Adding Protocols to JGroups

Matúš Harvan

Stack of Protocols

- Protocols organized in a stack
- Protocols pass events *up* and *down* the stack
- Events:
  - Messages
  - View changes
  - ...
- Stack described in a config file
  - conf/udp1.xml
  - conf/udp2.xml

Receiving a Message

- "Up the stack"
  - Protocol at *bottom* of the stack (top of config file) gets the message first
  - Processes the message
  - Passes it *up* the stack to the next protocol
- Received message is processed in

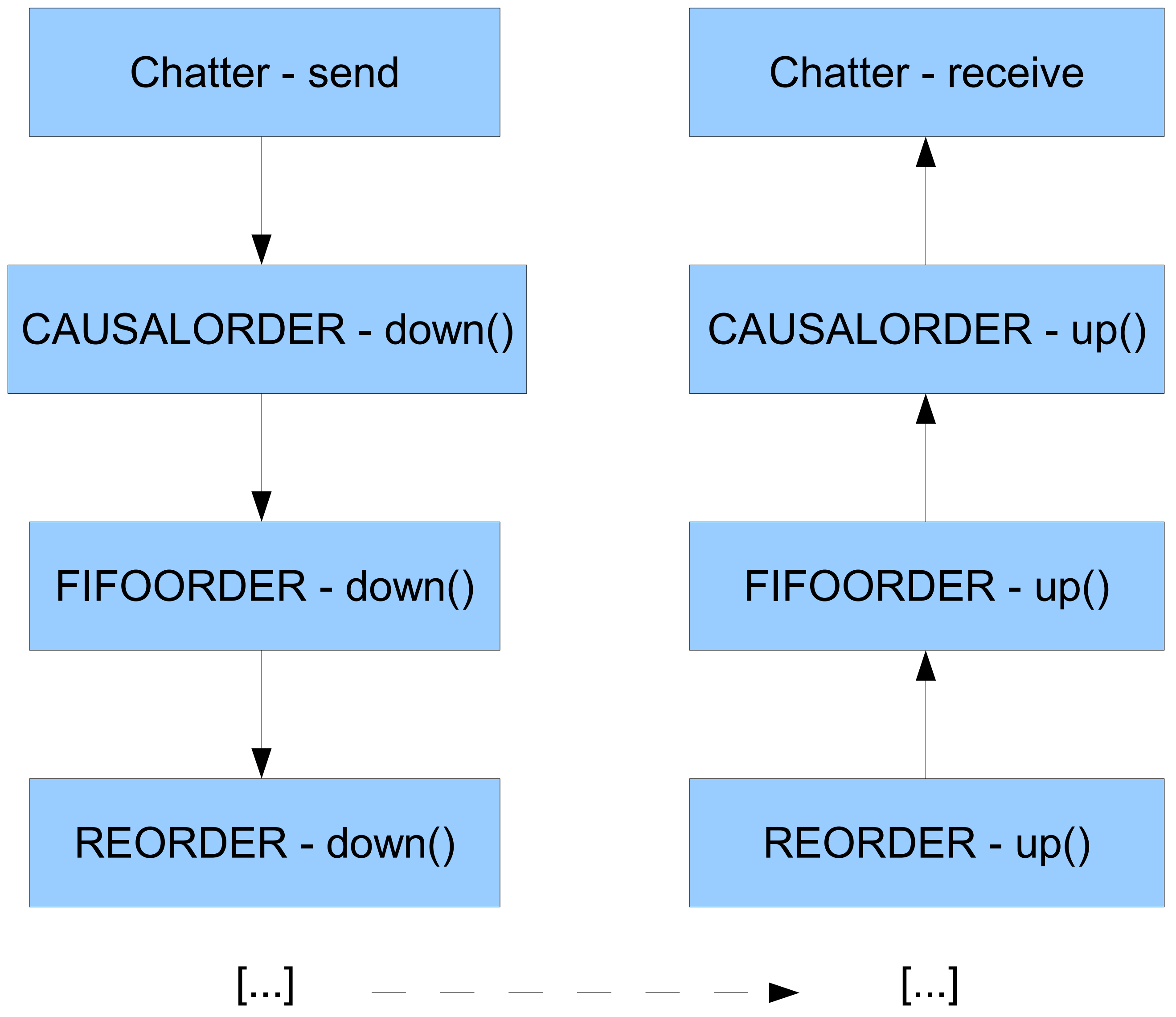
```
public Object up(Event evt)
```

Sending a Message

- “Down the stack”
  - Protocol on *top* of the stack (bottom of config file) gets the message first
  - Processes the message
  - Passes it *down* the stack to the next protocol
- Message to be sent is processed in

```
public Object down(Event evt)
```

Protocol Stack Example



Config file:

```
<config>
    [...]
    <REORDER/>
    <FIFOORDER/>
    <CAUSALORDER/>
</config>
```

Writing a Protocol

- Extend Protocol class
- Provide / Implement:
  - Unique protocol name
  - up, down functions

Up and Down - Details

Receiving Messages

```
public Object up(Event evt) {

    switch (evt.getType()) {
      case Event.MSG:
          // process the message
          // return an event or null
    }

   // pass on to the layer below us
   return up_prot.up(evt);
  }
```

Sending Messages

```
public Object down(Event evt) {

    switch (evt.getType()) {
      case Event.MSG:
          // process the message
          // return an event or null
    }

   // pass on to the layer below us
   return down_prot.down(evt);
  }
```

Processing Messages

- Getting the message out of an event

```
Message msg = (Message) evt.getArg();
```

- Testing for a broadcast / multicast message

```
Address dest = msg.getDest();
if (dest == null || dest.isMulticastAddress()) {
        ...
}
```

Processing Messages

- Messages may be stored and passed further up/down later
  - Timers
  - When a message comes, pass additional messages up/down the stack (in additional events)
- Creating an event for passing a message up/down the stack:

```
up_prot.up(new Event(Event.MSG, msg));
down_prot.down(new Event(Event.MSG, msg));
```

- Returning null in up()/down() is OK

Protocol Initialization and Properties

- Initialization

  public void init()

- Properties
  - Set in config file
  - Processing properties set in config file:

    public boolean setProperties(Properties props)

    - Example in DELAY.java

Adding a Protocol to JGroups

- put the protocol implementation into

```
./src/org/jgroups/protocols/
```

- rebuild the framework with

```
./build.sh jar
```

- modify the stack configuration to use the protocol

Protocol Example

- MYPROTOCOL.java

Headers

- Messages can contain headers
- Carry extra information used by protocols

Writing a Header

- Extend the Protocol class
- Provide / implement:
  - Unique header name
  - Externalizable interface (empty functions OK)
  - Streamable interface
  - Size for marshalling
- Register protocol
  - add to conf/jg-magic-map.xml

  or

  - ClassConfigurator.getInstance().add((short)1900, MyHeader.class);

Using a Header

- Adding a header to a message

```
msg.putHeader(MyHeader.name, new MyHeader());
```

- Accessing a header in a message

```
MyHeader hdr = (MyHeader)msg.getHeader(MyHeader.name);
 if (hdr != null) {
 ...
```

- Removing a header
  - removeHeader() deprecated
  - Problem with message retransmission
  - Trick with putHeader()
  - Only do on a copy of the message

Header Example

- MyHeader.java
- Put into the protocol class

```
public class MYPROTOCOL extends Protocol {
    [...]
    public static class MyHeader extends Header implements Streamable {
        [...]
    }
}
```

JGroups Documentation

- JGroups manual
  - http://www.jgroups.org/manual/html
  - Headers example in section 6.2
- JGroups javadoc
  - http://www.jgroups.org/javadoc/

or

  - ./build.sh javadoc -> dist/javadoc
- Source code of other protocols in src/org/jgroups/protocols/
- JGroups source code

Testing Your Solution

- REORDER protocol
  - Randomly delays and reorders messages
  - Insert just before FIFO and CAUSALORDER

Submitting Your Solution

- Send the relevant files via email to <mharvan@inf.ethz.ch>
  - Source code (protocols, headers)
  - Config files (if you changed something)
  - Short explanation how your code works (plain text or pdf)
  - Do NOT send the whole JGroups source tree!!

- These slides are available under http://www.inf.ethz.ch/personal/mharvan/teaching/sftds10