STARS ON ANDROID

T. Kosuge, Photon Factory, KEK, Tsukuba, 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.

We have succeeded in running a STARS GUI client on Android using the STARS Java interface library. This achievement has brought with it the capability of developing a user-friendly GUI terminal using smartphones or tablet devices. Such a GUI terminal will help beamline users check movement near optical devices.

OVERVIEW OF STARS

STARS consists of a server program (STARS server) and client programs (STARS clients). Each client is connected to the server through a TCP/IP socket, and communicates using text-based messages (Fig. 1).

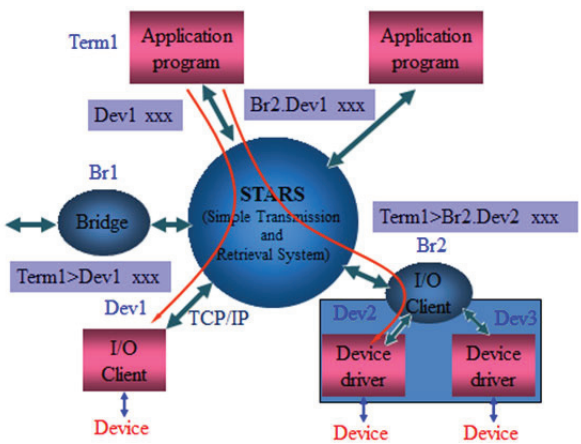

Figure 1: Message transfer between STARS clients and a STARS server.

Each client program has its own unique node name, and it sends text-based messages using the destination node name to the server, which then delivers the messages to the destination client. Through this extremely simple solution, STARS is able to provide basic control system functionality.

The STARS server program was written in Perl, and it can therefore run on various operating systems.

BEAMLINE CONTROL SYSTEM USING STARS

STARS has been installed in more than 20 beamlines of the Photon Factory as a beamline control system.

Before STARS was developed, the beamlines of the Photon Factory used various control systems (e.g., the originally developed software, LabVIEW, or SPEC) and the hardware was controlled using a software directory. At that time, staff members had to prepare their own hardware driver for each control system. Since the installation of STARS, however, hardware drivers are now developed by a "beamline control group," and the beamline control system using STARS provides a common interface to GUI programs, etc., (Fig. 2) for handling the beamline components. This interface can also be accessed by various data acquisition systems.

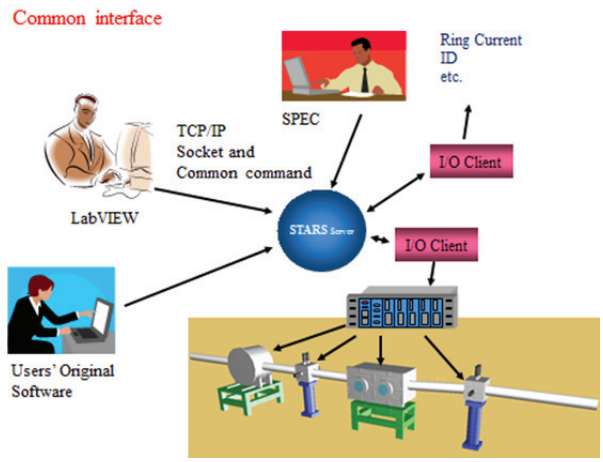

Figure 2: Common interface for handling beamline components using STARS.

Several types of driver software for beamline devices and common GUI programs have recently been developed at the Photon Factory.

STARS JAVA INTERFACE FOR ANDROID

STARS client programmers are required to use TCP/IP sockets and handle text-based messages, and although beginners may find it difficult, it is very easy for programmers with prior knowledge of TCP/IP socket programming to develop a STARS client. STARS is equipped with interface libraries for certain programming languages (Perl, VB, C# [3], C [4], and Java). Programmers need not be concerned with TCP/IP socket programming using these interface libraries. We modified a few parts of the source code for the STARS Java interface for Android.

DEVELOPMENT OF CLIENTS USING STARS JAVA INTERFACE FOR ANDROID

Development Environment

We used Eclipse for the development of the STARS client program for Android. Eclipse requires the

ISBN 978-3-95450-124-3

installation of the Android SDK and the Android Plugin. Figure 3 shows a STARS client for Android developed using Eclipse.

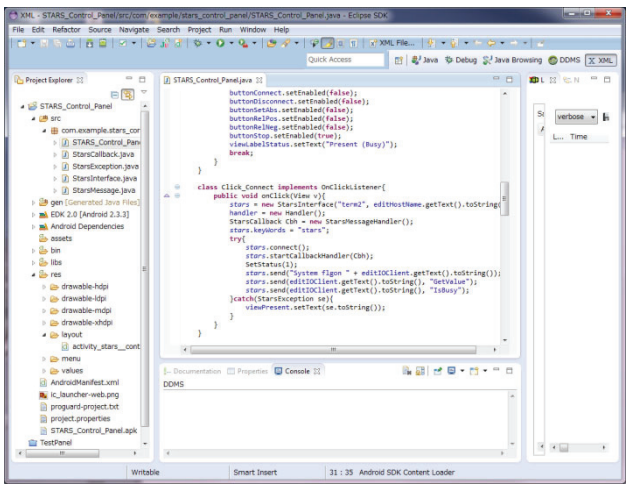

Figure 3: Development of STARS client for Android using Eclipse.

Methods

The STARS Java interface for Android was ported from the original STARS Java interface. The same methods used in the original interface were also used in the ported interface, although UI (User Interface) functions, such as Form Widgets and Text Fields, were handled differently. This difference is described further in the Callback Function section.

Before employing the STARS Java interface methods, the following objects need to be defined:

```
import com.example.stars_control_panel.StarsInterface;
import com.example.stars_control_panel.StarsCallback;
import com.example.stars_control_panel.StarsException;
import com.example.stars_control_panel.StarsMessage;
:
:
static StarsInterface stars
= new (myNodeName, starsServerName, keyFileName,
starsPort);
```

In the above program, "stars" is used as the object name, whereas "myNodeName," "starsServerName," and "keyFileName" are string values, and "starsPort" is an integer value. In addition, "keyFileName" and "starsPort" are omissible, in which case default values are used.

Connect

The "connect" method is used for establishing a connection. This method executes the keyword checking procedure of STARS automatically, and throws an exception if a connection is not established. An example of the "connect" method and error handling is given as follows:

```
try{
   stars.connect();
}catch(StarsException se){
   //Error handling
   viewPresent.setText(se.toString());
}
```

Send

The "send" method is used to send messages to the STARS server in the following manner:

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

The "send" method throws an exception, in which case the "try" and "catch" functions must be used.

Receive

The messages received by the client program can be read from the receive buffer using the "receive" method given below:

```
StarsMessage rcvMsg = stars.receive(timeOut);
```

"timeOut" is an integer value (in m seconds) and is omissible (default: 5000).

"StarsMessage" has the following fields and methods:

- The "getAllMessage()" method is used for receiving all messages.
- The "from" field is the message source node name.
- The "to" field is the destination node name.
- The "command" field retains the command part of the message.
- The "parameters" field retains some of the parameters used in the message.
- The "getMessage()" method returns "command" and "parameters."

Callback Function

When a message arrives from the STARS server, the function set by the programmer is called using "startCallbackHandler," an example of which is shown below.

```
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);
```


ISBN 978-3-95450-124-3

```
  }catch(StarsException se){
    //Write error handling codes here.
  }
}
```

```
//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());
        };
      });
    }
}
```

The "Callback" function of the STARS Java interface is delivered using a "thread." However, the Android UI cannot be handled from another thread. One solution to this issue is creating a "Runnable" object and then calling a "post" method. Thus, this is the difference between the STARS Java interface and the interface for Android.

MOTOR CONTROL PANEL GUI FOR SMART PHONES

We developed a Motor Control Panel for smart phones, which is anticipated to will become a common beamline GUI at the Photon Factory. Figure 4 shows a design view of the Motor Control Panel on Eclipse.

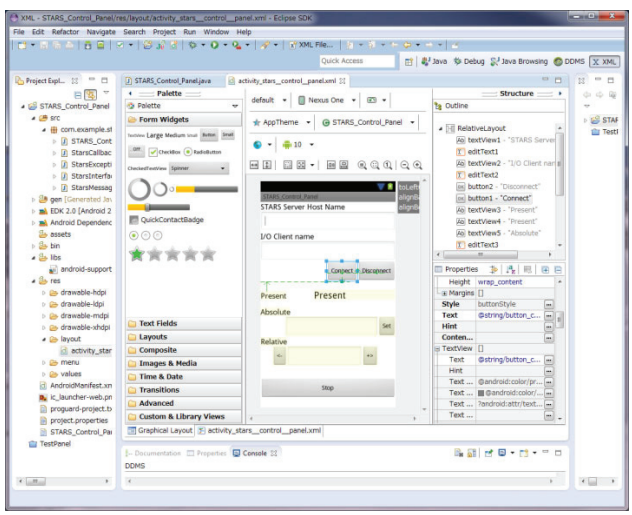

Figure 4: Design view of the Motor Control Panel.

Figure 5 shows a photograph of the Motor Control Panel running on a smart phone (Sony Ericsson Xperia). The Panel connects to the STARS server through a wireless LAN and communicates with clients using a stepping motor controller. Beamline users can remotely control the stepping motor (Fig. 6).

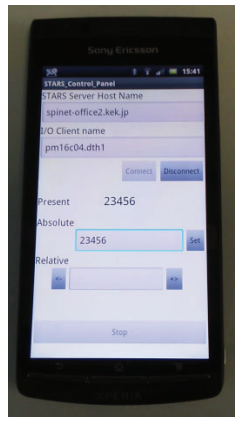

Figure 5: Motor Control Panel on a smart phone.

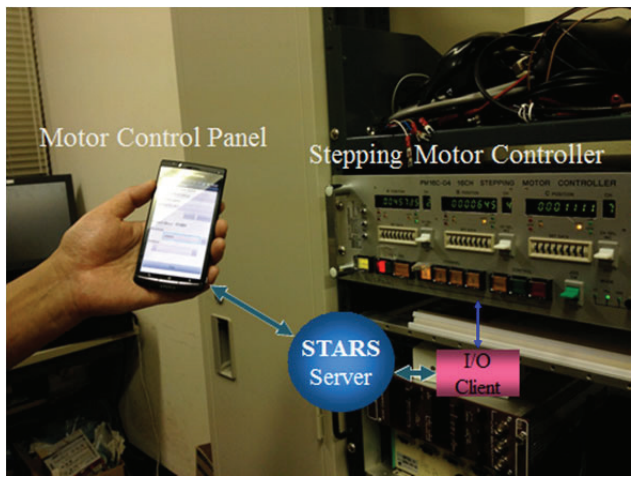

Figure 6: Stepping motor controller operated by the Motor Control Panel through STARS.

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/
[3] T. Kosuge, "STARS .NET INTERFACE FOR WINDOWS CE", Proceedings of PCaPAC2008, Ljubljana, Slovenia 2008.
[4] T. Kosuge, K. Nigorikawa, "STARS ON PLC", Proceedings of PCaPAC2010, Saskatoon, Canada, 2010.

ISBN 978-3-95450-124-3