SENG 471

Software Requirements Engineering

Modelling Functions - Behaviour

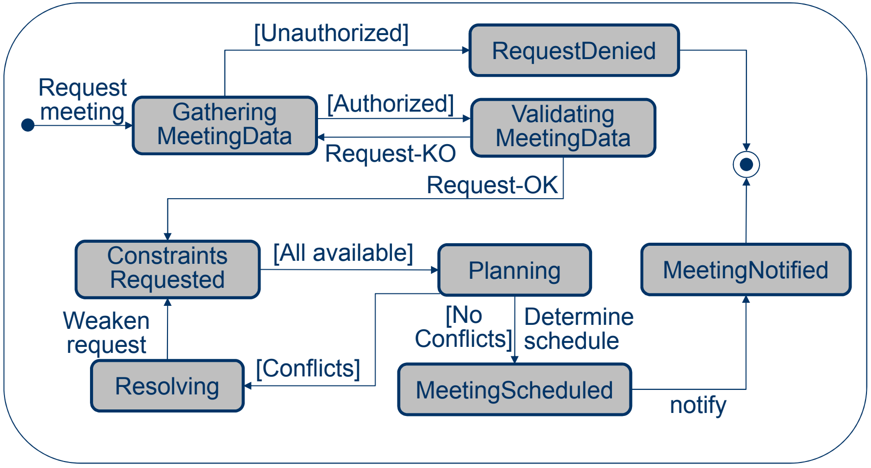
Example - Credit card

The credit card company issues a credit card. The customer calls the credit card company to activate the card. If the card has an outstanding balance and no payments are received for 60 days, then the card is suspended. Subsequently, if the minimum amount due is paid within 30 days, the card is re-activated; otherwise, the card is cancelled. At anytime the customer may call and cancel their credit card.

State Machine (SM) concepts: states, actions, transitions, guards, *and traces*

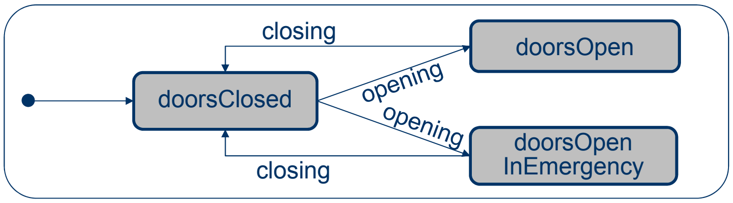
UML correspondence???

System Behaviour - SM diagrams



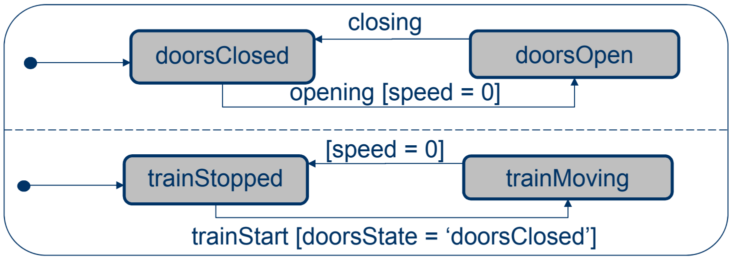
3

SM Diagrams - other behaviours



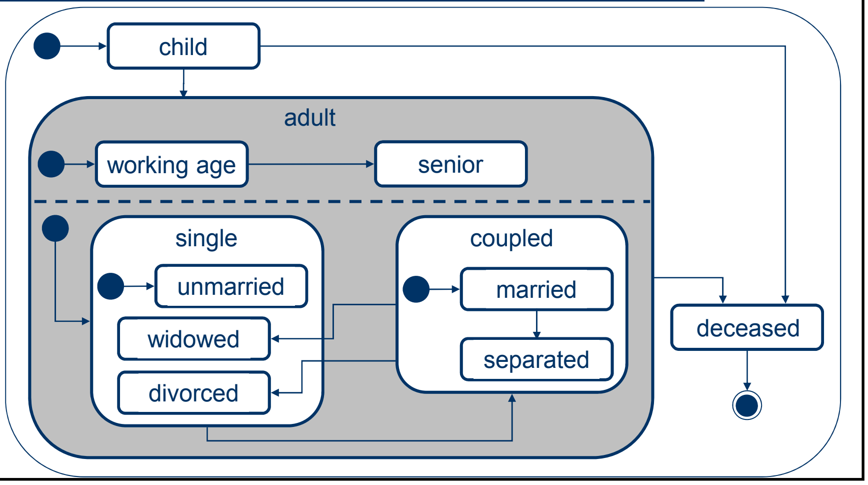
Non-deterministic

Concurrent



5

SM Diagram - Superstate



child

adult

working age

senior

single

unmarried

widowed

divorced

coupled

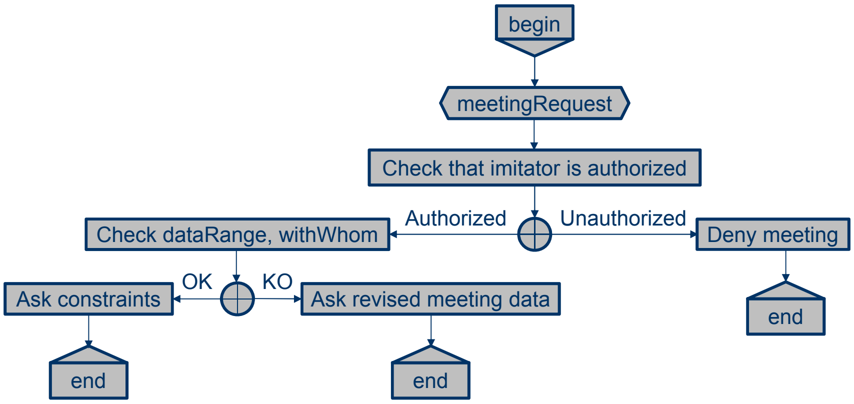
married

separated

deceased

R-net: To specify all operations that a system component is required to perform in response to a particular input stimulus, under a particular condition.

Stimuli-Responses - R-net Diag.



begin

meetingRequest

Check that imitator is authorized

Authorized

Unauthorized

Check dataRange, withWhom

Deny meeting

OK

KO

Ask constraints

Ask revised meeting data

end

end

end

[Amb03]

Style Tips

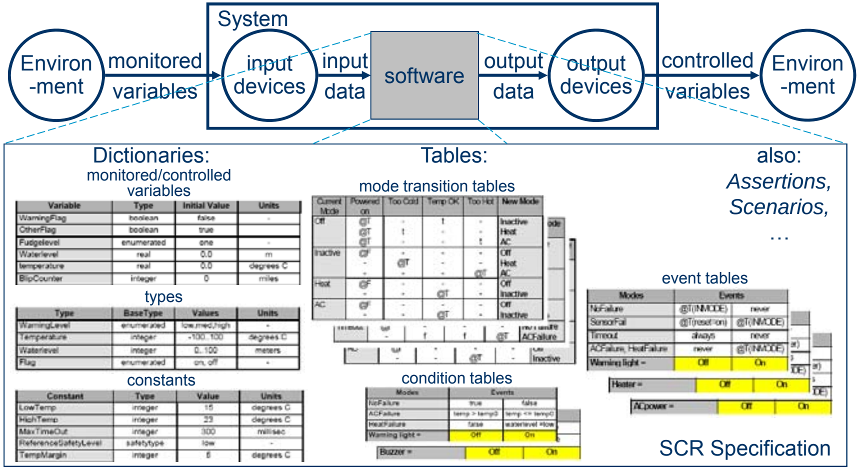
- The diagram should have start and end state(s).
- Diagrams are usually read from top-left to bottom-right, so put the start and end states in those locations.
- Each state should have at least one transition into it and at least one transition out of it.
- Use a concurrent (or superstate) state when multiple states have a common entry or exit condition.
- It is fine for guards on transitions from a state to not form a complete set.

10

Dr. Y. Hu

Software Cost Reduction (SCR) is a set of techniques for designing software systems.

SCR - Tabular Specifications



monitored/controlled variables

Variable	Type	Initial Value	Units
WarningFlag	boolean	false	-
OtherFlag	boolean	true	
Fudgelevel	enumerated	one	-
Waterlevel	real	0.0	m
temperature	real	0.0	degrees C
BlipCounter	integer	0	miles

types

Type	BaseType	Values	Units
WarningLevel	enumerated	low,med,high	-
Temperature	integer	-100..100	degrees C
Waterlevel	integer	0..100	meters
Flag	enumerated	on, off	-

constants

Constant	Type	Value	Units
LowTemp	integer	15	degrees C
HighTemp	integer	23	degrees C
MaxTimeOut	integer	300	millisec
ReferenceSafetyLevel	safetytype	low	-
TempMargin	integer	5	degrees C

mode transition tables

Current Mode	Powered on	Too Cold	Temp OK	Too Hot	New Mode
Off	@T @T @T	- t -	t - -	- - t	Inactive Heat AC
Inactive	@F - -	- @T -	- - -	- - @T	Off Heat AC
Heat	@F -	- -	- @T	- -	Off Inactive
AC	@F -	- -	- @T	- -	Off Inactive

event tables

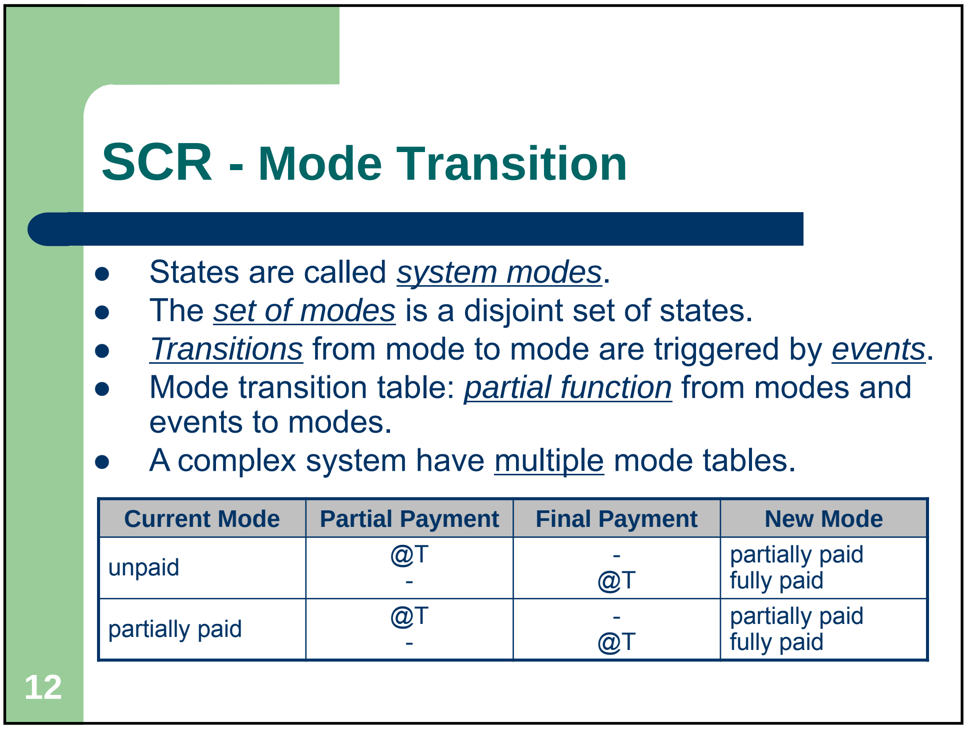
Modes	Events	
NoFailure	@T(INMODE)	never
SensorFail	@T(reset=on)	@T(INMODE)
Timeout	always	never
ACFailure, HeatFailure	never	@T(INMODE)
Warning light =	Off	On

Heater =	Off	On
ACpower =	Off	On

condition tables

Modes	Events	
NoFailure	true	false
ACFailure	temp > temp0	temp <= temp0
HeatFailure	false	waterlevel =low
Warning light =	Off	On
Buzzer =	Off	On

11

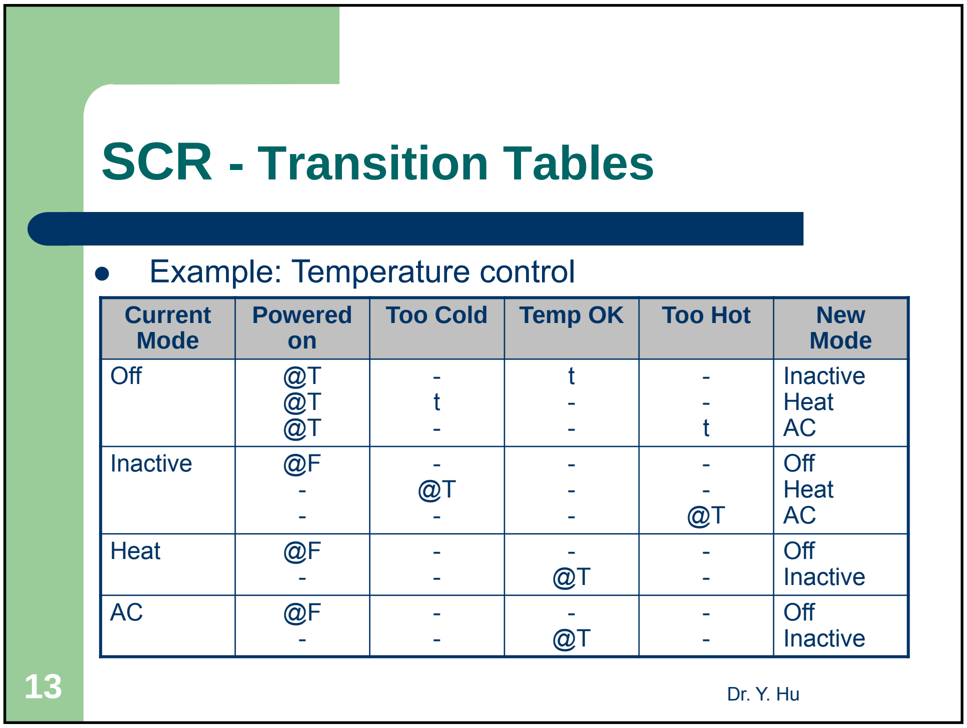


SCR - Mode Transition

- States are called *system modes*.
- The *set of modes* is a disjoint set of states.
- *Transitions* from mode to mode are triggered by *events*.
- Mode transition table: *partial function* from modes and events to modes.
- A complex system have multiple mode tables.

Current Mode	Partial Payment	Final Payment	New Mode
unpaid	@T -	- @T	partially paid fully paid
partially paid	@T -	- @T	partially paid fully paid

12



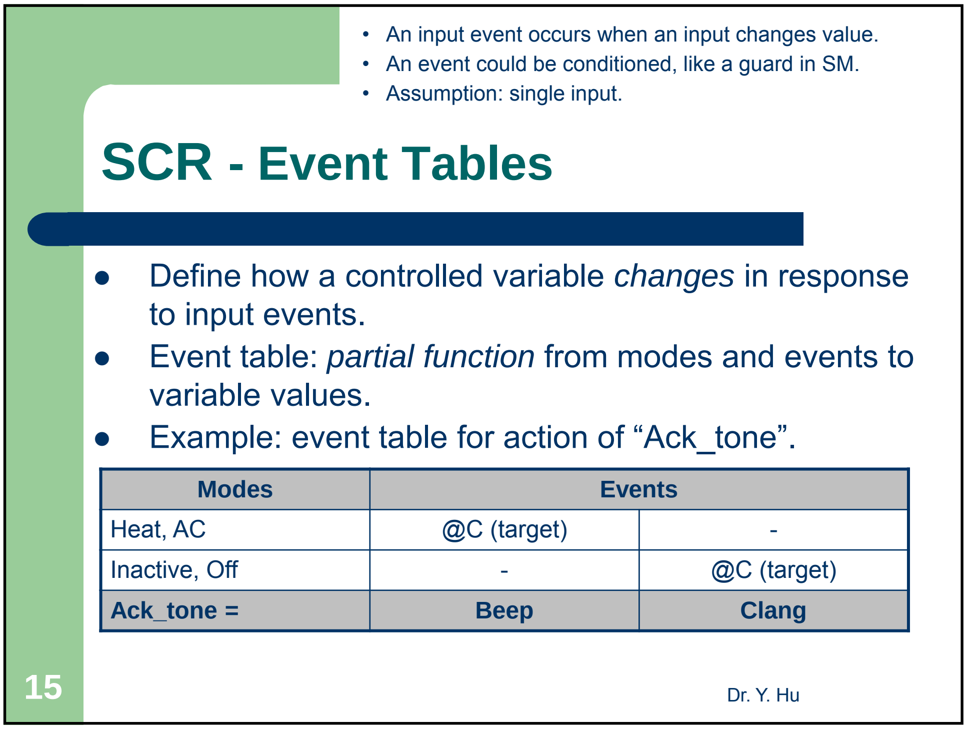
SCR - Transition Tables

- Example: Temperature control

Current Mode	Powered on	Too Cold	Temp OK	Too Hot	New Mode
Off	@T @T @T	- t -	t - -	- - t	Inactive Heat AC
Inactive	@F - -	- @T -	- - -	- - @T	Off Heat AC
Heat	@F -	- -	- @T	- -	Off Inactive
AC	@F -	- -	- @T	- -	Off Inactive

13

Dr. Y. Hu



- An input event occurs when an input changes value.
- An event could be conditioned, like a guard in SM.
- Assumption: single input.

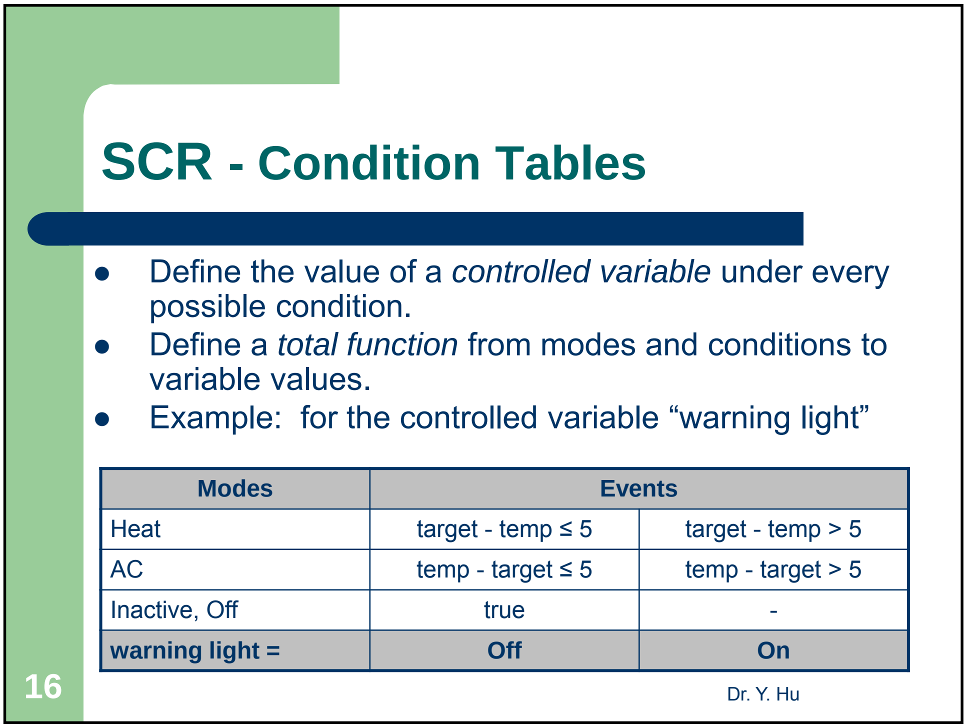
SCR - Event Tables

- Define how a controlled variable *changes* in response to input events.
- Event table: *partial function* from modes and events to variable values.
- Example: event table for action of “Ack_tone”.

Modes	Events	
Heat, AC	@C (target)	-
Inactive, Off	-	@C (target)
Ack_tone =	**Beep**	**Clang**

15

Dr. Y. Hu



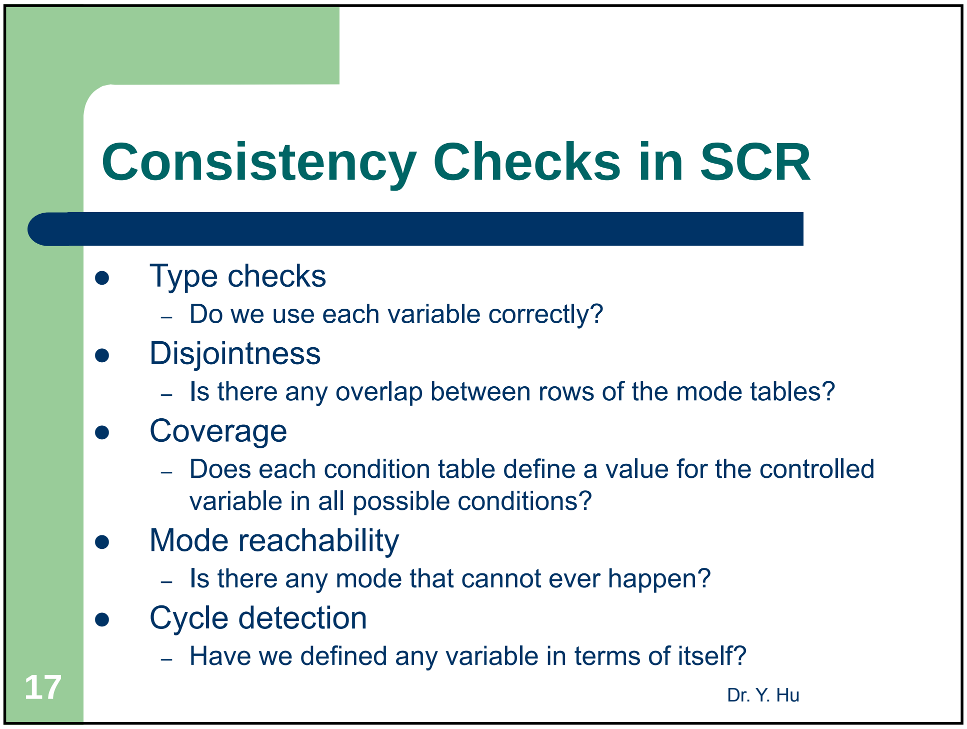
SCR - Condition Tables

- Define the value of a *controlled variable* under every possible condition.
- Define a *total function* from modes and conditions to variable values.
- Example: for the controlled variable “warning light”

Modes	Events	
Heat	target - temp ≤ 5	target - temp > 5
AC	temp - target ≤ 5	temp - target > 5
Inactive, Off	true	-
warning light =	**Off**	**On**

16

Dr. Y. Hu

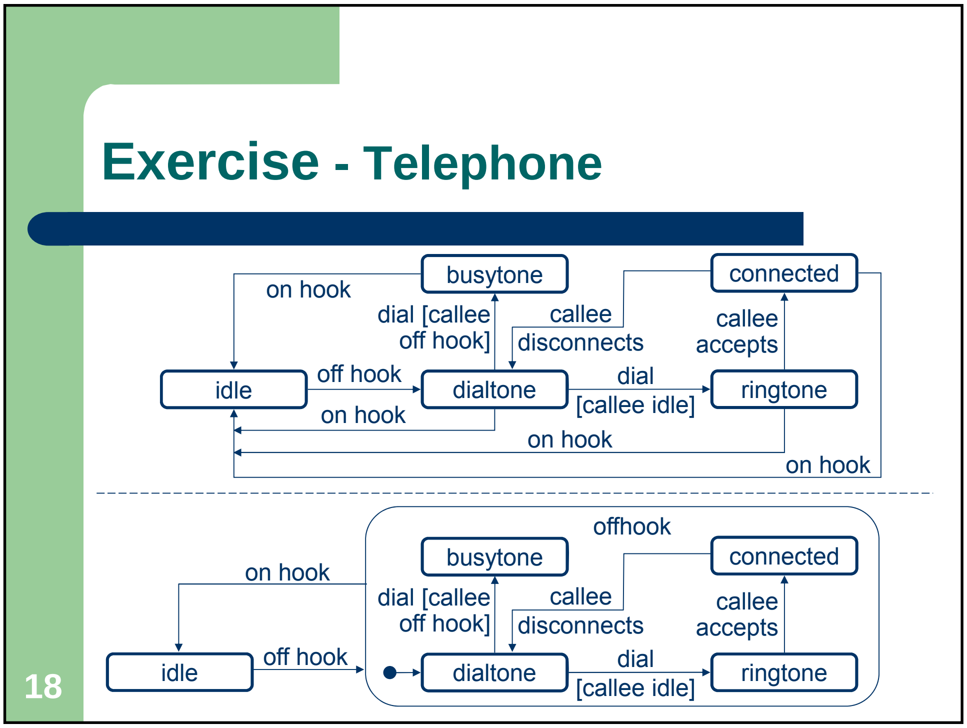


Consistency Checks in SCR

- Type checks
  - Do we use each variable correctly?
- Disjointness
  - Is there any overlap between rows of the mode tables?
- Coverage
  - Does each condition table define a value for the controlled variable in all possible conditions?
- Mode reachability
  - Is there any mode that cannot ever happen?
- Cycle detection
  - Have we defined any variable in terms of itself?

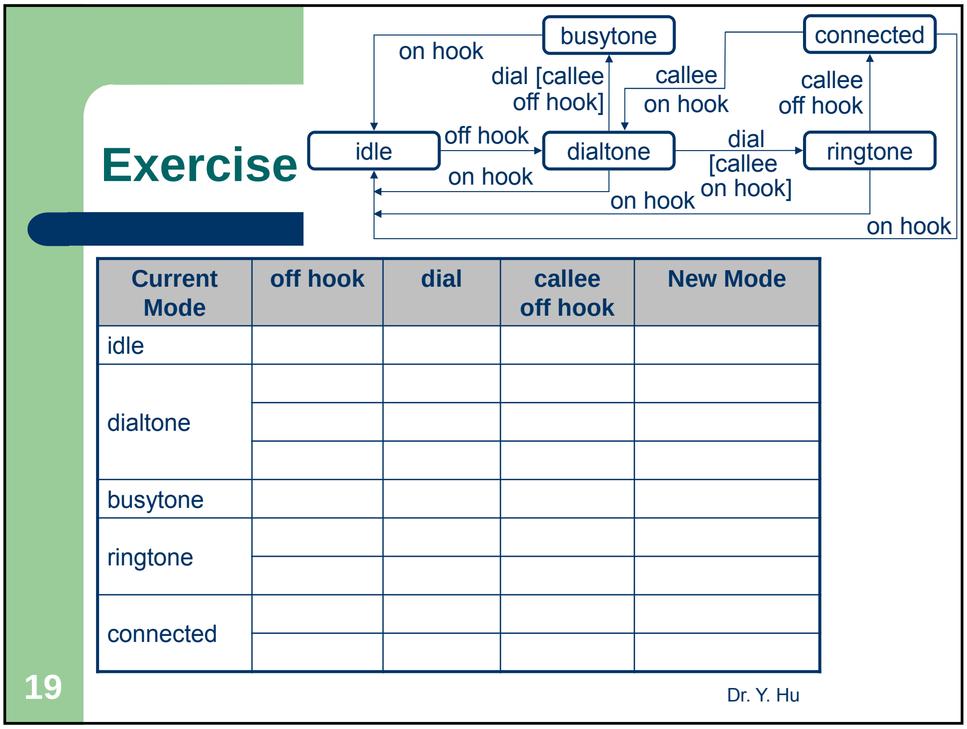
17

Dr. Y. Hu



Exercise - Telephone

18

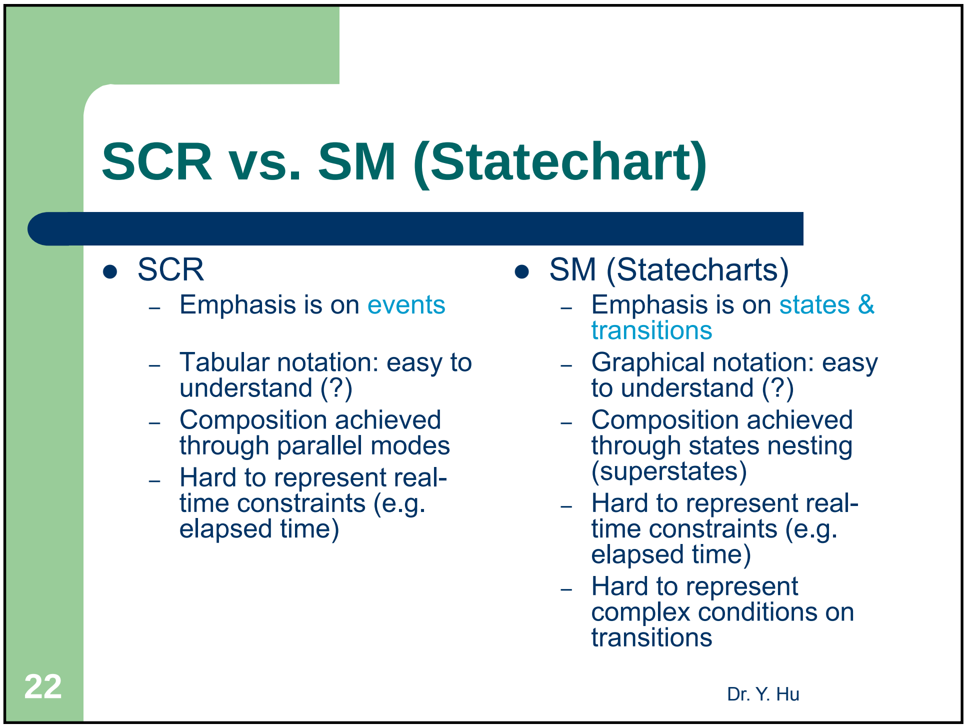


Exercise

Current Mode	off hook	dial	callee off hook	New Mode
idle				
dialtone				
busytone				
ringtone				
connected				

19

Dr. Y. Hu

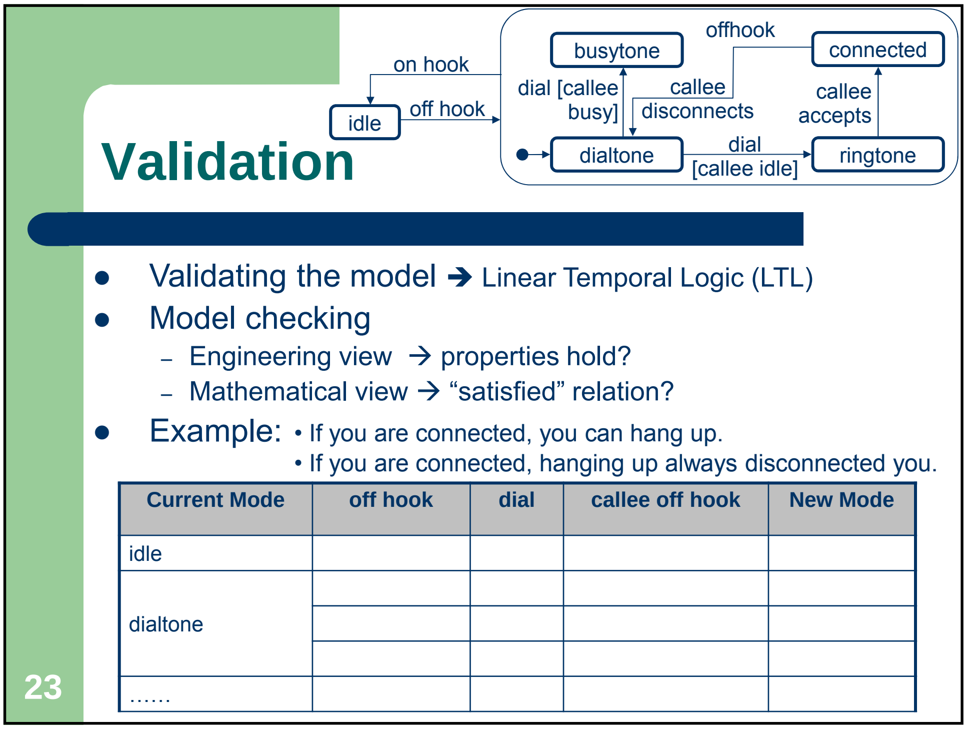


SCR vs. SM (Statechart)

- SCR
  - Emphasis is on events
  - Tabular notation: easy to understand (?)
  - Composition achieved through parallel modes
  - Hard to represent real-time constraints (e.g. elapsed time)
- SM (Statecharts)
  - Emphasis is on states & transitions
  - Graphical notation: easy to understand (?)
  - Composition achieved through states nesting (superstates)
  - Hard to represent real-time constraints (e.g. elapsed time)
  - Hard to represent complex conditions on transitions

22

Dr. Y. Hu

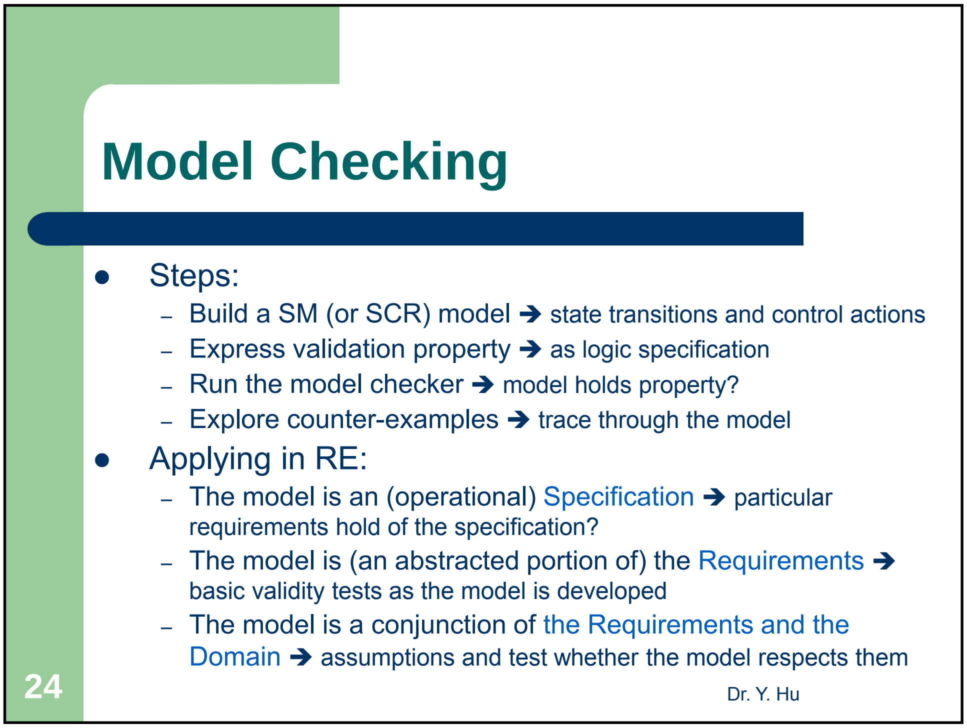


Validation

- Validating the model ➔ Linear Temporal Logic (LTL)
- Model checking
  - Engineering view → properties hold?
  - Mathematical view → "satisfied" relation?
- Example: • If you are connected, you can hang up.
  • If you are connected, hanging up always disconnected you.

Current Mode	off hook	dial	callee off hook	New Mode
idle				
dialtone				
……				

23

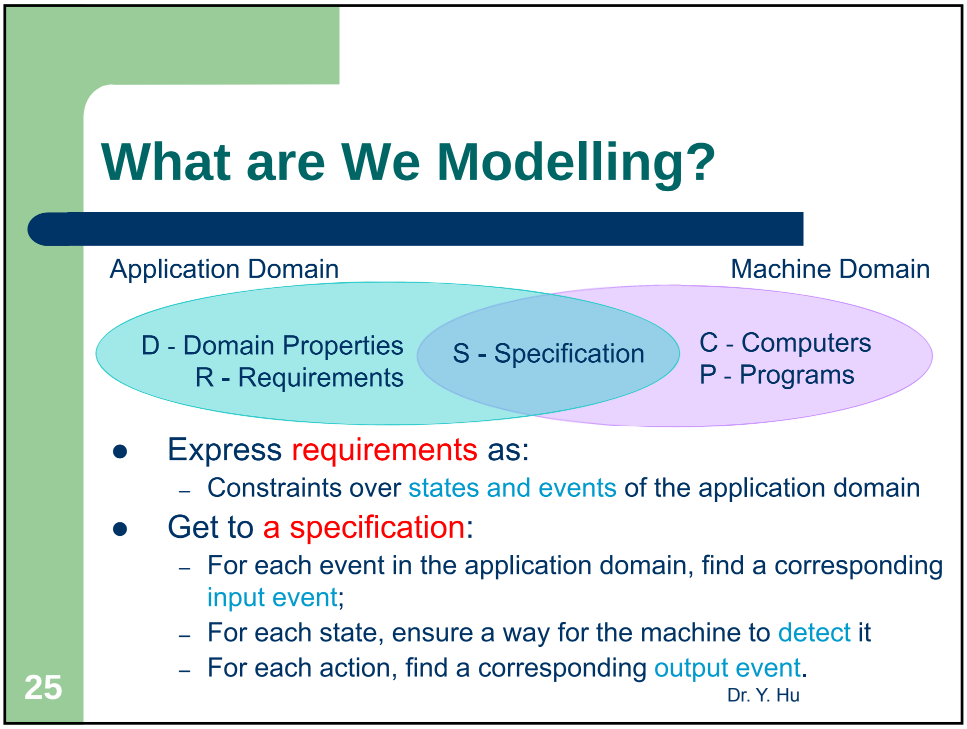


Model Checking

- Steps:
  - Build a SM (or SCR) model ➔ state transitions and control actions
  - Express validation property ➔ as logic specification
  - Run the model checker ➔ model holds property?
  - Explore counter-examples ➔ trace through the model
- Applying in RE:
  - The model is an (operational) Specification ➔ particular requirements hold of the specification?
  - The model is (an abstracted portion of) the Requirements ➔ basic validity tests as the model is developed
  - The model is a conjunction of the Requirements and the Domain ➔ assumptions and test whether the model respects them

24

Dr. Y. Hu



What are We Modelling?

Application Domain

Machine Domain

D - Domain Properties
R - Requirements

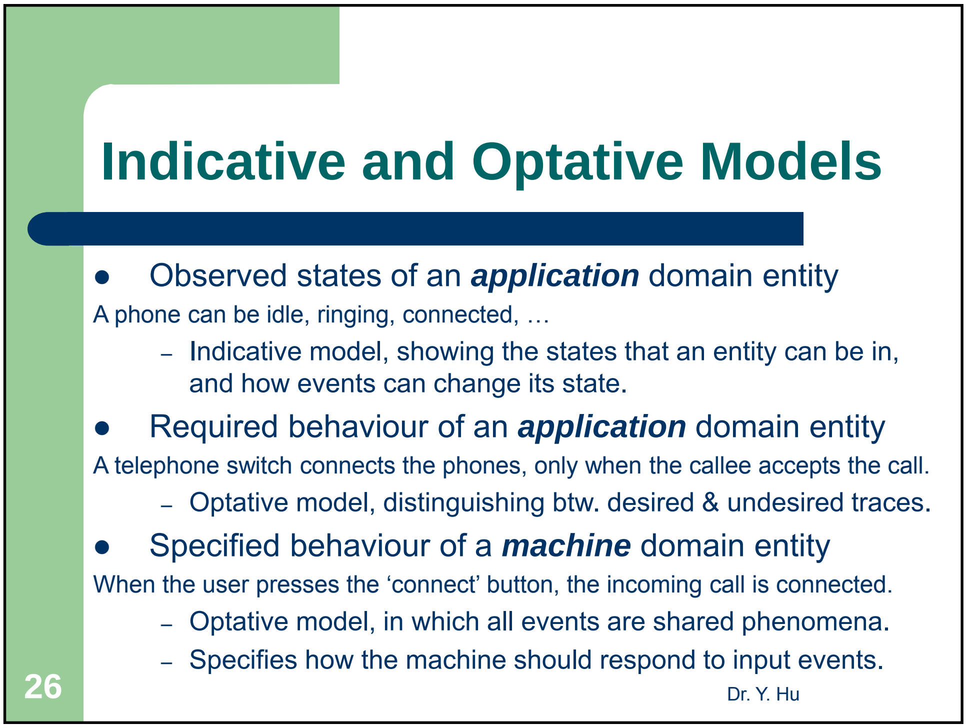
S - Specification

C - Computers
P - Programs

- Express requirements as:
  - Constraints over states and events of the application domain
- Get to a specification:
  - For each event in the application domain, find a corresponding input event;
  - For each state, ensure a way for the machine to detect it
  - For each action, find a corresponding output event.

25

Dr. Y. Hu



Indicative and Optative Models

- Observed states of an ***application*** domain entity

A phone can be idle, ringing, connected, …

  - Indicative model, showing the states that an entity can be in, and how events can change its state.

- Required behaviour of an ***application*** domain entity

A telephone switch connects the phones, only when the callee accepts the call.

  - Optative model, distinguishing btw. desired & undesired traces.

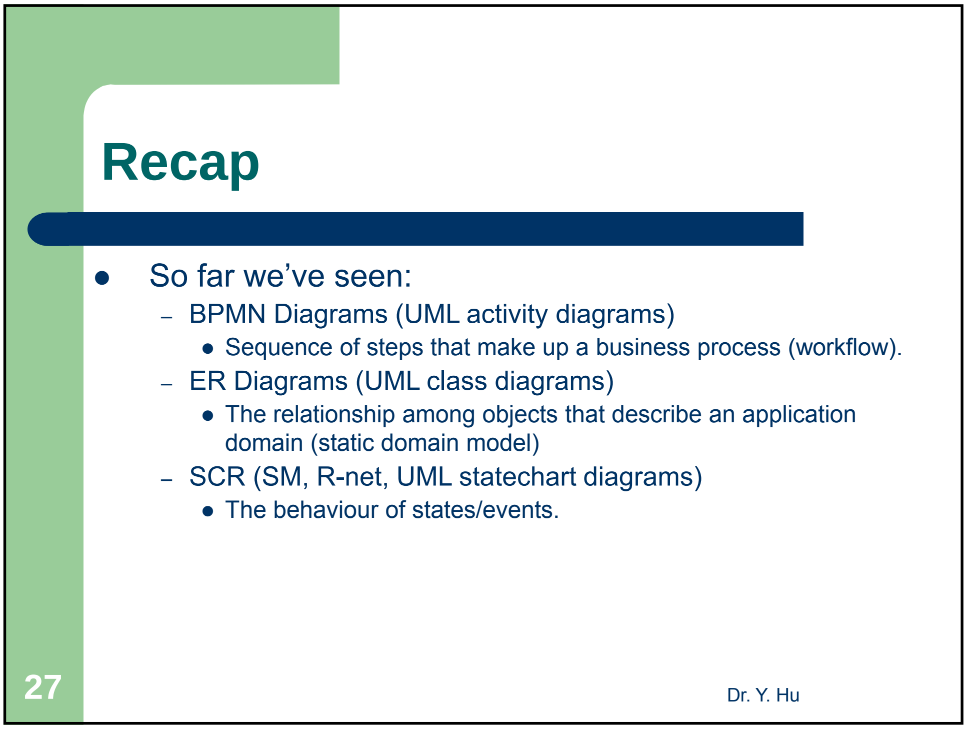
- Specified behaviour of a ***machine*** domain entity

When the user presses the ‘connect’ button, the incoming call is connected.

  - Optative model, in which all events are shared phenomena.
  - Specifies how the machine should respond to input events.

26

Dr. Y. Hu



Recap

- So far we've seen:
  - BPMN Diagrams (UML activity diagrams)
    - Sequence of steps that make up a business process (workflow).
  - ER Diagrams (UML class diagrams)
    - The relationship among objects that describe an application domain (static domain model)
  - SCR (SM, R-net, UML statechart diagrams)
    - The behaviour of states/events.