TCP/IP Communication for a De-Manufacturing Transport Line

David A. Amezquita Martinez
Mechatronics Engineering Student
Universidad Nacional de Colombia
Bogota
daamezquitam@unal.edu.co

Andrea Cataldo
Institute of Industrial Technologies and Automation
Milan, Italy
andrea.cataldo@itia.cnr.it

Ricardo E. Ramírez Heredia
Associate Professor
Universidad Nacional de Colombia
Bogota, Colombia
reramirezh@unal.edu.co

***Resumen*-Communication among machines in a production system is important as it provides synchronism and fluency in the manufacturing processes. Set out below is the development of the TCP/IP communication for a pallet transport line system intended to improve the working of the plant. This was carried out starting with the comprehension of the line and the machines in interaction, understanding the programs previously developed and setting up the TCP/IP functions in ISaGRAF. A human machine interface in Movicon was employed to supervise and test the line. The communication via TCP/IP was established, the protocol among servers and machines was correctly implemented and it was possible to supervise the whole process and carry out a real test using the human machine interface.**

***Palabras claves**: de-manufacturing; HMI; TCP/IP; Functions of communication.*

I. INTRODUCTION

In recent years, research around the productive sector has been oriented towards possible technologies to face the environmental contamination using de-manufacturing technologies. As a result, more economic benefits reducing costs, avoiding waste and minimizing pollution are seen in the short term. De-manufacturing means to recycle and disassemble obsolete electronics such as computers, printers, modems, memories, motherboards and telephones in order to reuse materials like glass, metal, wires, batteries and printed circuit boards materials.

A de-manufacturing transport line is characterized by a modular flexible structure based on transport modules which consist of moving pallets along different operating cells. The line consists of single modules connected to each other in order to move pallets among different operating stations. (Cataldo, Brusaferri, 2013). The pallet has been designed to transport electronic boards that have to be tested, repaired or disassembled. The control *software* used in the development process of the line is based on the ISaGRAF environment. ISaGRAF is a *software* to develop, debug, execute and test the functions implemented in Sequential Functional Chart (SFC) language.

The aim of the transport line is to apply de-manufacturing in Printed Circuit Boards (PCB), carrying it by a pallet among different cells (Figure 1).

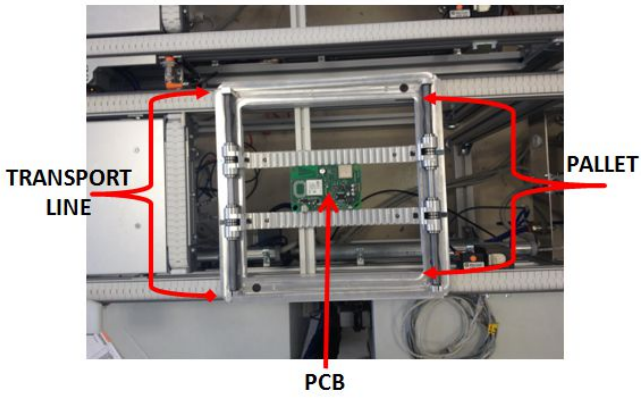

Figure 1. PCB on Pallet in the Transport Line System.

As shown in the Figure 2, the system has four different operating stations:

- M1 (Robot cell): two robotic arms load the pallets with the PCB and unload the pallets on the transport module (S1).
- M4 (SPEA machine): the device tests the PCB identifying whether it can be repaired or it must be carried away. Part of the communication process includes sending a damage report to the ZEVAC machine where the type of failure is identified.
- M6 (ZEVAC machine): it repairs the PCB unsoldering the damaged components and soldering new ones.
- S2 (Discharge Board): PCBs that cannot be repaired must be unloaded in this cell to be carried to the recycle cell (M3).

Figure 3 presents the flow chart of the general process. First of all, the PCB joins the system through a coordinate movement between the two robotic arms to load the PCB into the pallet.

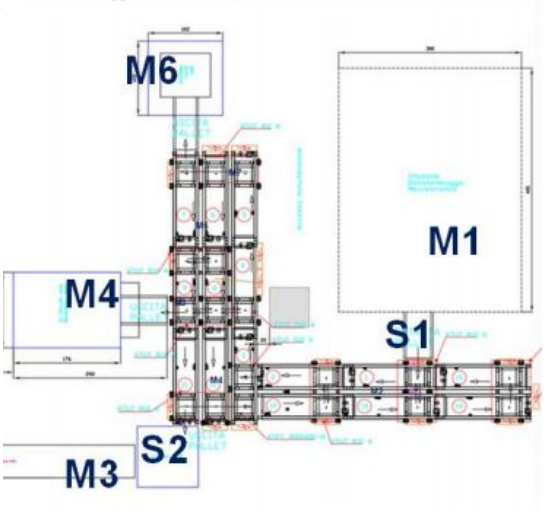

Figure 2. Top View of the Transport Line System.

The first station of the pallet is the machine SPEA, where the PCB is tested in order to know its state. If the PCB is in bad condition and is not possible to fix, the PCB will be taken to the discharge cell. But if the PCB is in bad condition and is reparable, it will be taken to the ZEVAC machine. Then, the PCB will be carried to the SPEA again in a cyclical way. If the PCB is repaired, it will be carried to the robot cell and taken out of the line.

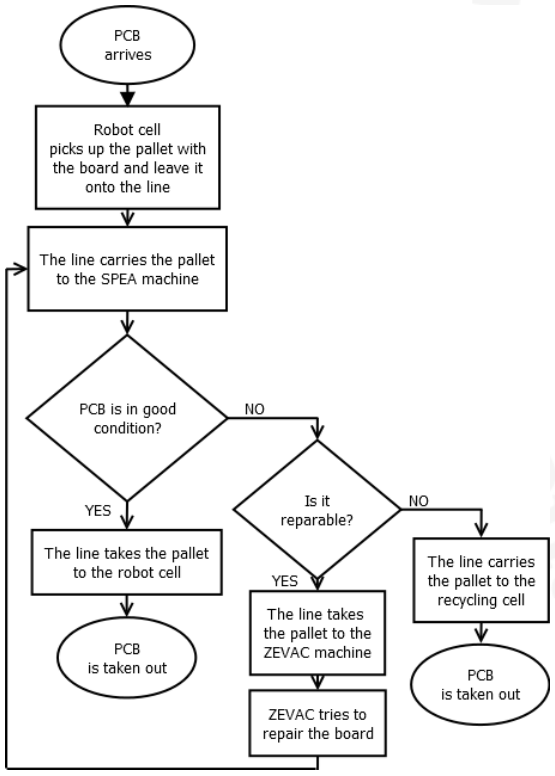

Figure 3. Flow Chart of the general process.

The de-manufacturing transport line has a monitoring station that is able to control all the remote inputs and outputs (I/O) modules through a PC. By contacting an established IP address the browser screens the transport module automation devices, displaying the status of I/O signals. A better supervision of the devices, the I/O signals and the modules is possible by the use of a human machine interface (HMI). A HMI system supervises the production, displays the execution of the plant processes, and allows the data saving. The HMI for the plant is implemented in Movicon with the main features.

Given a control platform for the mentioned industrial transport line, the purpose of this work was to develop the data communication functions via TCP/IP among the control system and the machines on board. The communication was implemented using the ISaGRAF *software* with the established protocols and the socket-server synchronism. Besides, a HMI was developed in Movicon in order to monitor the plant and to test the work done.

This paper is organized as follows: section II shows the implementation of the communication via TCP/IP; section III presents the development of the HMI for the plant; in section V the results are displayed; and section VI shows the conclusions and further research on the field.

II. Communication Via TCP/IP

Given a control platform for a de-manufacturing transport line, it was necessary to communicate via TCP/IP the general system, the servers and the machines. The synchronism of the process variables provides data exchange among the transport line, supervision and the machines in order to guarantee the correct operation.

The Transmission Control Protocol/Internet Protocol (TCP/IP) was created by the Department of Defense to ensure and preserve data integrity, as well as maintain communications in the event of catastrophic war [5]. TCP/IP enables the computers to communicate over a network. Its model describes the data format, i.e., how the data should be transmitted, addressed, routed and received.

A. General Overview of the ISaGRAF Project

ISaGRAF is a distributed, scalable architecture which comprises both a *hardware controller* and a *software* environment [3]. ISaGRAF allows the control and updates in a system from a single workstation, where it is possible to run several projects. Figure 4 shows the deployment view of the general project in ISaGRAF.

There are three main modules: Module Line Supervisor, Submodules 1 to 8, and Submodules 9 to 15. Submodules 1 to 8 and Submodules 9 to 15 have all the programs and sequences of each single module of the line. Each module has all the functions that make the movements possible and the variables of the sensors and actuators. The Module Line Supervisor has the sequences to be activated in order to start the secondary actions.

The control of the de-manufacturing line has been developed using the Sequential Functional Chart (SFC) language. SFC is a graphical programming language oriented to systems where the variations of the state belong to the particular situations.

1) Development of a target in ISaGRAF

To develop the communication among the machines and the transport line system it is necessary to build and fix some details. The data must be integer and real to read the sensors and write in the variables of the actuators.

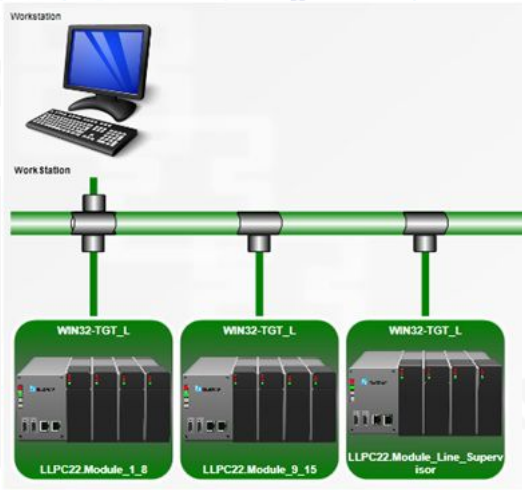

Figure 4. Deployment View of the ISaGRAF Project.

The Target Definition Builder allows the integrator to build the description of an add-on for ISaGRAF [4]. It describes: C functions, function blocks, I/O devices, complex data types, network drivers and targets. It is necessary to build a target, in order to implement the functions with their characteristics. The functions and the required parameters are saved in the target. In fact, using a target allows the portability between operating systems and program's versions, e.g. during the development of the functions a new ISaGRAF version was released and it was very useful to have a target in order to save time and avoid coming errors. Using the TDBuild tool, the target was created and modified. It is important to set up the main definitions such as the function blocks, the read and write functions (integer and real), the simple devices (Client FMC and Client RID) and the type of networks (ETCP, HSD and ISARSI).

2) Red and Write Functions

The functions and parameters for the functions read and write integer must be added to the file. At Table 1 is presented the parameters to establish the functions. A certain file of ISaGRAF was opened with the TDBuild and the parameters for the functions read and write were placed and implemented in the *software*.

Table 1. Function Parameters.

Parameters	Read_file_int	Write_file_int
Row_column (Input)	Read mode (0: row, 1: column)	Write mode (0: row, 1: column)
Index (Input)	Number of the file to be read (0:t token, 1: data)	File number to be write (0: token, 1: data)
First_r (Input)	Reading number (1: first read, 0:other read)	-
Open_mode (Input)	-	Write mode in the file (0: append, 1:overwrite)
Data (Input)	-	Data to be write
N_read (Local data)	Number of before readings to know which data must be read (N read: 3 indicate to read the fourth data from the file)	-
Store (Output)	Content read from the file	-

B. Communication Protocols

The functions were developed to communicate over a network a program executed from a computer (server), a PLC program and the machines. The advantage of the TCP/IP is the possibility to establish the communication among devices programmed with different technologies.

It is important to respect the communication protocol between devices, i.e., the messages must come in a prefixed order. The communication TCP/IP has three steps:

1. Connection: The server is always waiting the connection by the client (PLC). It's necessary to define a communication port and the machines (servers) must be identified by a unique IP address.
2. Data exchange: It started when the connection has been established.
3. Disconnection: The client asks the disconnection to the server or the server disconnects it.

1) Socket-Client in ISaGRAF to communicate machines

A new data type for the IP address and two new functions for the client and the server were added in TDBuild to the target developed before. Once the Target is completed, it was implemented in the main project in ISaGRAF.

There are two socket-clients to be implemented: one for the ZEVAC and one for the Robot Cell. Each one has the main parameters to establish the communication: start and stop client, the port, the functions to send and receive messages and the acknowledgment messages. The sockets in ISaGRAF are displayed in Figure 5.

For ZEVAC, SPEA and Robot cell, it was developed a program in ISaGRAF to control the communication and messages between the server and the client. Using a socket-client is possible to send and receive messages setting agreed protocols.

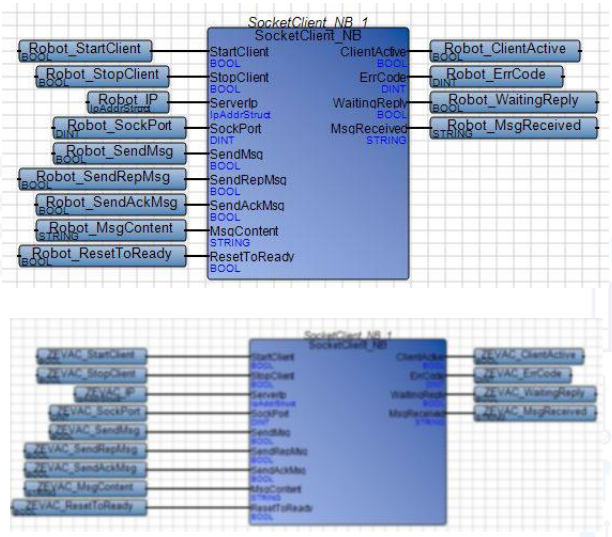

Figure 5. Sockets in ISaGRAF.

2) Communication with ZEVAC machine

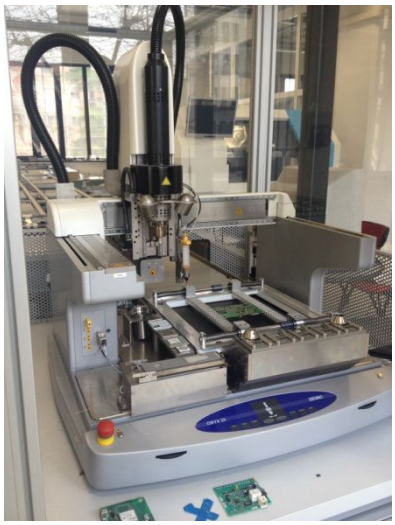

Figure 6. ZEVAC Machine.

The communication with the ZEVAC machine uses the default VisualMachines TCP commands found in the user manual of the device [7]. According to the reference manual, there are three mainly commands to establish the interchanging of data:

- Load Board: loads a board waiting at the input conveyor buffer.
- Unload Board: unloads the loaded board.
- Get State: the state response is a bit field of type unit. Bit 1 indicates a board is loaded, bit 2 indicates machine is busy.

The commands and their responses are presented in Table 2.

Table 2. ZEVAC Commands.

Commands	Response OK	Response NOK
CMD: LoadBoard	LoadBoard succeeded; Board loaded without error. LoadBoard succeeded; Board was already loaded.	LoadBoard failed; TB BoardSensor Timeout
CMD: UnloadBoard	UnloadBoard succeeded; Board unloaded. UnloadBoard succeeded; no board available.	UnloadBoard failed.
CMD: GetState	0: Free machine 1: Board Loaded 2: Busy	

It is important to keep the link while the process is running. The connection is used to send process progress information without any acknowledge of the server machine.

3) Communication with the Robot Cell

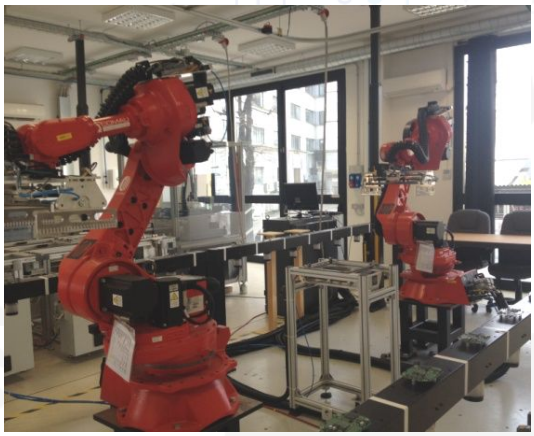

Figure 7.Robotic Arms Cell.

The TCP/IP communication with the robot cell was similar as the ZEVAC. There were set up some messages with the group of people in charge of the control and working of the cell. At Figure 8 is reported the order of the messages and commands that were agreed on.

The pallet ID (XXXX#) represents the typology of the Electronic Board loaded onto the pallet. For example, 0000# means pallet empty.

Robot Cell		Line
Request to load an electronic board on an empty pallet		
	⇐	PALLET FULL_XXXX#
STARTING_UNLOAD#	⇒	
Request to download an electronic board from a full pallet		
	⇐	PALLET_EMPTY#
STARTING_LOAD#	⇒	
	⇐	GET_PALLET_ID#
XXXX#	⇒	
	⇐	GET_STATUS#
OK# / ERROR#	⇒	
	⇐	WAIT#
OK# / KO#	⇒	
	⇐	RESTART#

Figure 8. Protocol between the Robot Cell and the Server.

4) Communication with the SPEA Machine

The communication with the SPEA is by wired electrical signals. After the board is tested in the SPEA, each board will have a type of failure given by the machine.

At table 3, it is presented the type of failure protocol decided. Depend on the failure, the SPEA will send the pallet with the PCB to the robot cell or ZEVAC or to the discharge board cell.

Figure 9. SPEA 4060 Machine.

III. HMI FOR THE SYSTEM WITH MOVICON

Movicon® is very flexible and offers a complete and powerful set of features that can be deployed in embedded HMI WinCE-based systems right through to the most modern SCADA platforms based on Windows 7/8 and servers [8].

Table 3. Failure Type From SPEA.

FAILURE TYPE	DESTINATION
Failure=0	Robot Cell
Failure = 1 to 50	ZEVAC
Failure = 51 to 99	Discharge Board Cell

In order to have a better control of the system it is necessary to implement an interface in which the user has the overview of the plant. Movicon® is useful because it works for large SCADA servers and for small HMIs. In addition to these features, Movicon® is suitable with ISaGRAF and has a free version downloadable in internet.

As it shows in the Figure 10, the developed platform gives a general panel where it is possible to know the state of the variables. Additionally, there are shown some useful commands like Start Line Supervisor, Stop Line Supervisor and Find Pallet Free and the messages between the client and servers.

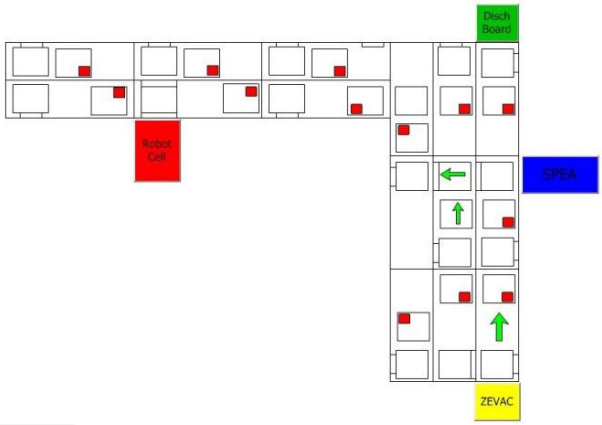

Figure 10. General view of the HMI developed in Movicon.

Similarly, it is developing an interface for a simple pallet where is feasible the access to manipulate the actuators and sensors in each module. Figure 11 displays the Module 8 with the buttons to actuate the valves and motors. The user can see the states of the sensors, actuators and the directions of the belts in order to know where the pallet is.

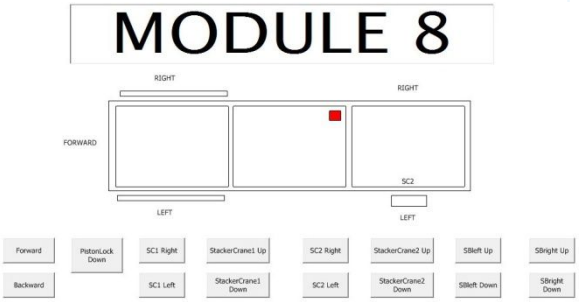

Figure 11. Interface for a single module.

IV. RESULTS AND DISCUSSION

As a result, the communication via TCP/IP was established. The protocol among servers and machines was correctly implemented and the use of the socket-client method saved time in the development of the functions.

There were some mechanical problems. Sometimes the pallet was blocked by delays in the upping and downing of the actuators in the modules. Also, there were delays because the sensors cannot read the presences of the pallets. It was possible to supervise and show real tests using the human machine interface that proved good implementation. Following the process in the interface, it can be taken preventive actions such as stop the line or taking decisions in case of obstacles in the pallet traffic.

V. CONCLUSIONS

The functions to communicate via TCP/IP the general system, the servers and the machines were correctly implemented. The synchronism in the process allowed the good working of the plant. It could be said that the TCP/IP communication is a very good way of communication but due to the short time to understand the operation of the machines, the messages among servers and machines were the basics.

There are some PLCs (*Programmable Logic Controller*) in the laboratory that could be used in order to have a faster communication. A PLC is likely programmed due to its programming language and for that reason is easier to implement different control functions. A longer work of each part of the team could help to make the integration faster. Future researches need to have enough time to organize a plan in order to test and integrate the whole system.

REFERENCES

[1] Cataldo Andrea, Brusaferri Alessandro, “Gecko Hw and Sw control architecture and functional specifications”, GECKO Technical Report, ITIA-CNR, 2013, p. 3-40.

[2] Information system resources, “Demanufacturing Whitepaper”, 2011.

[3] Rockwell Automation, “ICS Triplex AADvance”.

[4] ICS Triplex ISaGRAF Inc., “Target Definition Builder”, 2003.

[5] Copyright © 2000 SYBEX, TCP/IP, www.sybex.com.

[6] INFOTECH Automation, “User Guide”, Infotech AG.

[7] Brusaferri, Alessandro. “socketclient.c”

[8] Progea, “SCADA/HMI Platforms”, Industrial Automation *Software*, 2013.