What is time? / What is embodiment?

Interfacing with time

Specifying timing requirements

Satisfying timing requirements

Real-Time & Embedded Systems

What is time? / What is embodiment?

E. Kligerman, A. D. Stoyenko

"Real-time", a regular occurrence of events, happens in real time and real space.

What is embodiment?

E. Kligerman, A. D. Stoyenko

"Embodied" phenomena are those that by their very nature occur in real time and real space.

What is time? / What is embodiment?

Real-Time & Embedded Systems

Martin Heidegger (1889-1976, Freiburg)

What is embodiment?

Essential phenomenology: from a discussion about mental phenomena separated from the physical world (Cartesian Dualism) to a discussion about a conscious mind and physical phenomena.

The meaning 'is not in the head' but in the world.

What is time? / What is embodiment?

Real-Time & Embedded Systems

References for this chapter

(Editors) Paul Dourish

The Australian National University

Allanku, Barry and Andy Welfling

"Real-Time Systems and Programming Languages"

Addison Wesley, third edition, 2004

© 2004 Uwe R. Zimmer, The Australian National University

Page 235 of 769 (chapter 4: to 307)

Chapter 4: To 307

Practical consequences: clocks in satellites need to be adjusted accordingly.

Practical consequences: clocks in satellites need to be adjusted accordingly.

Practical consequences: clocks in satellites need to be adjusted accordingly.
Meaningfully embedded systems are part of an ‘ecological niche’ that distinguishes them from other interactive systems.

- The operational environment is supported and determined by the system.
- The embedded system is constructed as a part of the operational environment and according to the task.
- The task is meaningful considering the morphology and cognitive ability of the system, as well as the response from the environment.

meaningfully embedded, and self-sufficient
timeout on actions

```
task body Controller is
  begin
    whileContinue in time and with the timing requirements 
    improve, improve, improve, …
    continue
    declare
    end Controller;
```

The big topics:

- **What is time? / What is embodiment?**
- **Interfacing with time**
- Specifying timing requirements
- Satisfying timing requirements
Continuous time-scopes for the synchronous system on an actual system.

2. POSIX timers

Ada95

Specifying timing requirements

Ada95 has no explicit time-scope expressions at task-level.

Ada95 offers...

Specifying timing requirements

Real-Time Java

priority scheduler (the only required scheduler)

synchronization and communication primitives

Maximum execution time (not requested, in the meantime, the thread might be activated)

cost

Sporadic events are not allowed to come in here

Priority scheduling

on-time schedulability analysis

deadline violation handlers

...to create the basic for most kinds of hard real-time-scope's manually. But no automatic schedulability analysis!

Esterel

Since Esterel is a synchronous language, ...

...to the usual: use timers!

 POSIX

common continuation

usage of Ada95 together with POSIX timers as a basis for hard real-time-scope's and schedules
Real-Time & Embedded Systems
Notions of time and space

The big topics:

What is time? / What is embodiment?

Interfacing with time

Specifying timing requirements

Satisfying timing requirements

Two paths towards fulfilling r-t-requirements:

☞ Real-time logic approach
  Formal, correct in its specifications, & often calculates for asynchronous, real-time systems
  → results in ignoring real-world effects, like jitters, drifts, failures, interferences, etc.
  → gives a formal solution according to the specification

☞ Complex systems oriented approach
  Deals with existing computer systems, increments, & offers a set of approximating methods
  → not complete or correct in any formal sense
  → deals with real-world systems, gives "evaluation" systems, passes rigorous experiments

Fulfilling rt-requirements:

Complex systems oriented approach:
  • System identification and compile-time analysis:
    • Calculate or limit statement durations
    • Calculate or limit iterations and recursions
    • Analyze potential dead or life locks → chapter 8
    • Calculate schedulability → chapter 7
  • Run-time analysis and checks:
    • Dynamic scheduling schemes: Re-validate schedulability → chapter 7
    • Check for all constraints and assertions at runtime → chapter 8
  • Supply fault-tolerant behaviours:
    • Error recoveries, mode changes, ... → chapter 9

Real-time logic approach:
  • Reduce the problem:
    • Reduce any asynchronous, analog, dynamical, fractal, jitter, drift, or failure-affected parts of the system to a fully synchronous and discrete system → chapters 2 and 5
  • Estimate the specification on the basis of the reduced synchronous system
  • Verify the reduced system:
    • Verify the reduced system against the specifications → not covered in this course
  • Compile the reduced system to an actual system:
    • The resulting actual system will be executable on a real machine and employ real devices
  • Re-check the actual system (e.g. by means of a complex systems-approach) → ...