Samples

The following chart lists all the available IBM® Rational® Rhapsody® samples by development environment and with descriptions.

Table 1. Samples

Sample Name	Environment	Description
API	C++	This project contains files with applications for RPYReporter, RPYExplorer, and C++ Client. The C++ Client also contains a Write API and a Read API with Debug applications.
Cars	C++	The cars.rpy project describes an automated rail car system including the rail car terminal and a GUI for the system. The model contains standard operations for idle cars, passengers boarding cars, selecting destinations, and requesting rail cars among other scenarios.
CD_Player	C++	Launch the `CDPlayer.rpy` project file and open the ReadMe.txt in the Files for a full explanation of this sample project. The project includes graphics and these folders: CD_player, Web, HardwarePkg, and CDTools.
CoffeeMachine	C++	The Coffee Machine model demonstrates use of control panels in Rational Rhapsody. In order to see the machine in action, Generate-Build-Run the Debug configuration and start animation.
Command Line Interface	C++	This directory contains a sample script to run Rational Rhapsody using the command-line interface.
CORBA	C++	This model contains three projects: Sdm_Observers_factory, Sdm_observers, and client_Sdm_observers to demonstrate using Rational Rhapsody to create CORBA features including a CORBA interface.
DDSTutorial	C++	DDSTutorial.rpy shows a sample Data Distribution Service for Real-Time Systems (DDS) application. For more information, refer to the readme.txt file in the folder for the sample.

Table 1. Samples

Sample Name	Environment	Description
DesignPatterns	C++	The project contains samples of common model patterns that can be copied into your projects and modified as needed: • WatchDogPattern • TimeSlicingPattern • StaticPriorityPattern • StaticAllocationPattern • SanityCheckPattern • SafetyExecutivePattern • ProxyPattern • PriorityCeilingPattern • PreemptiveMultitaskingPattern • PollingPattern • MonitorActuatorPattern • MicrokernelPattern • MasterSlavePattern • HomogeneousRedundancyPattern • HeterogeneousRedundancyPattern • HandshakePattern • FixedSizeBlockAllocationPattern • ExecutionControlPattern • DynamicPriorityPattern • CyclicExecutivePattern • BrokerPattern
DiffMerge	C++	This model contains four versions of the Elevator project (original, main trunk, branch, and base) to demonstrate the types of comparisons and results produced from the Rational Rhapsody DiffMerge tool.
Dishwasher	C++	This project defines dishwasher parts and a GUI and contains a sequence diagram, collaboration diagram, and abstract and implementation versions of the dishwasher design.
Elevator	C++	This project contains applications for the GUI, host, and target. It also includes one main use case and several sequence and collaboration diagrams.
Handset	C++	This mobile telephone project contains Word

Table 1. Samples

Sample Name	Environment	Description
		documents for the overview and requirements. The system requirements are shown with their dependencies and in an analysis package.
hhs	C++	The Home Heating System (hhs) project contains a GUI, object model diagrams, sequence diagrams, and collaboration diagrams.
HomeAlarm	C++	Home alarm security system demonstrating an MFC user interface with separate component builds. This project is also a good example of using multiple sequence diagrams.
HomeAlarmWithPorts	C++	Home alarm security system illustrates the usage of UML 2 ports to specify part interaction points.
Multicore	C, C++	Multicore demonstrates the allocation of active classes and instances to hardware core resources. The sample contains an allocation diagram, and a sequence diagram showing color denoted classes, messages, and states along lifelines.
Pacemaker	C++	This project contains a GUI, a simulation application, and detailed packages and diagrams to create this medical device.
Pbx	C++	This Private Branch Exchanges (Pbx) project for telecommunications includes a Statechart, GUI and call router and connection classes with web-enabled attributes.
PingPong	C++	This simple model demonstrates the events in a ping pong game. The project contains an object model diagram, a sequence diagram, and an animation application.
PowerWindowWithSimulink	C++	This sample demonstrates how a continuous Matlab\Simulink Block can be used inside a Rational Rhapsody model. This model requires a Simulink license.
Radio	C++	This project contains detailed use cases, sequence diagrams, and GUI, Hardware, and Test applications to perform standard functions of a radio, such as select stations and adjust

Table 1. Samples

Sample Name	Environment	Description
		volume.
ReporterPLUS	C++	This project is opened in Rational Rhapsody ReporterPLUS to demonstrate producing reports with illustrations for the C++ Dishwasher project.
RequirementsWithTags	C++	This model demonstrates generation of documents using Rational Rhapsody ReporterPLUS with tags. Use this model with the template RequirementsTable.tpl.
TestConductor	C++	This TestConductor sample contains prebuilt GUI components: CppCashRegister, CppListUsage, CppPbx, CppTestConductorAPI, and CppTestingExternalFiles. There are no executables for the other components. In order to execute TestConductor tests with non-GUI components, run the GMR (generate make run) on the executable configuration.
Tetris	C++	The tetris.rpy creates an imitation of the Tetris game including a GUI and hardware package.
UML 2.0	C++	This model is a mock-up solution of a generic protocol stack to handle voice and data calls only. The use case diagram shows the functional requirements of the system.
UPDMProject	C++	This project is a starting point for a UPDM model. It contains the SysML and UPDM profiles.
vba	C++	This model demonstrates the uses of VBA with Rational Rhapsody to perform tasks such as generating code, building a model, generating a report, and adding classes to a model. The model contains two demonstration projects: VC_6vbaDemo (vbaSample.rpy) and G3WizardDemo.rpy.
CycleComputer	C	This model uses files with dependencies and statecharts instead of classes and objects as in C++ projects. The design allows the cycle computer to be easily ported to different hardware platforms and to run the following

Table 1. Samples

Sample Name	Environment	Description
		projects: • With a Visual C++ GUI • With a console using keys R, L and P • With the IDF (automatically simulates key presses)
DiffMerge	C	This model contains four versions of the Elevator project (original, main trunk, branch, and base) to demonstrate the types of comparisons and results produced from the Rational Rhapsody DiffMerge tool.
Dishwasher	C	This project contains a GUI with files, an abstract dishwasher in an object model diagram, and sequence diagrams.
Elevator	C	This project contains an executable GUI, an object model for the host configuration, and one main use case with several sequence diagrams.
FlowChart	C	The project contains the following basic flowchart patterns that can be used and modified for your projects: • DoWhileLoop(int n) • DoWhileSelfLoop(int n) • IfThenElse() • OrderedIfThenElse() • Sequence() • SimpleIf(char * buffer) • SimpleNegatedIf(char * buffer) • Unstructured() • WhileLoop() • WhileNotLoop()
FunctionalC	C	This model is a handset protocol stack using the FunctionalC profile and use case diagrams. The project includes Analysis with comments and Architecture packages.
hhs	C	The Home Heating System (hhs) project contains a prototype application with a GUI,

Table 1. Samples

Sample Name	Environment	Description
		object model diagrams, and sequence diagrams.
Pacemaker	C	This project contains a GUI, a simulation application, object model diagrams, and sequence diagrams to create this medical device.
Pbx	C	This Private Branch Exchanges (Pbx) project for telecommunications includes a GUI application and test scenarios.
Project	C	This project is a starting point for a MicroC model.
Radio	C	This project contains an application MFCGUI, files, a detailed radio package, an object model diagram, and many use cases. The model performs standard functions of a radio, such as select stations and adjust volume.
ReporterPLUS	C	This project is opened in Rational Rhapsody ReporterPLUS to demonstrate producing reports with illustrations for the C Elevator project.
S-Function	C	This model uses the SimulinkinC (automatically added profile) and requires a Simulink license.
TestConductor	C	This TestConductor sample contains prebuilt GUI components: CPbx, and CStopWatch. There are no executables for the other components. In order to execute TestConductor tests with non-GUI components, run the GMR (generate make run) on the executable configuration.
DiffMerge	Java	This model contains four versions of the Elevator project (original, main trunk, branch, and base) to demonstrate the types of comparisons and results produced from the Rational Rhapsody DiffMerge tool.
Dishwasher	Java	This project contains a GUI with files, an abstract dishwasher in an object model diagram, and sequence diagrams.
HomeAlarm	Java	Home alarm security system demonstrating an executable test with detailed packages and

Table 1. Samples

Sample Name	Environment	Description
		shows the use of multiple sequence diagrams.
LightsManager	AUTOSAR	This sample shows how to use the AR3x_BMT profile for implementing the behavior of an AUTOSAR Atomic Software Component type.
ReporterPLUS	Java	This project is opened in Rational Rhapsody ReporterPLUS to demonstrate producing reports with illustrations for the Java HomeAlarm project.
TestConductor	Java	This sample includes two projects: CPbx and CStopWatch.
TestConductor	C++	This sample demonstrates how to test applications with Rational Rhapsody Architect for Software using the Rational Rhapsody TestConductor Add On. The sample contains specific test cases, components and instances. Launch the `CarRadio.rpy` project file and open the ReadMe.txt for a full explanation of this sample project.
CPP	Extensibility Samples for Callback API	This simple application implements the EventListenerPlugin interface and registers with Rational Rhapsody in order to receive callbacks. For more information, refer to the `readme.txt` file in the Samples/ExtensibilitySamples/CallbackAPI Samples/CPP/EventListernerPlugin directory.
RS232	Ada	This project demonstrates design and testing of a simple computer system with a keyboard. The project includes animated sequence diagrams.
RS232_95	Ada	This project demonstrates design and testing of a simple computer system with a keyboard in the "95" version. The project includes animated sequence diagrams.
SPARK	Ada	This model contains two projects to illustrate the support for the SPARK standard in Rational Rhapsody Developer for Ada. The model includes the following special features:

Table 1. Samples

Sample Name	Environment	Description
		• Stack state machine • Stack class as an abstract data type • Monitoring class showing how a data type can be extended.
Dishwasher	Ada	This project defines dishwasher parts and a GUI and contains a sequence diagram, collaboration diagram, and abstract and implementation versions of the dishwasher design.
Java	Extensibility Samples for Callback API	In the Samples/ExtensibilityChecks/CallbackAPI Samples/Java directory are two subdirectories with three applications in each: • `Plug-in` contains the ApplicationListenerPlugin, CodeGenerationListenerPlugin, and the RoundtripListenerPlugin. • `Stand-alone` contains applications that implement Listener interfaces in order to receive callbacks. For more information, refer to the `readme.txt` files in the subdirectories containing the application files.
VB	Extensibility Samples for Callback API	This simple application implements the EventListenerTest interface and registers with Rational Rhapsody in order to receive callbacks. The EventListenerTest API uses Rational Rhapsody to listen for specified events (for example, beforeProjectClose, afterProjectClose, onDiagramOpen, onFeaturesDialogOpen, onCodeGenerationCompleted, and beforeRoundtrip). Use Rational Rhapsody to perform the appropriate actions to cause these events. When one of these events occurs, the API displays a message box indicating the name of the event.

Table 1. Samples

Sample Name	Environment	Description
		For more information, refer to the `readme.txt` file in the Samples/ExtensibilityChecks/CallbackAPI Samples/VB/EventListeners directory.
VBA	Extensibility Samples for Callback API	This simple application implements the EventListenerTest interface and registers with Rational Rhapsody in order to receive callbacks. As in VB Callback API, the VBA Application Listener Client listens to Rational Rhapsody for specified events (for example, beforeProjectClose, onDiagramOpen, onFeaturesDialogOpen, onCodeGenerationCompleted, and beforeRoundtrip). However, in the VBA sample, the afterProjectClose event is not used since a VBA project is part of a Rational Rhapsody project. If Rational Rhapsody is closed, the VBA project would also be closed. For more information, refer to the `readme.txt` file in the Samples/ExtensibilityChecks/CallbackAPI Samples/VBA/VBA_EventListeners_SampleRhpProject directory.
VB	Extensibility Samples for ExternalChecks Samples	This project runs `rpyexternalchecks.exe` to check the default configuration of the model using the tools model. You can examine the VB source in the project to see how this is accomplished.
Java	Extensibility Samples for ExternalChecks Samples	This Java project provides a use case to check a user-defined check. It also contains a .hep file with a path to the check in the rhapsody project directory, as well as a property pointing to the .hep file.
.settings	Extensibility Samples for Simple plug-in	This directory contains an Eclipse JDT preferences file. For more information about creating plug-ins for the product, refer to the `How to create a`

Table 1. Samples

Sample Name	Environment	Description
		`plug-in.rtf` document in the Simple plug-ins directory.
com	Extensibility Samples for Simple plug-in	This directory contains two sample plug-ins for the product: • SimplePlugin.class • SimplePlugin.java
APIExtension	Java API	This project demonstrates how to extend the product Java-API in a stand-alone application. For more information, refer to the `readme.txt` file in the Samples/JavaAPI Samples/APIExtension directory. To extend the API in a plug-in application, refer to samples in the Samples/JavaAPI Samples/Plug-in directory.
ClassDumper	Java API	This sample contains a batch file to dump a class. For more information, refer to the `readme.txt` file in the Samples/JavaAPI Samples/ClassDumper directory.
com.telelogic.rhapsody.wfi.rhapsodyListenersExample	Java API	This plug-in demonstrates how to use the Listener extension point to receive notifications on messages and commands from the product. This plug-in implements "rhapsodyListeners" extension points. For more information, refer to the `readme.txt` file in the Samples/JavaAPI Samples/com.telelogic.rhapsody.wfi.rhapsodyListenersExample directory.
Plug-in	Java API	This diagram formatter sample demonstrates how to create a Java plug-in and how to extend the Java API for the product.

Table 1. Samples

Sample Name	Environment	Description
		Note: You can create a Java plug-in without extending the API, as well as using the API extension in a stand-alone application. For detailed instructions to use this sample, refer to the `readme.rtf` file in the Samples/JavaAPI Samples/Plug-in directory.
Obfuscator	Obfuscator	This sample is used to remove classified information from a model by generating random names and by generating hard-coded descriptions and bodies. To use it, follow these instructions: 1. Run the sample. 2. Put the full path to the original model in the "Source" field. 3. Put the full path to the destination model in the "Target" field. The model is saved to this place. 4. Press the "Start" button.
ComputerShop	PCESamples	This sample uses a computer shop's goal of maximizing profit to show how to evaluate mathematical equations and perform linear optimization. For more information, refer to the `readme.txt` file in the Samples\SystemSamples\PCESamples\ComputerShop directory.
PowerSubsystem	PCESamples	This sample demonstrates 2D and 3D plots using a constraint view that combines two parametric diagrams.
SpringAndMass	PCESamples	This sample demonstrates how to simulate a damped harmonic oscillation using a mass of an object attached to a spring.
Adms	SystemSamples	This sample describes the ADMS (Aircraft Defense Management Model) in a Rational

Table 1. Samples

Sample Name	Environment	Description
		Rhapsody project. Refer to the model overview and requirements documents within the project for a complete description of this sample.
Distiller	SystemSamples	The Distiller model uses the SysML profile in the product to reconstruct the case study example, as described in Sandy Friedenthal's SysML Tutorial. The behavioral specification in this model is not executable because the model is composed of several blocks at different stages of development. The SysML profile does not contain the full set of stereotypes required for the Friedenthal Distiller model, so additional stereotypes are defined in the "ProfileExtension" package.
NetCentric	SystemSamples	This directory contains a model that simulates a network of meteorological weather stations and that provide weather reports of current and forecasted weather conditions for the different locations of the stations. To create this model as a tutorial and then compare your version to the finished version in the directory, refer to the instructions in the `Net Centric Mini Tutorialv1.1.doc` in this directory.
SysMLHandset	SystemSamples	This is a SysML version of the Rational Rhapsody Handset model.
VB_Post_Simplifier	CustomizeCG	The TestModel subdirectory contains a Rational Rhapsody project for post simplification.
Statechart_Simplifier_Writer	CustomizeCG	This directory contains four sample directories: • Statechart_ALT_Simplifier - This Rational Rhapsody plug-in demonstrates a user-defined simplifier. The simplifier adds a <*full state name*> member to the type of the class for each state in its statechart. The attribute type is "int" and a description associates it with the state from which it originated. The

Table 1. Samples

Sample Name	Environment	Description
		simplification is enabled when the user sets the `C_CG::Statechart::Simplify` property to the "`ByUser`" value. • Statechart_Java_Simplifier - This Rational Rhapsody plug-in demonstrates a user-defined simplifier using Java. The simplifier adds a <*full state name*> member to the type of the class for each state in its statechart. The attribute type is "int" and a description associates it with the state from which it originated. The simplification is enabled when the user sets the `C_CG::Statechart::Simplify` property to the "`ByUser`" value. • Statechart_VB_Simplifier - This Rational Rhapsody plug-in demonstrates a user-defined simplifier using VB. The simplifier adds a <*full state name*> member to the type of the class for each state in its statechart. The attribute type is "int" and a description associates it with the state from which it originated. The simplification is enabled when the user sets the `C_CG::Statechart::Simplify` property to the "`ByUser`" value. • Statechart_Writer_Rules - This package Statechart_Generation provides all the rules related to the statechart generation. You must make some modifications to enable this process. For more information, refer to the individual `readme.txt` files in the four directories.