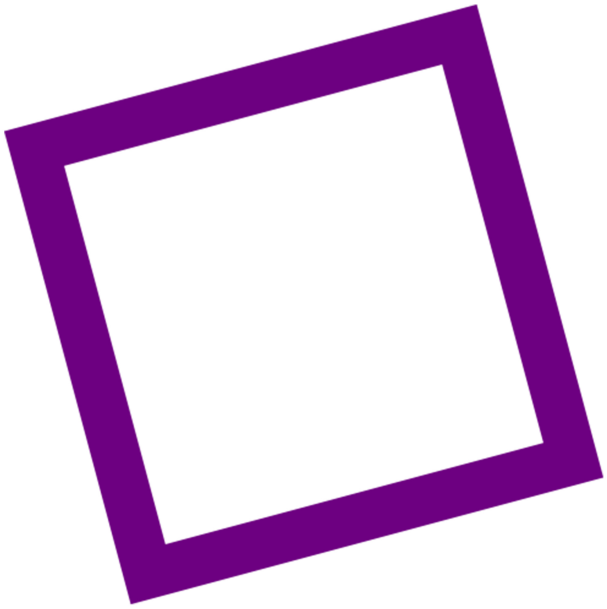


HAMPEL SOFTWARE ENGINEERING

TEMPLATE

HAMPEL SOFTWARE ENGINEERING

Version 3.4.0 (2026-02-26)

TABLE OF CONTENTS

Document generated automatically!

This document was created fully automated from the actual LabVIEW Source Code of this project using the Release Automation Tools of Hampel Software Engineering.

The Release Automation Tools (RAT) help automate the validating, testing, documenting, building, packaging and publishing of your projects. Built-in support for Git lets you trigger our tools from your repository, via GitLab CI/CD or Azure DevOps amongst others.

For a more detailed overview of what these tools do, see https://rat.hampel-soft.com/, where you can find information on the available tools, how we automate them using GitLab CI, when the next scheduled webinars are on, and how you can run those tools on your own servers using a commercial license for RAT.

CHAPTER 1. PROJECT DESCRIPTION

This application template showcases our UI framework for applications with graphical user interfaces. We use this template as a container which loads the project-specific DQMH modules dynamically from a configuration file.

1.1. UI FRAMEWORK

The UI framework helps us with:

- reusing UI management code
- project-specific UI layouts
- displaying a fancy splash screen
- populating the Runtime Menu dynamically
- populating a navigation module dynamically
- generating an event log for debugging

…and much more!

The UI framework is built on top of the HSE Libraries and consists of a collection of DQMH modules and a few helper VIs.

1.1.1. FRAMEWORK MODULES

These DQMH modules are designed to be reused, hence they are generic and not part of the project-specific code of an application:

- Event Manager
- UI Manager
- Navigation

1.1.2. FRAMEWORK VIS

In order to dynamically load DQMH modules and to achieve some of the UI framework functionality, the following VIs are part of the framework, too:

STARTUP VI

The /startup.vi is used to run the application. It…

- shows a splash screen
- reads the main configuration (containing the list of modules to load)
- loads the UI Manager and the Event Manager modules
- loads a list of project modules (no static linkage)
- calls the “configure” requests of all the modules
- displays the front panel of the UI Manager module

PROJECT VIS

These are project-related or project-specific VIs that are supplied by the UI Framework but can or have to be modified for each project.

- /Project/Project.lvlib: Contains the project-specific VIs
- /Project/PROJECT_InitLogging.vi: Starts the hse-logger
- /Project/PROJECT_Name–constant.vi: Name of the project
- /Project/PROJECT_RunTimeMenu.rtm: Run-Time Menu for the application
- /Project/PROJECT_SplashScreen.vi: Splash Screen
- /Project/PROJECT_StartupSteps.ctl: List of steps for the startup.vi to execute
- /Project/PROJECT_UserCredentials.vi: List of users for the built-in login feature

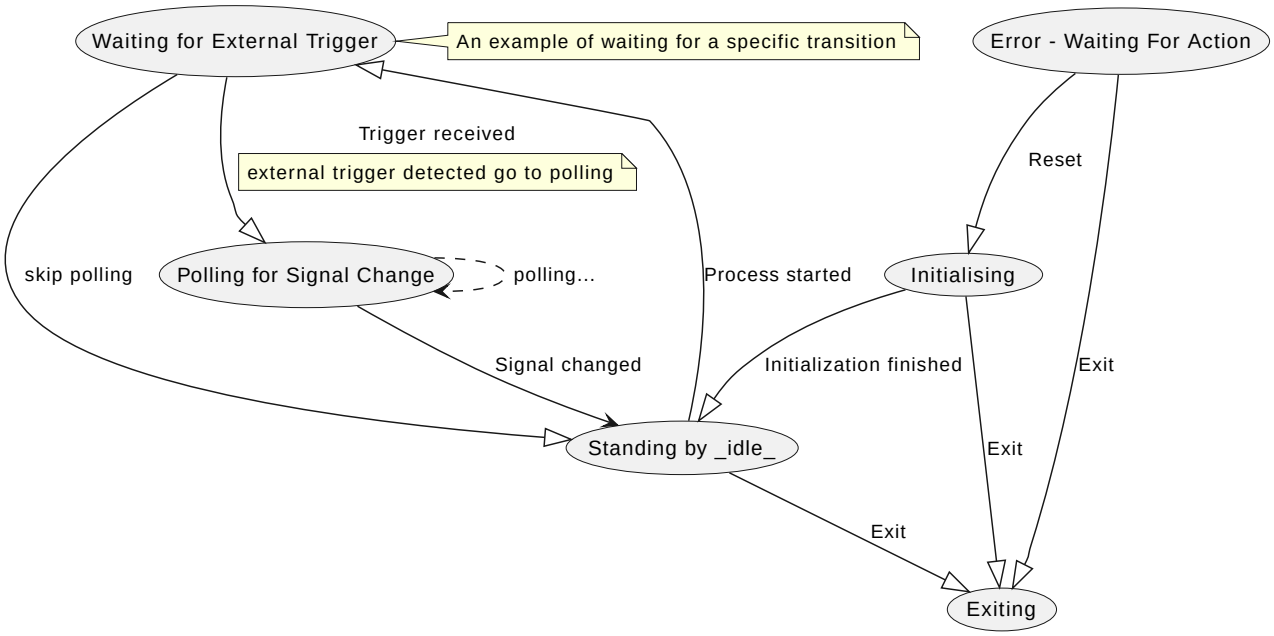
1.2. PROJECT-SPECIFIC DQMH MODULES

In order for DQMH modules to be compatible with our application template, these modules need to implement our HSE DQMH flavour.

CHAPTER 2. STATE MACHINES

2.1. STATEMACHINE

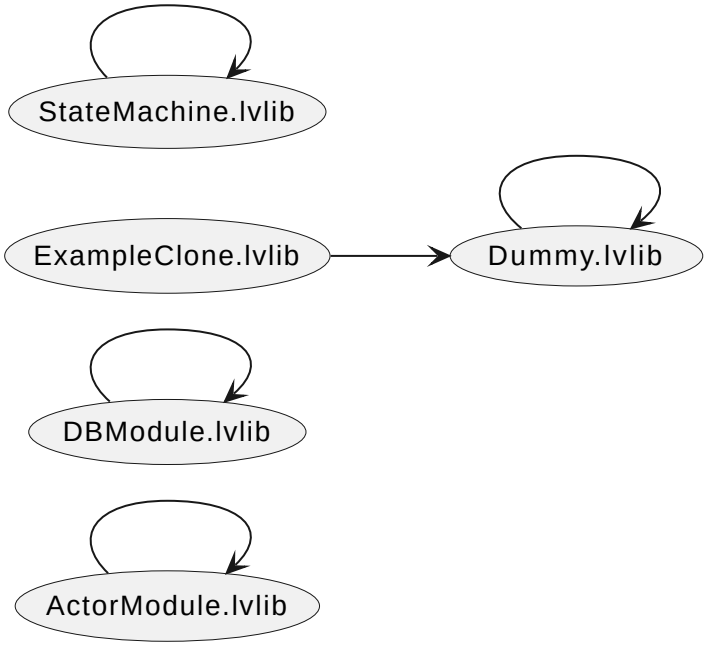
2.1.1. MAIN.VI



CHAPTER 3. CALLING DEPENDENCY DIAGRAMS

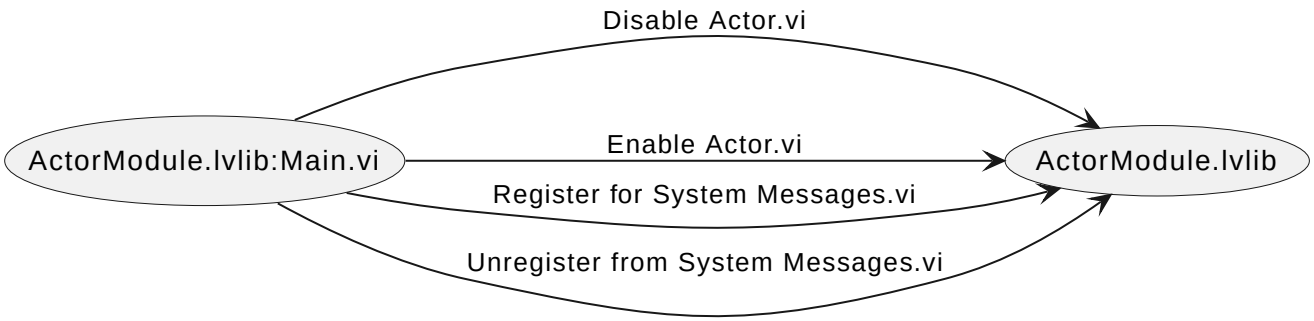
3.1. OVERVIEW

3.1.1. PROJECT

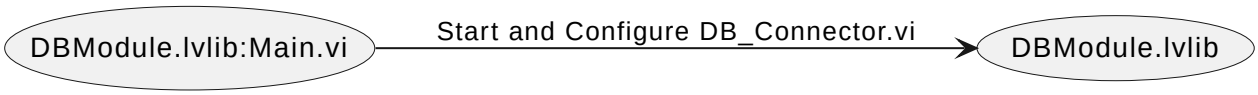


3.2. CALLERS

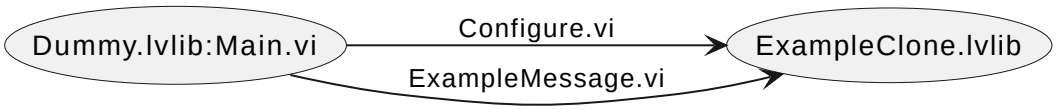
3.2.1. ACTORMODULE.LVLIB



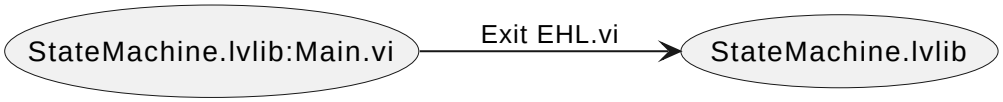
3.2.2. DBMODULE.LVLIB



3.2.3. EXAMPLECLONE.LVLIB



3.2.4. STATEMACHINE.LVLIB



3.3. LISTENERS

No elements found.

APPENDIX A: DQMH

DQMH modules documentation

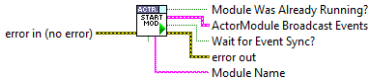
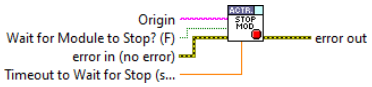
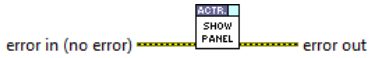
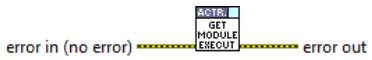
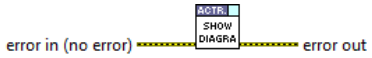
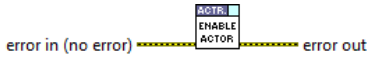
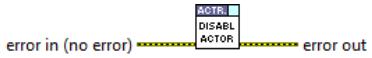
A.1. ACTORMODULE.LVLIB

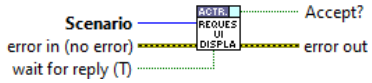
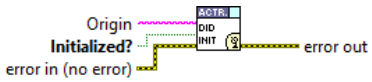
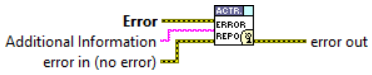
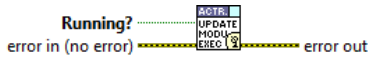
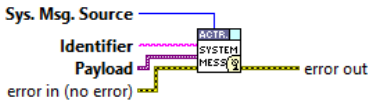
Type: Singleton

Responsibility: No description found (add content in DQMH module lvlib description)

A.1.1. EVENT LIST

Table 1. Events

Name	Type	Connector pane	Description	S.	R.	I.
Start Module			Launches the Module Main.vi.			
Stop Module			Send the Stop request to the Module's Main.vi. If Wait for Module to Stop? is TRUE, this VI will wait until the module main VI stops, and will timeout at the Timeout to Wait for Stop value. This value defaults to "-1", which means the VI will not timeout, and will always wait until the module main VI stops before completing execution. Note: The Timeout to Wait for Stop value is ignored if 'Wait for Module to Stop?' is set to FALSE.			
Show Panel			Send the Show Panel request to the Module's Main.vi.			
Hide Panel			Send the Hide Panel request to the Module's Main.vi.			
Get Module Execution Status			Fire the Get Module Execution Status request.			
Show Diagram			This VI tells the Module to show its block diagram to facilitate troubleshooting (add probes, breakpoints, highlight execution, etc).			
Prepare			Prepare (the UI of the) module for display			
Configure			Triggers the auto-configuration of the module			
Enable Actor			Activates the helper loop timeout structure			
Disable Actor			Stops the timeout polling of the helper loop			

Name	Type	Connector pane	Description	S.	R.	I.
Register for System Messages			Registers for other modules' System Message broadcast events			
Unregister from System Messages			Unregisters from other modules' System Message broadcast events			
Request UI Display			Requests the module to display its UI as specified in Scenario: "managed" => in the UI Manager's subpanel "stand-alone" => as a separate window			
Module Did Init			Send the Module Did Init event to any VI registered to listen to this module's broadcast events.			
Status Updated			Send the Status Updated event to any VI registered to listen to events from the owning module.			
Error Reported			Note: This VI was modified by the Validate DQMH Module tool to parse additional information tags out of the incoming error source string.			
Module Did Stop			Send the Module Did Stop event to any VI registered to listen to this module's broadcast events.			
Update Module Execution Status			Broadcast event to specify whether or not the module is running.			
System Message			Broadcasts a "System Message" consisting of an Identifier (string), a Payload (variant) and a Source (enum). Source values are: - "module": A broadcast from within a module itself - "application": A broadcast generated from outside a module - "network": A broadcast sent/received via network			

Type: → Request | → Request and Wait for Reply | → Broadcast

Scope: → Protected | → Community

Reentrancy: → Preallocated reentrancy | → Shared reentrancy

Inlining: → Inlined

A.1.2. MODULE RELATIONSHIP

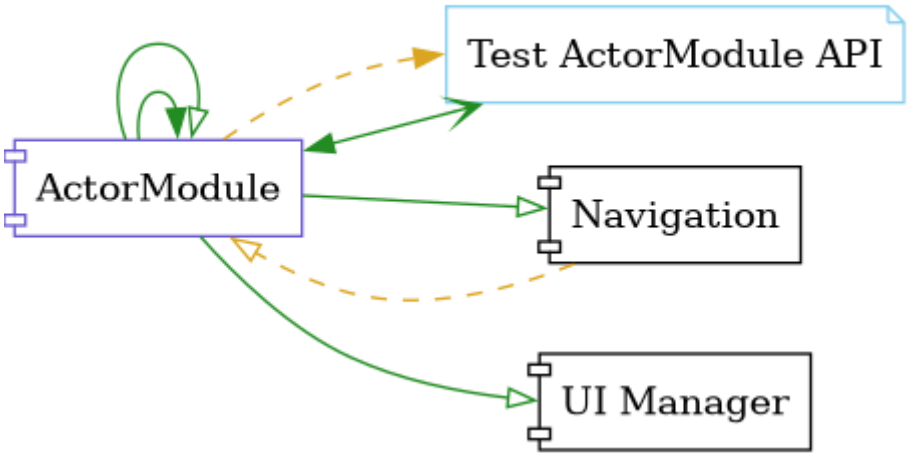


Table 2. Requests callers

Request Name	Callers
Configure	Test ActorModule API.vi
Disable Actor	ActorModule.lvlib:Main.vi Test ActorModule API.vi
Enable Actor	ActorModule.lvlib:Main.vi Test ActorModule API.vi
Get Module Execution Status	ActorModule.lvlib:Obtain Broadcast Events for Registration.vi ActorModule.lvlib:Start Module.vi
Hide Panel	Test ActorModule API.vi
Prepare	Test ActorModule API.vi
Register for System Messages	ActorModule.lvlib:Main.vi Test ActorModule API.vi
Request UI Display	Test ActorModule API.vi
Show Diagram	Test ActorModule API.vi
Show Panel	Test ActorModule API.vi
Unregister from System Messages	ActorModule.lvlib:Main.vi Test ActorModule API.vi

Table 3. Broadcasts Listeners

Broadcast Name	Listeners
Error Reported	Test ActorModule API.vi
Module Did Init	Test ActorModule API.vi
Module Did Stop	Test ActorModule API.vi
Status Updated	Test ActorModule API.vi
System Message	Test ActorModule API.vi
Update Module Execution Status	Test ActorModule API.vi

Table 4. Used requests

Module	Requests
ActorModule.lvlib	Disable Actor.vi Enable Actor.vi Register for System Messages.vi Stop Module.vi Unregister from System Messages.vi
Navigation.lvlib	Get Module Execution Status.vi Update Navigation.vi
UI Manager.lvlib	Get Module Execution Status.vi

Table 5. Registered broadcast

Module	Broadcasts
Navigation.lvlib	Sub-Control Pressed.vi

A.1.3. MODULE START/STOP CALLS

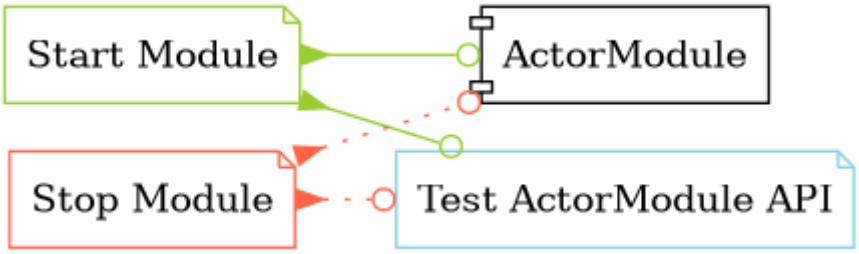


Table 6. Start and Stop module callers

Function	Callers
Start Module	ActorModule.lvlib:Load Module.vi Test ActorModule API.vi
Stop Module	ActorModule.lvlib:Handle Exit.vi Test ActorModule API.vi

A.1.4. MODULE CUSTOM ERRORS

Custom errors are added to the module via vi named `*--error.vi`.

Module ActorModule.lvlib use the following custom errors:

Table 7. Custom errors

Name	Code	Description
Module Not Running	403681	%s Module is not running.
Module Not Stopped	403682	The Stop Module VI for the %s module timed out while waiting for the module main VI to stop. The module main VI may still be running.
Module Not Synced	403683	%s Module was unable to synchronize events.
Request and Wait for Reply Timeout	403686	%s

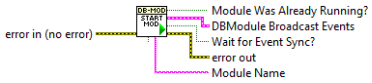
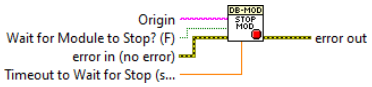
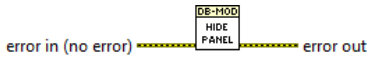
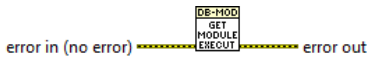
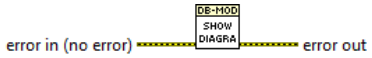
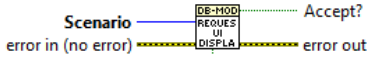
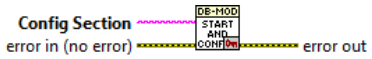
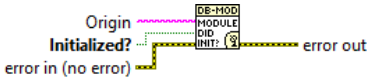
A.2. DBMODULE.LVLIB

Type: Singleton

Responsibility: No description found (add content in DQMH module lvlib description)

A.2.1. EVENT LIST

Table 8. Events

Name	Type	Connector pane	Description	S.	R.	I.
Start Module			Launches the Module Main.vi.			
Stop Module			Send the Stop request to the Module's Main.vi. If Wait for Module to Stop? is TRUE, this VI will wait until the module main VI stops, and will timeout at the Timeout to Wait for Stop value. This value defaults to "-1", which means the VI will not timeout, and will always wait until the module main VI stops before completing execution. Note: The Timeout to Wait for Stop value is ignored if 'Wait for Module to Stop?' is set to FALSE.			
Show Panel			Send the Show Panel request to the Module's Main.vi.			
Hide Panel			Send the Hide Panel request to the Module's Main.vi.			
Get Module Execution Status			Fire the Get Module Execution Status request.			
Show Diagram			This VI tells the Module to show its block diagram to facilitate troubleshooting (add probes, breakpoints, highlight execution, etc).			
Configure			Triggers the auto-configuration of the module.			
Prepare			Prepare (the UI of the) module for display.			
Request UI Display			Requests the module to display its UI as specified in Scenario: "managed" => in the UI Manager's subpanel "stand-alone" => as a separate window			
Start and Configure DB_Connector			starts and configures the DB_Connector Module			
Module Did Init			Send the Module Did Init event to any VI registered to listen to this module's broadcast events.			

Name	Type	Connector pane	Description	S.	R.	I.
Status Updated		Status error in (no error) DB-MOD STATUS UPDT error out	Send the Status Updated event to any VI registered to listen to events from the owning module.			
Error Reported		Error Additional Information error in (no error) DB-MOD ERROR RPRT error out	Note: This VI was modified by the Validate DQMH Module tool to parse additional information tags out of the incoming error source string.			
Module Did Stop		Origin error in (no error) DB-MOD MODULE DID STOP error out	Send the Module Did Stop event to any VI registered to listen to this module's broadcast events.			
Update Module Execution Status		Running? error in (no error) DB-MOD MODULE EXEC STAT error out	Broadcast event to specify whether or not the module is running.			
System Message		Sys. Msg. Source Identifier Payload error in (no error) DB-MOD SYSTEM MESS error out	Broadcast a system generic message.			

Type: → Request | → Request and Wait for Reply | → Broadcast

Scope: → Protected | → Community

Reentrancy: → Preallocated reentrancy | → Shared reentrancy

Inlining: → Inlined

A.2.2. MODULE RELATIONSHIP

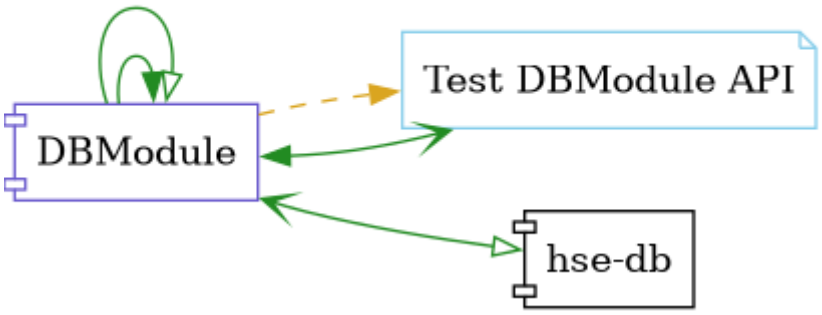


Table 9. Requests callers

Request Name	Callers
Configure	Test DBModule API.vi
Get Module Execution Status	DBModule.lvlib:Obtain Broadcast Events for Registration.vi DBModule.lvlib:Start Module.vi
Hide Panel	Test DBModule API.vi
Prepare	Test DBModule API.vi

Request Name	Callers
Request UI Display	Test DBModule API.vi
Show Diagram	Test DBModule API.vi
Show Panel	Test DBModule API.vi
Start and Configure DB_Connector	DBModule.lvlib:Main.vi

Table 10. Broadcasts Listeners

Broadcast Name	Listeners
Error Reported	Test DBModule API.vi
Module Did Init	Test DBModule API.vi
Module Did Stop	Test DBModule API.vi
Status Updated	Test DBModule API.vi
System Message	Test DBModule API.vi
Update Module Execution Status	Test DBModule API.vi

Table 11. Used requests

Module	Requests
DBModule.lvlib	Start and Configure DB_Connector.vi Stop Module.vi
hse-db.lvlib	DB_CONNECTOR.lvlib:Configure.vi DB_CONNECTOR.lvlib:Get DB-Type.vi DB_CONNECTOR.lvlib:Query.vi (5) DB_CONNECTOR.lvlib:Stop Module.vi

Table 12. Registered broadcast

Module	Broadcasts
——	——

A.2.3. MODULE START/STOP CALLS

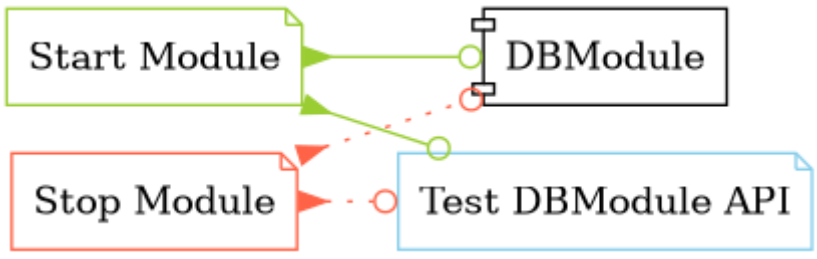


Table 13. Start and Stop module callers

Function	Callers
Start Module	DBModule.lvlib:Load Module.vi Test DBModule API.vi
Stop Module	DBModule.lvlib:Handle Exit.vi Test DBModule API.vi

A.2.4. MODULE CUSTOM ERRORS

Custom errors are added to the module via vi named `*--error.vi`.

Module DBModule.lvlib use the following custom errors:

Table 14. Custom errors

Name	Code	Description
Module Not Running	403681	%s Module is not running.
Module Not Stopped	403682	The Stop Module VI for the %s module timed out while waiting for the module main VI to stop. The module main VI may still be running.
Module Not Synced	403683	%s Module was unable to synchronize events.
Request and Wait for Reply Timeout	403686	%s

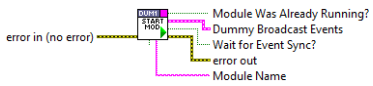
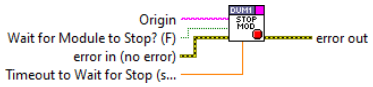
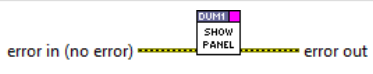
A.3. DUMMY.LVLIB

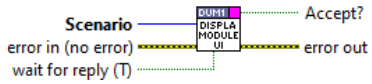
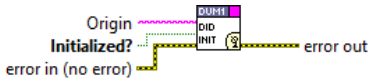
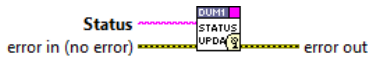
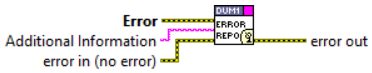
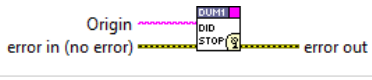
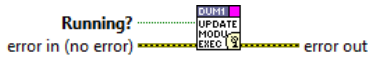
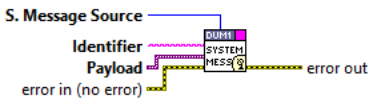
Type: Singleton

Responsibility: No description found (add content in DQMH module lvlib description)

A.3.1. EVENT LIST

Table 15. Events

Name	Type	Connector pane	Description	S.	R.	I.
Start Module			Launches the Module Main.vi.			
Stop Module			Send the Stop request to the Module's Main.vi. If Wait for Module to Stop? is TRUE, this VI will wait until the module main VI stops, and will timeout at the Timeout to Wait for Stop value. This value defaults to "-1", which means the VI will not timeout, and will always wait until the module main VI stops before completing execution. Note: The Timeout to Wait for Stop value is ignored if 'Wait for Module to Stop?' is set to FALSE.			
Show Panel			Send the Show Panel request to the Module's Main.vi.			
Hide Panel			Send the Hide Panel request to the Module's Main.vi.			
Get Module Execution Status			Fire the Get Module Execution Status request.			

Name	Type	Connector pane	Description	S.	R.	I.
Show Diagram			This VI tells the Module to show its block diagram to facilitate troubleshooting (add probes, breakpoints, highlight execution, etc).			
Configure			Self-configuration (triggered automatically from main.vi)			
Prepare			Prepare (the UI of the) module for display			
Register for UI Login			Registers for the UI module's "Login" broadcast event			
Request UI Display			Requests the module to display its UI as specified in Scenario: "managed" => in the UI Manager's subpanel "stand-alone" => as a separate window			
Module Did Init			No description found (add content in vi description)			
Status Updated			Note: This VI was modified by the Validate DQMH Module tool to have a broadcast event glyph overlay on its icon.			
Error Reported			Note: This VI was modified by the Validate DQMH Module tool to parse additional information tags out of the incoming error source string.			
Module Did Stop			No description found (add content in vi description)			
Update Module Execution Status			No description found (add content in vi description)			
System Message			Note: This VI was modified by the Validate DQMH Module tool to have a broadcast event glyph overlay on its icon.			

Type: → Request | → Request and Wait for Reply | → Broadcast

Scope: → Protected | → Community

Reentrancy: → Preallocated reentrancy | → Shared reentrancy

Inlining: → Inlined

A.3.2. MODULE RELATIONSHIP

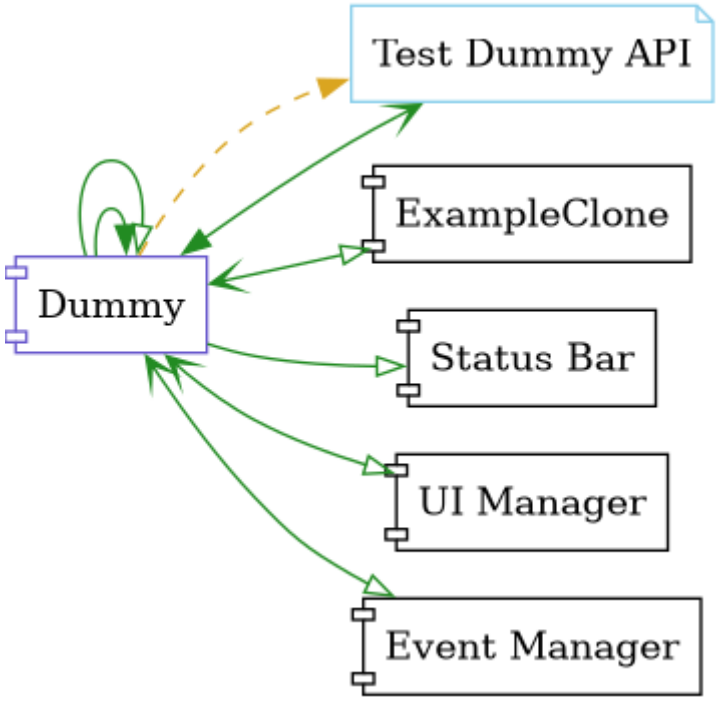


Table 16. Requests callers

Request Name	Callers
Configure	Test Dummy API.vi
Get Module Execution Status	Dummy.lvlib:Obtain Broadcast Events for Registration.vi Dummy.lvlib:Start Module.vi
Hide Panel	Test Dummy API.vi
Prepare	Test Dummy API.vi
Register for UI Login	
Request UI Display	Test Dummy API.vi
Show Diagram	Test Dummy API.vi
Show Panel	Test Dummy API.vi

Table 17. Broadcasts Listeners

Broadcast Name	Listeners
Error Reported	Test Dummy API.vi
Module Did Init	Test Dummy API.vi
Module Did Stop	Test Dummy API.vi
Status Updated	Test Dummy API.vi
System Message	Test Dummy API.vi
Update Module Execution Status	Test Dummy API.vi

Table 18. Used requests

Module	Requests
Dummy.lvlib	Stop Module.vi

Module	Requests
Event Manager.lvlib	Set Modules.vi
ExampleClone.lvlib	Configure.vi ExampleMessage.vi Request UI Display.vi Show Diagram.vi Stop Module.vi (2)
Status Bar.lvlib	Update Status Bar.vi
UI Manager.lvlib	Get CLI Arguments.vi Get Module Execution Status.vi Set Modules.vi Set Run-Time Menu Visibility.vi (2) Set Window Closeable State.vi (2)

Table 19. Registered broadcast

Module	Broadcasts
——	——

A.3.3. MODULE START/STOP CALLS

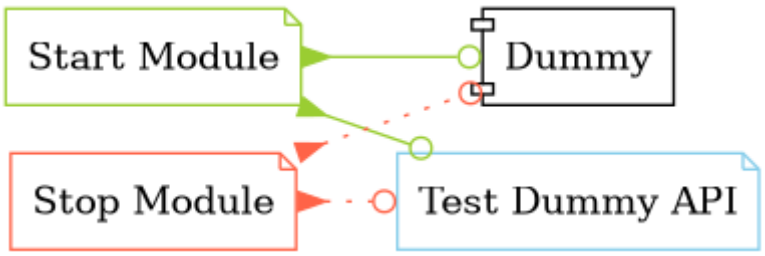


Table 20. Start and Stop module callers

Function	Callers
Start Module	Dummy.lvlib:Load Module.vi Test Dummy API.vi
Stop Module	Dummy.lvlib:Handle Exit.vi Test Dummy API.vi

A.3.4. MODULE CUSTOM ERRORS

Custom errors are added to the module via vi named `*--error.vi`.

Module Dummy.lvlib use the following custom errors:

Table 21. Custom errors

Name	Code	Description
My first custom error	0	
This is an error with input	0	
Module Not Running	403681	%s Module is not running.

Name	Code	Description
Module Not Stopped	403682	The Stop Module VI for the %s module timed out while waiting for the module main VI to stop. The module main VI may still be running.
Module Not Synced	403683	%s Module was unable to synchronize events.
Request and Wait for Reply Timeout	403686	%s
Error ring	500989	Error ring on block diagram

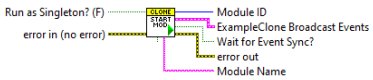
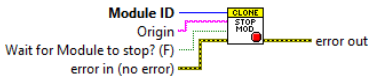
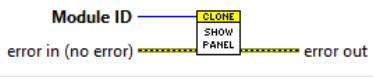
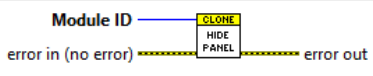
A.4. EXAMPLECLONE.LVLIB

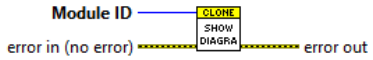
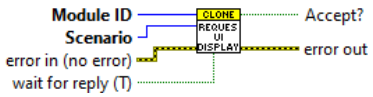
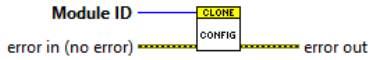
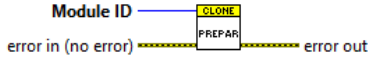
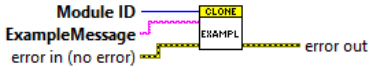
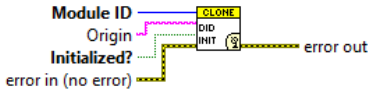
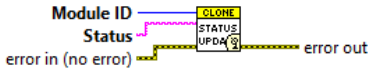
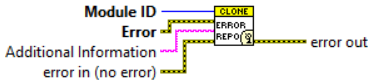
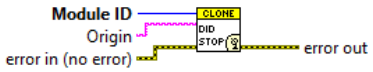
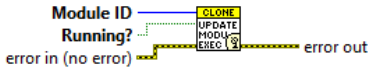
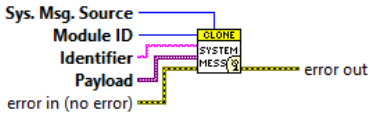
Type: Cloneable

Responsibility: No description found (add content in DQMH module lvlib description)

A.4.1. EVENT LIST

Table 22. Events

Name	Type	Connector pane	Description	S.	R.	I.
Start Module			Launches the Module Main.vi.			
Stop Module			Send the Stop request to the Module's Main.vi. If Wait for Module to stop? is TRUE, then this VI will not complete execution until the Module Main VI has stopped running. Note: If the cloneable module is running as singleton, then the 'Wait for Module to stop?' input is ignored... this VI will always wait until a cloneable Main VI running as singleton has stopped running. Note: This VI was modified by the Validate DQMH Module tool to upgrade it to the DQMH 6.0 approach to poll the execution state of a cloneable module running as singleton to know when the module has gone idle. Note: This VI was modified by the Validate DQMH Module tool to upgrade it to the DQMH 5.0 approach to destroying cloneable module event references. Note: This VI was modified by the Validate DQMH Module tool to remove the Status Updated.vi subVI call.			
Show Panel			Send the Show Panel request to the Module's Main.vi.			
Hide Panel			Send the Hide Panel request to the Module's Main.vi.			

Name	Type	Connector pane	Description	S.	R.	I.
Show Diagram			This VI tells the Module to show its block diagram to facilitate troubleshooting (add probes, breakpoints, highlight execution, etc).			
Request UI Display			Requests another module to display its UI in the given Display Scenario: The "Stand-Alone" scenario asks the module to open its front panel as a new windows. The "Managed" scenario asks the module to broadcast a request for display which the UI Manager then processes, showing the module in its subpanel.			
Configure			Triggers the auto-configuration of the module			
Prepare			Prepares the module UI (front panel) for display			
ExampleMessage			Send a message to the module that is displayed on its front panel			
Module Did Init			No description found (add content in vi description)			
Status Updated			No description found (add content in vi description)			
Error Reported			Note: This VI was modified by the Validate DQMH Module tool to parse additional information tags out of the incoming error source string.			
Module Did Stop			No description found (add content in vi description)			
Update Module Execution Status			No description found (add content in vi description)			
System Message			No description found (add content in vi description)			

Type: → Request | → Request and Wait for Reply | → Broadcast

Scope: → Protected | → Community

Reentrancy: → Preallocated reentrancy | → Shared reentrancy

Inlining: → Inlined

A.4.2. MODULE RELATIONSHIP

Table 23. Requests callers

Request Name	Callers
Configure	Dummy.lvlib:Main.vi Test ExampleClone API.vi
ExampleMessage	Dummy.lvlib:Main.vi Test ExampleClone API.vi
Hide Panel	Test ExampleClone API.vi
Prepare	Test ExampleClone API.vi
Request UI Display	Dummy.lvlib:Main.vi Test ExampleClone API.vi
Show Diagram	Dummy.lvlib:Main.vi Test ExampleClone API.vi
Show Panel	Test ExampleClone API.vi

Table 24. Broadcasts Listeners

Broadcast Name	Listeners
Error Reported	Test ExampleClone API.vi
Module Did Init	Test ExampleClone API.vi
Module Did Stop	Test ExampleClone API.vi
Status Updated	Test ExampleClone API.vi
System Message	Test ExampleClone API.vi
Update Module Execution Status	Test ExampleClone API.vi

Table 25. Used requests

Module	Requests
ExampleClone.lvlib	Stop Module.vi

Table 26. Registered broadcast

Module	Broadcasts
□—□	□—□

A.4.3. MODULE START/STOP CALLS

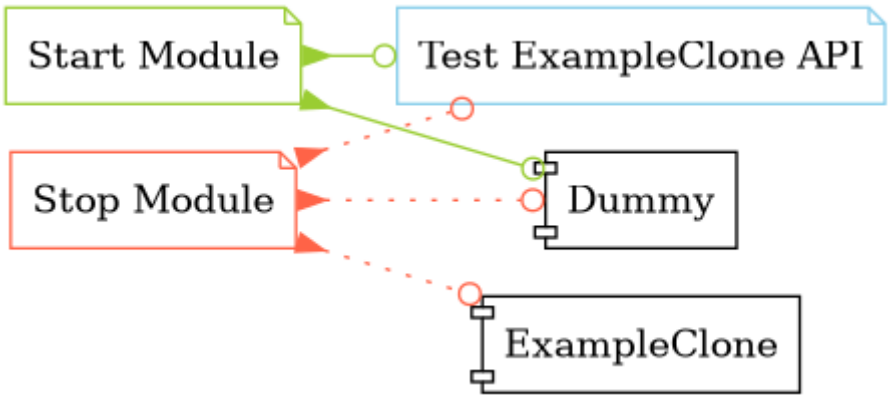


Table 27. Start and Stop module callers

Function	Callers
Start Module	Dummy.lvlib:Main.vi Test ExampleClone API.vi
Stop Module	Dummy.lvlib:Main.vi ExampleClone.lvlib:Handle Exit.vi Test ExampleClone API.vi

A.4.4. MODULE CUSTOM ERRORS

Custom errors are added to the module via vi named `*--error.vi`.

Module ExampleClone.lvlib use the following custom errors:

Table 28. Custom errors

Name	Code	Description
Module Running as Singleton	403680	The "%s" module is currently running as singleton, but the Start Module VI was called with 'Run as Singleton' specified as FALSE.
Module Not Stopped	403682	The Stop Module VI for the %s module timed out while waiting for the module main VI to stop. The module main VI may still be running.
Module Not Synced	403683	%s Module was unable to synchronize events.
Module Not Running	403684	Not a single instance of "%s" Module running.
Module Running as Cloneable	403685	The "%s" module is currently running as cloneable, but the Start Module VI was called with 'Run as Singleton' specified as TRUE.
Request and Wait for Reply Timeout	403686	%s
Master Reference Not Closed	403687	The "%s" module cannot be run as singleton because the Master Reference is still open from a prior run as cloneable. If you plan on running this module as both singleton and cloneable, consider changing your Main VI to wire a value of TRUE to the 'Close Master Reference' input of Init Module.vi.

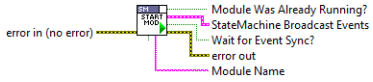
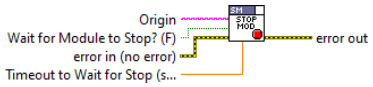
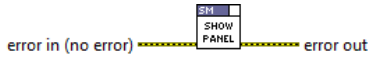
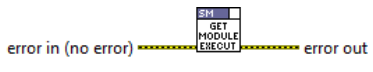
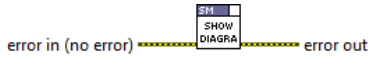
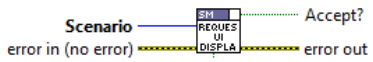
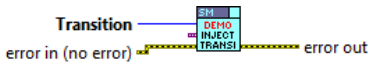
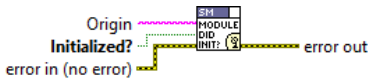
A.5. STATEMACHINE.LVLIB

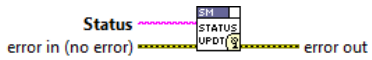
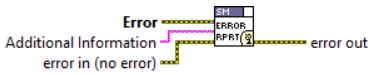
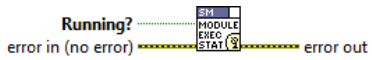
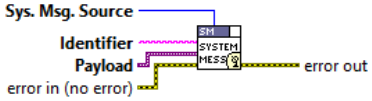
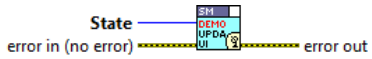
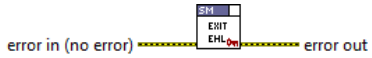
Type: Singleton

Responsibility: No description found (add content in DQMH module lvlib description)

A.5.1. EVENT LIST

Table 29. Events

Name	Type	Connector pane	Description	S.	R.	I.
Start Module			Launches the Module Main.vi.			
Stop Module			Send the Stop request to the Module's Main.vi. If Wait for Module to Stop? is TRUE, this VI will wait until the module main VI stops, and will timeout at the Timeout to Wait for Stop value. This value defaults to "-1", which means the VI will not timeout, and will always wait until the module main VI stops before completing execution. Note: The Timeout to Wait for Stop value is ignored if 'Wait for Module to Stop?' is set to FALSE.			
Show Panel			Send the Show Panel request to the Module's Main.vi.			
Hide Panel			Send the Hide Panel request to the Module's Main.vi.			
Get Module Execution Status			Fire the Get Module Execution Status request.			
Show Diagram			This VI tells the Module to show its block diagram to facilitate troubleshooting (add probes, breakpoints, highlight execution, etc).			
Configure			Triggers the auto-configuration of the module.			
Prepare			Prepares the module for displaying its front panel.			
Request UI Display			Requests the module to display its UI as specified in Scenario: "managed" => in the UI Manager's subpanel "stand-alone" => as a separate window			
DEMO Inject Transition			Triggers a transition via the public API. This request serves the purpose of testing/show-casing the state machine. By design, transitions should not be triggered from outside the DQMH module (i.e. via the public API)			
Module Did Init			Send the Module Did Init event to any VI registered to listen to this module's broadcast events.			

Name	Type	Connector pane	Description	S.	R.	I.
Status Updated			Send the Status Updated event to any VI registered to listen to events from the owning module.			
Error Reported			Note: This VI was modified by the Validate DQMH Module tool to parse additional information tags out of the incoming error source string.			
Module Did Stop			Send the Module Did Stop event to any VI registered to listen to this module's broadcast events.			
Update Module Execution Status			Broadcast event to specify whether or not the module is running.			
System Message			Broadcasts a "System Message" consisting of an Identifier (string), a Payload (variant) and a Source (enum). Source values are: - "module": A broadcast from within a module itself - "application": A broadcast generated from outside a module - "network": A broadcast sent/received via network			
DEMO Updated UI State			Broadcasts whenever the "Update UI State" private request was executed after a state change			
Exit EHL			this event will stop the event handling loop			

Type: → Request | → Request and Wait for Reply | → Broadcast

Scope: → Protected | → Community

Reentrancy: → Preallocated reentrancy | → Shared reentrancy

Inlining: → Inlined

A.5.2. MODULE RELATIONSHIP

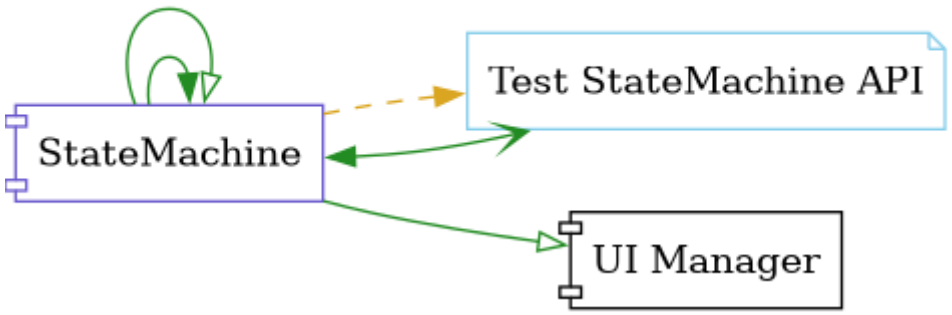


Table 30. Requests callers

Request Name	Callers
Configure	Test StateMachine API.vi
DEMO Inject Transition	Test StateMachine API.vi
Exit EHL	StateMachine.lvlib:Main.vi
Get Module Execution Status	StateMachine.lvlib:Obtain Broadcast Events for Registration.vi StateMachine.lvlib:Start Module.vi
Hide Panel	Test StateMachine API.vi
Prepare	Test StateMachine API.vi
Request UI Display	Test StateMachine API.vi
Show Diagram	Test StateMachine API.vi
Show Panel	Test StateMachine API.vi

Table 31. Broadcasts Listeners

Broadcast Name	Listeners
DEMO Updated UI State	Test StateMachine API.vi
Error Reported	Test StateMachine API.vi
Module Did Init	Test StateMachine API.vi
Module Did Stop	Test StateMachine API.vi
Status Updated	Test StateMachine API.vi
System Message	Test StateMachine API.vi
Update Module Execution Status	Test StateMachine API.vi

Table 32. Used requests

Module	Requests
StateMachine.lvlib	Exit EHL.vi Stop Module.vi
UI Manager.lvlib	Set Window Closeable State.vi (2) Shutdown Application.vi

Table 33. Registered broadcast

Module	Broadcasts
□—□	□—□

A.5.3. MODULE START/STOP CALLS

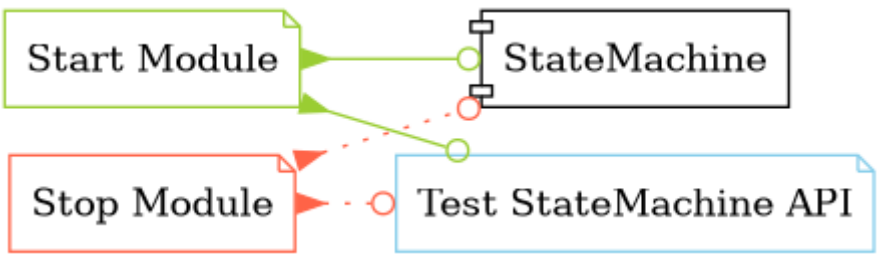


Table 34. Start and Stop module callers

Function	Callers
Start Module	StateMachine.lvlib:Load Module.vi Test StateMachine API.vi
Stop Module	StateMachine.lvlib:Handle Exit.vi Test StateMachine API.vi

A.5.4. MODULE CUSTOM ERRORS

Custom errors are added to the module via vi named `*--error.vi`.

Module StateMachine.lvlib use the following custom errors:

Table 35. Custom errors

Name	Code	Description
Module Not Running	403681	%s Module is not running.
Module Not Stopped	403682	The Stop Module VI for the %s module timed out while waiting for the module main VI to stop. The module main VI may still be running.
Module Not Synced	403683	%s Module was unable to synchronize events.
Request and Wait for Reply Timeout	403686	%s

APPENDIX B: LIBRARIES

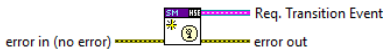
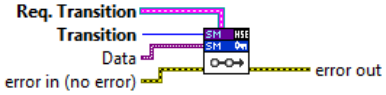
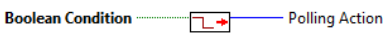
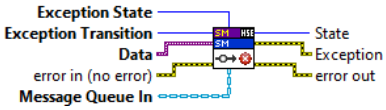
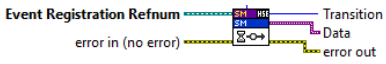
Misc. reuse libraries

B.1. HSE-STATE-MACHINE.LVLIB

Responsibility: No description found (add content in lvlib description)

Version: 1.0.0.0

Table 36. Functions (non private scope only)

Name	Connector pane	Description	S.	R.	I.
SM Create Event		Creates the local user event used for requesting a transition.			
SM Destroy Event		Destroys the local user event used for requesting a transition.			
SM Request Transition		Sends a Transition to the State Machine which will react dependent on it's state.			
SM Polling - Leave State if FALSE		Keep polling if the boolean condition is TRUE			
SM Polling - Leave State if TRUE		Keep polling if the boolean condition is TRUE			
SM DEMO		Demonstrates the use of the HSE-State Machine and can be used as a Copy and Paste Template.			
SM Process Exception Transition		If an error occurs, go to the "Erroring" state and send an update to the UI			
SM Update UI State		Propagate state change to UI. HSE: This is only a place holder, in DQMH modules this would use the Message Queue. Implement your own communication mechanism here.			
SM Wait for Transition		Waits for a Transition to occur in the duration of Timeout.			

Scope: → Protected | → Community

Reentrancy: → Preallocated reentrancy | → Shared reentrancy

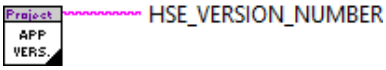
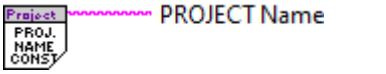
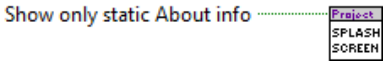
Inlining: → Inlined

B.2. PROJECT.LVLIB

Responsibility: No description found (add content in lvlib description)

Version: 1.0.0.0

Table 37. Functions (non private scope only)

Name	Connector pane	Description	S.	R.	I.
PROJECT_ApplicationVersion—constant		Version number of the application. This is set automatically during build.			
PROJECT_Evaluate Default Menu Selection		No description found (add content in vi description)			
PROJECT_InitLogging		Initiate the logging functions.			
PROJECT_Layout-VI		This is the default layout VI for the HSE UI Framework. It features three areas: - header (contains the "SP_Header" subpanel) - navigation (contains the "SP_Navigation" subpanel) - content (contains the "SP_Content" subpanel) This VI is loaded by the UI Manager module, the subpanels are then populated with DQMH modules as configured in the UI Manager's configuration file.			
PROJECT_Name—constant		Constant VI defining the name of the Project. This influences directory structure and other things.			
PROJECT_SplashScreen		No description found (add content in vi description)			
PROJECT_StaticDependencies—constant		Place static references to VIs that are loaded dynamically and therefor have no static linking and would be left away when building binaries.			
PROJECT_UserCredentials		No description found (add content in vi description)			

Scope: → Protected | → Community

Reentrancy: → Preallocated reentrancy | → Shared reentrancy

Inlining: → Inlined

APPENDIX C: CLASSES

LabVIEW Classes

C.1. CLASSES OVERVIEW

This project contains 0 classe and 0 interface.

Table 38. Classes list

Classes	Interfaces

APPENDIX D: CUSTOM ERRORS

List of Custom Error VIs

D.1. CUSTOM ERRORS

Custom errors are added via vi named `*--error.vi`.

Table 39. Custom errors

Name	Code	Description	Owned by
My first custom error	0		Dummy.lvlib
This is an error with input	0		Dummy.lvlib
Module Running as Singleton	403680	The "%s" module is currently running as singleton, but the Start Module VI was called with 'Run as Singleton' specified as FALSE.	ExampleClone.lvlib
Module Not Running	403681	%s Module is not running.	ActorModule.lvlib DBModule.lvlib Dummy.lvlib StateMachine.lvlib
Module Not Stopped	403682	The Stop Module VI for the %s module timed out while waiting for the module main VI to stop. The module main VI may still be running.	ActorModule.lvlib DBModule.lvlib Dummy.lvlib ExampleClone.lvlib StateMachine.lvlib
Module Not Synced	403683	%s Module was unable to synchronize events.	ActorModule.lvlib DBModule.lvlib Dummy.lvlib ExampleClone.lvlib StateMachine.lvlib
Module Not Running	403684	Not a single instance of "%s" Module running.	ExampleClone.lvlib
Module Running as Cloneable	403685	The "%s" module is currently running as cloneable, but the Start Module VI was called with 'Run as Singleton' specified as TRUE.	ExampleClone.lvlib
Request and Wait for Reply Timeout	403686	%s	ActorModule.lvlib DBModule.lvlib Dummy.lvlib ExampleClone.lvlib StateMachine.lvlib
Master Reference Not Closed	403687	The "%s" module cannot be run as singleton because the Master Reference is still open from a prior run as cloneable. If you plan on running this module as both singleton and cloneable, consider changing your Main VI to wire a value of TRUE to the 'Close Master Reference' input of Init Module.vi.	ExampleClone.lvlib

Name	Code	Description	Owned by
Error ring	500989	Error ring on block diagram	Dummy.lvlib

GLOSSARY

The rat-documentr tool facilitates the following LabVIEW-related tools and libraries:

- Antidoc by Wovalab
- Asciidoc Toolkit by Wovalab
- Graph Builder by C. Gambini
- Classy by T. Boy
- DQMH® by Delacor

Furthermore, it relies on the following tools and libraries:

- Ruby
- Asciidoctor
- Java
- GraphViz