Component based Colored Petri Net model for Ethernet based Networked Control Systems

Abouelabbas Ghanaim and Georg Frey
University of Kaiserslautern
Department of Electrical and Computer Engineering
Erwin-Schrödinger-Str. 12
D-67653, Kaiserslautern, Germany
{ghanaim,frey}@eit.uni-kl.de

Abstract

This paper presents a novel colored Petri net model for the simulation of fully-switched Ethernet based networked control systems (NCS). The model of the NCS is built from models of the automation system components (e.g. PLC, Ethernet Switch, IO Module) in a structure conserving way. CPNtools software is used to build and simulate the models. The proposed approach is used to study the overall response time of a given NCS as well as the two delays between actuator and controller and between sensor and controller respectively. These delays are often used in the design of controllers for NCS. An analysis of the results using periodograms shows a high amount of regularity in the delays that possibly can be exploited for controller design.

1. Introduction

The trend of increasing decentralization in automation systems by means of new networked structures leads to the Networked Control System NCS (Figure 1). By networked and decentralized structures, a variety of non deterministic delays are introduced into NCS. Those aspects have direct influences on stability and performance issues of controlled processes [1], [5], [6]. Therefore it is necessary to determine those delays when designing a controller.

![Figure 1. Networked control system.](image)

Well known approaches for the determination of delay times are measurement, static analysis, discrete and continuous simulation, and formal verification. This paper uses Hierarchical Timed Colored Petri Nets (HCTPN) to model NCS in a component based structure conserving way, i.e. each component of the NCS is modeled as a HCTPN and the overall model is built by interconnection of the component models representing the structure of the overall NCS.

The software CPNtools from the University of Aarhus [2] with existing simulation and monitoring features is used to implement and simulate the models. The aim of the simulation analysis are the controller-to-actuator-delay $\tau_{ca}$ or forward network delay and the sensor-to-controller-delay $\tau_{sc}$ or backward network delay.

This paper is organized as follows: The next section describes the modeling approach in more detail. Section 3 presents and discusses some simulation results. In the concluding section the results are summarized and an outlook on future work is given.

2. Model description

There are many modeling and simulation schemes for networked automation systems. Two recent approaches for Ethernet based systems are based on the Dymola simulation environment and on probabilistic Model checking respectively [1]. Another important category of models is based on colored Petri nets for the simulation of discrete event systems with object oriented features [4].

In the current work the discrete event approach based on colored Petri nets is used, because it provides high modeling power combined with fast simulation speed. The main difference to the approach in [4] is that the presented model is structure conserving in the sense that for each component in the modeled system there is a component in the model whereas in [4] the use of object oriented features is more prominent leading to a representation of components in tokens of...
the CPN model. The presented model has the following features:

- Tokens are used to model the data flow in frames. The tokens’ data structure represents the data as defined in the Modbus TCP/IP protocol [3].
- To realize the component based structure, a separate page (template) is designed for each automation component (PLC, Ethernet switch, IO module). The whole automation system can then be constructed by cloning the component pages to the system page.
- For the network fully-switched Ethernet is assumed.
- The communication is based on the Client/Server-model with cyclic I/O scanning.
- A process page is used to simulate the process events and can be used for open and closed loop time delay simulation.

Figure 2 shows an example of a system composed of one PLC connected to a switch. To this switch also two IO modules are connected. The IO modules are further connected to the process under control. The system works – in short – as follows: Periodically the PLC sends a Modbus message via the switch to each IO module. This message contains new actuator values and at the same time serves as request for new sensor values. After reception of such a message, an IO module writes the new actuator values to the process and sends current sensor values back to the PLC.

The model is composed of three main components. The first is the PLC (page) which is divided into two sections: the first one models the PLC scan cycle (Read-Execute-Write) with IO buffer update whereas the second section is concerned with the client mechanism of input/output cyclical scanning over the network. The second component is the Ethernet store-and-forward switch with switch table. The third component is the IO module with IO buffer and cyclical update section, as well as a section for the server mechanism to answer Modbus requests.

Figure 3 shows a more detailed view of the IO module. In this figure appear the two sections of the IO module. First section deals with reading of one integer input value coming from the sensor and writing of one integer output value going to the actuator. The read transition is started every 2 ms (in this case) which specifies the scan time of the IO module. The shift of the write transition is defined by a delay of 1 ms from the start of the scan cycle.

The second section deals with the incoming and outgoing frames through the usage of rec and send transitions. The two transitions use avail token to simulate the availability of the channel. Transition delays are added to simulate the time needed to build the outgoing Modbus TCP/IP frames, to decode the incoming frame, and to update the IO buffer through the backplane.

Figure 4 shows a CPN place of type packet with a network frame.

The Modbus TCP/IP frame structure is represented as a record of fields containing frame number, command (Read/Write), source address, destination address, input data, output data, and acknowledgement (Y/N). Figure 4 shows a CPN place of type packet with a network frame.
3. Simulation and results

The CPN model is used to simulate the case study presented in [1] to verify the model by comparing it with existing results. The simulation uses one PLC with a scan cycle of 10 ms, a cycle for input/output scanning of 17 ms of the Modbus client in the PLC, one Ethernet Switch, and one IO module with 2 ms scan time. The response time from sensor to actuator through the controller is derived for a large number of uniformly random input signal times as in [1]. Figure 5 shows the automation system layout for the response time, as well as the forward and backward delay determination. The result shown in (Figure 6) is comparable to the results in [1], [4] and takes a very short simulation time compared to the simulation approach presented there using a hybrid approach in the Dymola software.

After verification of the model relative to previous models, the analysis of the two network delays $\tau_{ca}$, $\tau_{sc}$ is considered. In this simulation, a continuously increasing signal with fast sampling time of 1 ms is used as sensor value. The PLC (controller) program is a unity function i.e. the output is equal to the input. The delay time from sensor to the read buffer of the PLC is calculated as $\tau_{sc}$, and the time from reading the value in the PLC input buffer (Read state of the PLC cycle) to affecting the actuator is calculated as $\tau_{ca}$.

Figure 7 shows the Input (Sensor) /Read (PLC buffer) /output (Actuator) signals in the experiment. It can be seen from the figure that the read and output signals are approximately first order sampled signals of the input signal.

From the simulation results, the forward network delay and the backward network delay are calculated and analyzed. Figure 8 shows the two delays with the total response time for parameters: PLC scan of 10 ms, IO scanning of 17 ms and IO module scan of 2 ms. The periodicity of the tree delays is clearly visible in the figure.

![Figure 5. Experiment layout: PLC with 10 ms scan cycle and 17 ms Modbus client scan, Ethernet Switch, and IO module with 2 ms scan.](image1)

![Figure 6. Probability density function for response time, PLC scan time 10 ms, IO scanning 17 ms, IO module scan 2 ms. Results derived for 500000 samples (sample every 340ms time slot) in 1118 s (PC with P4, 3 GHz, 1 GB RAM).](image2)

![Figure 7. Input/Read/output Signals.](image3)

![Figure 8. $\tau_{sc}$ (bottommost curve), $\tau_{ca}$ (middle curve), and response time (topmost curve). Parameters: PLC scan of 10 ms, IO scanning of 17 ms, and IO module scan of 2 ms.](image4)
Periodograms are used to measure the exact periods of the delays. Figure 9, Figure 10, and Figure 11 show the periodograms for the delays. These figures draw the maximum power as a function of the delay sample number and they can be used to calculate the periods of the delays as: 8.27 sample, 7.91 sample, 7.91 sample for $\tau_{sc}$, $\tau_{ca}$ and response time respectively. Hence, the period of each signal can be exactly determined.

The same experiment was then performed for a variety of scan cycles and skew times between these scans. The results indicate a kind of periodicity in each of the simulated cases.

In many experiments the results show a great correlation between the forward and the backward network delays with anti phase time shift which results in constant response time. The constant response time itself is a function of scan cycle times and skew times among the scans.

4. Conclusions and outlook

The paper presents a new model for networked control systems which is based on the CPNTools for modeling discrete event system. A comparative study shows that the model can be used for simulation of networked control systems. Results of periodicity measurements for the forward and backward network delay are presented. The model can be modified to include real program processing in the controller and a simple (timed but discrete) process model can also be included to close the loop and study the control performance.

Further work will concentrate on the analysis of the periodicity of the two delays and their interdependence. It is assumed that this will lead to an abstracted model to be used in analytical controller design

References