

Verwendung der Gruppenkommunikation JGroups

Entwickelt von Bela Ban als flexibel konfigurierbare Gruppenkommunikation in Java

Information und Sourcen

- ▶ Webseite: <http://www.jgroups.org/javagroupsnew/docs/index.html>
- ▶ Manual und Tutorial
- ▶ CIP: /local/JGroups/

Verwendung

- ▶ `jgroups-all.jar` die eigentliche Bibliothek
- ▶ `commons-logging.jar` Unterstützung für Logging

Verwendung der Gruppenkommunikation JGroups

JChannel

- ▶ Verbindung zur Gruppe
- ▶ Verschiedene Konstruktoren:
 - ▶ `JChannel()` – es wird die Standardkonfiguration verwendet
 - ▶ `JChannel(org.w3c.dom.Element properties)` – Konfiguration als XML Element
 - ▶ `JChannel(java.io.File properties)` – XML-Datei
 - ▶ `JChannel(java.lang.String properties)` – Verschiedene Quellen oder direkt als Zeichenkette im alten Format
- ▶ Verbinden des Channel:
 - ▶ `public void connect(String clustername) throws ChannelClosed;`
 - ▶ `public void connect(string cluster_name, address target, string state_id, long timeout) throws channelexception;`

Verwendung der Gruppenkommunikation JGroups

JChannel

► Informationen

- `public Address getLocalAddress();`
- `public String getClusterName();`
- `public View getView();`

► Versenden von Nachrichten:

- `public void send(Message msg) throws ChannelNotConnected, ChannelClosed;`
- Beispiel:

```
Address receiver;  
Message msg;  
Hashtable data;  
try {  
    receiver=channel.getView().getMembers().first();  
    channel.send(receiver, null, data);  
}catch(Exception ex) { // handle errors}
```

Verwendung der Gruppenkommunikation JGroups

JChannel

► Empfangen von Nachrichten

- `public Object receive(long timeout) throws ChannelNotConnected, ChannelClosed, Timeout;`
- Beispiel:

```
Object obj;  
Message msg;  
View v;  
obj=channel.receive(0); // wait forever  
if (obj instanceof Message)  
    msg=(Message) obj;  
    s=(String) msg.getObject();  
else if (obj instanceof View)  
    v=(View) obj;  
else  
    ; // don't handle suspicions or blocks
```

Verwendung der Gruppenkommunikation JGroups

MessageListener

- Empfang von Nachrichten und dem Zustand

```
public interface MessageListener {  
    public void receive(Message msg);  
    byte [] getState();  
    void setState(byte [] state);  
}
```

MembershipListener

- Benachrichtigung über Änderung der Gruppe

```
public interface MembershipListener {  
    public void viewAccepted(View new_view);  
    public void suspect(Object suspected_mbr);  
    public void block();  
}
```

Verwendung der Gruppenkommunikation JGroups

Receiver

- ▶ Alles was man braucht:

```
public interface Receiver extends MessageListener ,  
    MembershipListener {}
```

- ▶ Verwendung der Receiver Schnittstelle:

```
JChannel ch = new JChannel();  
ch.setReceiver(new ExtendedReceiverAdapter() {  
    public void receive(Message msg) {  
        System.out.println("received_message_" + msg);  
    }  
    public void viewAccepted(View new_view) {  
        System.out.println("received_view_" + new_view);  
    }  
    ...  
});  
ch.connect("bla");
```

Verwendung der Gruppenkommunikation JGroups

Receiver: Anforderung von Zustand

```
class MyReceiver extends ReceiverAdapter {
    final Map m=new HashMap();
    byte[] getState() {
        synchronized(m) {
            byte[] state=Util.objectToByteBuffer(m);
            return state;
        }
    }

    public void setState(byte[] state) {
        synchronized(m) {
            Map new_m=(Map) Util.objectFromByteBuffer(state);
            m.clear();
            m.addAll(new_m);
        }
    }
}
```

Verwendung der Gruppenkommunikation JGroups

View

- ▶ Enthält alle aktiven Mitglieder der Gruppe
- ▶ Wichtige Methode:

```
java.util.Vector<Address> getMembers()
```

- ▶ Beispiel: Nachricht an das erste Mitglied

```
View myview = channel.getView();  
Address first = myview.getMembers().first();  
Message msg = new Message(first, null, "Hello_world");  
channel.send(msg);
```

Verwendung der Gruppenkommunikation JGroups

Message

- ▶ Zieladresse - wenn `null` dann an alle
- ▶ Ursprungsadresse - `null` wird durch die Gruppenkommunikation ausgefüllt
- ▶ Flags werden nicht verwendet
- ▶ Payload die eigentlichen Nutzdaten
- ▶ Headers normalerweise für Protokollinformationen verwendet
- ▶ Beispiel: Nachricht an alle

```
Message msg = new Message(null, null,  
                           "Hello".getBytes());  
channel.send(msg);
```

Verwendung der Gruppenkommunikation JGroups

`org.jgroups.util.Util`

- ▶ Serialisierung eines Objektes mit `objectToByteBuffer()`
- ▶ Deserialisierung eines Objektes mit `objectFromByteBuffer()`
- ▶ Ausgabe der aktuell aktiven Mitglieder `printMembers()`

Verwendung der Gruppenkommunikation JGroups

Konfiguration der Gruppenkommunikation

► Beispielkonfiguration als XML-Datei oder Zeichenkette

```
<config>
<TCP start_port="7800" loopback="true" send_buf_size="150000"
recv_buf_size="400000" down_thread="false" up_thread="false"/>

<TCPPING timeout="3000" initial_hosts="XXXX[7800]" port_range="1"
num_initial_members="1" down_thread="false" up_thread="false"/>
...
<pbcast.NAKACK max_xmit_size="8192" use_mcast_xmit="true"
retransmit_timeout="200,400,800" down_thread="false" up_thread="false"/>
...
<TOTAL down_thread="false" up_thread="false"/>
<QUEUE down_thread="false" up_thread="false"/>
<STATE_TRANSFER down_thread="false" up_thread="false"/>
</config>
```

oder

```
TCP(start_port=8015):
TCPPING(initial_hosts=131.188.34.63[7800];port_range=2;num_initial_members=1):
FD(timeout=10000;max_tries=2;shun=true):FD_SOCK:VERIFY_SUSPECT(timeout=1500):
pbcast.NAKACK(gc_lag=100;retransmit_timeout=3000):
pbcast.GMS(join_timeout=5000;shun=true;print_local_addr=true):
pbcast.STATE_TRANSFER
```

► Achtung: Im Rahmen der Aufgabe muss `initial_hosts` angepasst werden durch die aktiven Mitglieder der Gruppe

Ergänzungen zu FORMI

Information und Sourcen

- ▶ <http://www-vs.informatik.uni-ulm.de/proj/aspectix/formi>
- ▶ `/local/formi`

Verwendung

- ▶ Bei der Verwendung am eigenen Rechner: **ant.properties** anpassen
 - ▶ `java_tools = /local/java-1.5/lib/tools.jar`
- ▶ Es gibt einen eigenen RMI-Compiler:
 - ▶ `java -jar /local/formi/lib/compiler.jar -classpath ../classes/ -keep example.audiosample.Radio`
- ▶ Zur einfachen Verwaltung von mehreren Shells hilft `screen`

Ergänzungen zu FORMI

Erzeugen einer remote Reference bzw. eines initial Fragments

- ▶ FragImpl – Fragmentimplementierung
- ▶ Factory zum erzeugen eines neuen Fragments
- ▶ comm_par – Kommunikationsparater
- ▶ addFactoryArgs – optionale Parameter für die Factory
- ▶ addFragArgs – optionale Parameter für die Fragmentimplementierung
- ▶ Beispiel:

```
Fragment frag = (Fragment)  
    FragmentedObjectFactory.createObject  
        (<FragImpl>.class, DefaultFragImplFactory.class,  
         comm_par, addFactoryArgs, addFragArgs);  
bzw.  
Fragment frag = (Fragment)  
    FragmentedObjectFactory.createObject  
(Messenger.class, DefaultFragImplFactory.class,  
 (Object[]) new SIpAddress[] {new SIpAddress(netAddress.getLocalHost().  
    getAddress(),7800)}, null, null);
```

