DOI 10.2195/LJ_Ref_ten_Hompel_E_042006

Distributed Control Nodes for Material Flow System Controls on the Example of Unit Load Conveyor and Sorter Facilities

PROF. DR. M. TEN HOMPEL,
FRAUNHOFER INSTITUTE FOR MATERIAL FLOW AND LOGISTICS (IML), DORTMUND, GERMANY,

DIPL.-ING. S. LIBERT; DIPL.-ING. (FH) U. SONDHOF
CHAIR OF TRANSPORTATION AND WAREHOUSING, UNIVERSITY OF DORTMUND, DORTMUND,

The Chair of Transportation and Ware-housing at the University of Dortmund together with its industrial partner has developed and implemented a decentralized control system based on embedded technology and Internet standards. This innovative, highly flexible system uses autonomous software modules to control the flow of unit loads in real-time. The system is integrated into Chair's test facility consisting of a wide range of conveying and sorting equipment. It is built for *proof of concept* purposes and will be used for further research in the fields of decentralized automation and embedded controls. This presentation describes the implementation of this decentralized control system.

1. State of the art

Improving performance of computers and a wide spectrum of their industrial applications offer new efficient solutions in many "conservative" industrial fields. Both developers and users of control systems in manufacturing and logistics can profit from such solutions. Novel, flexible manufacturing and transport control systems arise, which can easily be reconfigured and adopted to changing functional demands [Günthner02]. The decentralized and modular control software [tenHompel04] even makes possible in the future fully autonomous logistics processes [Freitag04]. With additional tools and techniques the engineering process is accelerated and the costs for an expensive implementation are reduced. Therefore, the developing of the decentralized autonomous controls built based on low-cost but powerful computer systems, play a similar important role comparable to the introduction of the field bus systems a couple of years ago.

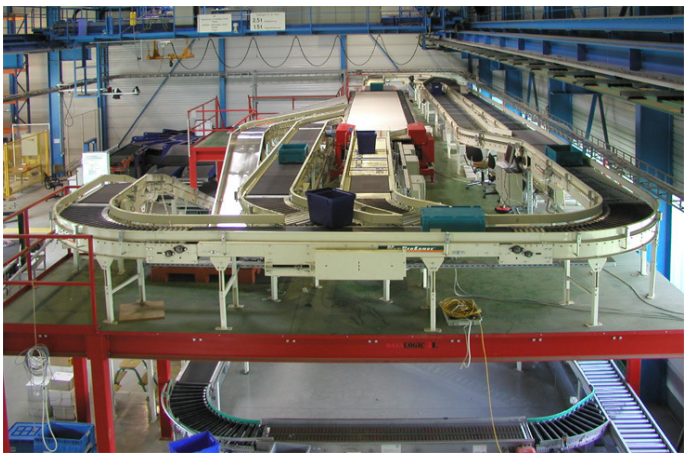

Fig. 1 Conveyor and sorter equipment

The control systems, existing in the classical industrial environment, generally consist of a central PLC [tenHompel03], [Wellenreuther98]. The PLC is usually installed in a relatively big cabinet together with fuses and relays. Sensors and actors are connected with the PLC through a single wire or a field bus component. For each sensor or actor the PLC has one input/output channel and all I/O channels are served by one CPU in the PLC, which has to be fast enough to meet real-time (approximately 20 ms) requirements for each of the several hundred I/O-pins. Such PLC is programmed in an assembler style language or possibly in a higher level programming language similar to Pascal, called IEC-1131. The program consists of one main loop, that sequentially serves all the I/Os. The PLCs are optimized to operate as fast as possible with the

cycle time in the order of microseconds. Due to their nature these control systems are relatively inflexible. Software changes require a lot of time because of the relatively high complexity of a single control, which is responsible for the whole facility. Therefore, these systems remain unchanged for a long time. To introduce changes the complete control program has to be revised.

At the Chair of Transportation and Warehousing exists a unit load conveyor and sorter facility, composed from several diverse mechanical elements (see Figure 1). In the past the facility was controlled by a central PLC. The decision was made to start a pilot project and to realize the idea of the decentralized control. The concept had to be integrated into the existing conveyor and sorter facility.

2. Objectives and the general approach

The requirement for the modularity is a central point of the current project. The idea is to combine the mechanical modules with the specialized software module to create the uniform, replaceable functional modules. This modularity concept results in a construction kit principle for the manufacturers of material flow systems. This construction kit principle is a further step forward and will make the system less complex. Moreover, in this case a part of the commissioning tasks can be fulfilled already in advance during the planning phase.

Another task that should be accomplished in the new control system is to distribute the processing performance to many decentralized controllers instead of one centralized control device. This will improve the total system performance.

The further goal of the project was development of a forward-looking, open and intelligent control concept based on the standards of the internet age such as TCP/IP, HTTP, FTP and XML. All the participants of the material flow had to be integrated into a general data communication network: the supplier at the start, the customer at the end and the material flow facility in the middle of the process.

The first step towards the realisation of the modularity principle was to break down the whole facility into several components and to classify the single functional elements (see Figure 2). The unit load conveyor and sorter facility consists of diverse mechanical segments (e.g. simple belt or roller conveyor, a buffering conveyor, a distributing conveyor). Each of these conveyor segments had to get its own control software module which controls the material flow on it.

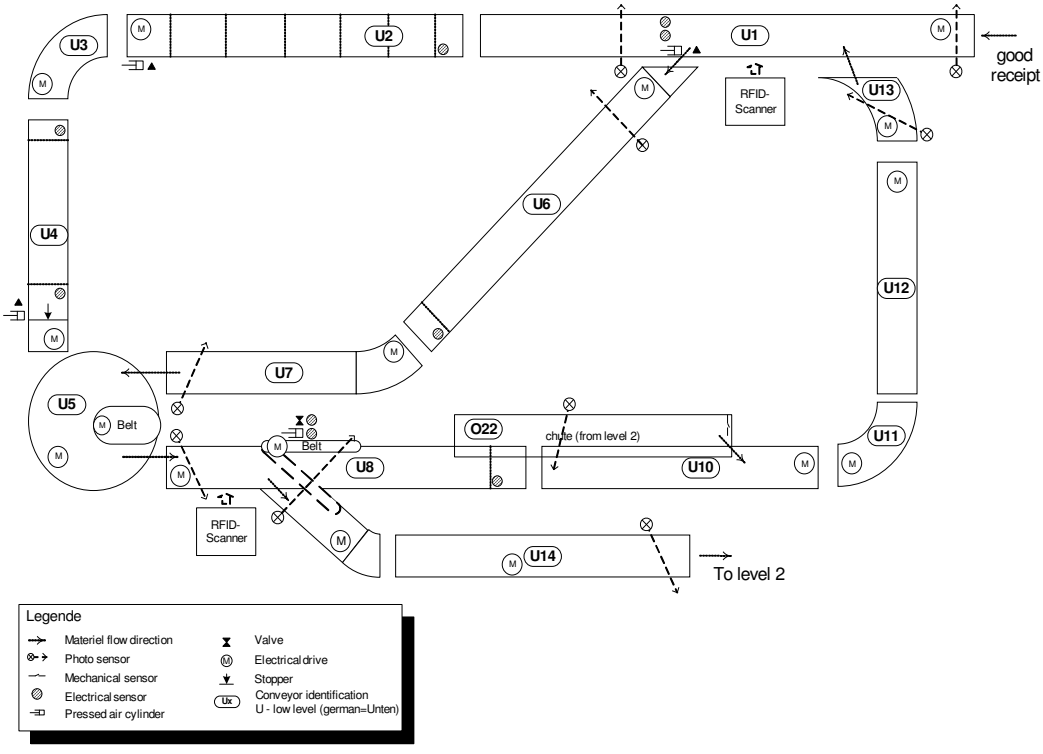

Fig. 2 Conveyor and sorter equipment layout (Level 1)

A hardware platform for running the distributed control modules are small embedded systems (industrial PCs). In this case one or more autonomously working software modules can run on the

same embedded computer. The embedded devices had to be linked together with an Ethernet cable and than installed in small cabinets close to the conveyor elements, which they control. Each embedded system, called node computer, has a small number of I/O channels which can easily be increased to the required number. The assignment of the mechanical modules together with the software modules to the node computers is shown in fig. 3.

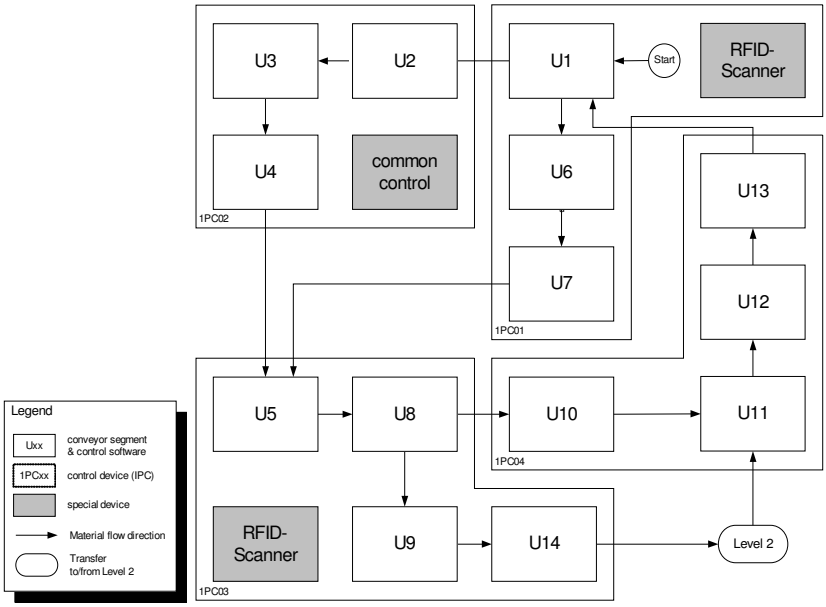

Fig. 3 Assignment of the conveying elements to the control nodes

3. Networking and platform specification

Figure 4 shows the system network structure. A total of 7 node PCs and 2 workstations are linked together. The network of the industrial computers using the Ethernet is a low-cost solution compared to the field-bus systems. The Ethernet technology allows for a high extensibility of the network, thus the communication is rather transparent and occurs through the standard TCP/IP protocol.

The workstations and the node PCs use the same Ethernet cable. This configuration is sufficient for a *proof of concept* facility. In an industrial environment, it is reasonable to separate the controller network from the workstations with a gateway. One has to keep in mind that the demands on the data transfer performance are relatively high. The node PCs act autonomously and therefore there is no need to transfer the I/O-data via the network. However, as soon as the material flow from one conveying module to another takes place, the corresponding data packets haveto be transferred as well.

Because the decentralized control modules are operating autonomously there is no need for any supervising control system. The two workstations included in the system perform the following functions (Figure 4):

1. Client for the configuration and the visualization of the facility
2. Front-end for a supplier/customer (data transfers according to EDI-standard)

The third computer represents the supplier workstation and is used for the simulation of the data communication via Internet between the supplier and the goods receipt of the conveyor and sorter facility.

A platform for running the distributed software controls has to be a compact embedded system with multiple tasking support and a real-time capability. For the current implementation the industrial PCs with a real-time supporting operating system were chosen. These industrial computers (IPCs) are compact and have a small but extendable number of I/O-plugs. They have a x86-compatible architecture and are placed in top-hat rail mountable cases. They don't have neither displays nor keyboards; the handling, the data communication and the configuration are achieved exclusively remotely over network (Ethernet).

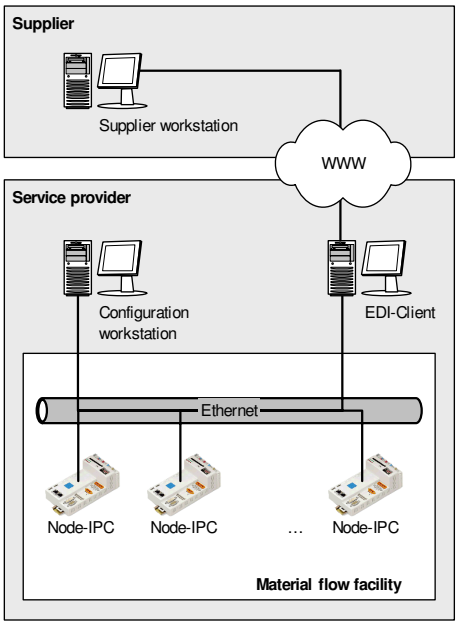

Fig. 4 Network structure

The used operating system is Linux with RTAI extension (see for example [Walter01], [Schwebel01]). The **R**eal-**T**ime **A**pplication **I**nterface is a hard real-time extension to the Linux kernel, developed in accordance with the free software guidelines. It provides the features of an industrial-grade RTOS, seamlessly accessible from the Linux environment.

4. Control software concept

A major requirement to the software design was high modularity of the decentralized control system. The best modularity and scalability can be achieved in the case when a separate control software module is responsible for each mechanical segments (e.g. conveyor or sorter). Such control software module should represent the functionality of the underlying mechanical part and ideally run on a separate node PC. The number of distributed control modules is the same as the number of mechanical segments in the conveying system. The mechanical part (conveyor) and the automation part (actors and sensor) together with the control part (embedded PC and software) form a unified functional module with two interfaces – mechanical and communicational. If these interfaces are uniform for a given system an easy exchange of singular functional modules or an extension of the system will be possible.

There is an obvious contradiction that has to be resolved in the process of developing of a highly flexible control system. On the one hand, the software modules should be uniform, independent of the mechanical and automation parts. On the other hand, there are more mechanical segments in the conveying equipment which act rather differently to each other. Therefore, the developed software consists of two important parts. Every control module (one for each conveyor segment) contains both a hardware independent task (logical part) and an I/O driver (see Figure 5). The I/O drivers depend on the underlying conveying facility. They process the sensor signals directly and give the commands to the actors. They run based on the RTAI (Real-Time Application Interface) extension of the Linux kernel and are therefore real-time compatible.

In nearly every large material flow equipment there are more than one conveying segments that have the similar functionality logic. There are several segments with the same base functionality in the described conveyor and sorter equipment as well. These conveyor types are, for example, a simple belt or roller conveyor, a buffering conveyor, a distributing conveyor, etc. Therefore, there was no need to develop completely different real-time software (I/O drivers) for every single conveyor segment. There are only about 10 different real-time units all together for more than 40 different conveyor segments in the whole equipment.

In contrast to real-time units the software architecture of the hardware independent control logic is exactly the same for all distributed control modules. The actual setting of the individual parameters is performed only at runtime. All the configuration information is stored in XML files. There are 3 configuration files all together which store the information about an I/O addressing on

I/O-plugs of IPC, network IP-addressing and the description of the equipment topology correspondingly.

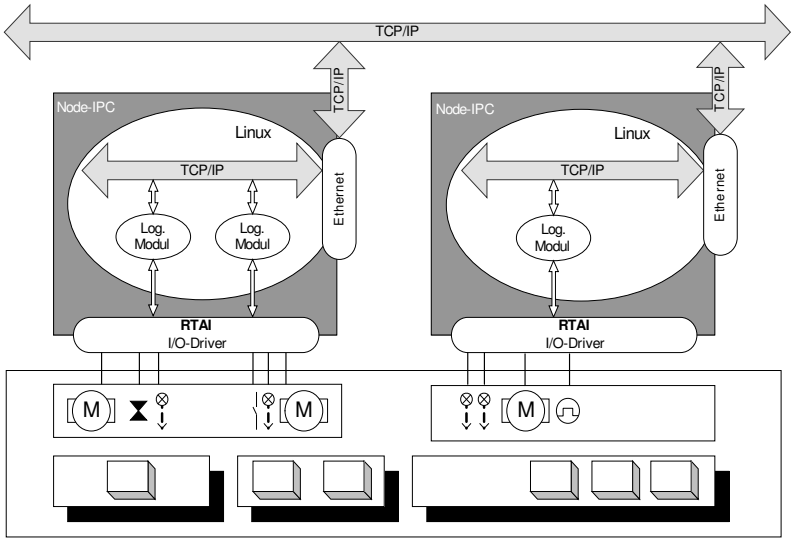

Fig. 5 Software architecture overview

The following figure shows an example of the I/O configuration using XML. The XML format is easy to read and to process both for human and machine [Shephard02]. The configuration changes which are supposed to be done by a developer or a system operator can be performed directly at runtime and do not require re-compilation of the control software. Hereby, a high configurability of the control system is achieved.

```
<IPC Name="UC1" ID="1">
  <Module Name="U1">
    <DigIO Type="INPUT" Name="BU1.1" Size="BIT" Addr="0100">
      <Description>Position at start U1</Description>
    </DigIO>
    …
    <DigIO Type="OUTPUT" Name="MU1" Size="BIT" Addr="1000">
      <Description>MU1 on</Description>
    </DigIO>
    …
  </Module>
  …
</IPC>
```

Fig. 6 Example of XML configuration file

It is possible that each control module can be run on its own embedded PC. However, because of the costs it could be better to run more software modules on the same IPC. Thus, different system architectures can be realized: from a single computer-architecture (all modules in one computer) to a fully distributed system (one computer for each module).

5. Data interchange and communication

All the control software modules work autonomously, independently of any central control system. Under such condition a fast, stable and transparent data communication is a very important factor. There are several tasks for the communication in the distributed control system to be fulfilled to guarantee for the continuous material flow:

- Providing a safe interchange of information about the status of the underlying conveyor equipment
- Providing a safe interchange of information about the packets and the jobs
- Making the status of the control modules available for the visualization at a remote workstation (e.g. customer PC or configuration PC)

A Java client as well as the HTTP-protocol for the data transfer are used for the visualisation of the status information of distributed control modules at the configuration PC. Thanks to Linux operating system installed on the node PCs the standard WEB-server can be used to deliver the control status information directly to the Internet, so the user can access it from a browser.

To interchange the process information the control modules communicate with each other using TCP/IP sockets. The form of the communication does not depend on the location of these software

and is the same both for the location on different PCs and for the location on the same PC. The telegrams which are sent from one control module to another, include equipment status requests and unit load data.

The unit load data include the packet identification number and the job information. These virtual packets in the form of telegrams are transferred from one software module to the following one at the same time as the real unit loads change to another conveyor line. Hereby, each control module manages the packets that are actually transported by the given conveyor. Thus, the real unit loads are simulated by the virtual ones and the information flow reproduce the material one.

The RFID-technology is very important to guarantee for a controllable and meaningful material flow through the conveyor system. It is applied both for the identification of unit loads (e.g. error correction) and for the dynamical routing of the packets towards one or more destination points. The job information for every packet contains a list of points on the equipment including the goods issue which must be visited by the packet. This information together with the packet identification number are stored in the RFID-tag on each packet and can be read and rewritten at four RFID-scanner stations.

6. Packet routing

There are two conditions which should be satisfied to route packets through the conveyor and sorter equipment to a defined destination point (e.g. good issue). On the one hand, the job information is required. On the other hand, the control module must have the exact knowledge about the topology of the equipment.

The job information includes the packet identification and the destination specification that was already mentioned above. The topology of the existing material flow system is known and can be mapped out. A formal description of the system topology is carried out using the XML format. The XML format is very suitable for data communication in the network and can be processed by the control program module.

The formal topology definition in XML has some special features. It contains the paths in the facility which the packets can follow. Each path consists of one or more lanes that correspond to one ore more consecutive conveyor lines in the facility. Each path has a start and a destination point. These can be conveyor lines or goods issues. Not all possible paths are described in this data structure but only the relevant ones. These are, for example, the alternative ways from a branching point (switch, sorter) towards some defined point (any conveyor line or goods issue) in the facility (see Figure 7).

Sometimes, it makes sense to define the special destination points for the special cases. For example, if there are several alternative ways to reach a certain place in the facility, a *default* destination point can be defined. This means that every alternative path can be used to transfer a packet to the given target.

Because there is no central control system the distributed control modules have to make the decision about the packet routing independently. For this purpose there are some functions, which have to be carried out by the control software:

- Every control module administrates its own path table
- Every control module routes the packets to their target using the path table
- Every control module routes the packets using the optimal path if several alternative paths are available
- Every control module gets the actual version of the path table from the configuration file at system start

It is not sufficient to know, where a packet should be routed to if there are more than one alternative way to reach the target. In such cases the decision about the path choice has to be made. Such decision is made based on the allocation of priorities for each path from the path table. The allocation of the path priority can be either static or dynamic. In the case of dynamic priority allocation the actual information about the loading state of all conveyor lines from this path is needed.

It requires additional data communication complexity and is not implemented in the current control version. For the allocation of static priorities the expert knowledge or experiences are used. So the path that contains conveyors with a higher loading capacity gets the higher priority value. Another advantage of the assignment of path priority is that the failed parts of the facility (for example due to a failure) can easily be excluded form the transport system. Setting a priority to zero for a particular path means that the conveyors of this path will not be used for transporting the unit loads at all and can be repaired or replaced.

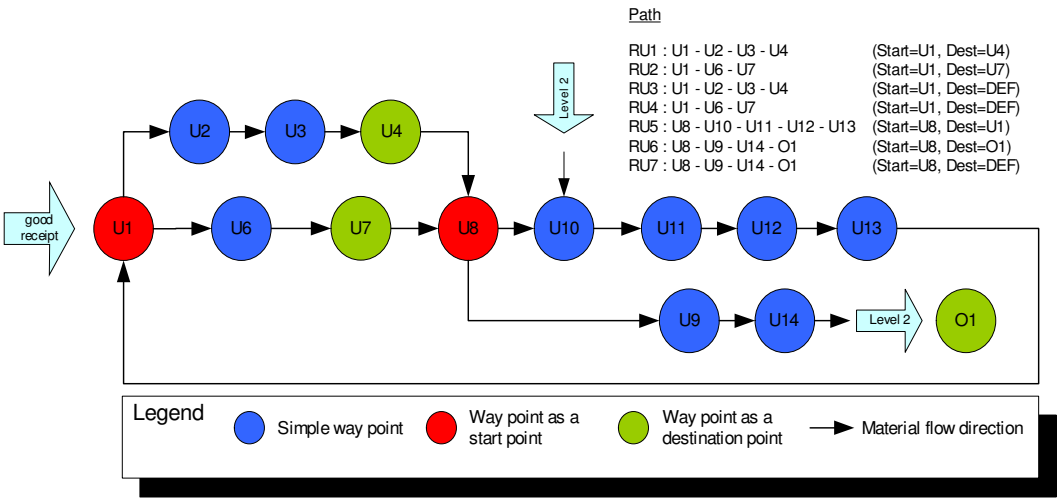

Fig. 7 Example of the equipment topology description (Level 1)

7. Conclusion

The attempts to replace a classic central control system by intelligent, autonomously operated, control modules are just another step to creation of modular and cost-effective material flow systems. The decentralized control system developed at the Chair of Transportation and Warehousing at the University of Dortmund demonstrates such an attempt.

The advantages of a decentralized control system are obvious: Its high extensibility and ability of reconfiguration allows for satisfying real life requirements quickly and cost-effectively. The uniform modular control software also decreases development and implementation costs. The installation complexity is reduced owing to a reduced cabling as well as due to the Ethernet networking. At the same time, this cost-saving solution increases the operational availability of the whole system.

The use of modern IT-standards such as XML format for system configuration, the Internet protocols (TCP/IP) for data communication and the RFID-technology for unit load identification makes the discussed control system a unique testing area for further research in the fields of decentralized automation and embedded controls.

Literature

[Freitag04] Freitag, M.; Herzog, O.; Scholz-Reiter, B.: Selbststeuerung logistischer Prozesse – Ein Paradigmenwechsel und seine Grenzen. In: Industrie Management 20 (2004) 1, p. 23-27.

[Günthner02] Günthner, W.A.; Wilke M.: Anforderungen an automatisierte Materialflusssysteme für wandelbare Logistikstrukturen. In: Tagungsband Wissenschaftssymposium Logistik der BVL 2002, Huss-Verlag GmbH, München, 2002, p. 335-345.

[Schwebel01] Schwebel, R.: Embedded Linux. mitp-Verlag, 2001.

[Shephard02] Shephard, D.: XML. Markt+Technik Verlag, 2002, p. 301-326.

[tenHompel03] ten Hompel, M.; Schmidt, T.: Warehouse Management. Springer, 2003, p. 161-165.

[tenHompel04] ten Hompel, M.; Stuer, P.: Echtzeitfähigkeit gefordert. In: Fördertechnik 7-8/2004, p. 52-54.

[Walterl01] Walter, K.-D.: Messen, Steuern, Regeln mit Linux. Franzis Verlag, 2001.

[Wellenreuther98] Wellenreuther, G.; Zastrow, D.: Steuerungstechnik mit SPS. Vieweg Verlagsgesellschaft, 1998.