Overview of JOpenShowVar v0.2 for Kuka Robots

F. Sanfilippo, L. I. Hatledal and H. Zhang
Department of Maritime Technology and Operations
Aalesund University College
Postboks 1517, 6025 Aalesund, Norway
{fisa, laht, hozh}@hials.no

K. Y. Pettersen
Department of Engineering Cybernetics
Norwegian University of Science and Technology
7491 Trondheim, Norway
kristin.y.pettersen@itk.ntnu.no

INTRODUCTION

JOpenShowVar, a Java open-source cross-platform communication interface to Kuka robots, was previously introduced by our research group [1]. JOpenShowVar allows for read-write use of the controlled manipulator variables and data structures. This interface, which is compatible with all Kuka robots that use KR C4 and previous versions, runs as a client on a remote computer connected with the Kuka controller. JOpenShowVar opens up to a variety of possible applications, making both the use of various input devices and sensors as well as the development of alternative control methods possible.

In this work, a first overview of the new, more flexible and efficient, procedures that have been introduced in the latest release of the library is introduced. These new methods replace the old fundamental reading and writing method that is now marked as deprecated. On top of these new methods, some other high-level functions can be built for angles and torques reading of the controlled manipulator.

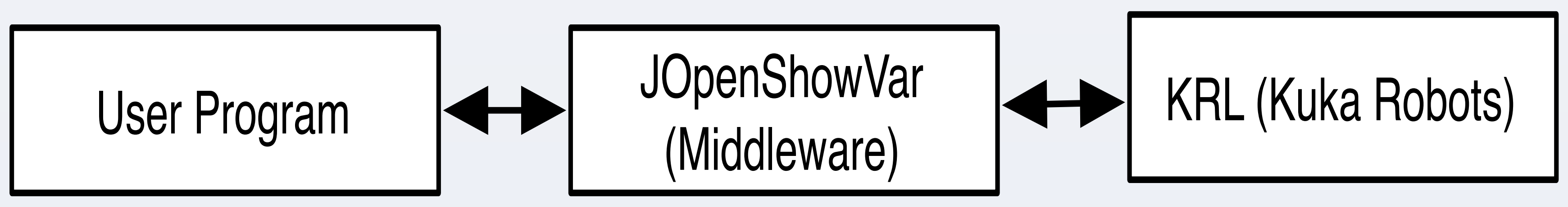

Fig. 1 The idea of realising a communication interface for Kuka robots that works as a middleware between the user program and the Kuka Robot Language (KRL).

BACKGROUND

JOpenShowVar works as a middleware between the user program and the KRL, as shown in Fig. 1. The proposed control system architecture is shown in Fig. 2. It is a client-server architecture with JOpenShowVar running as a client on a remote computer and KUKAVARPROXY acting as a server on the KRC. JOpenShowVar locally interacts with the user program and remotely communicates with the KUKAVARPROXY server via TCP/IP.

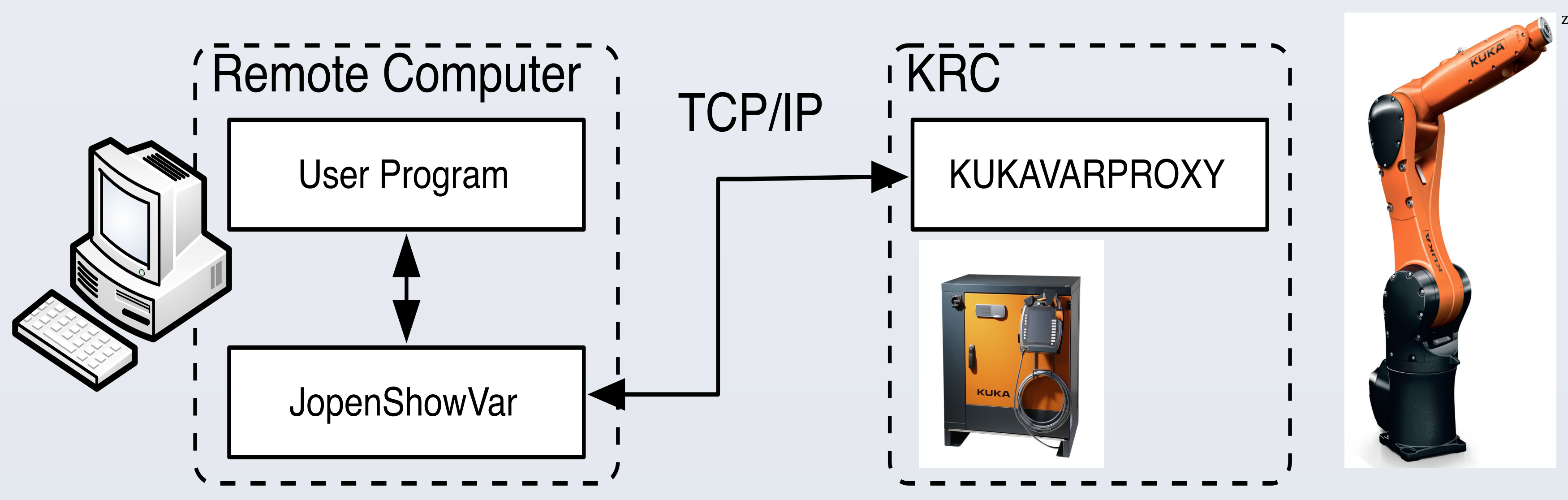

Fig. 2 The proposed architecture for JOpenShowVar: a client-server model is adopted.

In particular, KUKAVARPROXY is a multi-client server that is written in Visual Basic 6.0 and can serve up to 10 clients simultaneously. KUKAVARPROXY implements the Kuka CrossComm class. The interface of this class allows for the interaction with the real-time control process of the robot and makes it possible to perform several operations such as selection or cancellation of a specific program, errors and faults detection, renaming program files, saving programs, resetting I/O drivers, reading variables and writing variables. KUKAVARPROXY implements the reading and writing methods. All the variables that need to be accessed by these methods have to be declared as global variables in the predefined global system data list $CONFIG.DAT. All kinds of variables can be declared in this file from basic types such as INT, BOOL and REAL to more complex structures like E6POS and E6AXIS that allow for storing the robot configuration. Moreover, several system variables can be accessed provided there are no restrictions due to the type of data such as for $PRO_IP, $POS_ACT, $AXIS_ACT or $AXIS_INC. For example, the current robot position, $POS_ACT, cannot be written but only read. Restrictions of this nature are checked by the controller. However, it should be noted that the Kuka CrossComm class does not provide a real-time access to the robot's data. In fact, it takes a non-deterministic time to access a specific variable. Since Kuka does not offer any kind of documentation on this topic, several experimental tests were performed at our laboratory in order to asses this time interval. According to our experiments, the average access time is about 5 ms. Moreover, this time interval is not affected by the kind of access to be performed (whether it is a reading or a writing operation) or by the length of the message. For these reasons, it is advantageous to aggregate several variables in logical structures when reading or writing data. By using data structures it is possible to simultaneously access several variables, thereby minimising the access time. The only limitation to this approach is on the length of the logical structures that cannot exceed 255 bytes.

JOpenShowVar provides a client, CrossComClient, which is written in Java, thus making cross-platform support possible. As presented in our previous work [1], the client initially provided only one low level method, sendRequest. This method allows for both reading and writing variables. The sendRequest method returns a Callback instance containing the updated value. However, in the latest release of JOpenShowVar, starting from version v0.2, the sendRequest is marked as a deprecated method, since two new more flexible and efficient methods are introduced: readVariable and writeVariable.

NEW METHODS

Since the release version v0.2 of JOpenShowVar, the sendRequest is marked as a deprecated method. To replace this old method, two new more reliable methods are added to the CrossComClient:

- the readVariable method allows for reading any desired remote variable or structure from the controlled robot and store it locally. An exception is thrown if an error in the communication protocol occurs;
- the writeVariable method allows for updating any desired remote variable or structure of the controlled robot with the value of the corresponding local variable or structure, respectively. An exception is thrown if an error in the communication protocol occurs.

The deprecated sendRequest method is being kept as part of the JOpenShowVar library simply because the GUI still uses it for a practical reason. In fact, this old method does not require any knowledge on the internal structure of the variables to be accessed compared to the newly introduced methods. It should be noted that the new method writeVariable cannot handle arrays, this can only be done by using the old sendRequest method. In the Algorithm 1 sketch box a possible use-case example is shown.

```
try (CrossComClient client = new CrossComClient("
                      158.38.140.193", 7000)) {
//JOpenShowVar v0.1 reading
Callback readRequest = client.sendRequest(new Request(0,
                      "$OV_JOG"));
System.out.println(readRequest);
//JOpenShowVar v0.1 writing
Callback writeRequest = client.sendRequest(new Request(1,
                      "$OV_JOG", "100"));
System.out.println(writeRequest);
//JOpenShowVar v0.2 reading
KRLReal jog = KRLVariable.OV_JOG();
client.readVariable(jog);
System.out.println(jog);
//JOpenShowVar v0.2 writing
jog.setValue(10);
client.writeVariable(jog);
System.out.println(jog); }
```

Algorithm 1 A use-case example that highlights the differences between the new methods and the deprecated sendRequest method.

CONCLUSIONS

This paper highlights the features of JOpenShowVar latest release. Two more flexible and efficient, procedures are introduced in the latest version of the library to replace the old fundamental reading and writing method that is now marked as deprecated. In the future, different control algorithms such as the ones implemented in [2], [3] and [4] may be tested as alternatives to the standard kinematic method.

REFERENCES

[1] Filippo Sanfilippo, Lars Ivar Hatledal, Houxiang Zhang, Massimiliano Fago and Kristin Ytterstad Pettersen. JOpenShowVar: an Open-Source Cross-Platform Communication Interface to Kuka Robots. In Proceeding of the IEEE International Conference on Information and Automation (ICIA), Hailar, China. 2014, 1154-1159.

[2] Filippo Sanfilippo, Lars Ivar Hatledal, Hans Georg Schaathun, Kristin Ytterstad Pettersen and Houxiang Zhang. A Universal Control Architecture for Maritime Cranes and Robots Using Genetic Algorithms as a Possible Mapping Approach. In Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO), Shenzhen, China. 2013, 322-327.

[3] Filippo Sanfilippo, Lars Ivar Hatledal, Houxiang Zhang and Kristin Ytterstad Pettersen. A Mapping Approach for Controlling Different Maritime Cranes and Robots Using ANN. In Proceeding of the IEEE International Conference on Mechatronics and Automation (ICMA), Tianjin, China. 2014, 594-599.

[4] Lars Ivar Hatledal, Filippo Sanfilippo and Houxiang Zhang. JIOP: a Java Intelligent Optimisation and Machine Learning Framework. In Proceedings of the 28th European Conference on Modelling and Simulation (ECMS), Brescia, Italy. 2014, 101-107.

CONTACTS

Filippo Sanfilippo is a PhD candidate in Engineering Cybernetics at the Norwegian University of Science and Technology and research assistant at the faculty of Marine Technology and Operations, Aalesund University College. Supervisors: Professor Kristin Ytterstad Pettersen, Professor Houxiang Zhang and Professor Domenico Prattichizzo.

Email: fisa@hials.no