Cyber-physical Information Systems for Enterprise Engineering
Cyber-physical Applications Timing

Miroslav Sveda and Patrik Halfar
Faculty of Information Technology, Brno University of Technology, Bozetechova 2, Brno, Czech Republic

Keywords: Enterprise Information Systems, Industrial Cyber-Physical Applications, Time Models, Supervisory Control and Data Acquisition (SCADA).

Abstract: This paper discusses the role of time in industrial cyber-physical applications of information systems using SCADA as a representative example. It restates the basics of the notion of time, which is important not only in general but particularly in enterprise domain, and focuses on industrial applications. Also, the manuscript brings SCADA concepts and relates them to cyber-physical system architectures. The case study demonstrates some time-related techniques in frame of a simple but real-world application.

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

1 INTRODUCTION

The notion of Cyber-Physical Systems (CPS) refers to integration of computation with physical processes. Embedded computers and networks monitor and control the physical processes, usually with feedback loops where physical processes affect computations and vice versa. In physical world, the passage of time is unchangeable and concurrency is inherent property of related processes. There are considerable challenges, particularly because the physical components of such systems introduce safety and reliability requirements qualitatively different from those in general-purpose computing (Lee, 2007).

In a CPS application, the function of a computation is defined by its effect on the physical world, which is in this case not only a system environment, but evidently also a component of the designed application system. The passage of time in CPS becomes a crucial feature. Time is central to predicting, measuring, and controlling properties of the physical world: given a deterministic physical model, the initial state, the inputs, and the amount of time elapsed, the current state of the plant can be computed. This principle provides foundations of Control Theory. However, for current mainstream programming paradigms, given the source code, the program's initial state, and the amount of time elapsed, we cannot reliably predict future program states for various computing system configurations. Moreover, when that program is integrated into a system with physical dynamics, this makes every design of the application much more complex even if the concrete configuration is known.

The manuscript aims at Enterprise Engineering application domain that stems both from well-founded industrial practice with its long well-known history, and from currently emerging theoretical discipline, see (Dietz, 2011).

This paper discusses the role of time in industrial CPS applications of information systems using the SCADA architecture as a typical example. Section 2 restates the basics of the notion of time in general and focuses on industrial, real-time applications. Section 3 brings SCADA concepts and relates them to CPS architectures. In section 4 some time-related techniques are demonstrated in frame of a simple application.

2 TIME

Contemporary Cybernetics and Computer Science deal -- in frame of their branches such as Artificial Intelligence, Systems Theory, or Software Engineering -- with various concepts and refinements of directed time. The logical time means that time passes only because something happens -- it respects order of events only. The absolute time means that a reference is established in relation to a unique event for a given system; evidently, it relates to some origin of date/time. The relative time means that a reference is established in relation to an

Sveda M. and Halfar P..
Cyber-physical Information Systems for Enterprise Engineering - Cyber-physical Applications Timing.
DOI: 10.5220/0004951204410446
In *Proceedings of the 16th International Conference on Enterprise Information Systems* (ICEIS-2014), pages 441-446
ISBN: 978-989-758-029-1
Copyright © 2014 SCITEPRESS (Science and Technology Publications, Lda.)

arbitrary selected event in the given system; clearly, it relates to time intervals. The global time means that the time is considered to be valid for the whole (distributed) system while the local time means that the time is valid for a part of the (distributed) system.

2.1 Time Denotation

Basic meaning of the term "time" can be introduced in the following complementary couples: physical/logical, absolute/relative, global/local. To be more precise, we consider an event domain, E, and a time domain, T, such that instead of viewing the precedence relation "to causally affect" on events we use members of a time domain to mark the members of the event domain to introduce a temporal order (Sveda, 2013).

2.2 Time Models

Models of time can be classed according to individuals (points, intervals), order (partial order, branching towards future, linear), boundedness (unbounded, beginning, ending), local structure (discrete, dense, continuous) and global structure (connectedness, homogeneity). Whereas synchronous models of computation regard all concurrent activities happen in a lock-step, asynchronous models are not restricted in this sense. They can be treated as interleaving models of computation, which sequentialize simultaneous actions non-deterministically, or as true concurrency models of computation, which impose only a partial ordering between actions.

Let us have an event domain, E, and a time domain, T, such that the members of the event domain are marked by the members of time domain to introduce a temporal order. For each of the domains E and T there are two possibilities how to choose domain elements: points and intervals. To preserve simplicity, we select points for both domains. Consequently, events can be interpreted as changes of system states and members of time domain as time instants. In this case, timing of events is mapping $E \rightarrow T$. Actions with non-zero duration can be described by their starting and ending points that require individual timings.

Point structures of domains E and T simplify introduction of partial order in general. Local time concept requires employing a partial order consistent with locality either of events or of timing. There are at least two natural possibilities how to introduce a timed partial order on events:

(a) To define locality as an equivalence relation on events Loc = (E, ~) and then, for each class of that equivalence to specify a separate linear time, i.e. to use multiple time lines; or

(b) To connect locality explicitly with temporal partial order (Loc×T, ~).

From the application viewpoint, both possibilities correspond to the same relation: for case ad (a), temporal partial order is induced on T by manifold mapping; for case ad (b), partially ordered time generates a decomposition of event domain into classes so that events in each class are mutually comparable by linear temporal order. Nevertheless, we prefer the case ad (a).

Local time can be defined as the time of a local physical clock, based on some periodic physical oscillation, whose frequency suits to measuring a duration of local process actions. This notion enables to ascribe time-structured system environment, e.g. system acting as an interface between processes with very different time scales, geographically extensive system subject to different local geographic times, and relativistic system based on the notion of inherent time. Moreover, relationships between two or more physical clocks can be specified by local time to depict not only clocks scaling, clocks shift, real clocks drift or skew, but also relativistic clocks mapping.

2.3 Temporal Structures and Timing

We can distinguish three local temporal structures: discrete, dense, and continuous. The same can be applied to space or even space-time coordinates. In this paper, we prefer scalable discrete time structure and fix discrete finite space structure in form of finite set of locations.

An implicit time domain of a system process respects internal events (changes in the state) of that process. An explicit time domain, on the other hand, consists of events that are not produced in the process, but which bear an observable temporal relation of the local process. Both types of timing can be considered as either internal to the local process or external to the remote processes (e.g. environmental processes). Evidently, implicit timing suffices only for a synchronous system timed by a common global clock or for a system driven by only one sequential process while real-time (asynchronous) distributed systems require explicit time domains.

A model of real-time systems is natural if its internal time corresponds well with the external,

physical time of the environment. However, different timing mechanisms rule various parallel environmental processes. In addition, distributed applications consider a distinctive, locally measured time for each node. A useful time model, therefore, must conform with external events as well as to internal timing. Furthermore, it should provide unambiguous semantics for a specification and implementation of real-time distributed systems.

3 SCADA

SCADA (Supervisory Control and Data Acquisition) systems are commonly deployed to persistently monitor and control industrial processes to assure proper functioning by automating telemetry and data acquisition in real time.

3.1 Traditional Architecture of SCADA Systems

Historically, SCADA systems were believed to be secure because they were implemented as isolated, remotely inaccessible networks. They usually consisted of an operator console or human-machine interface (HMI), connected to remote terminal units (RTU) and programmable logic controllers (PLC) through a proprietary purpose-specific protocol.

3.2 SCADA as a Distributed CPS

Yielding to market pressure that demands industries to operate with low costs and high efficiency, these systems are becoming increasingly more interconnected. Many of modern SCADA networks are interconnected both with the company enterprise networks and with the Internet. Furthermore, it is common that the HMI is a commodity PC, which is connected to RTU and PLC using standard technologies, such as Ethernet and WLAN (Rysavy, Rab, Sveda, 2013).

In fact, when RTU and PLC appear to be interconnected embedded systems, which is current state of the art at industrial applications, the resulting SCADA system can be dealt with as distributed CPS.

Besides security concerns, the SCADA control systems raise the issue of safety causing harm and catastrophic damage when they fail to support applications as intended. CPS domain paradigms suggest considering both application requirements, namely time constraints defined by physical processes of the system environment, and implementation aspects, namely computation and communication capacity constraints, from the beginning of system design. The design of a CPS application must consider namely functionality and dependability measures. Functionality means services delivery in the form and time fitting specifications, where the service specification is an agreed description of the expected service.

Dependability is that property of an embedded system that allows reliance to be justifiably placed on the service it delivers. A failure occurs when the delivered service deviates from the specified service. Security is concerned with the risks originating from the environment and potentially impacting the system, whereas safety deals with the risks arising from the system and potentially impacting the environment. Dependability measures consist of reliability, availability, security, safety and survivability. Availability is the ability to deliver shared service under given conditions for a given time, which means namely elimination of denial-of-service vulnerabilities.

Security is the ability to deliver service under given conditions without unauthorized disclosure or alteration of sensitive information. It includes privacy as assurances about disclosure and authenticity of senders and recipients. Security attributes add requirements to detect and avoid intentional faults. Safety is the ability to deliver service under given conditions with no catastrophic affects. Safety attributes add requirements to detect and avoid catastrophic failures. Of course, all dependability measures relate to timing requirements.

4 CASE STUDY

The presented case study concerns petrol dispenser with an electronic counter/controller (Sveda, Vrba, Rysavy, 2007). The application unfolds as safety critical, because here is danger of explosion, and security critical, due to possibility of loss of money. The system is a state oriented and comprises two classes of interfaces with different timing.

4.1 Application

The application concerns petrol dispenser with an electronic counter/controller. The application appears as (1) safety critical from the point of view of danger of explosion in case of uncontrolled petrol issue and (2) security critical from the point of view of danger of loss of money in case of unregistered

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

issue.

This application is designed according to the fail-stop model, which can be described by the following way. A fail-stop system never performs an erroneous state transformation due to a fault. Instead, the system halts and its state is irretrievably lost. The fail-stop model, originally developed for theoretical purposes, appears as a simple and useful conception supporting the implementation of fail-safe systems. Since any real solution can only approximate the fail-stop behavior and, moreover, the halted system offers no services for its environment, common fault-tolerance and fault-avoidance techniques must support such implementation.

4.2 State-based System Description

A dispenser control system communicates with its environment through two classes of I/O variables. The first class describes an interface with volume meter (I), pump motor (O), and main and by-pass valves (O) that enable full or throttled issue. The timing for this class is defined by flow velocity and measurement precision requirements. Second class of I/O-variables models human interface that must respect timing constants of human-physiology. This class contains release signal, unhooked nozzle detection, and product's unit prices as inputs; as for outputs, volume and price displays belong to this class.

The case study consists in the logical structure description of the two-level structure, where higher level behaves as an event-driven component and lower level behaves as time-evolving interconnected component. The behavior of the higher level component can be described by the following state sequences of a finite-state automaton with states blocked-idle, ready, full fuel, throttled and closed, and with inputs release, (nozzle) hung on/off, close (the preset or maximal displayable volume achieved), throttled (to slow down the flow to enable exact dosage) and error:

blocked-idle $\xrightarrow{release}$ ready $\xrightarrow{hung\ off}$ full_fuel $\xrightarrow{hung\ on}$ blocked-idle
blocked-idle $\xrightarrow{release}$ ready $\xrightarrow{hung\ off}$ full_fuel $\xrightarrow{throttle}$ throttled $\xrightarrow{hung\ on}$ blocked-idle
blocked-idle $\xrightarrow{release}$ ready $\xrightarrow{hung\ off}$ full_fuel $\xrightarrow{throttle}$ throttled $\xrightarrow{close}$ closed $\xrightarrow{hung\ on}$ blocked-idle
blocked-idle $\xrightarrow{error}$ blocked-error
blocked-idle $\xrightarrow{release}$ ready $\xrightarrow{error}$ blocked-error
blocked-idle $\xrightarrow{release}$ ready $\xrightarrow{hung\ off}$ full_fuel $\xrightarrow{error}$ blocked-error
blocked-idle $\xrightarrow{release}$ ready $\xrightarrow{hung\ off}$ full_fuel $\xrightarrow{throttle}$ throttled $\xrightarrow{error}$ blocked-error

The states full_fuel and throttled appear to be hazardous from the viewpoint of unchecked flow because the motor is on and the liquid is under pressure -- the only nozzle valve controls an issue in this case. Also, the state ready tends to be hazardous: when the nozzle is unhooked, the system transfers to the state full_fuel with flow enabled. Hence, the accepted fail-stop conception necessitates the detected error management in the form of transition to the state blocked-error. To initiate such a transition for flow blocking, the error detection in the hazardous states is necessary. On the other hand, the state blocked-idle is safe because the input signal release can be masked out by the system that, when some failure is detected, performs the internal transition from blocked-idle to blocked-error.

4.3 Incremental Measurement for Flow Control

The volume measurement and flow control represent the main functions of the hazardous states. The next applied application pattern, incremental measurement, means the recognition and counting of elementary volumes represented by rectangular impulses, which are generated by a photoelectric pulse generator. The maximal frequency of impulses and a pattern for their recognition depend on electro-magnetic interference characteristics. The lower-level application patterns are in this case a noise-tolerant impulse detector and a checking reversible counter. The first one represents a clock-timed impulse-recognition automaton that implements the periodic sampling of its input with values 0 and 1. This automaton with b states recognizes an impulse after b/2 (b>=4) samples with the value 1 followed by b/2 samples with the value 0, possibly interleaved by induced error values, see an example timed-state sequence:

$(0, q_1) \xrightarrow{inp=0} \ldots \xrightarrow{inp=0} (i, q_1) \xrightarrow{inp=1} (i+1, q_2) \xrightarrow{inp=0} \ldots \xrightarrow{inp=0} (j, q_2) \ldots$
$\ldots \xrightarrow{inp=1} (k, q_{b/2+1}) \xrightarrow{inp=1} \ldots$
$\ldots \xrightarrow{inp=1} (m, q_{b-1}) \xrightarrow{inp=0} (m+1, q_b) \xrightarrow{inp=1} \ldots \xrightarrow{inp=1} (n, q_b) \xrightarrow{inp=0/IMP} (n+1, q_1)$

i, j, k, m, n are integers representing discrete time instances in increasing order.

For the sake of fault-detection requirements, the incremental detector and transfer path are doubled. Consequently, the second, identical noise-tolerant impulse detector appears necessary.

The subsequent lower-level application pattern used provides a checking reversible counter, which starts with the value (h + l)/2 and increments or decrements that value according to the impulse detected outputs from the first or the second recognition automaton. Overflow or underflow of the pre-set values of h or l indicates an error. Another counter that counts the recognized impulses from one of the recognition automata maintains the whole measured volume. The output of the letter

automaton refines to two displays with local memories not only for the reason of robustness (they can be compared) but also for functional requirements (double-face stand). To guarantee the overall fault detection capability of the device, it is necessary also to consider checking the counter. This task can be maintained by an I/O watchdog application pattern that can compare input impulses from the photoelectric pulse generator and the changes of the total value; evidently, the appropriate automaton provides again reversible counting.

5 CONCLUSIONS

This paper argues about the role of time in industrial cyber-physical applications of information systems using SCADA as a representative pattern. It restates the essentials of the notion of time important not only in general computer applications but also and particularly in enterprise domain while focusing on industrial supervision. Also, the manuscript brings SCADA concepts and relates them to cyber-physical system architectures. The case study demonstrates some time-related techniques in frame of a simple but real-world application.

This paper -- which was inspired originally by the reports (Lee, 2007) and (Dietz, 2011) -- excerpts and integrates some time-related ideas aiming at CPS design domain. They were previously published "per partes" in papers (Sveda and Vrba and Rysavy, 2007), (Sveda and Vrba, 2010), (Sveda and Vrba, 2011), (Rysavy and Sveda and Vrba, 2012) and (Sveda and Vrba, 2013).

The references that wind up this manuscript should be complemented by many more items related to time modeling, namely those that were the original source for the paradigm presented; nevertheless they can be retrieved entirely from the article (Sveda, 2013). Also, some technicalities related to SCADA applications are dealt in more detail by (Rysavy and Rab and Halfar and Sveda, 2012) and (Rysavy and Rab and Sveda, 2013).

ACKNOWLEDGEMENTS

This project has been carried out with a financial support from the Czech Republic state budget by the IT4Innovations Centre of Excellence, EU, CZ 1.05/1.1.00/02.0070 CEZ and by the MMT project no. MSM0021630528: Security-Oriented Research in Information Technology, by the Technological Agency of the Czech Republic through the grant no. TA01010632: SCADA system for control and monitoring RT processes, and by the Brno University of Technology, Faculty of Information Technology through the specific research grant no. FIT-S-11-1: Advanced Secured, Reliable and Adaptive Information Technologies and through research grant no. FIT-S-14-2299: Research and application of advanced methods in ICT. The authors acknowledge contributions to the presented work by his colleagues Ondrej Rysavy, Petr Matousek, Jaroslav Rab, Vladimir Vesely, Matej Gregr and Libor Polcak from the Faculty of Information Technology, the Brno University of Technology.

REFERENCES

Atlagic, B., Milinkov, D., Sagi, M., Bogovac, B., 2011. High-Performance Networked SCADA Architecture for Safety-Critical Systems, In: Proceedings of the *Second Eastern European Regional Conference on the Engineering of Computer Based Systems EERC-ECBS*, pp.147–148.

Dietz, Jan L. G. (editor), 2011. *Enterprise Engineering – The Manifesto*. www.ciaonetwork.org.

Lee, E. A., 2007. *Computing Foundations and Practice for Cyber-Physical Systems: A Preliminary Report*. Technical Report No.UCB/EECS-2007-72. Electrical Engineering and Computer Sciences, University of California at Berkeley.

Rysavy, O., Rab, J., Halfar P., Sveda, M., 2012. A Formal Authorization Framework for Networked SCADA Systems, In: Proceedings of the *19th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems ECBS*, Novy Sad, RS, IEEE CS, pp.298-302.

Rysavy, O., Sveda, M., Vrba, R. 2012. A Framework for Cyber-Physical Systems Design -- A Concept Study, In: Proceedings of the *International Conference on Systems, ICONS 2012*, Saint Gilles, Reunion Island, US, IARIA, pp.79-82.

Rysavy, O., Rab, J., Sveda, M., 2013. Improving security in SCADA systems through firewall policy analysis, In: Proceedings of the *Federated Conference on the Computer Science and Information Systems 2013*, Krakow, PL, IEEE CS, pp.1435-1440.

Sveda, M., Vrba, R., Rysavy, O., 2007. Pattern-Driven Reuse of Embedded Control Design -- Behavioral and Architectural Specifications in Embedded Control System Designs. In: Proceedings of the *Fourth International Conference on Informatics in Control, Automation and Robotics ICINCO 2007*, Angers, FR, INSTICC, 8pp.

Sveda, M., Vrba, R., 2010. An Embedded Application Regarded as Cyber-Physical System, In: Proceedings of the Fifth International Conference on Systems

ICONS 2010, Les Menuires, FR, IEEE CS, pp.170-174.

Sveda, M., Vrba, R., 2011. A Cyber-Physical System Design Approach, In: Proceedings of *Sixth International Conference on Systems - ICONS 2011*, St. Maarten, AN, IARIA, pp.12-18.

Sveda, M, Vrba, R., 2013. Cyber-Physical Systems Networking with TCP/IP -- A Security Application Approach, In: IEEE Proceedings *AFRICON 2013*, New York, US, IEEE, 2013, pp.101-106.

Sveda, M., 2013. Time Specification, Modeling and Measurement in frame of Cyber-Physical System Applications Design. *International Journal of Systems Applications, Engineering & Development*, Vol.7, No.6, 2013, pp.263-270.

SCITEPRESS
SCIENCE AND TECHNOLOGY PUBLICATIONS