Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 109C (2017) 249–256

Procedia Computer Science

www.elsevier.com/locate/procedia

The 8th International Conference on Ambient Systems, Networks and Technologies (ANT 2017)

Planning for Heterogeneous IoT with Time Guaranties

José Cecílio[a,b,*], Pedro Martins[a,b], Pedro Furtado[a]

[a]*University of Coimbra, Coimbra, Portugal*
[b]*Polytechnic Institute of Viseu, Viseu, Portugal*

Abstract

Distributed IoT Systems with time guaranties need to be carefully dimensioned to provide correct operations timings. In heterogeneous systems, with wired and wireless components, there are many factors that have implications regarding timings. Beside of developing real-time operating systems, IoT platforms for real-time communications need to be carefully dimensioned to provide correct operation timings. In this paper we propose a planning algorithm that allows building a heterogeneous IoT system to operate with timing constrains.
Current solutions to plan and predict latencies in distributed IoT systems are currently mostly based on a set of rules-of-thumb. Since time division multiple access (TDMA), token-ring and Carrier Sense Multiple Access (CSMA) based medium access control protocols can coexist in the same network, new approaches are needed to work with any of these protocols and heterogeneity. We consider protocols used in industry-strength solutions and propose a latency model, which can be used by the engineering teams to guarantee strict timing compliance in their heterogeneous plans.

Peer-review under responsibility of the Conference Program Chairs.

Keywords: Heterogeneity, Model, IoT Systems, Time Guarantees.

1. Introduction

In an IoT context, a communication network is used to connect sensors and actuators which play a significant role in distributed systems. Typically, sensors, and actuators may be connected with a real-time control network. This

* Corresponding author. Tel.: +351 239 790 000.
E-mail address: jcecilio@dei.uc.pt

Peer-review under responsibility of the Conference Program Chairs.
10.1016/j.procs.2017.05.347

network should be able to transmit real-time periodic data, sporadic data for alarms and non real-time message data for network maintenance. Existing real-time communications are mainly based on wired technology.

However, with advancements in wireless technology, wireless devices can be integrated into existing wired systems, making it a heterogeneous system, where different protocols, mediums and software parts may coexist in the same network.

The future of communication resides in the Internet of Things, which is a growing technology. Recently, a lot of efforts have been made to design operating systems for IoT devices because neither traditional Windows/Unix, nor the existing real-time operating systems are able to meet the demands of heterogeneous IoT applications. Beside of developing real-time operating systems, IoT platforms for real-time communications need to be carefully dimensioned to provide correct operation timings. In this paper, we propose a planning algorithm that allows building a heterogeneous IoT system to operate with timing constraints.

The rest of this paper is organized as follows. Section 2, describes a typical IoT system with time constraints, its operations and requirements. In section 3 a planning algorithm for heterogeneous IoT systems is proposed. The algorithm is composed by a latency model and a tool that based on user requirements, dimensions the distributed IoT system to meet those requirements. Then, in section 4 the proposed algorithm is evaluated. Lastly, in section 5 we present some conclusion related with the defined model and how it can help with heterogeneous distributed IoT systems.

2. Distributed IoT Architecture for real-time and Operations

Distributed IoT systems use equipment spread over a plant (monitoring and control space) and interconnected by a communications network. Standards such as Ethernet, IPv4 or IPv6 are used to transmit data. However those standards, by itself, do not provide precise timing[1,2]. Token-ring approaches such as Fieldbus[3], define how to transmit data and commands, which are used within monitor and control loop operations, under precise timing requirements.

2.1. Distributed IoT System - Organization

A heterogeneous distributed IoT system is typically made of various sub-systems. It includes resource-constrained nodes and more powerful nodes such as embedded computer or normal computer nodes. We are assuming a distributed IoT system that includes wireless nodes organized into wireless sub-networks.

Each device in the system has a unique identifier (device ID), which includes a serial number unique to the device.

The Ethernet is the common communication protocol used in those systems. It uses 10, 100 Mbps or 1 Gbps as the physical layer and provides a high-speed backbone for the network. However, typically, IoT sensors and actuators do not support that speed due to its limited capacities and a token-ring approach to control communication with devices is needed.

The token-ring approach prevents the collision of data between two nodes that want to send messages at the same time, using a special three-byte frame called a token that travels around the network (ring). Token-possession grants the possessor permission to transmit on the medium. Token ring frames travel completely around the loop[6].

In this approach three types of devices should be considered: link masters, basic devices, and bridges[7,8].

A link master device is capable of controlling the communications traffic on a link by scheduling the communication on the network.

A basic device is a device which is not capable of scheduling communication. It is a field device that, typically, connects sensors and actuators on the field.

Bridge devices connect links together into a spanning tree. It is a device connected to multiple links whose data link layer performs forwarding and republishing between and among the links.

Concerning heterogeneous distributed IoT systems with wireless and wired nodes, wireless nodes are connected to a Bridges and may run a TDMA protocol, which delivers a high degree of time predictability.

In this context and in order to provide some guaranties to the network, the TDMA protocol must create and

follow a schedule or time frame for all the network activity: each node is assigned at least one slot in a time frame. The time frame is considered to be the number of slots required to get a packet from each source to the bright node. It is also called the epoch size (E).

Typically, nodes will send one message in their slot per epoch, which requires them to wait until the next epoch to send another message. If a very large network was considered, the time to visit all nodes would be high and operation latency would be too high. To avoid this problem, network sizing is necessary, resulting in multiple smaller network partitions.

Generally, wireless devices can be placed as they would have been placed with a wired installation. But several conditions must be considered and could result in a relocation of the wireless devices (or at least the antenna). Typically, the gateway (bridge) is placed first, since this is the core element of a network. There are three basic options for placing a gateway:

- Where it is easiest to integrate with the distributed system or plant network;
- Central to the planned network. Placing the gateway in the center of the network provides the best position for most of the devices to have a direct communication link with the gateway;
- Central to the process unit. This placement provides the most flexibility for future expansion of the initially-planned network to other areas within the process unit.

It is desirable to have at least 25 % of the wireless devices with a direct communication path to the gateway[4,5]. This ensures an ample number of data communication paths for the devices farther away.

To provide high-degree of timing guarantees, an operation schedule (cycle time) must be developed to avoid unwanted delays and latencies, which influences the overall system responsiveness. To develop the correct schedule, a static topology must be assumed, and the network must be sized according to timing requirements.

Token-ring based-protocols can be applied to both wired and wireless networks. However a latency model and explicit operation times planning (operations scheduling) based on timing constraints is needed.

2.2. Operations and Requirements

High level operations, such as configuration, actuation, monitoring and closed-loop control, can be defined over those heterogeneous distributed IoT systems:

- Configuration – sending commands to a node to reprogramming/configure it;
- Actuation – sending commands to a node to actuate over the environment;
- Monitoring – sense and send data measures to a control station, where they will be processed/delivered to users. The monitoring operation can be periodic or event-based. If the operation executes with a specific period, it is periodic, but if the operation executes only when an external event occurs, it is event-based. Periodic monitoring is based on a configured sampling rate;
- The closed-loop control operation corresponds to sensing and sending data measures to supervision control logic, processing them, determining an actuation command to send to an actuator, sending the actuation command and actuating. Similar to monitoring, supervision control logic can react based on events (asynchronous control) or with a pre-defined periodicity (synchronous control). These are defined as:
    - Asynchronous or event-based control can be defined as: “upon reception of a data message, the supervision control logic computes a command and sends it to an actuator”.

- Synchronous or periodic control can be defined as a periodic control, where the periodicity of computations is defined by users. The supervision control logic runs in specific instants (period).

Each of those operations can be associated with timing requirements. For instance, users can specify the maximum latency for monitoring messages to be delivered to a control station. Since the message must travel through several parts of the distributed system, the latency should be predicted to conclude if the desired maximum monitoring latency bound is achieved or not.

3. Planning for Heterogeneous Distributed IoT Systems

Current solutions to plan and predict latencies in distributed IoT systems are currently mostly based on a set of rules-of-thumb. Since time division multiple access (TDMA), token-ring and Carrier Sense Multiple Access (CSMA) based medium access control protocols can coexist in the same network, the approach is designed to work with any of these protocols and assumes heterogeneity. We consider protocols used in industry-strength solutions and propose a latency model, which can be used by the engineering teams to guarantee strict timing compliance in their heterogeneous plans. End-to-end latency can be divided into several parts: the time elapsed between when an event happens and its detection by the node (t_{Event}); the latency to acquire a sensor/event value (t_{Aq}); the time needed to reach the transmission slot (this time can be neglected if we synchronize acquisition with the upstream slot for the sensor node) ($t_{WaitTX\,Slot}$); wireless latency ($t_{W_{UP}}$); bright latency, which corresponds to the time consumed to do some processing in the gateway ($t_{Gateway}$) (e.g. translation between protocols); the time needed to take the next token to deliver data to the control station ($t_{WaitToken\,Slot}$); the local area network latency, which represents the latency needed to forward messages coming from the gateway to computer nodes or control stations (t_{LAN}); the latency associated with processing in the control station ($t_{\mathrm{Processing}}$).

Based on this set of latency parts, the end-to-end latency for each operation can be predicted.

Since heterogeneous distributed systems can have different configuration of the network (e.g. wireless TDMA plus wired CSMA, token-based End-to-End and wireless TDMA plus token-based wired) is useful to understand which parts of latencies are involved in each operation (monitoring and actuation).

Equation (1) is used to predict the maximum monitoring latency for the combination of TDMA-CSMA.

$$\max\left(Monitoring_{Latency}\right) = \max\left(t_{W_{AqE}}\right) + \max\left(t_{W_{UP}}\right) + \max\left(t_{Middleware}\right) + \max\left(t_{\mathrm{Processing}}\right) \tag{1}$$

Where $\max\left(t_{W_{AqE}}\right)$ is given by:

$$\max\left(t_{W_{AqE}}\right) = \max\left(t_{Event}\right) + \max\left(t_{Aq}\right) + \max\left(t_{WaitTX\,Slot}\right) \tag{2}$$

End-to-end latency for other configurations such as token-based End-to-End and wireless TDMA plus token-based wired can be predicted by (3) and (4), respectively.

$$\begin{aligned}\max\left(Monitoring_{Latency}\right) = {} & \max\left(t_{Middleware}\right) + \max\left(t_{W_{Down}}\right) + \max\left(t_{CMD\mathrm{Processing}}\right) + \max\left(t_{W_{AqE}}\right) + \\ & \max\left(t_{W_{UP}}\right) + \max\left(t_{Middleware}\right) + \max\left(t_{\mathrm{Processing}}\right)\end{aligned} \tag{3}$$

$$\max\left(Monitoring_{Latency}\right) = \max\left(t_{W_{AqE}}\right) + \max\left(t_{W_{UP}}\right) + \max\left(t_{Wait\ for\ Token}\right) + \max\left(t_{Middleware}\right) + \max\left(t_{CMD\,\mathrm{Processing}}\right) + \max\left(t_{Middleware}\right) + \max\left(t_{\mathrm{Processing}}\right) \tag{4}$$

In the down path, used by configuration or actuation commands, there are also latency parts that can be identified.

Consider a command sent from a control station to a node. Similar to upstream data transmission, there is LAN transmission latency (t_{LAN}). In the bright, there are three time intervals that can be considered ($t_{Gateway}$ and $t_{Wait\ for\ TX\ slot}$). $t_{Gateway}$ is similar to upstream data latency, It corresponds to the time needed for the bright to receive the command and do any processing.

If the wireless sub-network is running a TDMA schedule, upon receiving the command by the bright node, it needs to wait $t_{Wait\ for\ TX\ slot}$ to reach the transmission slot to send the command to the target node. This latency part represents the amount of time that a command is kept in the bright node until it gets a downstream slot.

Since the wireless sub-network schedule is planned, it is possible to place downstream slots in specific positions in the schedule to have a reasonable small value for this latency.

If the wireless sub-network is running a CSMA protocol, upon receiving the command by the bright node, it is sent to the target node immediately. In this case, the $t_{Wait\ for\ TX\ slot}$ will be zero.

Lastly, there are latencies associated with the wireless path ($t_{W_{Down}}$) and the target node processing the command ($t_{CMD\,\mathrm{Processing}}$). $t_{W_{Down}}$ corresponds to the time taken to transmit a command from the bright node to the target node, while $t_{CMD\,\mathrm{Processing}}$ corresponds to the time taken to process the command inside the target node.

Equation (5) is used to predict the maximum command latency.

$$\max\left(Command_{Latency}\right) = \max\left(t_{Middleware}\right) + \max\left(t_{W_{CMD}}\right) + \max\left(t_{CMD\,\mathrm{Processing}}\right) \tag{5}$$

Where $\max\left(t_{W_{CMD}}\right)$ is given by:

$$\max\left(t_{W_{CMD}}\right) = \max\left(t_{Wait\ forTX\ slot}\right) + \max\left(t_{W_{Down}}\right) \tag{6}$$

Considered a heterogeneous distributed system with token-based protocols for wired parts and a TDMA protocol for each wireless sub-network, we propose an algorithm that creates appropriate schedules for the wired network and wireless sub-networks, in order to meet the latency requirements.

The logic of the algorithm is to plan the size of the network so that the whole schedule is run sufficiently fast to provide latency guarantees. For instance, if an epoch is 2 seconds, it may take in the worst case 2 seconds plus the time in the wired part for a physical occurring event to be sensed and reach the control station. By dividing the wireless sub-network into two equal-sized parts with schedules of 1 second each, the same event will take at most 1 seconds plus the wired part to be detected in the control station. Figure 2 shows a control flow of the algorithm.

The algorithm receives as inputs the network configuration (nodes and their relation – e.g. leaf nodes, parents and grandparents), plus latency requirements. The algorithm starts by defining default TDMA and token-ring schedules for all nodes. The default schedule that the algorithm defines is based on each node sending its sampled data all the way to the control station at a time. This results in the first schedule sketches, which allows all nodes to send and receive data, but still needs to be completed or redefined to meet latency requirements. This first schedule did not define downstream slots yet.

After the default schedule is completed, the algorithm analysis command latency requirements for commands and closed loop operations, and determines how many downstream slots need to be added to the TDMA epoch to meet those requirements. Assuming that *n* downstream slots are needed, they are added at equally-spaced intervals, and the latencies of commands are recomputed to test if command latency requirements are already met. Otherwise, *n+1* downstream slot are tried and so on.

In the next step, and based on latency requirements for monitoring and closed loop operations, the algorithm verifies if those are met. If latency requirements are not met, it means that it is not possible to meet the timing requirements with the current size of the network, therefore the network is partitioned and the algorithm re-starts for each partition. The network partitioning is configured by the user and should be in similarly sized parts.

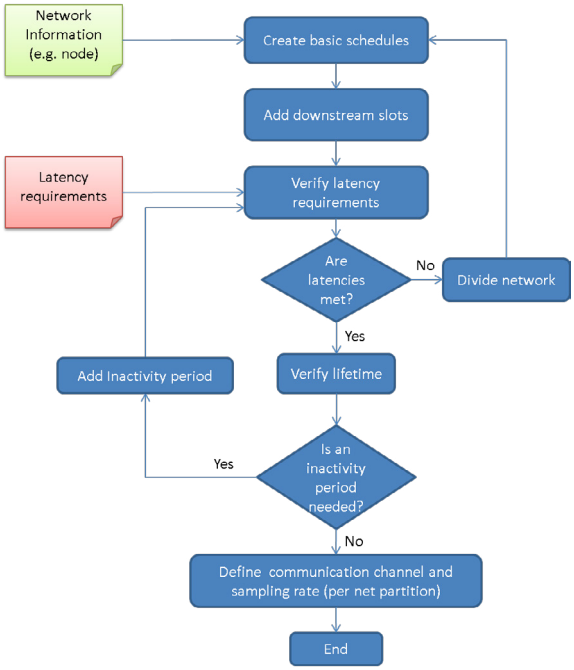

Fig. 1. Planning algorithm control flow.

4. Evaluation of Planning Algorithm

In this section, we present some results of experimental evaluation of the latency model and planning algorithm that were discussed in this paper. We will consider a heterogeneous network where wireless sub-networks coexist with wired-networks.

A testbed with a control station, a cabled network, one gateway, three wireless sub-network and four computer nodes was considered for this evaluation. A wireless sub-network including 13 TelosB nodes organized hierarchically in a 1-1-2 tree and one sink node (composed by one TelosB node and one computer that acts as gateway of the sub-network) is used. Besides, there were also used two other wireless sub-networks composed by 5 ESP8266 and 5 Arduinos with a wifly shield each. Figure 3 shows a sketch of our setup.

The testbed also includes a control station that receives the sensor samples, monitors operations performance using bounds, and collects data for testing the debugging approach.

The control station, gateway and four computer nodes are computers with Ethernet connection. They are connected through Ethernet cables and GigaBit network adapters. The TelosB sub-network is connected to the gateway using the serial interface provided by TelosB nodes. That interface is configured to operate at 460800 baud/second.

TelosB sensor nodes run Contiki OS and generate one message per time unit with a specified sensing rate or by request. Each message includes data measures such as temperature and light. GinMAC[5] is used as a TDMA protocol in TelosB network while a token-based protocol (similar to the Fieldbus) was developed for remaining wired and wireless parts.

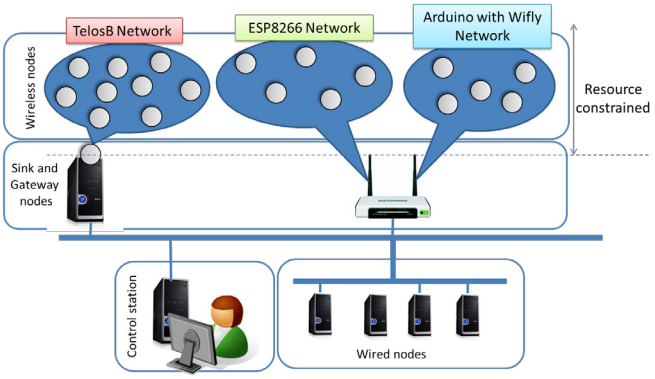

Fig. 2. Experimental setup.

4.1. Testing planning algorithm monitoring operations

Based on the setup and considering a monitoring operation (periodic sensor sampling) with a minimum sampling rate of 1 second and a maximum desirable monitoring latency of 200 ms, running the planning algorithm will result in the schedule of Fig. 3 for whole network.

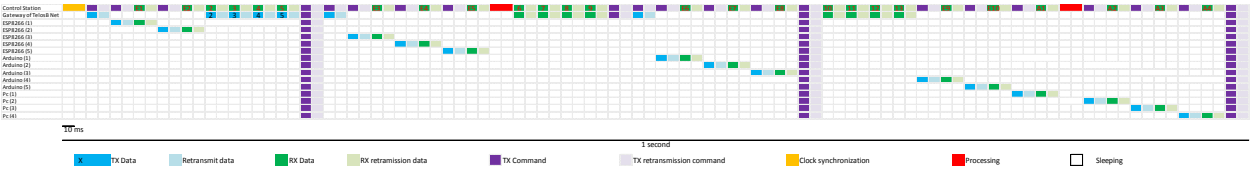

Fig. 3. Full network schedule.

From the schedule shown in Fig. 3 we conclude that all nodes were accommodate in the necessary epoch (1 second) and the control station hasn´t sleeping time. In our opinion, this is not a problem because the schedule already accommodates time to processing data and clock synchronizing.

In terms of the TDMA schedule for applying to the TelosB network, the schedule has an epoch with 1 second of length and an inactivity period of 300 ms. It includes sufficient slots for each node to transmit its data upwards. The schedule also includes transmission slots for sending configuration or actuation commands, slots for time synchronization and slots for node processing.

With resulting schedules, we ran the setup and collect statistical information of latency. We collect data per network level (nodes at the same level have similar latencies) for the TelosB network. We ran the experiment three times during, approximated, 1 day (24 hours) each time. Figure 5 shows statistical information of latency (per level), as well as the values corresponding to the prediction given by the latency formulas. Regarding to remaining nodes (nodes that run the token-based protocol), the average latency was about 12 ms and the maximum valor estimated by our latency model was 20 ms, which concluding that the proposed model predicts and dimensions the network well. The observed maximum value gathered during the test is always below the prediction. It is near, but below the planned maximum.

4.2. Testing planning algorithm for closed-loop operations

To exercise closed-loop operations and latency predictions, we define two testing setups:

Case 1: in order to test the planning algorithm for closed-loop operations, it was configured with a leaf node as a sensor and another leaf node as an actuator and 1 second as maximum operation latency.

Case 2: with the same configuration of case 1, we reduce the maximum admissible operation latency to 500 ms, and analyze the result latencies.

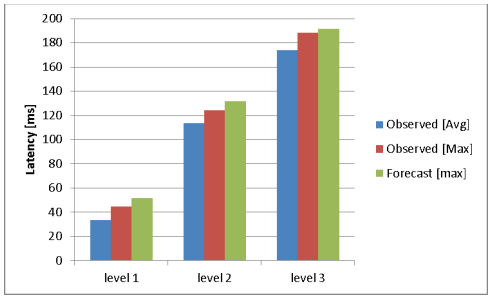

Fig. 4. Monitor latency per level with forecast.

Table 1. Closed-loop Latencies [ms].

Configuration		*WSN up*	*Middleware*	*Processing*	*Wait for TX slot*	*W down*	*CMD processing*	*Total*
Case 1	Testbed result	180	13.15	0.96	380	60	0.91	635
	Forecast	180	24.3	1	380	60	1	646.3
Case 2	Testbed result	180	11.95	0.86	11	60	0.71	264.5
	Forecast	180	24. 2	1	20	60	1	286.2

In the first case, the algorithm creates a schedule with one second of epoch and adds just one downstream slot to command a node. In the second case, since the latency requirement is strictest, the planning algorithm needs to add four downstream slots equally-spaced in the epoch to provide timing guarantees.

Table 1 shows the values for the two configurations of closed-loop control. The values resulted from applying the latency model are marked as “Forecast”, while the remaining values represent the maximum values obtained from experimental validation.

Each time measure of Table 1 represents the total amount of time consumed per part. From Table 1 we conclude that the observed latencies were always within the bounds defined by the forecasted maximum latencies, which allows concluding that the planning algorithm dimensions well the network and the latency model predicts well operation latencies.

5. Conclusions

End-to-end operation times planning in heterogeneous systems is an important research challenge: token-based and TDMA protocols offer provisions for dimensioning and link active schedule design. This process requires expert knowledge to result in appropriate timings and becomes complex for large networks, so that further investigation and development of approaches to automate and facilitate these steps are useful.

In this paper we proposed a planning algorithm that allows building a heterogeneous IoT system to operate with timing constrains. We consider heterogeneous IoT networks with wired and wireless parts to design a planning algorithm that deals with end-to-end operation timings. The proposed approach was evaluated in a real setup and the observed results were always within the bounds. With this work we conclude that the defined latency model predicts well operation latencies, providing operation time guaranties.

References

1. L. Zheng, “Industrial wireless sensor networks and standardizations: The trend of wireless sensor networks for process automation”, in Proceedings of SICE Annual Conference, 2010, pp. 1187 – 1190.
2. V. Lakkundi, J. Beran and Marko Krätzig, “Wireless Sensor Network Prototype in Virtual Automation Networks,” in The First IEEE International Workshop on Generation C Wireless Networks, 2008, pp. 89 – 94.
3. Fieldbus: Technical white paper planning and deploying, P. Automation, 2011.
4. P. Suriyachai, J. Brown, and U. Roedig, “Time-Critical Data Delivery in Wireless Sensor Networks,” in Distributed Computing in Sensor Systems, vol. 6131, 2010, pp. 216–229.
5. A. Dunkels, “The contikimac radio duty cycling protocol”, SICS Technical Report T2011:13, 2011.
6. I. Ungurean, N. C. Gaitan and V. G. Gaitan, "An IoT architecture for things from industrial environment," 2014 10th International Conference on Communications (COMM), Bucharest, 2014, pp. 1-4. doi: 10.1109/ICComm.2014.6866713.
7. Z. G. Liang, Q. Yang, Y. D. Wan, F. He, X. C. Wang, M. Wang, "The Study of IOT Architecture for Steel Strip Production Process", Advanced Materials Research, Vol. 572, pp. 364-370, 2012.
8. P. Bellagente, P. Ferrari, A. Flammini, S. Rinaldi and E. Sisinni, "Enabling PROFINET devices to work in IoT: Characterization and requirements," 2016 IEEE International Instrumentation and Measurement Technology Conference Proceedings, Taipei, 2016, pp. 1-6. doi: 10.1109/I2MTC.2016.7520417