DEVELOPMENT OF ACOP .NET STARS TRANSPORT LAYER

T. Kosuge[†], H. Nitani, Y. Nagatani, H. Ishii
High Energy Accelerator Research Organization, 3050801 Tsukuba, Japan
J. Szczesny, P. Duval, Deutsches Elektronen-Synchrotron, 22607 Hamburg, Germany

Abstract

Simple transmission and retrieval system (STARS) is an extremely simple and flexible software for small scale control systems with TCP/IP sockets. It is used by systems such as the beamline control system, and other systems at the KEK Photon Factory (KEK-PF). STARS works on various operating systems; therefore, STARS client developers can choose their preferred programming language. .NET is commonly used to develop graphical user interface (GUI) applications for beamline control at the KEK-PF.

Advanced component oriented programming (ACOP), which was developed by DESY, is very useful for GUI development, and a .NET version of ACOP was recently developed. ACOP communicates with various systems through a transport layer. We are currently developing ACOP .NET STARS transport layer. So far, we have succeeded in adding very primitive functionality.

OVERVIEW OF STARS

Simple transmission and retrieval system (STARS) is a software for small-scale control systems [1,2]. STARS consists of a STARS server and STARS clients. Each client is connected to the STARS server via a TCP/IP socket and handles text-based messages. The current version of the STARS server is written in Perl; therefore, STARS users can choose their preferred operating system and programming language for STARS client development.

Node Name and Hierarchical Structure

Every STARS client has its own unique node name that is used to identify the destination of the STARS text-

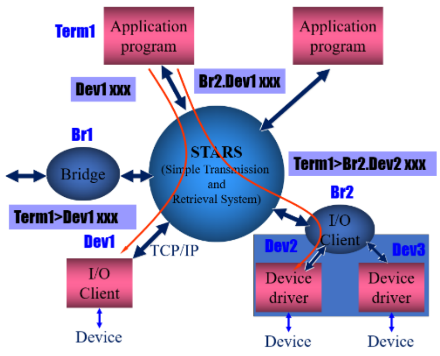


Figure 1: Message transportation on STARS.*

based message. Figure 1 shows examples of STARS message transportation. If a STARS client that has "Term1" as a node name sends a text-based message of the form "Dev1 xxx", the message will be delivered to the client "Dev1". Hierarchical structure node names that are separated with a period "." are available, and the server uses the first part of the node name as the destination.

Command, Reply and Event

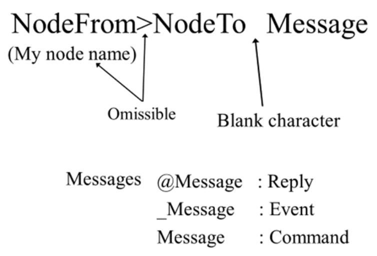


Figure 2: STARS message structure.*

Figure 2 shows the structure of a STARS message. The first part before ">" shows the origin of the message, and the next part before the first whitespace shows the destination of the message. The string which follows the first whitespace character can be either a command, a reply, or an event. A reply string starts with "@", an event string starts with "_", and a command string does not contain a preceding symbol. The strings can also contain values.

Event Delivery Function

STARS has an event delivery function. An event delivery request is registered by sending a "flgon" command to the server, which also has a node name "System". Figure 3 shows an example the event delivery function. "Tm1" and "Tm2" request an event from "Dev1" by sending "System flgon Dev1" to the server. After the registration, if "Dev1" sends an event to the server, the message is delivered to "Tm1" and "Tm2".

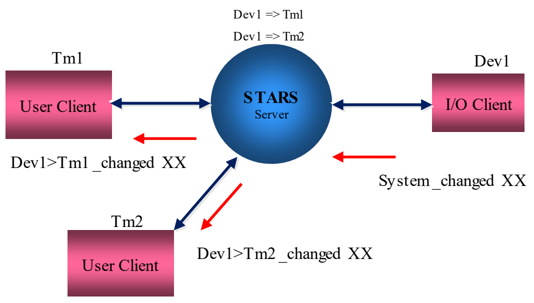


Figure 3: Event delivery function of STARS.*

† takashi.kosuge@kek.jp
* Source: http://stars.kek.jp/

Certification Procedure

STARS has a simple certification procedure.

- Host name certification
- Node name and keyword certification
- Node name and host name certification (optional)

The server checks the client's host name first. If the client passes host name certification, the server then sends a random number to the client. The client must then send its node name and keyword corresponding to the number. After keyword validation, the server checks node name and host name, if necessary.

STARS Interface Library

STARS is a very simple system, and the development of a STARS client is easy if the programmer can handle TCP/IP socket and text. However, there may be some difficulty for beginners. STARS provides interface libraries and tools that help in the development of a STARS client as follows:

- Perl interface library
- Python interface library
- C interface library
- Java interface library
- .NET interface library
- Perl STARS client development wizard
- Windows form C# template for Visual Studio
- Windows form VB .NET template for Visual Studio

Using the interface library, programmers can develop a STARS client while avoiding TCP/IP complexity.

STARS CLIENTS

STARS users can add new functions to their control system by developing STARS clients. STARS clients can be roughly classified into two types as IO clients and User clients. An IO client behaves like a device driver and controls hardware. User interface clients, including graphical user interface (GUI), are called User clients. A general purpose system can be developed if common commands are defined and a standard control system with STARS for KEK-PF beamline is introduced. Figure 4 shows a common beamline control system at the KEK-PF. The system provides the same command interface (getting values or setting values etc. of the hard ware). The beamline users can choose their preferred software such as LabVIEW, SPEC, common GUI program provided by the KEK-PF, or their own developed program.

Debugger

A client with the node name "Debugger" has a special purpose. When connected to the STARS server, the client can receive a copy of messages sent to all clients from the STARS server. "Debugger" works effectively during troubleshooting.

ACOP .NET

Advanced component oriented programming (ACOP) helps in the development of GUI programs [3]. The .NET version of ACOP was newly developed by DESY and is still under development.

ACOP tools appear in the Microsoft Visual Studio toolbox if ACOP .NET is installed (Figure 5). Powerful GUI tools (e.g., AcopChart and AcopLabel) can be used to choose and deploy the designer screen. ACOP supports TINE [4, 5] protocol, and programmers can develop the GUI without having to write the source code.

ACOP .NET STARS TRANSPORT LAYER

We started the development of ACOP .NET STARS transport layer to support STARS protocol. The primitive part of the transport layer was developed. ACOP .NET transport layer is written in C# and we added code for STARS.

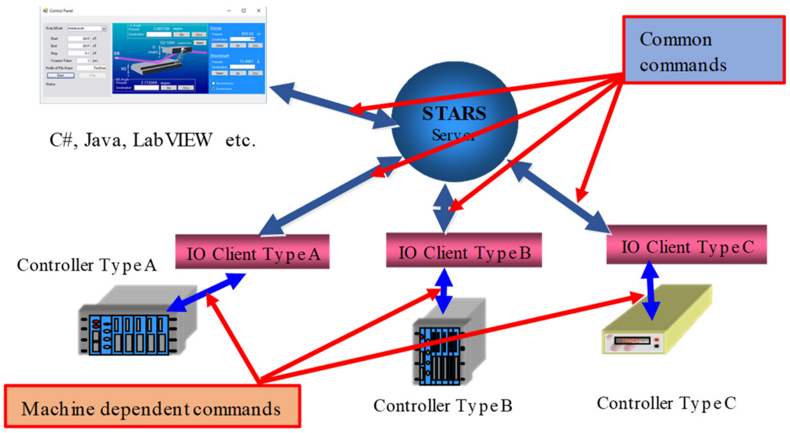
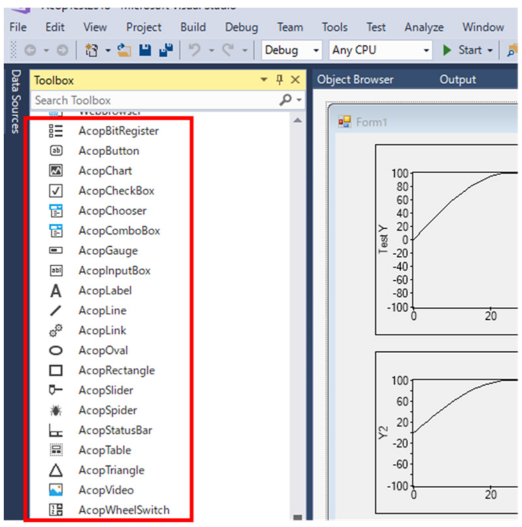


Figure 4: Standard control system for KEK-PF beamline that provides common interface commands.



AcopBitRegister
AcopButton
AcopChart
AcopCheckBox
AcopChooser
AcopComboBox
AcopGauge
AcopInputBox
AcopLabel
AcopLine
AcopLink
AcopOval
AcopRectangle
AcopSlider
AcopSpider
AcopStatusBar
AcopTable
AcopTriangle
AcopVideo
AcopWheelSwitch

Figure 5: ACOP tools in Toolbox of Microsoft Visual Studio.

STARS .NET Interface Library

ACOP .NET STARS transport layer communicates with the STARS server through STARS .NET interface library (Fig. 6), which processes the connection and the keyword certification procedure during connection.

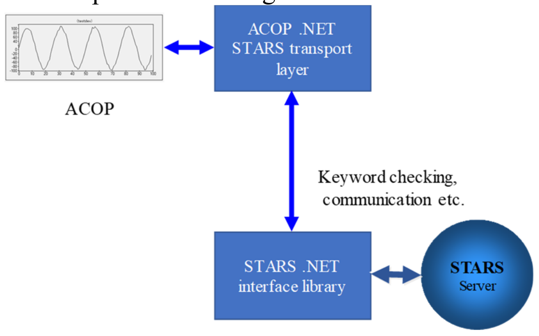

ACOP .NET STARS transport layer
ACOP
Keyword checking, communication etc.
STARS .NET interface library
STARS Server

Figure 6: ACOP .NET STARS transport layer and STARS .NET interface library.

Mapping of Name Space

Mapping of name space between ACOP and STARS is one of the primary issues. The procedure to allocate name space for device names, properties, and commands was discussed, and the first version of the mapping table was generated (Table 1).

Table 1: Name Space Mapping Table

ACOP Properties	STARS Name Space
Context	Node name of GUI
Subsystem	Not allocated
Server	Host name : Port number
Device	Node name of IO Client
Property	[Command /] Reply / Event

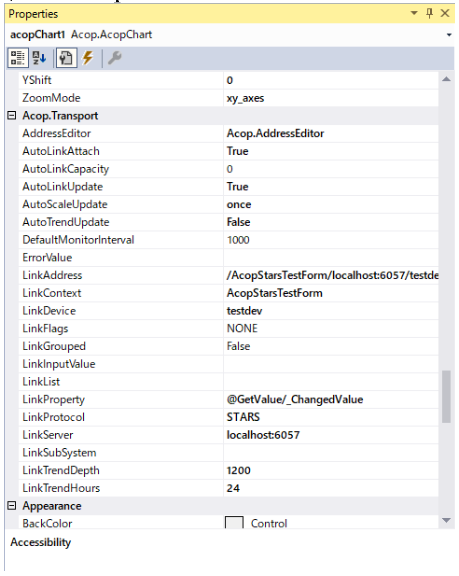
Context of ACOP is mapped to its own GUI node name of STARS. Server is mapped to host name and port number of STARS server, which are separated with the ":" character. Device is mapped to node name of STARS IO client that is being monitored by this GUI client. Property is mapped to the part of STARS command, reply, or event. Multiple messages can be defined using the "/" character as a separator. This first version of mapping will be improved after practical examination.

Attach and STARS Event

If the attach method in ACOP .NET STARS transport layer is called, the event delivery function of STARS is enabled by sending the "flgon" command to the STARS server, and the information of the request such as host name, node name, etc. is registered in the address list. The attach method does not send the "flgon" command again if the same information is in the list.

EXAMPLE OF PROGRAMMING

STARS Windows form template for Visual Studio (C# or Visual Basic) helps in the development of the STARS GUI Client. Once the new Visual Studio project is created with the template, primitive source code that handles the method of the STARS .NET interface library is created automatically. However, this means that the programmer must carefully examine the source code and handle functions for communication with the STARS server. ACOP .NET with STARS transport layer allows for source code with less programing on the part of the programmers. Figure 7 shows an example of ACOP chart properties in the property window of Visual Studio. The host name of STARS server, node name etc. can be set with ACOP by editing these transport properties. The programmer requests the procedure that chooses an ACOP component from the tool box, deploys the component on the form, and sets parameters.



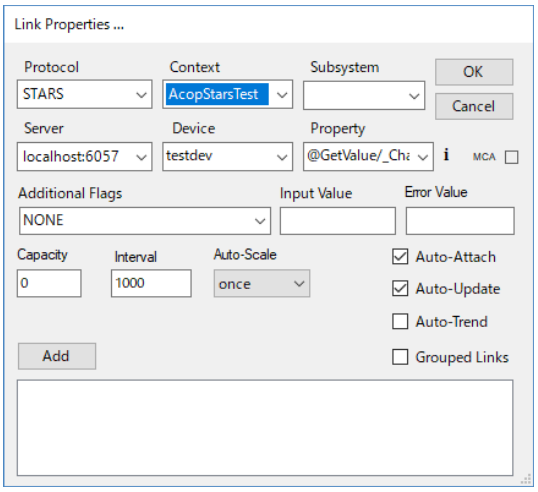
Properties
acopChart1 Acop.AcopChart

YShift	0
ZoomMode	xy_axes
⊟ Acop.Transport	
AddressEditor	Acop.AddressEditor
AutoLinkAttach	True
AutoLinkCapacity	0
AutoLinkUpdate	True
AutoScaleUpdate	once
AutoTrendUpdate	False
DefaultMonitorInterval	1000
ErrorValue	
LinkAddress	/AcopStarsTestForm/localhost:6057/testde
LinkContext	AcopStarsTestForm
LinkDevice	testdev
LinkFlags	NONE
LinkGrouped	False
LinkInputValue	
LinkList	
LinkProperty	@GetValue/_ChangedValue
LinkProtocol	STARS
LinkServer	localhost:6057
LinkSubSystem	
LinkTrendDepth	1200
LinkTrendHours	24
⊟ Appearance	
BackColor	Control

Accessibility

Figure 7: Property window of ACOP chart.

The ACOP address editor is a useful tool for setting ACOP transport properties (Figure 8). The editor window appears by clicking the button in the address editor field. "LinkAddress" property is generated automatically by entering "Context," "Server," "Device," and "Property" in the text fields of the address editor. If the "Auto-Attach" check box is marked, the "AutoLinkAttach" property becomes "True." The component is attached automatically when the program is started, and parameters are set if the "OK" button is clicked.


Link Properties ...

Protocol: STARS	Context: AcopStarsTest	Subsystem:
Server: localhost:6057	Device: testdev	Property: @GetValue/_Cha
Additional Flags: NONE	Input Value:	Error Value:
Capacity: 0	Interval: 1000	Auto-Scale: once

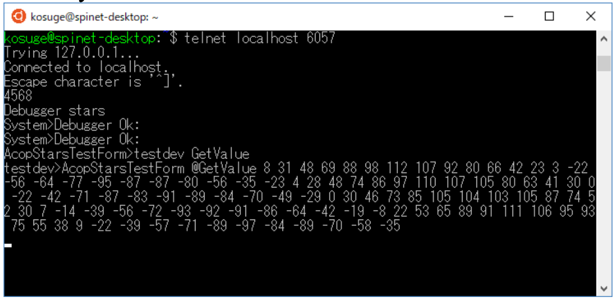
[✓] Auto-Attach
[✓] Auto-Update
[ ] Auto-Trend
[ ] Grouped Links
[ ] MCA

Figure 8: ACOP address editor.

Test with IO Client Simulator

We prepared an IO client simulator that returned values of the sine curve, added small noise values, and created a GUI client having an ACOP chart, and an ACOP button. Figure 9 shows a screenshot of the telnet terminal connected to the STARS server with the node name "Debugger." The "GetValue" command from the GUI named "AcopStarsTest" to IO client simulator named "testdev," and the reply message from "testdev" is observed on the screen.

STARS handles text-based messages only, and the IO client simulator returns values in text. Type-conversion (strings to integer, float etc.) is achieved by ACOP .NET STARS transport layer automatically. Figure 10 is a screenshot of a GUI client. We verified that the ACOP chart shows graph of values from the IO client simulator correctly.



```
kosuge@spinet-desktop:~$ telnet localhost 6057
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
4568
Debugger stars
System>Debugger Ok:
System>Debugger Ok:
AcopStarsTestForm>testdev GetValue
testdev>AcopStarsTestForm @GetValue 8 31 48 69 88 98 112 107 92 80 66 42 23 3 -22
-56 -64 -77 -95 -87 -87 -80 -56 -35 -23 4 28 48 74 86 97 110 107 105 80 63 41 30 0
 -22 -42 -71 -87 -83 -91 -89 -84 -70 -49 -29 0 30 46 73 85 105 104 103 105 87 74 5
2 30 7 -14 -39 -56 -72 -93 -92 -91 -86 -64 -42 -19 -8 22 53 65 89 91 111 106 95 93
 75 55 38 9 -22 -39 -57 -71 -89 -97 -84 -89 -70 -58 -35
```

Figure 9: Screenshot of debugger.

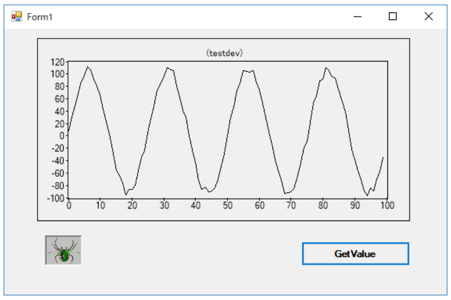
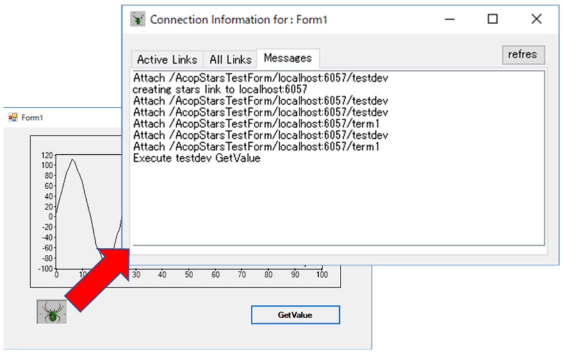


Figure 10: Screen shot of STARS GUI client with ACOP.

ACOP Spider

ACOP has a debugging tool called ACOP spider. The debugging function is enabled by deploying "AcopSpider" on the form. If the "AcopSpider" icon is clicked when the program is running, the debugging window appears on the screen and shows the connection information. The debugging window has "Active Links," "All Links," and "Messages" tabs. Figure 11 shows the debugging window with the "Messages" tab active. We verified that the ACOP spider works satisfactorily on debugging as well.



```
Attach /AcopStarsTestForm/localhost:6057/testdev
creating stars link to localhost:6057
Attach /AcopStarsTestForm/localhost:6057/testdev
Attach /AcopStarsTestForm/localhost:6057/testdev
Attach /AcopStarsTestForm/localhost:6057/term1
Attach /AcopStarsTestForm/localhost:6057/testdev
Attach /AcopStarsTestForm/localhost:6057/term1
Execute testdev GetValue
```

Figure 11: Debugging with ACOP spider.

CONCLUSION

We have succeeded in developing a first version of ACOP .NET STARS transport layer and obtained sufficient results upon making a GUI STARS client with ACOP. The name space mapping between ACOP and STARS still needs to be refined. Checking all the ACOP components and replacing the name space mapping table (if necessary) will be the next steps in the development.

REFERENCES

[1] STARS, http://stars.kek.jp

[2] T. Kosuge and Y. Nagatani, "STARS: Current Development Status," in *Proc. PCaPAC'14*, Karlsruhe, Germany, Oct. 2014, paper WPO019, p. 75.

[3] I. Krznar *et al.*, "New ACOP Beans and TINE General Purpose Diagnostic Applications," in *Proc. ICALEPCS'09*, Kobe, Japan, Oct. 2009, paper TUP034, p 161.

[4] TINE, http://tine.desy.de

[5] P. Duval and S. Herb, "The TINE Control System Protocol: How to Achieve High Scalability and Performance," in *Proc. PCaPAC'10*, Saskatoon, Canada, Oct. 2010, paper WECOAA02, p. 19.