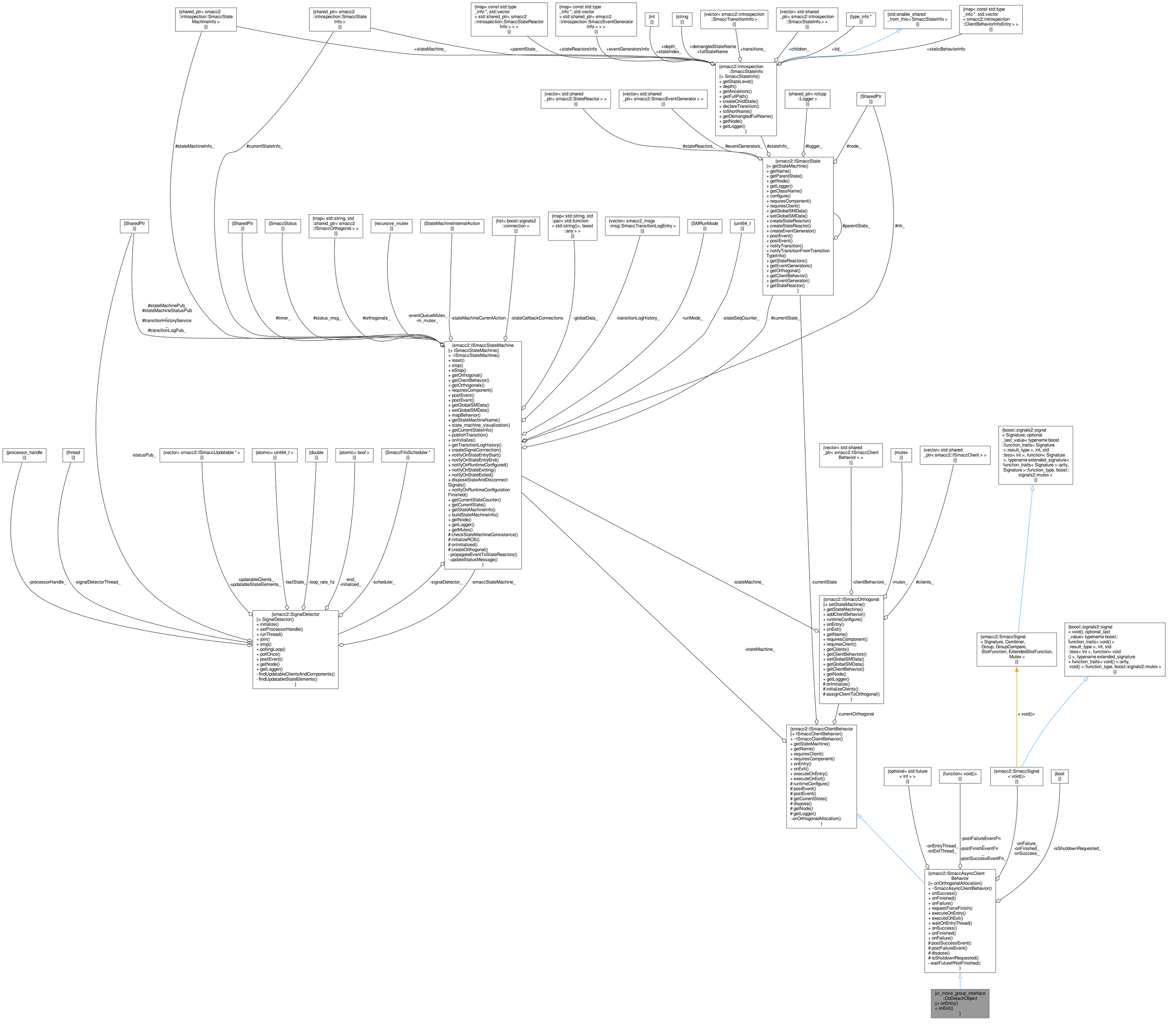
{shared_ptr< smacc2 ::introspection::SmaccState MachineInfo > ||}
{shared_ptr< smacc2 ::introspection::SmaccState Info > ||}
{map< const std::type _info *, std::vector < std::shared_ptr< smacc2 ::introspection::SmaccStateReactor Info > > > ||}
{map< const std::type _info *, std::vector < std::shared_ptr< smacc2 ::introspection::SmaccEventGenerator Info > > > ||}
{int ||}
{string ||}
{vector< smacc2::introspection ::SmaccTransitionInfo > ||}
{vector< std::shared _ptr< smacc2::introspection ::SmaccStateInfo > > ||}
{type_info * ||}
{std::enable_shared _from_this< SmaccStateInfo > ||}
{map< const std::type _info *, std::vector < smacc2::introspection ::ClientBehaviorInfoEntry > > ||}
+stateMachine_
+parentState_
+stateReactorsInfo
+eventGeneratorsInfo
+depth_ +stateIndex_
+demangledStateName +fullStateName
+transitions_
+children_
+tid_
+staticBehaviorInfo
{smacc2::introspection ::SmaccStateInfo ||+ SmaccStateInfo() + getStateLevel() + depth() + getAncestors() + getFullPath() + createChildState() + declareTransition() + toShortName() + getDemangledFullName() + getNode() + getLogger() }
{vector< std::shared _ptr< smacc2::StateReactor > > ||}
{vector< std::shared _ptr< smacc2::SmaccEventGenerator > > ||}
{shared_ptr< rclcpp ::Logger > ||}
{SharedPtr ||}
#stateMachineInfo_
#currentStateInfo_
#stateReactors_
#eventGenerators_
#stateInfo_
#logger_
#node_
{smacc2::ISmaccState ||+ getStateMachine() + getName() + getParentState() + getNode() + getLogger() + getClassName() + configure() + requiresComponent() + requiresClient() + getGlobalSMData() + setGlobalSMData() + createStateReactor() + createStateReactor() + createEventGenerator() + postEvent() + postEvent() + notifyTransition() + notifyTransitionFromTransition TypeInfo() + getStateReactors() + getEventGenerators() + getOrthogonal() + getClientBehavior() + getEventGenerator() + getStateReactor() }
#parentState_
#nh_
{SharedPtr ||}
{SharedPtr ||}
{SmaccStatus ||}
{map< std::string, std ::shared_ptr< smacc2 ::ISmaccOrthogonal > > ||}
{recursive_mutex ||}
{StateMachineInternalAction ||}
{list< boost::signals2 ::connection > ||}
{map< std::string, std ::pair< std::function < std::string()>, boost ::any > > ||}
{vector< smacc2_msgs ::msg::SmaccTransitionLogEntry > ||}
{SMRunMode ||}
{uint64_t ||}
#stateMachinePub_ #stateMachineStatusPub
#transitionHistoryService
#transitionLogPub_
#timer_
#status_msg_
#orthogonals_
-eventQueueMutex_ -m_mutex_
-stateMachineCurrentAction
-stateCallbackConnections
-globalData_
-transitionLogHistory_
-runMode_
-stateSeqCounter_
#currentState_
{smacc2::ISmaccStateMachine ||+ ISmaccStateMachine() + ~ISmaccStateMachine() + reset() + stop() + eStop() + getOrthogonal() + getClientBehavior() + getOrthogonals() + requiresComponent() + postEvent() + postEvent() + getGlobalSMData() + setGlobalSMData() + mapBehavior() + getStateMachineName() + state_machine_visualization() + getCurrentStateInfo() + publishTransition() + onInitialize() + getTransitionLogHistory() + createSignalConnection() + notifyOnStateEntryStart() + notifyOnStateEntryEnd() + notifyOnRuntimeConfigured() + notifyOnStateExitting() + notifyOnStateExited() + disposeStateAndDisconnect Signals() + notifyOnRuntimeConfiguration Finished() + getCurrentStateCounter() + getCurrentState() + getStateMachineInfo() + buildStateMachineInfo() + getNode() + getLogger() + getMutex() # checkStateMachineConsistence() # initializeROS() # onInitialized() # createOrthogonal() - propagateEventToStateReactors() - updateStatusMessage() }
{processor_handle ||}
{thread ||}
-statusPub_
{vector< smacc2::ISmaccUpdatable * > ||}
{atomic< uint64_t > ||}
{double ||}
{atomic< bool > ||}
{SmaccFifoScheduler * ||}
-processorHandle_
-signalDetectorThread_
-updatableClients_ -updatableStateElements_
-lastState_
-loop_rate_hz
-end_ -initialized_
-scheduler_
-signalDetector_
-smaccStateMachine_
{smacc2::SignalDetector ||+ SignalDetector() + initialize() + setProcessorHandle() + runThread() + join() + stop() + pollingLoop() + pollOnce() + postEvent() + getNode() + getLogger() - findUpdatableClientsAndComponents() - findUpdatableStateElements() }
{vector< std::shared _ptr< smacc2::ISmaccClient Behavior > > ||}
{mutex ||}
{vector< std::shared _ptr< smacc2::ISmaccClient > > ||}
{boost::signals2::signal < Signature, optional _last_value< typename boost ::function_traits< Signature >::result_type >, int, std ::less< int >, function< Signature >, typename extended_signature< function_traits< Signature >::arity, Signature >::function_type, boost:: signals2::mutex > ||}
-stateMachine_
-currentState
-clientBehaviors_
-mutex_
#clients_
{smacc2::ISmaccOrthogonal ||+ setStateMachine() + getStateMachine() + addClientBehavior() + runtimeConfigure() + onEntry() + onExit() + getName() + requiresComponent() + requiresClient() + getClients() + getClientBehaviors() + setGlobalSMData() + getGlobalSMData() + getClientBehavior() + getNode() + getLogger() # onInitialize() # initializeClients() # assignClientToOrthogonal() }
{smacc2::SmaccSignal < Signature, Combiner, Group, GroupCompare, SlotFunction, ExtendedSlotFunction, Mutex > ||}
{boost::signals2::signal < void(), optional_last _value< typename boost:: function_traits< void() > ::result_type >, int, std ::less< int >, function< void () >, typename extended_signature < function_traits< void() >::arity, void() >::function_type, boost::signals2::mutex > ||}
< void()>
-currentOrthogonal
{smacc2::ISmaccClientBehavior ||+ ISmaccClientBehavior() + ~ISmaccClientBehavior() + getStateMachine() + getName() + requiresClient() + requiresComponent() + onEntry() + onExit() + executeOnEntry() + executeOnExit() # runtimeConfigure() # postEvent() # postEvent() # getCurrentState() # dispose() # getNode() # getLogger() - onOrthogonalAllocation() }
{optional< std::future < int > > ||}
{function< void()> ||}
{smacc2::SmaccSignal < void()> ||}
{bool ||}
-onEntryThread_ -onExitThread_
-postFailureEventFn
-postFinishEventFn
-postSuccessEventFn_
-onFailure_ -onFinished_ -onSuccess_
-isShutdownRequested_
{smacc2::SmaccAsyncClient Behavior ||+ onOrthogonalAllocation() + ~SmaccAsyncClientBehavior() + onSuccess() + onFinished() + onFailure() + requestForceFinish() + executeOnEntry() + executeOnExit() + waitOnEntryThread() + onSuccess() + onFinished() + onFailure() # postSuccessEvent() # postFailureEvent() # dispose() # isShutdownRequested() - waitFutureIfNotFinished() }
{cl_move_group_interface ::CbDetachObject ||+ onEntry() + onExit() }