COMP1531

🐶Software Engineering
8.1 - Design - System Modelling

In this lecture

Why?

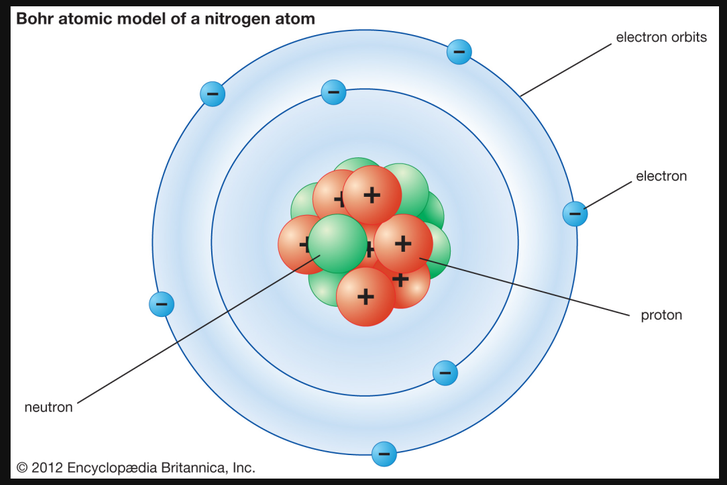
- A critical element of software design is to be able to translate complex system ideas into something high level and understandable

What?

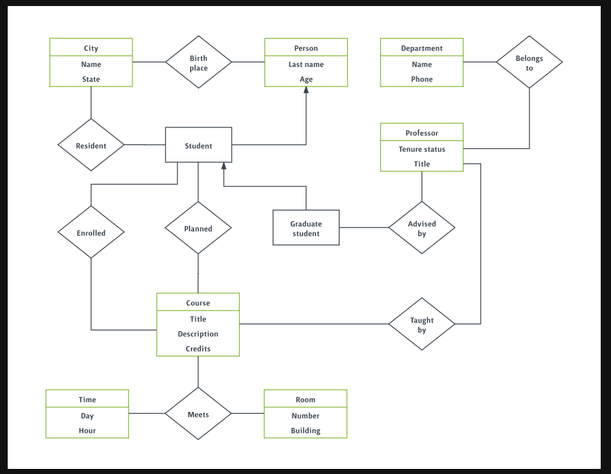
- Conceptual Model
- State Diagrams

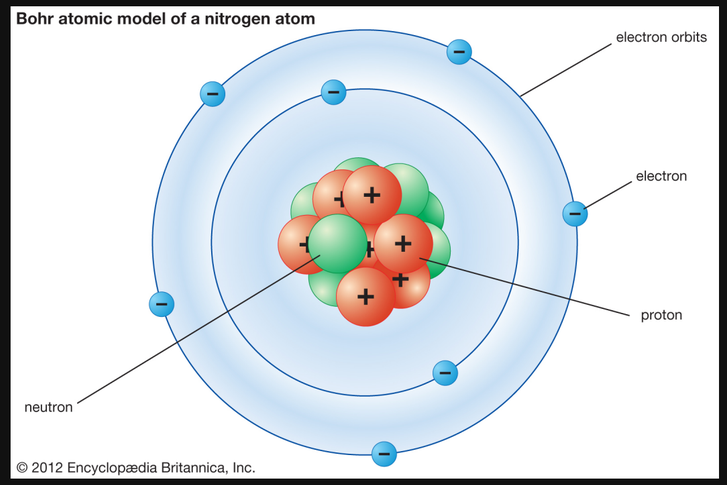
What's a model?

What's a model?

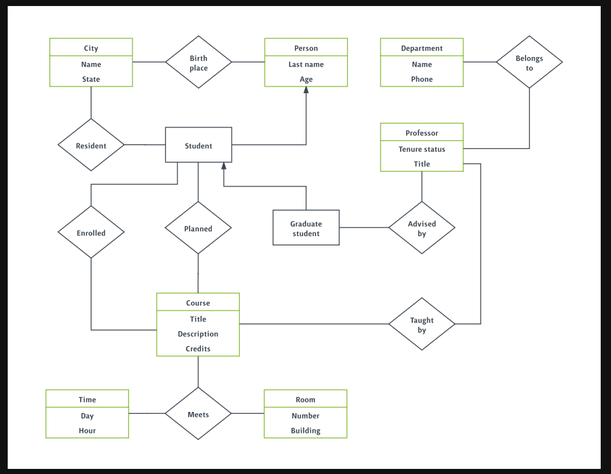


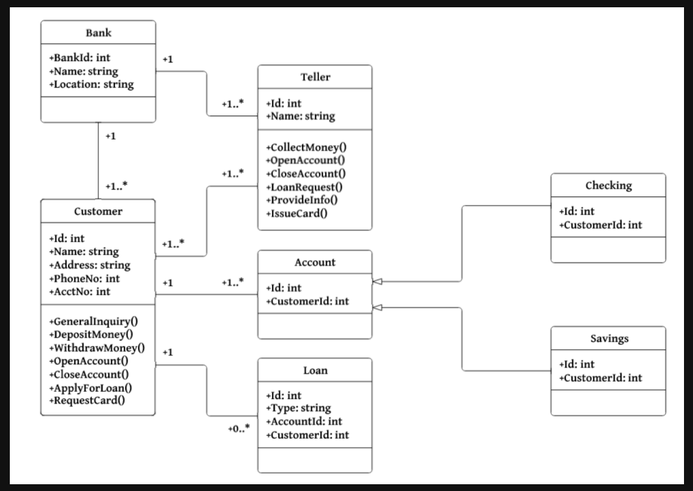
What's a model?

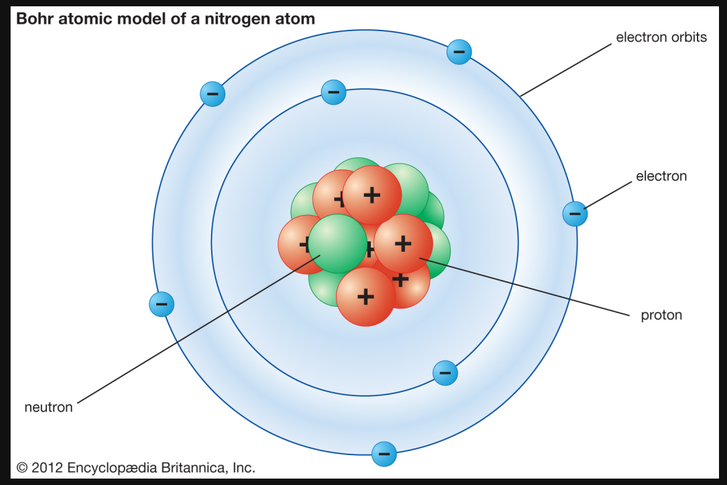




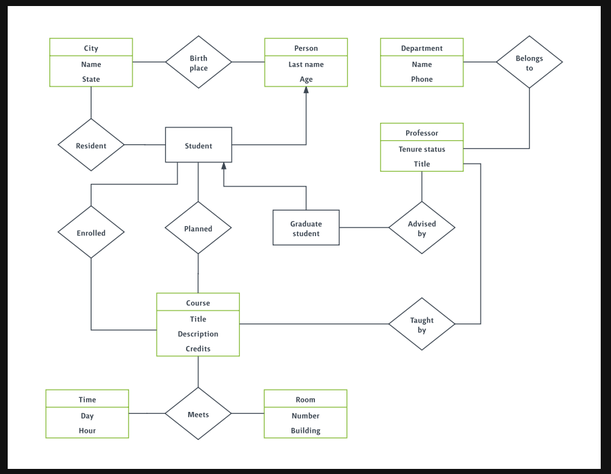
What's a model?

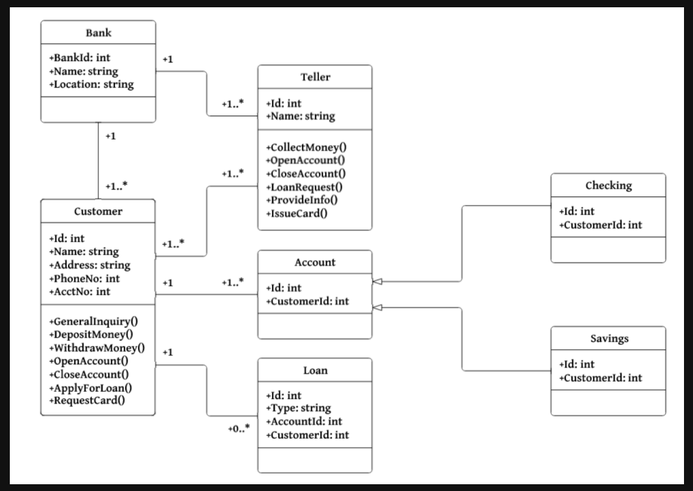


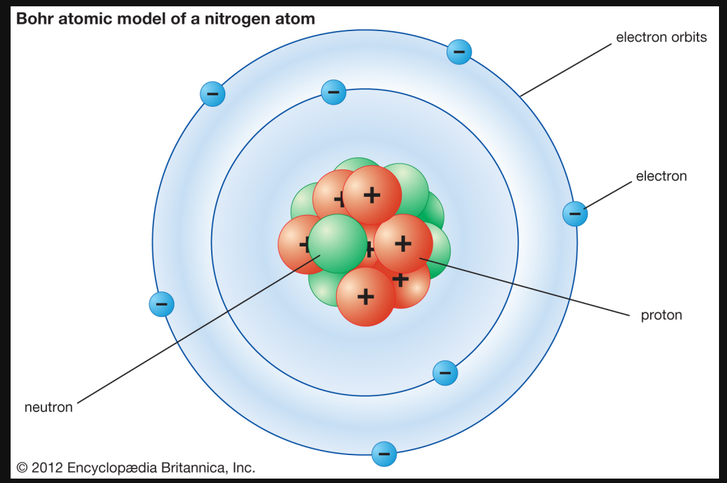


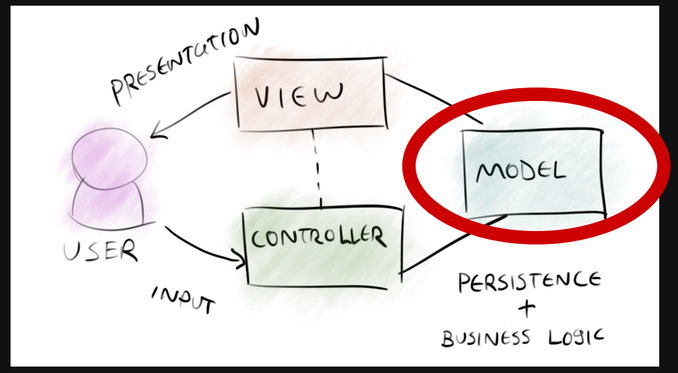


What's a model?









What's a model?

Figure 1: Mathematical Model

$$\frac{dC}{dt} = \frac{s_c C}{1+(R_b+N_b+T_b+L_b)/b_{c\infty}} + \frac{k_{cp}}{v_m}(P-C) - (k_{crc}+k_{cnc}+k_{ctc}+k_{clc})C - \mu_c C$$

$$\frac{dP}{dt} = \frac{k_{cp}}{v_b}(C-P) - \mu_p P$$

$$\frac{dR_c}{dt} = \frac{s_{rc}R_c}{1+R_b/b_{r\infty}} + k_{crc}C - k_{rcm}R_c - \mu_{rc}R_c \qquad \frac{dR_m}{dt} = k_{rcm}R_c - \frac{k_{rmb}}{v_m}R_m - \mu_{rm}R_m$$

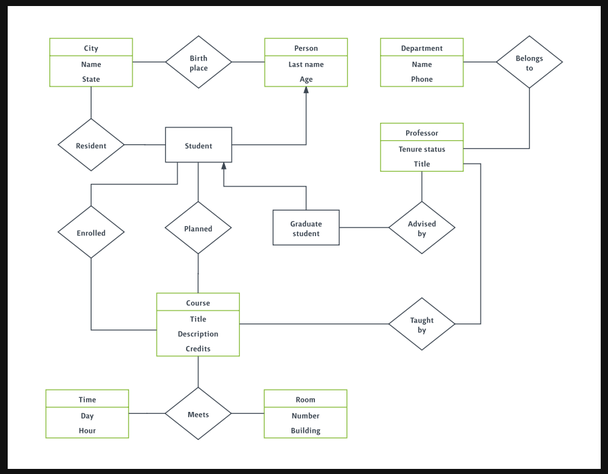
$$\frac{dN_c}{dt} = \frac{s_{nc}N_c}{1+N_b/b_{n\infty}} + k_{cnc}C - k_{ncm}N_c - \mu_{nc}N_c \qquad \frac{dN_m}{dt} = k_{ncm}R_c - \frac{k_{nmb}}{v_m}N_m - \mu_{nm}N_m$$

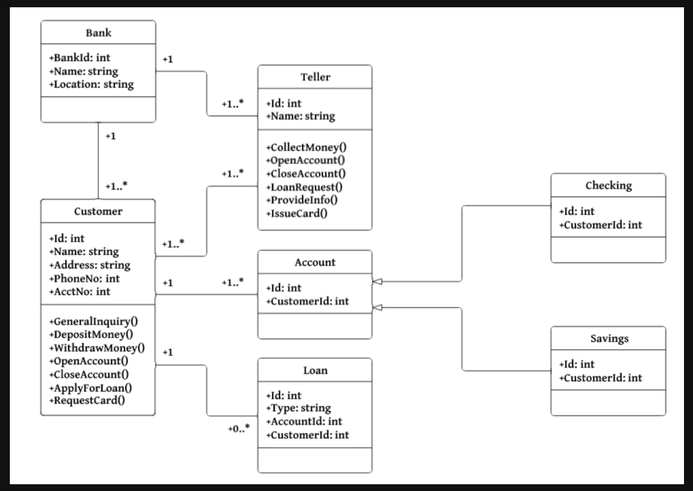
$$\frac{dT_c}{dt} = \frac{s_{tc}T_c}{1+T_b/b_{t\infty}} + k_{ctc}C - k_{tcm}T_c - \mu_{tc}T_c \qquad \frac{dT_c}{dt} = k_{tcm}T_c - \frac{k_{tmb}}{v_m}T_c - \mu_{tc}T_c$$

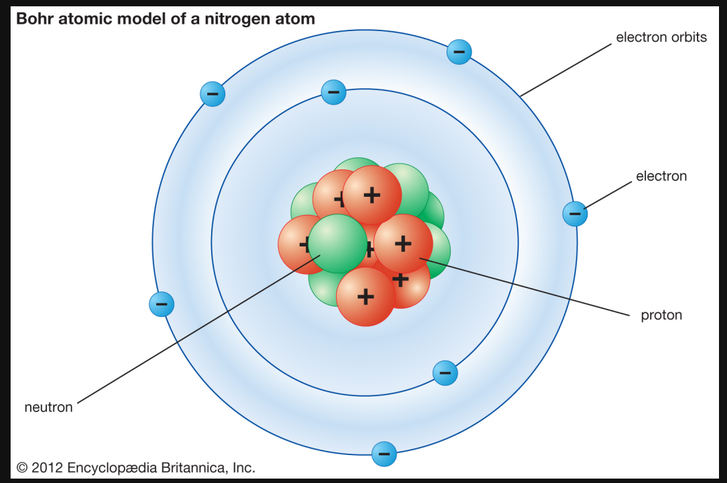
$$\frac{dL_c}{dt} = \frac{s_{lc}L_c}{1+L_b/b_{l\infty}} + k_{clc}C - k_{lcm}L_c - \mu_{lc}L_c \qquad \frac{dL_c}{dt} = k_{lcm}L_c - \frac{k_{lmb}}{v_m}L_c - \mu_{lc}L_c$$

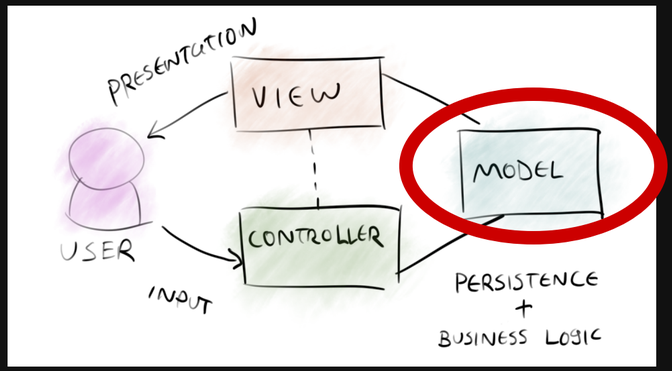
$$\frac{dR_b}{dt} = \frac{k_{rmb}}{v_m}R_m - \mu_{rb}R_b \qquad \frac{dT_b}{dt} = \frac{k_{tmb}}{v_m}T_m - \mu_{tb}T_b$$

$$\frac{dN_b}{dt} = \frac{k_{nmb}}{v_m}N_m - \mu_{nb}N_b \qquad \frac{dL_b}{dt} = \frac{k_{lmb}}{v_m}L_m - \mu_{lb}L_b$$









What's a model?

Figure 1: Mathematical Model

$$\frac{dC}{dt} = \frac{s_c C}{1+(R_b+N_b+T_b+L_b)/b_{c\infty}} + \frac{k_{cp}}{v_m}(P-C) - (k_{crc}+k_{cnc}+k_{ctc}+k_{clc})C - \mu_c C$$

$$\frac{dP}{dt} = \frac{k_{cp}}{v_b}(C-P) - \mu_p P$$

$$\frac{dR_c}{dt} = \frac{s_{rc}R_c}{1+R_b/b_{r\infty}} + k_{crc}C - k_{rcm}R_c - \mu_{rc}R_c \qquad \frac{dR_m}{dt} = k_{rcm}R_c - \frac{k_{rmb}}{v_m}R_m - \mu_{rm}R_m$$

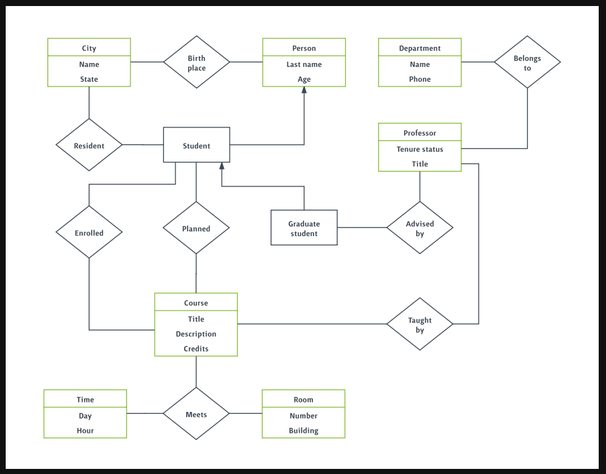
$$\frac{dN_c}{dt} = \frac{s_{nc}N_c}{1+N_b/b_{n\infty}} + k_{cnc}C - k_{ncm}N_c - \mu_{nc}N_c \qquad \frac{dN_m}{dt} = k_{ncm}R_c - \frac{k_{nmb}}{v_m}N_m - \mu_{nm}N_m$$

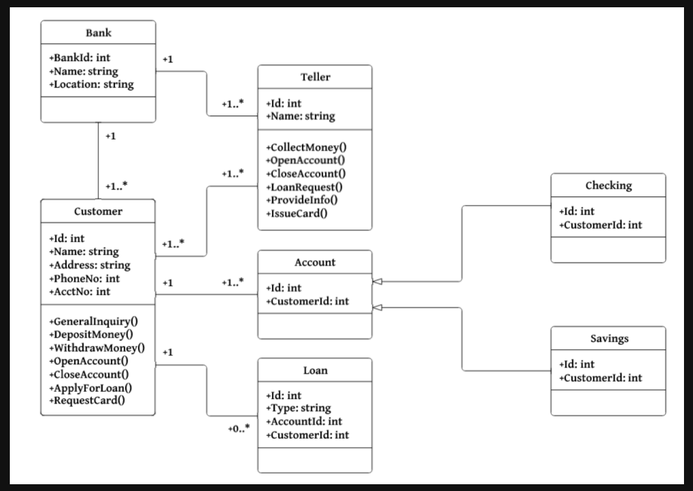
$$\frac{dT_c}{dt} = \frac{s_{tc}T_c}{1+T_b/b_{t\infty}} + k_{ctc}C - k_{tcm}T_c - \mu_{tc}T_c \qquad \frac{dT_c}{dt} = k_{tcm}T_c - \frac{k_{tmb}}{v_m}T_c - \mu_{tc}T_c$$

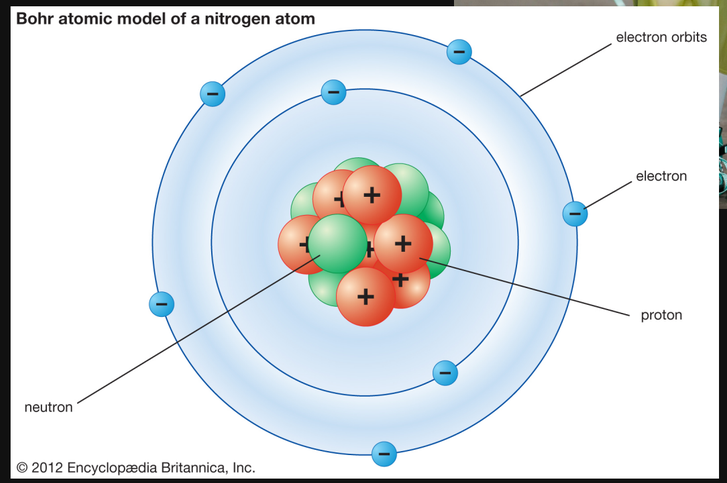
$$\frac{dL_c}{dt} = \frac{s_{lc}L_c}{1+L_b/b_{l\infty}} + k_{clc}C - k_{lcm}L_c - \mu_{lc}L_c \qquad \frac{dL_c}{dt} = k_{lcm}L_c - \frac{k_{lmb}}{v_m}L_c - \mu_{lc}L_c$$

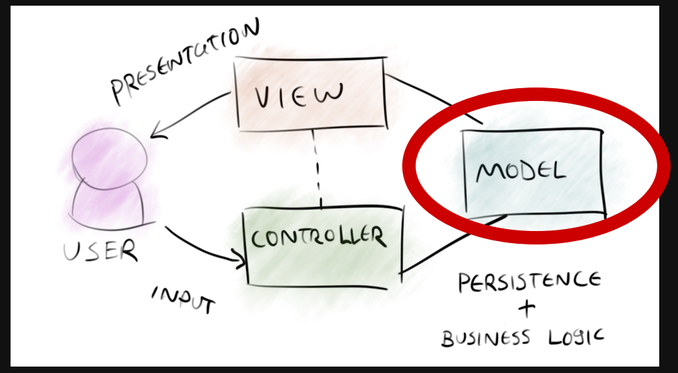
$$\frac{dR_b}{dt} = \frac{k_{rmb}}{v_m}R_m - \mu_{rb}R_b \qquad \frac{dT_b}{dt} = \frac{k_{tmb}}{v_m}T_m - \mu_{tb}T_b$$

$$\frac{dN_b}{dt} = \frac{k_{nmb}}{v_m}N_m - \mu_{nb}N_b \qquad \frac{dL_b}{dt} = \frac{k_{lmb}}{v_m}L_m - \mu_{lb}L_b$$









Conceptual Modelling

- A model is an attempt to represent a more complex system
- A conceptual model captures a system in a conceptual way
  - High level abstraction
  - Tends to be diagramatic or visual

Conceptual models software engineers care about

- Data models
- Mathematical models
- Domain models
- Data flow models
- State transition models (today)

How models are used

- To predict future states of affairs.
- Understand the current state of affairs.
- Determine the past state of affairs.
- **To convey the fundamental principles and basic functionality of systems (communication)**

Communicating models

- Four fundamental objectives of communicating with a conceptual model:
  1. Enhance an individual's understanding of the representative system
  2. Facilitate efficient conveyance of system details between stakeholders
  3. Provide a point of reference for system designers to extract system specifications
  4. Document the system for future reference and provide a means for collaboration

Kung and Solvberg (1986)

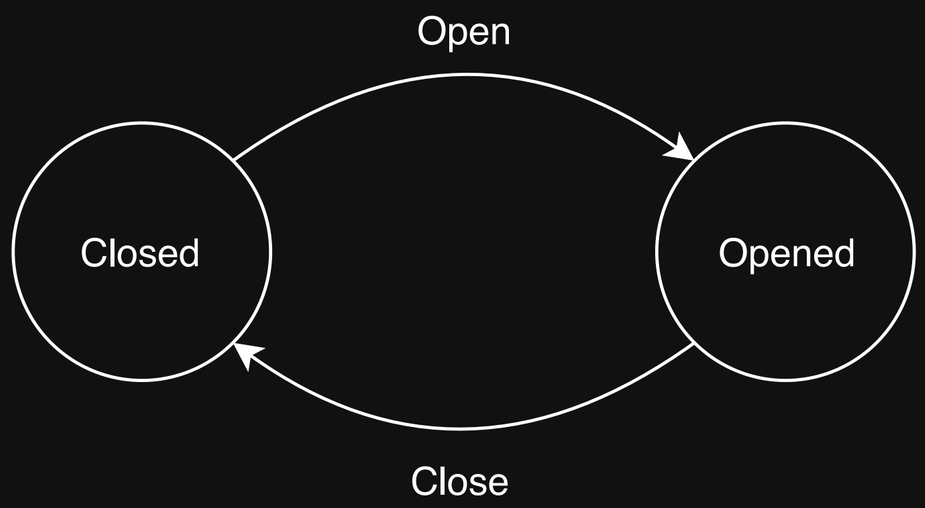
Conceptual Models

- Structural – Emphasise the static structure of the system
  - UML class diagrams (object oriented programming)
  - ER diagrams (database design)
  - … many others
- Behavioural - Emphasise the dynamic behaviour
  - State diagrams (state machines)
  - Use case diagram (user flows)
  - … some others

We will discuss some of these, and explore **state diagrams** in detail.

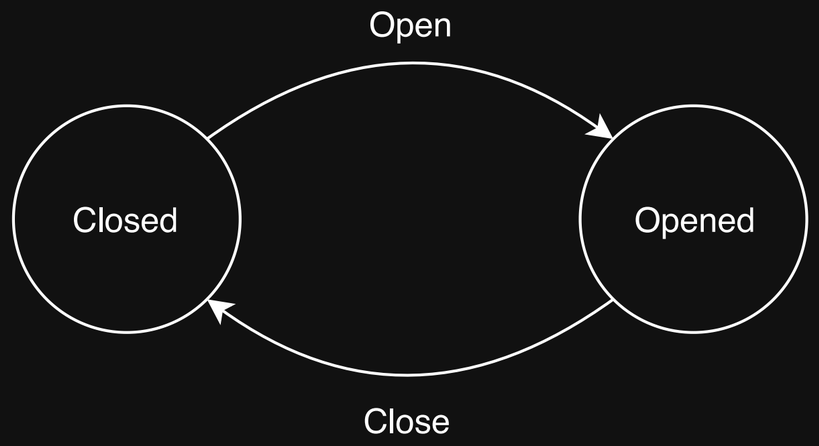
State Diagrams

- State machines made up of a finite number of states.
- The machine can be *transitioned* from one state to another through an action
- Simple example: a door



State diagrams

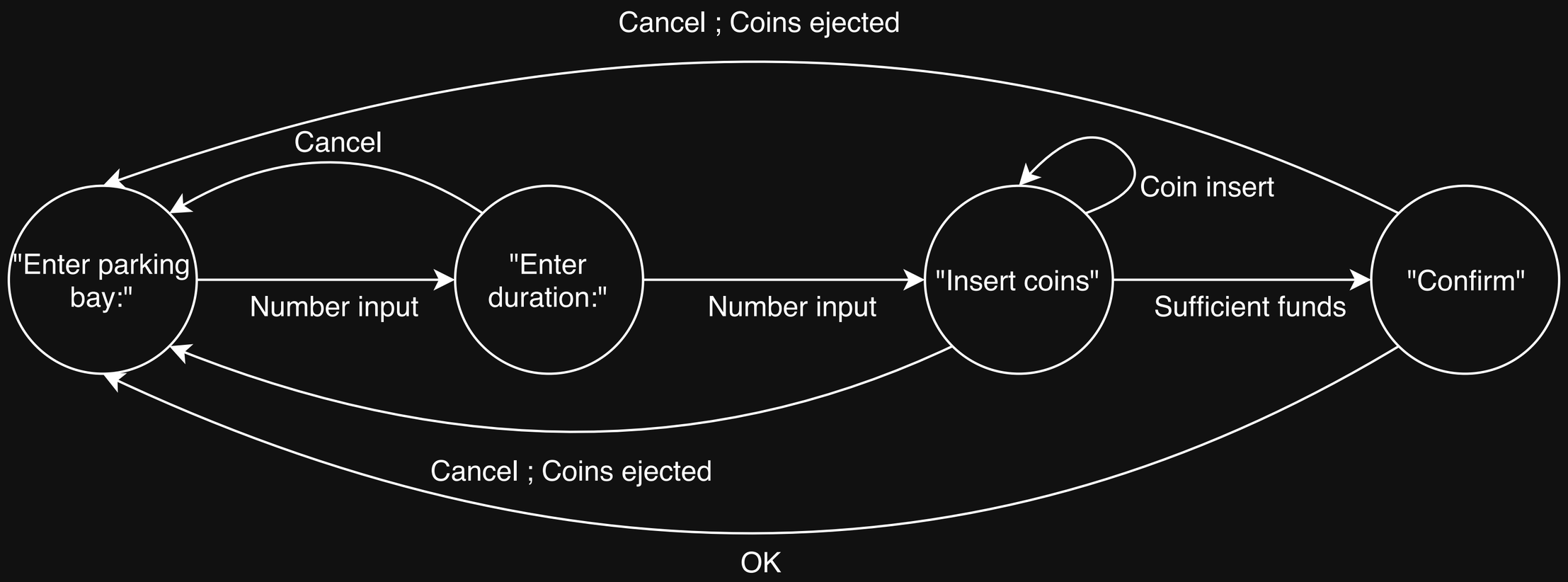
- A diagrammatic representation of a state.
- Some variation in notation.
- Typically: states are circles, transitions are labelled arrows connecting them



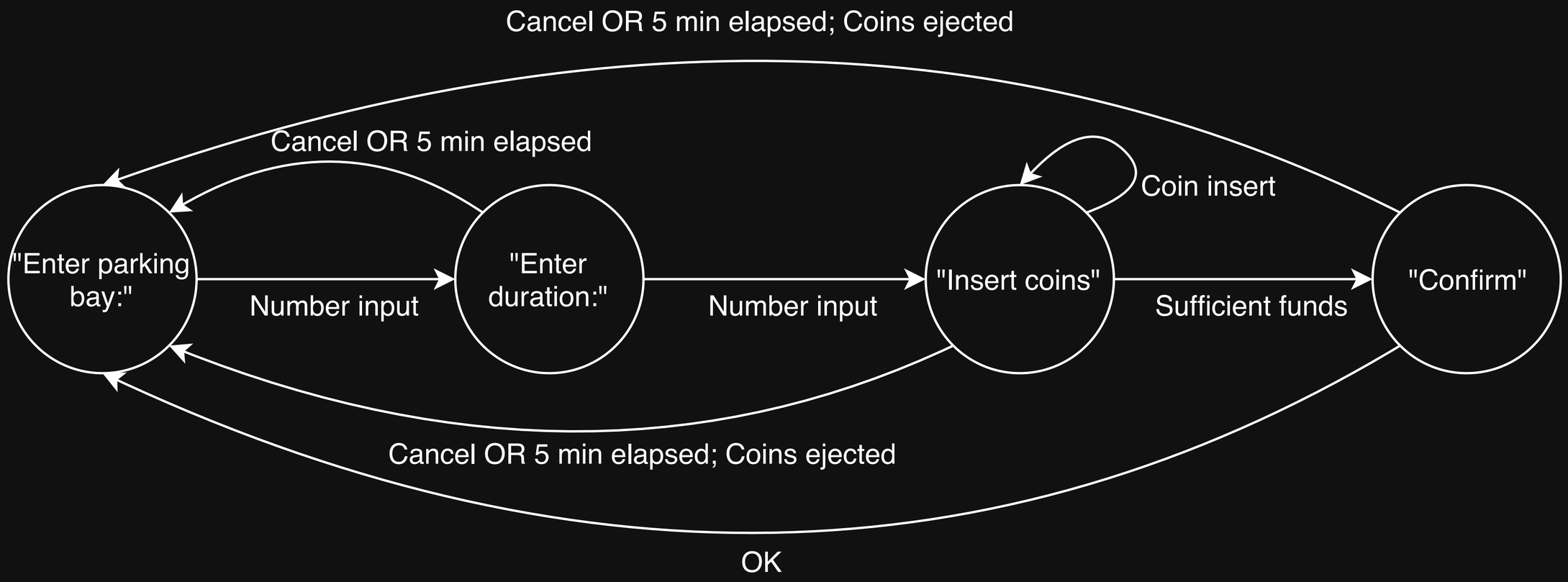
State machines

- Useful for modelling systems that have clearly defined states. For example:
  - UIs with different screens
  - Network protocols
  - Conversational interfaces

Parking meter



Parking meter



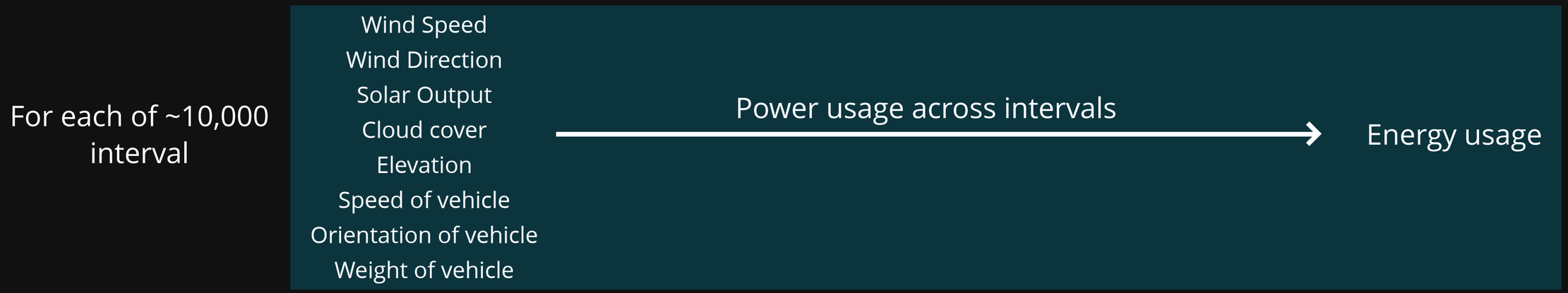
Opal Card

- Can we model the opal card system as a state machine?

For fun: A complex conceptual model

- In 2015 a UNSW student wrote a conceptual model "Gallium" based off of previous research to represent the energy usage of a solar car as it drove from Darwin to Adelaide
- The model was written in python and modeled the physical system of the vehicle and it's energy consumption over a fixed distance in response to a dynamic environmental and physical characteristics

The model calculated the energy usage across a number of fixed-distance intervals (e.g. a 200 metre stretch of road) over a 3000km journey across Australia.



Feedback