STARS on Android

Takashi Kosuge

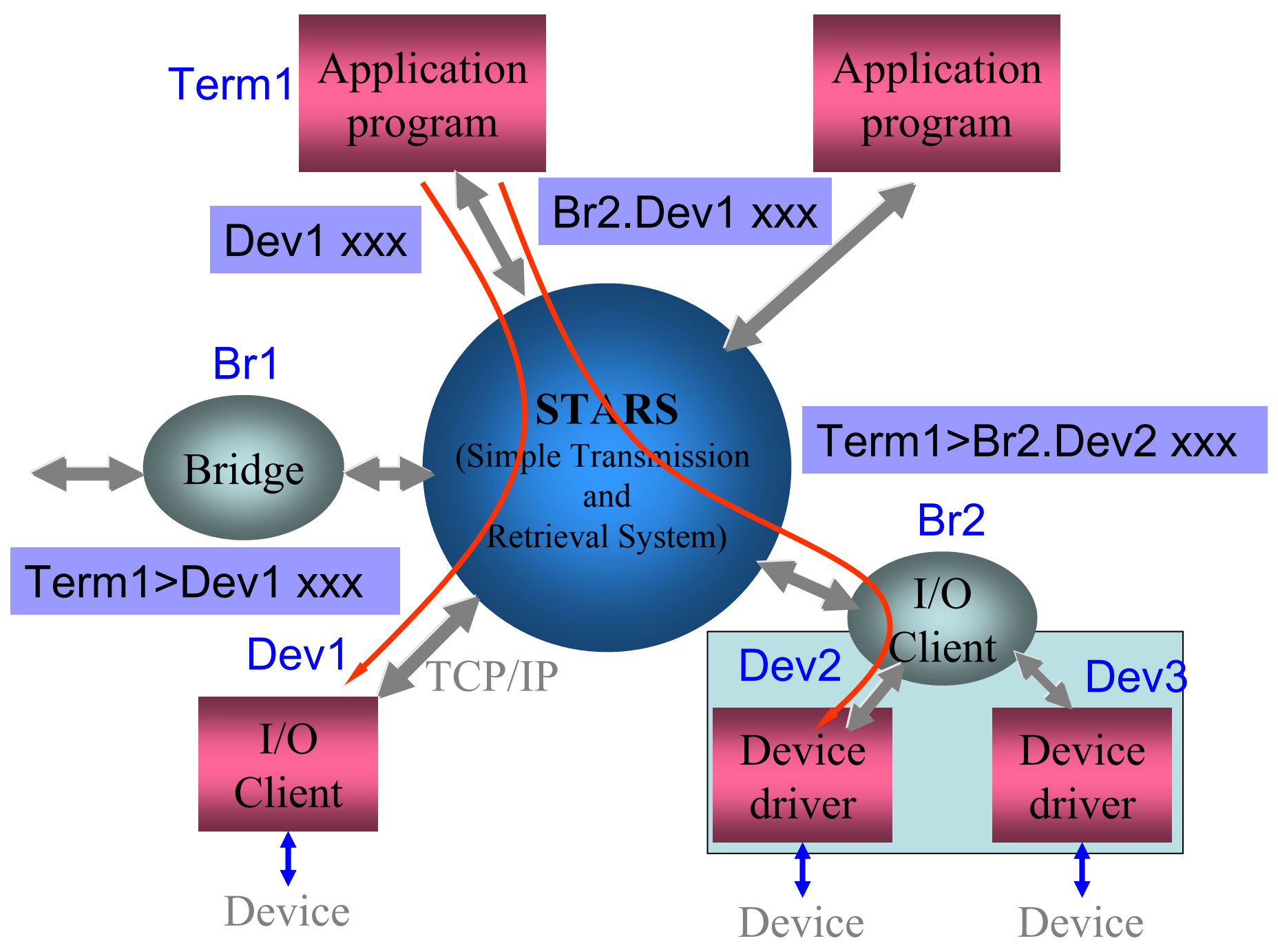
Photon Factory, KEK 1-1 Oho Tsukuba-shi Ibaraki-ken 305-0801 JAPAN

Abstract

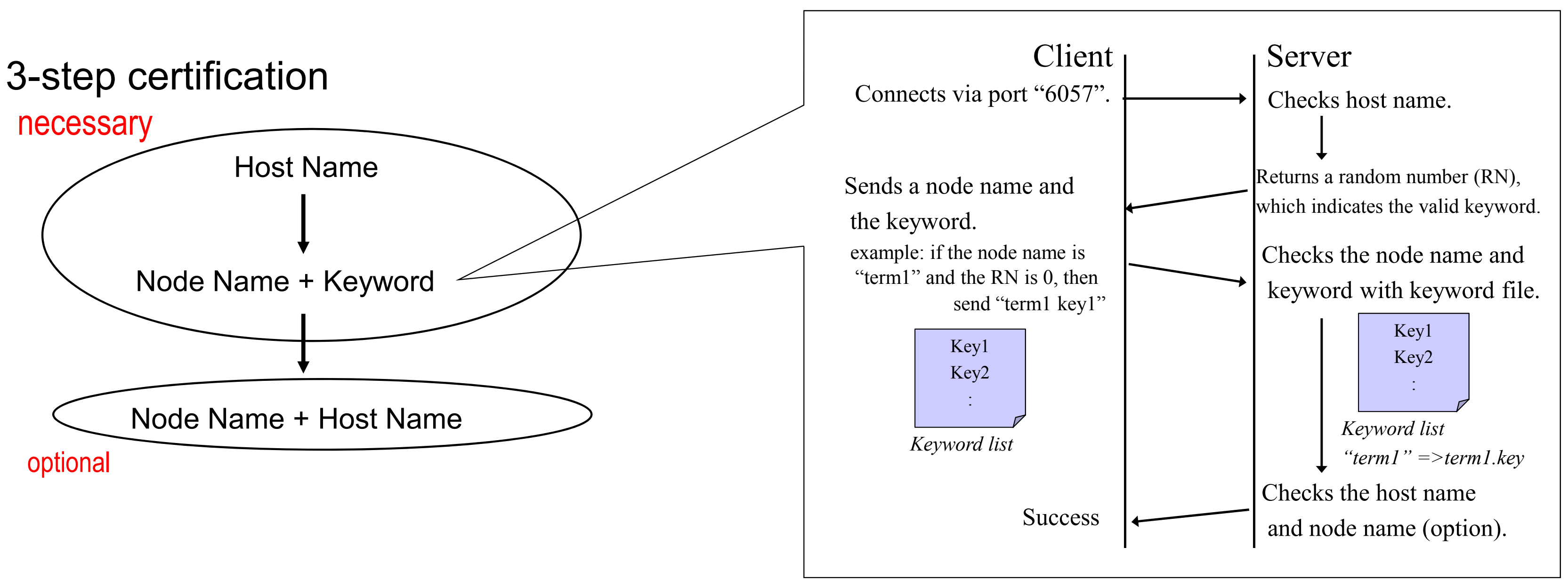
STARS (Simple Transmission and Retrieval System) [1][2] is a message transferring software for small-scale control systems with a TCP/IP socket, and it works on various types of operating systems. STARS is used as a beamline control system for controlling the optical devices (mirrors, monochrometers, etc.) for beamlines at the Photon Factory.

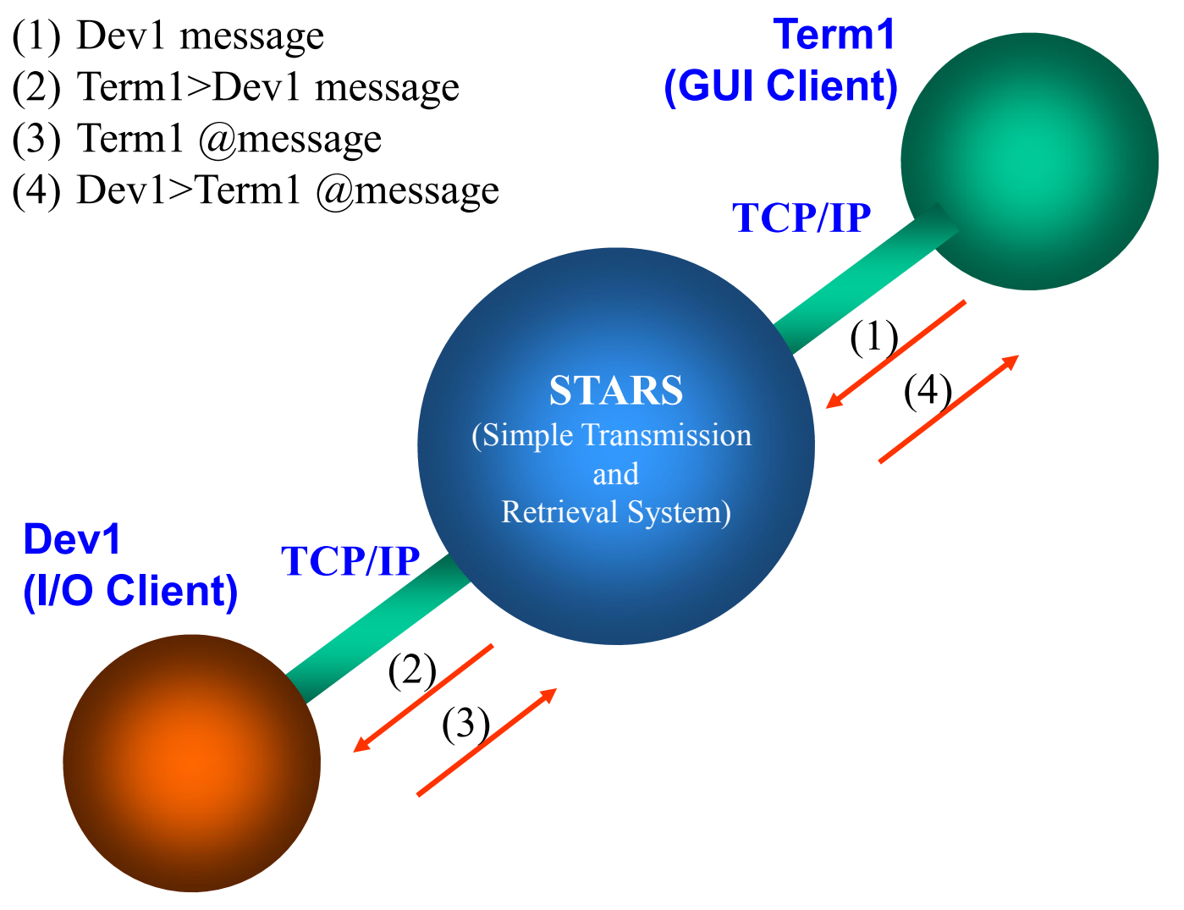
Overview of STARS

- Simple
- TCP/IP socket
- Server and client architecture (Each client has its own unique node name)
- Text-based message transfer
- Server is written in Perl => Runs on various OS



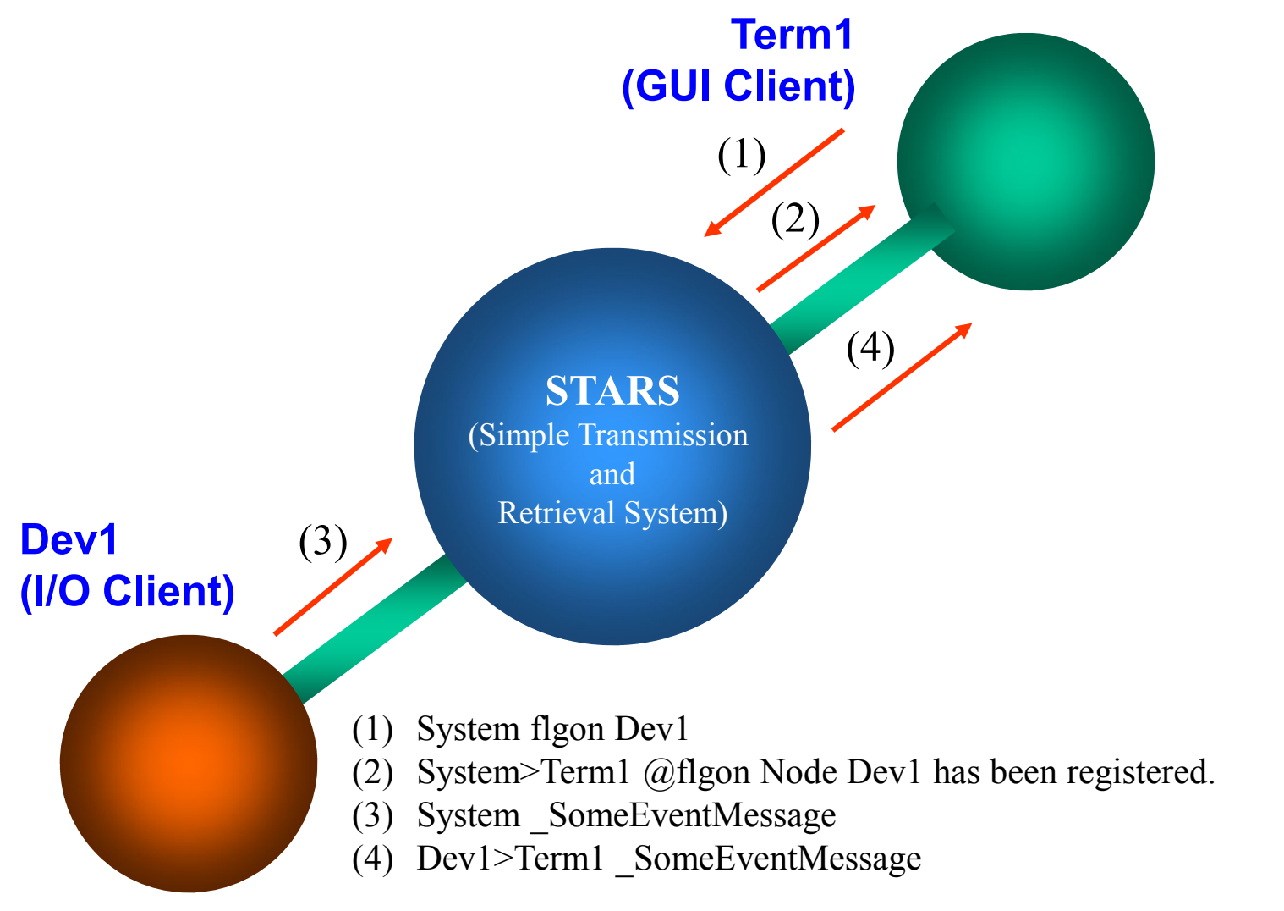
STARS has simple certification procedure at the connection time.


3-step certification
necessary
Host Name
Node Name + Keyword
Node Name + Host Name
optional
Client
Server
Connects via port "6057".
Checks host name.
Sends a node name and the keyword.
example: if the node name is "term1" and the RN is 0, then send "term1 key1"
Returns a random number (RN), which indicates the valid keyword.
Checks the node name and keyword with keyword file.
Key1
Key2
:
Keyword list
Key1
Key2
:
Keyword list
"term1" =>term1.key
Success
Checks the host name and node name (option).



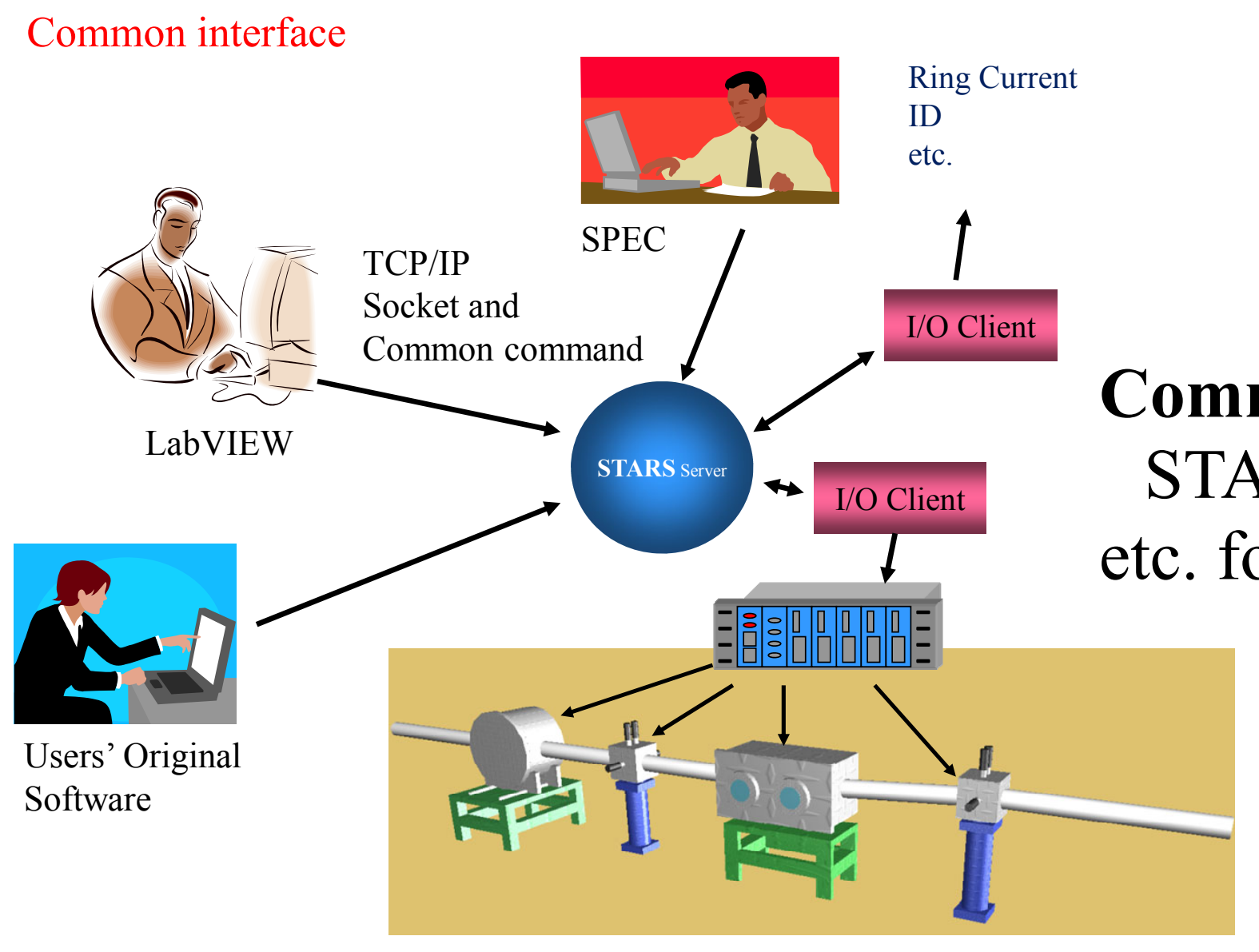
Command and Reply

(1)Term1 sends a command to the server.
(2)The server sends the command to Dev1.
(3)Dev1 sends a reply (starts with "@").
(4)The server sends the reply to Term1.


Term1 (GUI Client)
(1)
(2)
(4)
STARS (Simple Transmission and Retrieval System)
Dev1 (I/O Client)
(3)
(1) System flgon Dev1
(2) System>Term1 @flgon Node Dev1 has been registered.
(3) System _SomeEventMessage
(4) Dev1>Term1 _SomeEventMessage

Event Delivery Function

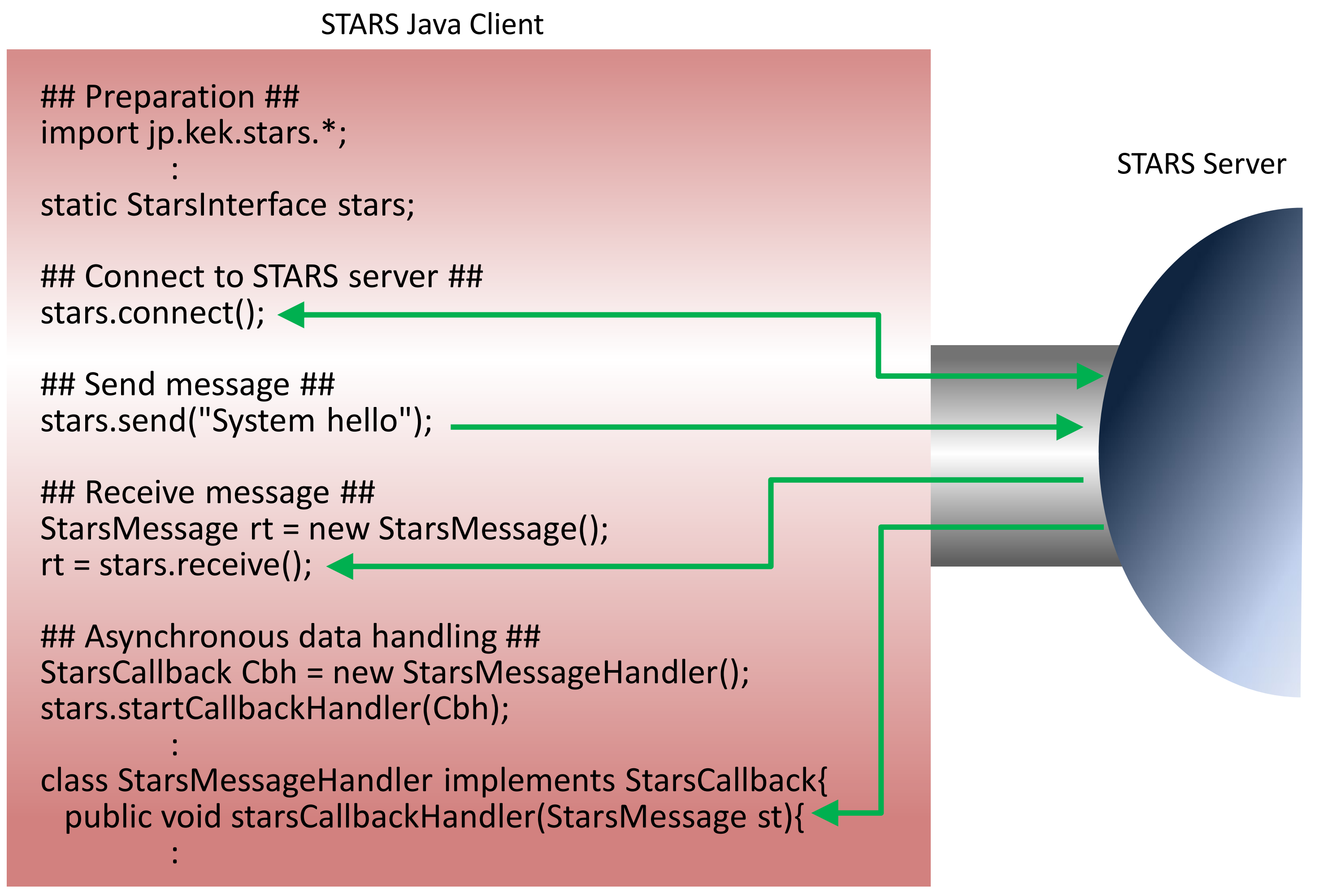
(1)Term1 sends an event delivery request.
(2)The server returns a reply of the request.
(3)Dev1 sends an event to the server (starts with "_").
(4)The event is delivered the reply to Term1.



Common Interface of Beamline

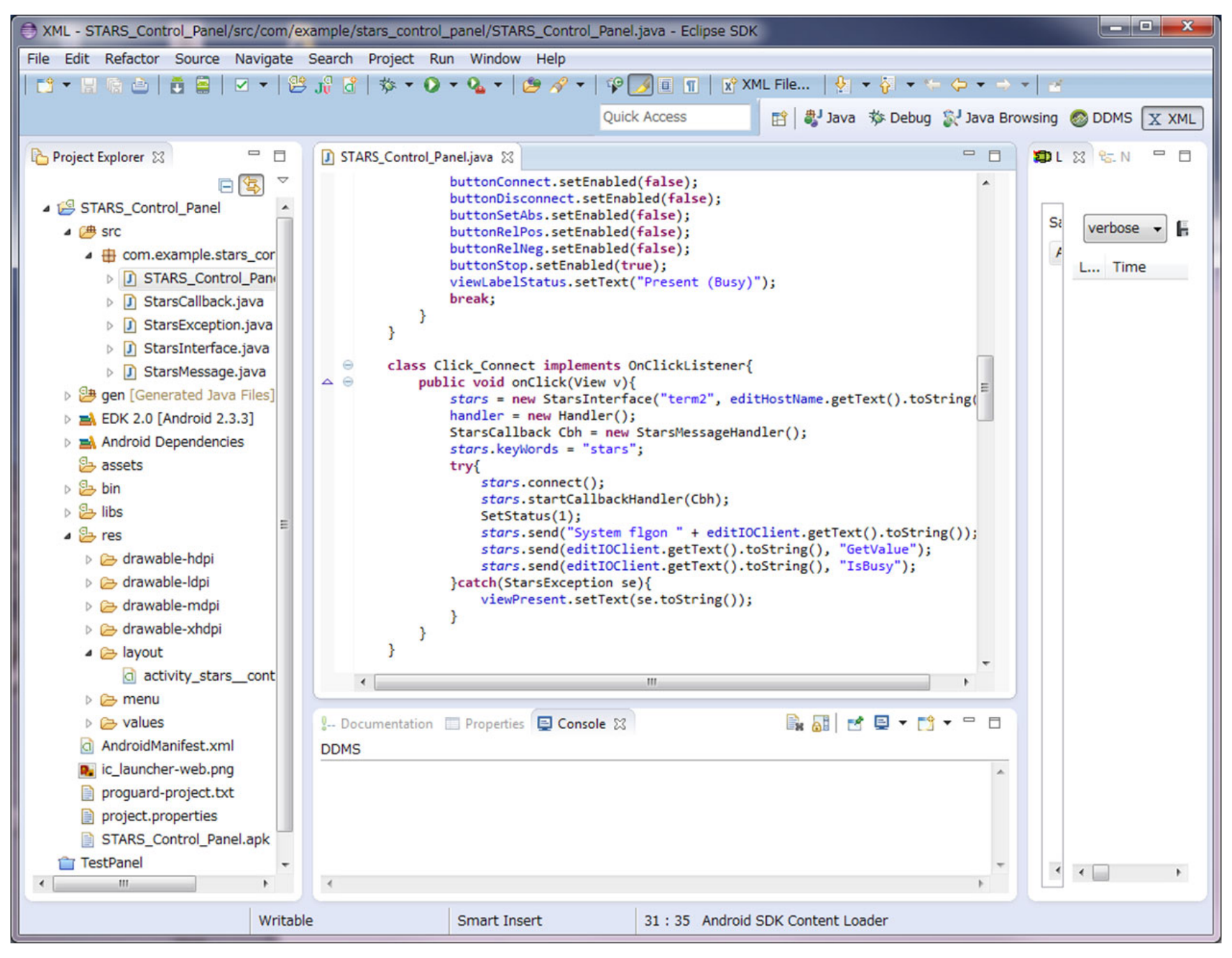
STARS provides common interface to GUI programs, etc. for handling beamline components.

STARS Java Interface

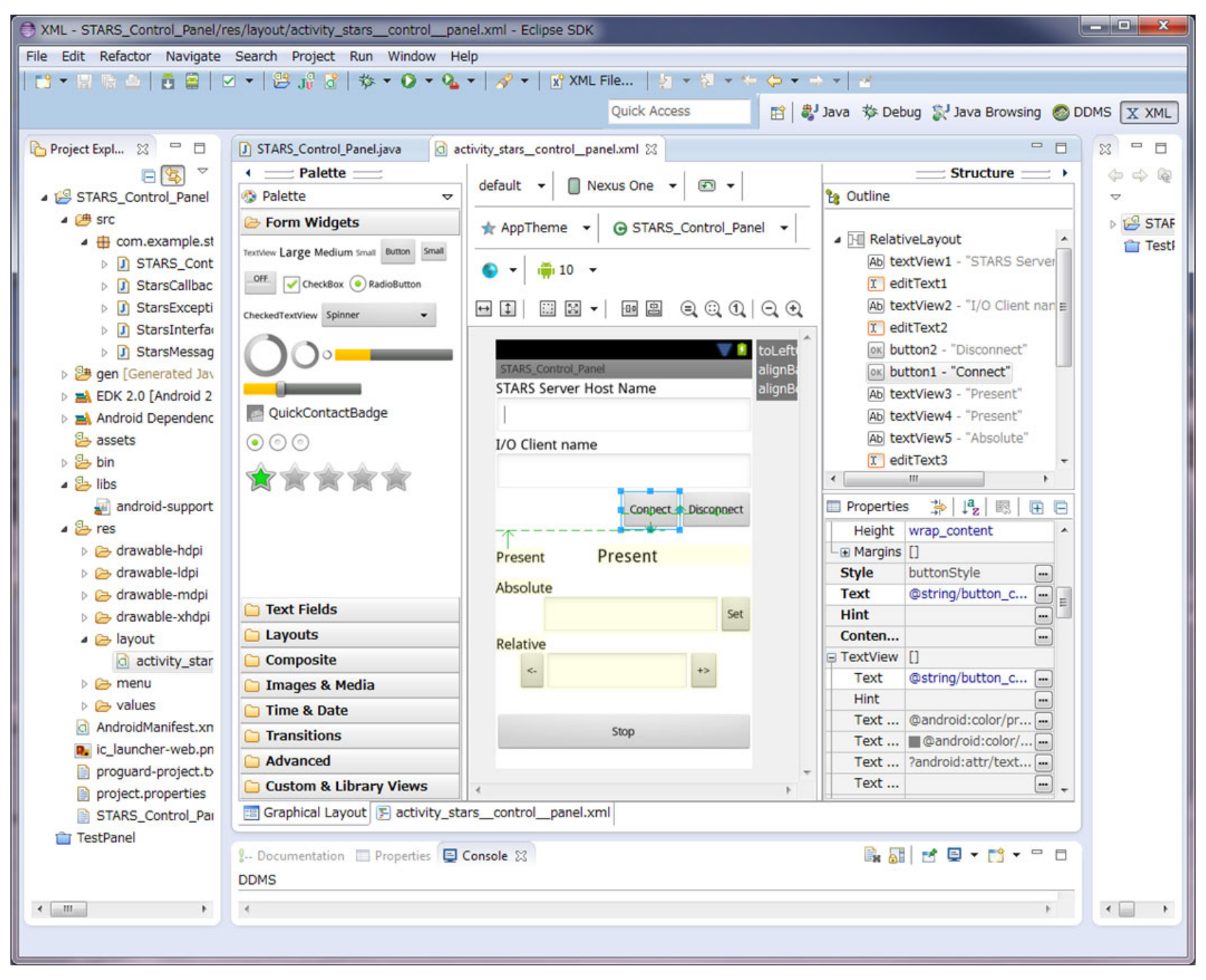


```
## Preparation ##
import jp.kek.stars.*;
          :
static StarsInterface stars;

## Connect to STARS server ##
stars.connect();

## Send message ##
stars.send("System hello");

## Receive message ##
StarsMessage rt = new StarsMessage();
rt = stars.receive();

## Asynchronous data handling ##
StarsCallback Cbh = new StarsMessageHandler();
stars.startCallbackHandler(Cbh);
          :
class StarsMessageHandler implements StarsCallback{
   public void starsCallbackHandler(StarsMessage st){
          :
```

Development of Clients Using STARS Java Interface for Android



```
                buttonConnect.setEnabled(false);
                buttonDisconnect.setEnabled(false);
                buttonSetAbs.setEnabled(false);
                buttonRelPos.setEnabled(false);
                buttonRelNeg.setEnabled(false);
                buttonStop.setEnabled(true);
                viewLabelStatus.setText("Present (Busy)");
                break;
            }
        }

    class Click_Connect implements OnClickListener{
        public void onClick(View v){
            stars = new StarsInterface("term2", editHostName.getText().toString(
            handler = new Handler();
            StarsCallback Cbh = new StarsMessageHandler();
            stars.keyWords = "stars";
            try{
                stars.connect();
                stars.startCallbackHandler(Cbh);
                SetStatus(1);
                stars.send("System flgon " + editIOClient.getText().toString());
                stars.send(editIOClient.getText().toString(), "GetValue");
                stars.send(editIOClient.getText().toString(), "IsBusy");
            }catch(StarsException se){
                viewPresent.setText(se.toString());
            }
        }
    }
```



Methods

Preparation
static StarsInterface stars
= new (myNodeName, starsServerName, keyFileName, starsPort);

Connect
//Send "GetValue" command to a node name "Dev1".
stars.send("Dev1 GetValue");
//or
stars.send("Dev1", "GetValue");

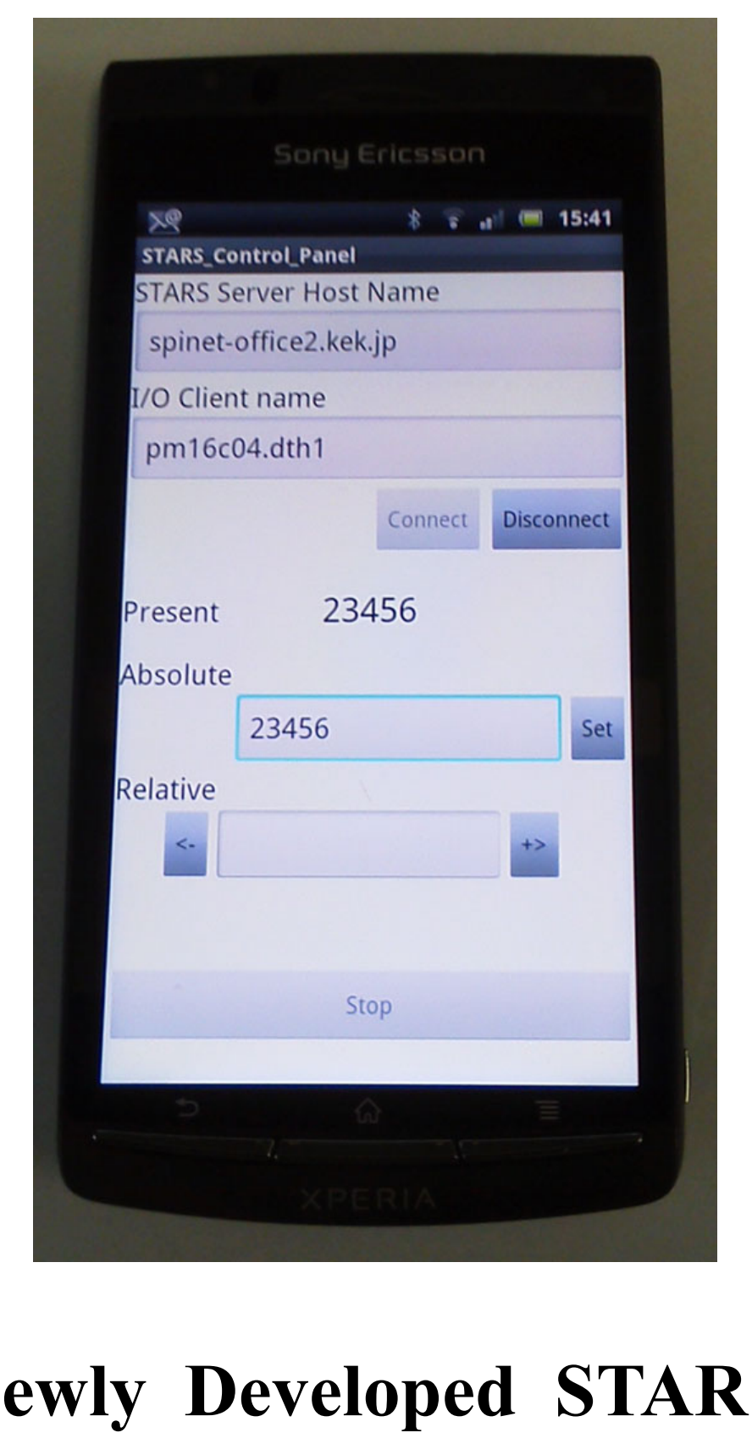
Receive
StarsMessage rcvMsg = stars.receive(timeOut);

Callback Function

```
public void onCreate(Bundle savedInstanceState){
  super.onCreate(savedInstanceState);

  setContentView
(R.layout.activity_stars__control__panel);
  //Set callback "StarsMessageHandler()" will be called
  // when a message arrives from the STARS Server.
  handler = new Handler();
  StarsCallback Cbh = new StarsMessageHandler();
  try{
     stars.starsCallbackHandler(Cbh);
  }catch(StarsException se){}
}
```

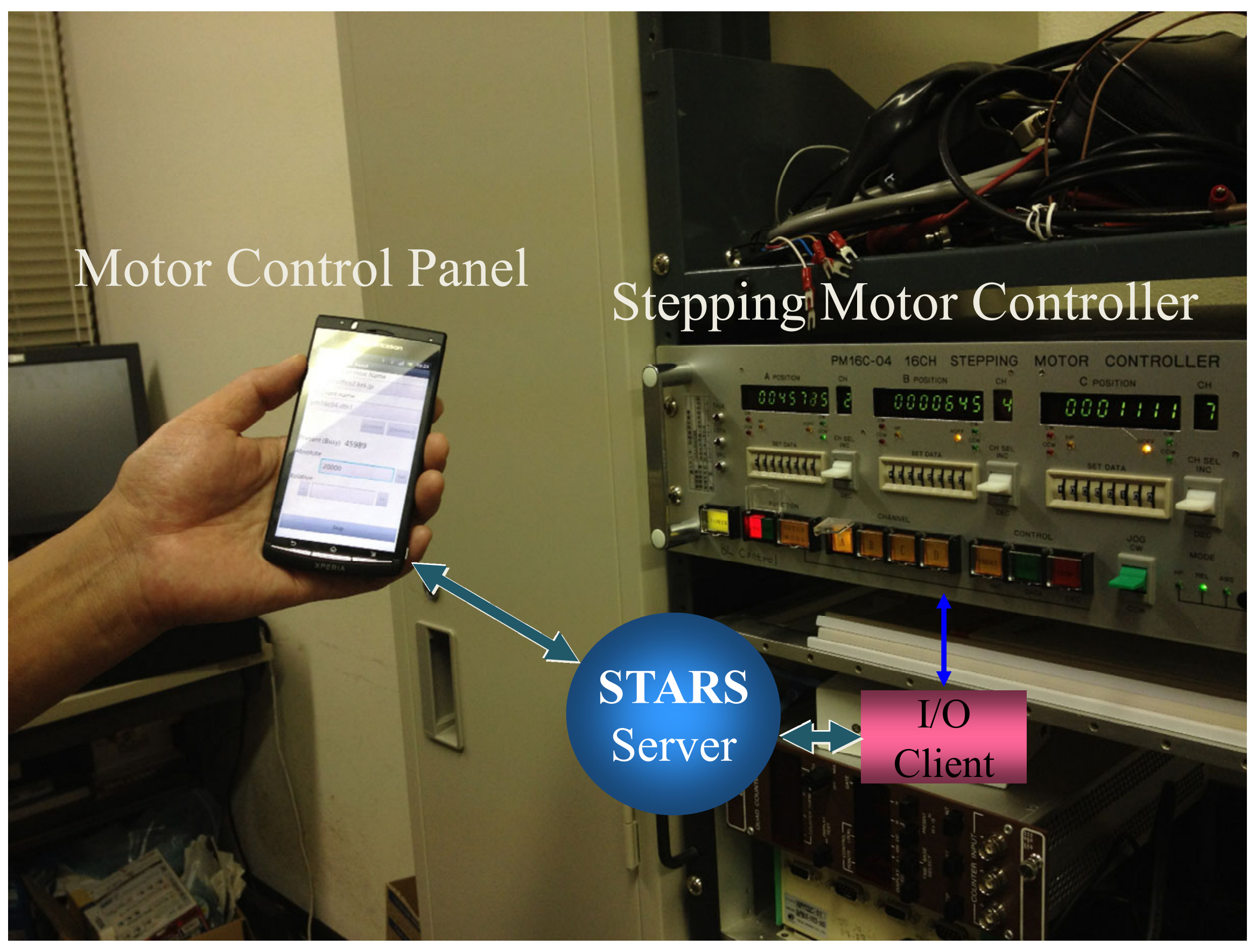
```
//Handle messages.
class StarsMessageHandler implements StarsCallback
{
 public void starsCallbackHandler(StarsMessage st){
  if(st.command.equals(@GetValue)"{
   //Write message handling codes.
    handler.post(new Runnable(){
     public void run(){
      viewPresent.setText(curVal.toString());
     };
    });
   }
}
```

Motor Control with Android GUI



STARS_Control_Panel
STARS Server Host Name
spinet-office2.kek.jp
I/O Client name
pm16c04.dth1
Connect Disconnect
Present 23456
Absolute
23456
Set
Relative
<- +>
Stop

Newly Developed STARS Motor Control Panel with Android



Motor Control Panel and Stepping Motor Controller
Stepping motors are controlled by Motor Control Panel through STARS.

Conclusion

- We ported the STARS Java interface to Android, which was demonstrated to work satisfactorily. We then developed a Motor Control Panel for smart phones. The development of a GUI for an Android tablet is also possible using a similar method.
- The STARS Java interface for Android and the GUI for smart phones or tablets will be useful tools on STARS-based control systems.

References

[1] T. Kosuge, Y. Saito, "RECENT PROGRESS OF STARS", Proceedings of PCaPAC2005, Hayama, Japan, 2005.
[2] http://stars.kek.jp/

http://stars.kek.jp/