

CPN Models as Enhancements to a Traditional Software Specification for an Elevator Controller

Jens Bæk Jørgensen

Department of Computer Science

University of Aarhus

Denmark

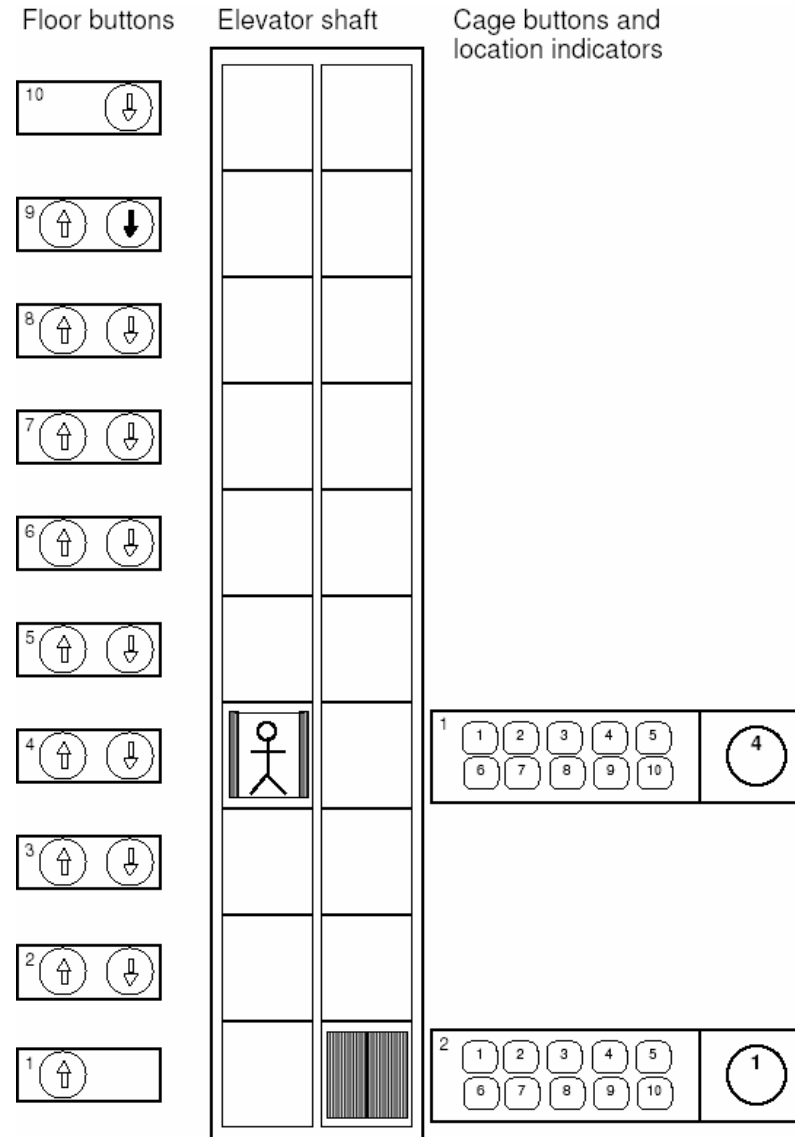
Problem: design of an elevator controller

⌘ Subject domain

- ☑ Ten floors
- ☑ Two cages
- ☑ Buttons, doors, sensors, ...

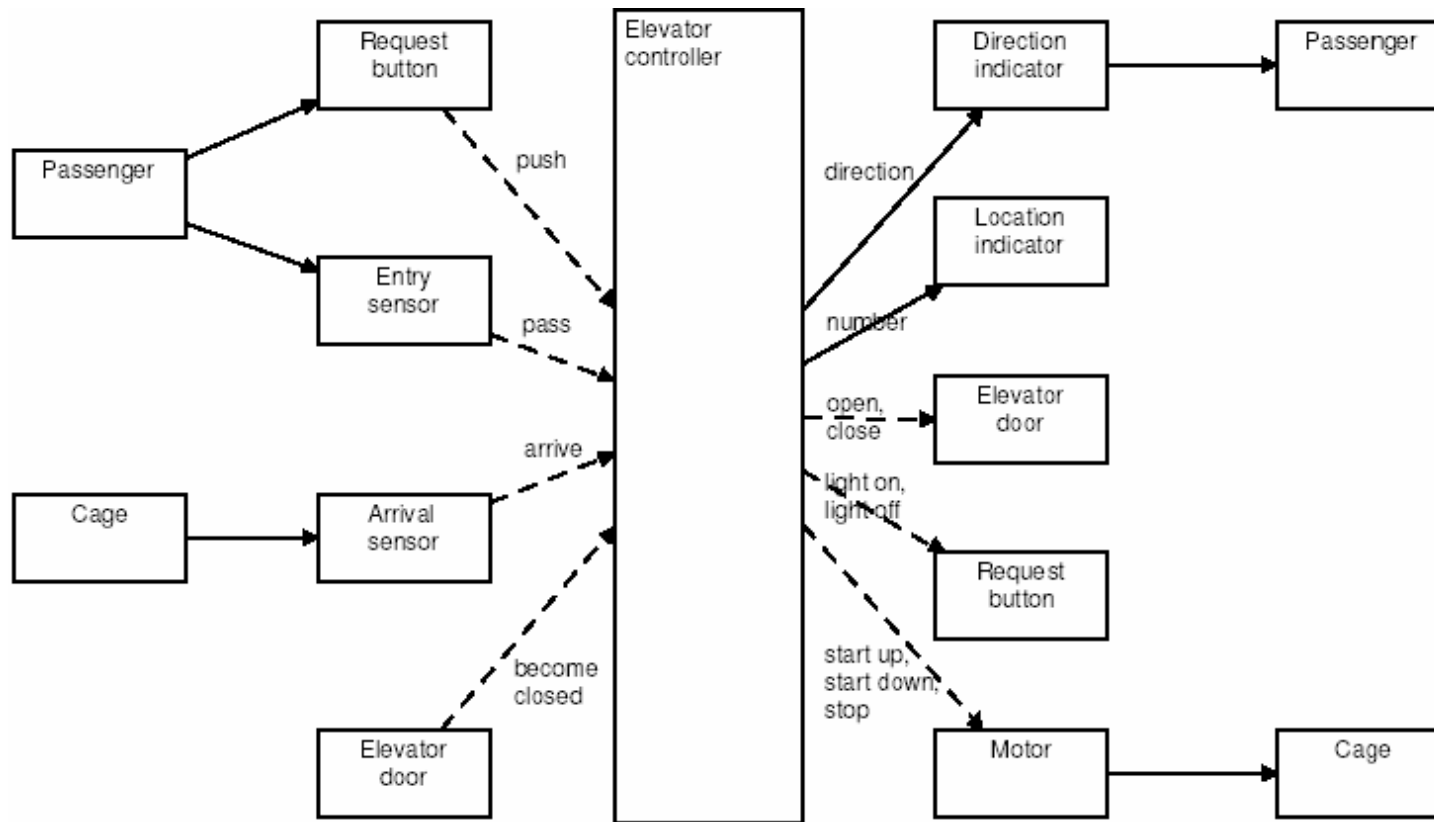
⌘ Controller responsibilities

- ☑ Control movement of cages
- ☑ Display information



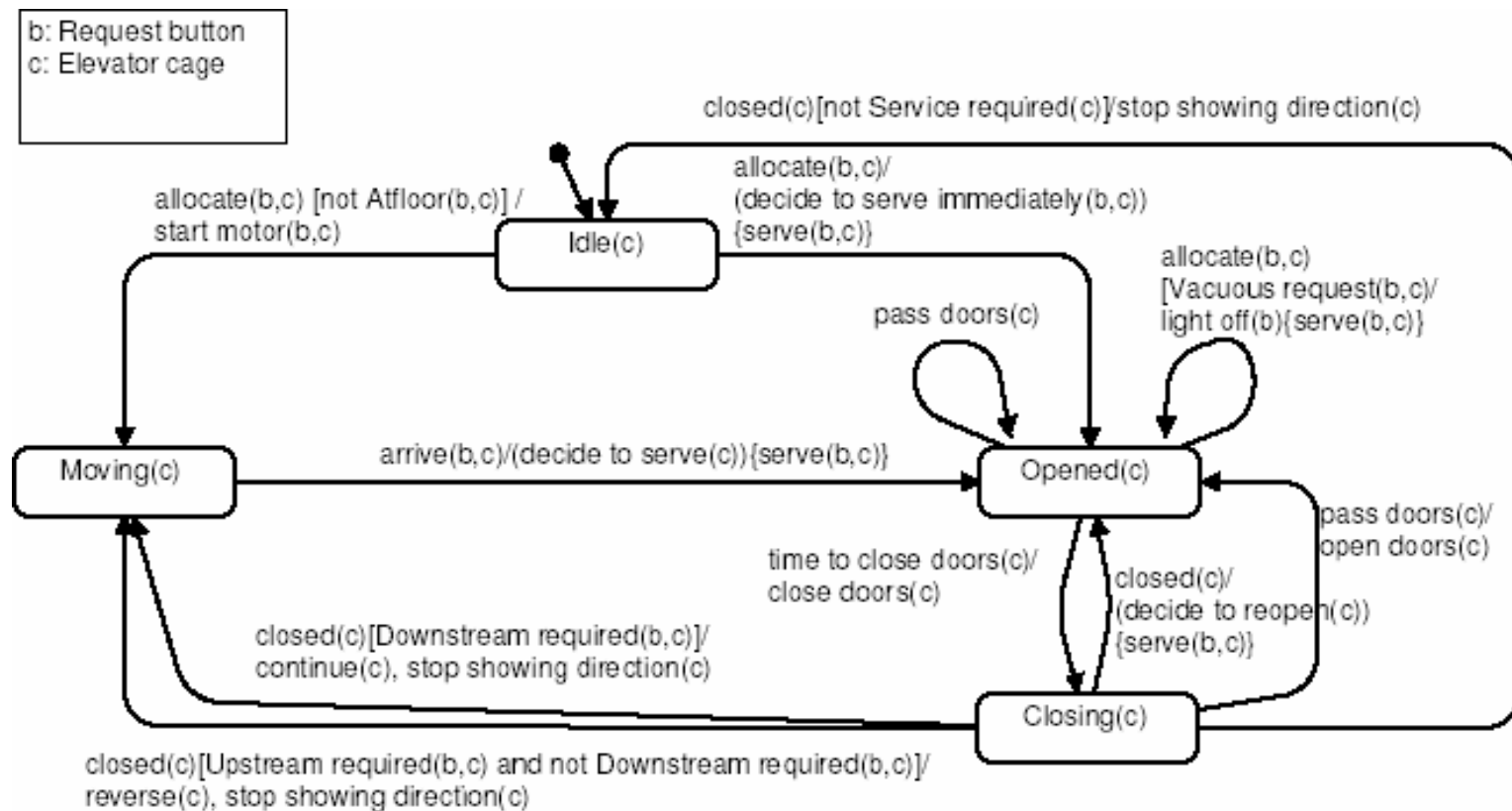
Wieringa's specification: desired functionality

- ⌘ Mission statement, function refinement tree, service descriptions
- ⌘ Partial context diagram



- ⌘ Also dictionary and descriptions of entities in subject domain

Wieringa's specification: desired behaviour of cage movement



Also descriptions of desired behaviour of allocation of request to cages, location indication, etc.

CPN model: representation of controller specification

⌘ Set of Standard ML functions

- ☒ setdirection
- ☒ stophere
- ☒ turnidle
- ☒ servenow
- ☒ resetdirection
- ☒ addrequest
- ☒ removerequest
- ☒ updatelocationindicators

CPN model: requirements-level architecture

⌘ Representation of controller

- ☒ Processes ~ Standard ML functions
- ☒ Data stores ~ tokens

⌘ Representation of subject domain

- ☒ Entities ~ tokens

⌘ Communications

- ☒ Possible internal communications in system ~ transitions
- ☒ Possible communications between controller and entities in environment ~ transitions

CPN model: basis for system engineering argument

- ⌘ Argue that specification and domain properties together entail requirements

- ⌘ Prerequisites for argument

 - ☑ CPN model executable

 - ☑ CPN model coherent

- ⌘ Example requirement: Collect passengers

 - ☑ Trigger: Passenger pushes floor button F

 - ☑ Delivered service: Controller ensures that cage stops at floor F and allows passengers to enter

Some perspectives on CPN in software engineering

⌘ Compliance with Jackson's basic tenets

- ☑ Distinguish the machine from the problem domain
- ☑ Don't restrict description to the machine
- ☑ State explicitly what is described

⌘ Advantages compared with statecharts

- ☑ CPN adequate to address scheduling
- ☑ CPN conveniently describe two cages together
- ☑ CPN facilitate prototyping and experiments

Conclusions and discussion

⌘ Advantages of adding CPN model

- ☑ Can be used as requirements-level architecture
- ☑ Facilitates system engineering argument

⌘ Cost-benefit issues of adding CPN model

- ☑ Gap between model and implementation
- ☑ Can existing specification be improved with simpler means?

⌘ Formal verification viable?

- ☑ Improve quality of system engineering argument
- ☑ Argue about more advanced behavioural properties
- ☑ Scalability problems