Sborník vědeckých prací Vysoké školy báňské - Technické univerzity Ostrava
číslo 2, rok 2006, ročník LII, řada strojní
článek č. 1531

Grzegorz HAYDUK[*], Zbigniew MIKOŚ[*], Grzegorz WRÓBEL[*], Paweł KWASNOWSKI[**], Marcin JACHIMSKI[*], Henryk ZYGMUNT[***]

RESULTS OF EXPERIMENTAL VERIFICATION OF MIXED PROTOCOL FOR PROCESS DATA TRANSMISSION

VÝSLEDKY EXPERIMENTÁLNÍHO OVĚŘOVÁNÍ SLOUČENÉHO PROTOKOLU PŘI PROCESU PŘENOSU DAT

Abstract

The paper presents mixed protocol (periodic and event triggered) for transmission of process variables. It is well suited for application in PLCs and supervisory systems. The protocol specifies that the variables are transmitted (one at a time) when the variable triggers an event and logical connection (between conversating nodes) is active. The connection is activated (or reactivated) periodically (with period T) and is valid for the period T. The activation of connection for given variable also transmits the variable value, thus the period T also specifies variable refresh period. Those rules allows to periodically check communication status and fast (event triggered) transmission of current process variables values. This is far more effective in terms of throughput than periodic-only communication. Next, results of experimental research of the protocol are presented. The research was done both in LAN and Internet and was focused on delay of process variables transmission (event triggered).

Abstrakt

Příspěvek prezentuje sloučený protokol (periodický a událostní) pro přenos proměnných, vhodných pro aplikace v PLC a dohlížecí systémy. Protokol specifikuje proměnné, které jsou jednou přeneseny, když je aktivní měnící se spuštěná událost a logická návaznost mezi komunikujícími prvky. Dále jsou prezentovány výsledky experimentálního výzkumu protokolu. Výzkum byl proveden v LAN i Internet prostředí a zaměřil se na zpoždění procesu při přenosu proměnných.

1 INTRODUCTION

Modern field networks use various communication standards implementing many different protocols. While the author was participating in numerous industrial application projects which involved communication between PLCs themselves and between PLCs and supervisory systems, it came out that each of those communication standards has disadvantages or weaknesses.

As a result of those experiences it appeared that there's need for protocol which would be easy to implement and efficiently using available bandwidth, thus introducing transmission delays as small

[*] Ph.D., Faculty of Electrical Engineering, Automatics, Computer Science and Electronics, AGH - University of Science and Technology, al. Mickiewicza 30, Kraków, Poland, tel. +48 (12) 617-28-83, e-mail [hayduk,mikos,wrobel,jachim]@kaniup.agh.edu.pl
[**] M.Sc., Faculty of Electrical Engineering, Automatics, Computer Science and Electronics, AGH - University of Science and Technology, al. Mickiewicza 30, Kraków, Poland, tel. +48 (12) 617-28-83, e-mail kwasn@kaniup.agh.edu.pl
[***] Assoc. Prof., Faculty of Management, Computer Science and Social Sciences, WSB – Academy of Business, ul. Cieplaka 1c, 41-300 Dąbrowa Górnicza, Poland, tel. +48 (601) 54-56-79, e-mail hazet@agh.edu.pl

as possible. The protocol should be optimised for PLC needs: transmitting analog (integer and float) and discrete process variables. The protocol shouldn't be general purpose (allowing transmitting buffer of data of any desired length), instead it should efficiently transmit process variables.

2 MIXED PROTOCOL

To satisfy those needs, there has been specified and implemented mixed protocol, where transmission is both periodic and triggered by events. The protocol is loosely based on Producer-Consumer model of data exchange, because there are nodes referred to as producers and consumers but transmission is not based on broadcasts and source addressing. The protocol can be located in upper layers of ISO/OSI reference model, because it doesn't specifies physical layer nor medium access control layer, and it has no mechanisms of retransmission. Instead it depends on the network selected by the developer and adds layer which allow sending and receiving process variables values.

2.1. Specification

Producers are nodes which define process variables (i.e. inputs/outputs, coefficients, alarms) and consumers are nodes which want to receive those variables. The node can act both: as a producer and as a consumer. The producer node knows nothing about specific consumers – it's the consumer who knows which producer have given process value and creates logical connection to the producer. Consumer can create many logical connections to one or many producers.

The protocol specifies that the producer transmits variable value when the value changes (triggering event) to all consumers which made logical connection for this variable. The connection has to be activated and then periodically (with period T) reactivated by the consumer. The connection is valid for the period T and after that period it expires. Acknowledgement of the connection activation for given variable also transmits the variable value, thus the period T also specifies also variable refresh period.

To satisfy above rules, the protocol specifies several services (data frame types) for producer and for consumer. Consumer frame types are:

- registration of variable (which activates connection) – "Reg"
- setting new value of producer variable (meant to be used i.e. for setpoints) – "Set"

On the other hand, producer can transmit following frames:

- acknowledgement of registration – "RegAck"
- unacknowledgement of registration – "RegUnack"
- update of variable value (the variable has new value) – "Upd"

It is the assumption that the periodic transmission (establishing logical connection with "Reg" and "RegAck" frame) occurs with period T which results only from requirement of refreshing communication status and is not used as variable value update or refresh service, because event triggered transmission does the updates. Nevertheless, "RegAck" frame also holds current value of variable, therefore there's no need of "Upd" frame immediatelly after "RegAck" frame.

As it is straightforward to imagine events generated by discrete or Boolean variables, but it's more complex with analogue variables. The events generated by analogue value changes are defined using threshold, with reference value equal to last value sent to the consumer. If the difference between current value and reference value is greater than threshold, then it's said that analogue variable triggers an event which is sent to consumer.

2.2. Event avalanche

However, communication system following only above rules could collapse and get blocked in case of event avalanche. Therefore nodes implementing the mixed protocol must also implement event anti-avalanche mechanism, which defines anti-avalanche time T_{AA} that specifies minimum time between consecutive transmissions of given variable (regardless of variable type). The T_{AA} time is defined separately for each (variable, consumer) pair and can be different (it depends on how fast events really have to be sent over the network). Thus even if events will be triggered faster than T_{AA}, they will be transmitted (and use available bandwidth) with this minimum period T_{AA}. The T_{AA} also specifies minimum time which has to elapse between acknowledgement of variable registration and first event triggered variable update.

2.3. Protocol summary

Above rules allow for periodic check for communication status and fast (event triggered) transmission of current process variables values. This is more efficient in terms of throughput demand than periodic-only communication (i.e. master-slave or request-reply). But it's important that connection reactivation period should be adjusted for proper communication status validation and use as low bandwidth as possible.

The protocol, including frames format is described in detail in reference [1].

3 APPLICATIONS: GATEWAY NODES

Described protocol can be successfully implemented not only between controllers themselves or between controllers and SCADA systems, but also to implement variables concentrator node and variables distributor node. They can intermediate between control network and remote SCADA system or remote service (i.e. for diagnostic purposes).

Such special nodes can efficiently work as a gateway connecting networks which use different types of medium access control or act as a gateway between fieldbus network and remote system which is using modem or other slow connection.

Variables concentrator acts as a consumer on controllers side and as a producer on the remote side. It collects process variables and passes them to remote system.

On the other hand, variables distributor receives setpoint variables from remote system (SCADA, panel, webserver), stores them locally and acts as a producer to nodes which need those setpoint variables.

4 THE PROTOCOL VERIFICATION

Certain practical tests were made to verify the mixed protocol especially due to bandwidth demand and transmission delays, compare it with other protocols, but also to prove correctness of implementation. Programs representing producer and consumer were written in C++ and the mixed protocol was implemented as a layer based on TCP/IP. Periodic protocol was implemented the same way, but was periodically transmitting all variables, using all available bandwidth.

To measure the delays, transmitted frames with process variables values also were including event time. Based on this it was possible to tell (on consumer side) what is the time delay introduced by all the layers between producer and consumer.

Another issue was to synchronise realtime clocks on communicating nodes precisely enough (in terms of single milisecond). It was accomplished by rough synchronisation using NTP servers, followed by precise synchronisation with consumer triggered bidirectional communication and measurements of average delay in that communication. Then, producer was notified of the average delay and synchronising its realtime clock by half of the average delay. Without surprise, the measurements were consistent with results given by ping tool.

The event delays were collected by consumers and then saved to a file, which was used to draw charts of delay distribution. The chart (i.e. shown on fig. 1) shows how many variables were transmitted with delay which fits to delay [t; t + 10ms).

The tests were run on nodes working in the same LAN and on nodes connected with heavily loaded Ethernet connection.

The first case (LAN) is not quite interesting, because 100Mbps LAN gives too much throughput to impose really heavy load with event triggered variables – their number had to be so big that it wouldn't have reference in practice.

Much more interesting is the second case, where 100Mbps internet connection used between nodes was used by more than ten thousand users (university campus) and producer node was connected to university network with 10Mbps hub and additionally producer node was running building automation server, performing periodic (T=10s) communicating with IP/LonTalk router. Transfer speed between consumer and producer was at 20KBps level.

Length of mixed (and periodic) protocol frame was 20 bytes, producer node had 800 variables, 500 of them was randomly changed 2 times per second (so there was 1000 events per second).

The results of tests in form of delay distribution were shown on fig. 1. The values correspond to 10 seconds of communication of mixed and periodic protocol. It shows that in case of mixed protocol, most events (variable transmissions) – about 2700 – had delay in range [10ms; 20ms). As the delays are longer, the number of variables that corresponds to them decreases and delay longer than 70ms was introduced to very small percentage of variables.

In case of periodic protocol, delay distribution was steady, there was no relation between the variable event and its transmission. Transmission of all 800 variables took 0.8 seconds and comparable number of variables had delays 10ms and 800ms (the chart in Fig. 1 shows only delays up to 250 ms).

5 CONCLUSIONS

Results showed here show that mixed protocol introduces considerably shorter delays in transmission of variables than periodic-only protocol. It uses available bandwidth much more efficiently; transmission is taking place only when it carries new information. Despite of event–triggered variable transmission, it offers periodic communication status checking, which doesn't create meaningful transmission load and doesn't increase the delays.

As a next step, the creation of software libraries for various operating systems is considered, which will allow easy protocol implementation in embedded controllers and will allow using the mixed protocol between controllers and supervisory system.

It should be pointed out, that the difference between described protocols is increasing as the number of variables increase. It depends also on process which generates events (new values of variables) – the network load is proportional to rate of the events. In particular, mixed protocol gives very short delays when used for variables like alarms, state of operation or failure; their nature makes that they require immediate transmission and they don't change very frequently, thus doesn't produce heavy load.

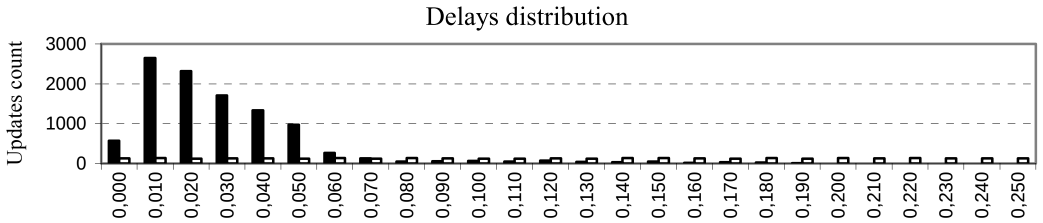

Fig. 1 Transmission delays distribution for mixed and periodic protocols

REFERENCES

[1] HAYDUK, G. Time delays in distributed realtime systems in industry and building automation, *PhD thesis*, University of Science and Technology – AGH, 2005, Krakow, Poland.

Reviewer: doc. Ing. Radim Farana, CSc., VŠB-Technical University of Ostrava