

CxStates

Little Smalltalk Exercise:
A dynamically defined state model
not based on the state pattern

presented by
Alfred Wullschleger
Swiss National Bank

The Author

- Smalltalker since 1992
- Project OVID at Fides Informatik (1992-1999)
 - OVID currently in production more than 11 years
- Project OASE at Swiss National Bank (since 1999 in production)
 - Financial Statistics from Swiss Banks and Companies
 - based on Gemstone/S and VisualWorks
 - ongoing development under full production

Motivation

- We needed a user configurable state model
 - so, preferably should not be class based
 - should allow easy communication with the „outer system“ = the system components which are not part of the state model itself
- Solution: CxStates
 - ANSI Event Model as implementation pattern

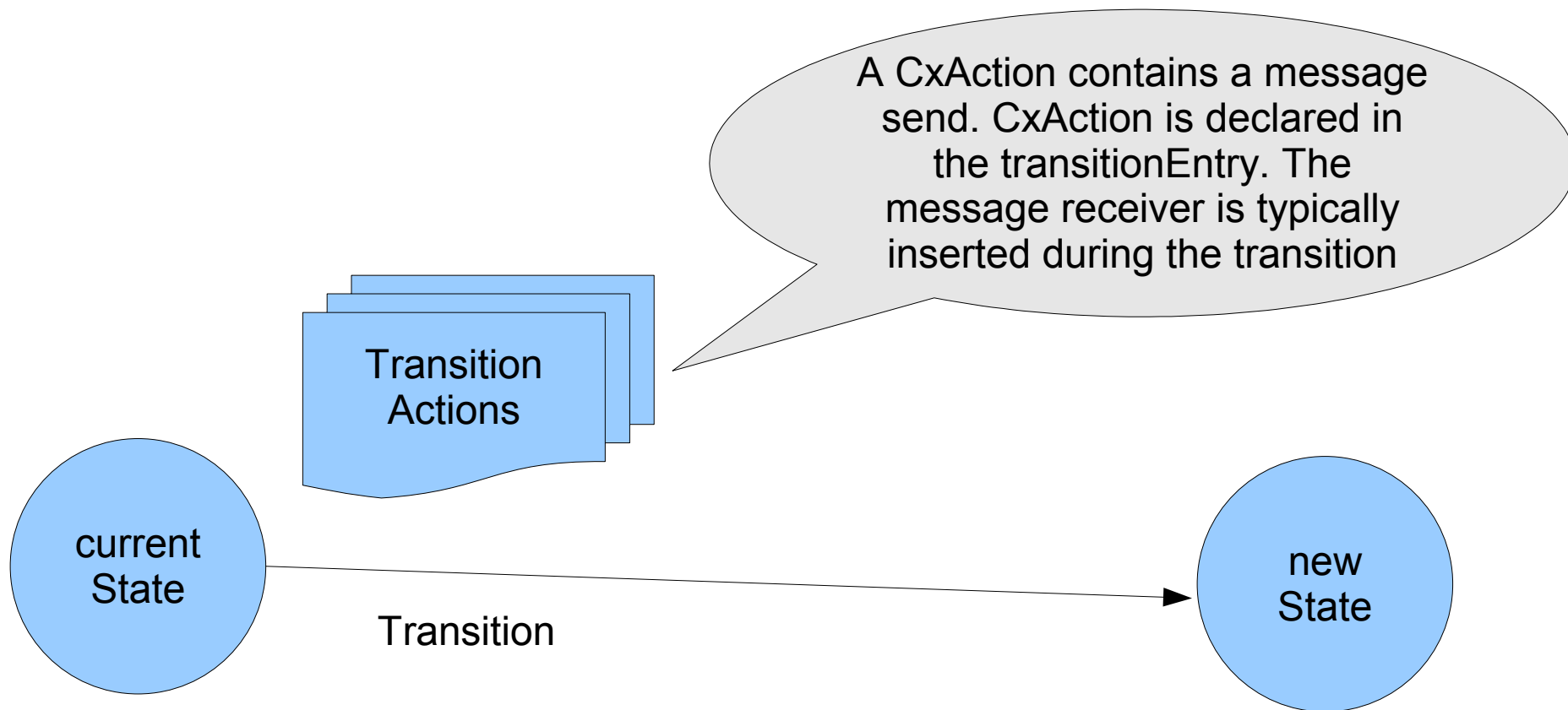
Basic classes (1)

- CxBaseState
 - defines a state
 - has a name
 - has a transitionTable
 - contains legal transitions to other states through transition entries
- CxTransitionEntry
 - defines a transition by a symbol and the new state
 - includes transitionActions

Basic Classes (2)

- CxActionSequence and CxAction
 - for execution of methods in the „outer System“
 - extensible analogous to the event model
- CxStateEnsemble
 - Defines a complete state diagram
 - Defines all legal transition symbols

Action execution



Declaration of actions

- in CxBaseState:
 - whenCxTransition: aTransitionSymbol send: aSelector to: aReceiver
 - adds a CxAction to the TransitionActionSequence
 - aReceiver can be nil: the receiver can be dynamically set when the transition is executed
 - polymorphically delegated to CxTransitionEntry

CxBaseState>>doTransition:

- Executes the defined CxActions
 - By delegation to CxTransitionEntry>>doTransition: ,
and finally to CxAction>>doTransition:
- Returns
 - nil, when the transition is illegal, state unchanged
 - the new state otherwise

Modification of StateEnsembles

- add new state
- add new transitionSymbol
- change transitions
 - fully dynamic change of models under full production
 - uses become: for mutation of of states
 - practical experience with a model with 36 states and 42 transitionSymbols using **415** transitions
 - very easy, to make errors!
 - but, very easy, to correct errors!

Summary

- CxStates are instance based
- CxStateEnsemble can be extended very easily under full production
- highly metaprogrammed:
 - Difficult to implement in Non-Smalltalk-environments
-
- =====> Questions?