Remote control of CAN-based industrial equipment using Internet technologies

Prof. Dr.-Ing. **Gerhard Gruhler,** University of Applied Sciences Reutlingen,
Steinbeis Technology Transfer Center Automation (STA), Germany

The paper presents different methods for observing, monitoring, parameterising and directly controlling CAN-based automation equipment using Internet technologies. Examples that are described are: a commercial industrial robot, a cartesian handling system, a workpiece separation unit and other CAN devices. Furthermore generic interfaces are presented to access CAN systems via the internet on CAN layer 2 and to access CANopen modules using the CANopen object dictionary / electronic data sheet.

Since industrial equipment tend to be located behind firewalls, a method was developed and is described to get access to CAN even through firewalls and proxies.

For demonstration purposes the described systems are online 7 days / 24 hours and can be accessed by everyone from all over the world.

1 Introduction

Obviously CAN is a bus system with an extremely wide range of possible applications. CAN is not only used to great extent in cars and other mobile applications but also in industrial equipment, machine control, building control, medical and other devices in the service/public area. Due to the global market place, in most these fields remote system access becomes more and more important. Where in former years peer-to-peer telephone connections have been used, nowadays internet-based communication (fig. 1) offers big advantages.

These are e.g.:

- more reliable communication channels,
- use of PC-based standard technologies and software like TCP/IP-stack, web servers, browsers, Java, XML etc.
- significant cost reduction for development, equipment and operation
- less effort in particular on the client´s side: usually the client just needs a standard web browser,
- worldwide availability of internet access at low costs.

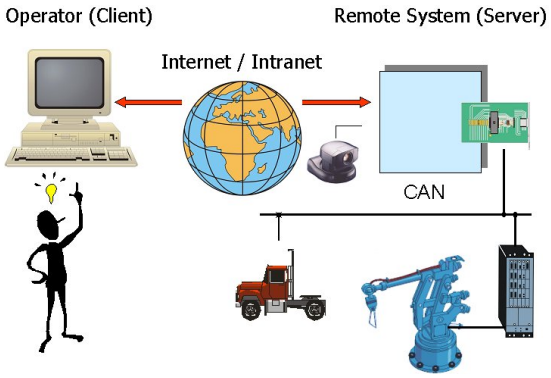

Fig.1: Remote CAN access via internet

The range of possible remote access applications and the related benefit is large:

- remote production and machine data collection,
- remote system monitoring,
- remote error detection,
- remote observation purposes, also for security reasons,
- customer support for programming and running the system, help for bug fixing,
- remote device parameterisation,
- software/firmware updates via internet,
- better support for maintenance staff,
- sometimes even avoiding service trips,

- direct system control by remote operator interface,
- and last but not least better marketing support: some device manufacturers not only provide the usual bunch of documentation on their web-site but start to make accessible real devices for demonstration purposes to their perspective customers.

It is self-contained that a dedicated security concept is and must be an inherent part of any internet-based remote access system, some times only providing a well-defined and very limited set of functions to the remote user.

Related to the very different remote access scenarios, varying methods have been developed which provide system access at different levels. Examples hereof are described in this paper and may be accessed worldwide anytime by everybody at http://robo16.fh-reutlingen.de, button “Demonstrations”. All these examples do not require the implementation of any specific software on the client´s computer, only a Java-enabled web browser is needed. Most of these concepts can be applied generally, some even allowed us to develop generic remote access software to a wide range of CAN-based equipment.

2 High-level web interfaces

High-level remote access concepts are usually provided for non-expert system users. The system (or parts of it) is usually presented to the remote client by graphical user interfaces which allow access only to certain data or specific control functions. Remote access in this case must definitely be fail-safe. In-system CAN nodes are affected but they are as such completely transparent to the operator.

Fig. 2 gives a first example for a high-level web-interface to a commercially available industrial robot system in the Reutlingen CAN- and Telematics Lab.

For demonstration of the fail-safe behaviour only a set of predefined application programs and procedures can be activated via Internet. CAN is used as a link between the web server and the external control interface of the robot. Actual status information of the robot system is displayed online in the remote operator interface. During application program execution sensor and production data are generated.

The data file is transferred via TCP/IP to the web server where a dynamic HTML page is generated. The data are then displayed in the clients´s web browser by accessing the HTML file.

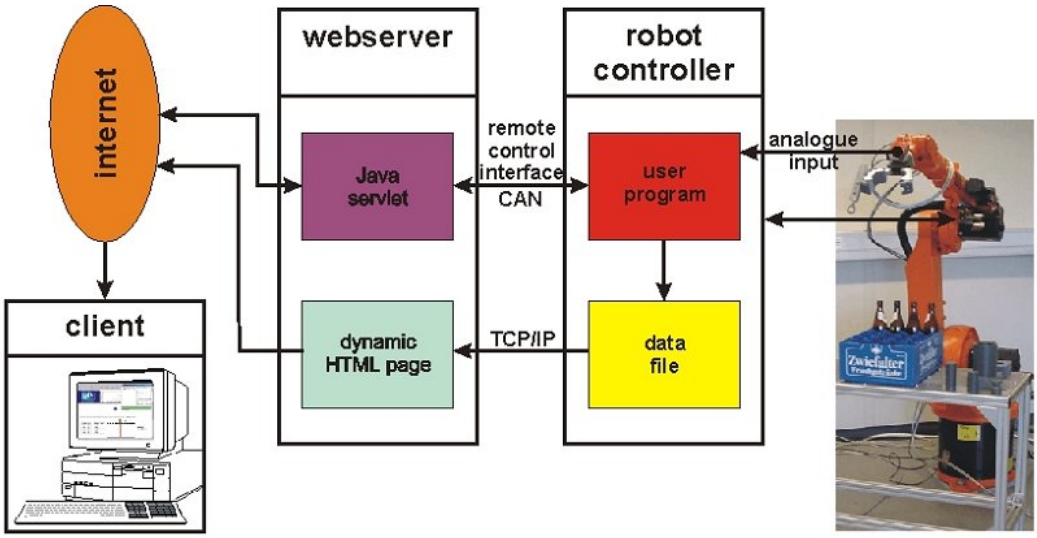

Fig 2.: High-level remote access to a KUKA robot via internet and CAN

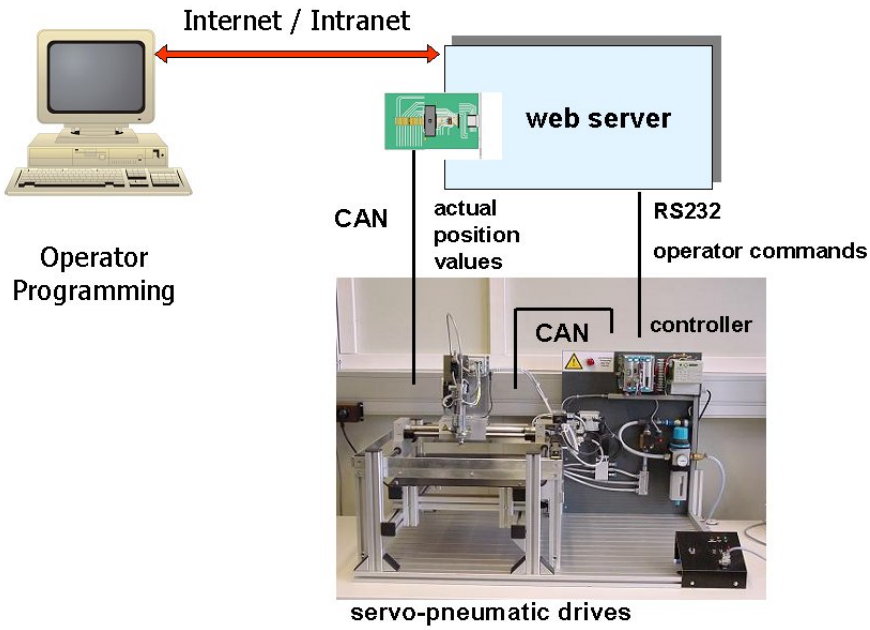

Fig. 3: High-level remote programming and control concept for a FESTO handling robot using CAN

An additional example is given by remote programming and control of a cartesian handling system (fig. 3). The web-server is connected via serial link to the embedded system controller. CAN is used internally to connect the servo-pneumatic drives and externally to provide the actual position values of the axes to generate a trace file on the server.

The system can be programmed by the remote user either textually (using the complete application programming language command set) or graphically by defining robot movements via the web-based operator interface. In either case the resulting application program is transferred over the web to the remote system for testing and execution. During program execution the actual position values are transferred via CAN in real-time to the web server where a trace file is generated. This trace file is then transferred over the web and additionally displayed in form of a trajectory in the client's browser (fig. 4).

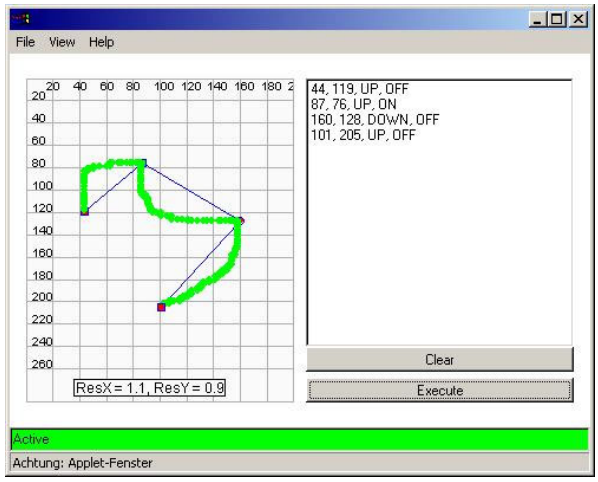

Fig. 4: Web-based graphical programming interface

A particularly nice example for high-level web-based control and monitoring interface is shown in fig. 5

As a demonstrator, a workpiece separation unit is used. The machine is controlled via off- the-shelf CANopen I/O-modules. The control algorithm in form of a Java program can be transferred to and is running on the web server where the CANopen system is connected. Due to our experience the client is usually very fond of not only getting data from but also an optical impression of the remote system. Instead of consuming a lot of internet bandwidth by

transfering live images from a web camera, only a photo of the actual machine is used. The photo is then overlaid by frequently up-dated on-line status information. This information only contains a few bytes representing the state of sensors and pneumatic components and is shown as color-changes to the user.

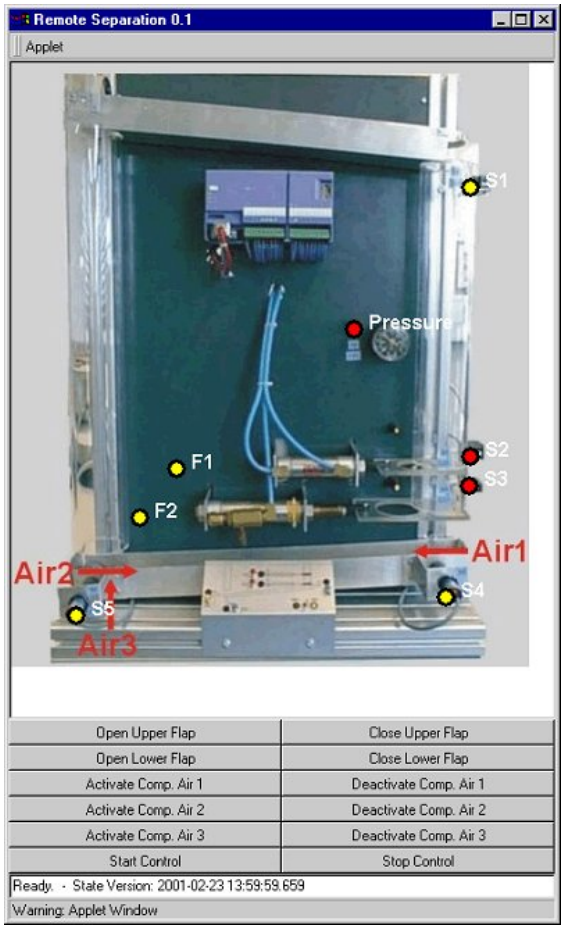

Fig. 5: Web interface of a CANopen-based workpiece separation unit with on-line up-date of status information

It is common to all examples of this section that CAN is transparent to the user. Nevertheless, the system developer has to interface CAN/CANopen system to web severs. Fortunately this interface software can be implemented in a generic way and can be mainly the same as used for the examples of the following sections.

3 Generic internet access to CANopen devices via object dictionary

While high-level web interfaces usually represent CAN-based systems as a whole and provide limited fail-safe access, maintenance staff need much deeper insight and direct access to all parameters of a certain CAN node. Therefore an additional software solution was developed which allows access to generic CANopen devices using their object dictionaries. Fig. 6 shows the related Java applet.

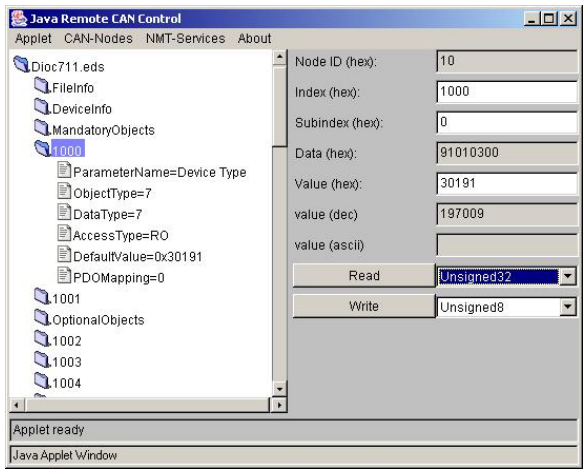

Fig. 6: Web-interface for generic CANopen nodes

The electronic data sheet (EDS) of the node is displayed in several detailing layers (left part of control applet in fig. 6). The right hand part of the applet allows read and/or write access to all parameters of the object dictionary via internet. Addionally, the interface allows to switch to all CANopen nodes in a network and to alter their states due to the CANopen state diagram. This is quite a powerful tool for monitoring, error detection and parameterisation purposes. In due course, a more or less skilled operator is expected on the client side.

The software implementation is mainly based on Java. The Java user interface is sitting on top of a Java CANopen stack. The low layer driver software for the CAN controller which is provided by the manufacturer of the CAN adapter card (CANcardX) is still written in C. Therefore Java Native Interface (JNI) is currently used for interfacing the driver software. Our most recent research on real-time Java indicates that it is quite possible nowadays

to implement all layers of CAN software even down to the hardware-related parts in Java. Java hardware processors (e.g. aJile) execute directly Java code and provide therefore a very high real-time performance.

4 Remote access to and monitoring of lower CAN layers

A still more generic approach provides monitoring and read/write access directly to CAN messages on communication layer 2. This concept is therefore useful not only for systems featuring one of the standardised higher layer protocols but also for proprietary networks.

The software called CAN Massage Manager (user front end see fig. 7) allows to define the object identifier and the content of the up to 8 data bytes of a message directly by the operator. Messages occurring on the bus are displayed in the field below.

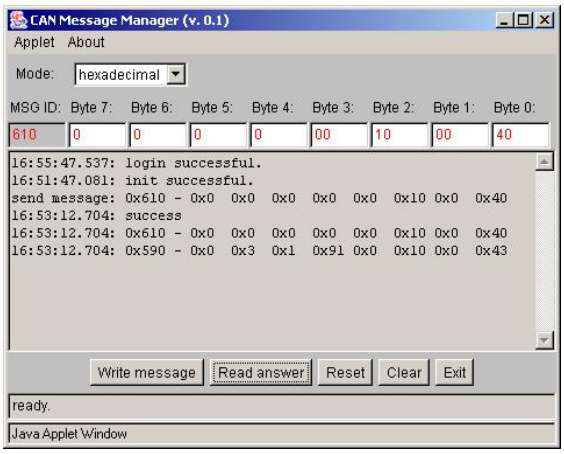

Fig. 7: CAN Message Manager's web interface

Since the operator interface of the CAN Message Manager software is of rather low complexity, this software is of particular interest for remote education and training on CAN systems.

A completely different approach – still using the internet to access CAN layer 2 – features the following example. The idea is to set-up CAN-TCP/IP-CAN gateways by wrapping CAN telegrams into TCP/IP messages and vice versa (fig. 8 and 9). This allows to "extend" CAN systems over the internet e.g. for analysing purposes. Since the access point is CAN on both sides, all existing CAN-based software tools may be applied remotely by using this concept. Due to the usual delay time caused by the internet, there is of course no real-time performance. Time stamps will be invalid and watchdogs etc. will have to be deactivated. To limit necessary internet bandwidth, the gateway software may be configured to filter and transfer only interesting CAN telegrams, all other massages remain just local.

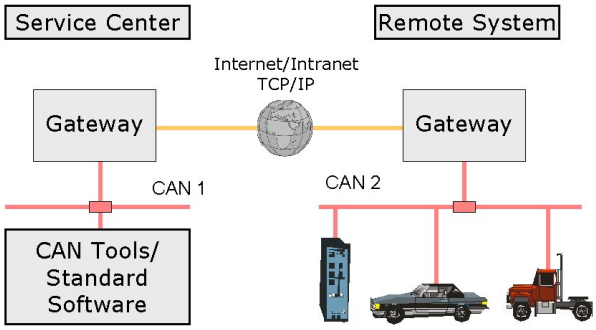

Fig. 8: Concept of extending CAN systems over the internet using CAN-TCP/IP-CAN gateways

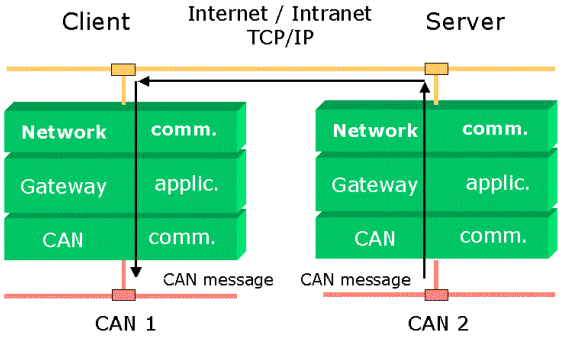

Fig. 9: Software structure of CAN-TCP/IP-CAN gateways

In a lab experiment we have been able to demonstrate the use of a standard CAN analysing tool (Vector CANalyzer) to monitor remote CAN segments over the internet using the described gateways. This is particularly useful in case the occurance of certain specific messages (like e.g. error frames) on the remote system is to be detected. The gateways can be used as small stand-alone embedded modules or just the gateway software stack is used running on PC´s on both sides. Due to the "symmetrical" gateway solution this is the only one of the presented remote CAN access

concepts where at least specific software installation is required on both, the client and the server system. All other concepts only require a Java-enabled standard web browser on the client side.

5 PC-based and embedded systems for web server and client

Since PC-based control systems become increasingly widespread, the use of internet technologies is also growing rapidly. On the other hand CAN has particular advantages also in low cost system where PC technology is still too expensive. However, these low-cost systems can also be connected to the internet by use of low-cost embedded web servers. Fig. 10 gives examples of embedded web server modules with CAN support.

Fortunately software components like embedded operation systems, TCP/IP-stacks, Java Virtual Machines, web server packages etc. are often provided free of charge also for embedded systems. Consequently, a rough estimation shows that complete web server modules with CAN interface can be implemented for far below 100,- EUR (snap-on module, including housing and connectors).

Fig. 10: Examples of embedded web servers, partially available with CAN interface

Also on the client side embedded systems like handheld computers, web pads or even mobile phones are more and more in use. The more internet compatible technologies (e.g. Java) are employed to implement the access system, the less effort is required to support embedded servers and clients.

6 Access to CAN systems through firewalls

Frequently it is to be expected that either the client or the accessed CAN system are located behind firewalls. In certain cases firewalls prevent the remote system access Operating systems communicate with remote resources by so called communication ports. These ports are identified by their number. “Well-known” ports carry a specific number and are connecting to well-known software. E.g. a standard web-server, using the HTTP protocol, is connected to port 80 of an operating system. One important functionality of firewalls is to prevent communication through others than well-known ports.

Therefore it is mondatory for remote access systems to overcome firewalls by only connecting via well-known ports.

Fig. 11 shows a system concept not able to communicate through firewalls. Reason is, the connection between Java Server and Java Applet on the client´s computer is established using not pre-defined port numbers.

Fig. 12 illustrates the modified concept, capable of passing firewalls. All communication is done via HTTP. The local Java resources (Applets, Servlets) on either side are only communicating via web server respectively web browser.

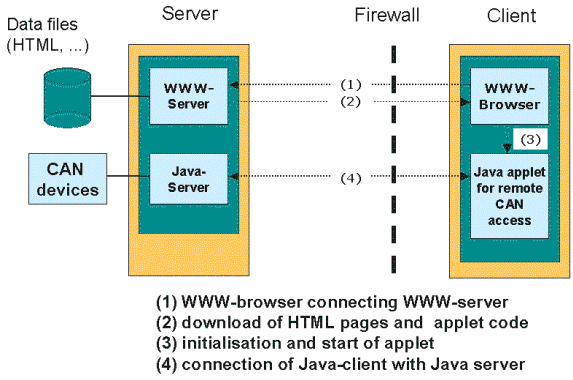

Fig. 11: Remote access concept, not capable of passing firewalls.

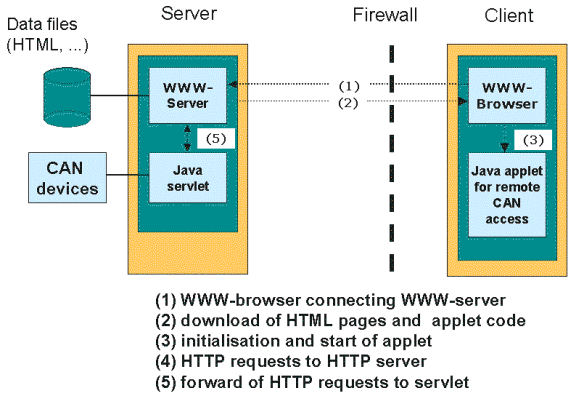

Fig. 12: Modified access method, to overcome firewalls

7 Summary

Various concepts and examples have been presented to access CAN-based devices on different levels over the internet. Respective demos are available 24 hours / 7 days at the internet address http://robo16.fh-reutlingen.de.

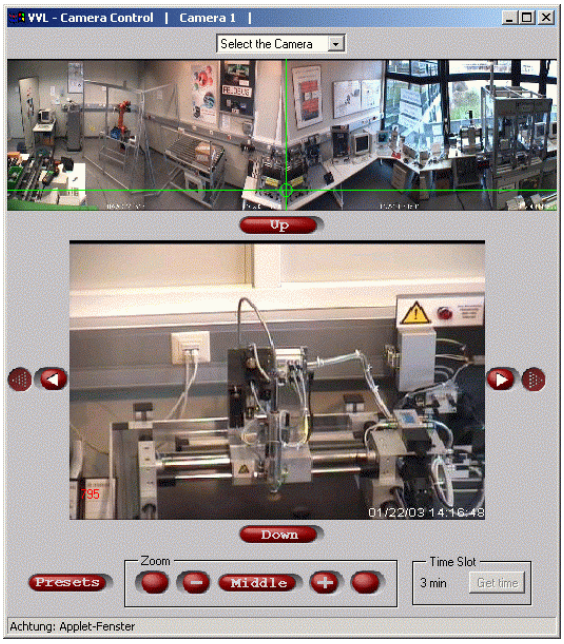

Fig. 13: User interface of interactive web cameras of CAN- and Telematics Lab in Reutlingen (camera control elements, panorama view and live image)

Since the access is not restricted, the demos may be operated and additionally observed by web cameras every time from everywhere.

References

[1] D. Bühler; G. Nusser; G. Gruhler; W. Küchlin: A Java Client/Server System for Accessing Arbitrary CANopen Fieldbus Devices via the Internet. South African Computer Journal 24, Nov. 1999, pp. 239…234.

[2] G. Gruhler; G. Nusser; D. Bühler; W. Küchlin: Teleservice of CAN Systems via Internet. Proceedings of the 6th International CAN Conference ICC´99, 2 – 4. Nov. 1999, Torino, pp. 06 – 02..06 – 09. Erlangen: CAN in Automation 1999.

[3] G. Gruhler: New Experiments for Tele-Education in Robotics. Proceedings of the 2nd Workshop on Tele-Education in Engineering Using Virtual Laboratories, Sherbrooke, Canada, August 8 – 9 2002.

[4] D. Schmid; G. Gruhler; A. Fearns (Eds.).: Proceedings of Symposium Telelaboratorien, 14th May 2003, Aalen, Germany

[5] http://robo16.fh-reutlingen.de

Address:

Prof. Dr.-Ing. Gerhard Gruhler
Institute for applied research in Automation (IFA)
Steinbeis Technology Center Automation (STA)
Reutlingen University
Alteburgstrasse 150
D-72762 Reutlingen, Germany
email: gerhard.gruhler@fh-reutlingen.de
Website: www-sta@fh-reutlingen.de