

smacc::ISmaccStateMachine

nh_
 # private_nh_
 # timer_
 # stateMachinePub_
 # stateMachineStatusPub_
 # transitionLogPub_
 # transitionHistoryService_
 # currentState_
 # currentStateInfo_
 # status_msg_
 # orthogonals_
 - m_mutex_
 - eventQueueMutex_
 - stateMachineCurrentAction
 - stateCallbackConnections
 - globalData_
 - transitionLogHistory_
 - runMode_
 - signalDetector_
 - stateSeqCounter_
 - stateMachineInfo_

+ ISmaccStateMachine()
 + ~ISmaccStateMachine()
 + reset()
 + stop()
 + eStop()
 + getOrthogonal()
 + getOrthogonals()
 + requiresComponent()
 + postEvent()
 + postEvent()
 + getTransitionLogHistory()
 + getGlobalSMDData()
 + setGlobalSMDData()
 + mapBehavior()
 + getStateMachineName()
 + state_machine_visualization()
 + getCurrentStateInfo()
 + publishTransition()
 + onInitialize()
 + getTransitionLogHistory()
 + createSignalConnection()
 + notifyOnStateEntryStart()
 + notifyOnStateEntryEnd()
 + notifyOnRuntimeConfigured()
 + notifyOnStateExiting()
 + notifyOnStateExited()
 + notifyOnRuntimeConfiguration
 Finished()
 + getCurrentStateCounter()
 + getCurrentState()
 + getStateMachineInfo()
 + buildStateMachineInfo()
 + getNode()
 # checkStateMachineConsistence()
 # initializeROS()
 # onInitialized()
 # createOrthogonal()
 # getParam()
 # setParam()
 # param()
 - lockStateMachine()
 - unlockStateMachine()
 - propagateEventToStateReactors()
 - updateStatusMessage()

boost::statechart::
asynchronous_state_machine
< DerivedStateMachine, InitialState
Type, SmaccFifoScheduler, SmaccAllocator >

smacc::SmaccStateMachine
Base< DerivedStateMachine,
InitialStateType >

+ SmaccStateMachineBase()
 + ~SmaccStateMachineBase()
 + reset()
 + stop()
 + eStop()
 + initiate_impl()

boost::statechart::
asynchronous_state_machine
< SmAtomicServices, State1,
SmaccFifoScheduler, SmaccAllocator >

smacc::SmaccStateMachine
Base< SmAtomicServices,
State1 >

+ SmaccStateMachineBase()
 + ~SmaccStateMachineBase()
 + reset()
 + stop()
 + eStop()
 + initiate_impl()

sm_atomic_services
::SmAtomicServices

+ onInitialize()

< SmAtomicServices,
State1 >