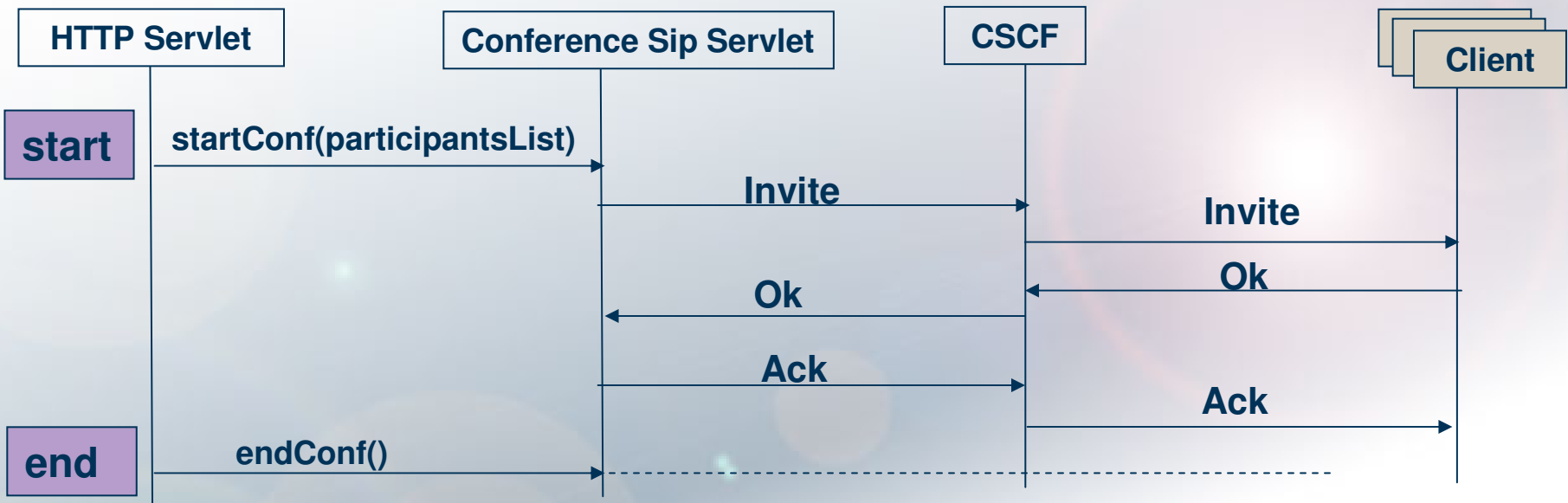
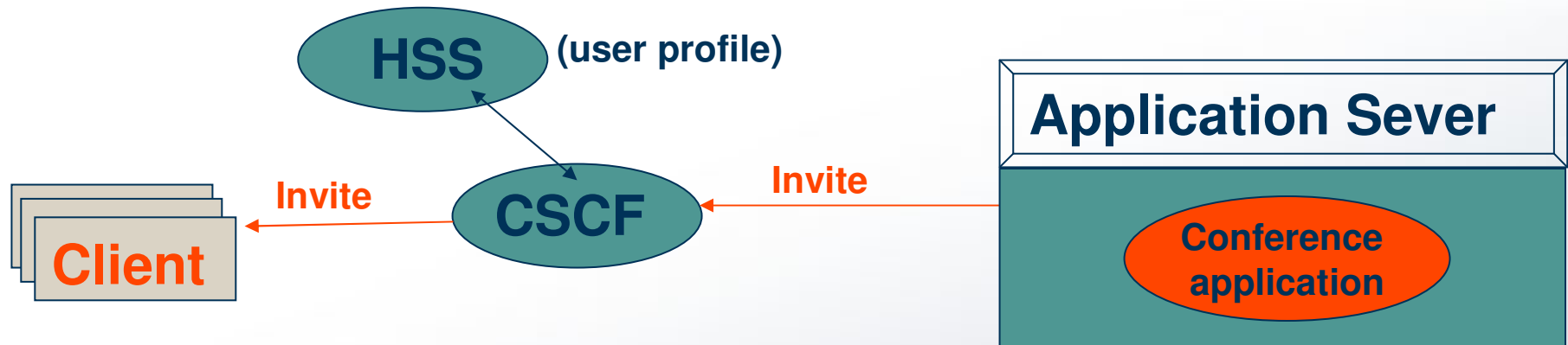


# Hints for the project

# What to implement?



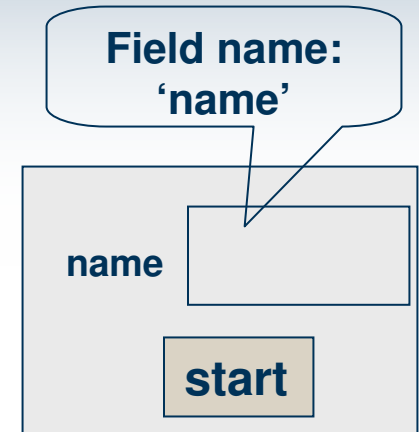
# Communication between HTTP Servlet & SIP Servlet

- **web.xml**

```
<listener>
  <listener-class> ConferenceSipServlet</listener-class>
</listener>
```

- **HTTP Servlet:**

```
private void processRequest (HttpServletRequest request,
                             HttpServletResponse response) throws ServletException, IOException {
    .....
    // if start button was clicked {
    String name = request.getParameter("name");
    context.setAttribute("Name", name); }
    .....
}
```



# Communication between HTTP Servlet & SIP Servlet

- **Conference SIP Servlet:**

public class ConferenceSipServlet extends SipServlet implements

ServletContextAttributeListener {

static ServletContext context;

.....

public void attributeAdded (ServletContextAttributeEvent attrEv) {  
    handleTrigger(attrEv);  
}

public void attributeReplaced (ServletContextAttributeEvent attrEv) {  
    handleTrigger(attrEv);  
}

handleTrigger(attrEv){  
    if (attrEv.getName().equalsIgnoreCase("Name")) {  
        String name = (String) attrEv.getValue();  
        .....  
    }

}

# Media Handling

The screenshot displays the Eclipse IDE interface for a Java EE project. The main editor window shows the code for `MyFirstServlet.java`, which implements two methods: `doAck` and `doInvite`. The `doInvite` method is currently selected and highlighted, showing its implementation which creates a `MyMediaHandler` instance and calls its `playFile` method to play a test audio file.

```
protected void doAck(SipServletRequest sipServletRequest)
    throws ServletException, IOException {
    //TODO: Implement this method
    System.out.println("\nACK MESSAGE RECEIVED::");
}

/**
 * @inheritDoc
 */
protected void doInvite(SipServletRequest sipServletRequest)
    throws ServletException, IOException {
    //TODO: Implement this method
    sipServletRequest.createResponse(200).send();
    System.out.println("\nINVITE MESSAGE RECEIVED:: Congratulation");
    System.out.println("\nCreating Media Handler.....");
    handler = new MyMediaHandler();
    System.out.println("\nCalling the player.....");
    handler.playFile("C:/fatna/music/test.mpg");
}
```

The Project Explorer on the left shows the project structure, including the `com.ericsson` package and the `media` sub-package containing `MyMediaHandler.java` and `test.mp3`. The Console window at the bottom displays the output of the application, showing DNS console log messages.

```
DNS Console Log [Java Application] DNS Console Log
16-Mar-2009 16:21:10.068 cache cleaning timer fired, cleaner state = 0
16-Mar-2009 16:21:10.765 cache cleaning timer fired, cleaner state = 0
16-Mar-2009 16:21:10.804 dns_zone_dialup: zone e164.arpa/IN: notify = 0, refresh = 0
16-Mar-2009 16:21:10.805 dns_zone_dialup: zone ericsson.com/IN: notify = 0, refresh = 0
16-Mar-2009 16:21:10.805 dns_zone_dialup: zone authors.bind/CH: notify = 0, refresh = 0
16-Mar-2009 16:21:10.805 dns_zone_dialup: zone hostname.bind/CH: notify = 0, refresh = 0
16-Mar-2009 16:21:10.805 dns_zone_dialup: zone version.bind/CH: notify = 0, refresh = 0
16-Mar-2009 16:21:10.805 dns_zone_dialup: zone id.server/CH: notify = 0, refresh = 0
```