

Peer-To-Peer and Fault-Tolerance: Towards Deployment-Based Technical Services

Denis Caromel, Christian Delbe
and Alexandre di Costanzo



Roadmap

- Context

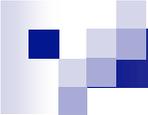
- ProActive, a Java Middleware
- Deployment in ProActive

- P2P infrastructure

- Fault-Tolerance

- Mixing P2P and Fault-Tolerance

- Problems and solutions



ProActive: A Java API + Tools for Parallel, Distributed Computing

- A uniform framework: *An Active Object* pattern
- A formal model: *Determinism, Insensitivity to deployment*
- Programming Model:
 - Remote Objects (Classes, not only Interfaces, Dynamic)
 - Asynchronous Communications, Automatic dataflow synchro: Futures
 - Groups, Mobility, Components, Security
 - Environment:
 - **XML Deployment Descriptors**
 - Interfaced with various protocols: rsh,ssh,LSF,Globus,Jini,RMIregistry
 - Visualization and monitoring: *IC2D*

*In the ObjectWeb Consortium (Open Source middleware)
since April 2002 (LGPL license)*



Abstract Deployment Model

- Difficulties and lack of flexibility in deployment
 - Avoid deployment specific source code
 - Avoid scripting for configuration, getting nodes, connecting, etc.
- A key principle: **Abstract Away** from source code
 - Machines
 - Creation Protocols
 - Lookup and Registry Protocols

XML deployment file ↔ **Virtual Node (VN)** ↔ Application

- Supported protocols:
 - Globus, ssh, rsh, LSF, PBS, ... Web Services, WSRF, ...



Virtual Nodes and Nodes

- Virtual Node (VN):
 - Identified as a string name
 - Used in program source
 - Configured (mapped) in an XML descriptor file
- Node:
 - ProActive execution environment
 - Mapping of VN to JVMs leads to Node

Program source	Descriptor (runtime)
Activities → Virtual Nodes	Virtual Nodes → JVMs → Nodes

A Deployment Descriptor

Definitions and mapping

```
<virtualNodesDefinition>
  <virtualNode name="vn1"/>
</virtualNodesDefinition>
<map virtualNode="vn1">
  <jvmSet>
    <vmName value="Jvm1"/>
    <vmName value="Jvm2"/>
  </jvmSet>
</map>
<jvms>
  <jvm name="Jvm1">
    <creation>
      <processReference
        refid="localJVM"/>
    </creation>
  </jvm>
  <jvm name="Jvm2">
    <creation>
      <processReference
        refid="ssh_cluster"/>
    </creation>
  </jvm>
</jvms>
```

**Definition of
Virtual Nodes**

**Mapping of
Virtual Nodes**

A Deployment Descriptor

Definitions and mapping

```
<virtualNodesDefinition>
  <virtualNode name="vn1"/>
</virtualNodesDefinition>
<map virtualNode="vn1">
  <jvmSet>
    <vmName value="Jvm1"/>
    <vmName value="Jvm2"/>
  </jvmSet>
</map>
<jvms>
  <jvm name="Jvm1">
    <creation>
      <processReference
        refid="localJVM"/>
    </creation>
  </jvm>
  <jvm name="Jvm2">
    <creation>
      <processReference
        refid="ssh_cluster"/>
    </creation>
  </jvm>
</jvms>
```

Definition of
Virtual Nodes

Mapping of
Virtual Nodes

A Deployment Descriptor

Definitions and mapping

```
<virtualNodesDefinition>
  <virtualNode name="vn1"/>
</virtualNodesDefinition>
<map virtualNode="vn1">
  <jvmSet>
    <vmName value="Jvm1"/>
    <vmName value="Jvm2"/>
  </jvmSet>
</map>
<jvms>
  <jvm name="Jvm1">
    <creation>
      <processReference
        refid="localJVM"/>
    </creation>
  </jvm>
  <jvm name="Jvm2">
    <creation>
      <processReference
        refid="ssh_cluster"/>
    </creation>
  </jvm>
</jvms>
```

Definition of
Virtual Nodes

Mapping of
Virtual Nodes

A Deployment Descriptor

Definitions and mapping

```
<virtualNodesDefinition>
  <virtualNode name="vn1"/>
</virtualNodesDefinition>
<map virtualNode="vn1">
  <jvmSet>
    <vmName value="Jvm1"/>
    <vmName value="Jvm2"/>
  </jvmSet>
</map>
<jvms>
  <jvm name="Jvm1">
    <creation>
      <processReference
        refid="localJVM"/>
    </creation>
  </jvm>
  <jvm name="Jvm2">
    <creation>
      <processReference
        refid="ssh_cluster"/>
    </creation>
  </jvm>
</jvms>
```

Definition of
Virtual Nodes

Mapping of
Virtual Nodes

A Deployment Descriptor

Definitions and mapping

```
<processDefinition id="localJVM">  
  <jvmProcess class="process.JVMNodeProcess"/>  
</processDefinition>
```

JVM on the current Host

```
<processDefinition id="ssh_cluster">  
  <sshProcess class="process.ssh.SSHProcess"  
    hostname="Node1.cluster.inria.fr">  
    <processReference refid="localJVM"/>  
  </sshProcess>  
</processDefinition>
```

JVM started using SSH



P2P Infrastructure: Motivations and Goals

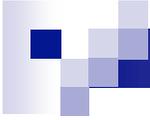
- Using spare CPU cycles of desktop machines:
 - Host not available all the time
 - Used by their normal users
- Providing a permanent shared JVMs network for computing
- Not a new communication protocol, not a DHT
- Self-Organized and Configurable



The P2P Infrastructure

- **Dynamic environment:**
 - Bootstrapping (First contact)
 - Discovering peers
 - Acquiring Computational nodes

- **Self Organized and Configurable:**
 - Time To Update (TTU): peers availability
 - Number Of Acquaintances (NOA): keep the infrastructure up
 - Time To Live (TTL): in host hop for message life
 - First Gnutella message protocol version inspired



P2P in Deployment Descriptor

```
<virtualNodesDefinition>
  <virtualNode name="p2pSlaves" property="multiple"/>
</virtualNodesDefinition>
<mapping>
  <map virtualNode="p2pSlaves">
    <jvmSet>
      <vnName value="slaves"/>
    </jvmSet>
  </map>
</mapping>

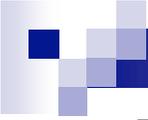
...

<jvm name="slaves">
  <acquisition>
    <aquisitionReference refid="p2pLookup"/>
  </acquisition>
</jvm>
```



P2P in Deployment Descriptor

```
<infrastructure>
  <acquisition>
    <acquisitionDefinition id="p2pLookup">
      <P2PService NodesAsked="MAX">
        <peerSet>
          <peer>rmi://registry1:3000</peer>
          <peer>rmi://registry2:3000</peer>
        </peerSet>
      </P2PService>
    </acquisitionDefinition>
  </acquisition>
</infrastructure>
```



Fault-tolerance in ProActive

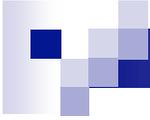
- **Rollback-Recovery** fault-tolerance
 - After a failure, revert the system state back to some earlier and correct version
 - Based on periodical checkpoints of the active objects
- **Two protocols** are implemented
 - Communication Induced Checkpointing (CIC)
 - + Low failure free overhead
 - Slow recovery
 - Pessimistic Message Logging (PML)
 - Higher failure free overhead
 - + Fast recovery
- **Choose best strategy at deployment time**
 - No FT concerns in source code
 - FT configuration in the Nodes
 - Set in deployment descriptor



Mixing P2P and FT: Towards Technical Services

- **Two kind** of nodes
 - Created at deployment time
 - Configured by the deployer
 - Acquired through the P2P infrastructure
 - Statically configured by the administrator

- Provide a **simple and unified** configuration mechanism
 - In the deployment descriptor
 - Attached to the *Virtual Node*



Technical Services

- In Deployment Descriptor :
 - Define configuration in a technical service
 - Apply a technical service on a virtual node
- Virtual Node abstracts the nature of a node
 - The configuration is similarly applied on a created and acquired node
- Provide a simple API to propose new technical services



Technical Services

Deployment
Descriptor

```
<virtualNodesDefinition>
  <virtualNode name="vn1" serviceRefid="service1"/>
  <virtualNode name="vn2" serviceRefid="service2"/>
</virtualNodesDefinition>

...

<technicalServiceDefinitions>
  <service id="service1" class="services.Service1">
    <arg name="name1" value="value1"/>
    <arg name="name2" value="value2"/>
    ...
  </service>
  <service id="service2" class="services.Service2">
    ...
  </service>
</technicalServiceDefinitions>
```

Library

```
public interface TechnicalService {
  public void init(HashMap argValues);
  public void apply(Node node);
}
```



Conclusion

- We propose a solution for dynamic configuration :
 - Apply the most adapted configuration regarding the execution environment
- Currently adding load balancing as Technical Service
- Combining Technical Services?