Aglets

Programming Mobile Agents in Java

A Technology Project

Pat O'Connor

http://www.ibm.com/java
IBM United Kingdom Laboratories

Outline

- Motivation
- What are Aglets?
- Technical Overview of the Aglets API
- Potential Applications

Motivation

- *Making Agent Technology Real*
  - Open and Pervasive

    Freely available, Java, Standards
  - Real application

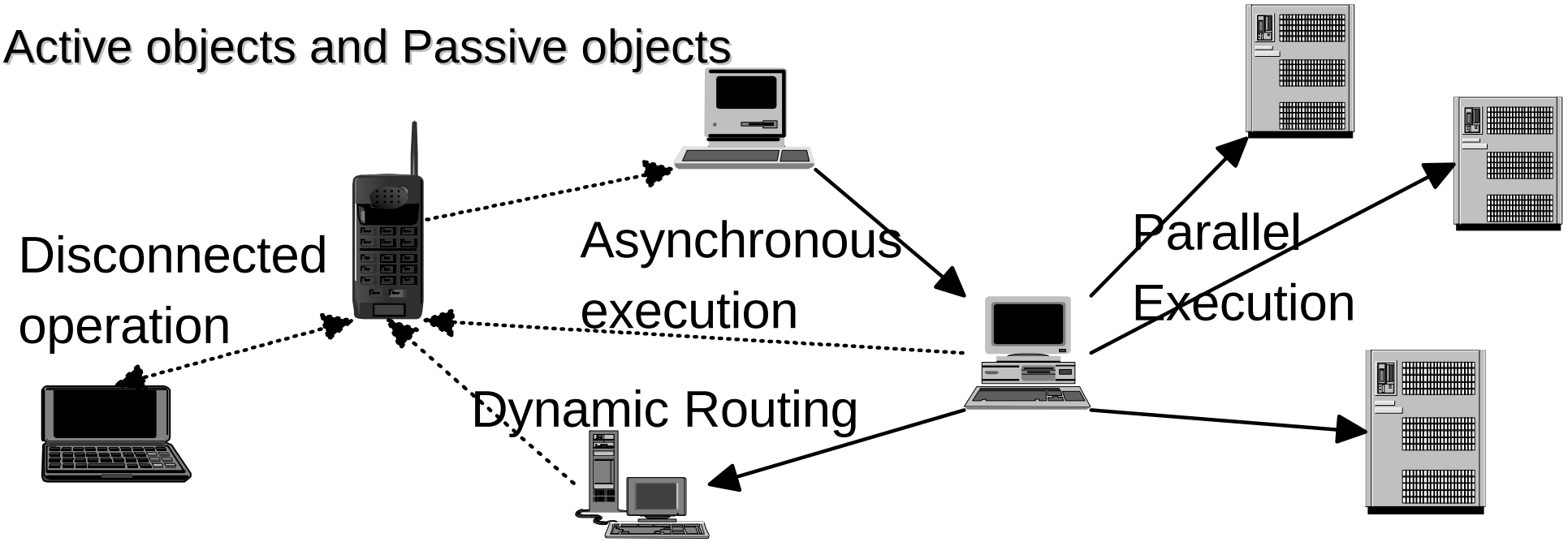
    Ability to access external resources - DB, GUI, etc.
  - Internet

    Internet enabled agents
  - User Interface

    Friendly, Compelling Interface

What are Aglets?

- Single uniform paradigm for distributed computing
  - Asynchronous and Synchronous
  - Object-passing and Message-passing
  - Mobile objects and Stationary objects
  - Active objects and Passive objects



What are Aglets? (cont'd)

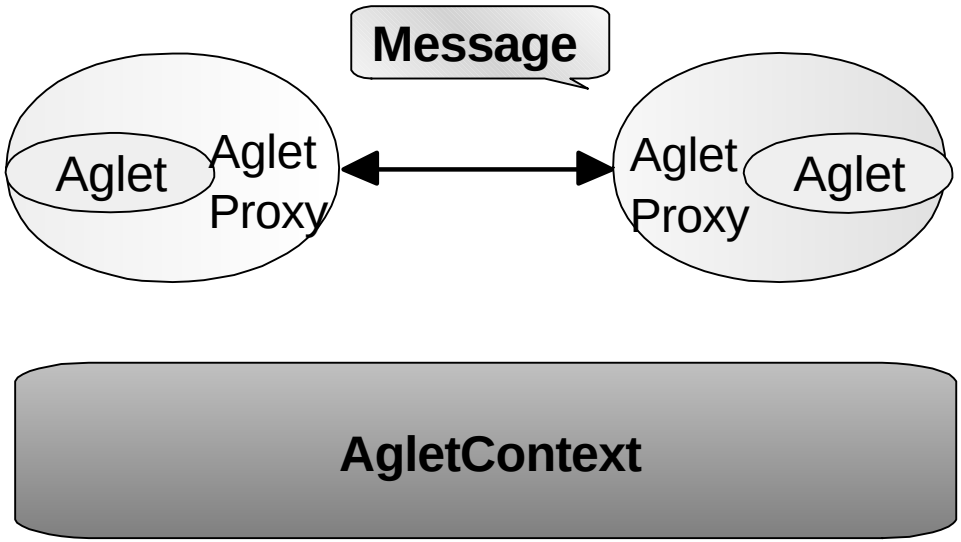
- Runtime/Development kit for Java based mobile agents
  - Aglets Application Programming Interface (AAPI)
- Autonomous execution
  - Decides what to do, where to go, and when to do/go
- Platform-independence
  - Totally written in Java
  - Create once, go everywhere!

Technical Overview of Aglets

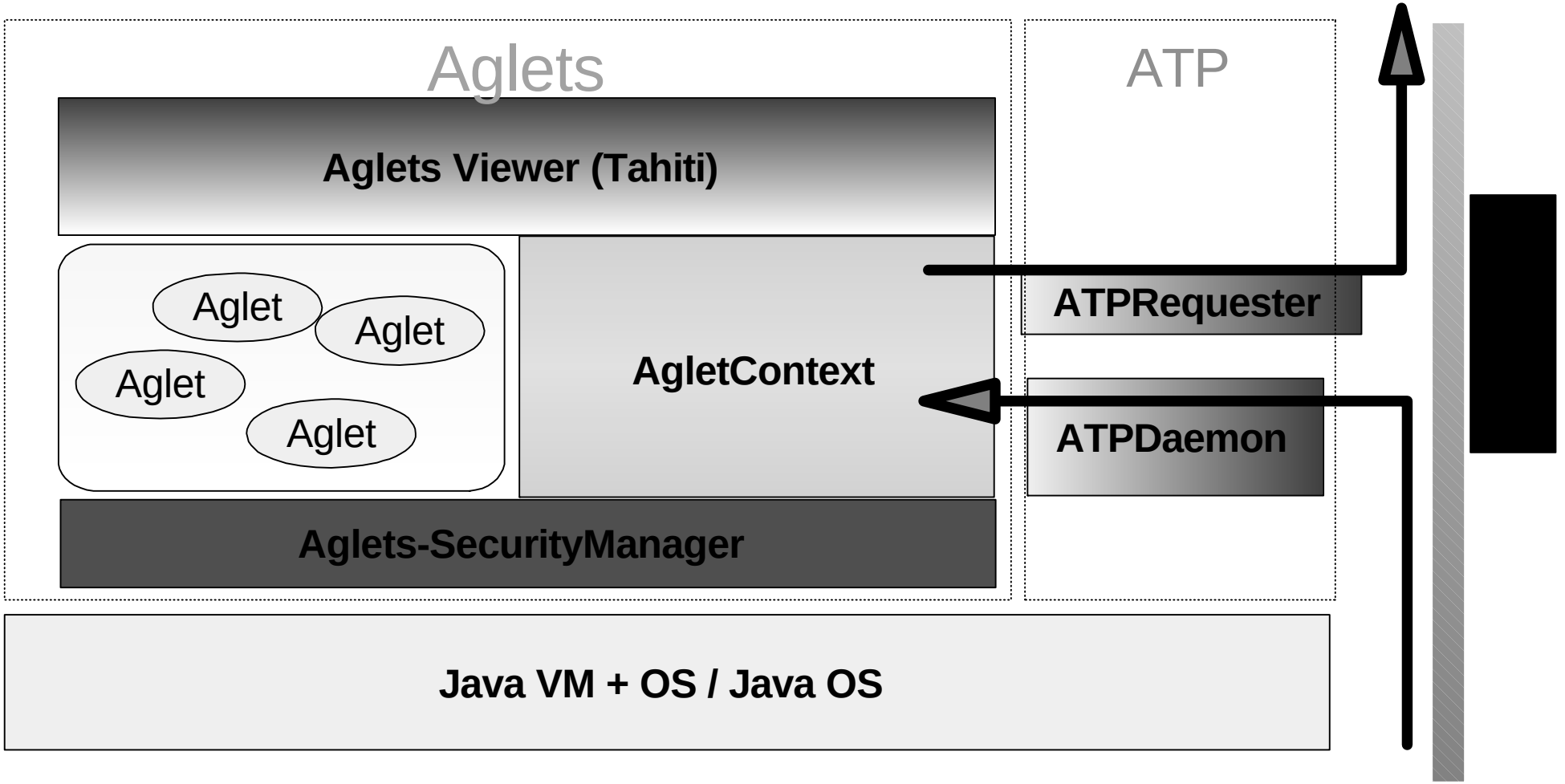
- AAPI and Aglets Runtime Framework
- Agent Transfer Protocol Overview
- Aglets Object Model
- Patterns

Aglet API (AAPI)

- Aglet
  - Aglet
  - AgletIdentifier
  - Itinerary
- AgletProxy
- AgletContext
- Message
  - Message
  - FutureReply
  - MessageManager

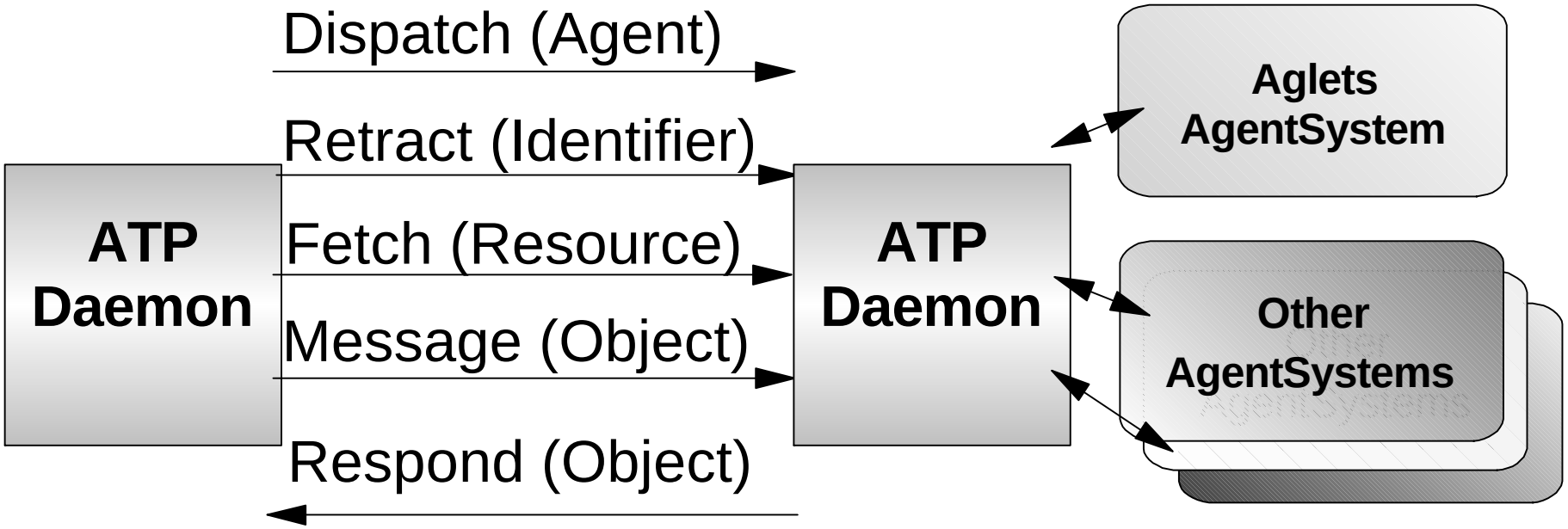


Aglets Runtime Framework



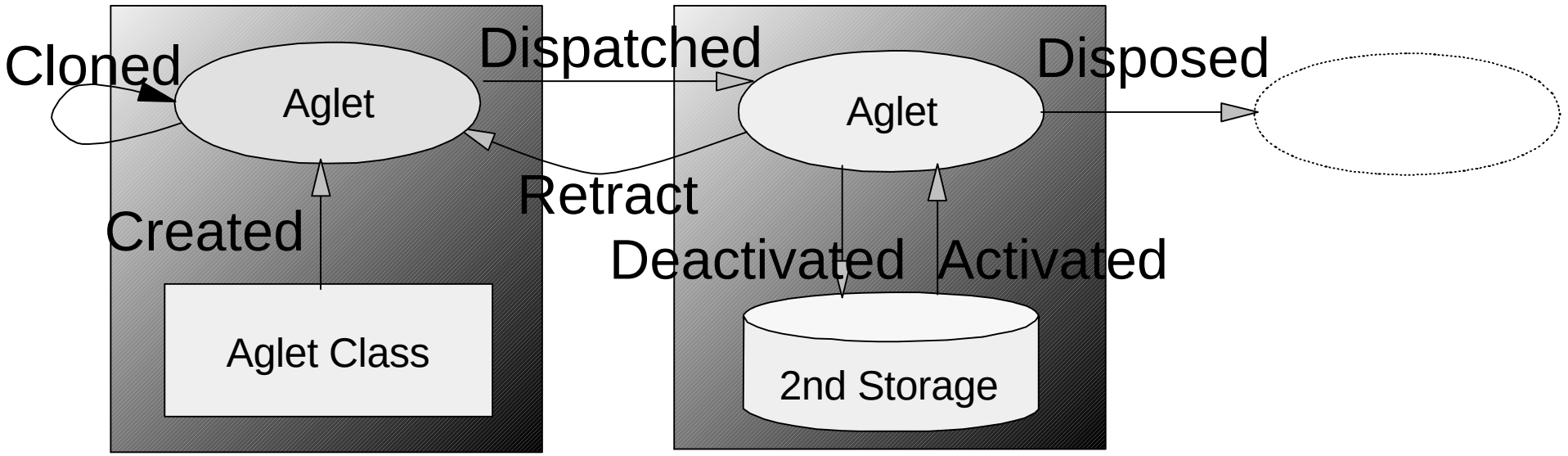
Agent Transfer Protocol

- Simple Request-Response type Protocol for agents



Life Cycle Events of an Aglet

- The aglet may be:
  - Created / Cloned in a context
  - Dispatched to / Retracted from a destination context
  - Deactivated into / Activated from storage
  - Disposed

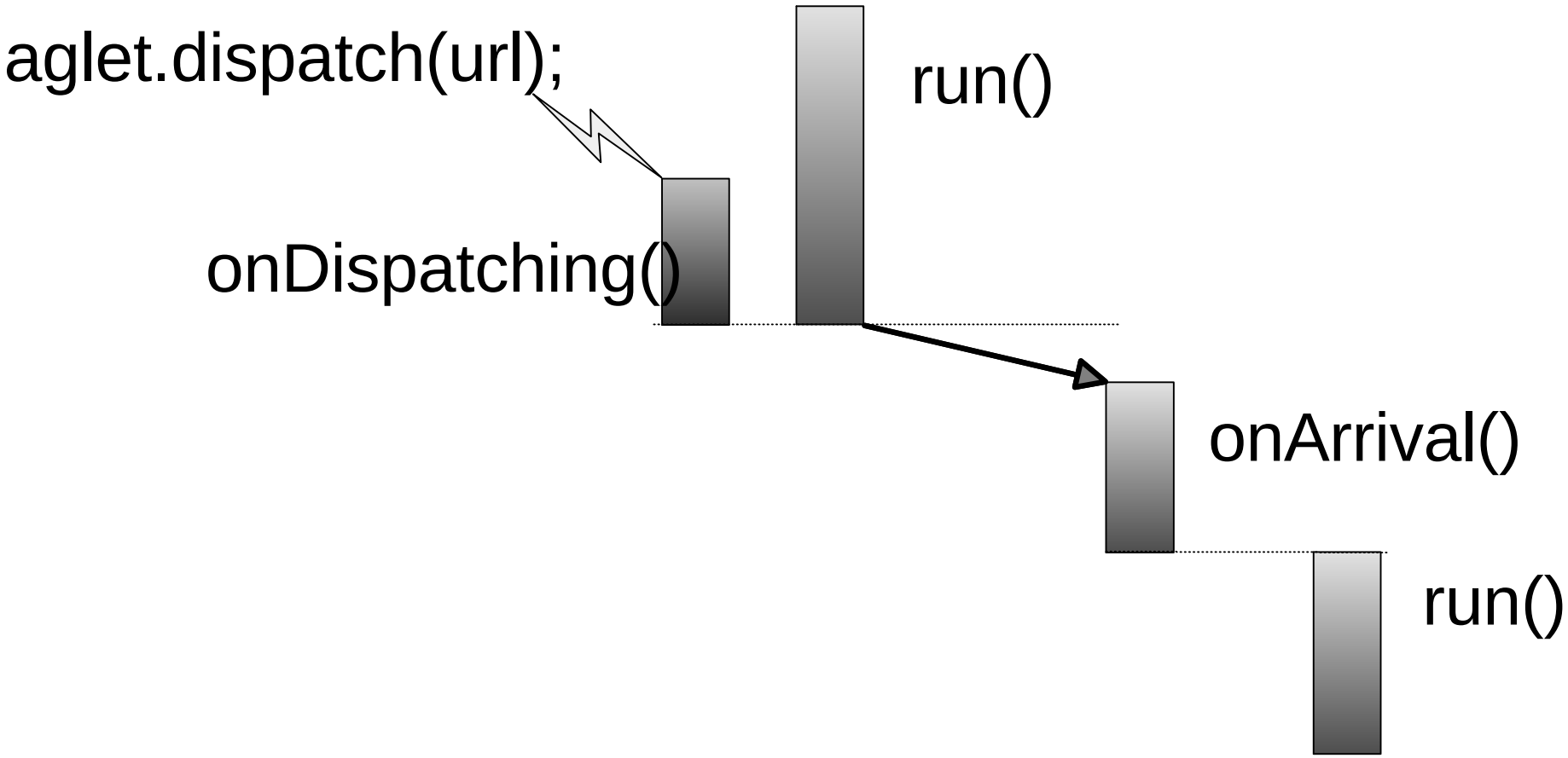


Aglet Object Model

- Aglets doesn't migrate a process
- Callback (Event Driven) approach
  - onCreation, onArrival, etc...

The event	*about to take place*	*After the event has taken place*
Creation		onCreation()
Clone	onCloning()	onClone()
Dispatch	onDispatching()	onArrival()
Retract	onReverting()	onArrival()
Dispose	onDisposing()	
Deactivate	onDeactivating()	
Activate		onActivation()
Message		handleMessage()

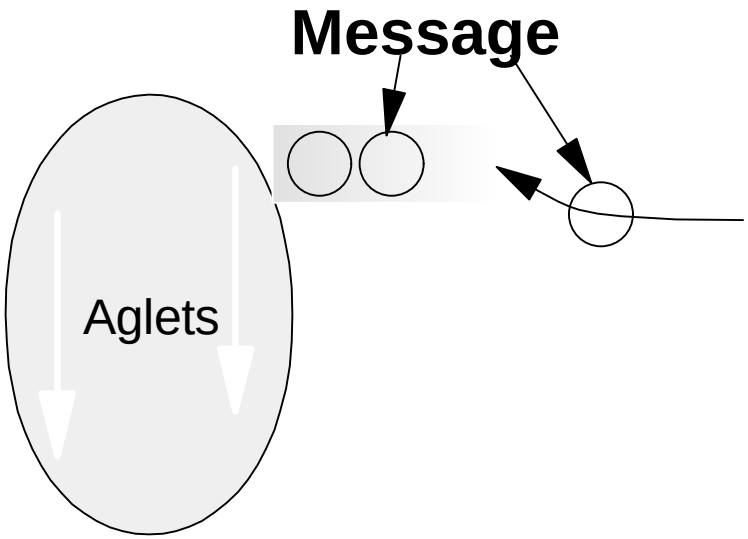
Dispatch



```
import aglet.*;
public class MyAglet extends Aglet {
    URL origin;
    public void onCreation(Object init) {
        origin = getAgletContext().getHostingURL();
    }
    public void onDispatching(URL urlb) {
        setText("ByeBye..");
    }
    public void onArrival(URL url) {
        if (url.equals(origin)) {
            setText("I'm back!");
        } else {
            doJob();
            dispatch(origin);
        }
    }
}
```

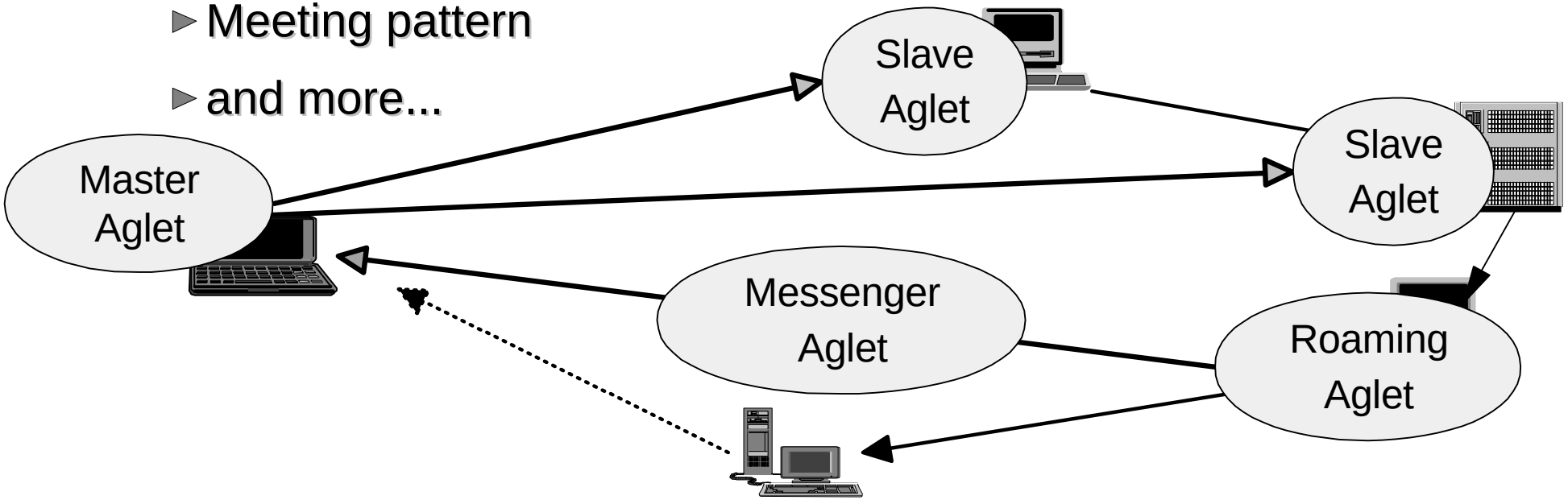
Aglet Object Model (cont'd)

- An aglet can be persistent
  - Deactivation/Activation
- Communication by message passing
  - Future type message passing
  - Subscribe/Multicast in a context
- An aglet can have multiple activities
  - Multiple threads in an aglet



Patterns

- Common usage patterns for agents
  - Master-Slave pattern
  - Messenger pattern
  - Meeting pattern
  - and more...



Messaging in Aglets

```
AgletProxy proxy = ....;

Message m = new
Message("hello");
FutureReply future =
proxy.sendMessage(m);
.....
Object reply = future.getReply();
```

```
boolean
handleMessage(Message m) {
    if ("hello".equals(m.kind)) {
        m.sendReply("");
        return true;
    }
    return false;
}
```

Potential Applications....

- Network system management
- Database access
- Auction
- Shopping mall

Contact Information

- Latest news, updates, tutorials, etc.:

  http://www.ibm.co.jp/aglets

- Public mailing list (discussions, questions, etc.):

  aglets@javalounge.com

- Contacting the Aglets team, Bug reports:

  aglets@yamato.ibm.co.jp

Any Questions?