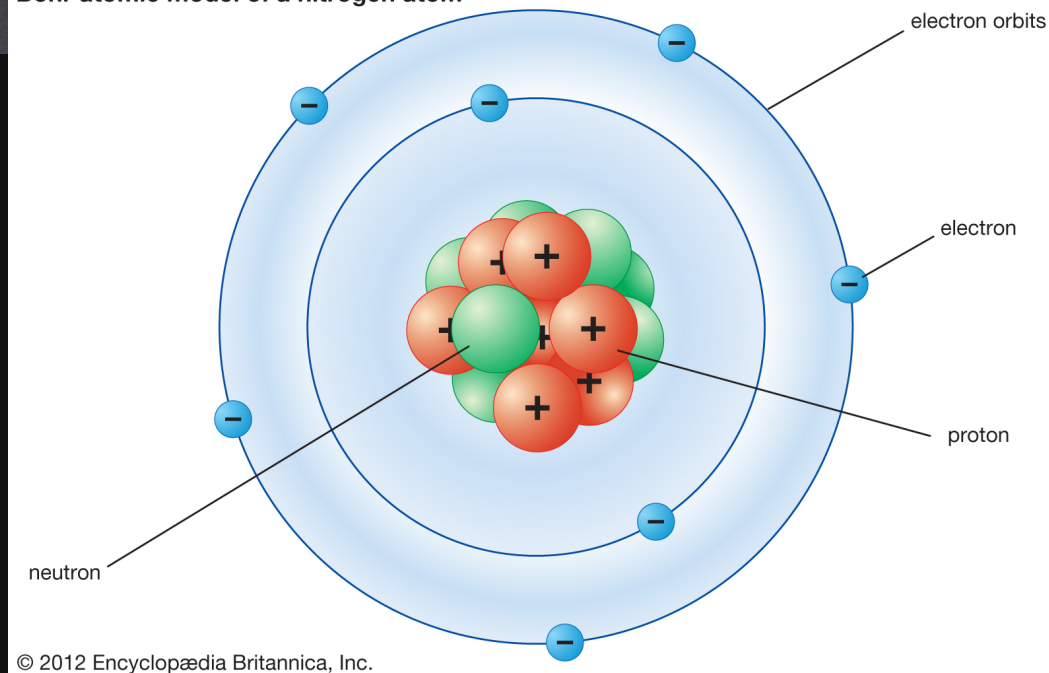# COMP1531

7.4 - SDLC Design - System Modelling

What's a model?
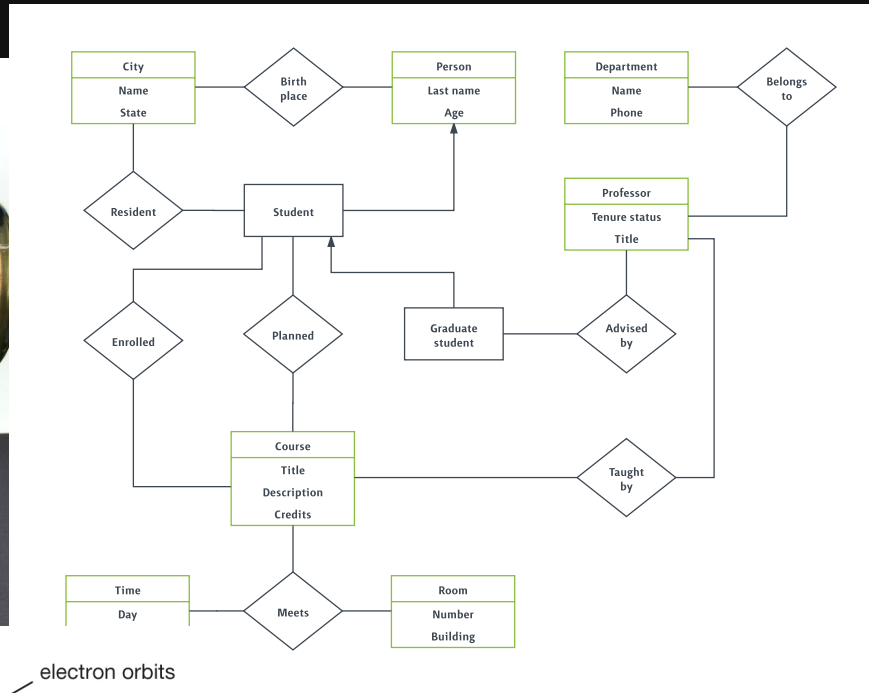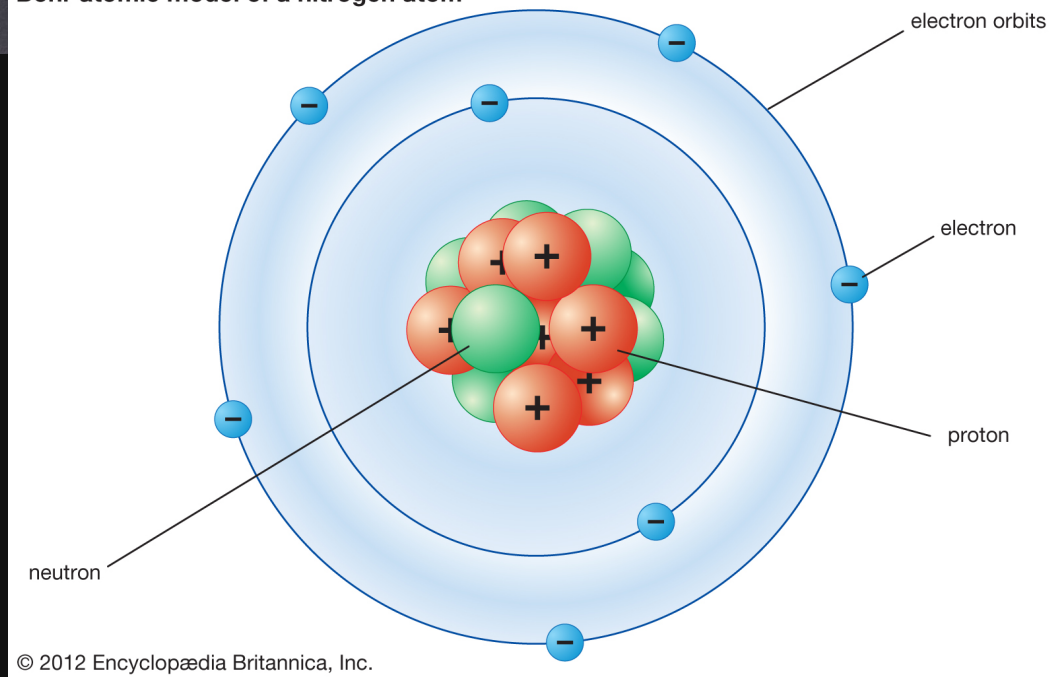
Bohr atomic model of a nitrogen atom

electron orbits

electron

proton

neutron

Bohr atomic model of a nitrogen atom

electron orbits

electron

proton

neutron

© 2012 Encyclopædia Britannica, Inc.

ER Diagram:

City — Name, State
Birth place
Person — Last name, Age
Department — Name, Phone
Belongs to
Resident
Student
Professor — Tenure status, Title
Enrolled
Planned
Graduate student
Advised by
Course — Title, Description, Credits
Taught by
Time — Day
Meets
Room — Number, Building

UML Class Diagram:

Bank
+BankId: int
+Name: string
+Location: string

Teller
+Id: int
+Name: string
+CollectMoney()
+OpenAccount()
+CloseAccount()
+LoanRequest()
+ProvideInfo()
+IssueCard()

Customer
+Id: int
+Name: string
+Address: string
+PhoneNo: int
+AcctNo: int
+GeneralInquiry()
+DepositMoney()
+WithdrawMoney()
+OpenAccount()
+CloseAccount()
+ApplyForLoan()
+RequestCard()

Account
+Id: int
+CustomerId: int

Checking
+Id: int
+CustomerId: int

Savings
+Id: int
+CustomerId: int

Loan
+Id: int
+Type: string
+AccountId: int
+CustomerId: int

on orbits

ctron

oton

on orbits

ER Diagram:

- City — Name, State
- Birth place
- Person — Last name, Age
- Department — Name, Phone
- Belongs to
- Resident
- Student
- Professor — Tenure status, Title
- Enrolled
- Planned
- Graduate student
- Advised by
- Course — Title, Description, Credits
- Taught by
- Time — Day
- Meets
- Room — Number, Building

Bank Class Diagram:

Bank
- +BankId: int
- +Name: string
- +Location: string

Teller
- +Id: int
- +Name: string
- +CollectMoney()
- +OpenAccount()
- +CloseAccount()
- +LoanRequest()
- +ProvideInfo()
- +IssueCard()

Customer
- +Id: int
- +Name: string
- +Address: string
- +PhoneNo: int
- +AcctNo: int
- +GeneralInquiry()
- +DepositMoney()
- +WithdrawMoney()
- +OpenAccount()
- +CloseAccount()
- +ApplyForLoan()
- +RequestCard()

Account
- +Id: int
- +CustomerId: int

Loan
- +Id: int
- +Type: string
- +AccountId: int
- +CustomerId: int

MVC Diagram:

PRESENTATION
VIEW
MODEL
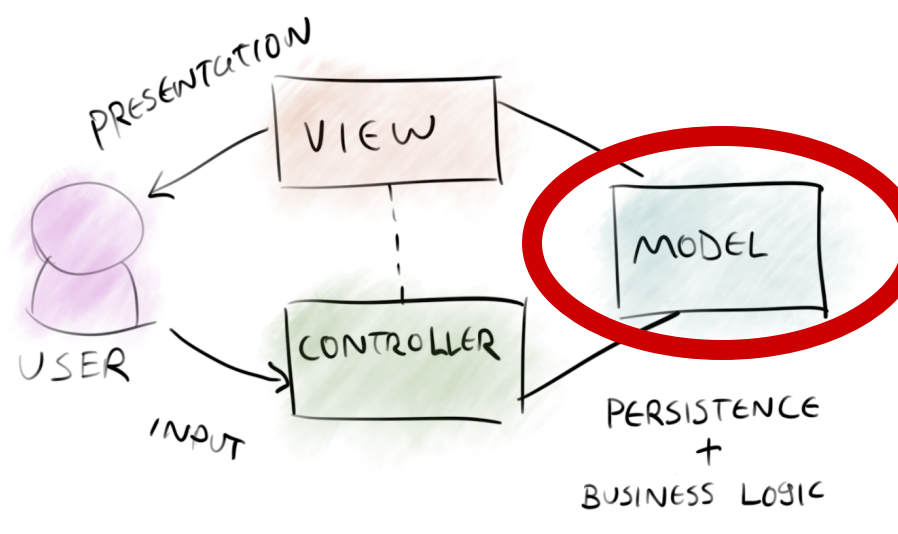USER
CONTROLLER
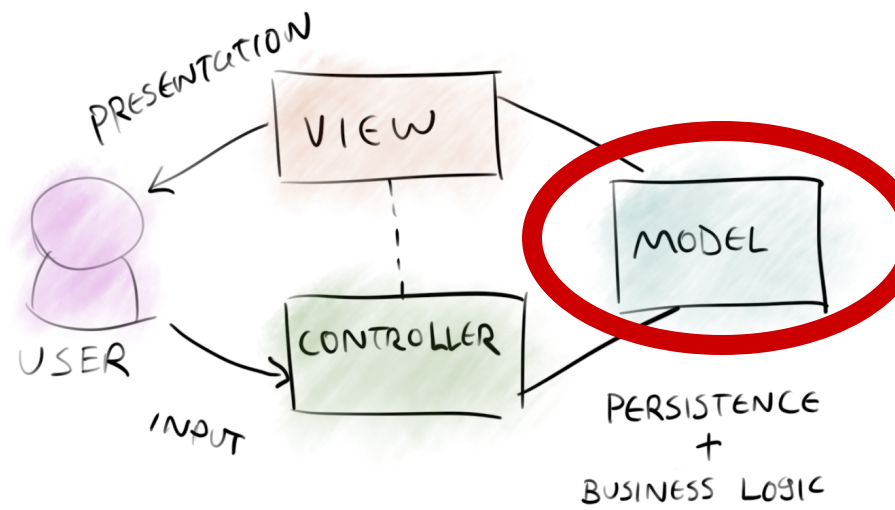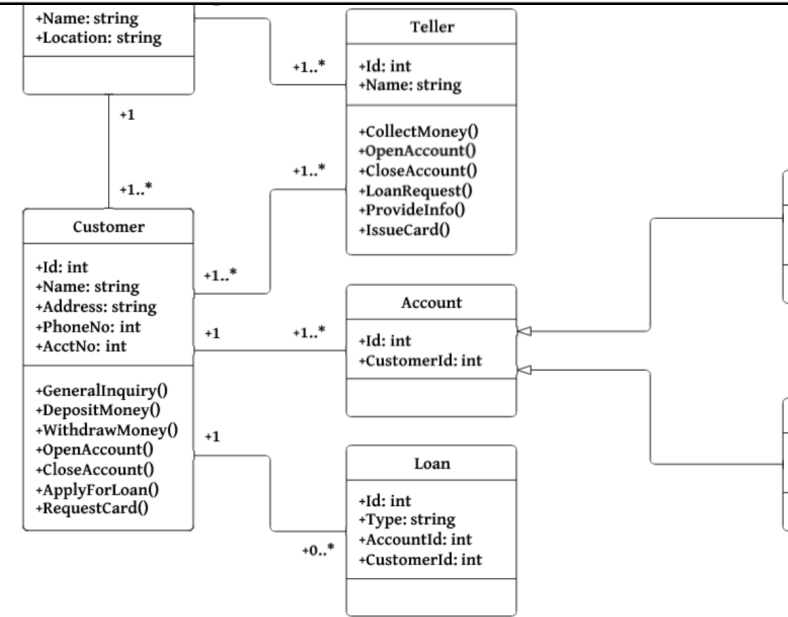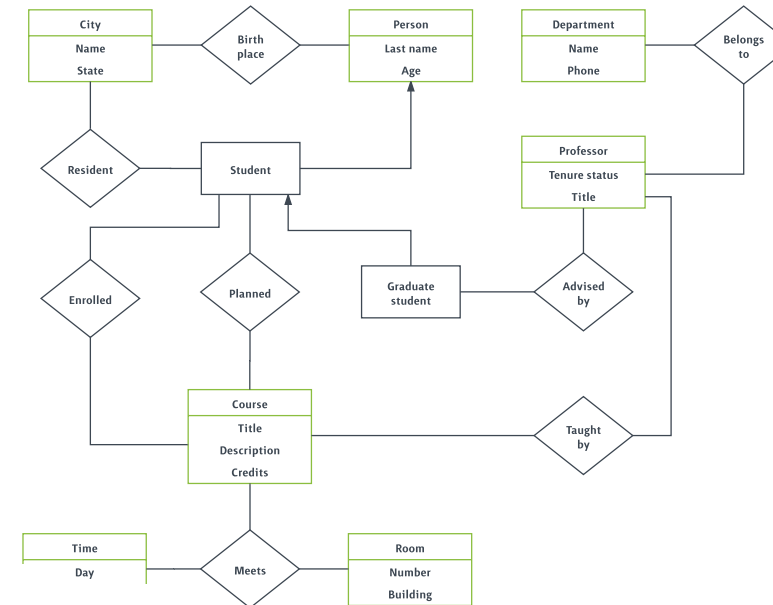INPUT
PERSISTENCE + BUSINESS LOGIC

Figure 1: Mathematical Model

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

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

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

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

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

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

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

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

on orbits

ER Diagram

City — Name — State | Birth place | Person — Last name — Age | Department — Name — Phone | Belongs to

Resident | Student | Professor — Tenure status — Title

Enrolled | Planned | Graduate student | Advised by

Course — Title — Description — Credits | Taught by

Time — Day | Meets | Room — Number — Building

UML Class Diagram

+Name: string
+Location: string

Teller
+Id: int
+Name: string

+CollectMoney()
+OpenAccount()
+CloseAccount()
+LoanRequest()
+ProvideInfo()
+IssueCard()

Customer
+Id: int
+Name: string
+Address: string
+PhoneNo: int
+AcctNo: int

+GeneralInquiry()
+DepositMoney()
+WithdrawMoney()
+OpenAccount()
+CloseAccount()
+ApplyForLoan()
+RequestCard()

Account
+Id: int
+CustomerId: int

Loan
+Id: int
+Type: string
+AccountId: int
+CustomerId: int

MVC Diagram

PRESENTATION — VIEW — MODEL — USER — CONTROLLER — INPUT — PERSISTENCE + BUSINESS LOGIC

Figure 1: Mathematical Model

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

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

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

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

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

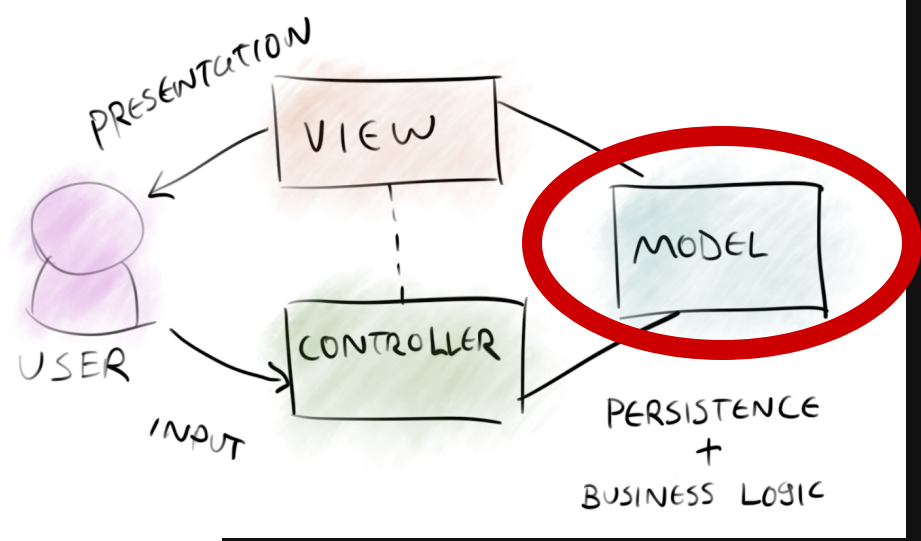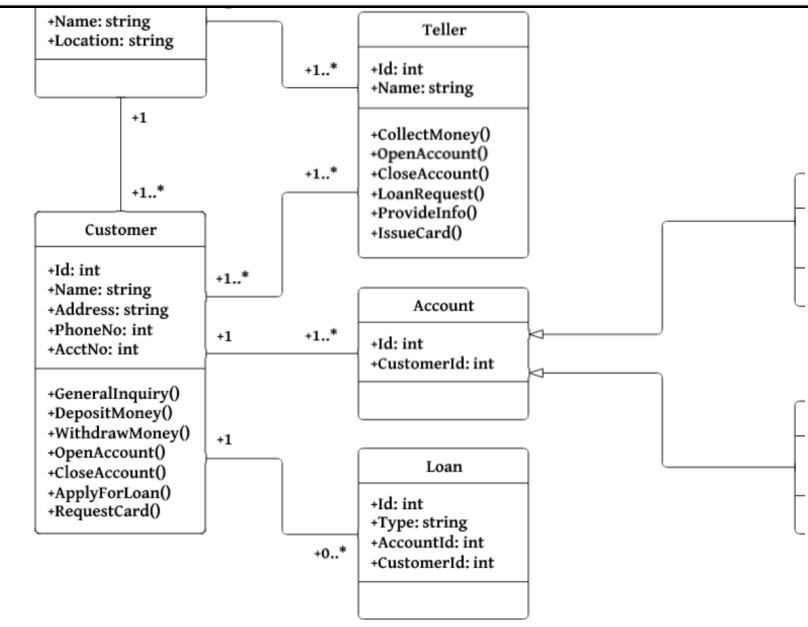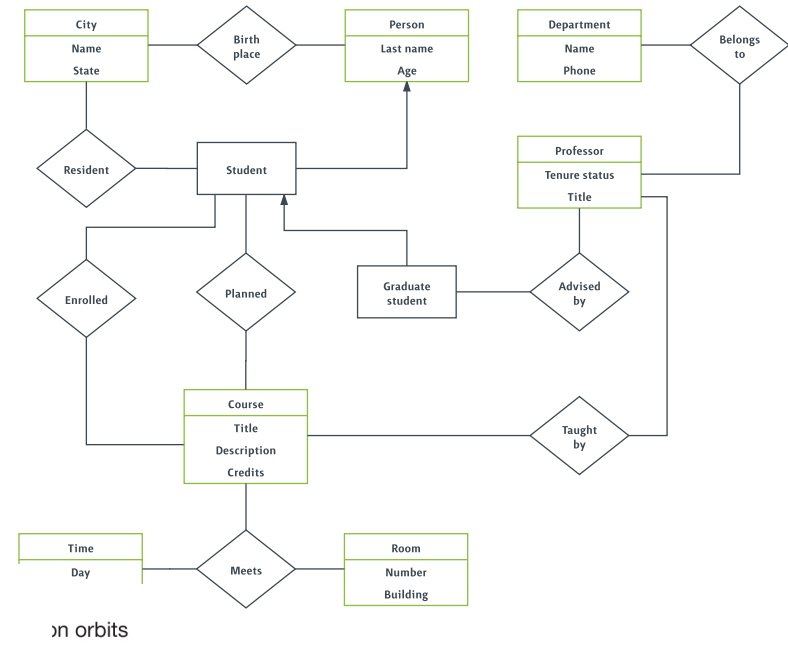$$\frac{dL_c}{dt} = \frac{s_{lc}L_c}{1+L_b/b_{l\infty}} + k_{clc}C - k_{lcm}L_c - \mu_{lc}L_c \qquad \frac{dL_c}{dt} = k_{lcm}L_c - \frac{k_{lmb}}{v_m}L_c - \mu_{lc}L_c$$

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

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

on orbits

ER Diagram

Entities and relationships:

- City (Name, State) — Birth place — Person (Last name, Age)
- Resident — Student
- Department (Name, Phone) — Belongs to
- Professor (Tenure status, Title) — Advised by
- Enrolled — Planned — Graduate student
- Course (Title, Description, Credits) — Taught by
- Time (Day) — Meets — Room (Number, Building)

UML Class Diagram

Bank
+Name: string
+Location: string

Teller
+Id: int
+Name: string
+CollectMoney()
+OpenAccount()
+CloseAccount()
+LoanRequest()
+ProvideInfo()
+IssueCard()

Customer
+Id: int
+Name: string
+Address: string
+PhoneNo: int
+AcctNo: int
+GeneralInquiry()
+DepositMoney()
+WithdrawMoney()
+OpenAccount()
+CloseAccount()
+ApplyForLoan()
+RequestCard()

Account
+Id: int
+CustomerId: int

Loan
+Id: int
+Type: string
+AccountId: int
+CustomerId: int

MVC Diagram

PRESENTATION — VIEW — MODEL (circled)

USER — INPUT — CONTROLLER

PERSISTENCE + BUSINESS LOGIC

Conceptual Modelling

- *A model that is conceptual*
  - *... with a real world correspondence*
  - *... without a real world correspondence*
- *A model of a concept*

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)

System Modelling

- Structural – Emphasise the static structure of the system
  - UML class diagrams
  - ER diagrams
  - … many others
- Behavioural - Emphasise the dynamic behaviour
  - State diagrams
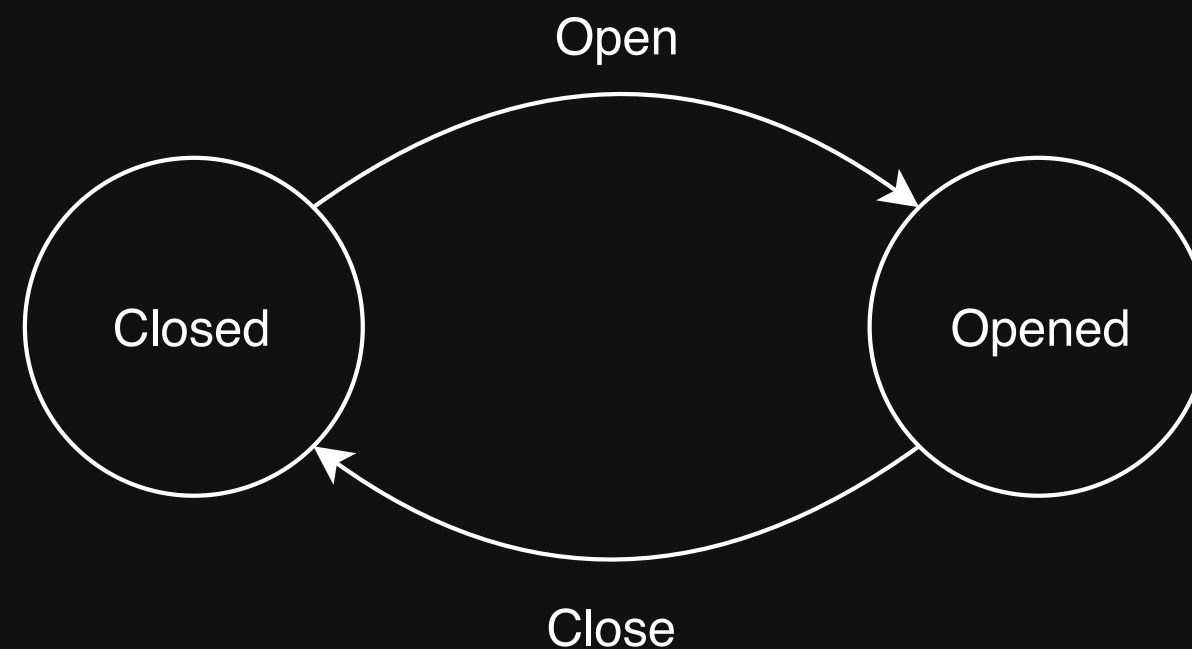 - Use case diagram
  - … some others

State Machines

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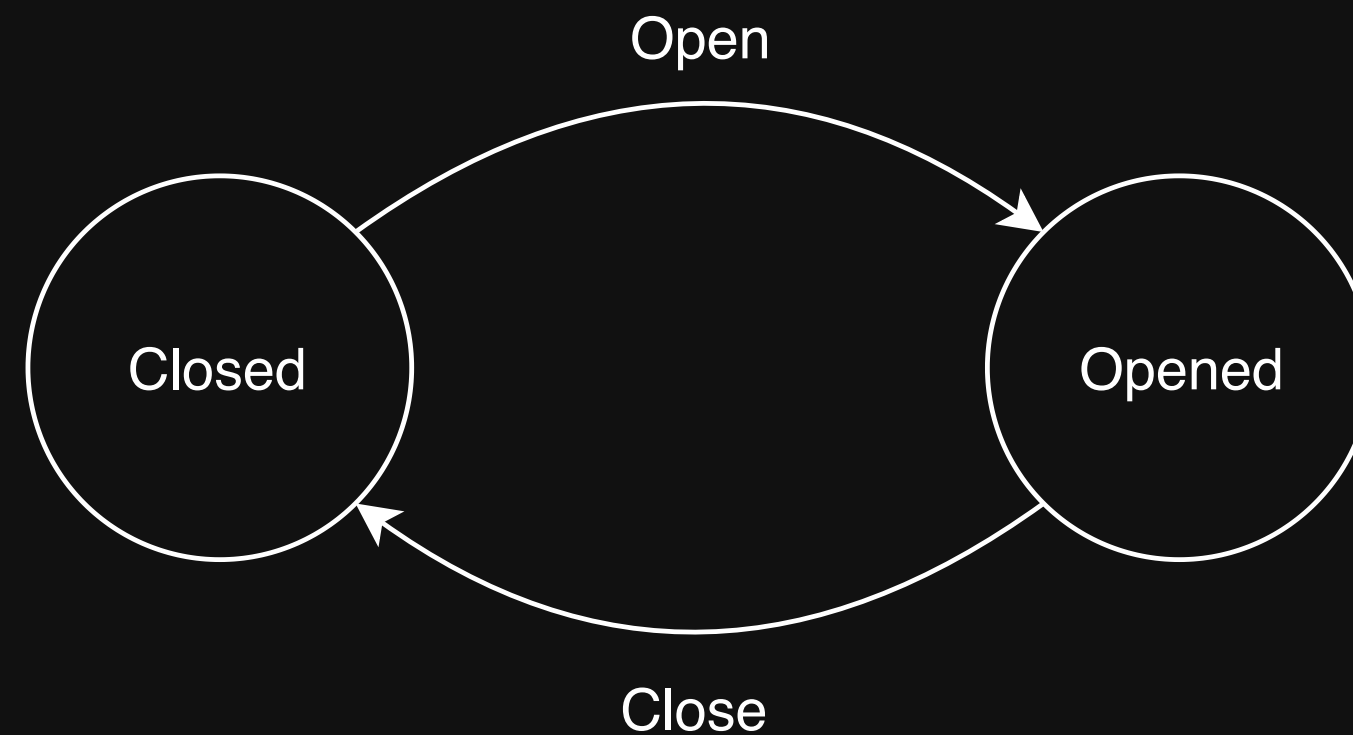
State diagrams

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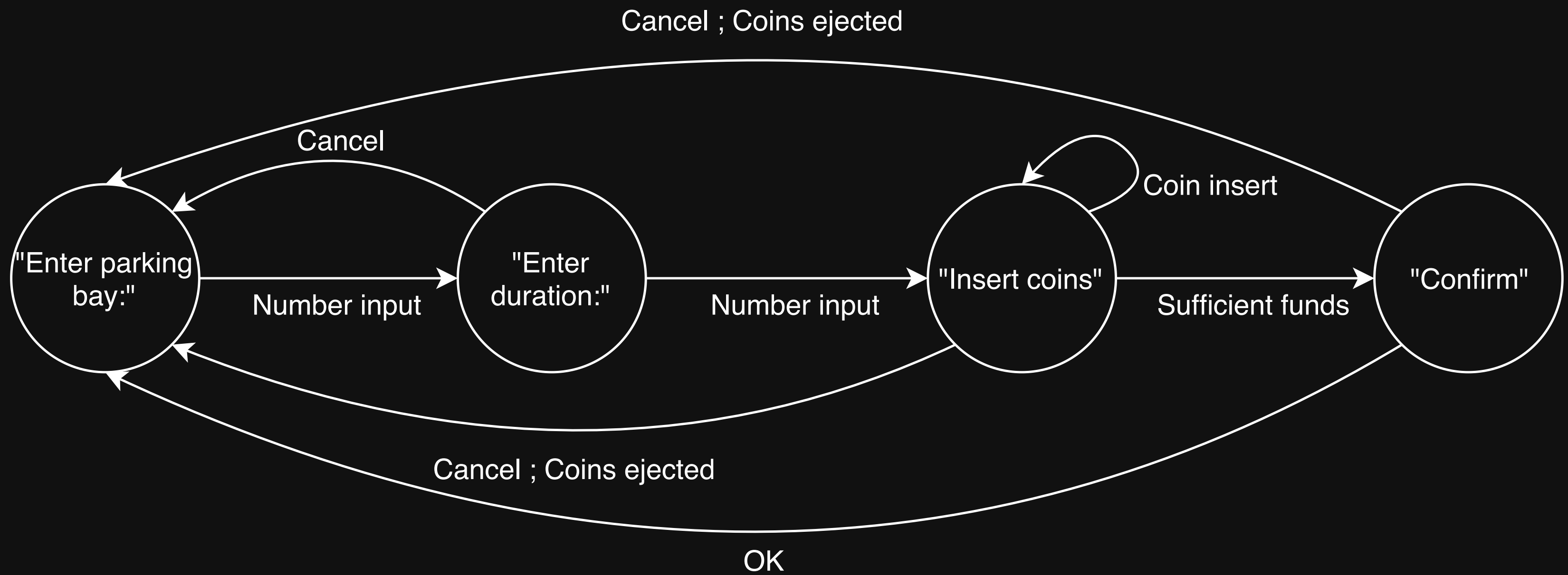- Typically: states are circles, transitions are labelled arrows connecting them
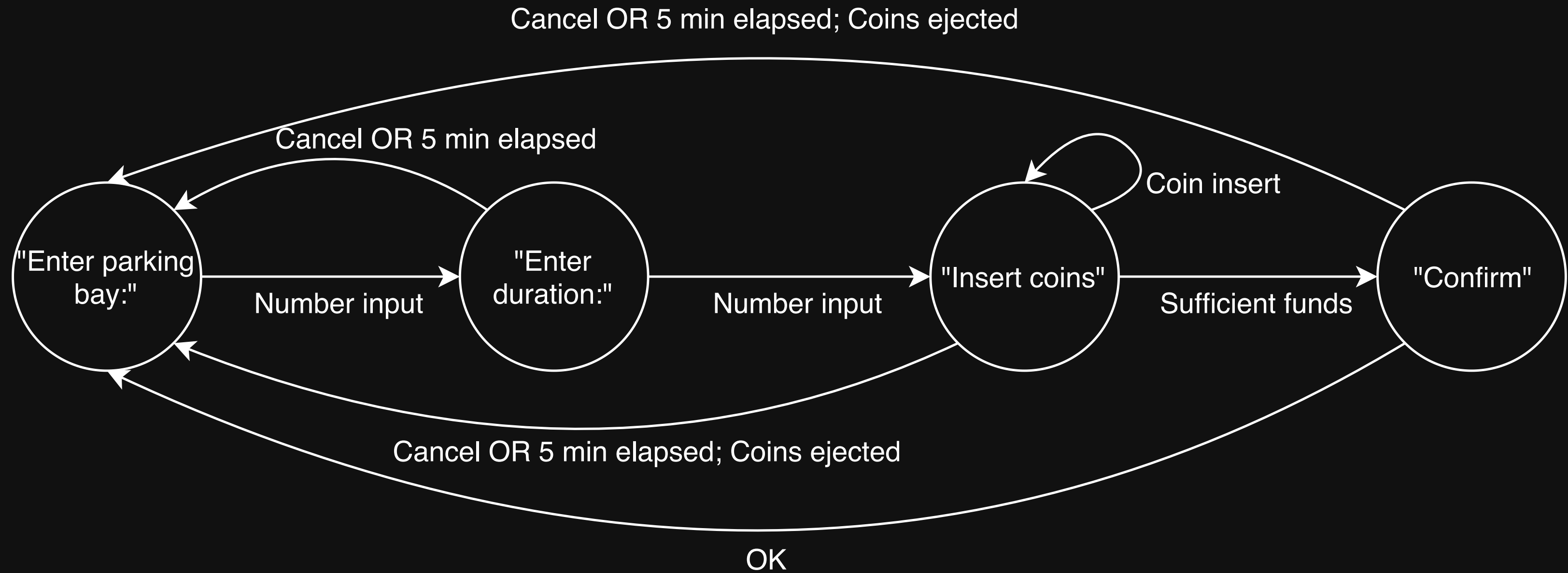
State machines

- 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?