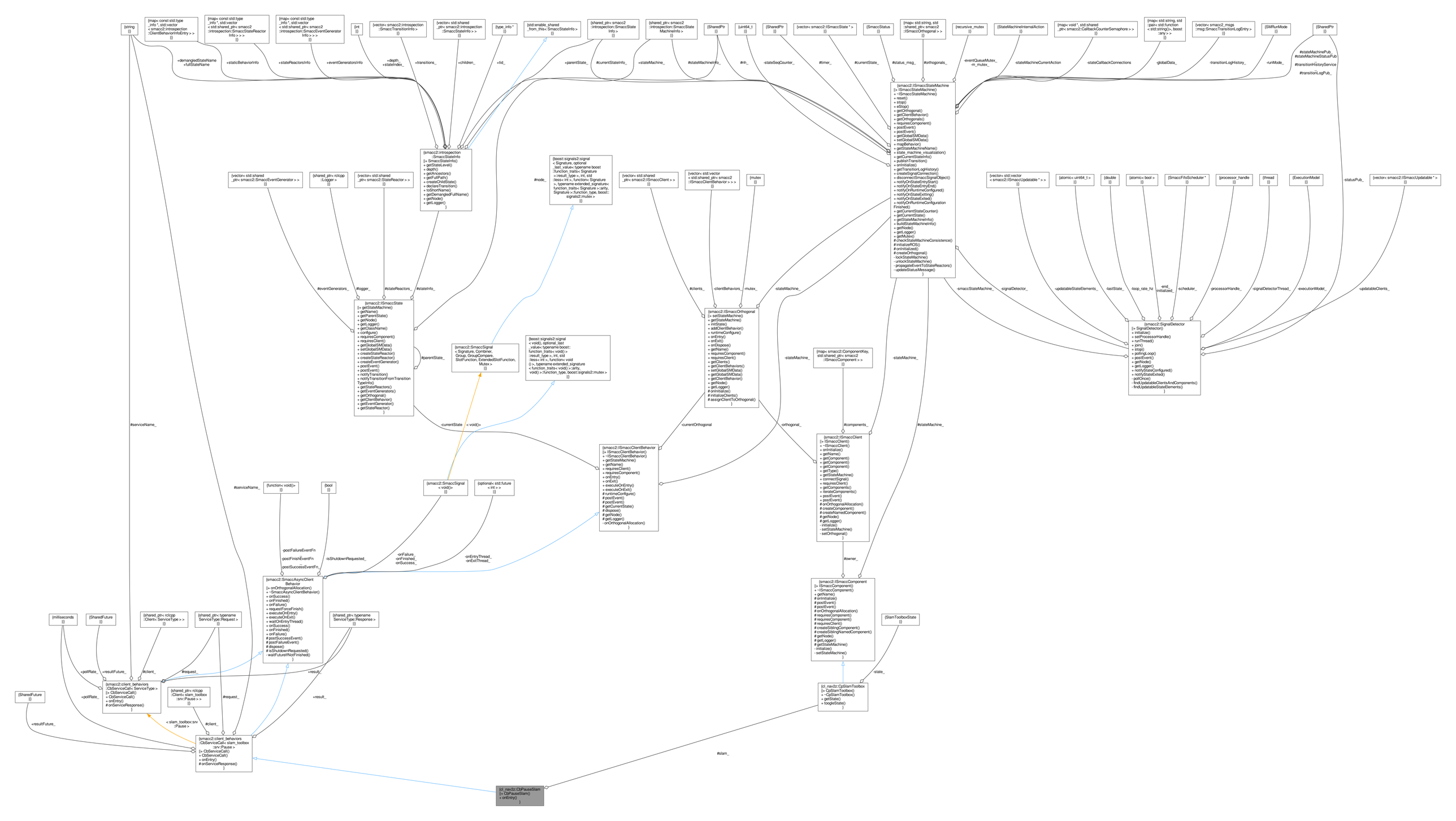
{string }
{map< const std::type_info *, std::vector< smacc2::introspection::ClientBehaviorInfoEntry > > }
{map< const std::type_info *, std::vector< std::shared_ptr< smacc2::introspection::SmaccStateReactorInfo > > > }
{map< const std::type_info *, std::vector< std::shared_ptr< smacc2::introspection::SmaccEventGeneratorInfo > > > }
{int }
{vector< smacc2::introspection::SmaccTransitionInfo > }
{vector< std::shared_ptr< smacc2::introspection::SmaccStateInfo > > }
{type_info * }
{std::enable_shared_from_this< SmaccStateInfo > }
{shared_ptr< smacc2::introspection::SmaccStateInfo > }
{shared_ptr< smacc2::introspection::SmaccStateMachineInfo > }
{SharedPtr }
{uint64_t }
{SharedPtr }
{vector< smacc2::ISmaccState * > }
{SmaccStatus }
{map< std::string, std::shared_ptr< smacc2::ISmaccOrthogonal > > }
{recursive_mutex }
{StateMachineInternalAction }
{map< void *, std::shared_ptr< smacc2::CallbackCounterSemaphore > > }
{map< std::string, std::pair< std::function< std::string()>, boost::any > > }
{vector< smacc2_msgs::msg::SmaccTransitionLogEntry > }
{SMFRunMode }
{SharedPtr }
+demangledStateName
+fullStateName
+staticBehaviorInfo
+stateReactorsInfo
+eventGeneratorsInfo
+depth_
+stateIndex_
+transitions_
+children_
+tid_
+parentState_
#currentStateInfo_
+stateMachine_
#stateMachineInfo_
#nh_
-stateSeqCounter_
#timer_
#currentState_
#status_msg_
#orthogonals_
-eventQueueMutex_
-m_mutex_
-stateMachineCurrentAction
-stateCallbackConnections
-globalData_
-transitionLogHistory_
-runMode_
#stateMachinePub_
#stateMachineStatusPub
#transitionHistoryService
#transitionLogPub_
{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()
+ disconnectSmaccSignalObject()
+ notifyOnStateEntryStart()
+ notifyOnStateEntryEnd()
+ notifyOnRuntimeConfigured()
+ notifyOnStateExiting()
+ notifyOnStateExited()
+ notifyOnRuntimeConfigurationFinished()
+ getCurrentStateCounter()
+ getCurrentState()
+ getStateMachineInfo()
+ builtStateMachineInfo()
+ getNode()
+ getLogger()
+ getMutex()
# checkStateMachineConsistence()
# initializeROS()
# onInitialized()
# createOrthogonal()
- lockStateMachine()
- unlockStateMachine()
- propagateEventToStateReactors()
- updateStatusMessage()
}
{smacc2::introspection::SmaccStateInfo
|+ SmaccStateInfo()
+ getStateLevel()
+ depth()
+ getAncestors()
+ getFullPath()
+ createChildState()
+ declareTransition()
+ toShortName()
+ getDemangledFullName()
+ getNode()
+ getLogger()
}
{vector< std::shared_ptr< smacc2::SmaccEventGenerator > > }
{shared_ptr< rclcpp::Logger > }
{vector< std::shared_ptr< smacc2::StateReactor > > }
#node_
{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 >
}
{vector< std::shared_ptr< smacc2::ISmaccClient > > }
{vector< std::vector
< std::shared_ptr< smacc2
::ISmaccClientBehavior > > > }
{mutex }
{vector< std::vector
< smacc2::ISmaccUpdatable * > > }
{atomic< uint64_t > }
{double }
{atomic< bool > }
{SmaccFifoScheduler * }
{processor_handle }
{thread }
{IExecutionModel }
-statusPub_
{vector< smacc2::ISmaccUpdatable * > }
-eventGenerators_
#logger_
#stateReactors_
#stateInfo_
-clients_
-clientBehaviors_
-mutex_
-stateMachine_
-smaccStateMachine_
-signalDetector_
-updatableStateElements_
-lastState_
-loop_rate_hz
-end_initialized_
-scheduler_
-processorHandle_
-signalDetectorThread_
-executionModel_
-updatableClients_
{smacc2::ISmaccState
|+ getStateMachine()
+ getName()
+ getParentState()
+ getNode()
+ getLogger()
+ getClassName()
+ configure()
+ requiresComponent()
+ requiresClient()
+ getGlobalSMData()
+ setGlobalSMData()
+ createStateReactor()
+ createStateReactor()
+ createEventGenerator()
+ postEvent()
+ postEvent()
+ notifyTransition()
+ notifyTransitionFromTransitionTypeInfo()
+ getStateReactors()
+ getEventGenerators()
+ getOrthogonal()
+ getClientBehavior()
+ getEventGenerator()
+ getStateReactor()
}
#parentState_
{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 >
|}
{smacc2::ISmaccOrthogonal
|+ setStateMachine()
+ getStateMachine()
+ initState()
+ addClientBehavior()
+ runtimeConfigure()
+ onEntry()
+ onExit()
+ onDispose()
+ getName()
+ requiresComponent()
+ requiresClient()
+ getClients()
+ getClientBehaviors()
+ setGlobalSMData()
+ getGlobalSMData()
+ getClientBehavior()
+ getNode()
+ getLogger()
# onInitialize()
# initializeClients()
# assignClientToOrthogonal()
}
{map< smacc2::ComponentKey,
std::shared_ptr< smacc2
::ISmaccComponent > >
|}
{SignalDetector
|+ SignalDetector()
+ initialize()
+ setProcessorHandle()
+ runThread()
+ join()
+ stop()
+ pollingLoop()
+ postEvent()
+ getNode()
+ getLogger()
+ notifyStateConfigured()
+ notifyStateExited()
+ pollOnce()
- findUpdatableClientsAndComponents()
- findUpdatableStateElements()
}
#serviceName_
-currentState
< void()>
-currentOrthogonal
-orthogonal_
-stateMachine_
-stateMachine_
#stateMachine_
#components_
{smacc2::ISmaccClientBehavior
|+ ISmaccClientBehavior()
+ ~ISmaccClientBehavior()
+ getStateMachine()
+ getName()
+ requiresClient()
+ requiresComponent()
+ onEntry()
+ onExit()
+ executeOnEntry()
+ executeOnExit()
# runtimeConfigure()
# postEvent()
# postEvent()
# getCurrentState()
# dispose()
# getNode()
# getLogger()
- onOrthogonalAllocation()
}
{smacc2::ISmaccClient
|+ ISmaccClient()
+ ~ISmaccClient()
+ onInitialize()
+ getName()
+ getComponent()
+ getComponent()
+ getComponent()
+ getType()
+ getStateMachine()
+ connectSignal()
+ requiresClient()
+ getComponents()
+ iterateComponents()
+ postEvent()
+ postEvent()
# onOrthogonalAllocation()
# createComponent()
# createNamedComponent()
# getNode()
# getLogger()
- initialize()
- setStateMachine()
- setOrthogonal()
}
{function< void()>
|}
{bool
|}
{smacc2::SmaccSignal
< void()>
|}
{optional< std::future
< int > >
|}
-postFailureEventFn
-postFinishEventFn
-postSuccessEventFn
-isShutdownRequested_
-onFailure_
-onFinished_
-onSuccess_
-onEntryThread_
-onExitThread_
{smacc2::SmaccAsyncClientBehavior
|+ onOrthogonalAllocation()
+ ~SmaccAsyncClientBehavior()
+ onSuccess()
+ onFinished()
+ onFailure()
+ requestForceFinish()
+ executeOnEntry()
+ executeOnExit()
+ waitOnEntryThread()
+ onSuccess()
+ onFinished()
+ onFailure()
# postSuccessEvent()
# postFailureEvent()
# dispose()
# isShutdownRequested()
- waitFutureIfNotFinished()
}
#owner_
{smacc2::ISmaccComponent
|+ ISmaccComponent()
+ ~ISmaccComponent()
+ getName()
# onInitialize()
# postEvent()
# postEvent()
# onOrthogonalAllocation()
# requiresComponent()
# requiresComponent()
# requiresClient()
# createSiblingComponent()
# createSiblingNamedComponent()
# getNode()
# getLogger()
# getStateMachine()
- initialize()
- setStateMachine()
}
{milliseconds }
{SharedFuture }
{shared_ptr< rclcpp::Client< ServiceType > > }
{shared_ptr< typename ServiceType::Request > }
{shared_ptr< typename ServiceType::Response > }
{SlamToolboxState }
+pollRate_
+resultFuture_
#client_
#request_
+result_
-state_
{SharedFuture }
+pollRate_
{shared_ptr< rclcpp::Client< slam_toolbox::srv::Pause > > }
{cl_nav2z::CpSlamToolbox
|+ CpSlamToolbox()
+ ~CpSlamToolbox()
+ getState()
+ toogleState()
}
+resultFuture_
< slam_toolbox::srv::Pause >
#client_
{smacc2::client_behaviors::CbServiceCall< ServiceType >
|+ CbServiceCall()
+ CbServiceCall()
+ onEntry()
# onServiceResponse()
}
{smacc2::client_behaviors::CbServiceCall< slam_toolbox::srv::Pause >
|+ CbServiceCall()
+ CbServiceCall()
+ onEntry()
# onServiceResponse()
}
#slam_
{cl_nav2z::CbPauseSlam
|+ CbPauseSlam()
+ onEntry()
}