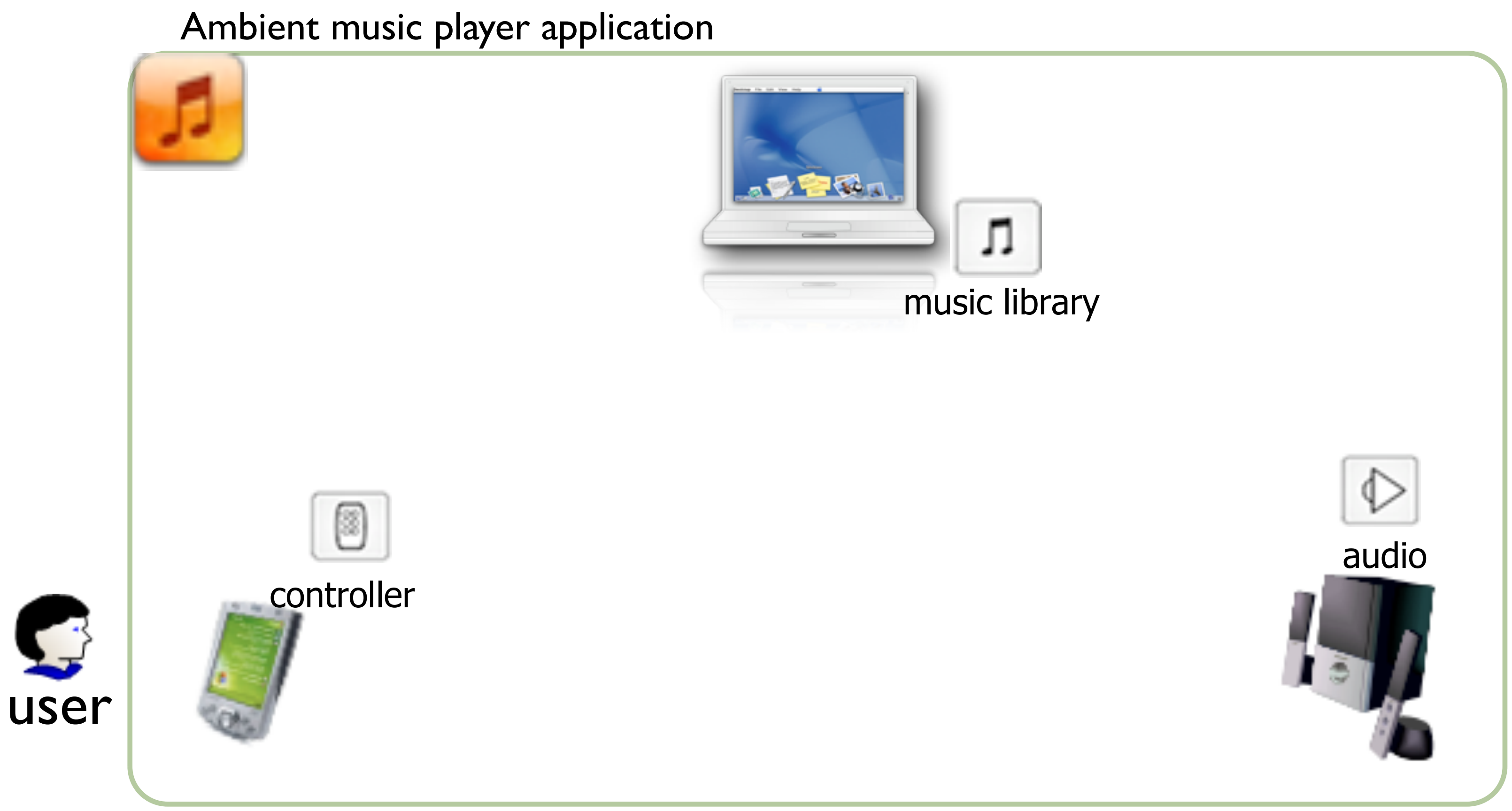
Applications spread their functionality amongst several devices

Partitioning of Pervasive Computing Services



Applications spread their functionality amongst several devices

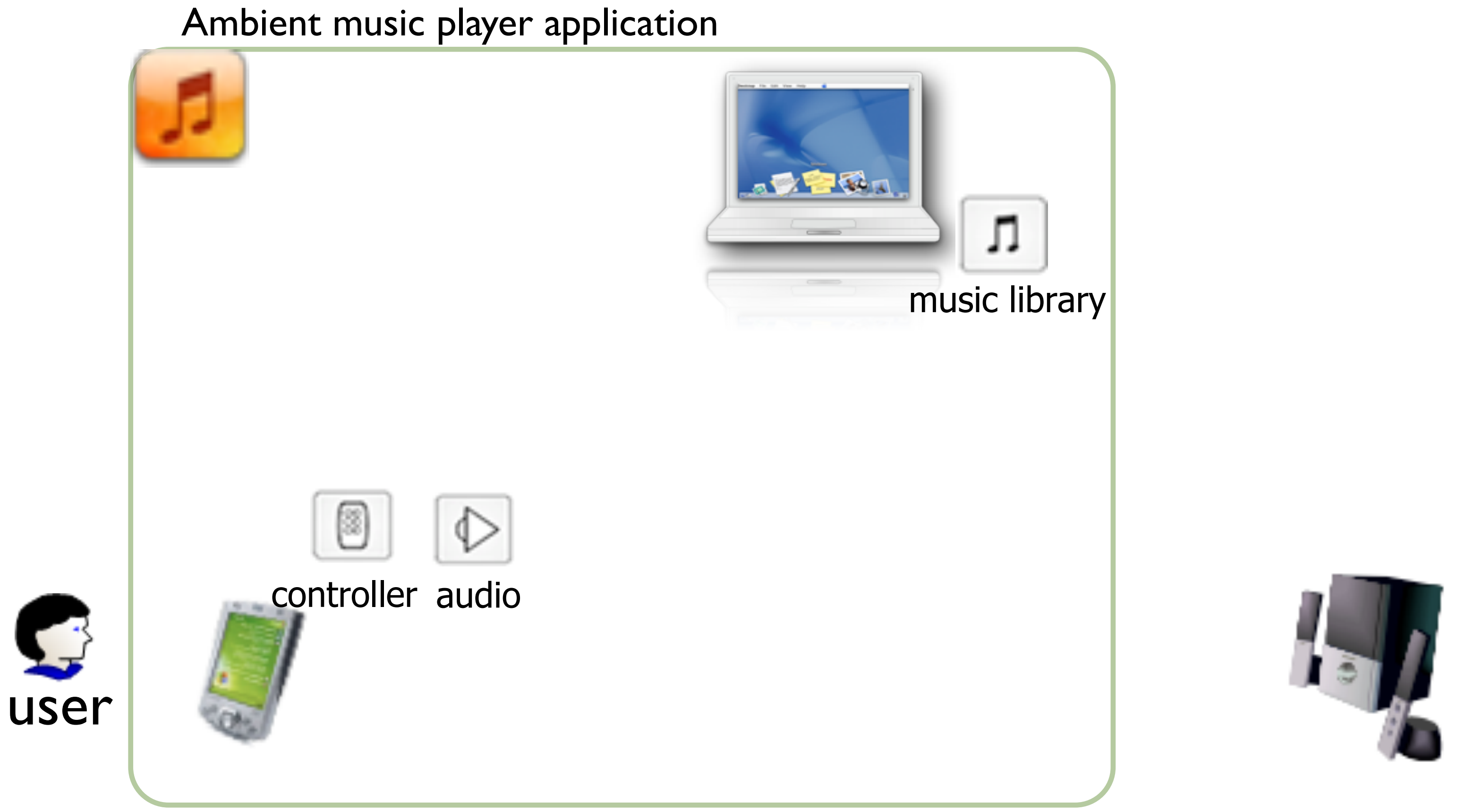
Partitioning of Pervasive Computing Services



Applications are decomposed to run on multiple devices

- Runtime application partitioning

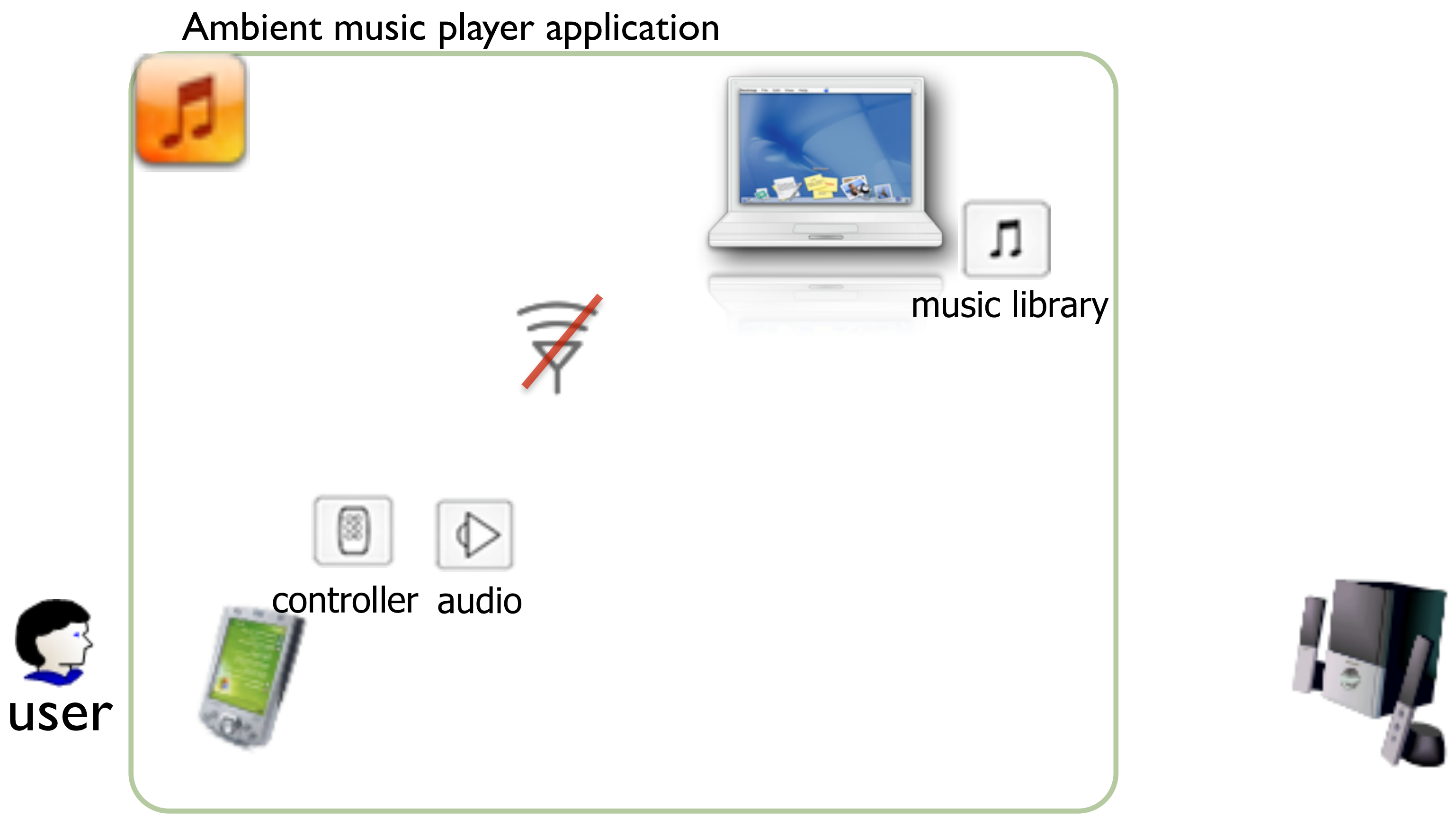
Partitioning of Pervasive Computing Services



Applications are decomposed to run on multiple devices

- Runtime application partitioning
- Retractable

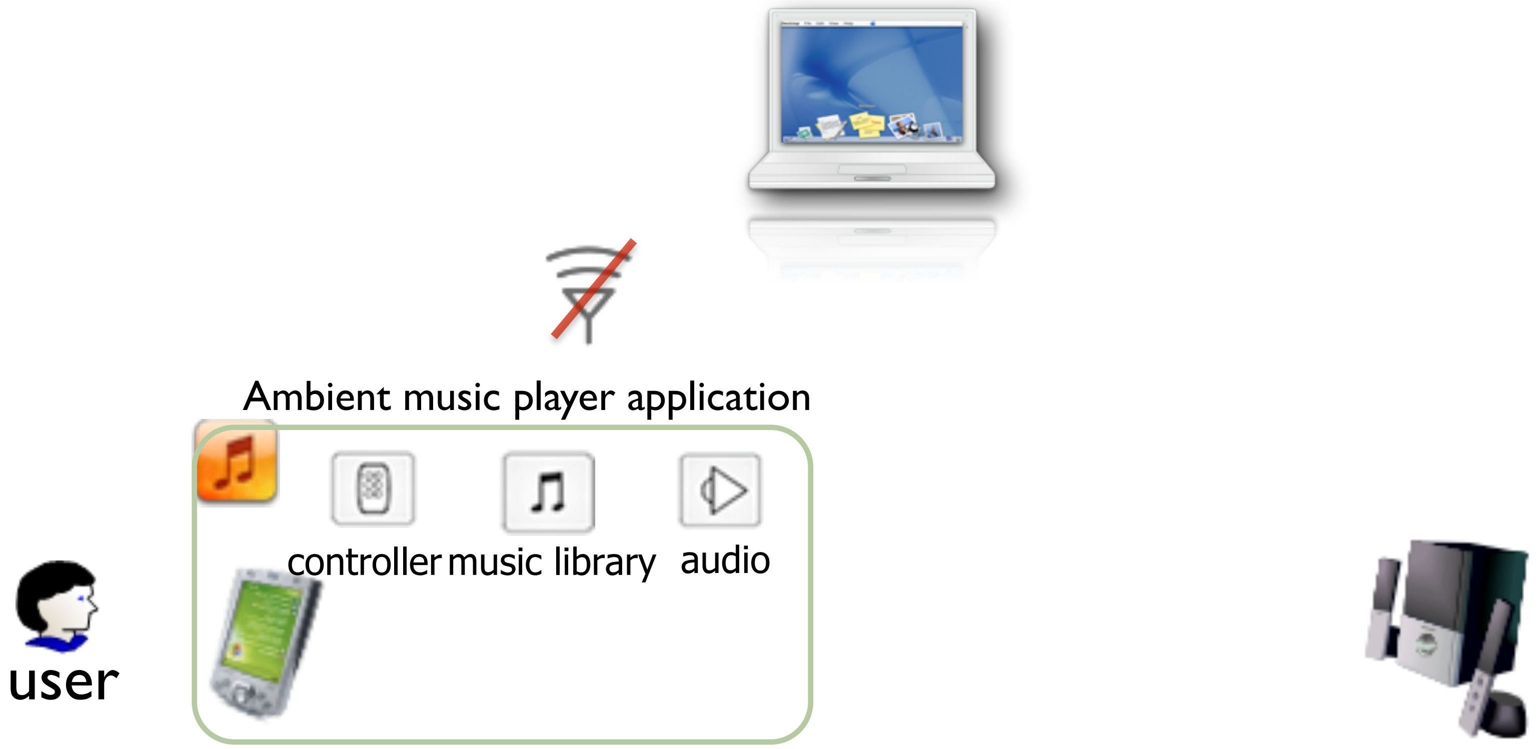
Partitioning of Pervasive Computing Services



Applications are decomposed to run on multiple devices

- Runtime application partitioning
- Retractable

Partitioning of Pervasive Computing Services



Applications are decomposed to run on multiple devices

- Runtime application partitioning
- Retractable
- Resilient to network disconnections

Existing Approaches

- Mostly static and controlled by the programmer (e.g. J-Orchestra, Addistant, ..)
- Object migration automatic or based on algorithms
- No network failure handling mechanisms

Service Partitioning Requirements

Runtime Application Partitioning

Retractable

Resilient to Network Failures

Resilient Actor Model

A resilient actor:

- A program entity that encapsulates a set of objects
- **Elastic bindings** to other actors

Resilient Actor Model

A resilient actor:

- A program entity that encapsulates a set of objects
- **Elastic bindings** to other actors

Two partitioning operations

- **Stretch** - moves an actor to another device
- **Retract** - moves an actor back to original device

Resilient Actor Model

A resilient actor:

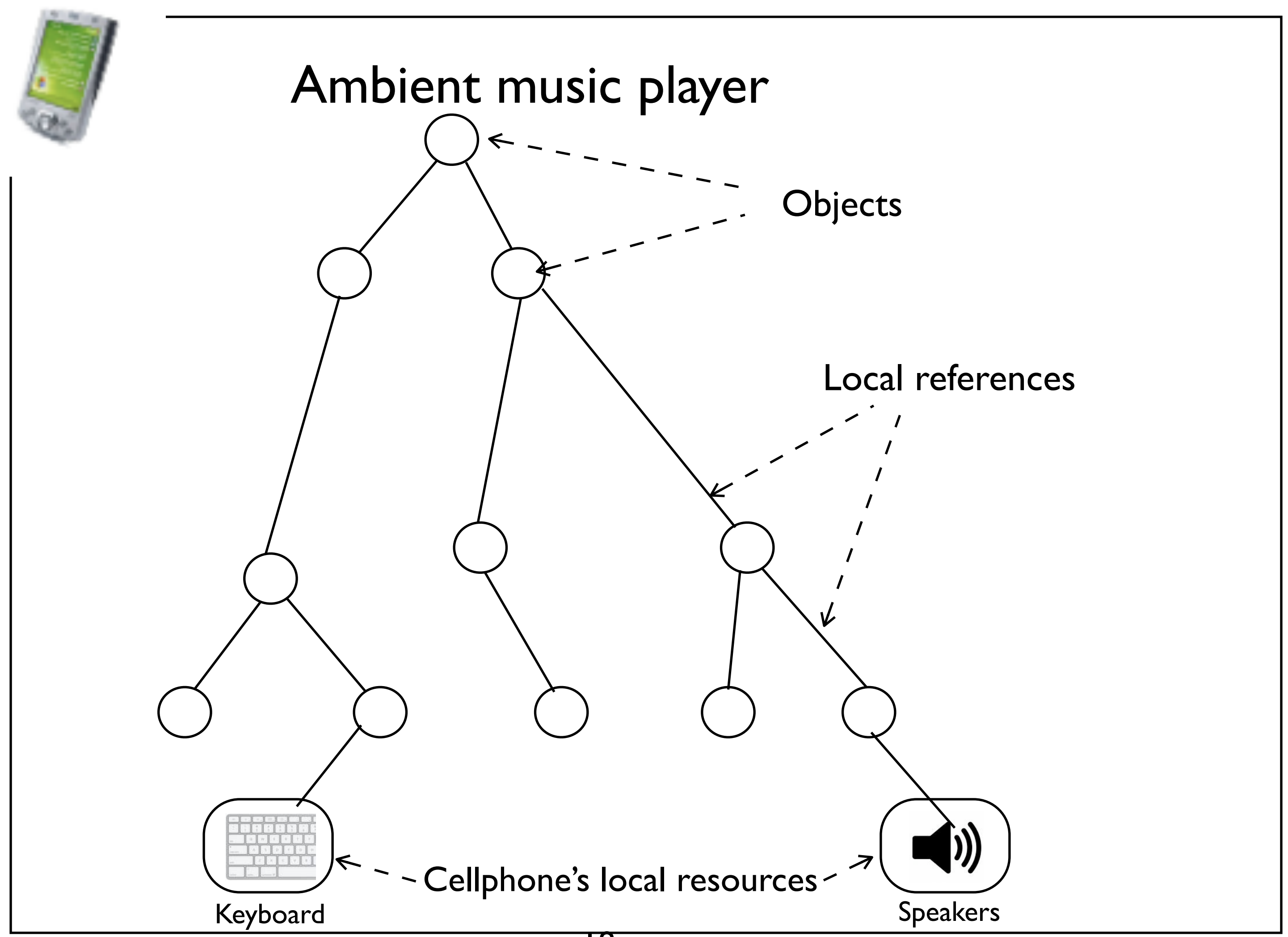
- A program entity that encapsulates a set of objects
- **Elastic bindings** to other actors

Two partitioning operations

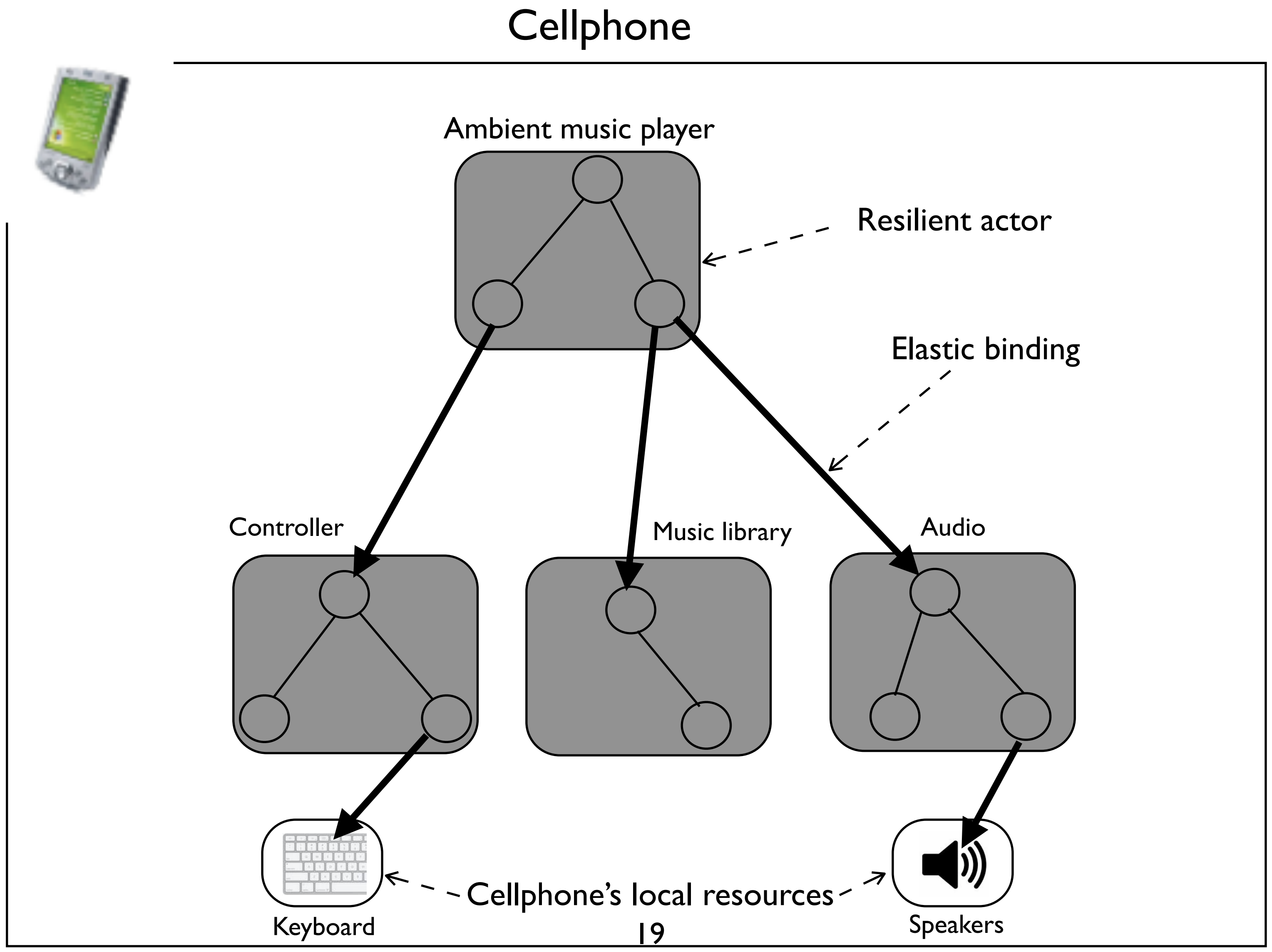
- **Stretch** - moves an actor to another device
- **Retract** - moves an actor back to original device

Resilience strategies (move, copy, rebind, standstill)

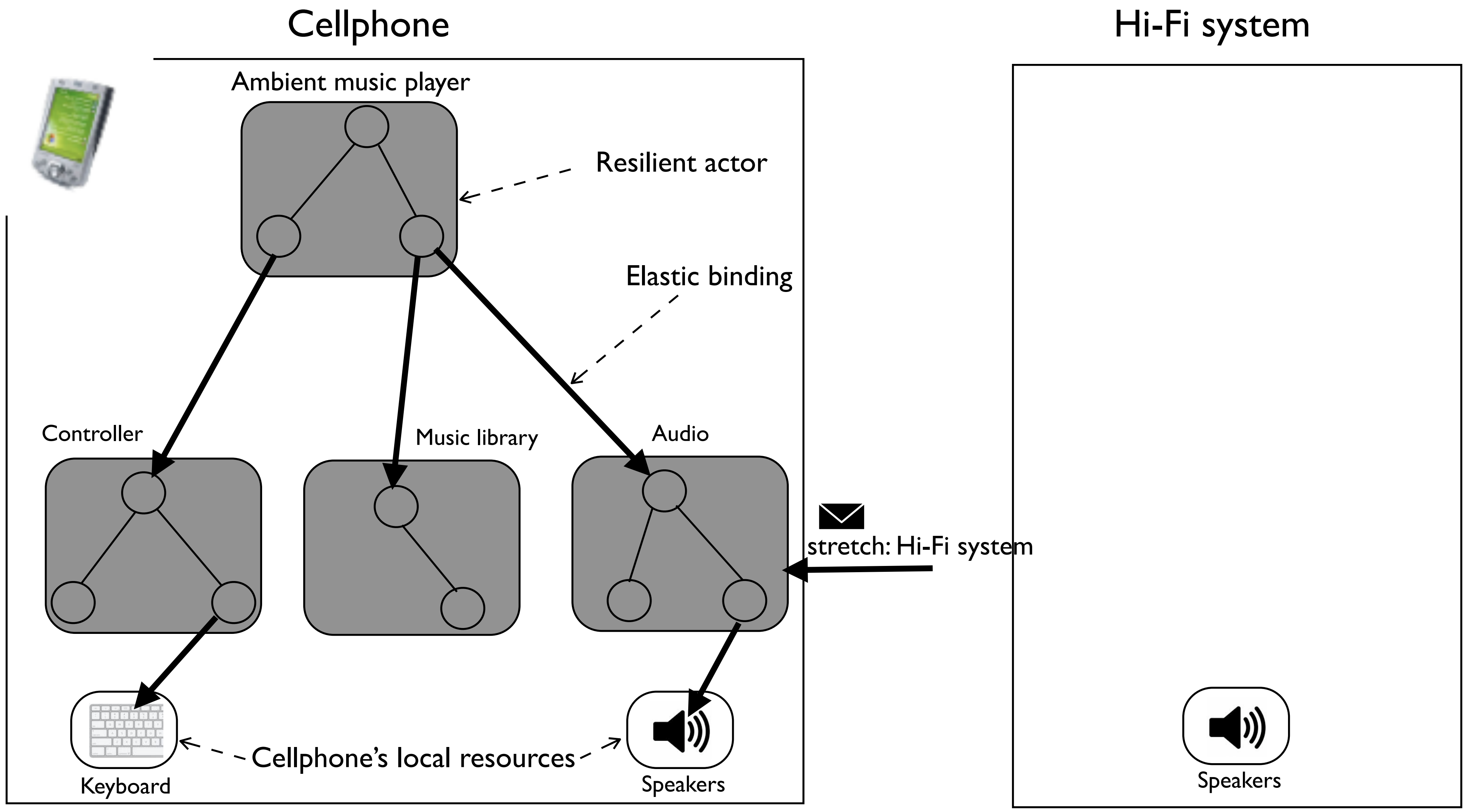
Service Partitioning



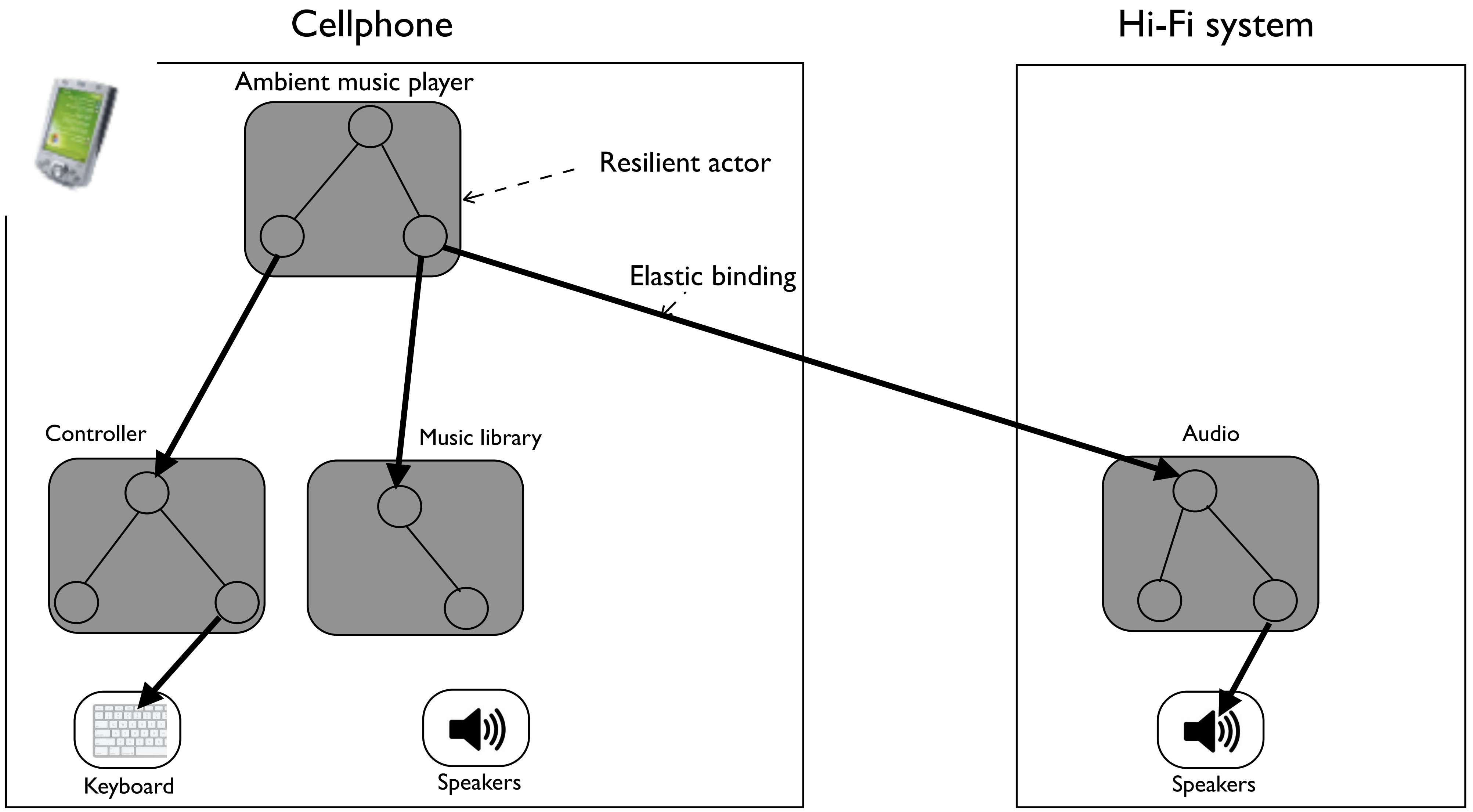
Service Partitioning



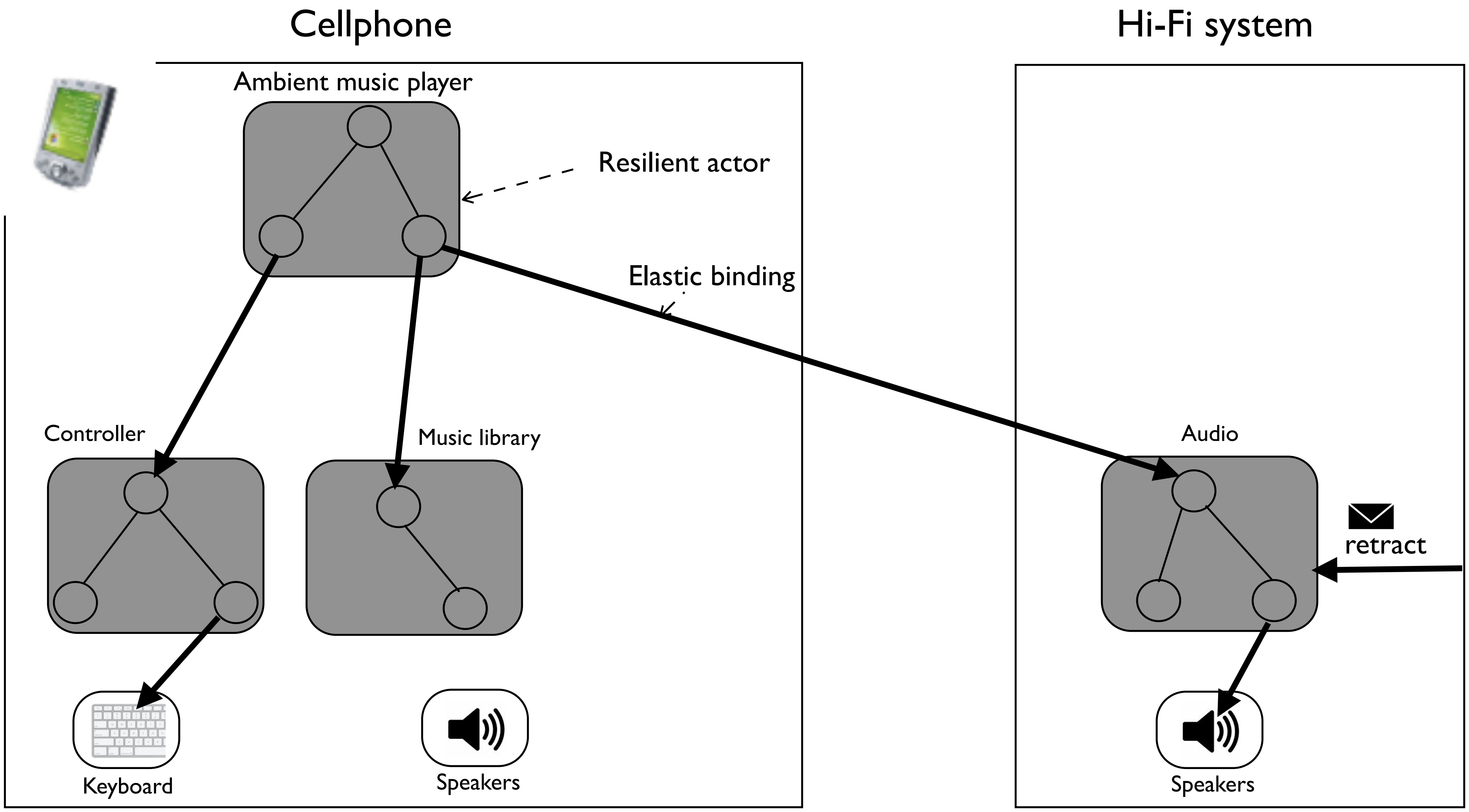
Service Partitioning



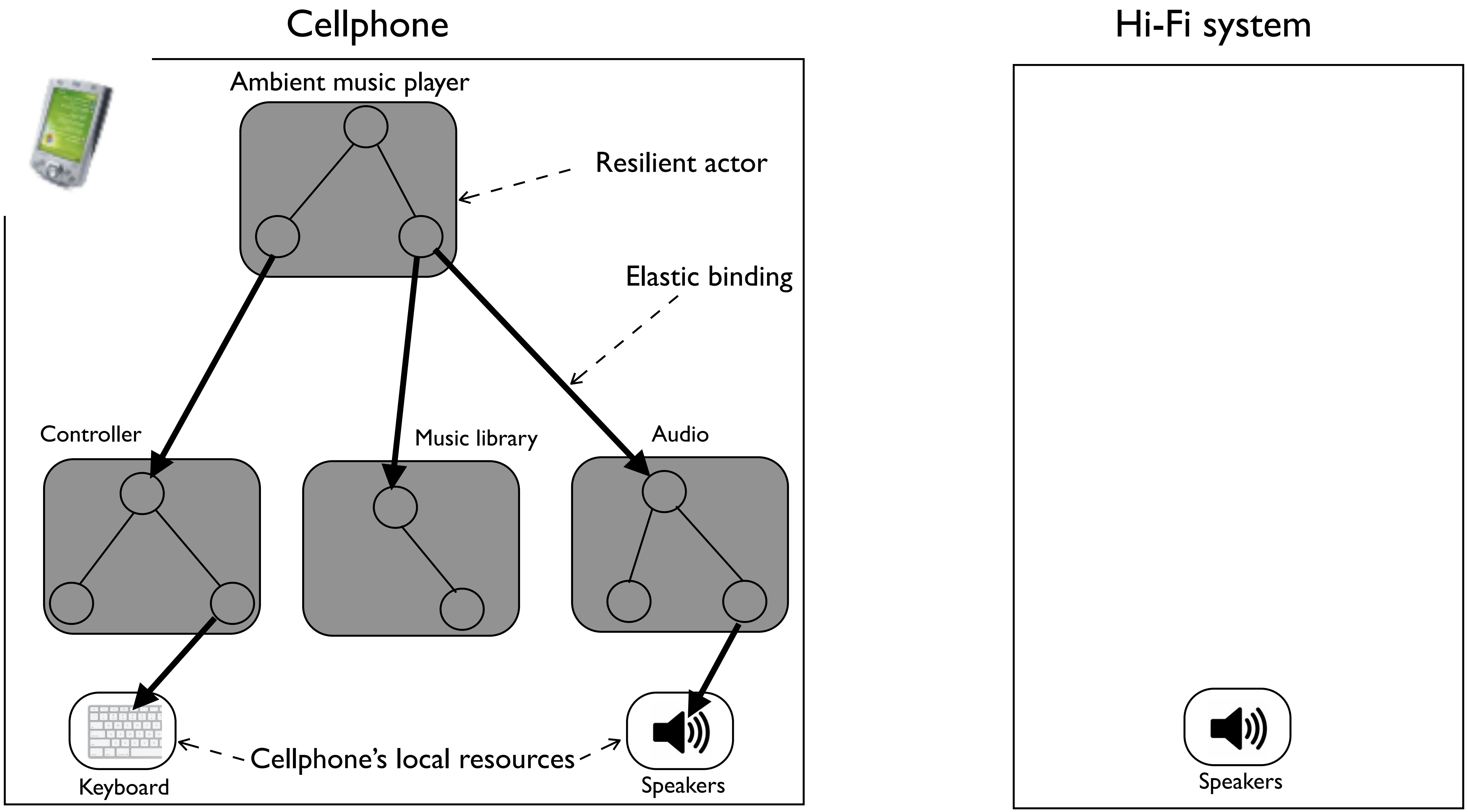
Service Partitioning



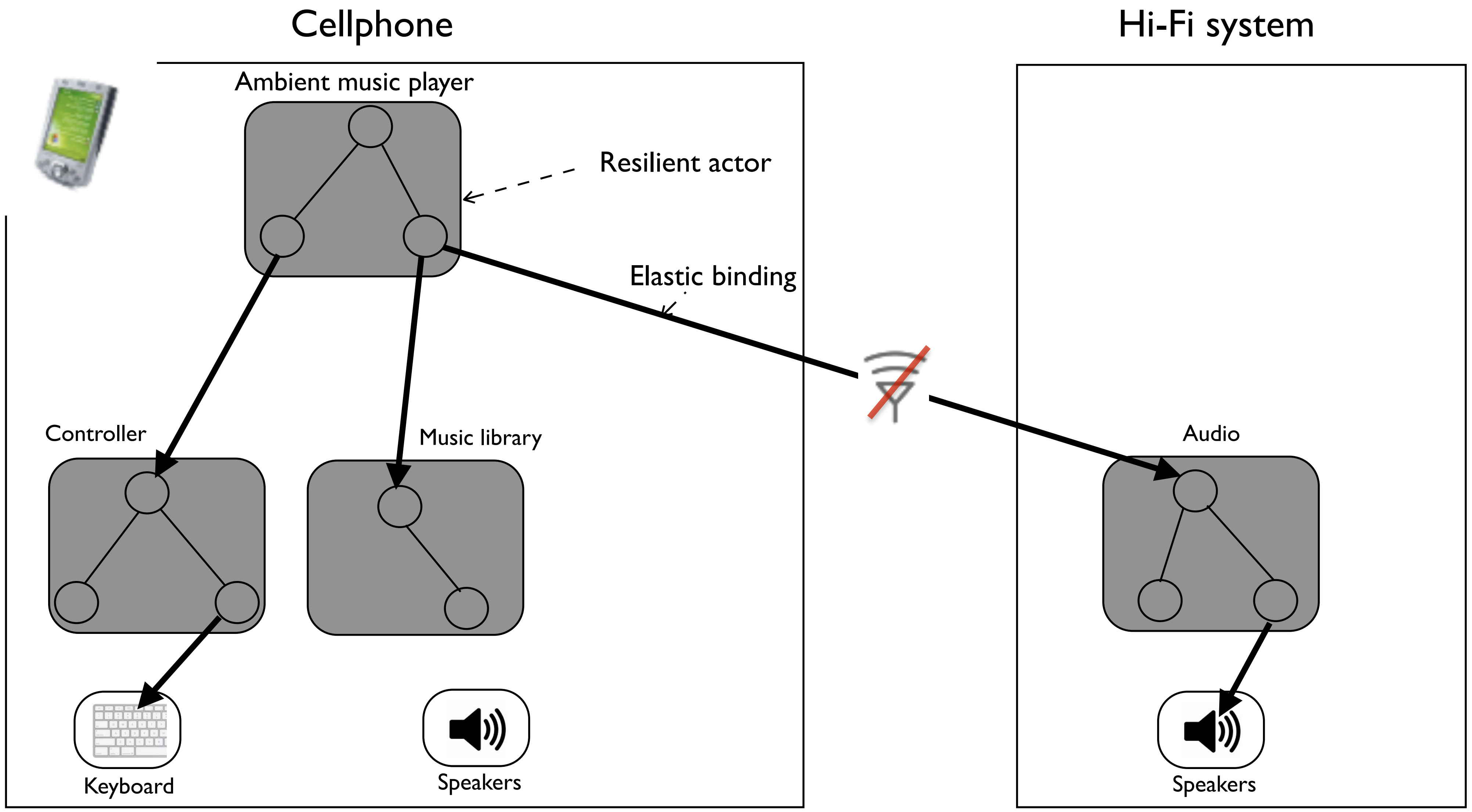
Retraction


Cellphone
Hi-Fi system
Ambient music player
Resilient actor
Elastic binding
Controller
Music library
Audio
retract
Keyboard
Speakers
Speakers

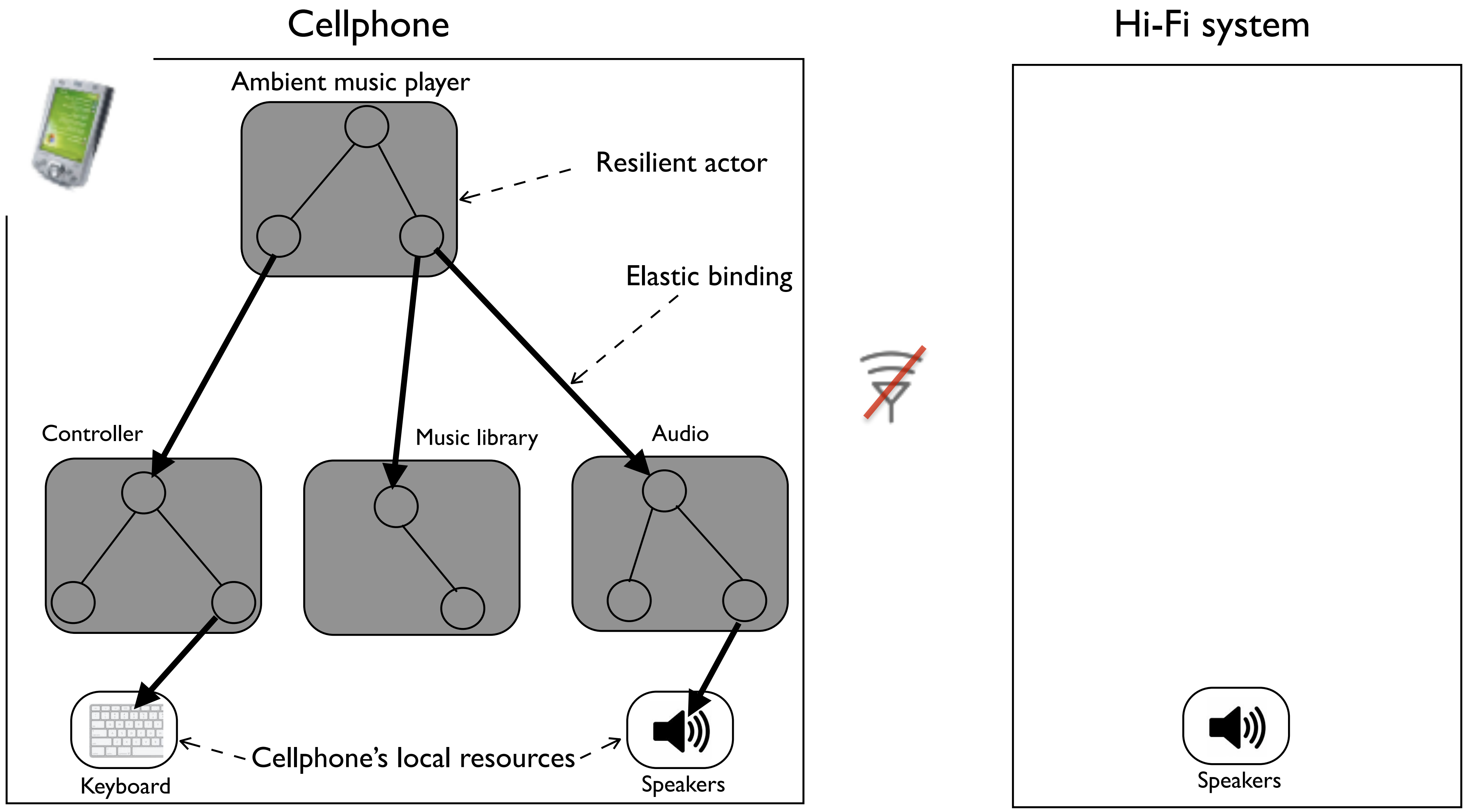
Manual Retraction



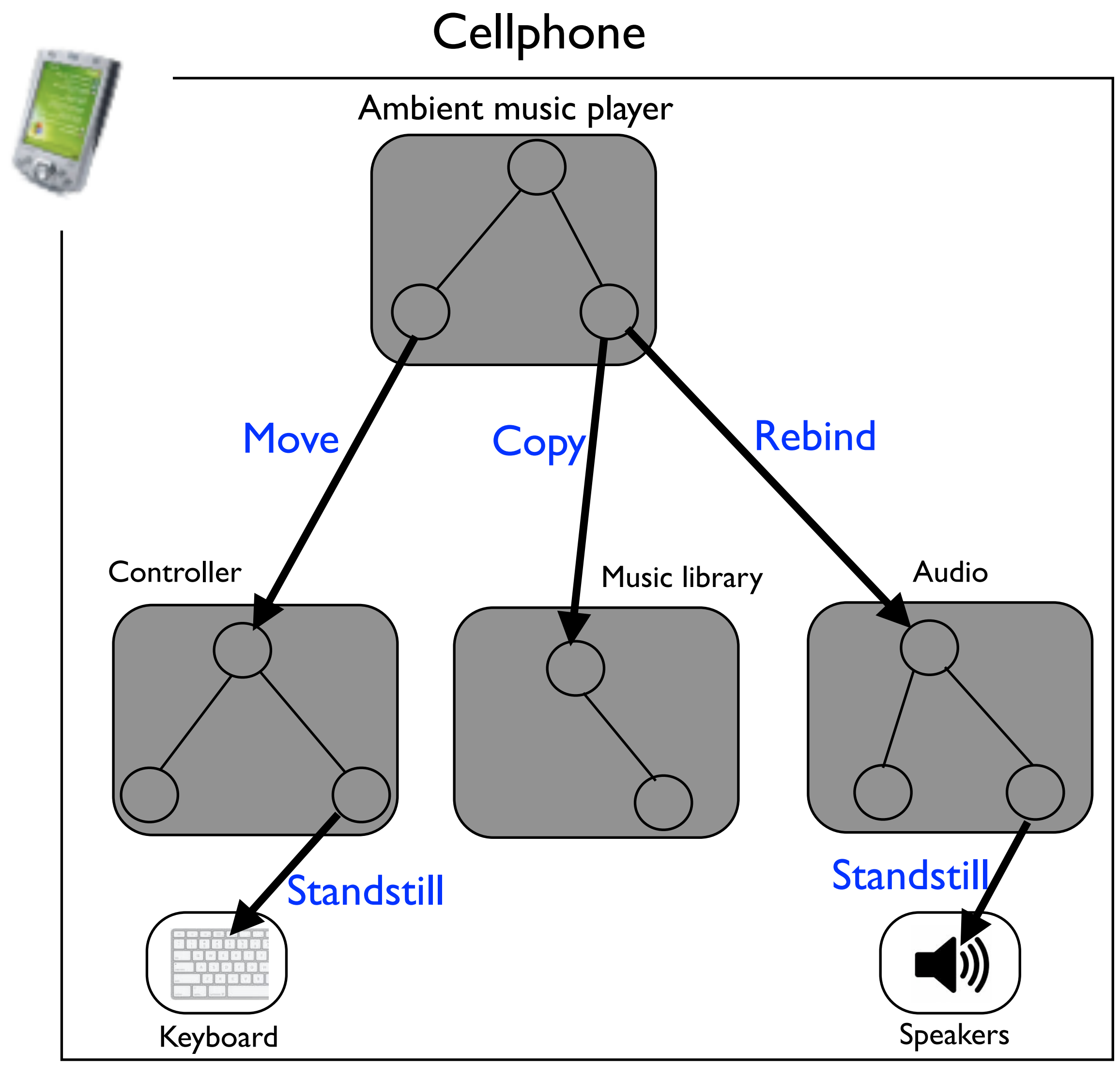
Automatic Retraction



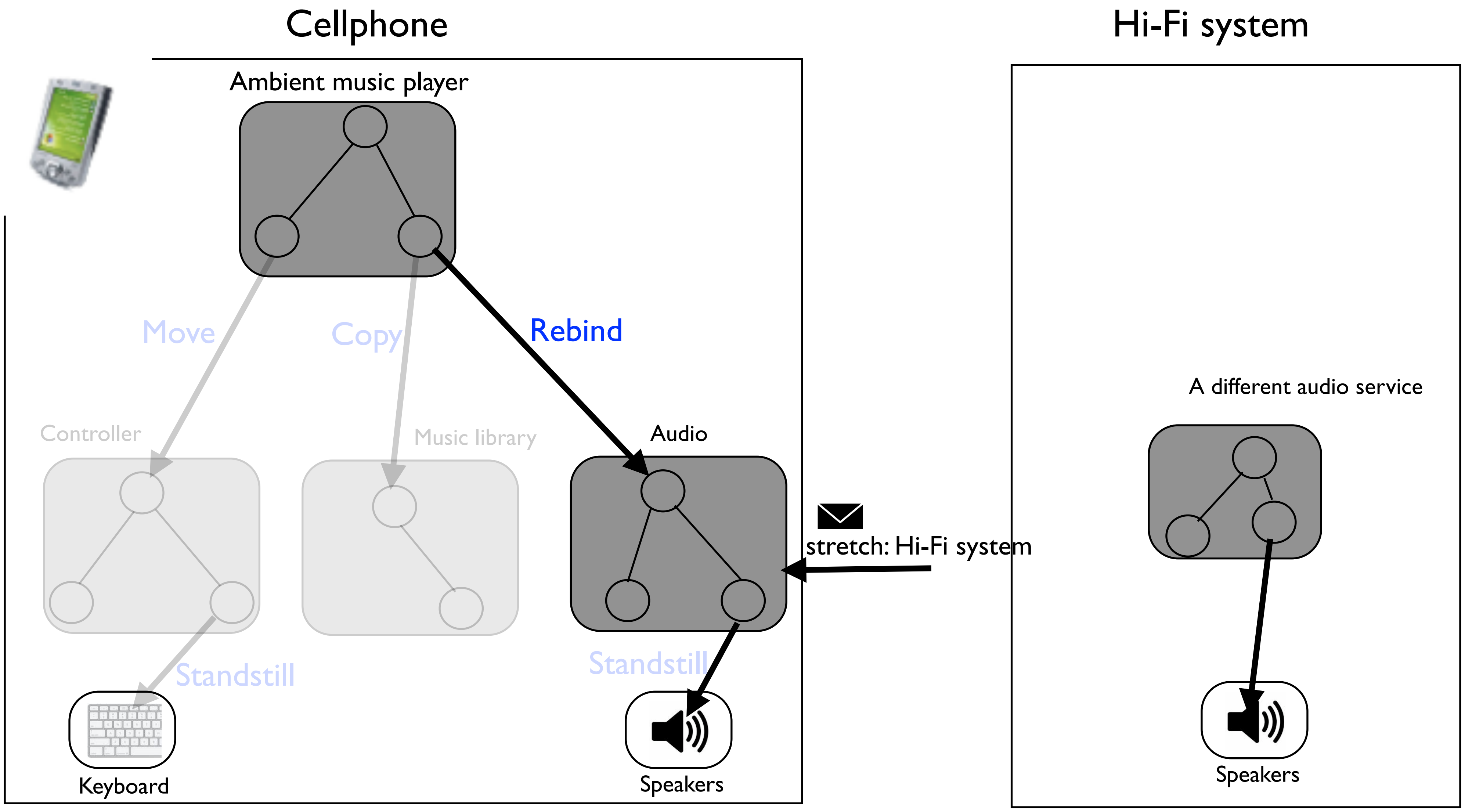
Automatic Retraction



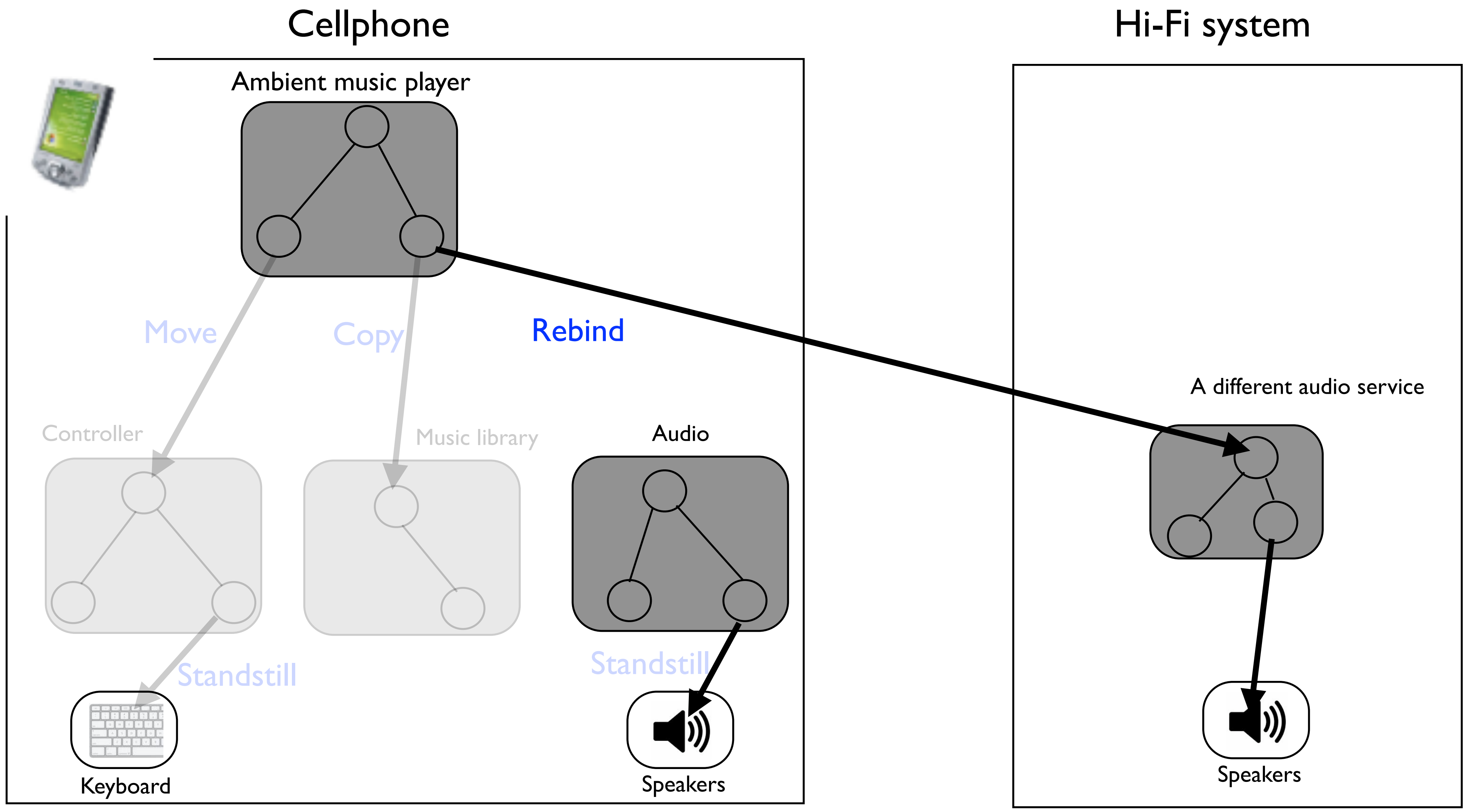
Resilience Strategies



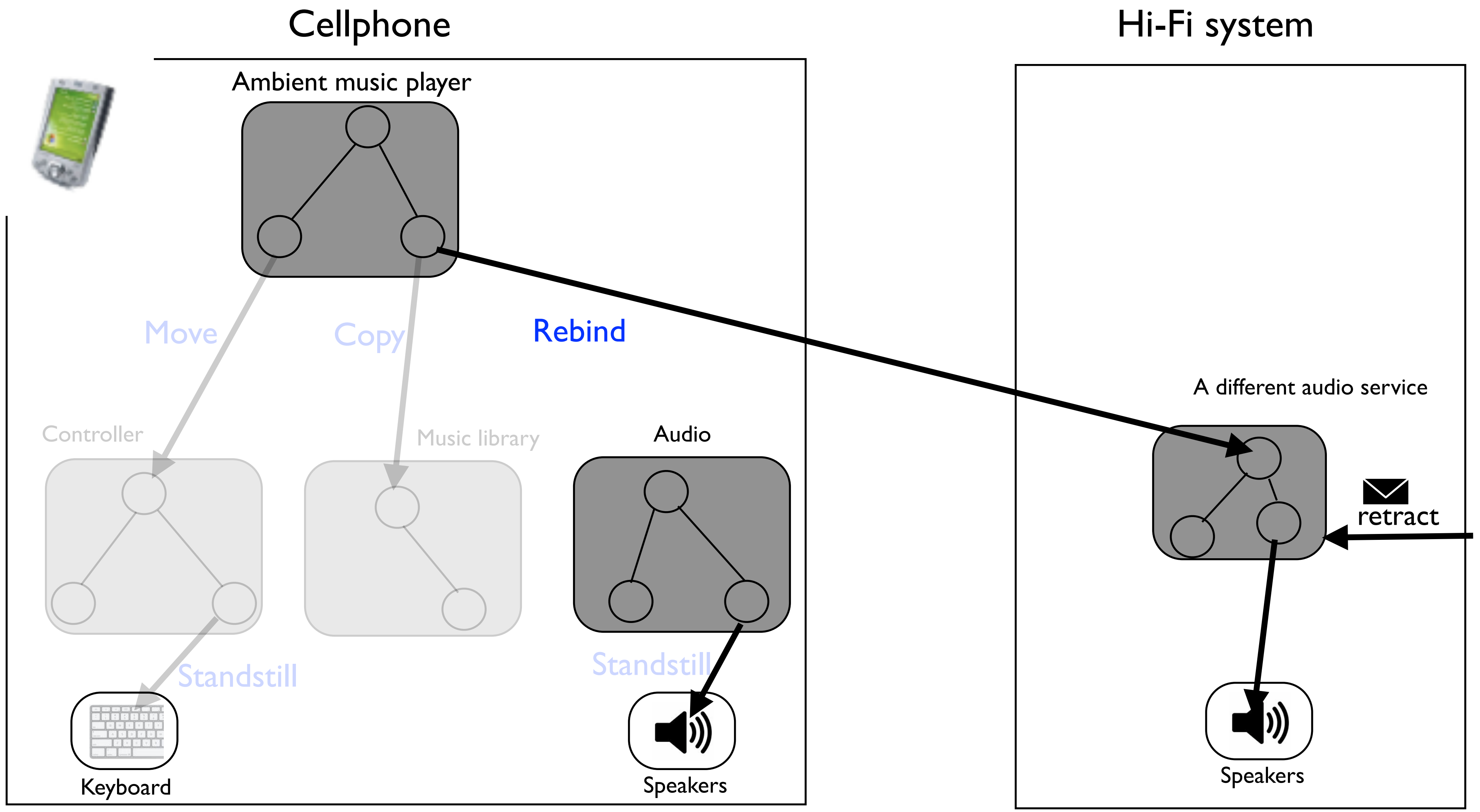
Stretch under Rebind Strategy



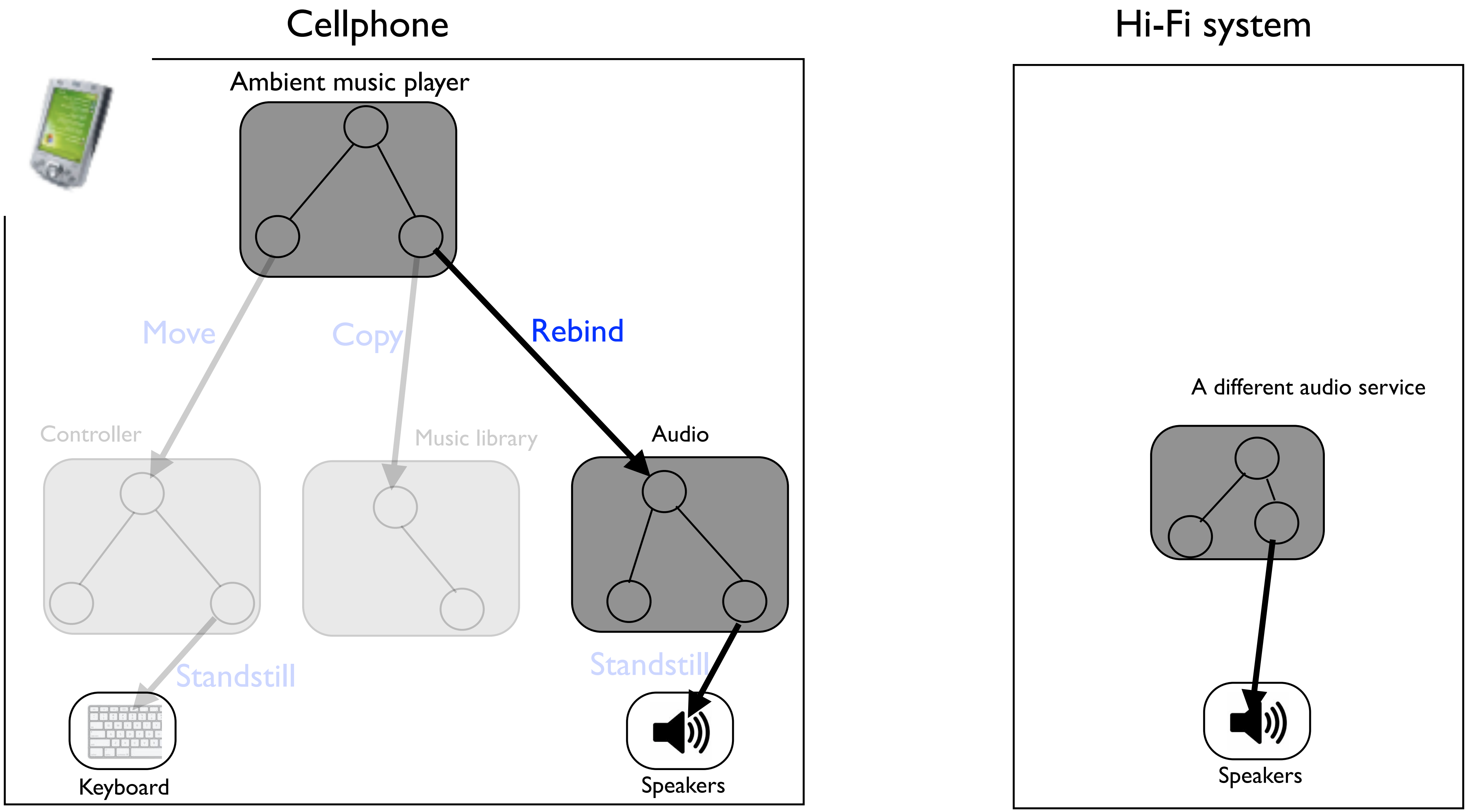
Retract under Rebind Strategy



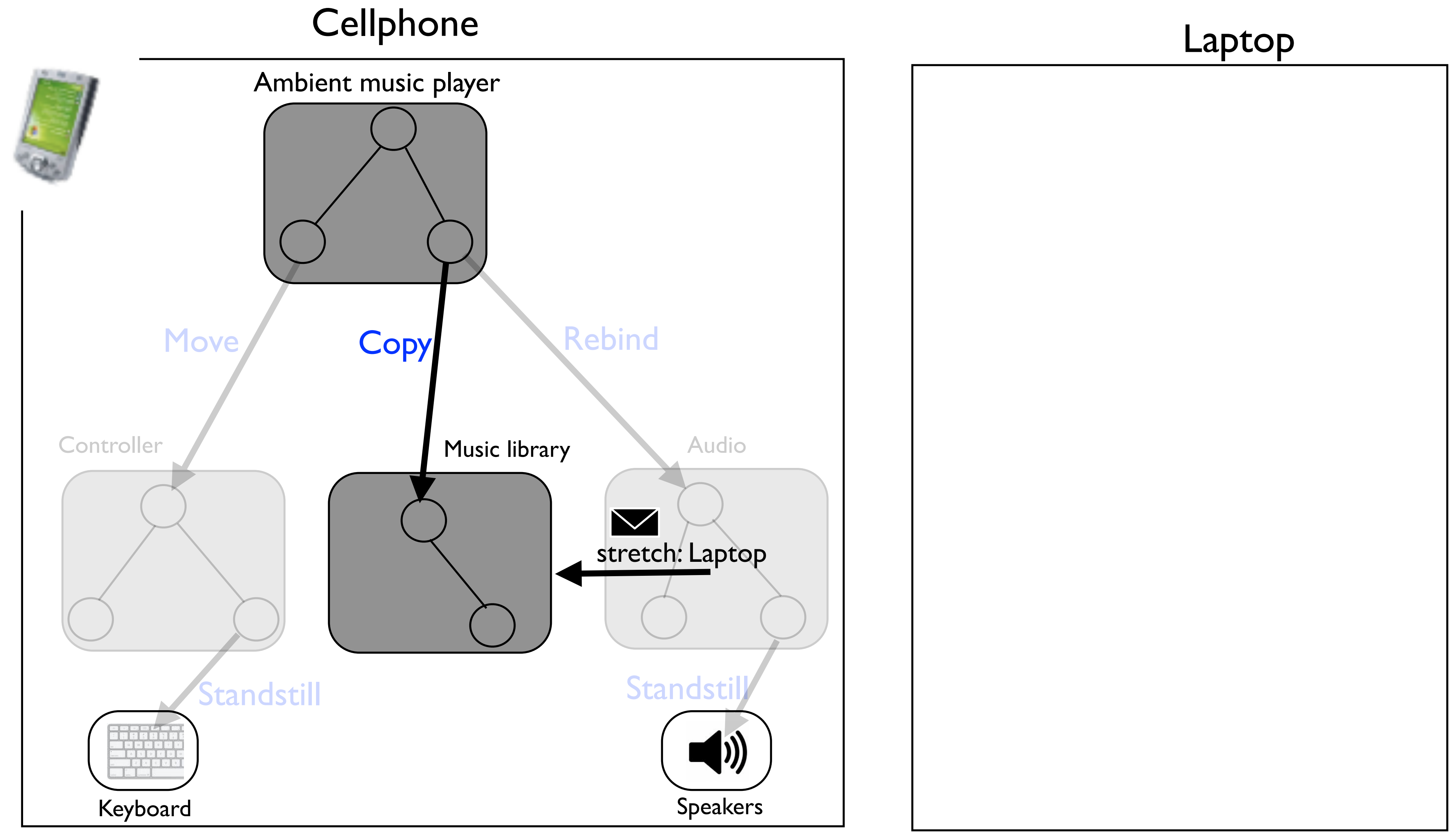
Retract under Rebind Strategy



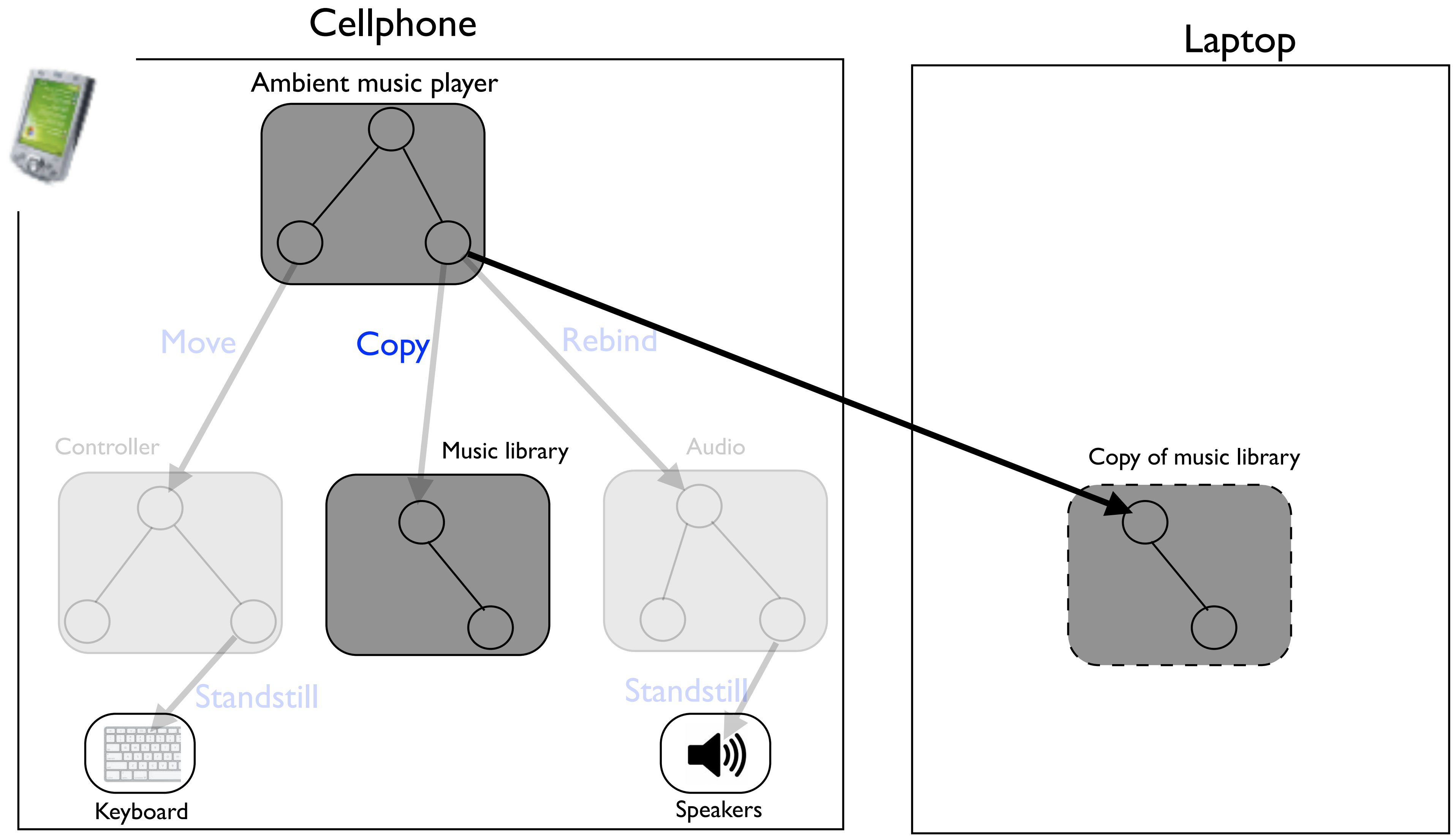
Retract under Rebind Strategy



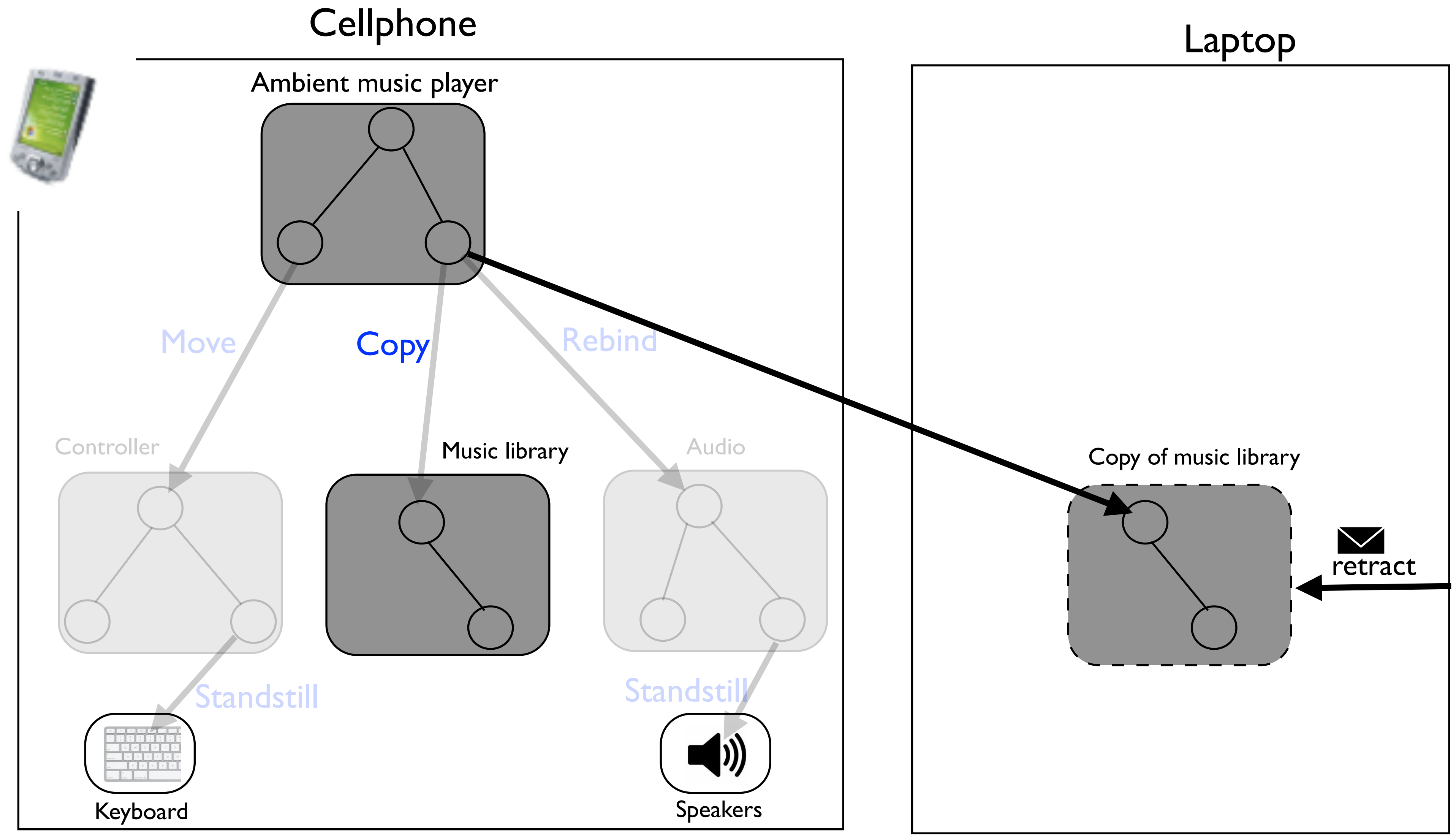
Stretch under Copy Strategy



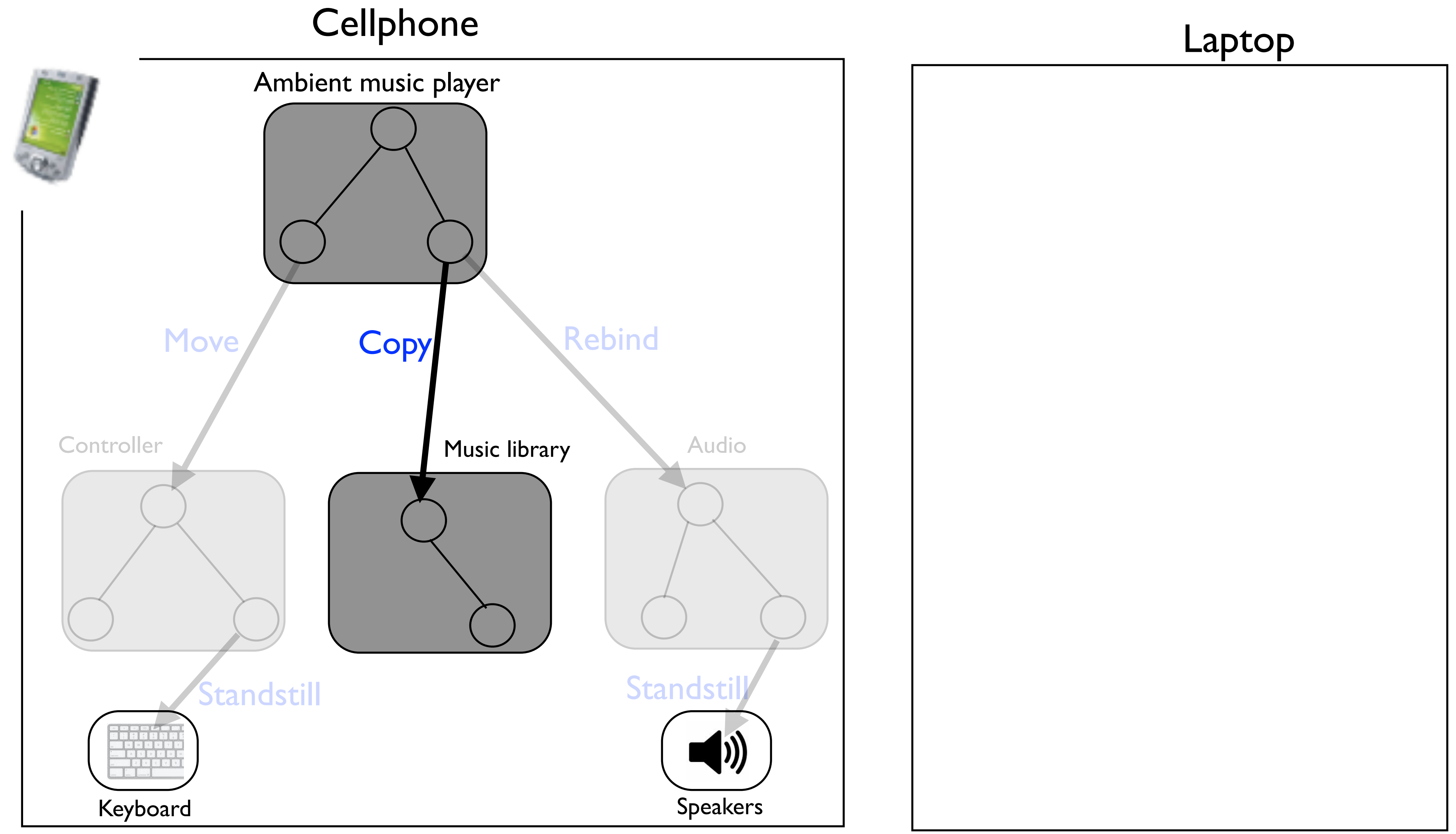
Stretch under Copy Strategy



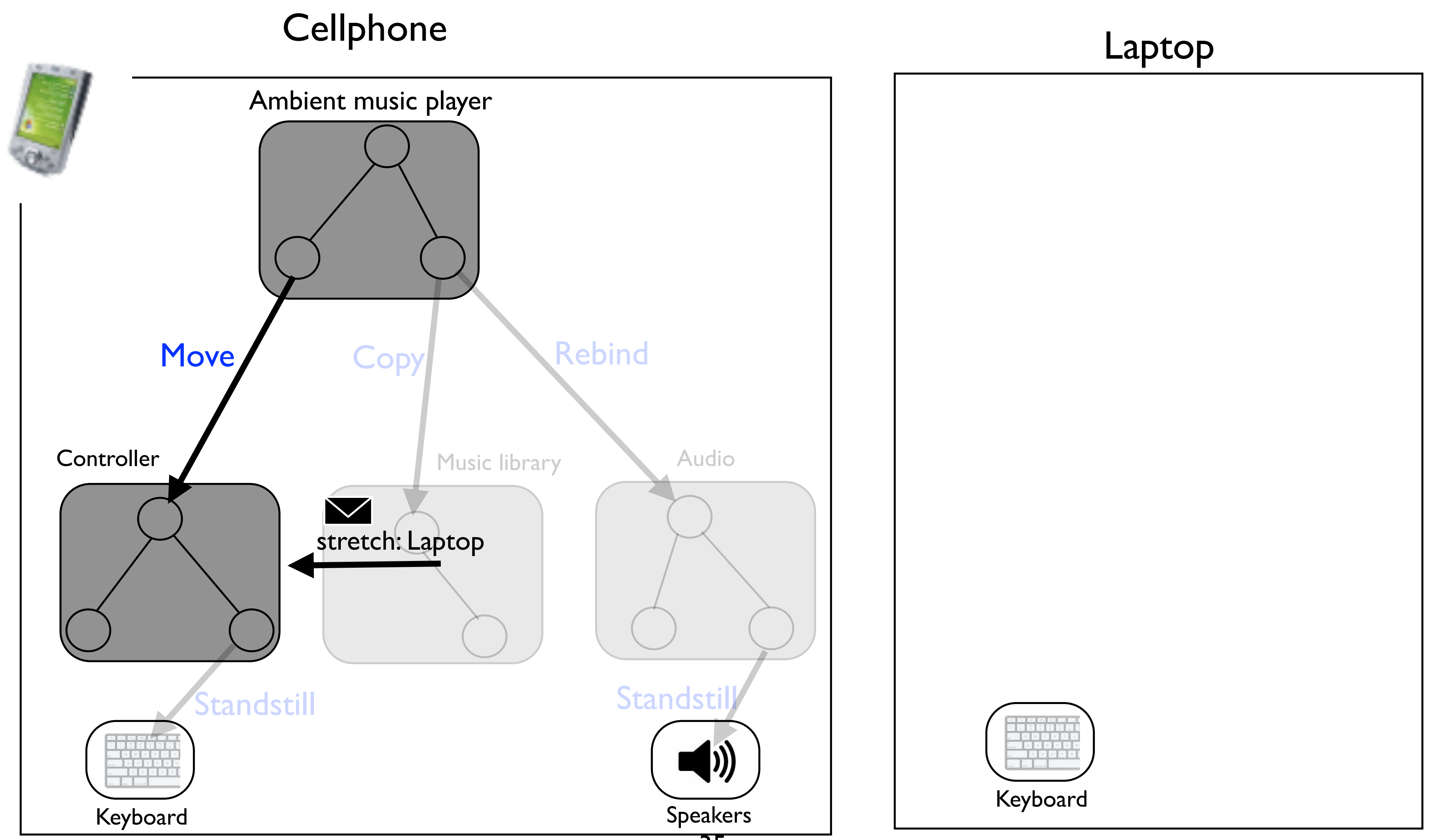
Retract under Copy Strategy



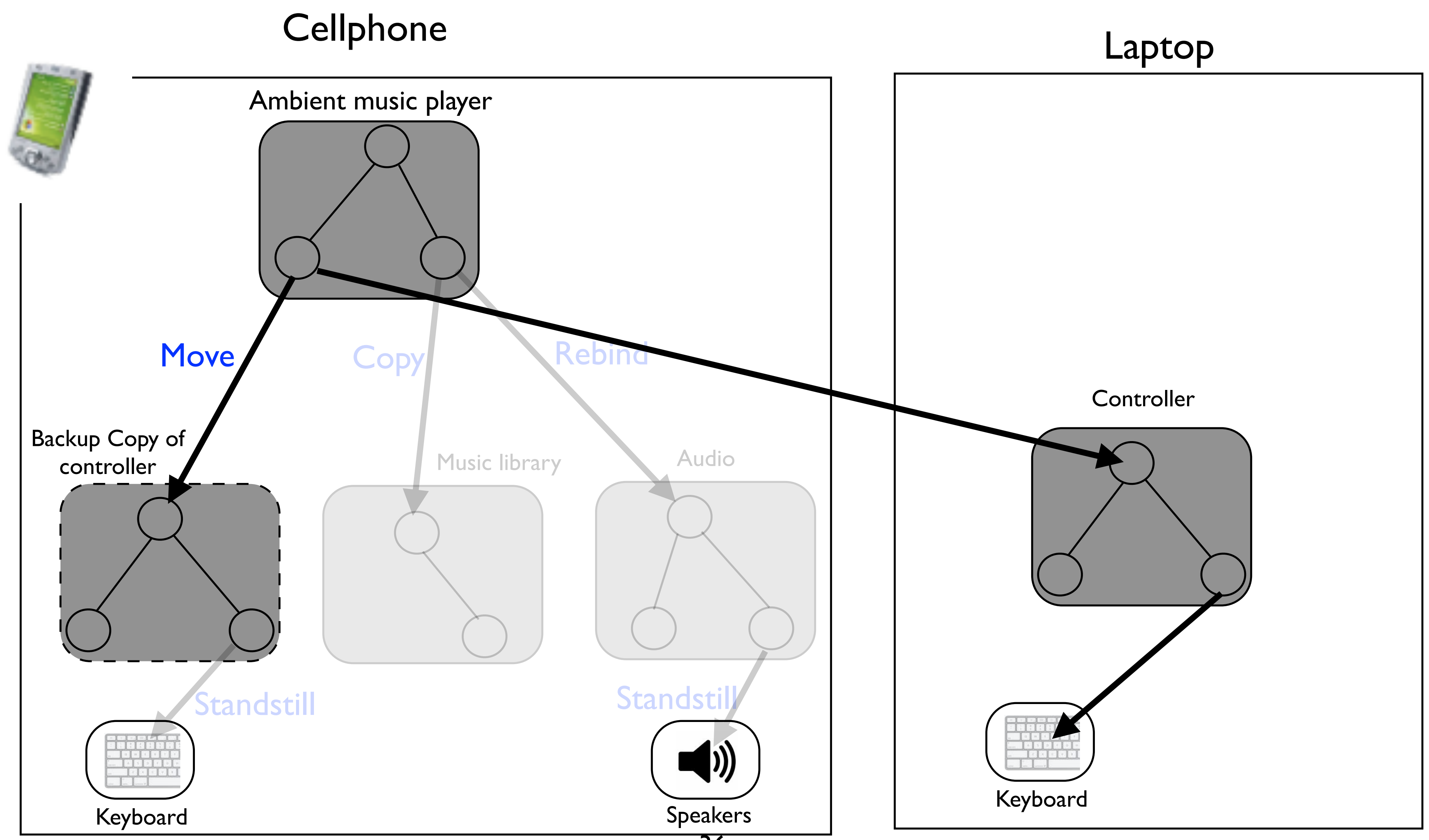
Retract under Copy Strategy



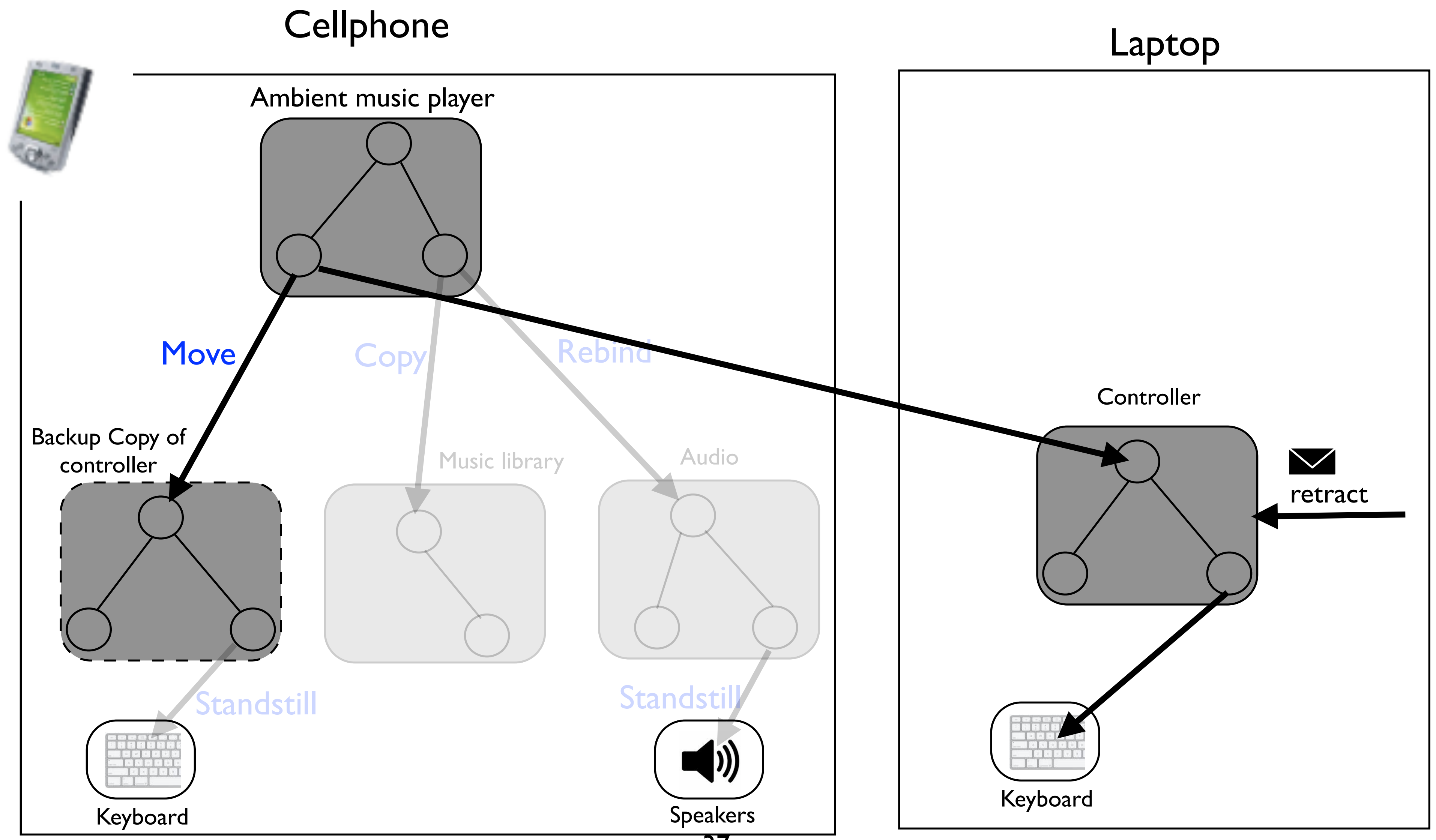
Stretch under Move Strategy



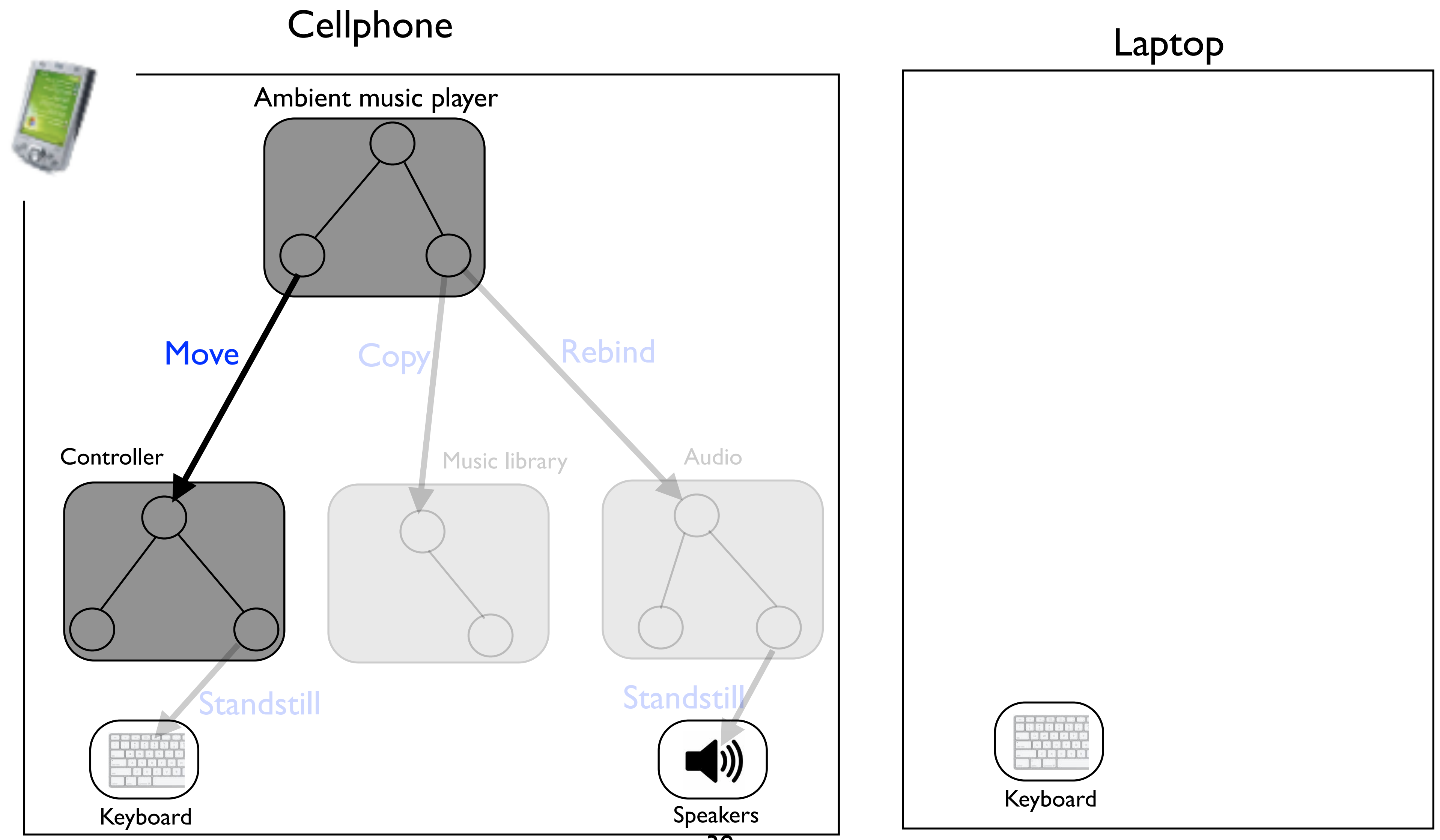
Stretch under Move Strategy



Retract under Move Strategy



Retract under Move Strategy



Resilient Actors in AmbientTalk

AmbientTalk (Van Cutsem et. al, 2007)

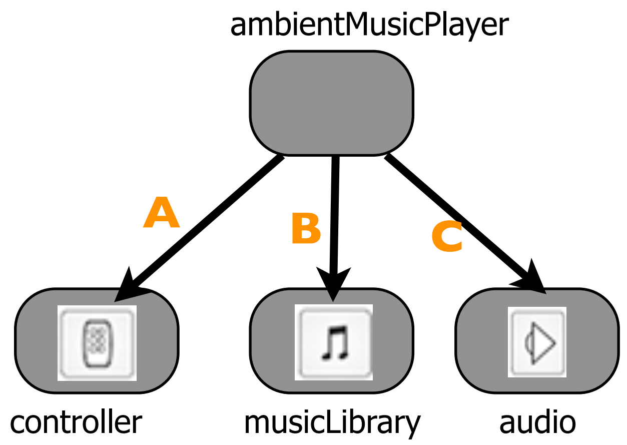
- An actor-based language for pervasive computing environments
- Publish/Subscribe service discovery
- Network failure handling mechanisms

Four language constructs for service partitioning:

actor: resilientAs: and **bindTo: resilientAs:**

stretch: and **retract:**

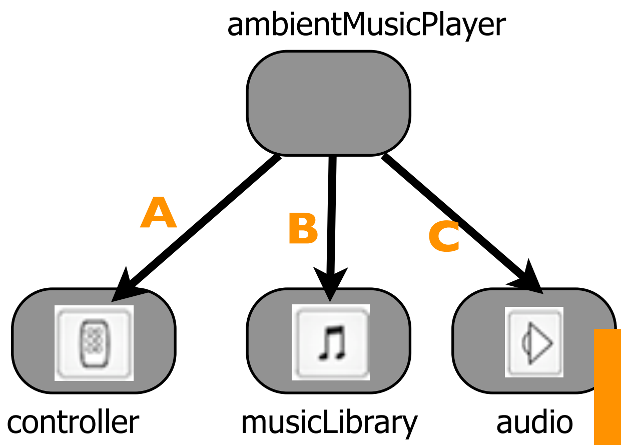
Example: Ambient music player



```
def controller := actor: {
   def theKeyboard := ...;
   def getInput() { ... };
   ....
} resilientAs: [move];

def musicLibrary := actor: {
   def myLib := Vector.new();
   def getPlayList(){ ... }
   ....
} resilientAs: [copy];

def audio := actor: {
    def theSpeaker := ..;
    .....
} resilientAs: [move];
```

```
def ambientMusicPlayer :=actor:{
    |controller, audio, musicLibrary|                       A
   def theController := bindTo: controller  resilientAs:[move];
                                                                  B
   def theAudio := bindTo: audio resilientAs: [rebind(audioService)];

   def theMusicLib := bindTo: musicLibrary resilientAs: [copy]; C

} resilientAs: [standstill];
```

Example: Ambient music player

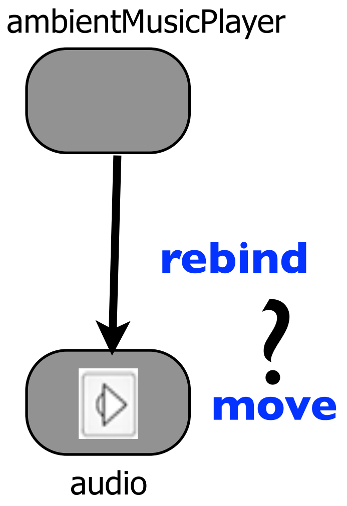


```
def controller := actor: {
   def theKeyboard := ...;
   def getInput() { ... };
   ....
} resilientAs: [move];

def audio := actor: {
    def theSpeaker := ..;
    .....
} resilientAs: [move];
```

```
def Hi-FiSystem := ...//reference to a Hi-Fi system actor
audio <-  stretch: Hi-FiSystem;

audio <- retract;
```

```
def ambientMusicPlayer :=actor:{
    |controller, audio, musicLibr
   def theController := bindTo: controller  resilientAs:[move];
                                                           B
   def theAudio := bindTo: audio resilientAs: [rebind(audioService)];

   def theMusicLib := bindTo: musicLibrary resilientAs: [copy];C

} resilientAs: [standstill];
```

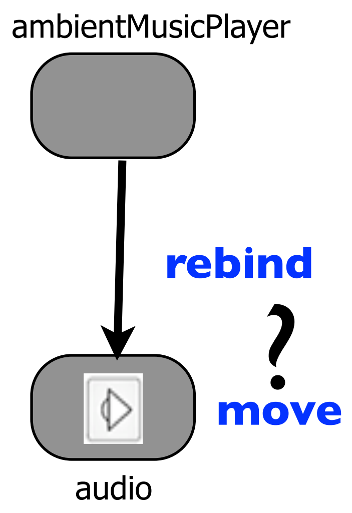
Resolution of Strategies



```
def audio := actor: { .....} resilientAs: [move];
```

```
def ambientMusicPlayer :=actor:{
   def theAudio := bindTo: audio resilientAs: [rebind(audioService)];

} resilientAs: [standstill];
```

Resolution of Strategies



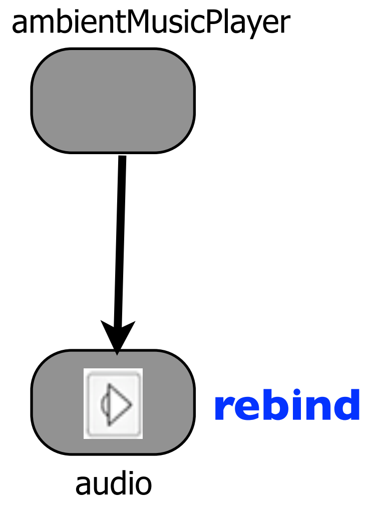
```
def audio := actor: { .....} resilientAs: [move];
```

```
def ambientMusicPlayer :=actor:{
   def theAudio := bindTo: audio resilientAs: [rebind(audioService)];

} resilientAs: [standstill];
```

Conflict resolution mechanism

Elastic binding strategy	**Resilient actor strategy**
{rebind, standstill}	{resilience strategy}*
{move, copy}	{resilience strategy}*

Resolution of Strategies



```
def audio := actor: { .....} resilientAs: [move];
```

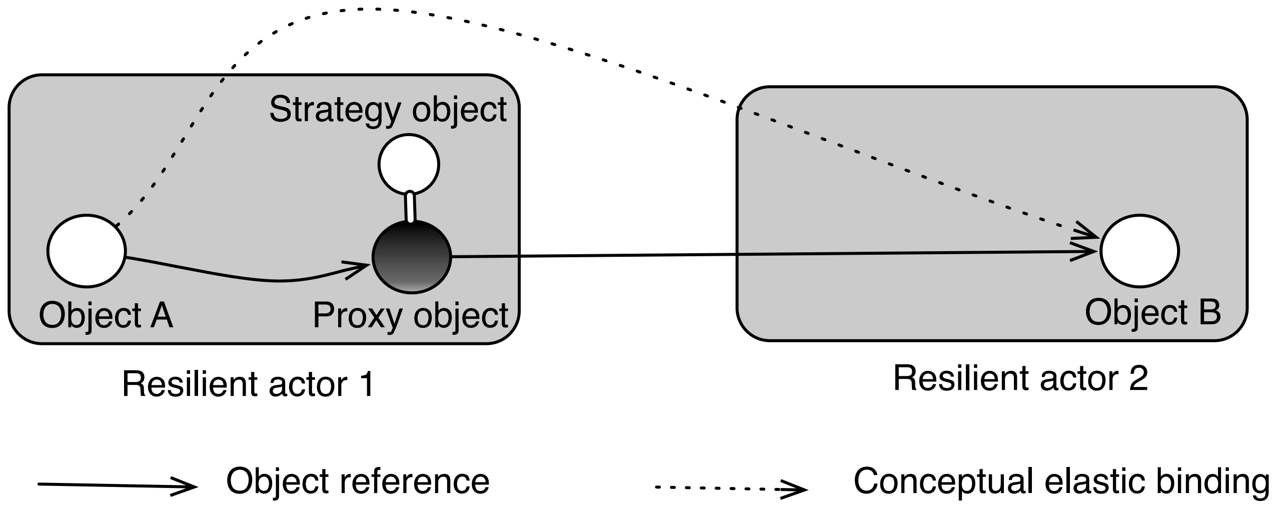
```
def ambientMusicPlayer :=actor:{
   def theAudio := bindTo: audio resilientAs: [rebind(audioService)];

} resilientAs: [standstill];
```

Conflict resolution mechanism

Elastic binding strategy	Resilient actor strategy
{rebind, standstill}	{resilience strategy}*
{move, copy}	{resilience strategy}*

Implementation

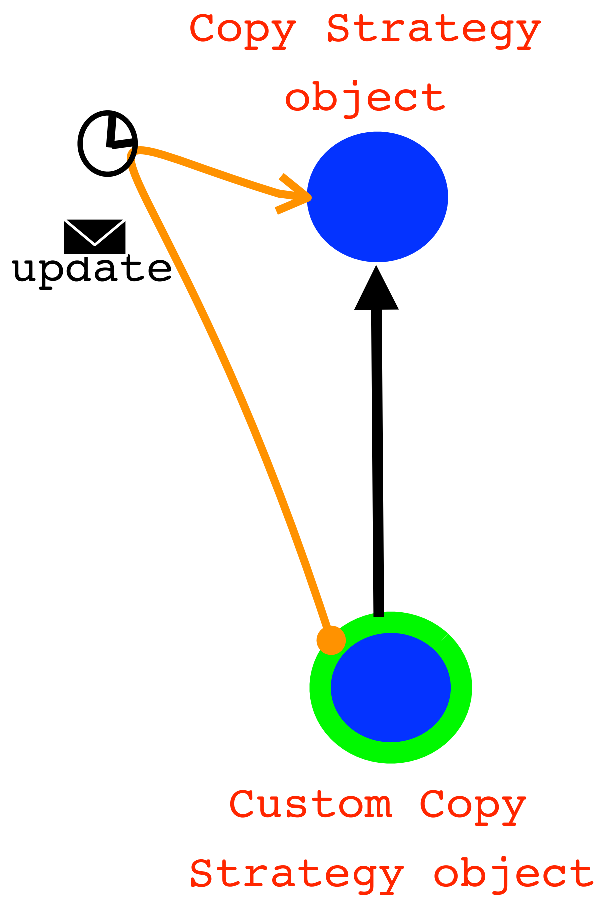
- Reflectively implemented on top of AmbientTalk
- Resilience strategies as objects
- Elastic bindings as proxy objects



Extensible Implementation

Custom Resilience Strategies

e.g Towards proactive replication



```
def copyStrategyExtension := extend:
 copyStrategy with: {
    def time := 10;
    def stretch: location {
     super^stretch: location;
      whenever: seconds(time) elapsed: {
       //...update state
       };
    };
};
```

In Summary

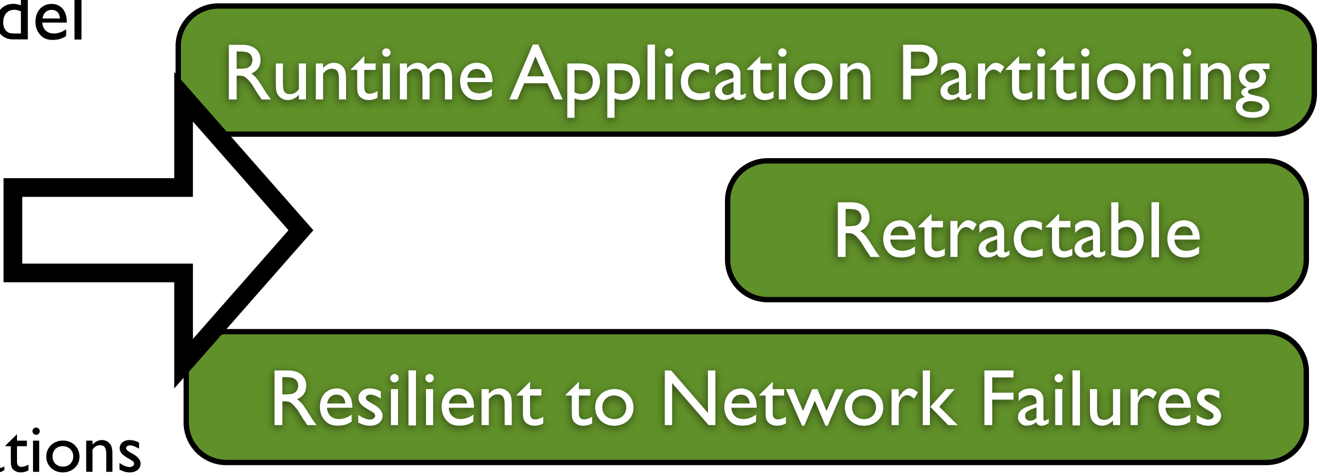
- Need for Resilient Partitioning of Pervasive Computing Services

In Summary

- Need for Resilient Partitioning of Pervasive Computing Services
- Resilient Actor Model
    - Resilient actors
    - Elastic bindings
    - Partitioning operations

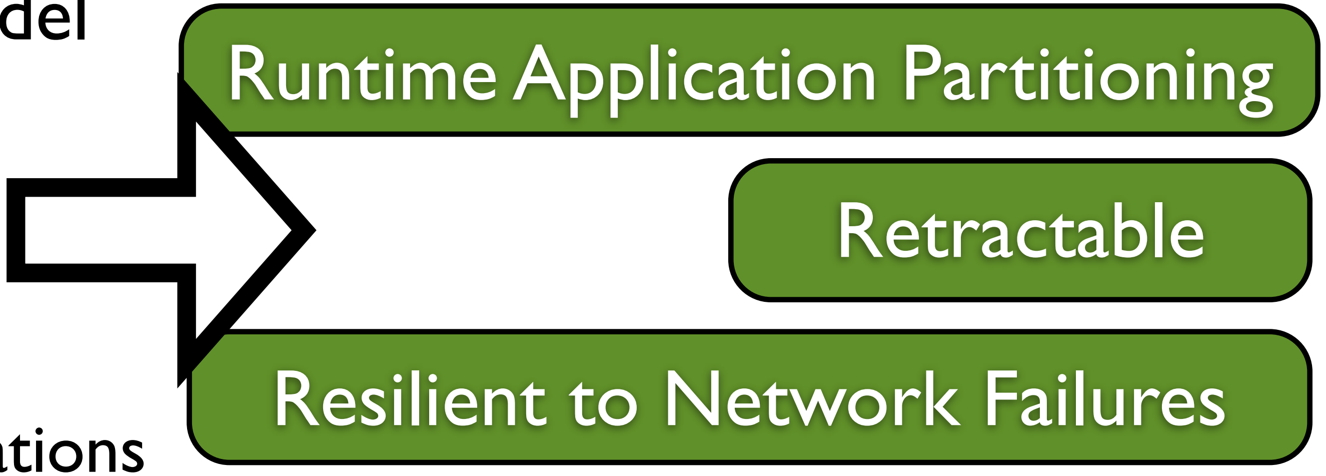
In Summary

- Need for Resilient Partitioning of Pervasive Computing Services
- Resilient Actor Model
  - Resilient actors
  - Elastic bindings
  - Partitioning operations


Runtime Application Partitioning
Retractable
Resilient to Network Failures

In Summary

- Need for Resilient Partitioning of Pervasive Computing Services
- Resilient Actor Model
  - Resilient actors
  - Elastic bindings
  - Partitioning operations


Runtime Application Partitioning
Retractable
Resilient to Network Failures

- Extensible Implementation (Resilience strategies)
- Formal Definition (see *paper*)

Thank You

ebainomu@vub.ac.be

http://prog2.vub.ac.be/~ebainomu/