Johan J. Lukkien, j.j.lukkien@tue.nl
TU/e Informatica, System Architecture and Networking

Real-Time Architectures 2003/2004

Specification concepts

Johan Lukkien

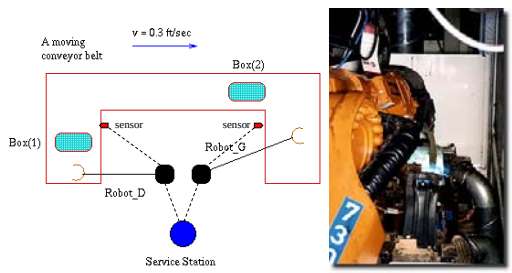
Johan J. Lukkien, j.j.lukkien@tue.nl
TU/e Informatica, System Architecture and Networking

Overview

- Physical system
- Specification aspects

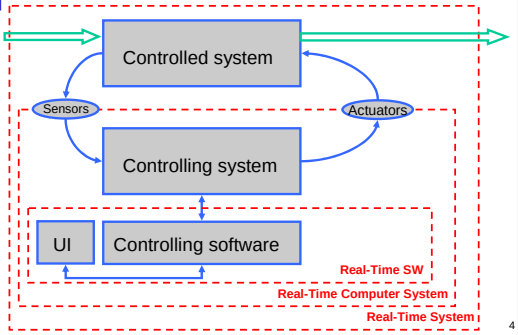
Johan J. Lukkien, j.j.lukkien@tue.nl
TU/e Informatica, System Architecture and Networking

Example: robots at conveyor belt



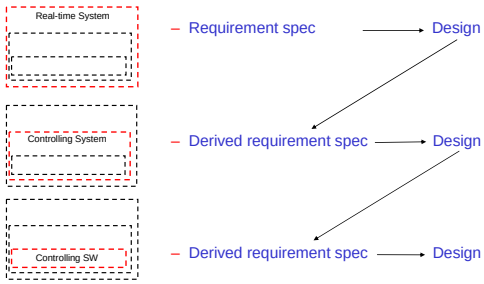
Johan J. Lukkien, j.j.lukkien@tue.nl
TU/e Informatica, System Architecture and Networking

RT(C)S components



Johan J. Lukkien, j.j.lukkien@tue.nl
TU/e Informatica, System Architecture and Networking

Top down refinement



Johan J. Lukkien, j.j.lukkien@tue.nl
TU/e Informatica, System Architecture and Networking

System components

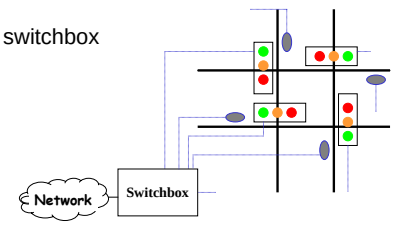
- Unintelligent part
  - Terminals: input/output media (not programmable by us)
    - physical, e.g. lights, sensors
  - Connections (plain wiring)
- Intelligent part
  - Devices (programmable)
  - Networks (digital media carrying layered messages)

System components (cnt'd)

- Input and output
  - Sensors & Actuators
    - simple input/output mechanisms
    - a/d conversion of external state
  - Input, output ports
    - data streams, usually to be operated upon
- Communication types
  - data
  - control .... real-time focus

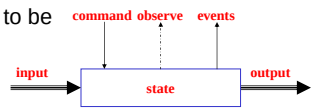
Example: traffic lights

- Terminals: traffic lights, sensing ovals
- Connections: wires
- Device: switchbox



Example (cnt'd)

- Terminals are modeled inside a device as a component
- Three interface elements for control
  - command
  - events
  - oberve (internal state)
- Control 'steers' the data streams
- Not all parts need to be there at any time
  - traffic light?
  - road sensor?



Johan J. Lukkien, j.j.lukkien@tue.nl
TU/e Informatica, System Architecture and Networking

Overview

- Physical system
- Specification aspects
  - Functional behaviour
  - Timeliness
  - Physical constraints and boundary conditions
  - Dependability
  - Evolution
- **NOTE:** specifications must be *total (..)*

Johan J. Lukkien, j.j.lukkien@tue.nl
TU/e Informatica, System Architecture and Networking

Functional behaviour

- Top level description of functionality of (controlled*) system.* E.g.,
  - The temperature must remain between 300K and 301K
  - The items that go on the conveyor belt must be painted
  - K items must be packed in each box
  - The traffic light system operates according to the rules
  - The video is displayed smoothly, with the highest possible quality
- *Basis for the relation between (sensor) input and (actuator) output*

Johan J. Lukkien, j.j.lukkien@tue.nl
TU/e Informatica, System Architecture and Networking

Timeliness

- Behavioural constraints of the *system* (no hard deadline)
  - "Lift doors close after 5 seconds"
  - "The test alarm sounds at 12 o'clock"
- Deadline constraints that are part of - or derivable from - the functional specification of the *system* (and translate into real-time constraints on the controlling system)
  - "The alarm sounds within 5 msec. after pressing the button"

Exercise

- Which of the following are behavioral constraints and which are deadline constraints:
  - The class starts at nine o'clock
  - Everybody should be present before five minutes to nine
  - After one hour we should stop for a break
  - The coffee should be ready before the break
  - The instructor should not talk more than 7 hours
  - The course finishes within 8 hours
  - Everybody wants to be home by five o'clock

Timeliness (cnt'd)

- *Derived* constraints on *controlling system* (from functional behaviour) are determined by design choices in the *system*
  - "Not more that 5% of the paint is wasted"
    - opening the faucet must be precise enough w.r.t. speed of conveyor belt:
  - "Variations in temperature are within 0.05K"
    - response must be quick enough w.r.t. possible variations in environment and size and thermal isolation of system
  - "K items must be packed in each box"
    - response must be quick enough w.r.t. transport speed of items to pick one
- System design decisions
  - speed of conveyor belt (following productivity requirements)
  - transport speed
  - size and thermal isolation

Physical constraints

- *Environment of the RTCS is given: the controlled system*
  - though it may be part of the design cycle
- Environment dictates part of the RTCS architecture
  - distribution, several distinct locations
  - concurrency
    - respond to *simultaneous* events
    - control *concurrent* physical processes
    - interconnect medium
  - embedding
    - limitations in resources: size, power, communication, memory
    - choice of processor or controller
    - software difficult to access

Dependability

- Availability:
  - probability of the system being ready to use
- Reliability:
  - expected time until not being available
- Safety:
  - catastrophic states not reachable
- Security:
  - protection of system assets
- Predictability
  - fixed and known system performance
  - ...under varying circumstances
- Maintainability: ease of repair, update etc.

Reliability and availability

- Robustness: system must be prepared
  - to work with common faults
  - to deal gracefully with less common faults
    - and then go back to operation quickly
  - .... fault tolerant
- Correctness: it *must*
  - work according to specification
  - for 99.999% of the time (can't afford to ctrl-alt-del)
- **Question**: can a system be
  - highly available but unreliable?
  - highly reliable but unavailable?

Notions in fault tolerance

- **Failure**: not meeting the specification
- **Error**: system state that may lead to failure
- **Fault**: cause of an error. Fault types:
  - transient: exists and then disappears
  - intermittent: repeatedly; disappears in between
  - permanent: remains until repair
- **Notes**:
  - faults can occur for all system components
    - processes, channels, ...
  - analysis starts with the development of a *fault model*
    - a set of assumptions describing nature and probability of faults

Failure Models

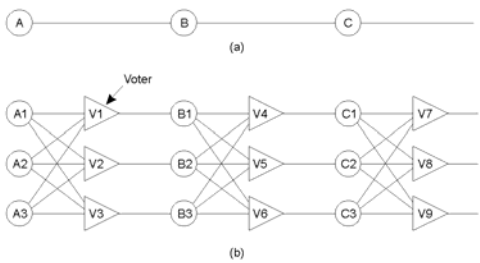
Type of failure	Example
Crash failure (fail-stop)	A server halts, but is working correctly until it halts
Omission failure *Receive omission* *Send omission*	A server fails to respond to incoming requests A server fails to receive incoming messages A server fails to send messages
Timing failure	A server's response lies outside the specified time interval
Response failure *Value failure* *State transition failure*	The server's response is incorrect The value of the response is wrong The server deviates from the correct flow of control
Arbitrary failure ("byzantine")	A server may produce arbitrary responses at arbitrary times

- **Note:** hard to distinguish some differences

Fault masking

- Redundancy is **the** method to mask faults or to recover from them
- Redundancy in
  - information: add extra bits
  - time: repeat failing operation (only useful for transient or intermittent faults)
  - physical redundancy: multiply hardware and/or software components
    - use voting

Example: triple modular redundancy



Safety, security

- The controlled system *must* remain safe
  - hazardous states unreachable (e.g., extremely high temperatures)
  - even in erroneous conditions, safety must be maintained (no “error exit”)
- Security: when the system is open to external observation and control (e.g., via Internet)
  - confidentiality, integrity and non-repudiation
    - validation of privileges (authentication)
    - secure protocols to make intrusion impossible

Predictability

- Essential ingredient for real-time systems
- The predictability is endangered by
  - anomalies of the hardware and system software
    - use of DMA that locks the bus
    - cache behaviour
    - interrupts
    - memory management
  - constraints
    - concurrency control: locking of resource
    - precedence constraints
  - absence of transparency in high level languages
    - dynamic variables, garbage collection (memory management)
    - repetition, recursion

Evolution

- Platform architecture
  - admit supporting a series of products
    - over time - improvement
    - at any time – products with varying features
  - scalability
    - grow larger in some dimensions
  - re-use
- Extensibility
  - support updates, compatibilities, new services ....
- Interoperability
  - co-operation with later products, products of other vendors

Don't forget

- Any form of correctness is always and only *with respect to assumptions about the environment*
  - which are generally a part of the specification

Exercises

- A certain control system is designed to be restarted after a certain internal state is observed. The time needed to restart is *r* seconds. The probability of the faulty condition is *k* times per day.
  - What is the reliability? (Assume a negative exponential distribution, if needed.)
  - What is the availability of this system?
    - first, give a rough estimate;
    - next, take into account the time that the system is off;
    - give two ways to improve the availability. What about reliability in these cases?

Exercise

- A piece of technical equipment has to remain in an exact temperature range given by an interval [*L*,*H*]. To that end a real-time system controls a heater and a cooler via a simple on/off mechanism.
  - draw a system sketch
    - indicate controlling and controlled system
    - explain the functionality of actuators and sensors you use
    - go through the requirements and discuss them for this example. Assume a maximum rate of temperature change of *r* degrees per second.
  - suppose that it is vital that this system is not overly heated or overly cooled.
    - describe possible faults (a fault model) and alternatives to improve the safety.

Exercise

- Do a similar exercise for
  - the traffic light example
  - an elevator system (check e.g. the Internet for samples)