State Transition Testing

Tran Thi Bich Hanh

Faculty of Information Technology
University of Science, HCMC

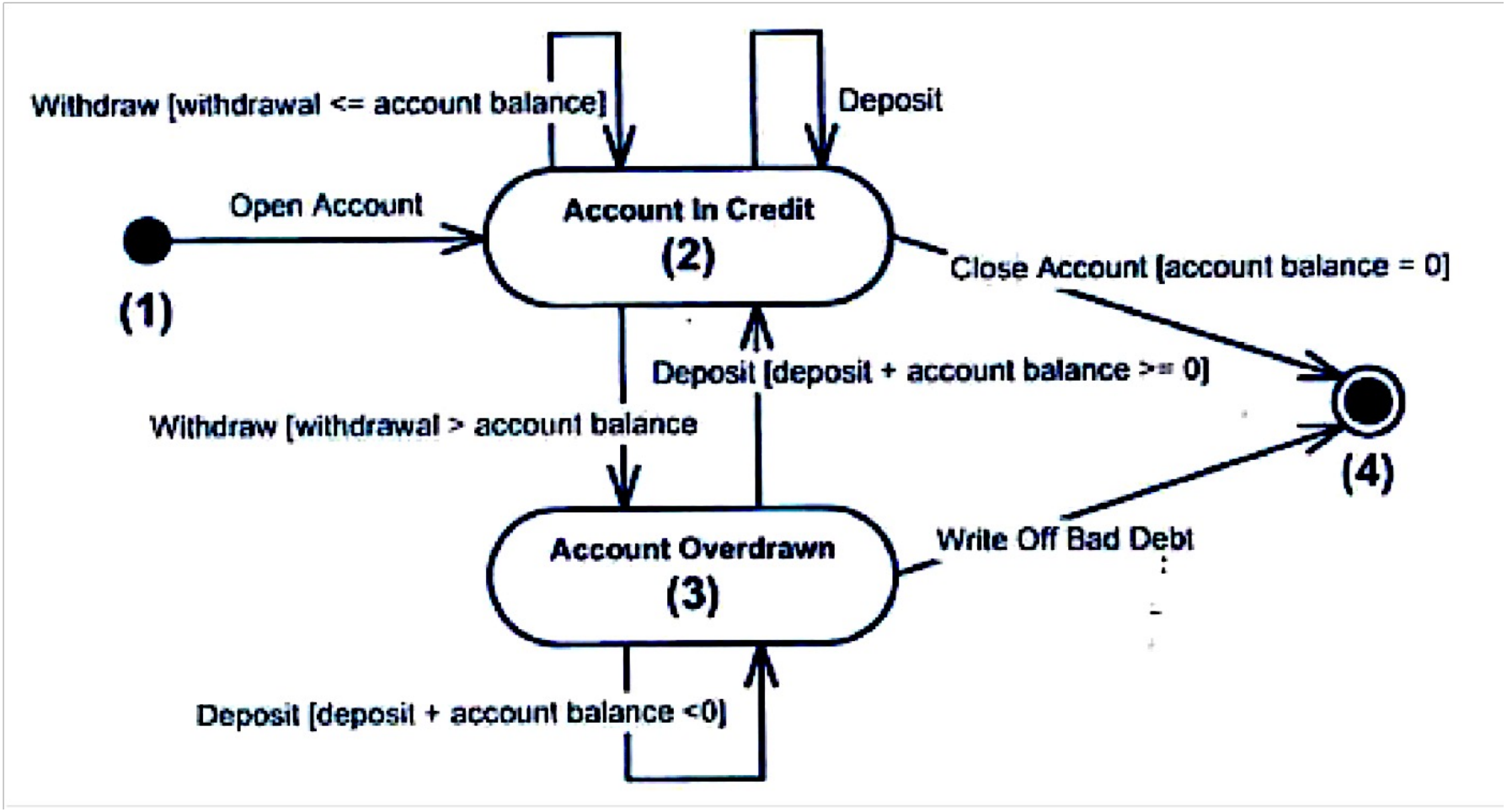
State Transition Testing

- Helpful to **test different system transitions**
- General approach
  - Describe the System Under Test as a “finite state machine” or “state transition diagram”
  - Create State Table based on the diagram to evaluate **Invalid Transitions**

State Transition Diagram

- The **states** that the SUT may occupy
- The **transitions** from one state to another (not all transitions are allowed)
- The **events** that cause a transition (withdrawing money, closing a file)
- The **actions** that result from a transition (an error message…)

State Diagram for a Bank Account



State Transition Table

- List all possible state-transition combinations, not just the valid ones
  - Advantage: Help detect defects in implementation that enable invalid paths from one state to another
  - Disadvantage: state tables become very large very quickly as the number of states and events increases

State Table for Bank Account

Prior State	New State	Valid Transition	Comment
1	1	N	
1	2	Y	New account
1	3	N	Possible negative test case
1	4	N	
2	1	N	
2	2	Y	Deposit and withdraw [withdrawal <= account balance]
2	3	Y	Withdraw [withdrawal > account balance]
2	4	Y	Closed account [account balance=0]
3	1	N	
3	2	Y	Deposit [deposit + account balance >= 0]
3	3	Y	Deposit [deposit + account balance < 0]
3	4	Y	Write Off Bad Debt [account balance < 0]
4	1	N	
4	2	N	Possible negative test case
4	3	N	Possible negative test case
4	4	N	Possible negative test case

Test Cases from State Table

#TC	Precondition (State)	Condition (Event)	Expected Result (Action)	Note
TC1	No account	Open account	Account created with balance >=0	S1 => S2
TC2	No account	Withdraw	Message: Account does not exist	S1 => S3
TC3	No account	Close account	Message: Account does not exist	S1 => S4
TC4	Account with balance >= 0	Deposit D	Balance = balance + D	S2 => S2
TC5	Account with balance >= 0	Withdraw W<=balance	Balance = balance – W >= 0	S2 => S2

Test Cases from State Table

#TC	Precondition (State)	Condition (Event)	Expected Result (Action)	Note
TC6	Balance >=0	Withdraw W>balance	Balance = (balance – W) < 0	S2 => S3
TC7	Balance >=0	Close account	Account closed Balance = 0	S2 => S4
TC8	Balance < 0	Deposit D+Balance>=0	Balance = Balance + D >= 0	S3 => S2
TC9	Balance < 0	Deposit D+Balance<0	Balance = Balance + D < 0	S3 => S3
TC10	Balance < 0	Write bad debit	Account in bad debit	S3 => S4

Test Cases from State Table

#TC	Precondition (State)	Condition (Event)	Expected Result (Action)	Note
TC11	Account in bad debit	Deposit D+Balance>=0	Balance = Balance + D >= 0	S4 => S2
TC12	Account in bad debit	Deposit D+Balance<0	Balance = Balance + D < 0	S4 => S3
TC13	Account closed	Deposit	Message: Account already closed	S4 => S2
TC14	Account in bad debit	Withdraw	Message: Account in bad debit	S4 => S3
TC15	Account closed	Withdraw	Message: Account already closed	S4 => S3
TC16	Account in bad debit	Close	Message: Account in bad debit	S4 => S4
TC17	Account closed	Close	Message: Account already closed	S4 => S4

Exercise

- Behavior of Login
  - Correct Password → Access
  - Incorrect Password 3 times consecutively → Application Closed