A Versatile Real-Time Experiment: Balancing a Ball on a Flat Board

Dieter Zöbel

Universität Koblenz-Landau
Fachbereich Informatik, Institut für Softwaretechnik
Rheinau 1, D-56075 Koblenz
zoebel@uni-koblenz.de
http://www.uni-koblenz.de/~zoebel

Abstract

Comparable to the dining philosophers in parallel programming we look for a paradigmatic problem for real-time systems. Our experimental system for balancing a ball on a flat board is described and assessed under its animative, curricular, and scientific aspects. Particularly under the curricular aspect the experimental system offers transferable solutions for various subjects in real-time systems education e.g. modeling, analysis, scheduling, software engineering, techniques of programming, operating systems and computer networks

1 Introduction

Thomas S. Kuhn [13] states in his book "The Structure of Scientific Revolutions" that the existence of adequate paradigms is an indicator for the maturity of a scientific subject. With respect to real-time systems these paradigms should constitute the unifying conception of a community of scientists in what is essential for their subject. Kuhn particularly emphasizes the necessity of exemplary problems and corresponding solutions which are transferable to application-specific problems of the real world.

The lack of such community-wide problems became evident when we began to write out textbook "Echtzeitsysteme" on real-time systems [18]. We would have appreciated to use paradigmatic problems to demonstrate our approach in analyzing, designing, scheduling and programming components of real-time systems and to compare our approach with similar approaches. Other authors of profound textbooks on real-time systems equally suffer from this lack (see [14], [12], [11], and [7]).

However, to give an example, the use of paradigmatic problems to accentuate the essentials of a scientific subject is excellently demonstrated in parallel programming. Various textbooks and surveys are based on this conception (see [6], [15], [10], [4], and [3]). With respect to parallel programming the community of scientists is familiar with a variety of paradigmatic problems (e.g. producer-consumer, reader-writer, dining philosophers) to assess the aptness of conceptions, methods and languages for parallel programming.

Our experimental system has the objective to catch a rolling ball in the center of a flat board by merely inclining the board in an adequate way. The paradigmatic character of the problem is not satisfactory, yet, because the question is too complex and the solution is too technical. But we are convinced to be on the right way in that on one hand we have a problem which is easy to survey, with a correspondence to human capabilities. On the other hand this problem and our solution touches a variety of subjects in real-time computing, e.g.:

- system analysis, modeling, and scheduling
- programming techniques
- the phases of software development
- computer architecture and network architecture

which are central subjects in real-time education. The experimental system also permits different modifications

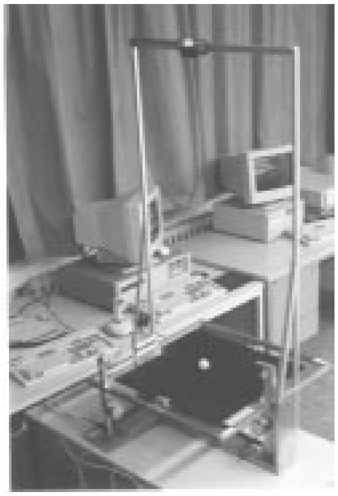

Figure 1: Picture of the experimental system

and extensions and will be examined under animative, curricular and scientific aspects.

The paper is organized as follows. Section 2 reflects the structure of the experimental system referring to its technical, electrical, and computational design. Then the next three sections are dedicated to educational aspects. First we consider it as very important to present an experimental system where students enjoy to work on. The result of their endeavor should be visible for themselves and presentable to others. So section 3 considers the animative aspects of the experimental system. Section 4 concentrates on curricular aspects. As listed above a variety of subjects of real-time computing are covered and demand for systematically designed solutions in the style of an engineering science. Beyond that the experimental system offers the possibility of scientific studies. Some questions which immediately arise in this context are discussed in section 5. Finally section 6 summarizes the benefits and the defects of the experimental system and looks for further directions.

2 The structure of the experimental system

The principle intend was to balance a ball on a board just like a human being would do, e.g. a waiter with his tray. To make such a system operable it is required to think and decide about a technical construction, controllable drives, and an adequate computer system (see figure 1).

The technical construction consists of an aluminium frame carrying the board, the video camera and the two electronic motors. The board has two orthogonal axes of rotation. Each of the motors is connected to the board by a belt and able to incline the board between -15° and +15° around one axis (α- or β-axis). The object of control is a white ball (the ball of a mouse) which rolls on the black painted surface of the board. So the ball marks a hard contrast to the board when shot from the video camera situated at about one meter above.

A black and white video camera is sufficient for the detection of the ball. The corresponding frame grabber card delivers 50 half images or 25 full images per second. Hence from this follows a periodic triggering of the corresponding position detection of the ball every $40ms$. An external real-time clock with the resolution of $1ms$[1] is used to record the time for taking the image.

The detection of the ball's position is based on a 64×64 bitmap. This incomming data must be reduced to the tuple (x, y, t) marking the x- and y-coordinates of the ball at (real-)time[2] t. The necessary data reduction is characteristic for real-time systems and corresponds to equally characteristic data expansion for controlling the electric motors (see figure 2).

In the experimental system step motors with variable reluctance are use. They require the right impulses at fine grained instances of time. The necessary resultion of time is achieved by external real-time clocks for each of the motors.

To divide the load imposed by data reduction and data expansion purposes a system of 4 PCs (one 486DX-33 and three 386DX-25) corresponds to the functional decomposition of the entire system:

[1] The standard time resolution of $1s/18.2$ by a standard PC is too coarse grained.

[2] Here real-time (is defined (see [18]) by a mapping from physical time to a discrete scale

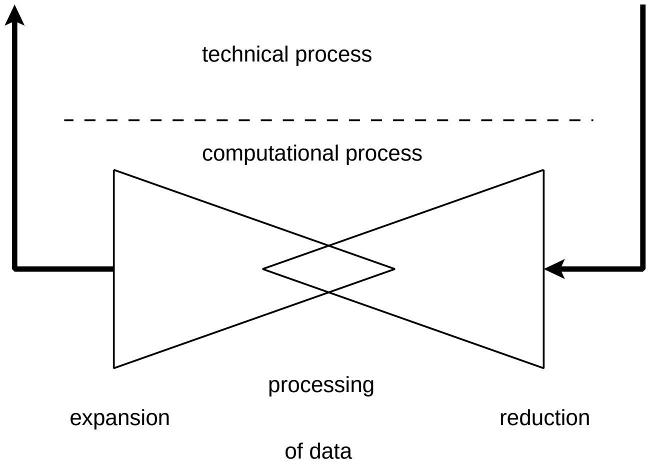

Figure 2: Reduction and expansion of sensor and actor data in real-time systems

PC1	detection of the ball's position
PC2	counteraction: computation of the inclinations α and β
PC3 and PC4	inclining the board to the actual angles α and β

All PCs run MS-DOS and thereupon the real-time executive RTKernel[3].

The necessary data and event propagation between the PCs is carried out by a CAN-bus. The important real-time feature of its CSMA/CA-protocol[4] is that in the case of conflicting demands for sending a message always the most urgent one gains the medium. Though run at a rate of only $0.1 Mbit/s$ the feature of urgency combined with the compressed data structure suffices to propagate the necessary information in time. The main traffic consists of:

PC1 to PC2	(x, y, t)
PC2 to PC3	(α)
PC2 to PC4	(β)

Additionally, by another mode of operation PC2 does not take its input from PC1. Instead in a manual mode PC2 reacts on the input from a joystick and translates its movements to an inclination of the board. So a human being can replace the autocontrolled counteraction and directly execute the right (or wrong) balancing actions.

[3]See Web-page http://www.on-time.com

[4]carrier sense multiple access / collision detection

3 Animative aspects

The experimental system described in the last section was constructed and implemented by two students of computer science as full term exam task in the second part of their studies. After a long haul with the selection of the adequate mechanical and electrical components the devotion of the students grew from cool to passionate. Various steps of success, e.g. driving the motors with the joystick or detecting the ball from the bitmap image, led to increasing courage in acting and reflecting independently.

The animative aspects, emerging from this experimental system, must not be neglected. The task of the two students is already completed. But we expect stimulating impulses for our next courses on real-time systems. Even if a student has no analytical understanding of the system it becomes obvious that the correctness of the computational processes not only depends on functional correctness. Instead it is evident that timeliness is indispensable for the computational processes together with the technical processes. This should result in a request for analytical considerations on how fast the ball may become, which is the maximal angular velocity of the board, or what are the criteria for an adequate functional decomposition of the system, hence in a request for a structured approach to the development of real-time software.

Additionally the experimental system, as it exists now, is open for modifications and extensions, e.g. in the context of practical (laboratory) studies as they are part of the formation in computer science. With respect to the modification of the existing system it is most attractive to substitute the inner functional component: the counteraction. Our solution based on a collection of heuristics may – due to the simple interface – be replaced by

- a solution based on control theory,
- a fuzzy set solution,
- a neural network solution.

With respect to an extension of the animative facilities it would be a challenge to get the ball to a steady movement on an imaginary circle instead of a final standstill.

A very interesting experience with the animative character of the experimental system has already been made. From time to time our department is visited by high school classes. These occasions are used to acquaint potential students with computer science. Real-time systems is only one constituent of this formation, but with tangible impacts on our actual and future life. The abilities of the computational system can be experienced when the autocontrol mode is replaced by the manual mode, that is to say when the joystick is passed to one of the schoolboys or -girls who has to balance to ball manually.

Finally we are thinking about the possibility to make our experimental system operable via the internet. Up to now we are present in the internet with a video clip (an avi-file loadable from the Web-page http://www.uni-koblenz.de/˜zoebel[5]). But watching a video clip is not very animative. Instead we are looking for an extended experimental system with:

- a technical mechanism to kick of the ball
- a fence around the board which registers any touch of the ball
- an applet for remote configuration, operation and visualization of the experimental system (primarily the actual position of the ball and the inclination of the board should be visible on the client side)
- a programming facility to execute client's algorithms in the place of the original counteraction

As a summarizing experience it schuld be perceived that the animative aspect of the experimental system is more than a supplement. We believe that the figurative and metaphorical properties of such a system – comparable to the attraction by the metaphor of the dining philosophers – is a driving force for the degree of acceptance a paradigmatic problem encounters in an community of scientists.

4 Curricular aspects

The experimental system is integrated into the formation of computer scientists in the second part of their studies[6]. In the following we list those skills and subjects which are covered by the existing experimental system, its modifications, and extensions. Most of them are essential for real-time education.

- First of all the technical system has to be analyzed. There are various physical and technical considerations which are interesting for keeping the ball on the board. So one measurable property of the system is that the board can change its inclination from -15° to +15° within less than a second (independently for the angles α and β). Other properties, e.g. the acceleration of the ball on the board or the maximum velocity, have to be derived from physical laws or engineering rules.

  Example: The gravitational acceleration of a body sliding on an inclined surface with inclination α is give by:

  $$g \sin \alpha$$

  For a solid ball which also accumulates an increasing impulse the following formula of acceleration is used [8]:

  $$\frac{5}{7} g \sin \alpha$$

  Based on this formula we can compute the run time of the ball while steadily rotating the board from +15° to 0°. For a distance of $25cm$ which equals the distance from one corner to the middle of the board we derive by formula

  $$s(t) = \frac{1}{2} t^2 \int_0^{15} \frac{5}{7} g \sin \alpha d\alpha$$

  the ball's run time of about $1.44s$.With the formula

  $$v(t) = t \int_0^{15} \frac{5}{7} g \sin \alpha d\alpha$$

  this corresponds to velocity of $0.34m/s$ at the center of the board. To catch the ball on the other side of the center it is necessary to reach an angular velocity of

  $$\frac{10.7°}{s} \quad (= \frac{15°}{1.44s})$$

  which can easily be achieved by the step motors.

[5] $30 Mbyte$!!

[6] Compared to the Anglo-Saxon formation somewhere between the bachelor's and the master's degree.

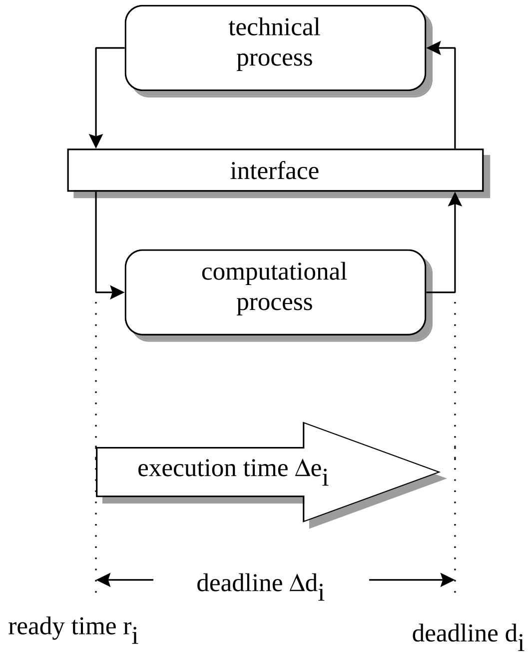

Figure 3: Basic model of a real-time system

- A characterizing property of hard real-time systems is that the execution time for certain computations is bound by deadlines. In our context there is a dominating ready time and deadline given by the camera and the frame grabber card. So any functionality with reference to the actual position of the ball on the board must be completed within $40ms$.

  Example: A characteristic property of real-time systems is given by a definition on probability [18]:

  $$P[r + \Delta e \leq d|B] = p_c$$

  The value 1 for p_c determines a hard real-time system where timeliness is guaranteed. In this context above timeliness is given when a computation becomes ready at time r and its computation time Δe including all predictable delays does not exceed deadline d. But also real-time systems may fail, e.g. due to technical faults. Therefore predicate B explicitly accumulates all cases for which a real-time system operates well. With respect to the functionality of PC1 the time bound imposed by the camera and the frame grabber card states that the detection algorithm including the transmission of the tuple (x, y, t) to PC2 must be completed before the next image is to be examined (see figure 3):

  $$P[\Delta e \leq 40ms|\text{no faults}] = 1$$

- Software systems exceeding a certain level of complexity should be developed with methods which are well understood by the team of software developers and also by their successors in subsequent phases of the software live cycle. This is also true for real-time system development. The experimental system has enough complexity to demonstrate the conceptual, organizational and modeling framework constituting such a methodology. In the educational context the method itself (here RTSA[7], see [9]) is of secondary interest, the primary interest is motivating and experiencing methodology and its framework.

  Example: In RTSA the decomposition of software follows the paradigm of functionality. The modeling framework is given by hierarchical diagrams, starting with a context diagram assembling all the relevant I/O-activities. The process of refinement and its discussion in the team of software developers is a excellent discipline for students which otherwise prefer to study solitarily.

[7] real-time structured analysis

- With respect to real-time systems education the subject of scheduling is probably the most significant. The major part of scientific literature on real-time systems focuses on scheduling and its impacts on other subjects e.g. synchronization of processes with priorities and time bounds. There exists an enormous variety of computational models, analytical results and efficient algorithms (for a good overview see [7]). So it is obvious that the breadth of this subject is out of scope when looking for adequate scheduling techniques in the experimental system. But we can ask those questions which necessarily had to be answered in the sequel of developing its software:

  - Which is the apt computational model for processes: preemptive vs. non-preemptive?
  - What is the triggering mechanism to set a process to a ready state: time-triggered vs. event-triggered?
  - What kind of time bounds are imposed by the technical system for the execution of a certain process: sporadic vs. periodic?
  - How can the computational load be split on a multiprocessor system or a distributed system (in contrast to a uni-processor system): static placement policies vs. dynamic balancing policies?

- The question which operating system (or executive) to use for the development of a real-time system is very important with respect to predictability, versatility, portability, and performance. Particularly for long term projects it is necessary to assess and compare the features of operating systems. The characterizing feature for real-time operating systems is the predictability of service times and the interleaving of the process' execution. Unfortunately this feature is not realized yet with the available programming interfaces (such as POSIX 1003.4) or methodologies (e.g. methods to propagate priorities between processes). Experiencing the observable needs and the existing defects is very instructive for both students and lecturers.

  Example: Any of the distributed PCs of our experimental system has a simple structure and an executive on each PC is sufficient to run the few processes. Assuming the modification that the experimental system were realized with a powerful computer there would be a lot of concurrent processes. These would run at different levels of priority and use synchronization operations to access critical data. Hence we implicitly meet the problem of priority inversion and are urged to look for an operating system which applies some counterstrategy to this phenomenon. In this context it is very exciting to compare the theoretical approaches (initiated by [16]) with their realizations based as certain system calls to operating systems.

Various other subjects with educational relevance emerge from the experimental system: e.g. the integration of interrupt driven routines, the exploitation of a real-time communication protocol, the balancing of load, and laboratory skills in general. Modifications and extensions open the possibility to consider other relevant subjects such as security, robustness and clock synchronization.

5 Scientific aspects

University studies are charged with the goal to acquaint students with the world of science, both in a receptive and performing manner. Particularly the latter can be exercised in the context of the experimental system. Various challenging questions of scientific interest immediately emerge from a deeper insight into our system. Two of them are listed here:

- Relatively underestimated in scientific literature on real-time systems is the problem of attaching time to events or observations. There are different sources of failure and drift between physical time and real-time. Analysis and scheduling tend to operate with physical time whereas the real-time is the basis of operation for the computational processes. Only few papers (e.g. [5] and [11]) try to classify and assess these defects.

  Example: The computation of position, velocity and acceleration of the ball is an excellent example to study temporal and data accuracy as well as their interaction. In this context sources of failures and drifts emerge from:

  - the granularity of the real time clock ($1ms$)
  - the resolution of the video image (64×64 pixels for a board of $40cm \times 40cm$)
  - the detection of the ball's position at a given time (PC1 computing the tuple (x, y, t))
  - the latency for the transmission of the tuple (x, y, t) via network (CAN-bus)
  - the time spared until the tuple (x, y, t) is used by PC2 to compute the counteraction

- The experimental system is also an excellent example to demonstrate a new type of time constraint. This constraint may be characterized by the paradigm that the real-time system – in the view of data-bases – contains an image of the real world. The time bound states that this image must be sufficiently fresh. Efficiently scheduling processes with this type of constraints requires new strategies and algorithms (e.g. [1], [2], and [17]).

Certainly these are not the only scientific directions which deserve to be considered in the context of the experimental system.

6 Conclusion

Finally let us focus on some pros and contras of the experimental system under the aspect of a paradigmatic problem for real-time systems. Compared with the dining philosophers our exemplary problem is still too extensive and too technical with to many details (e.g. size of the board, model of acceleration, etc.) On the other hand we are able to apply methodologies (e.g. the functional decomposition), to ask the right categorizing questions (with respect to modeling and scheduling decisions) and to look for the adequate techniques of programming (e.g. avoiding priority inversion). Moreover the experimental system in its actual form both integrates and demonstrates the breadth of subjects which have to be covered by real-time systems education.

Our experience with the existing system is encouraging. Therefore we continue to think about paradigmatic problems for real-time systems, first in that we strive for further abstraction of the existing system and secondly in that we look for other problems with a strongly animative aspect, e.g. the problem to back up a lorry with trailer.

References

[1] W. Albrecht and D. Zöbel. Hard real-time scheduling for age constraint computations in monitoring and control. In *Proceedings of the 2nd IFAC Workshop on New Trends in Design of Control Systems*, Smolenice, Slovak Republic, September 1997. Elsevier Science Ltd.

[2] Wolfgang Albrecht. *Echtzeitplanung bei Altersanforderungen.* PhD thesis, Universität Koblenz-Landau, Fachbereich Informatik, 1998.

[3] G. R. Andrews. *Concurrent Programming.* The Benjamin/Cummings Publishing Company, 1991.

[4] G. R. Andrews. Paradigms for process interaction in distributed programs. *ACM Computing Surveys*, 23(1):49–90, March 1991.

[5] N. Audsley, A. Burns, M. Richardson, K. Tindell, and A. Wellings. Absolute and relative temporal constraints in hard real-time databases. In *Proc. of IEEE Euromicro Workshop on Real Time Systems*, February 1992.

[6] M. Ben-Ari. *Principles of Concurrent Programming.* Prentice Hall Int., Eaglewood Cliffs, NJ., 1982.

[7] G. C. Buttazzo. *Hard Real-Time Computin Systems: Predictable Scheduling, Algorithms and Applications.* Kluwer Academic Publishers, 1997.

[8] Chr. Gerthsen and H. O. Kneser. *Physik: Ein Lehrbuch zum Gebrauch neben Vorlesungen*, volume 11. Springer Verlag, Heidelberg, 1971.

[9] Hassan Gomaa. *Software Design Methods for Concurrent and Real-Time Systems.* The SEI Series in Software Engineering. Addison-Wesley, Reading, Mass., 1993.

[10] R. G. Herrtwich and G. Hommel. *Kooperation und Konkurrenz. Nebenläufige, verteilte und echtzeitabhängige Programmsysteme.* Springer-Verlag, Berlin, 1989.

[11] Hermann Kopetz. *Rel-Time Systems - Design Principles for Distributed Embedded Applications.* Kluwer Academic Publishers, Boston, 1997.

[12] C.M. Krishna and Kang G. Shin. *Real-Time Systems.* McGraw-Hill Companies, New York, 1996.

[13] Thomas S. Kuhn. *The Structure of Scientific Revolutions.* University of Chicago Press, 1969.

[14] Phil Laplante. *Real-Time Systems Design and Analysis: An Engineer's Handbook.* IEEE Press, New York, 1993.

[15] R. H. Perrot. *Parallel Programming.* Addison-Wesley Publishing Company, Reading, Mass., 1987.

[16] Lui Sha, Ragunathan Rajkumar, and John P. Lehoczky. Priority inheritance protocols: An approach to real-time synchronisation. *IEEE Transactions on Computers*, 39(9):1175–1185, September 1990.

[17] Dieter Zöbel. Scheduling algorthms for time-critical systems. In *Informatics and Algorithms*, pages 21–26, Presov, Slovakia, September 1998. Faculty of Production Technologies at the Technical University of Košice with residence in Prešov.

[18] Dieter Zöbel and Wolfgang Albrecht. *Echtzeitsysteme: Grundlagen und Techniken.* Internat. Thomson Publ., Bonn, Albany, 1995.