

HAMPEL SOFTWARE ENGINEERING

TEMPLATE

HAMPEL SOFTWARE ENGINEERING

Version 3.1.0 (2025-06-18)

TABLE OF CONTENTS

1. Project Description	2
1.1. UI Framework	2
1.2. Project-Specific DQMH Modules	3
2. State Machines	4
2.1. StateMachine	4
3. Architecture	5
3.1. Overview	5
3.2. Callers	5
3.3. Listeners	6
Appendix A: DQMH	7
A.1. ActorModule.lvlib	7
A.2. DBModule.lvlib	11
A.3. Dummy.lvlib	15
A.4. ExampleClone.lvlib	19
A.5. StateMachine.lvlib	23
Appendix B: Libraries	27
B.1. HSE-State-Machine.lvlib	27
B.2. Project.lvlib	28
Appendix C: Classes	30
C.1. Classes overview	30
Appendix D: Custom Errors	31
D.1. Custom errors	31
Glossary	33



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.

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

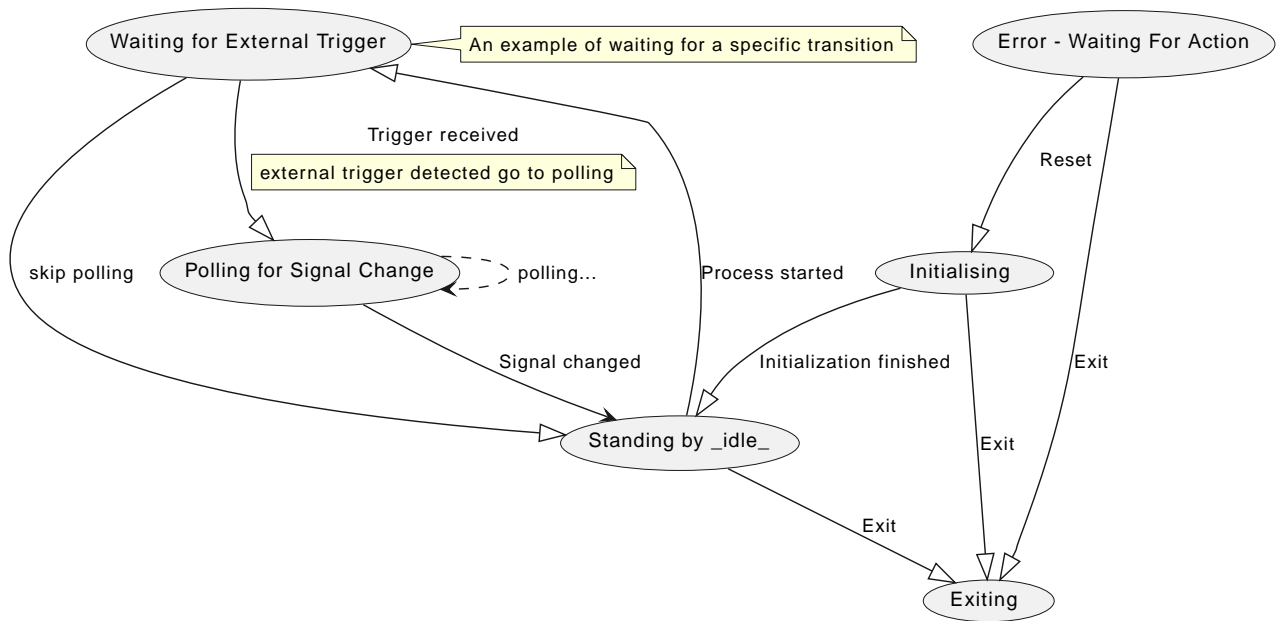
1.2. PROJECT-SPECIFIC DQMH MODULES

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

2. STATE MACHINES

2.1. STATEMACHINE

2.1.1. MAIN.VI



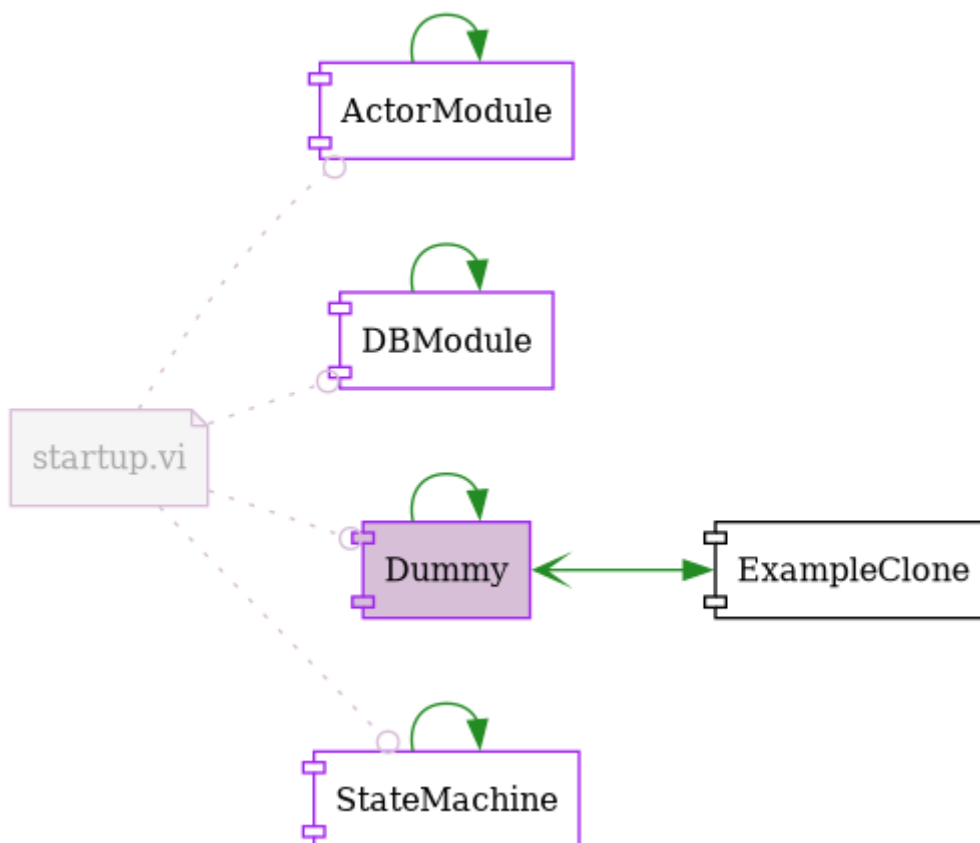
3. ARCHITECTURE

3.1. OVERVIEW

This project contains 4 singleton modules and 1 cloneable module.

Table 1. Modules list

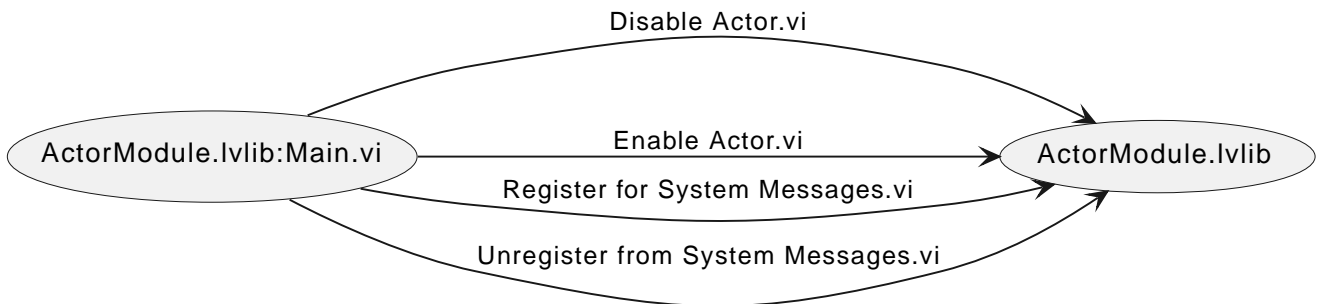
Module	Type
ActorModule.lvlib	Singleton
DBModule.lvlib	Singleton
Dummy.lvlib	Singleton
StateMachine.lvlib	Singleton
ExampleClone.lvlib	Cloneable



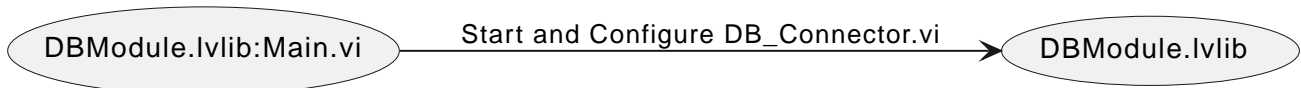
This graph represents the links between all DQMH modules. Purple borders denote HSE-ified modules. A lilac background denotes the module being displayed first (the "startup module").

3.2. CALLERS

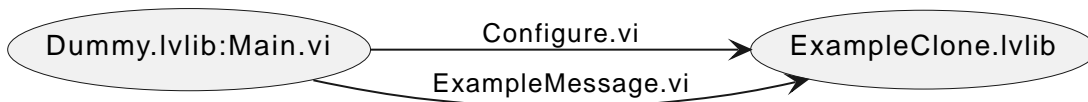
3.2.1. ACTORMODULE.LVLIB



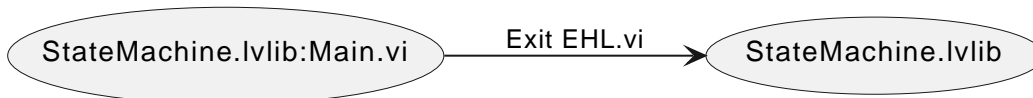
3.2.2. DBMODULE.LVLIB



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 lplib description)

A.1.1. MODULE RELATIONSHIP

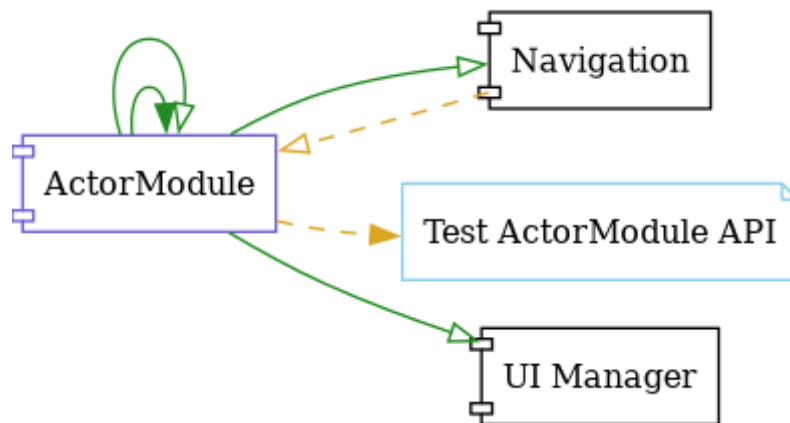


Table 2. Requests callers

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

Table 3. Broadcasts Listeners

Broadcast Name	Listeners
Error Reported	Test ActorModule API.vi
Module Did Init	Test ActorModule API.vi

Broadcast Name	Listeners
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

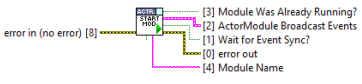
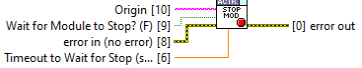







Module	Requests
ActorModule.lvlib	Disable Actor.vi Enable Actor.vi Register for System Messages.vi Unregister from System Messages.vi
Navigation.lvlib	Get Module Execution Status.vi Update Navigation.vi
UI Manager.lvlib	Get Module Execution Status.vi

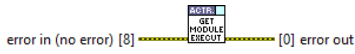















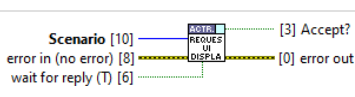

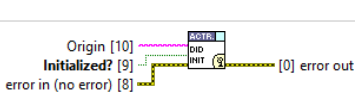

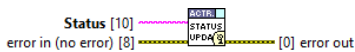

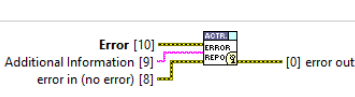

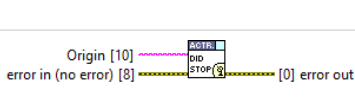

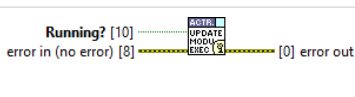

Table 5. Registered broadcast


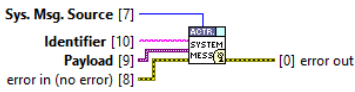

Module	Broadcasts
Navigation.lvlib	Sub-Control Pressed.vi

A.1.2. EVENT LIST

Table 6. Events

Name	Type	Connector pane	Description	S.	R.	I.
Start Module		 <p>error in (no error) [8] — [3] Module Was Already Running? [2] ActorModule Broadcast Events [1] Wait for Event Sync? [0] error out [4] Module Name</p>	Launches the Module Main.vi.			
Stop Module		 <p>Origin [10] — [0] error out Wait for Module to Stop? (F) [9] error in (no error) [8] Timeout to Wait for Stop (s...) [6]</p>	<p>Send the Stop request to the Module's Main.vi.</p> <p>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.</p> <p>Note: The Timeout to Wait for Stop value is ignored if 'Wait for Module to Stop?' is set to FALSE.</p>			
Show Panel		 <p>error in (no error) [8] — [0] error out</p>	Send the Show Panel request to the Module's Main.vi.			
Hide Panel		 <p>error in (no error) [8] — [0] error out</p>	Send the Hide Panel request to the Module's Main.vi.			

Name	Type	Connector pane	Description	S.	R.	I.
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			
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" <input checked="" type="checkbox"/> in the UI Manager's subpanel "stand-alone" <input checked="" type="checkbox"/> 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.			

Name	Type	Connector pane	Description	S.	R.	I.
System Message			<p>Broadcasts a "System Message" consisting of an Identifier (string), a Payload (variant) and a Source (enum).</p> <p>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</p>			

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

Scope:  → Protected |  → Community

Reentrancy:  → Preallocated reentrancy |  → Shared reentrancy

Inlining:  → Inlined

A.1.3. MODULE START/STOP CALLS

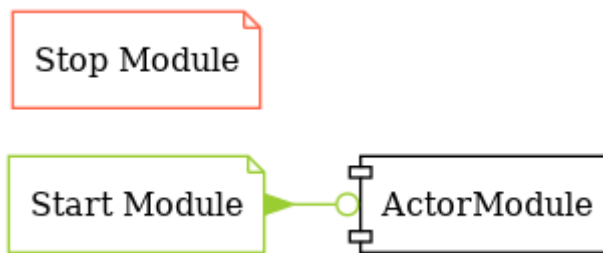


Table 7. Start and Stop module callers

Function	Callers
Start Module	ActorModule.lvlib:Load Module.vi
Stop Module	

A.1.4. MODULE HELPER LOOPS

Table 8. Helper Loops Found

Name	Helper Loop Type	Events Handled
Helper (Actor) Loop	Custom	Timeout ActorModule Request Events.Enable Actor ActorModule Request Events.Disable Actor ActorModule Request Events.Register for System Messages UI.System Message ActorModule Request Events.Unregister from System Messages ActorModule Request Events.Stop Module

A.1.5. MODULE CUSTOM ERRORS



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

Module ActorModule.lvlib use the following custom errors:

Table 9. 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.1.6. MODULE CONSTANT VIS

Table 10. Constant VIs Found

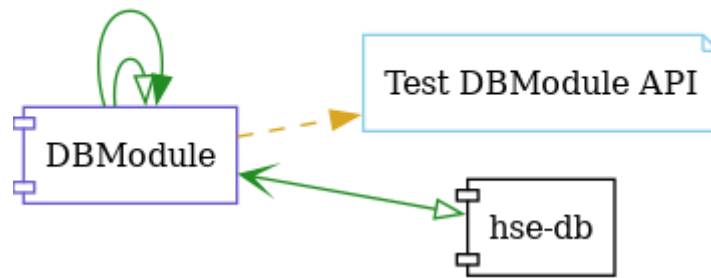
VI Name	Data Type	Value
Module Name—constant.vi	String	ActorModule
Module Timeout—constant.vi	I32	5000

A.2. DBMODULE.LVLIB

Type: Singleton

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

A.2.1. MODULE RELATIONSHIP


Table 11. Requests callers

Request Name	Callers
Configure	
Get Module Execution Status	
Hide Panel	
Prepare	
Request UI Display	
Show Diagram	
Show Panel	
Start and Configure DB_Connector	DBModule.lvlib:Main.vi

Table 12. 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 13. Used requests

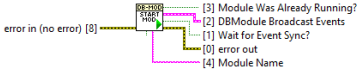
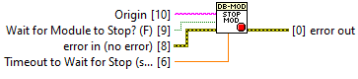
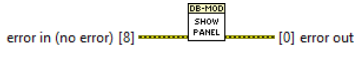

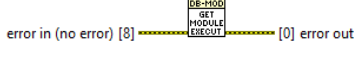
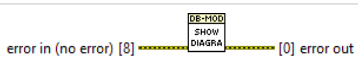


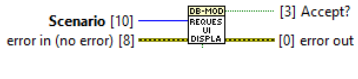

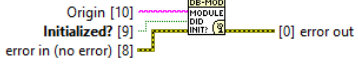

Module	Requests
DBModule.lvlib	Start and Configure DB_Connector.vi
hse-db.lvlib	DB_CONNECTOR.lvlib:Configure.vi DB_CONNECTOR.lvlib:Get DB-Type.vi DB_CONNECTOR.lvlib:Query.vi (5) DB_CONNECTOR.lvlib:Stop Module.vi


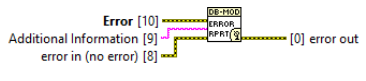


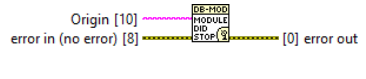


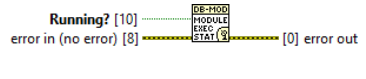


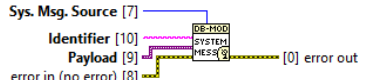

Table 14. Registered broadcast

Module	Broadcasts
☒—☒	☒—☒


A.2.2. 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.			☒
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.			☒
Status Updated	⚙️		Send the Status Updated event to any VI registered to listen to events from the owning module.			☒

Name	Type	Connector pane	Description	S.	R.	I.
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			Broadcast a system generic message.			

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

Scope:  → Protected |  → Community

Reentrancy:  → Preallocated reentrancy |  → Shared reentrancy

Inlining:  → Inlined

A.2.3. MODULE START/STOP CALLS

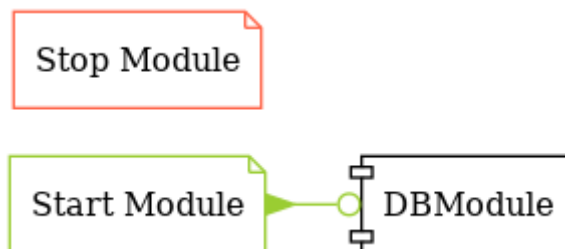


Table 16. Start and Stop module callers

Function	Callers
Start Module	DBModule.lvlib:Load Module.vi
Stop Module	

A.2.4. MODULE HELPER LOOPS



No Helper Loops Found

A.2.5. MODULE CUSTOM ERRORS



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

Module DBModule.lvlib use the following custom errors:

Table 17. 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.2.6. MODULE CONSTANT VIS

Table 18. Constant VIs Found

VI Name	Data Type	Value
Module Name—constant.vi	String	DBModule
Module Timeout—constant.vi	I32	5000

A.3. DUMMY.LVLIB

Type: Singleton

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

A.3.1. MODULE RELATIONSHIP

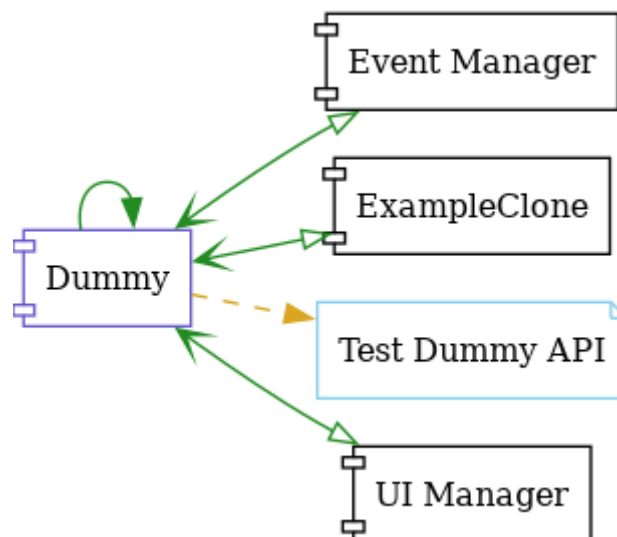


Table 19. Requests callers

Request Name	Callers
Configure	
Get Module Execution Status	
Hide Panel	

Request Name	Callers
Prepare	
Register for UI Login	
Request UI Display	
Show Diagram	
Show Panel	

Table 20. 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 21. Used requests

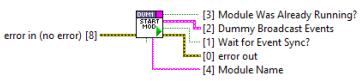
Module	Requests
Event Manager.lvlib	Set Modules.vi
ExampleClone.lvlib	Configure.vi ExampleMessage.vi Request UI Display.vi Show Diagram.vi Stop Module.vi (2)
UI Manager.lvlib	Get CLI Arguments.vi Get Module Execution Status.vi Set Modules.vi Set Run-Time Menu Visibility.vi (2) Set Window Closeable State.vi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

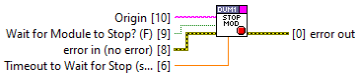















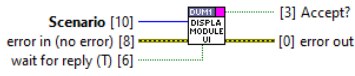

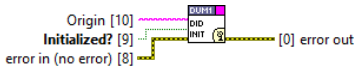



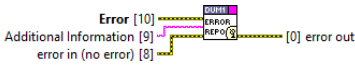



Table 22. Registered broadcast





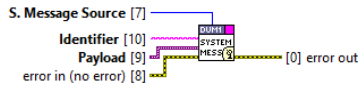

Module	Broadcasts
☒—☒	☒—☒

A.3.2. EVENT LIST

Table 23. Events


Name	Type	Connector pane	Description	S.	R.	I.
Start Module			Launches the Module Main.vi.			

Name	Type	Connector pane	Description	S.	R.	I.
Stop Module			<p>Send the Stop request to the Module's Main.vi.</p> <p>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.</p> <p>Note: The Timeout to Wait for Stop value is ignored if 'Wait for Module to Stop?' is set to FALSE.</p>			
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	→		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" <input checked="" type="checkbox"/> in the UI Manager's subpanel "stand-alone" <input checked="" type="checkbox"/> 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)			

Name	Type	Connector pane	Description	S.	R.	I.
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.3.3. MODULE START/STOP CALLS

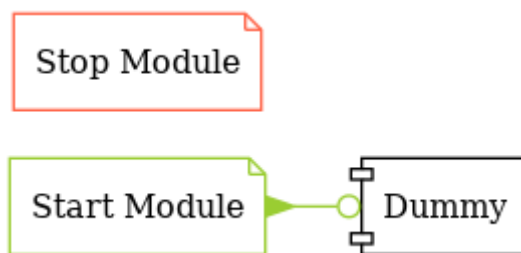


Table 24. Start and Stop module callers

Function	Callers
Start Module	Dummy.lvlib:Load Module.vi
Stop Module	

A.3.4. MODULE HELPER LOOPS



No Helper Loops Found

A.3.5. MODULE CUSTOM ERRORS



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

Module Dummy.lvlib use the following custom errors:

Table 25. 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.

Name	Code	Description
Module Not Synced	403683	%s Module was unable to synchronize events.
Request and Wait for Reply Timeout	403686	%s
My first custom error	500987	My first custom error
This is an error with input	500988	This is an error with input %s
Error ring	500989	Error ring on block diagram

A.3.6. MODULE CONSTANT VIS

Table 26. Constant VIs Found

VI Name	Data Type	Value
Module Name—constant.vi	String	Dummy
Module Timeout—constant.vi	I32	5000

A.4. EXAMPLECLONE.LVLIB

Type: Cloneable

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

A.4.1. MODULE RELATIONSHIP

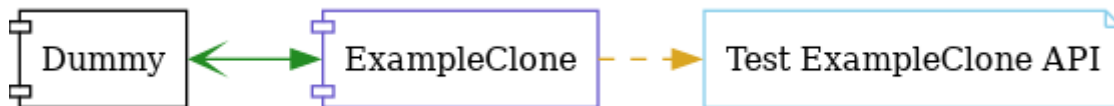


Table 27. Requests callers

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

Table 28. 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

Broadcast Name	Listeners
Status Updated	Test ExampleClone API.vi
System Message	Test ExampleClone API.vi
Update Module Execution Status	Test ExampleClone API.vi

Table 29. Used requests

Module	Requests
☒—☒	☒—☒

Table 30. Registered broadcast

Module	Broadcasts
☒—☒	☒—☒

A.4.2. EVENT LIST

Table 31. Events

Name	Type	Connector pane	Description	S.	R.	I.
Start Module			Launches the Module Main.vi.			
Stop Module			<p>Send the Stop request to the Module's Main.vi.</p> <p>If Wait for Module to stop? is TRUE, then this VI will not complete execution until the Module Main VI has stopped running.</p> <p>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.</p> <p>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.</p> <p>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.</p> <p>Note: This VI was modified by the Validate DQMH Module tool to remove the Status Updated.vi subVI call.</p>			☒
Show Panel	☒→		Send the Show Panel request to the Module's Main.vi.			☒

Name	Type	Connector pane	Description	S.	R.	I.
Hide Panel	Request		Send the Hide Panel request to the Module's Main.vi.			
Show Diagram	Request		This VI tells the Module to show its block diagram to facilitate troubleshooting (add probes, breakpoints, highlight execution, etc).			
Request UI Display	Request and Wait for Reply		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	Request		Triggers the auto-configuration of the module			
Prepare	Request		Prepares the module UI (front panel) for display			
ExampleMessage	Request		Send a message to the module that is displayed on its front panel			
Module Did Init	Broadcast		No description found (add content in vi description)			
Status Updated	Broadcast		No description found (add content in vi description)			
Error Reported	Broadcast		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	Broadcast		No description found (add content in vi description)			
Update Module Execution Status	Broadcast		No description found (add content in vi description)			
System Message	Broadcast		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.4.3. MODULE START/STOP CALLS

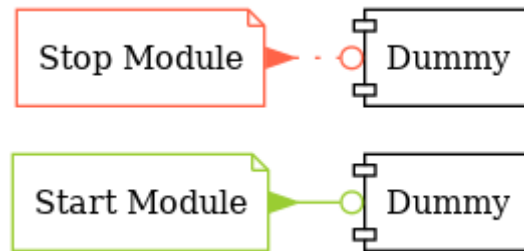


Table 32. Start and Stop module callers

Function	Callers
Start Module	Dummy.lvlib:Main.vi
Stop Module	Dummy.lvlib:Main.vi

A.4.4. MODULE HELPER LOOPS



No Helper Loops Found

A.4.5. MODULE CUSTOM ERRORS



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

Module ExampleClone.lvlib use the following custom errors:

Table 33. 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.4.6. MODULE CONSTANT VIS

Table 34. Constant VIs Found

VI Name	Data Type	Value
Module Name—constant.vi	["String","String"]	[" %d","ExampleClone"]
Module Timeout—constant.vi	I32	5000

A.5. STATEMACHINE.LVLIB

Type: Singleton

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

A.5.1. MODULE RELATIONSHIP

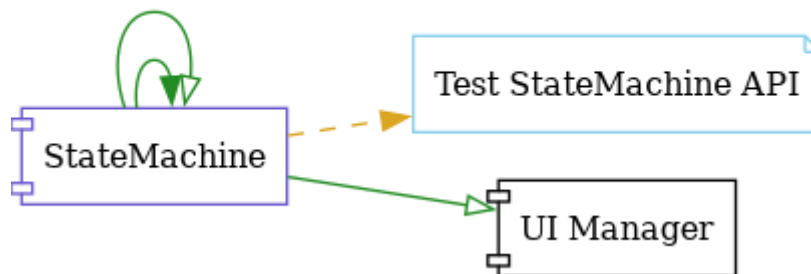


Table 35. Requests callers

Request Name	Callers
Configure	
DEMO Inject Transition	
Exit EHL	StateMachine.lvlib:Main.vi
Get Module Execution Status	
Hide Panel	
Prepare	
Request UI Display	
Show Diagram	
Show Panel	

Table 36. 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

Broadcast Name	Listeners
Update Module Execution Status	Test StateMachine API.vi

Table 37. Used requests

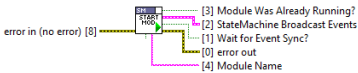
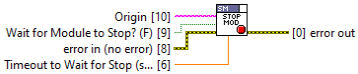
















Module	Requests
StateMachine.lvlib	Exit EHL.vi
UI Manager.lvlib	Set Window Closeable State.vi (2) Shutdown Application.vi

Table 38. Registered broadcast

Module	Broadcasts
☒—☒	☒—☒

A.5.2. EVENT LIST

Table 39. Events

Name	Type	Connector pane	Description	S.	R.	I.
Start Module			Launches the Module Main.vi.			
Stop Module			<p>Send the Stop request to the Module's Main.vi.</p> <p>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.</p> <p>Note: The Timeout to Wait for Stop value is ignored if 'Wait for Module to Stop?' is set to FALSE.</p>			
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.			

Name	Type	Connector pane	Description	S.	R.	I.
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.			
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.5.3. MODULE START/STOP CALLS

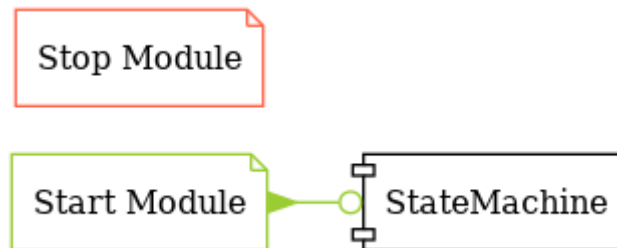


Table 40. Start and Stop module callers

Function	Callers
Start Module	StateMachine.lvlib:Load Module.vi
Stop Module	

A.5.4. MODULE HELPER LOOPS

Table 41. Helper Loops Found

Name	Helper Loop Type	Events Handled
State Machine Loop	Custom	No Event Structure Found

A.5.5. MODULE CUSTOM ERRORS



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

Module StateMachine.lvlib use the following custom errors:

Table 42. 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.5.6. MODULE CONSTANT VIS

Table 43. Constant VIs Found

VI Name	Data Type	Value
Module Name—constant.vi	String	StateMachine
Module Timeout—constant.vi	I32	5000

APPENDIX B: LIBRARIES

Misc. reuse libraries

B.1. HSE-STATE-MACHINE.LVLIB

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

Version: 1.0.0.0

B.1.1. FUNCTIONS

Table 44. 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.1.2. LIBRARY CONSTANT VIS



No Constant VIs Found

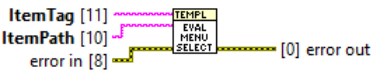


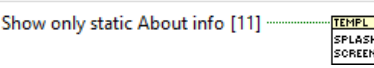
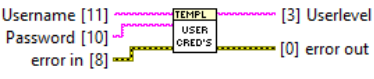
B.2. PROJECT.LVLIB

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

Version: 1.0.0.0

B.2.1. FUNCTIONS

Table 45. Functions (non private scope only)

Name	Connector pane	Description	S.	R.	I.
PROJECT_Evaluate Default Menu Selection		No description found (add content in vi description)			
PROJECT_InitLogging		Initiate the logging functions.			
PROJECT_Layout-VI		<p>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)</p> <p>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.</p>			
PROJECT_SplashScreen		No description found (add content in vi description)			
PROJECT_User Credentials		No description found (add content in vi description)			

 Scope:  → Protected |  → Community

 Reentrancy:  → Preallocated reentrancy |  → Shared reentrancy

 Inlining:  → Inlined

B.2.2. LIBRARY CONSTANT VIS

Table 46. Constant VIs Found

VI Name	Data Type	Value
PROJECT_ApplicationVersion—constant.vi	String	unreleased
PROJECT_Name—constant.vi	String	TEMPLATE

APPENDIX C: CLASSES

LabVIEW Classes

C.1. CLASSES OVERVIEW

This project contains 0 classe and 0 interface.

Table 47. 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 48. Custom errors

Name	Code	Description	Owned by
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
My first custom error	500987	My first custom error	Dummy.lvlib
This is an error with input	500988	This is an error with input %s	Dummy.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. Boylston
- DQMH® by the DQMH Consortium

Furthermore, it relies on the following tools and libraries:

- Ruby
- AsciiDoctor
- Java
- GraphViz
- PlantUML
- Kroki.io