Embedded Systems Design

Course Plan

- Intro to ES, V-model and Model-based Design
- *Project intro*
- Requirements and Functional Testing
- *Project step1 – Requirements and test plan*
- Models and systems
- FSMs part1
- Communication buses – event driven - Controller Area Network
- *Project step2 – Intro to Flex, programming IDE, Erika, Device drivers*
- FSMs part2
- Communication buses – time driven – FlexRay
- FSMs part3
- Extensions to FSMs – Statecharts
- *Project step3 – Models*
- Timed automata
- An Introduction to Uppaal and verification
- *Project step4 – Verification*
- Translating an FSM into code: Part 1
- Statecharts in practice – Mathworks Stateflow
- Translating an FSM into code: Part 2
- Automatic code generation with Mathworks tools
- *Project step5 – Production of Code*
- Conformance testing
- Structural testing – coverage
- *Project step6 – Testing*
- Advanced topic: semantics preservation in multiprogramming implementations
- Advanced topic: component-based design and AUTOSAR

Course organization and evaluation

- Evaluation is based on
    - Project: development of simple application on Flex board using model-based derivation techniques. Teams of 4. Each member of the team must be capable of discussing the project steps. 40%
    - Discussion of a research paper. Individual, randomly selected paper 20%
    - Final test. 40%
- Available Projects
    - Elevator controller mockup (all)
    - Elevator subsystem (pick one)
    - Interface prototyping (Linux with Qt or Microchip)

Embedded systems

- An **embedded system** is a special-purpose computer system designed to perform one or a few dedicated functions, sometimes with real-time computing constraints. It is usually *embedded* as part of a complete device including hardware and mechanical parts. In contrast, a general-purpose computer, such as a personal computer, can do many different tasks depending on programming.

- Since the embedded system is dedicated to specific tasks, design engineers can optimize it, reducing the size and cost of the product, or increasing the reliability and performance.

Embedded systems

- **Embedded = Dedicated**
- Interaction with physical processes
  - sensors, actuators, processes
- Critical properties are not all functional
  - real-time, fault recovery, power, security, robustness
- Heterogeneity
  - hardware/software tradeoffs, mixed
- Concurrency
  - interaction with multiple processes
- Reactivity
  - operating at the speed of the environment
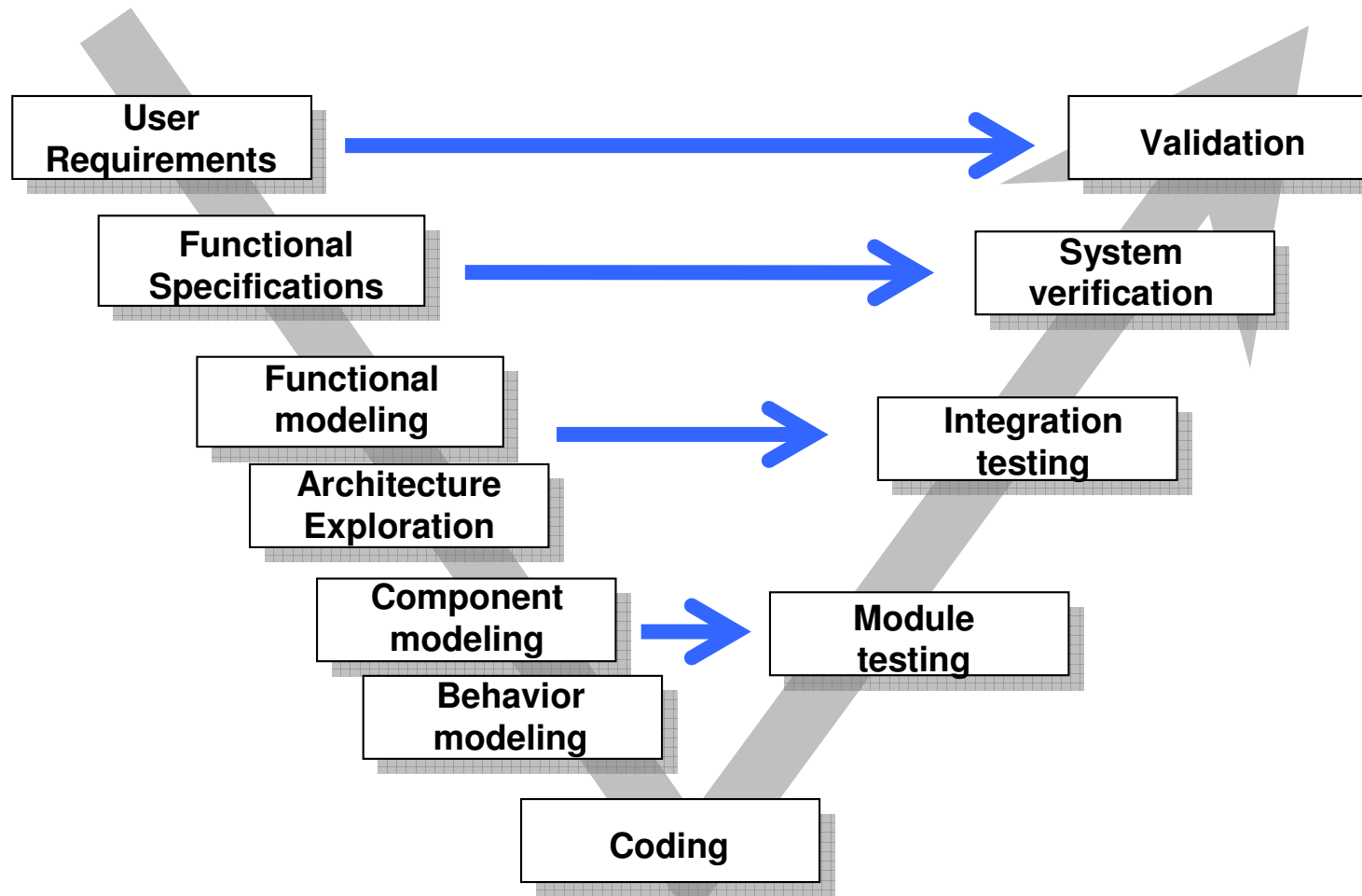- Resource constrained
  - Because of cost, energy, space

Timing constraints (latency, jitter)

Heterogeneous models, methods, tools
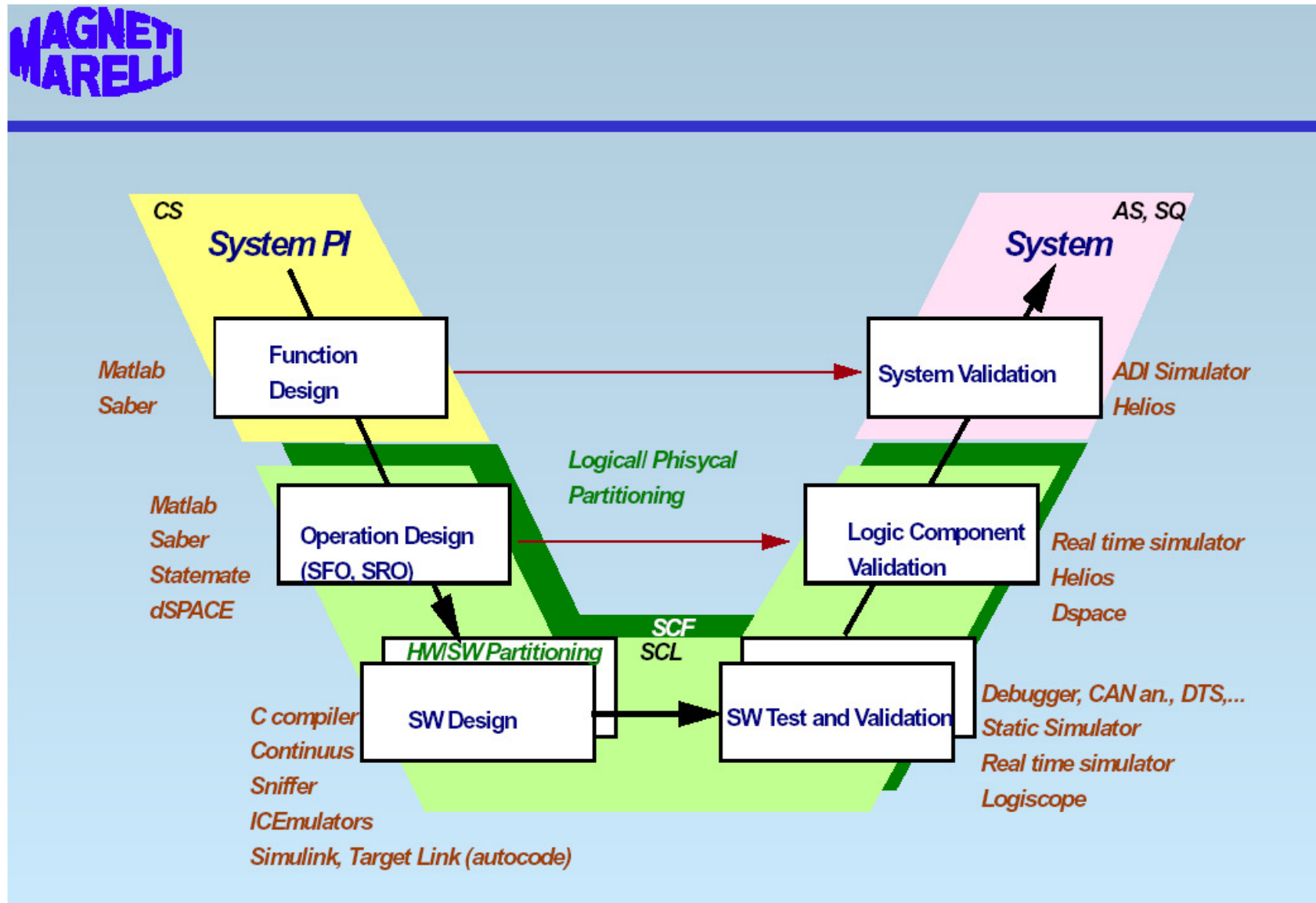
Resource sharing, scheduling
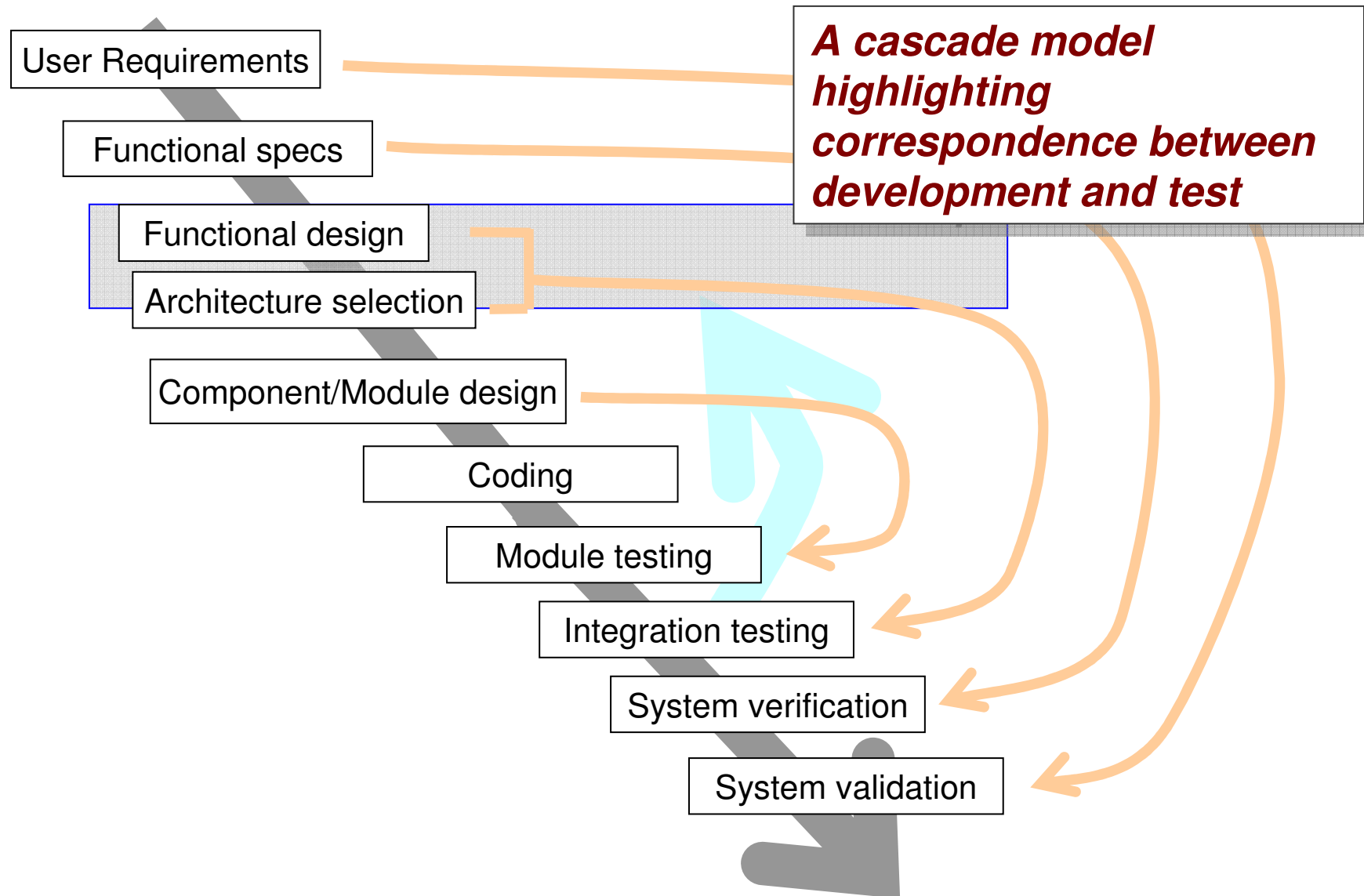
(source Edward Lee – UC Berkeley)

The V-shape development cycle (V-model)

User
Requirements → Validation

Functional
Specifications → System
verification

Functional
modeling → Integration
testing

Architecture
Exploration

Component
modeling → Module
testing

Behavior
modeling

Coding

A development cycle

What's special in the V-model?

User Requirements

Functional specs

Functional design

Architecture selection

Component/Module design

Coding

Module testing

Integration testing

System verification

System validation

*A cascade model highlighting correspondence between development and test*

Model-based design

On August 19, 1418, a competition was announced in Florence, where the city's magnificent new cathedral, Santa Maria del Fiore, had been under construction for more than a century

Whoever desires to make any model or design for the vaulting of the main Dome of the Cathedral under construction by the Opera del Duomo-for armature, scaffolding or other thing, or any lifting device pertaining to the construction and perfection of said cupola or vault shall do so before the end of the month of September. If the model be used he shall be entitled to a payment of 200 gold Florins.

From Brunelleschi's Dome:
How a Renaissance Genius
Reinvented Architecture
by Ross King

Competition between architects was an old and honored custom. Patrons had been making architects compete against one another for their commissions since at least 448 B.C., when the Council of Athens held a public competition for the war memorial it planned to build on the Acropolis. Under these circumstances, it was normal practice for architects to produce models as a means of convincing patrons or panels of judges of the virtues of their particular designs.
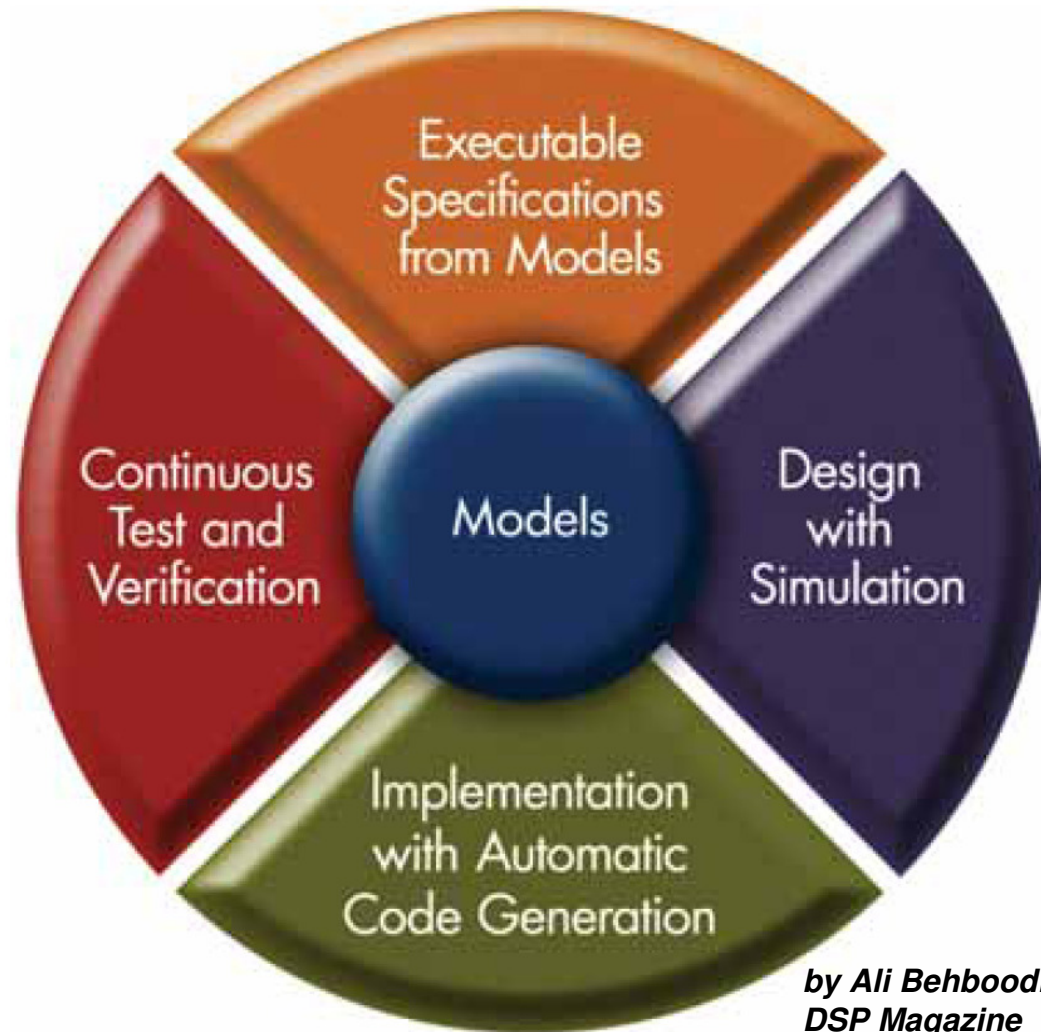
Model-based design
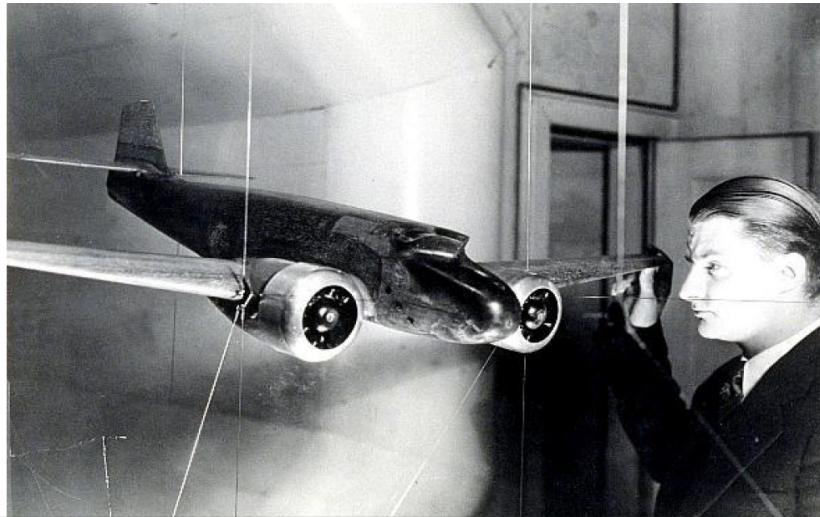
Engineering has made use of models since its very early days

Filippo Brunelleschi's design for the dome of the cathedral of Santa Maria del Fiore in Florence remains one of the most towering achievements of Renaissance architecture. Completed in 1436, the dome remains a remarkable feat of design and engineering. Its span of more than 140 feet exceeds St Paul's in London and St Peter's in Rome, and even outdoes the Capitol in Washington, D.C., making it the largest dome ever constructed using bricks and mortar. When work on the dome began in 1420 Brunelleschi was virtually unknown. Sixteen years later the dome was built, and its architect was a superstar.

Model-based design

- The four tenets on the right are fundamental to model-based design
- ***Key is: test, verification, simulation, validation***
- ***Why automatic code generation?***
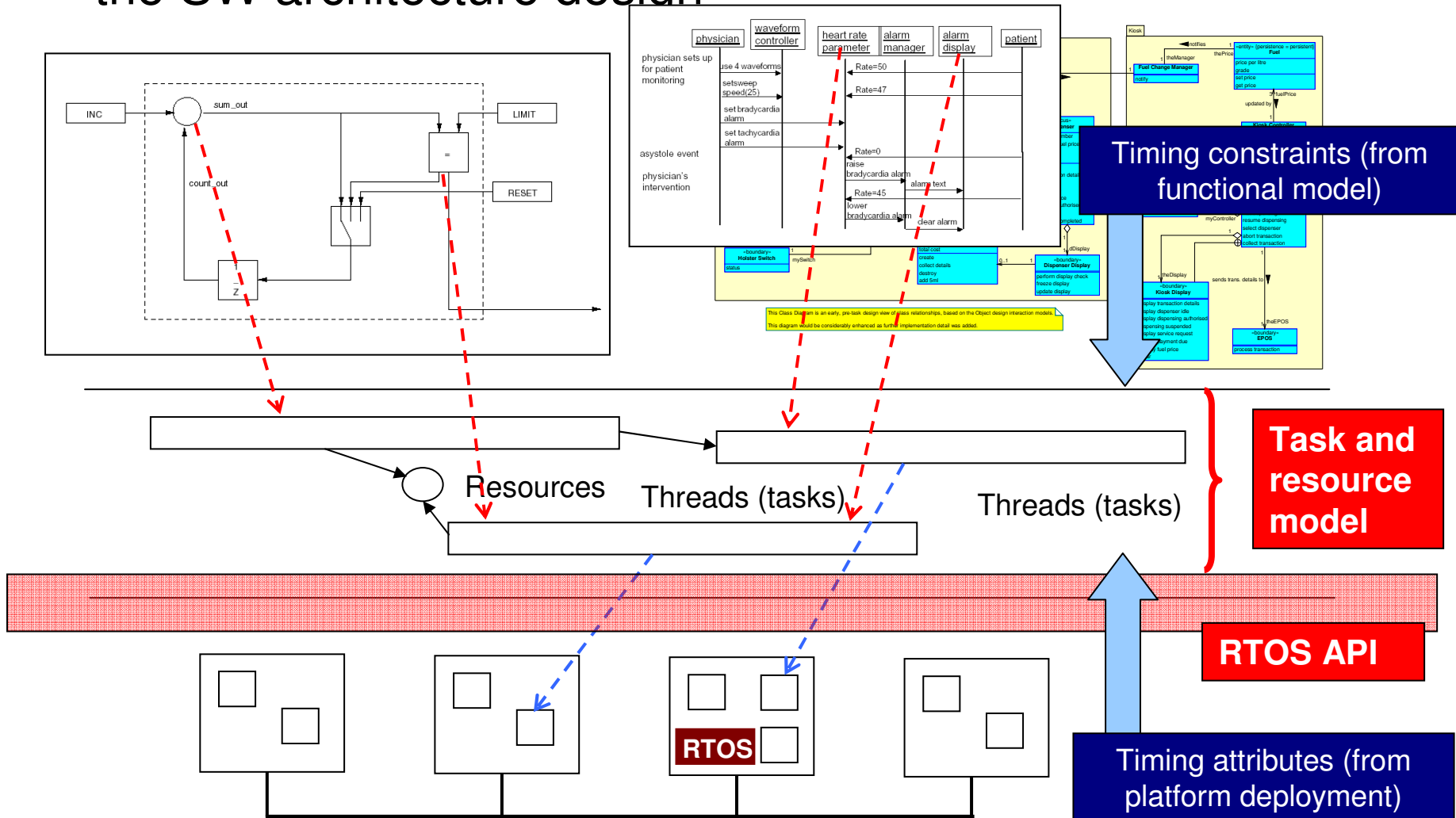- Of course, you must select a modeling language that allows to do everything in the most natural and easy way …

by Ali Behboodian, DSP Magazine

Figure 1 – Elements of model-based design

RTS and Platform-Based Design

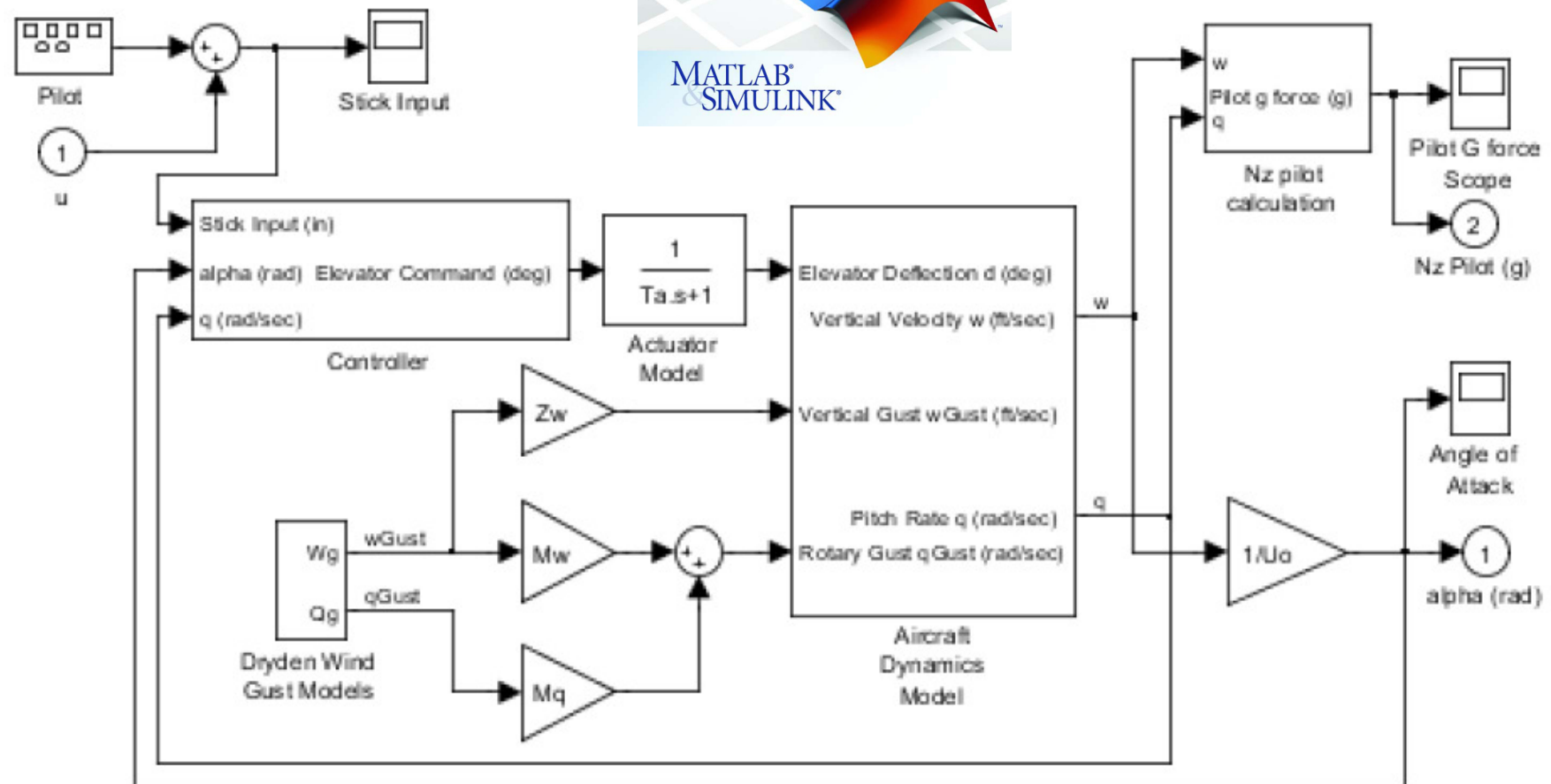- Design (continued): matching the logical design into the SW architecture design

Timing constraints (from functional model)

Task and resource model

Resources

Threads (tasks)

Threads (tasks)

RTOS API

RTOS

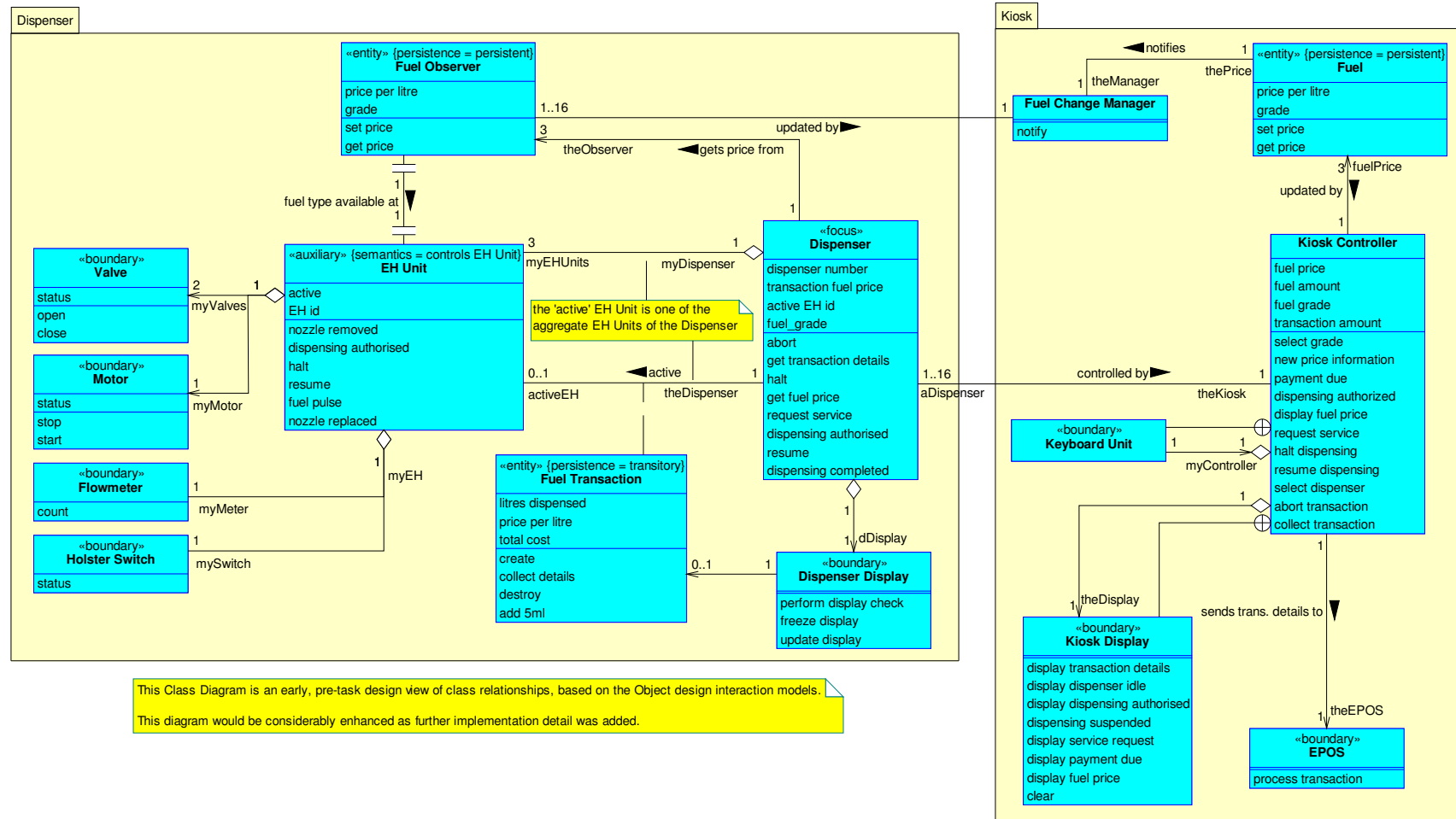Timing attributes (from platform deployment)

End of introduction

- Questions/Issues?

Models and implementation: Simulink

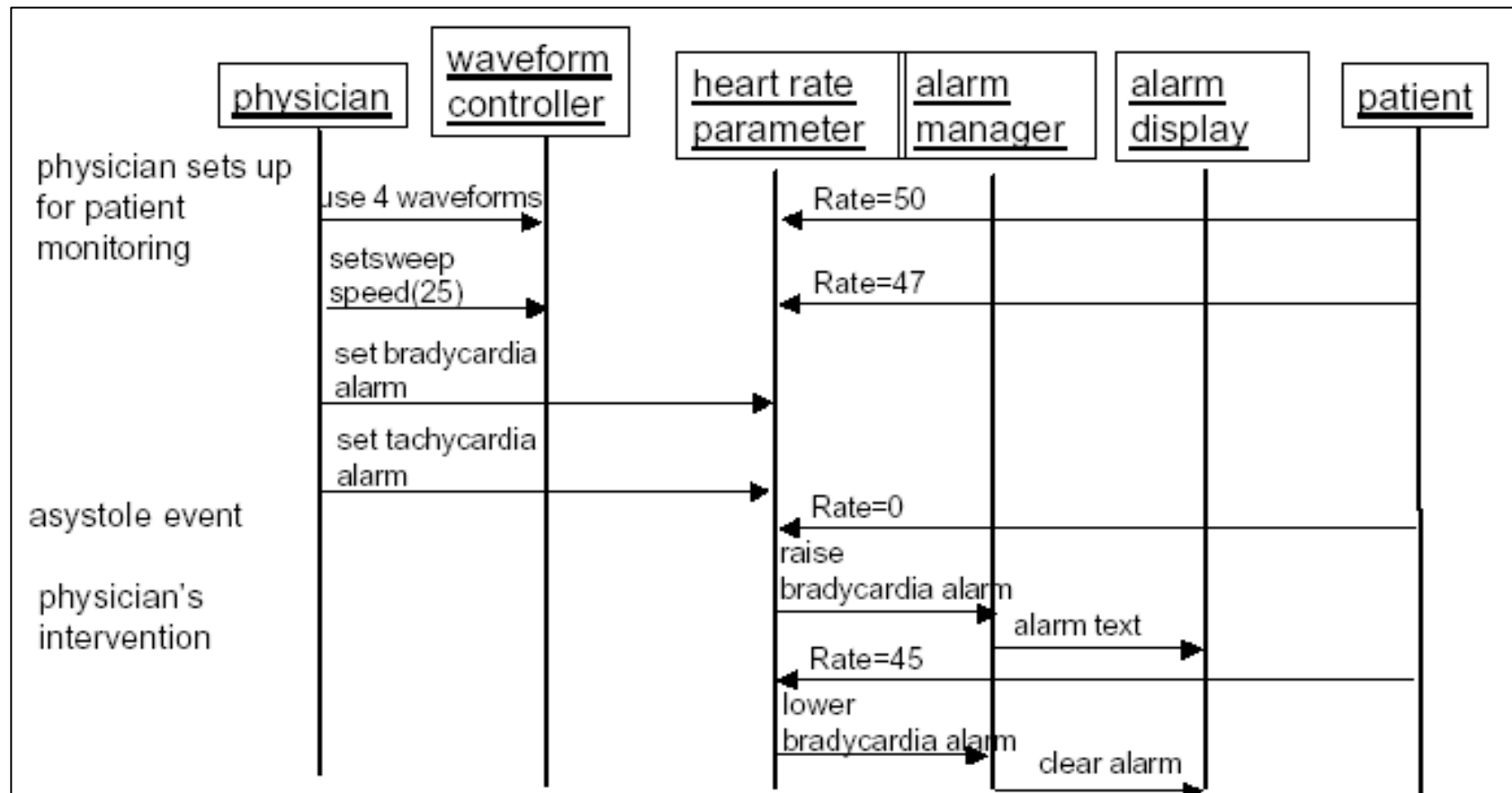**Where are the tasks?**

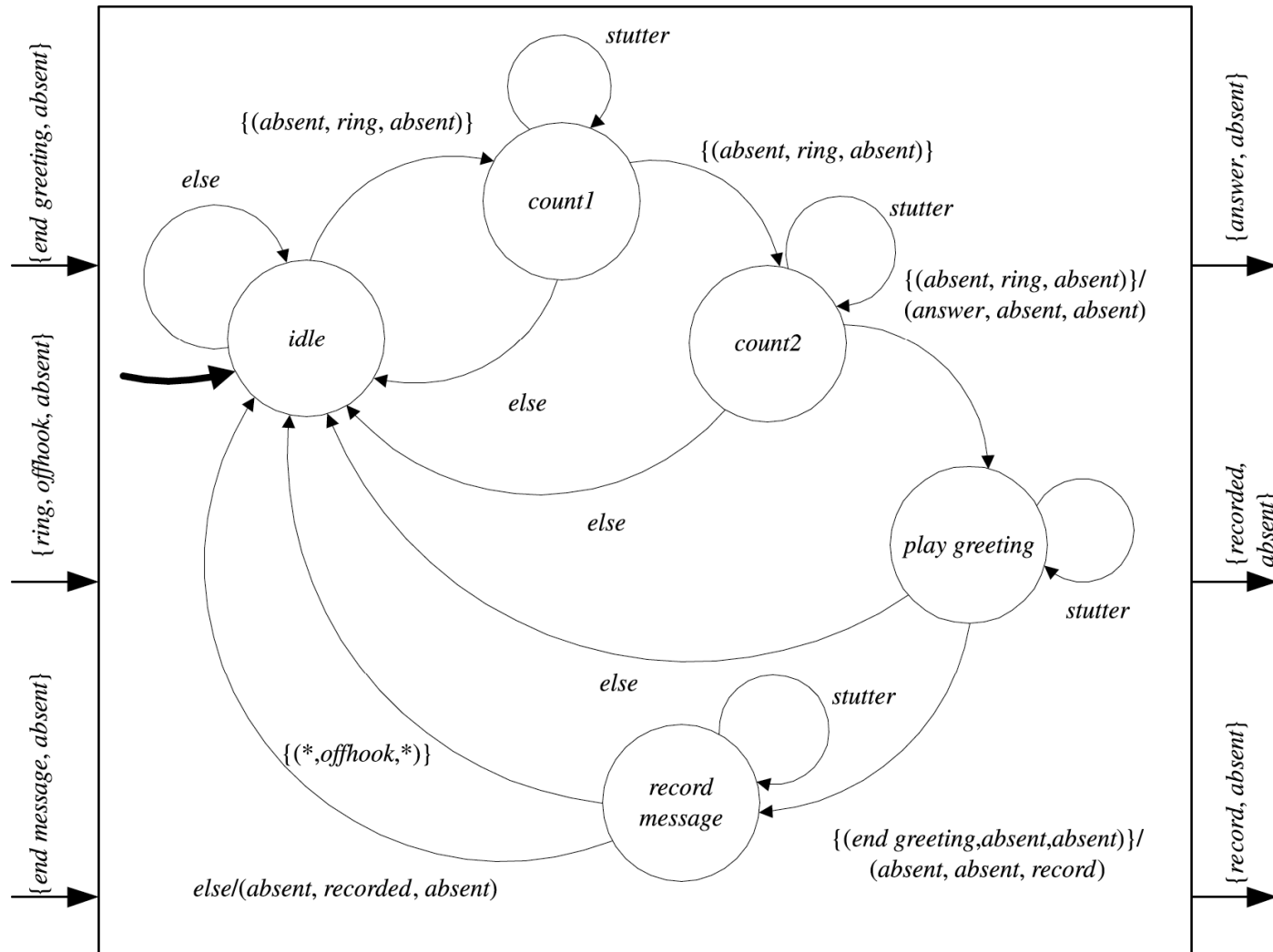# Models and implementation: UML

Where are the tasks?

Models and implementation: UML

Models and implementation: FSM

NOTE: *stutter* = {(*absent, absent, absent*)}

Some things are there … maybe not in plain sight