

The Mondocorp Inc. Chat Project

Phase II

II:1

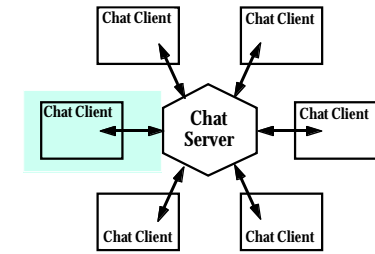
Project Manager's System Integration Meeting

1. System architecture overview
2. Class and interface summaries
3. API documentation
4. Implementation notes

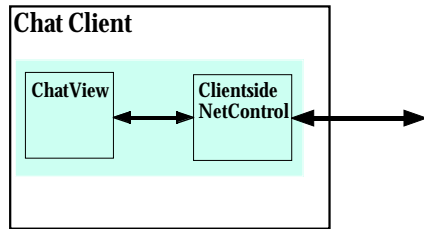
Note: next meeting is Friday November 12, 1999

II:2

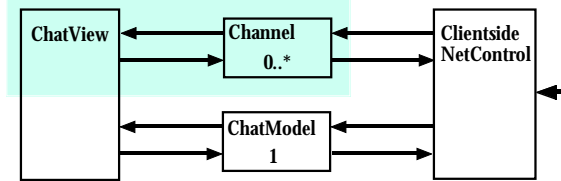
System architecture



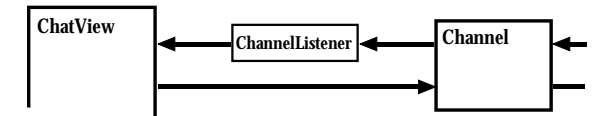
II:3



II:4

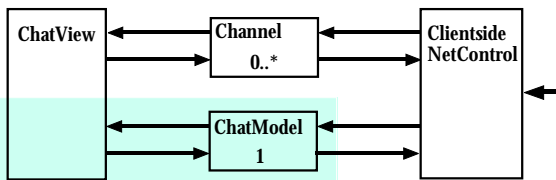


II:5



II:6

Outgoing messages: call Channel methods
Incoming messages: listen for Channel events



II:7



II:8

Outgoing messages: call ChatModel methods
Incoming messages: listen for ChatModel events

Class and interface summaries

- Channel class
- ChannelListener interface
- ChatModel class
- ChatListener interface

...plus Events and other miscellaneous classes/interfaces

II:9

Channel objects

are like GUI components, e.g., JButton, etc.
(except they aren't contained in anything)

use method calls to operate on them

e.g. `passOutMessage(String msg, String sendr)`

including registering listeners

`addChannelListener(ChannelListener cl)`

`removeChannelListener(ChannelListener cl)`

II:10

Channel objects

create one for each channel joined
(discard it when leaving the channel)

you send messages out to a particular channel
using its `passOutMessage` method

they notify all of their registered listeners
of each incoming message

II:11

ChannelListener objects

instances of classes that you write
that implement the ChannelListener interface

their `inMessageReceived` method is called
when the channel gets a message from the net
(use the `getContent` method of the ChannelEvent
object to retrieve the message; likewise `getSender`)

also have a `userListChanged` method

II:12

The ChatModel object

just one, interacts with the server about things
having to do with the network overall:

`login(String user)`

`openChannel(String user, Channel channel)`

`requestChannelListUpdate(String user)`

`addChatListener(ChatListener cl)`

plus `logout`, `closeChannel`, `removeChatListener`
and `init()` to set up model initially

II:13

ChatListener objects

instances of classes that you write
that implement the ChatListener interface

their `channelListProvided` method is called
when the client gets a message from the server
about all of the channels currently available

also have a `serverResponded` method

II:14

ChatListener objects

the `serverResponded` method is called
when the client gets a response from the server

e.g., `NICK_IN_USE`
`NO_SUCH_CHANNEL` these are defined
`CONNECTI ON_LOST` constants in the
`BAD_SENDER` ChatMessage class
`LOGIN_CONFIRMED`

use the `getType` method of the ChatEvent object
to determine what the response was

II:15

API documentation

is available at

http://www.cs.uiowa.edu/~oden/courses/API_docs/index.html

...or link to it from the project web page

II:16

Implementation notes

Your main method will now be as follows:

```
public static void main(String[] args){
    ChatView view = new ChatView();
    ChatModel model = new ChatModel();
    view.assignModel(model);
    model.init();
}
```

where `assignModel` is a method you must add to your
ChatView class to save a reference to the ChatModel

II:17

Implementation notes

You need to keep track of your Channel objects and of
which GUI components each connects to both
coming and going (think composition)

There are many more methods of these classes that
are intended for use by the network controller
module, but some might be of use to you

More notes are likely to be added later

II:18