

## Wind River Test Management 3.3

As the amount of software embedded within intelligent devices continues to increase, the challenge of integrating, verifying, and validating it also expands. Engineering teams are routinely required to implement and test an ever increasing number of features within fixed

schedules. The testing challenge is multiplied when complex new run-time technologies are employed in the device and when development and test teams are geographically distributed across the globe. While the goal of testing is to find defects, the isolation and elimination of

faults in running devices can also be complex and delay time-to-market.

Given the business pressures on today's device manufacturers, best-of-class teams are turning to more agile, iterative development cycles that link testing and development tighter than ever before. They are also realizing that developing test and diagnostics tools in-house is time-consuming and expensive and distracts from the implementation of new features. Leading firms are turning to commercial solutions to improve test automation and their resulting competitive market position.

The Wind River Test Management system is a new productivity solution for device test and development teams. It complements existing test environments to help teams focus their time and resources on what really needs testing so they can deliver higher-quality products, on time and on budget, with greater confidence.

### Focus Testing Efforts

Wind River Test Management is a test execution optimization system for embedded devices. It enables unprecedented run-time visibility into devices while under test (see Figure 1). The system provides the operational feedback that test teams and their management need to achieve the highest possible quality in the least amount of time.

Wind River Test Management is a scalable system that adds value to your existing test and development environments to enhance automation, control, traceability, and feedback. It leverages unique sensorpoint dynamic instrumentation technology to measure test coverage, map test-to-code traceability, profile performance, enable white-box access, and speed diagnostics of complex devices, all at run-time.

<b>Table of Contents</b>	
Focus Testing Efforts.....	1
Optimize Device Test Execution ....	2
Optimize Testing with Run-Time Analytics.....	2
Find Untested Code .....	2
Change-Driven Optimized Testing .....	3
Identify Performance Regressions.....	3
Powerful, Open Test Engine.....	3
Import or Synchronize Tests.....	3
Hybrid Test Cases.....	4
Component-Based Test Modularity .....	4
Custom Fields.....	4
File Attachments for Data-Driven and GUI Testing .....	4
Run Any Test .....	5
Assemble and Manage Test Suites.....	5
Optimize Execution for Changed Code.....	5
Parallel, Distributed Operation	
Manage Test Devices and Labs.....	5
Organize and Reserve Networked Devices .....	6
Remotely Provision Builds .....	6
Manage Multi-core Devices and Test Beds .....	6
Tight Integration with Test Execution .....	6
Mine and Manage Test Results .....	6
Unified Test Data Management .....	6
Interactive Dashboards .....	6
Test Log Parsing and Subresults Display .....	6
Custom Reporting .....	6
Advanced White-Box Testing.....	7
Validate Fault Handling and Exception Conditions .....	7
Isolate, Repair, and Validate Run-Time Defects .....	8
Patch Software on the Fly .....	8
Multiple Deployment Options .....	8
Open CLI.....	8
Scalable, Distributed Architecture.....	9
Scalable Platform .....	9
Web-Based Applications .....	9
Developer Plug-Ins .....	9
Downloadable Agent.....	9
Powerful Sensorpoint Technology.....	10
Technical Specifications.....	10
Target Device OS Support .....	10
Target Device Processor Support .....	10
Developer Plug-ins Host OS Support .....	10
Wind River Test Management Server OS Support .....	10
About Wind River .....	11
Partner Ecosystem.....	11
Professional Services.....	11
Education Services .....	11
Support.....	11

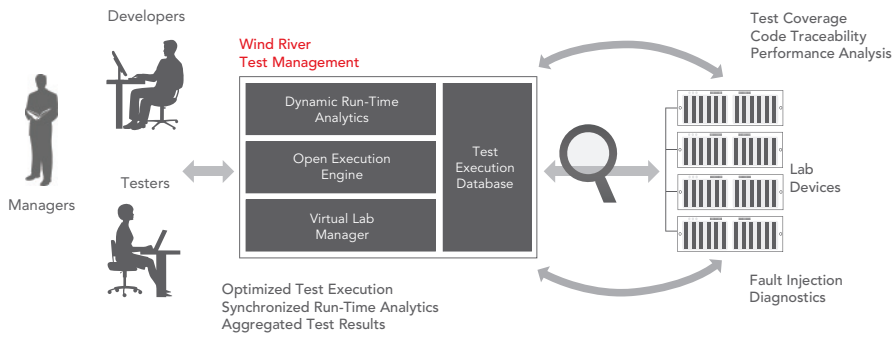


Figure 1: Wind River Test Management leverages operational feedback from devices under test to achieve the highest quality in the least amount of time

### Optimize Device Test Execution

Wind River Test Management lets you optimize test execution while providing run-time visibility into device operation under test. It provides the information you need to improve testing, reduce test times, and find bottlenecks and defects before your customers do. With the click of their mouse, system testers can do the following:

- Identify what portions of device software are not tested.
- Determine automatically what tests need to be run when code changes.
- Identify performance regressions between builds or releases.

These capabilities are delivered within an open test execution engine that was specifically designed for embedded device testing. It features a build-oriented data model and a modular, hybrid test type and utilizes an integral virtual lab manager to handle the complexity normally associated with managing and provisioning devices in a test lab.

Advanced test teams can get further value by working with their development teams to employ new white-box testing techniques that allow them to do the following:

- Automate testing of critical exception conditions.
- Isolate and repair run-time defects faster to avoid costly schedule delays.

### Optimize Testing with Run-Time Analytics

Most testing today is limited to a black-box view of the device under test. Testers work from the “outside” and observe results through whatever pass/fail information, static logs, or error codes are available. Wind River Test Management enables a “white-box” view into actual device operations when under test. Critical information can be gathered from running equipment, providing a clearer picture of test effectiveness, performance, defects, and more. The net result is testers can gather the key feedback they need from complex devices and store and manage this information in sync with test execution.

### Find Untested Code

It’s often difficult for test managers to measure the quality of their test plan or optimize test execution. It’s often impossible to know specifically what

modules of device software were actually tested for any given test run. Traditional code coverage tools provide a fine-grained view of unit test comprehensiveness but are typically too heavy for use in the QA lab on functional and system tests.

Wind River Test Management can detect and record specifically what software functions within a build are being tested or not tested. This run-time test coverage capability is an on-demand feature that can be enabled and disabled for specific modules and test runs. Run-time test coverage uses Wind River’s unique sensorpoint dynamic instrumentation technology to enable on-demand metrics with minimal hassle for testers and minimal impact on performance or footprint.

To enable this feature, Test Management automatically generates test coverage instrumentation on user-selected modules. An agent that is downloaded on demand into the lab device allows the test coverage instrumentation to be deployed and run without requiring a recompilation of the software or reloading or stopping the device.

After running the build content analysis operation on a module, users can view a list of the functions and modules in the build. If multiple modules or source files were included in the analysis, their relationships are shown in a tree structure for easier viewing. Similar views are available when users select functions for test coverage analysis. The test manager or tester can select test coverage for a

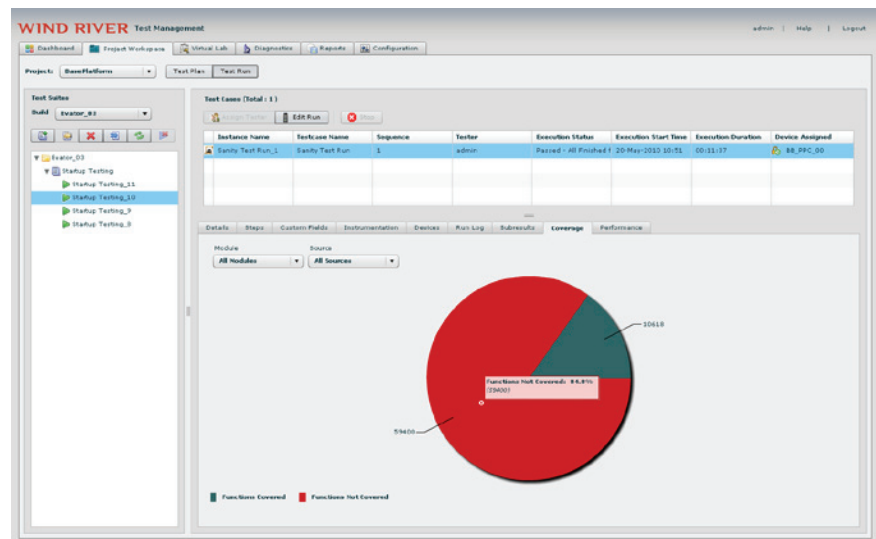


Figure 2: Run-time test coverage provides visibility into which device software functions have been tested and which have not

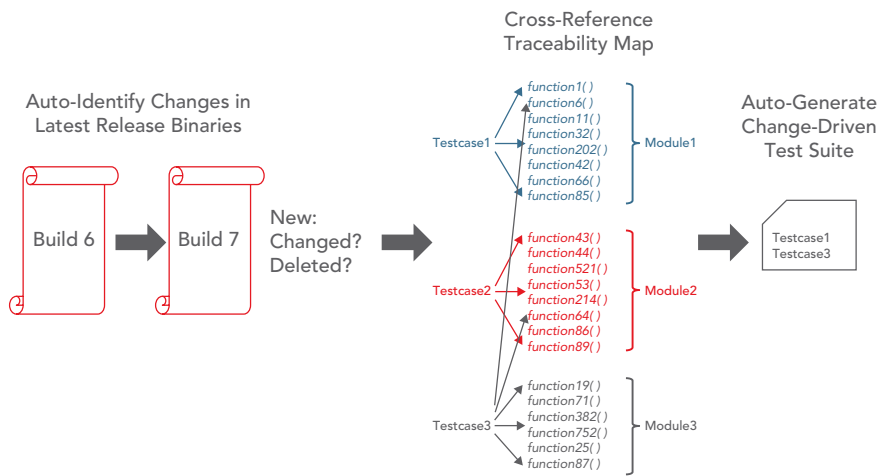


Figure 3: Wind River Test Management binary differencing and test-to-code traceability helps teams save time and focus regression testing efforts on changed code

given test run and coverage metrics will be automatically gathered. Coverage information can be aggregated over multiple test runs to give a complete picture. Coverage instrumentation is disabled and removed automatically when complete so there is no residual impact on the device.

Test coverage results are reported using Wind River Test Management graphical reports and dashboard. Developers and testers can click charts to explore which functions have not been executed so they can select or develop additional test cases to improve tested quality (see Figure 2). Comparative graphs allow trends to be analyzed between builds.

### Change-Driven Optimized Testing

Wind River Test Management automatically identifies changes between software builds (changed, new, and deleted code). It also leverages run-time test coverage information to generate a traceability map that shows the linkage between test cases and code at run-time. The system can then use this information to automatically determine just what test cases are required to run when code has changed. Optimized test suites can be generated by the system, saving time and guesswork for each new build (see Figure 3). This map can also be leveraged to verify traceability requirements all the way to the code level.

### Identify Performance Regressions

Often testers experience performance slowdowns between builds and need to do comparative studies to find the source of regressions. Teams may also be required to measure performance of devices under test to validate that execution times meet specifications or to help isolate bottlenecks. The ability to do this is usually limited to performance that can be observed manually or gathered through predefined measurements exposed externally through standard interfaces.

Wind River Test Management provides a unique way to profile run-time device performance by allowing testers to

dynamically add performance measurement probes on any function within the device software, on demand and without a software rebuild. Performance probes allow performance metrics to be gathered with minimal hassle for testers and minimal impact on device performance or footprint and validated against test requirements.

Run-time performance profiles are captured for functions executed by selected test cases, and results are reported using the Test Management graphical interface and dashboard (see Figure 4). Developers and testers can click charts to explore function execution times to validate timing or compare results across builds to isolate problem areas.

### Powerful, Open Test Engine

Wind River Test Management includes an open text execution engine that can run tests from any source or synchronize the gathering of run-time analytics with external test harnesses.

### Import or Synchronize Tests

Wind River Test Management can act as a test repository or can synchronize with externally stored tests. Users can bring their test cases into the system with a variety of individual or batch import mechanisms, including add-in tools for Microsoft Office.

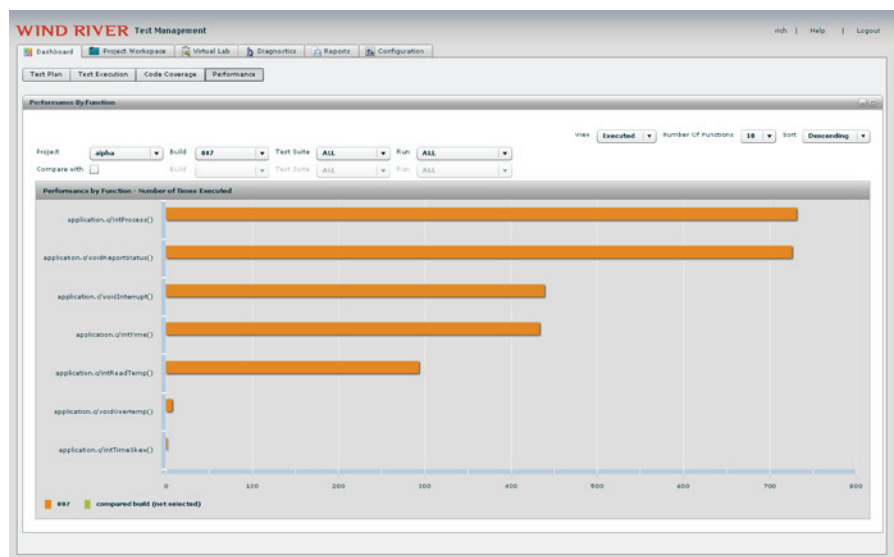


Figure 4: Run-time performance profiling offers a nonintrusive, dynamic approach to monitoring execution times of critical device functions under test

Alternatively, test developers can specify the file system location of the script, and the test case will only include a pointer to the script rather than the script itself.

Open interfaces can be utilized to synchronize external test harnesses with the Wind River Test Management run-time analytics rather than tests moving from legacy systems.

### Hybrid Test Cases

Wind River Test Management is based on a hybrid test case type that can consist of both manual and scripted test steps. This enables sophisticated device test sequences and facilitates migration from manual to automated testing (see Figure 5). Test teams can utilize automated scripts within a manual test process, executing these interactively along with manual steps within a managed environment. This hybrid test feature also helps provide a migration path from manual to fully automated testing as test teams gradually replace manual steps with scripts. A fully automatic test execution mode is offered for test cases that contain only automated test scripts and can run to completion as batch operations without tester involvement (e.g., overnight regression testing).

### Component-Based Test Modularity

Embedded device testers often have test scripts that are used over and over in multiple test cases. Test suites can be simplified if these common tests are stored only in one location and then included by reference elsewhere. It can also be advantageous to compartmentalize tests along with related “setup” and “cleanup” sequences. For example, a set of scripts may be required to initialize test equipment, reboot a device, or restore a device to a “clean” state for further testing.

Wind River Test Management implements a modular test case structure and user-defined, reusable, scripted test components that let test developers build more self-contained and modular tests. A reboot or login script that is used repeatedly need only be defined once then used in multiple test cases

throughout the system. This test model enhances maintainability as the test component need only be updated in one location.

This component-based test model has been designed so that its unique run-time analytics information, such as test coverage, is only gathered from the main test functions, not from ancillary setup or cleanup sequences. Modularity also helps Wind River Test Management better optimize test execution based on changed code. Better compartmentalization allows the system to more easily “mix and match” tests from any source, knowing that each test case is self-contained.

### Custom Fields

Often companies or projects have their own standard test template that includes unique attributes they want to associate with test cases. Wind River Test Management allows users to create their own test case templates by adding custom fields to the base test case definition. The system also allows users to customize test execution status descriptions, search test cases based on standard and custom fields, and modify test run duration metrics.

### File Attachments for Data-Driven and GUI Testing

Device test developers often prefer to segment reusable or changing test data from the actual test scripts by moving test datasets into external files. Graphical user interface (GUI) test designers often capture images of correct screens so that testers can compare current vs. expected results. Device testers may also need to send datasets, scripts, or commands directly to the device as part of a test sequence.

All these scenarios are now supported with the file attachments feature in Wind River Test Management. Test developers can attach any type of file to a test case or to the individual test steps within an interactive test sequence, allowing development of sophisticated data-driven tests for all types of devices. Attachments can also be useful for test teams that are migrating from manual test processes captured in Microsoft Word or Excel files to a more automated approach. These existing test descriptions can be attached to test cases, allowing users to refer to familiar documents while running their tests, thereby speeding adoption of the automated system.

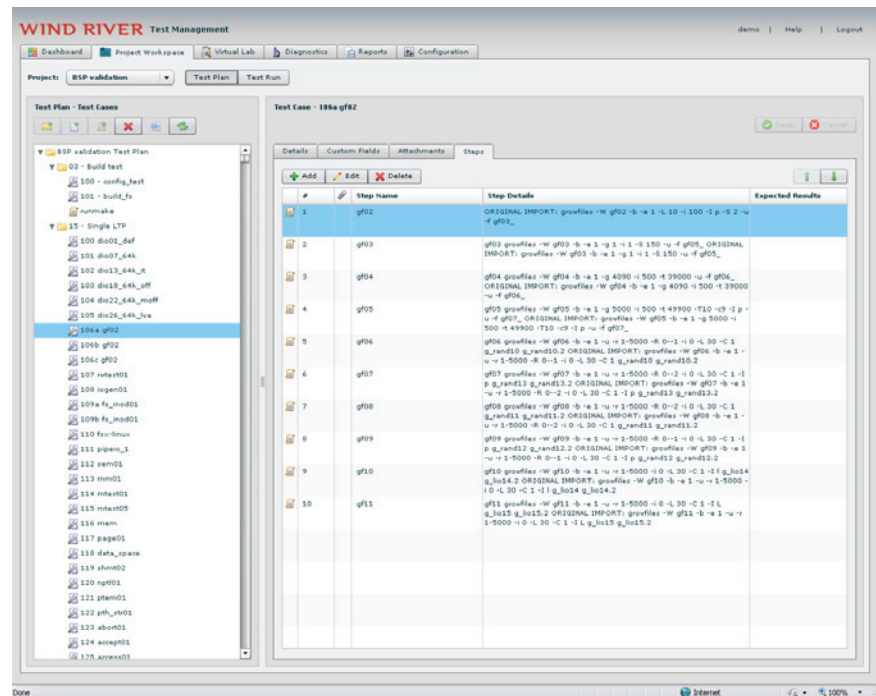


Figure 5: Users can create a modular, interactive or automated test case repository within the system or import or synchronize with external repositories

## Run Any Test

Wind River Test Management can capture and execute manual or scripted tests, whether external or embedded. Users can utilize any scripting or programming language (TCL, PERL, PYTHON, C, custom, proprietary, etc.). The system can be configured to manage and execute any type of test.

## Assemble and Manage Test Suites

In Wind River Test Management, users create purpose-built test suites, selecting any test cases from the test repository into the optimal groupings for test runs on reserved devices.

The test planner sets up one or more test suites—collections of test cases grouped to validate a certain build or modules—based on their knowledge of what software features are available and ready to test (see Figure 6). Test suites are aligned with specific software builds of the target application. The test manager can also assign a specific tester to run the suite on a reserved virtual lab device.

Before running the test suites, Wind River Test Management validates that the correct software build is loaded on the assigned test device. If not, the correct build can be downloaded and installed on the device through the virtual lab manager. These actions can also be triggered automatically via an open command-line interface (CLI) that allows, for example, a test suite to be run nightly after a new build is complete.

The test execution application automatically executes the scripted tests on the Test Management server targeting the designated device, then collects and stores the resulting data from the test. For manual tests, the system walks testers through the sequence of instructions to execute on the designated device and requires them to input pass/fail results and comments for each test case. The system automatically collects and stores test results, stdio, coverage, performance, diagnostics, or other logs for each test run in a historical database.

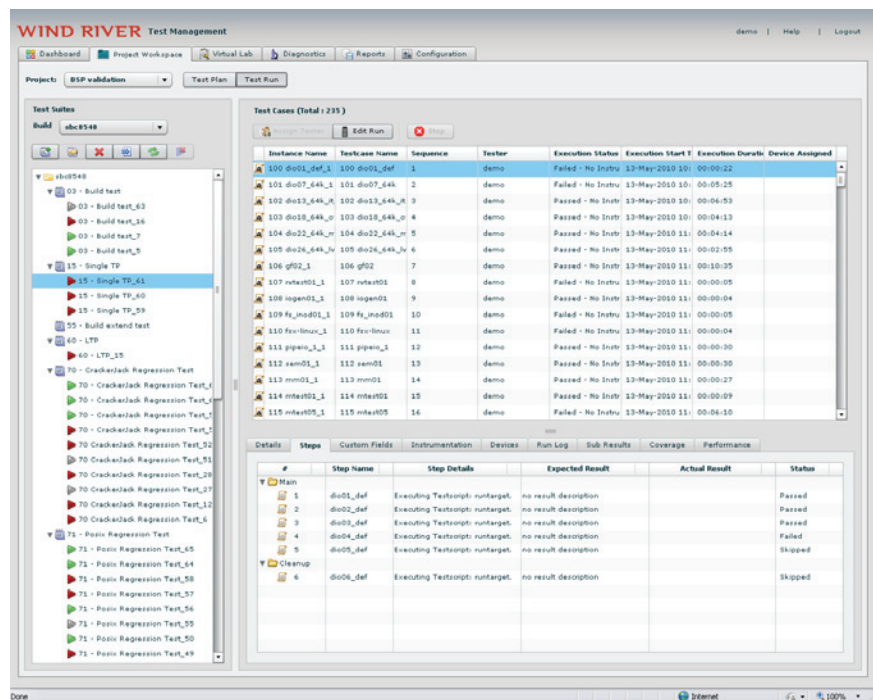


Figure 6: Testers can assemble and run specific test suites against reserved devices, with test results and associated logs, coverage, and performance data collected and managed

## Optimize Execution for Changed Code

In addition to manual creation of test suites, Wind River Test Management can leverage the test-to-code traceability map created by the test coverage process to auto-generate test suites for new builds. The system's build comparison facility identifies functions that have changed between two binary software builds. This information can be viewed by test personnel using the included Build Change Summary report to assess code churn for planning purposes.

The system then uses this build change information and the test-to-code traceability map to automatically create a list of recommended test cases that need to be re-run based on code changes. This can significantly minimize the time and resources required to verify the new builds and eliminate guesswork.

## Parallel, Distributed Operation

Users can choose the test case execution order to allow sequential or parallel execution. Test execution can be

distributed over multiple execution engines covering different labs or devices. The scripted test case mechanism and sequence control can also be used to manage and execute command scripts that initialize test equipment or simulators as part of a test sequence.

## Manage Test Devices and Labs

Many companies today have expensive capital equipment spread across the many labs in their company for use by testers. However, it's often hard to know what devices are available that have the desired configuration, and it's difficult to provision them with appropriate software for testing. This problem is compounded when teams are located at multiple sites.

Wind River Test Management includes a virtual lab manager that provides tight integration of test target management with test execution and analytics, simplifying lab setup and management and providing a full life cycle test-device audit trail (see Figure 7). The virtual lab manager saves time and hassle when

working with test lab devices and allows maximum utilization of people and capital equipment resources.

### Organize and Reserve Networked Devices

Set up virtual lab groupings of your test equipment in local labs or across corporate networks. You can search for desired equipment characteristics and check out and reserve devices for testing and track their utilization.

### Remotely Provision Builds

Remotely track and load new software builds at the push of a button. Save time while ensuring correct test configurations. You can leverage terminal servers or power controllers to remotely access or reboot devices.

### Manage Multi-core Devices and Test Beds

Control and run tests against multiple cores or devices at the same time. The system also manages the built-in run-time analytics to intelligently gather white-box data from the live device under test. Test coverage analysis, performance measurement, fault injection, or dynamic diagnostics instrumentation can be deployed across multiple cores or processors, and information is then collected in sync with tests.

This version of Test Management allows the definition, reservation, and management of “test beds,” collections of devices and associated test equipment, allowing easy management of varied devices, simulators, and instruments embedded testers often need. The test bed feature enables design of sophisticated tests that operate on a group of devices and may, for example, communicate with each other as part of a test. The system provides the means to send both tests and instrumentation to each of the devices in the test bed and manage the test results.

### Tight Integration with Test Execution

With Wind River Test Management there is no need for hard coding IP addresses into test scripts. Using the virtual lab

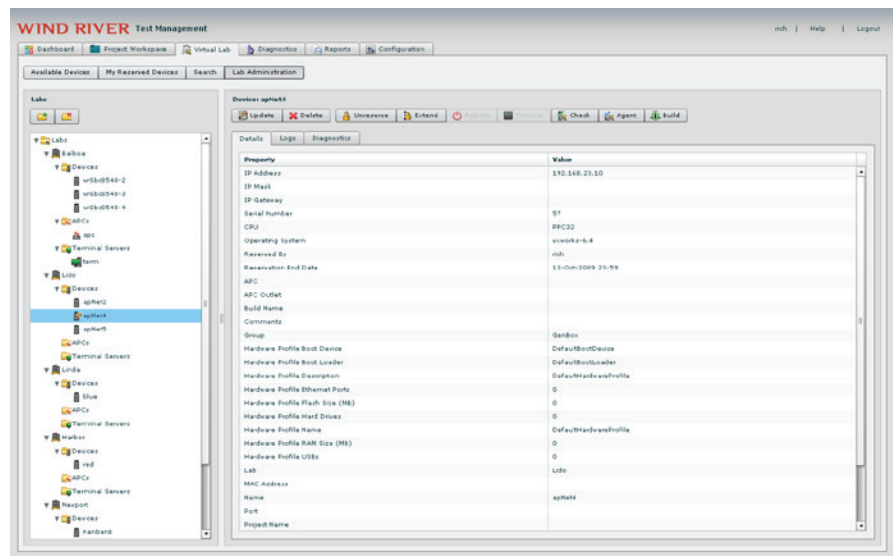


Figure 7: Virtual lab manager lets testers find, reserve, provision, and use test devices worldwide

manager, users assign test suites to available devices at run-time. The system maps test execution to selected devices and tracks and records results in the system database.

### Mine and Manage Test Results

Wind River Test Management is built around a choice of scalable, relational databases that allow storage and management of all test results and analytics for mining, analysis, and archiving.

### Unified Test Data Management

All test information is stored in an underlying database so you can retain and mine information. Projects, builds, test cases, test suites, and device information are stored with associated test results. This includes test coverage, performance, traceability, and diagnostics information. The system also captures any logs that are generated during the test process and stores them along with pass/fail results.

### Interactive Dashboards

Key data is easily accessed for review through your web browser using the system’s user interface and interactive graphical dashboards. Users can select specific test runs, modules, builds, projects, and so on, and access current or cumulative data displays.

### Test Log Parsing and Subresults Display

Often custom-built test suites or third-party test tools generate custom test results logs during a run of a block of tests. This facility allows users to create parser scripts that post-process the custom or third-party logs and expose summary data from the log file to Test Management. A “subresults” feature allows summary results to be displayed in the new subresults area of the Test Run tab so that users can easily drill down to see subresults of a test run. These subresults are also stored in the Wind River Test Management database along with other test information for review and reporting.

Subresults can be generated as a batch operation at the end of a test run, or alternatively, users can leverage a Test Management command-line interface command and user-created script to update the subresults periodically during the test run. This can let users see intermediate points of execution or results during a complex, automated test run.

### Custom Reporting

Wind River Test Management includes the industry-standard Jasperreports as its built-in reporting engine. This allows you to use reporting tools such as iReports to create custom reports from stored data.

Once defined, report formats can be stored in the system and run from the Reports tab using your web browser. Reports can be generated in popular formats including PDF, XLS, RTF, and HTML so they can be viewed or shared as standard files.

This release of Test Management includes a Build Change Summary report that shows exactly what code has been changed/added/deleted between builds, allowing managers to understand code churn and do change-based risk assessment in iterative or agile processes. Other reports such as Coverage Trends and Functions Not Covered allow test managers to understand how thorough their tests are and monitor coverage progress throughout the development life cycle. Additional summary reports include test coverage, test execution results, and lab device asset utilization.

### Advanced White-Box Testing

Wind River Test Management's run-time analytics engine provides most testing capabilities with the click of a mouse. For teams that have access to the device software sources or can work with their developers, the system provides some additional powerful white-box testing features.

### Validate Fault Handling and Exception Conditions

Black-box testing is effective for certain functional testing but it can be difficult to get the device into a desired state or to validate that unusual states such as error conditions are handled correctly. Many testers want to get access to device internals at run-time, using white-box techniques.

For advanced test teams, Wind River Test Management provides unique white-box testing features through use of sensorpoint dynamic instrumentation. Developers, working to the test plan, can create reusable sensorpoint probes that can be dynamically injected into the device by testers (or automatically by test scripts) before a test run to input data, force state, read or log data at a given subsystem, or inject faults or failures.

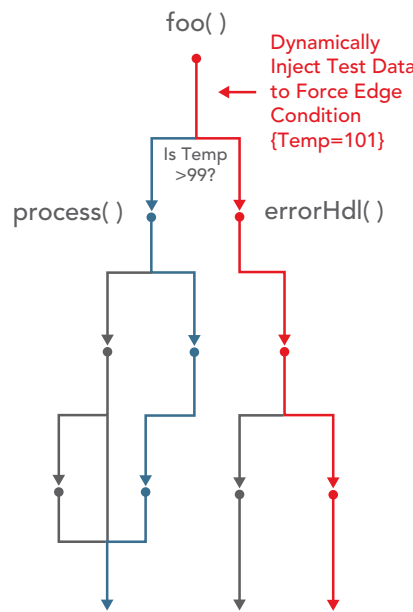


Figure 8: Dynamic fault injection features enable manual or automated manipulation of device data or software in the device under test at run-time.

The following are example uses of sensorpoints for white-box testing:

- Inject specific values into variables.
- Force faults or conditional expressions to execute code paths, error handlers, or failover.
- Log internal device data generated at run-time for validation against expected parameters.
- Alter the system's environment (e.g., change values read from sensors, set CPU and peripheral controllers register values, modify streaming data).

- Set the device to a specific state that may only be reached after hours of continuous operation (e.g., buffers full, queues empty).
- Eliminate the use of sophisticated test equipment (e.g., forcing the hop count to a large value in a data packet, stubbing out interfaces).

Sensorpoints can be created to manipulate any data, register, or line of code on demand and on the fly (see Figure 8). Sensorpoints can be enabled dynamically in sequence with the associated test cases, manually or automatically via scripts. After the test is run, the sensorpoint can be disabled or removed before subsequent testing. Sensorpoints and the data collected by them (e.g., test coverage or performance data) can also be set to be persistent across device reboots if required by sophisticated test sequences.

This white-box testing technique enables design-for-testability methods and provides system testers access to device internals so they can better validate every device state, saving time and hassle. White-box testing with sensorpoints can be combined with other Wind River Test Management features such as run-time test coverage so testers can, for example, inject error conditions to force error handler execution while validating improved test coverage dynamically.

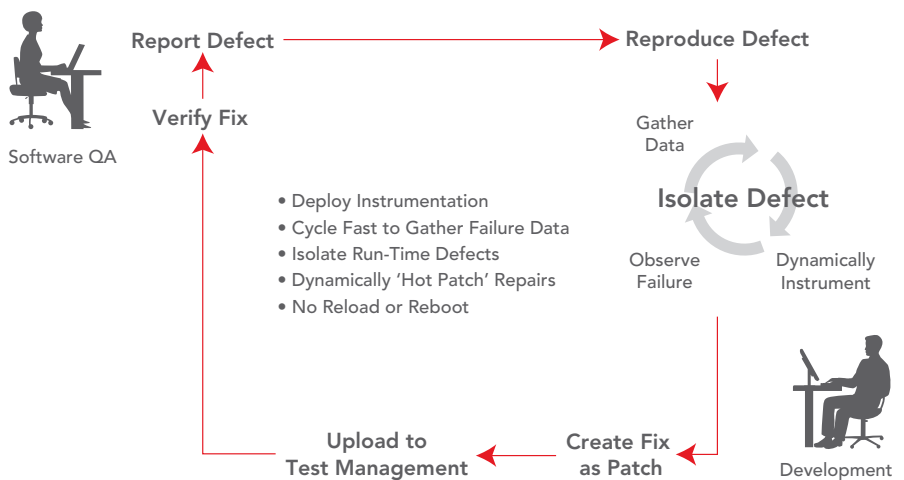


Figure 9: Dynamic sensorpoint instrumentation enables developers to access defective device run-time data, skipping build, reload, restart cycles, to accelerate time to fault resolution

## Isolate, Repair, and Validate Run-Time Defects

An inevitable outcome of testing is the discovery of defects. Isolating and resolving defects in embedded devices requires the collaboration of testers and developers. Ideally system testers can assist in the isolation of defects so they can provide developers the details they need to resolve the problem. And developers have access to detailed device operational data rather than just observed symptoms, so they can get to the root cause fast.

Unfortunately the real world is rarely ideal. Traditional debug tools are too intrusive to be deployed in system test labs, and typical device user interfaces, static logs, and error codes don't often provide visibility into device internals or key defect data. Wind River Test Management provides a new way to diagnose run-time defects that allows developers and testers to work together to rapidly solve tough problems in the lab (see Figure 9).

Wind River Test Management allows developers to create sensorpoint instrumentation that probes within a lab device to gather run-time data dynamically. Developers can focus on an area of concern by temporarily injecting instrumentation (e.g., printf or log statements) to rapidly collect data that will point to the source of the problem. Device code can be instrumented on the fly, without rebuilding or rebooting the device, to dramatically decrease the cycle time required to isolate defects.

Diagnostic sensorpoints are created by software developers using developer plug-ins (or command-line equivalents) included with Test Management. The Test Management system is then used to deploy the sensorpoints to the failing device and rerun the tests or work through their test team counterparts to accomplish this. Any log data that is generated by sensorpoints is captured in the Test Management system and viewed via a browser or developer's desktop. This cycle can be repeated as often as necessary to rapidly isolate the root cause of the problem.

Sensorpoints are managed in the Test Management system along with relevant device and build information and can be easily reused by testers. This enables a growing knowledgebase of reusable diagnostics instrumentation that can be deployed by testers when they encounter a defect, without the need for developer assistance.

Many development and test organizations today are in different physical locations and time zones. This can significantly limit test productivity as developers and testers try to work together to solve tough problems. Wind River Test Management allows remote diagnostic access to managed lab devices and rapid deployment of defect-isolating instrumentation. This lets developers and testers collaborate much more productively around real device data rather than failure symptoms.

### Patch Software on the Fly

When developers isolate a defect to a given function or module, they typically create a fix by modifying the source code in their development environments at their desks. Once complete, the fix needs to be validated. Typically the process to validate is to check in the fix, rebuild the device software, and provide it to the tester as a new build to reload and retest on the device. This process can be quite time consuming and can hold up test execution for hours or days. There are further delays and resource requirements if the lab has to be reset and any long-run test suites run again to the point of failure.

Wind River Test Management provides a dynamic patching mechanism where once a fix is created by developers, the fix can be published in the form of a patchpoint that can be simply injected into the lab device—without a full recompile, rebuild, and reload—without stopping it. This allows the defect repair cycle to be dramatically reduced and allows prospective fixes to be added to the failing device in the lab, ideally while still in its failing state, so that the repair can be quickly validated by the test team.

The workflow and technology for dynamic patchpoints is the same as with sensorpoint instrumentation. The main difference is that patchpoints replace existing code with new code, while sensorpoints insert new code or data within existing code.

If the fix does not actually address the problem or, as is often the case, the fix introduces other problems, the developer and tester can efficiently collaborate to gather further diagnostic data and iterate on subsequent patches until validation is complete. Once validated, patchpoints can be made persistent and reused indefinitely in the lab and can be made obsolete when the fix is permanently committed to a subsequent build.

### Multiple Deployment Options

Wind River Test Management fits within your existing test environment, offering multiple deployment options to leverage your investments. Wind River Test Management's run-time analytics can be connected with your existing test systems allowing external test harnesses to gather synchronized run-time information. Alternatively, Wind River Test Management can use open interfaces to control your legacy test beds, or you can migrate your tests and execution environment into the Wind River system for both storage and execution.

Users can select the appropriate strategy for their projects based on life cycle stage, business needs, and state of existing test assets. Most often, it is a combination of approaches.

### Open CLI

Wind River Test Management can perform automated or batch operations as commanded by external scripts or applications. For example, when a new build comes available, Wind River Test Management can be driven to set up and run regression test suites on the new build automatically. An open CLI provides access to server commands, allowing integration with third-party applications or external scripts for automated control.

## Scalable, Distributed Architecture

Wind River Test Management is designed to provide a scalable, distributed architecture that brings together all the testing players into a common environment:

- The Wind River Test Management server manages device and test data. It resides on a Windows or Linux server on the user's network.
- Web-browser-based application interfaces run in standard browsers using Flash 10 and allow users to access the Test Management system over their intranet.
- Developer plug-ins are installed on software developers' desktops to allow sensorpoint and patchpoint development within their existing programming environments.
- A device management (DM) agent is built in conjunction with the user's software and downloaded to the device under test on demand to manage dynamic instrumentation.

## Scalable Platform

At the center of Wind River Test Management is a scalable server environment built on standard, proven, high-performance Java, J2EE, and relational database technology. The server acts as the test and data repository and the central collaboration point for all users. The Test Management server stores tests, manages the virtual labs, deploys sensorpoints on demand, and gathers test results and device data during test execution.

The server has built-in security at multiple levels to protect system access or device information. Users have secure logins mapped to assigned privileges; all application and device communication can be encrypted and authenticated.

The server environment has been specifically architected to promote scalability in distributed teams. A

component of the server, the DM engine, can be deployed separately to enable multiple host and scripting environments or remotely to move test processing closer to labs.

## Web-Based Applications

Users interact with Wind River Test Management through their standard web browser, whether it's for executing tests, gathering run-time analytics, managing labs, or reviewing results. The interactive user interface and Flash-based graphs speed user operation while avoiding the cumbersome system administration common with heavyweight desktop clients.

## Developer Plug-Ins

Developers are offered other interfaces to support sensorpoint and patchpoint development from their development desktop. Developer plug-ins extend the Wind River Workbench Eclipse-based integrated development environment with tools for creation and deployment of sensorpoints (see Figure 10). They also

allow for the analysis of resulting logs and diagnostics data.

The plug-ins let developers simply point to a function or line of code within the Workbench graphical editor and create a sensorpoint or patchpoint where desired. The plug-ins support sensorpoint compilation and the packaging of dynamic instrumentation for deployment.

All this developer capability is alternatively available via command-line tools or a set of Eclipse plug-ins that will work with other environments.

## Downloadable Agent

The run-time analytics and dynamic instrumentation technology is enabled by a small run-time DM agent that can be embedded in the device software or downloaded on demand. The DM agent performs the housekeeping functions required for dynamic instrumentation such as sensorpoint load/unload placement within the running device and enable/disable on demand.

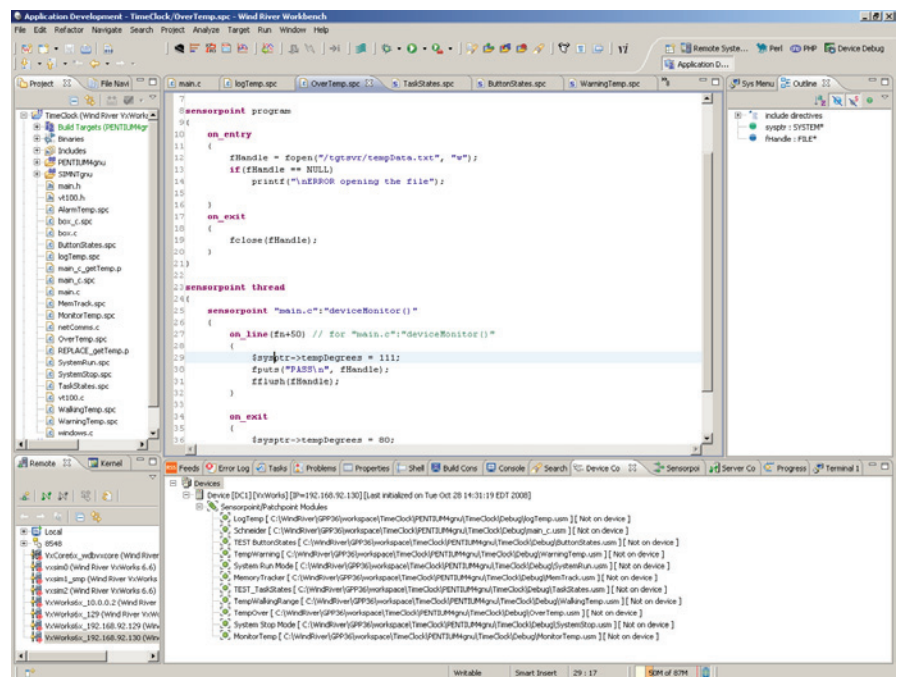


Figure 10: Create dynamic sensorpoint instrumentation or 'hitless' patchpoints with Wind River Workbench or Eclipse-based developer plug-ins or command-line tools

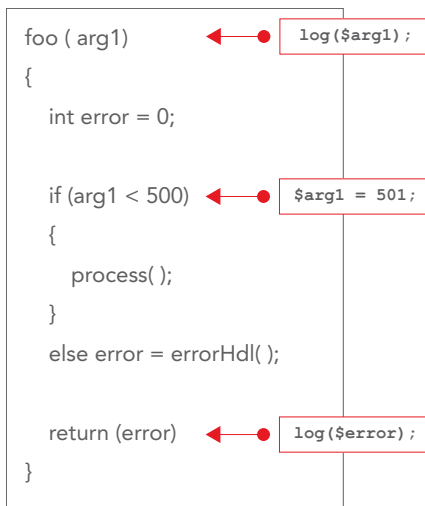


Figure 11: Sensorpoint instrumentation allows dynamic injection of code on entry, online, or on exit of a software function, anywhere within a device application, kernel, driver, or third-party libraries

For dynamic loading of the DM agent into lab devices, the system leverages VxWorks downloadable kernel modules (DKM) or Wind River Linux RPM package manager. Testers can use the virtual lab manager to query whether the DM agent is resident in a target device and, if not, it can initiate agent download, install, and startup. The agent can be left in the device or software build or removed automatically after testing is complete.

### Powerful Sensorpoint Technology

Integral to Wind River Test Management is Wind River's sensorpoint technology. Sensorpoints let users gain visibility into running devices and measure test coverage, perform white-box testing, speed diagnostics and patching, or measure run-time performance.

A sensorpoint is C language software used to instrument "live" C and C++ applications dynamically without modifying the application source code, rebuilding the application, reflashing boards, or rebooting the device. Sensorpoints allow developers and testers to inject custom code into a running executable to gather data, add functionality, or change the flow of control. Sensorpoints can be inserted on entry or on exit of a function or on a specific line of code (see Figure 11). Sensorpoints can access local and global variables within the scope of a function

Sensorpoints are minimally intrusive on device performance and footprint. They can be disabled and enabled remotely, individually, or in groups, separate from their installation. Sensorpoints can be created by development engineers, then used by test teams to identify defects on running systems.

Sensorpoints do not need to be defined upfront when original source code is written. Their power stems from the fact that they can be defined and applied very late in the development cycle. Whether inserting a log function for diagnostics, injecting data, or forcing a failover, sensorpoints can be created any time in the project life cycle. The target application does not have to be rebuilt and the device does not need to be restarted in order to put sensorpoints to work.

The system also offers patchpoints to enable rapid update of device software. Patchpoints are a variation of sensorpoints that perform dynamic patching of code. The main difference is that patchpoints replace existing code with new C or C++ code, while sensorpoints insert new code within existing code.

### Technical Specifications

Wind River aims to support the widest range of processor, operating system, host, and infrastructure software combinations with Wind River Test Management. This includes delivering the same key test and diagnostics capabilities in a Wind River Hypervisor-based environment running single or dual guest operating systems (VxWorks and/or Wind River Linux). This helps speed the testing and time-to-market of complex hypervisor and multi-core based devices. The list of supported platforms is subject to change and frequently expanding to cover both Wind River and non-Wind River platforms.

Note that test case development, test execution, virtual lab, and dashboard capabilities are independent of device operating system, processor, and scripting language; these capabilities will generally work with any device offering cross-platform test execution.

The run-time analytics features—diagnostics, dynamic test coverage, dynamic performance profiling, dynamic patching, and white-box testing capabilities—are platform-specific and rely on the downloadable DM agent to be resident in the target device. DM agent support is available for a growing number of target OS and target processor architecture combinations as described here.

The following is the support matrix at the time of publication. Contact your Wind River representative for the most recent information on supported platforms, hardware requirements, and technical specifications.

### Target Device OS Support

- VxWorks 6.1–6.8 platforms
- Wind River Linux 1.4, 1.5, 2.0.2, 3.0 based platforms
- VxWorks 5.5.1
- Other Linux, UNIX, and real-time operating systems

### Target Device Processor Support

- ARM/XScale
- IA-32
- MIPS
- PowerPC

*Note: VxWorks 5.5.1 is currently supported only on PowerPC and IA-32 with Windows hosts; 64-bit processors are not supported at this time.*

Wind River Test Management generally supports processors within these architecture families that are supported by the VxWorks or Wind River Linux platforms. See Wind River Sales for further details and exceptions.

### Developer Plug-ins Host OS Support

- Windows XP Professional SP2 or Vista
- Red Hat Enterprise Linux 4 or 5, Red Hat Fedora Core 7
- SUSE Desktop Linux 10, SP1, SUSE Linux/openSUSE 10.2
- Solaris 9 (update 9/05) or 10

### Wind River Test Management Server OS Support

- Red Hat Enterprise Linux Workstation with option 5
- openSUSE Linux 11.0
- Microsoft Windows 2000 Professional (Service Pack 4) or XP Professional (SP2)

## About Wind River

### Partner Ecosystem

Wind River's world-class partner ecosystem ensures tight integration with solutions from a wide range of premier software providers. Our partners extend the capabilities of Wind River Test Management by enabling support for key technologies in a number of fast-moving industries.

### Professional Services

Wind River Professional Services, a CMMI Level 3-certified organization, enables you to reduce risk and focus on development activities that add value and differentiate your design. Our team delivers device software expertise within structured engagements that directly address key development challenges and contribute to the success of our clients. Our track record of timely delivery and in-depth understanding of market and technology dynamics makes Wind River a valuable implementation partner for clients worldwide.

Wind River Professional Services offers a variety of services around Wind River Test Management to meet your needs, including installation and configuration, tool integration, deployment best practices, and platform migration.

## Education Services

Education is fundamentally connected not only to individual performance but to the success of your project or your company. Lack of product knowledge can translate into longer development schedules, poor quality, and higher costs. The ability to learn—and to convert that learning into improved performance—creates extraordinary value for individuals, teams, and organizations. To help your team achieve that result, Wind River offers flexible approaches to delivering Wind River Test Management product education that best fits your time, budget, and skills development requirements.

### Support

Wind River provides full technical support for Wind River Test Management at centers worldwide. Support is also available 24 hours a day at our Online Support website ([www.windriver.com/support](http://www.windriver.com/support)) or by email at [support@windriver.com](mailto:support@windriver.com). Visit Wind River Online Support for fast access to product manuals, downloadable software, and other problem-solving resources. Additional features, including patches and technical tips for common problems, are available for all customers on subscription. Online Support visitors can also access a community of developers to discuss their issues and experiences.