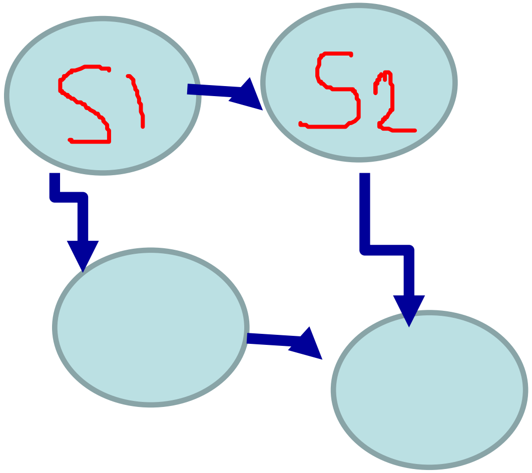
Advanced Software Engineering CSE608



UML State Machine Modeling and Testing

Dr. Islam El-Maddah

State Machine Diagram

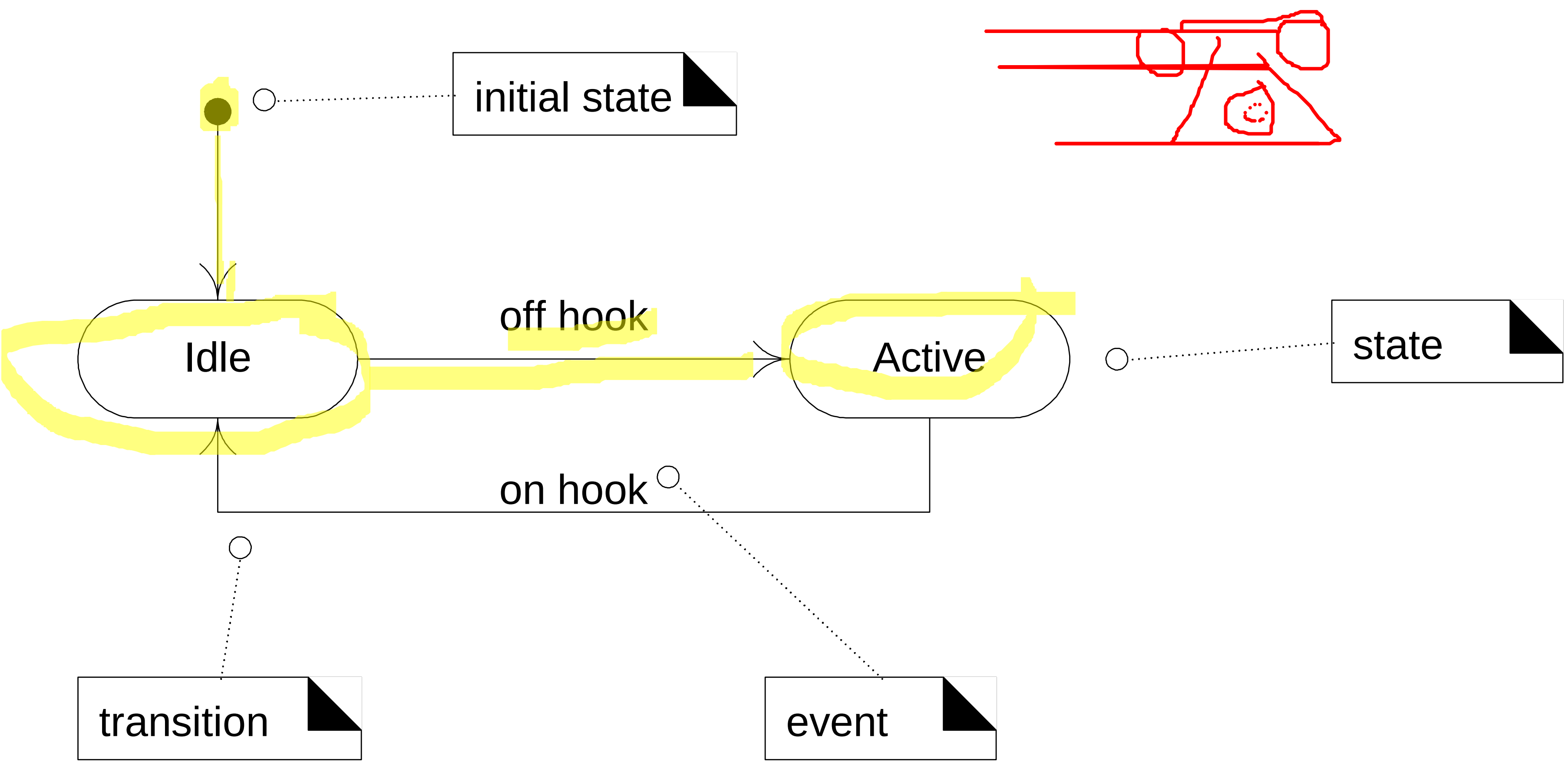
- Illustrates the interesting events and states of an object and the behavior of an object in reaction to an event.
  - Event: significant or noteworthy occurrence.
    - E.g., telephone receiver taken off hook.
  - State: the condition of an object at a moment in time (between events).
  - Transition: a relationship between two states; when an event occurs, the object moves from the current state to a related state.

UML State Machine Diagram

- States shown as rounded rectangles.
- Transitions shown as arrows.
- Events shown as labels on transition arrows.
- Initial pseudo-state automatically transitions to a particular state on object instantiation.
- Events with no corresponding transitions are ignored.

Fig. 29.1 State machine diagram for a telephone

Telephone



initial state

Idle

off hook

Active

state

on hook

transition

event

Transition Actions and Guards

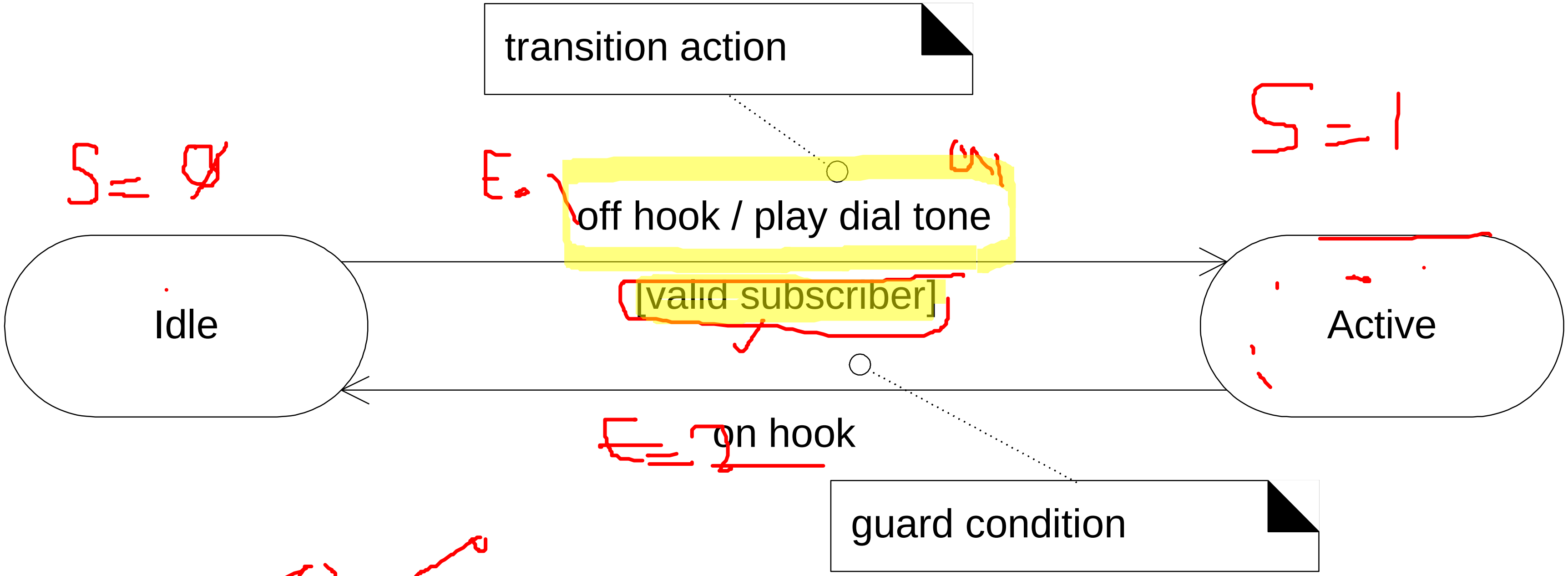
- A transition can cause an action to fire.
  - In software implementation, a method of the class of the state machine is invoked.
- A transition may have a conditional guard.
  - The transition occurs only if the test passes.

Fig. 29.2 Transition action and guard notation

e

e1 e2 e3

event [cond] /action



do do

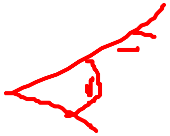
sub sub

ev

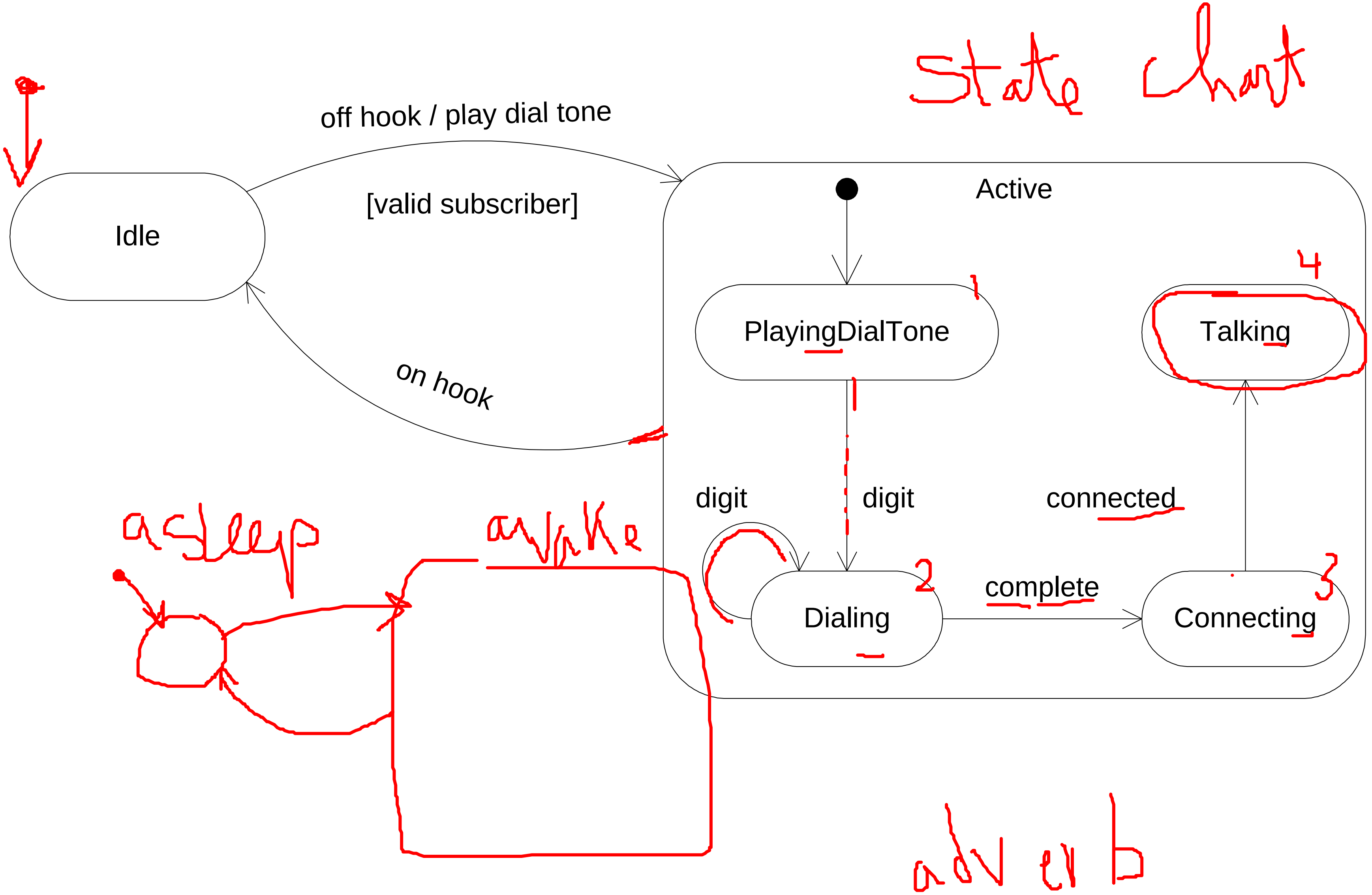
ev / action
[cnd]

Nested States

- A state may be represented as nested substates.
  - In UML, substates are shown by nesting them in a superstate box.
- A substate inherits the transitions of its superstate.
  - Allows succinct state machine diagrams.



Nested states

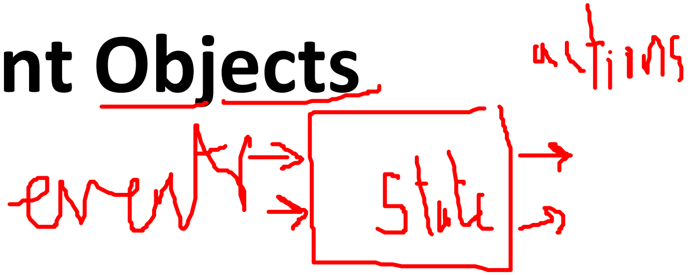


State-Independent vs. State-Dependent

- State-independent (modeless) — type of object that always responds the same way to an event.
- State-dependent (modal) — type of object that reacts differently to events depending on its state or mode.

Use state machine diagrams for modeling state-dependent objects with complex behavior, or to model legal sequences of operations.

Modeling State-dependent Objects



- Complex reactive objects
  - – Physical devices controlled by software
    - E.g., phone, microwave oven, thermostat
  - – Transactions and related business objects
- Protocols and legal sequences
  - – Communication protocols (e.g., TCP)
  - – UI page/window flow or navigation
  - – UI flow controllers or sessions
  - – Use case system operations

Fig. 29.4 Web page navigation modeling

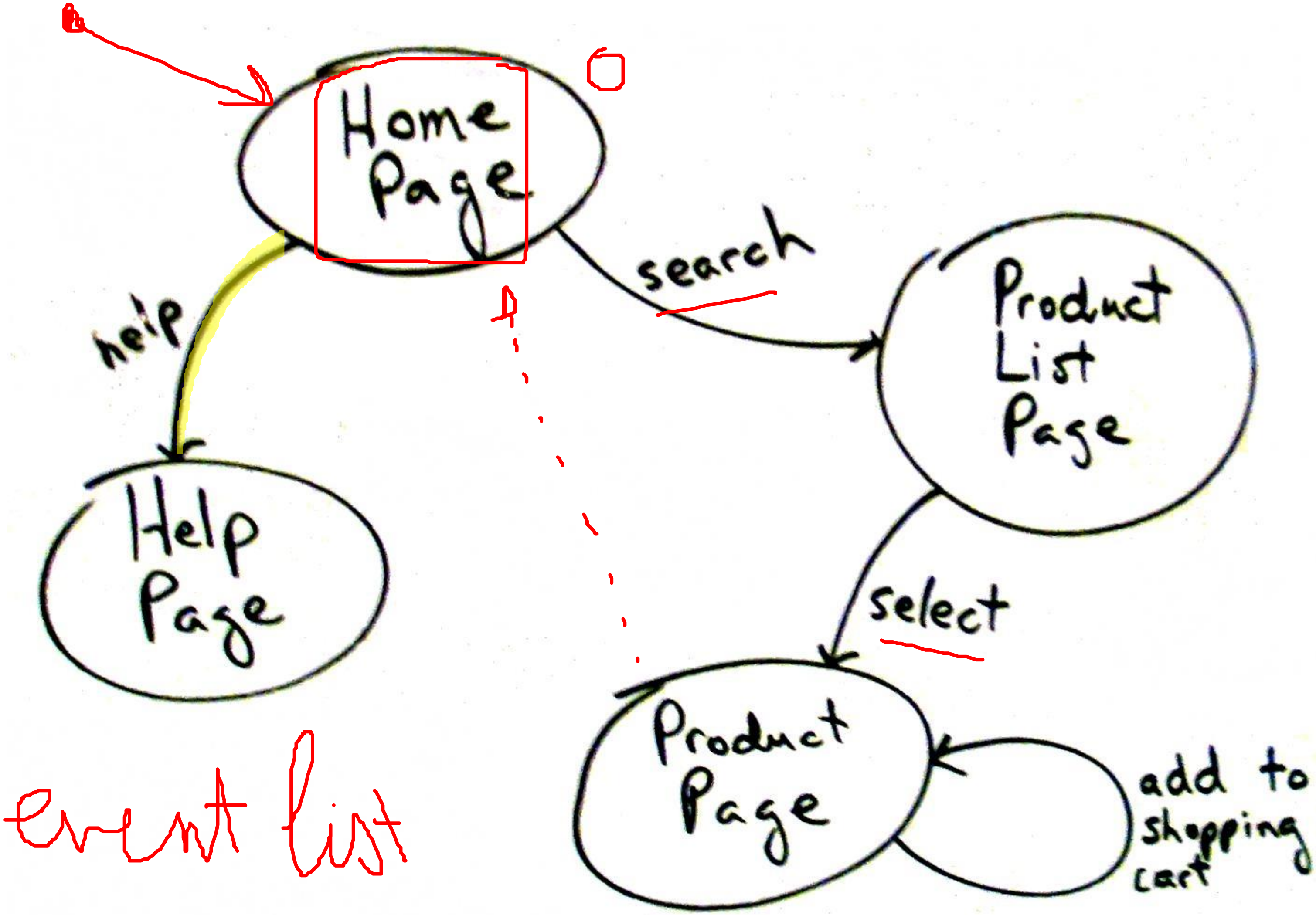
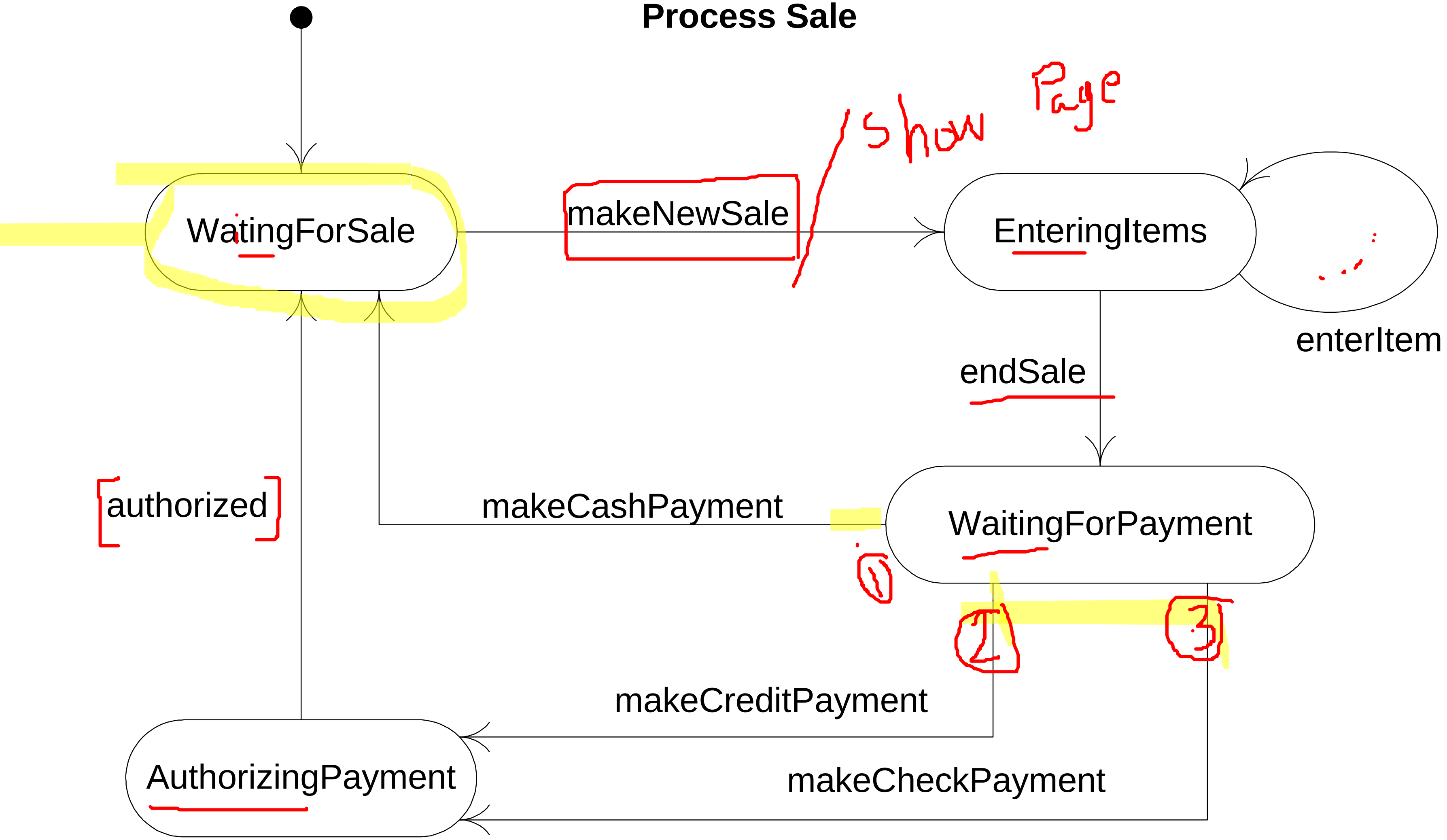
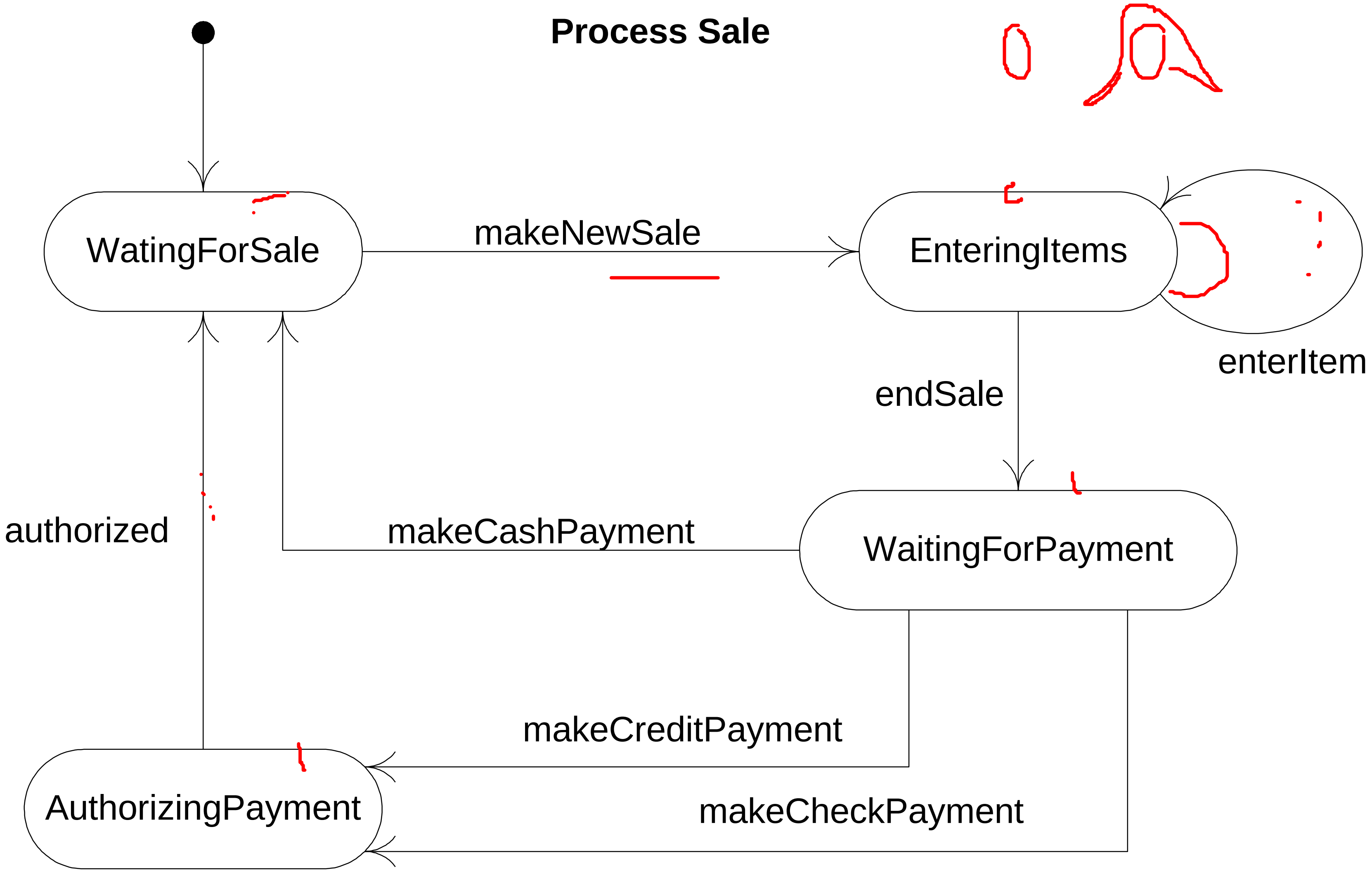


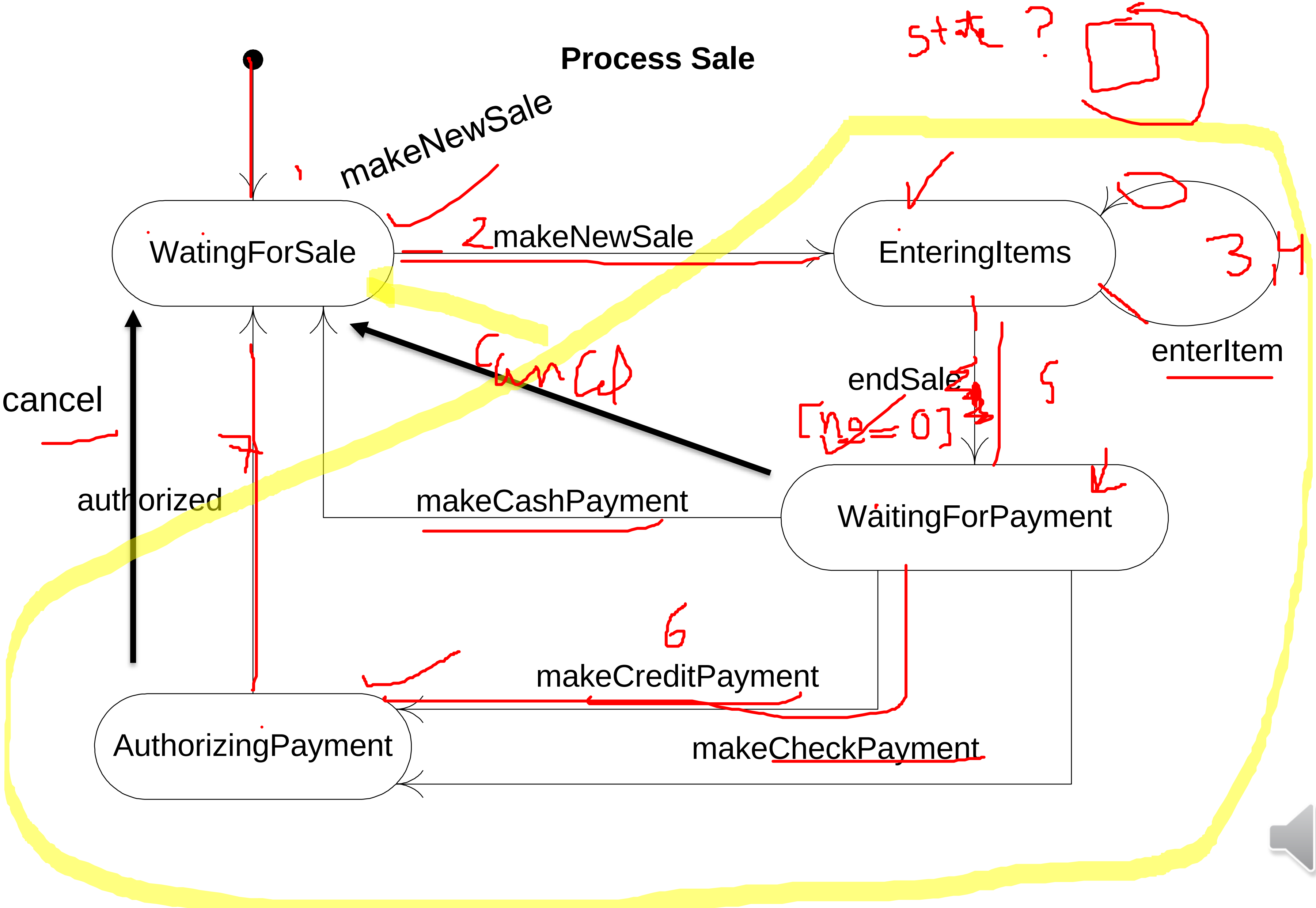
Fig. 29.5 Legal sequence of use case operations



Adding Cancel Events

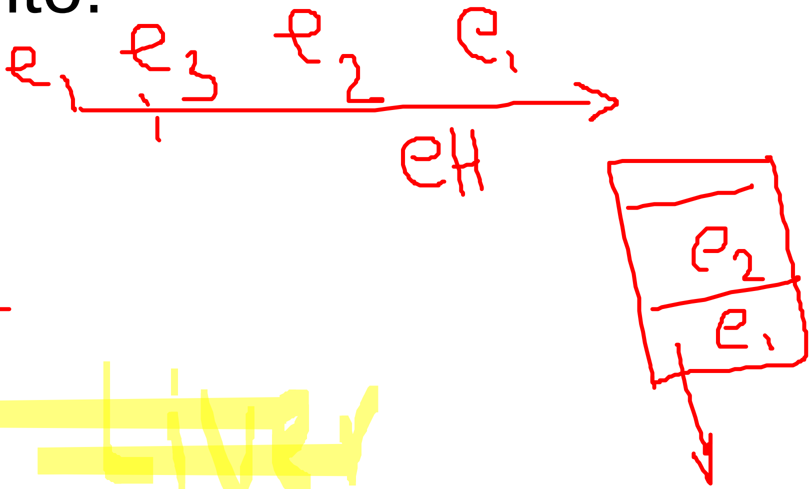


Adding removeitem event, making sure at least one item



Testing State Diagram

- Event list validation
- Translating state diagram into:
  - Event driven code
  - State driven code
- State coverage test
- Event coverage test
- Transition coverage test
- Path coverage test (not applicable all times)
- Model checking/ state model verification against set of temporal properties



Live

Safety

Safety

liveness

SMV

SPIN

TL

Q Q3

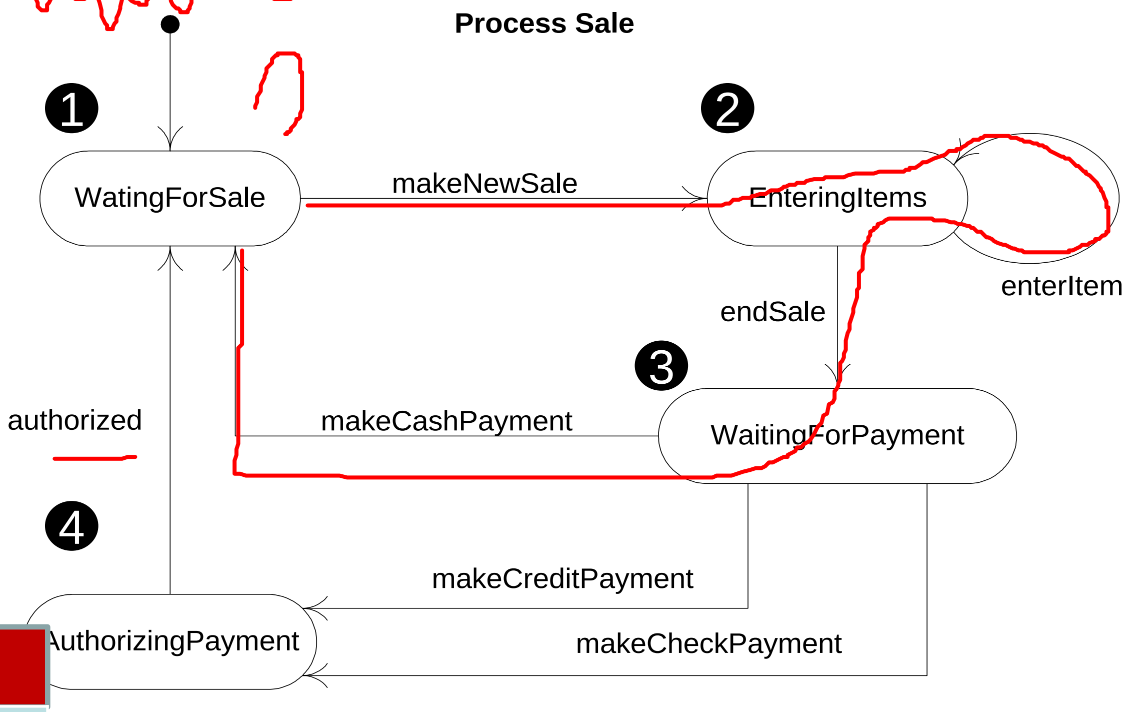
Q Q4

Testing: event list validation

validate

events

<MNS, ES, McrP, Auth>			
state	Event	Next state	comment
1	MNS	2	
2	ES	3	
3	MCrP	4	✓
4	Auth	1	

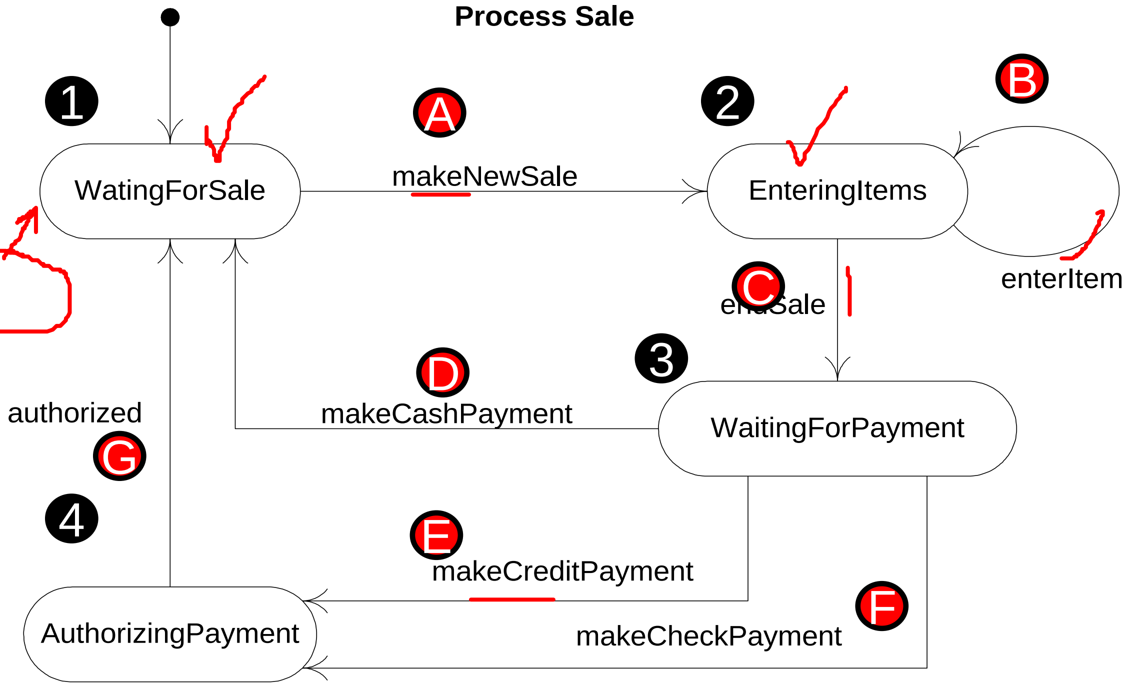


<EI, MNS, EI, ES, McshP>			
state	Event	Next state	comment
1	EI	1	
1	MNS	2	
2	EI	2	
2	ES	3	
3	MCshP	1	

Testing: State coverage

- 4 states need to cover all of them

5



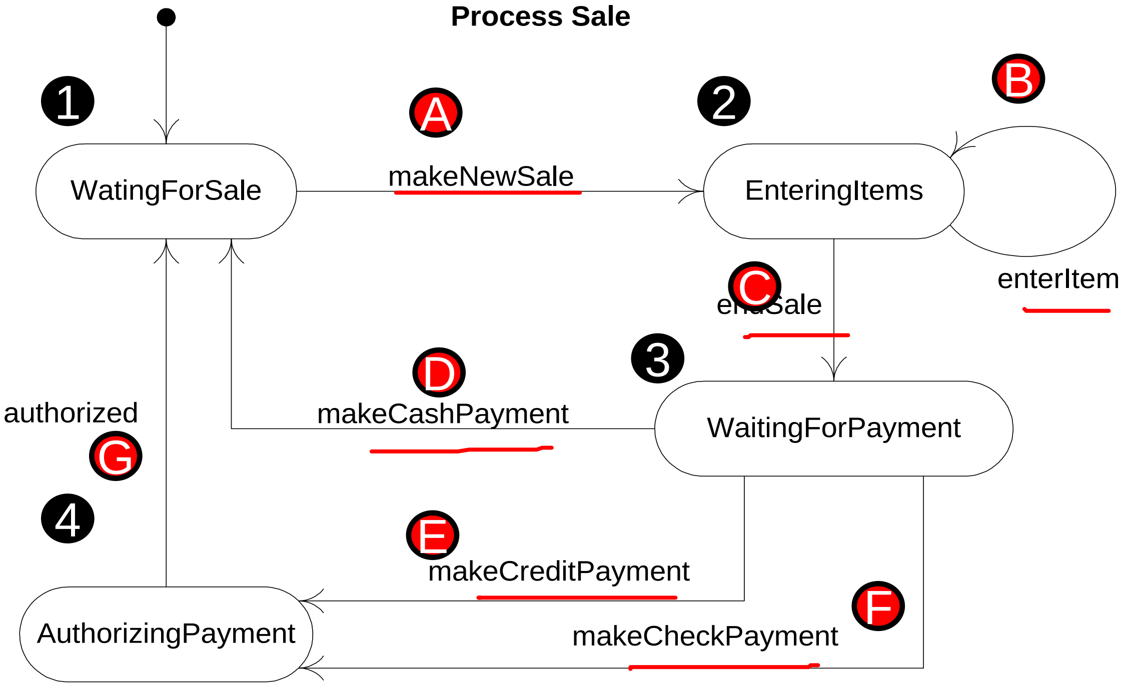
[input, expected]

actual o/p

Test case	Event list	Expected output	States covered
1	MNS	States=<1,2>	❶ ❷
2	MNS, EI,ES	States=<1,2,2,3>	❶ ❷ ❸
3	MNS, EI,ES,MCrP	States=<1,2,2,3,4>	❶ ❷ ❸ ❹

Testing: Event coverage

- 7 events need to cover all of them



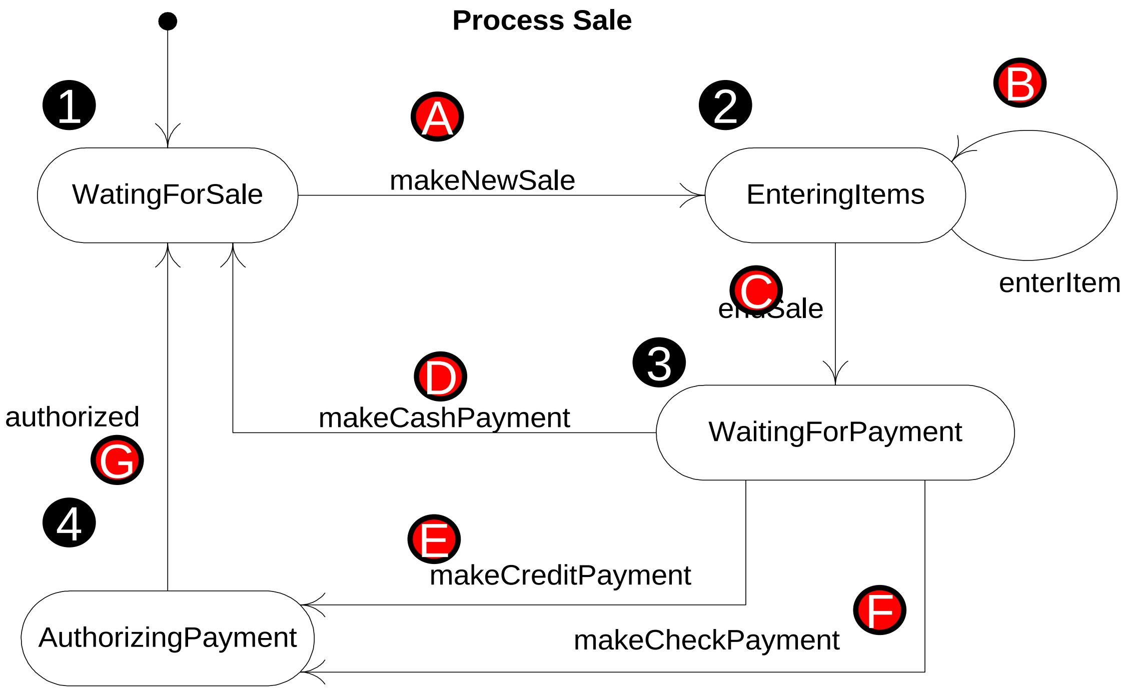
TC1
TC2
TC3

TS

Test case	Event list	Expected output	Events covered
1	MNS	States=<1,2>	A
2	MNS, EI,ES	States=<1,2,2,3>	A B C
3	MNS, EI,ES,MCP	States=<1,2,2,3,4>	A B C E

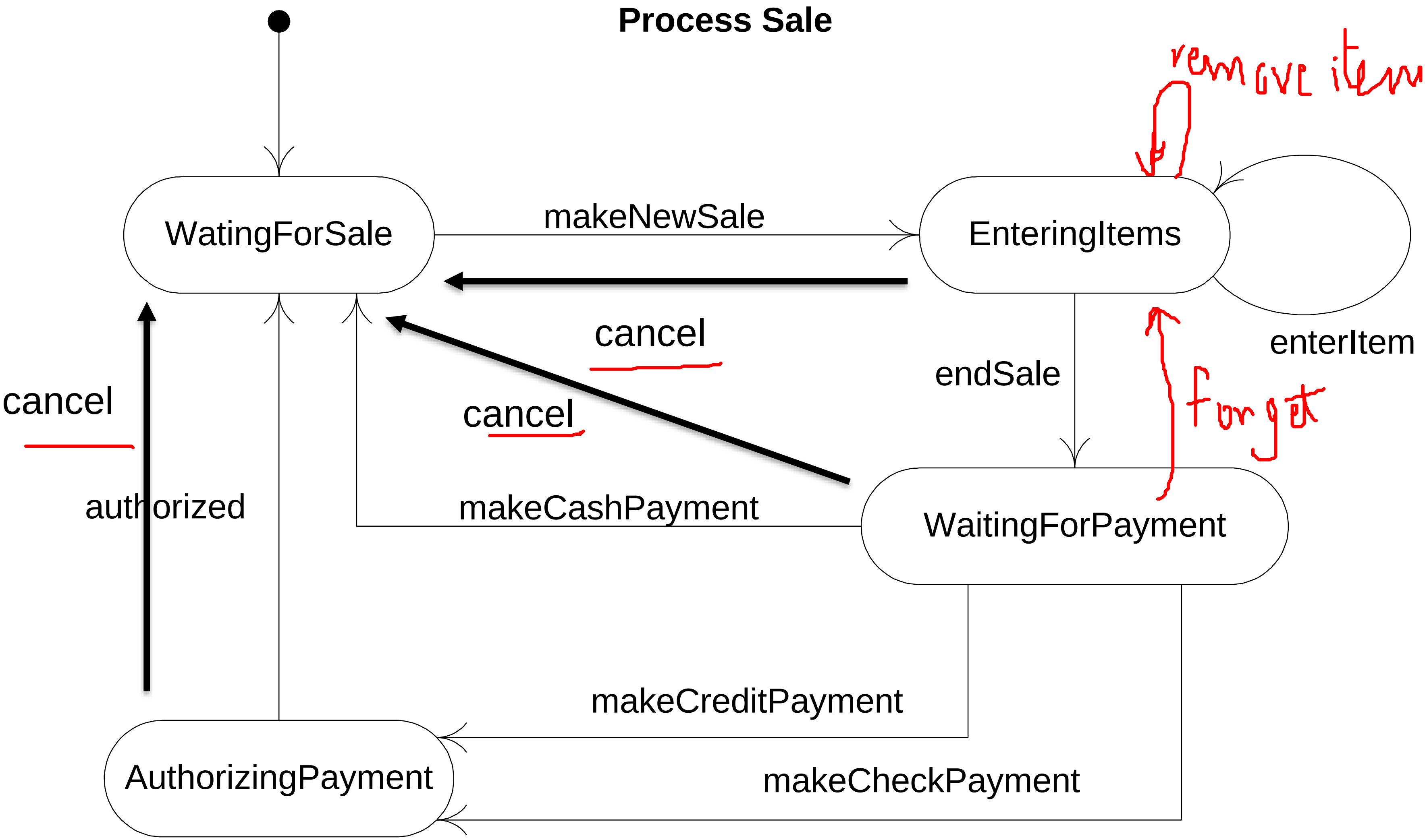
Testing: Transition coverage

- Both are the same because each event found once in one transition



Test case	Event list	Expected output	Transition covered
1	MNS	States=<1,2>	A
2	MNS, EI,ES	States=<1,2,2,3>	A B C
3	MNS, EI,ES,MCP	States=<1,2,2,3,4>	A B C E

Testing event coverage:
one of the three Cancel transitions just need to be tested



Which test

Test	How Easy	Discover/reveal	Must cover all	
State coverage	Very easy	State reachability shallow bugs	States	1
Event coverage	Easy	Deeper Bugs responding to some event	Events	
Transition coverage	Difficult	Subtle bugs including state and event related ones	Transitions	2
Path coverage	Very difficult	Discover dependency between transitions for example problem when transition 3 is done after transition 5 Or transition 2 is carried out five times	Transitions possible sequences	