MAPUSOFT®

WHITE PAPER

Beyond Performance Measurement

How Runtime Profiling Enables Intelligent Embedded Software Optimization

MapuSoft
Application/Platform Profiler

Measure. Analyze. Optimize. Automate.

www.mapusoft.com | info@mapusoft.com

Table of Contents

Executive Summary

Embedded software engineers have long relied on profilers to measure execution times, identify bottlenecks, and improve application performance. While these capabilities remain essential, today's embedded software modernization projects demand significantly more than timing statistics. They require actionable insights that can be translated directly into optimized software.

The MapuSoft Application-OS Profiler transforms runtime performance analysis into a closed-loop optimization framework. Rather than simply reporting execution metrics, the profiler captures detailed runtime behavior of both applications and operating system services. This information is then consumed by Application Common Operating Environment (AppCOE®) and the Optimized Target Code Generator (OTCG) to generate application-specific runtime environments optimized for actual execution behavior.

Unlike traditional profilers that serve primarily as debugging tools, the MapuSoft profiler becomes an integral part of the software modernization workflow. It provides engineers with objective runtime measurements, identifies performance bottlenecks, validates optimization efforts, and supplies the intelligence required to generate smaller, faster, and more deterministic runtime environments.

Introduction

Performance has always been a primary concerns in embedded software development. Whether developing aerospace, automotive, industrial automation, telecommunications, or medical systems, engineers must understand precisely how software behaves under real operating conditions.

Traditional profilers provide valuable information regarding execution times and function usage, but they generally stop there. Engineers must manually interpret the collected data and determine how to optimize the application.

Modern embedded software modernization projects introduce additional engineering challenges:

- Porting applications across operating systems
- Supporting multiple hardware architectures
- Preserving deterministic real-time behavior
- Reducing runtime footprint
- Validating performance after migration
- Optimizing operating system services based on actual application behavior

Meeting these objectives requires far more than a timing report.

The MapuSoft Application-OS Profiler addresses these challenges by transforming runtime measurements into application-specific optimization decisions that improve both performance and maintainability.

Challenges with Traditional Profilers

Most commercial profilers were designed primarily as debugging utilities.
Although they provide execution timing information, they typically suffer from several limitations:

- Require intrusive instrumentation
- Depend on live debugger connections
- Provide little insight into operating system behavior
- Offer limited thread-level analysis
- Produce reports without automated optimization
- Operate independently of code generation tools

Consequently, performance analysis often becomes an isolated engineering activity rather than an integrated component of software optimization.

The MapuSoft Approach

The MapuSoft Application-OS Profiler was designed specifically for embedded software modernization.

Unlike conventional profiling tools, it executes independently on the target platform, collecting runtime information without requiring modifications to application source code or a continuous host-target debugging connection.

Fully integrated with AppCOE®, the profiler provides comprehensive visibility into application execution while supplying runtime intelligence directly to the Optimized Target Code Generator (OTCG).

The profiler supports:

- Application profiling
- Platform profiling
- Thread-level analysis
- Operating system API analysis
- Runtime timing analysis
- Differential performance reporting
- Profile-guided optimization
- Automated application-specific runtime generation

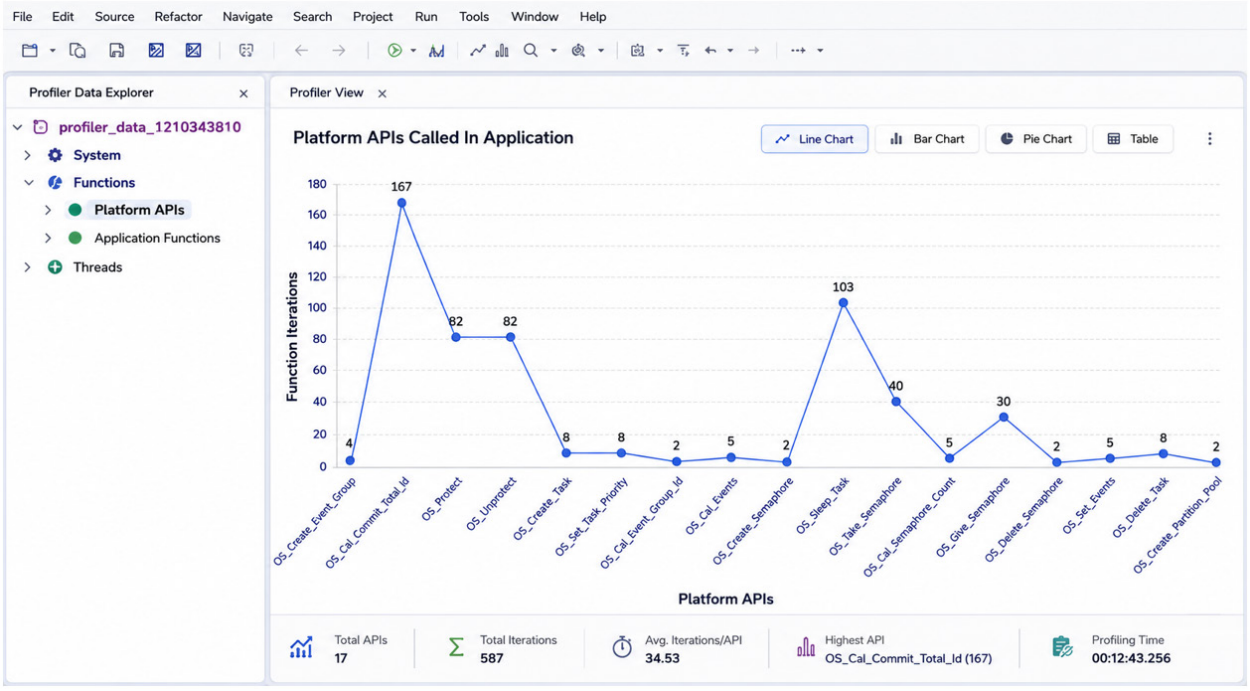

Figure 1 – Execution Frequency of MapuSoft APIs and Application Functions

How the Profiler Works

The Application-OS Profiler integrates seamlessly into the software modernization workflow with minimal impact on application execution. Once enabled during the build process, the profiler transparently monitors runtime activity while the application executes normally on the target hardware.

The workflow consists of six straightforward steps:

1. **Enable the Profiler**

   The Application-OS Profiler is enabled during the application build process using the AppCOE configuration.

2. **Execute on the Target Platform**

   The application executes normally while the profiler transparently monitors runtime behavior.

3. **Collecting Runtime Measurements**

   High-resolution timers record every OS Abstractor® API call, operating system service invocation, and selected application function, capturing detailed execution statistics with minimal runtime overhead.

4. **Generate the Profiling Data File**

   Runtime performance information—including execution counts, execution times, thread activity, API usage, and resource utilization—is continuously written to a profiling data file stored locally on the target.

5. **Import Results into AppCOE**

   Following execution, the profiling data file is imported into AppCOE®, where runtime information is processed for comprehensive analysis.

6. **Analyze and Optimize**

   Engineers analyze collected data using a rich collection of graphical reports, including:

   - Bar charts
   - Line charts
   - Pie charts
   - Scatter plots
   - Execution timelines
   - Timing reports
   - Function call relationships
   - CPU utilization reports
   - Differential comparison reports

The profiling data is also supplied directly to the Optimized Target Code Generator, allowing AppCOE to optimize OS Abstractor® APIs according to actual runtime behavior and generate application-specific optimized target code.

Application Profiling

Application profiling measures execution characteristics of application functions and user-defined code segments.

1. **Automatic Profiling**

   Automatic profiling requires no modifications to application source code.
   Using the AppCOE graphical interface, engineers simply select which functions should be monitored.
   During execution the profiler automatically records:

   - Function execution count
   - Minimum execution time
   - Maximum execution time
   - Average execution time
   - Total execution time
   - Function call relationships

   This enables immediate performance analysis while preserving application integrity.

2. **Manual Profiling**

   For specialized investigations, developers may insert lightweight profiling hooks around any code region requiring detailed measurement.

   Typical examples include:

   - Device drivers
   - Communication protocols
   - Third-party libraries
   - I/O operations
   - Critical algorithms
   - Processing loops

   This flexibility allows virtually any portion of an application to be measured precisely.

Platform Profiling

Application performance and deterministic behavior depend heavily on operating system behavior.

Platform profiling extends analysis beyond application functions by measuring operating system interface usage.
The profiler records:

- Cross-OS API calls
- Native operating system services
- API invocation frequency
- Thread utilization
- Resource usage
- Execution timing

When migrating software from one RTOS to another, engineers can simultaneously analyze both the legacy operating system APIs and their corresponding OS Abstractor® implementations.

Runtime Performance Metrics

During execution the profiler transparently captures detailed runtime information. The profiler captures several categories of runtime information, including:

Operating System Analysis

- API invocation frequency
- Operating system service utilization
- Thread usage
- Resource consumption

Timing Analysis

- Minimum execution time
- Maximum execution time
- Average execution time
- Total cumulative execution time

Application Analysis

- Function execution
- Driver performance
- Library execution
- Custom code segments

Thread-Level Analysis

Performance may be analyzed on a per-thread or per-task basis, enabling engineers to isolate:

- Scheduling delays
- Synchronization issues
- Priority inversion
- Resource contention
- Execution bottlenecks

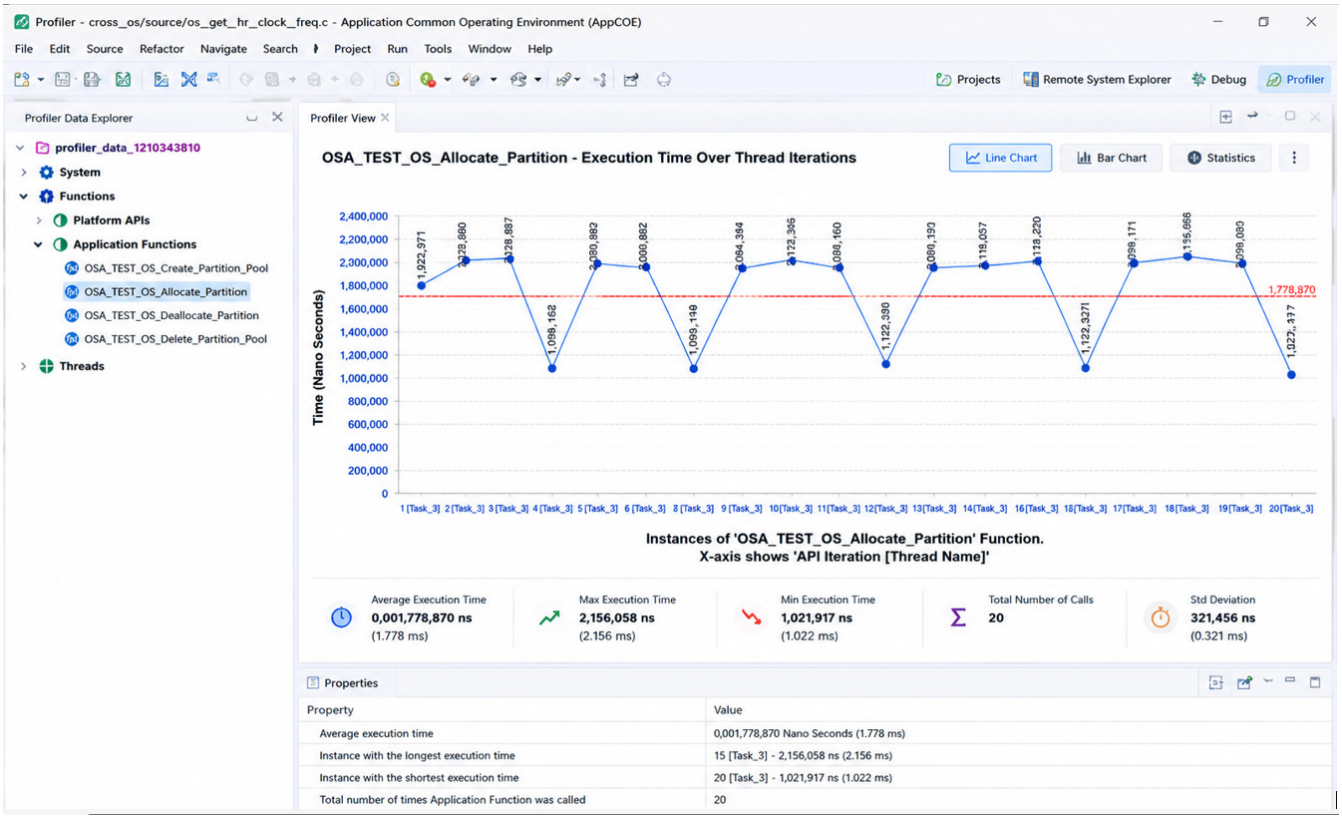

Figure 2 – Execution Time of Application and OS Functions

From Measurement to Optimization

This is where the MapuSoft profiler fundamentally differs from conventional profilers.
Traditional profilers answer:
“Where is execution time being spent?”
The MapuSoft profiler answers much more:

- Which operating system APIs dominate execution?
- Which APIs are rarely executed?
- Which operating system services are never used?
- Which services deserve optimization?
- Which runtime components can be removed?

These measurements become direct inputs to the Optimized Target Code Generator.

Engineering Insight

Static analysis identifies what operating system services an application requires. Runtime profiling reveals how those services are actually used. Combining both provides a level of optimization that neither approach can achieve independently.

Profile-Guided Optimization

One of the most powerful capabilities of the MapuSoft platform is the integration between runtime profiling and automated code generation.

The optimization workflow consists of:

1. Analyze application dependencies.
2. Execute the application with profiling enabled.
3. Collect runtime behavior.
4. Identify:
    - Frequently executed APIs
    - Rarely used APIs
    - Unused services
    - Execution bottlenecks
    - Resource utilization patterns
5. Supply profiling information to OTCG.
6. Generate an optimized application-specific runtime.

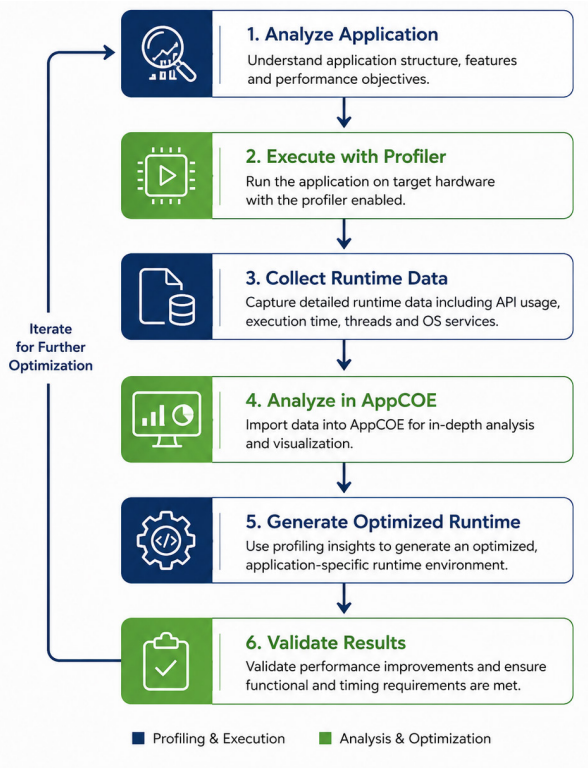

Figure 3 – Closed-Loop Optimization Workflow

Rather than optimizing every operating system service equally, AppCOE concentrates optimization effort on those APIs that dominate execution time.

Further, each OS API service can be optimized based on its actual usage level by the application using profiler data, ensuring optimization effort is focused where it delivers the greatest performance benefit.

The resulting runtime is:

- Smaller
- Faster
- More deterministic
- Easier to maintain

Engineering Insight

Traditional embedded optimization relies heavily on engineering assumptions. The MapuSoft Application/Platform Profiler replaces assumptions with measured execution data, enabling optimization decisions based on actual runtime behavior rather than estimates.

Differential Performance Analysis

Optimization should always be measurable.

The Profiler Differential Report compares two profiling sessions and quantifies performance improvements across:

- Software revisions
- Compiler optimizations
- Hardware platforms
- Operating systems
- Runtime configurations

Compared metrics include:

- Execution counts
- Average execution time
- Minimum execution time
- Maximum execution time
- Cumulative execution time
- Thread behavior

Application-OS Profiler Differential Report

Comparison of Two Profiling Sessions

Profile Information	Baseline	Optimized
Project	Flight_Control	Flight_Control
Target OS	VxWorks	VxWorks
CPU	ARM Cortex-R52	ARM Cortex-R52
Profiling Duration	300 sec	300 sec
Generated By	AppCOE Application-OS Profiler	AppCOE Application-OS Profiler

API Timing Comparison

OS API	Calls	Avg Time Before (µs)	Avg Time After (µs)	Improvement
OS_Create_Task	25	18.4	12.7	**31%**
OS_Delete_Task	25	14.8	10.2	**31%**
OS_Create_Semaphore	112	7.3	5.1	**30%**

Figure 4 – Differential Timing Report

Differential reporting provides objective evidence that optimization efforts have produced measurable improvements.

Engineering Benefits

The Application-OS Profiler delivers significantly more than runtime measurement.

It establishes a continuous optimization feedback loop that improves application efficiency throughout the software modernization lifecycle.

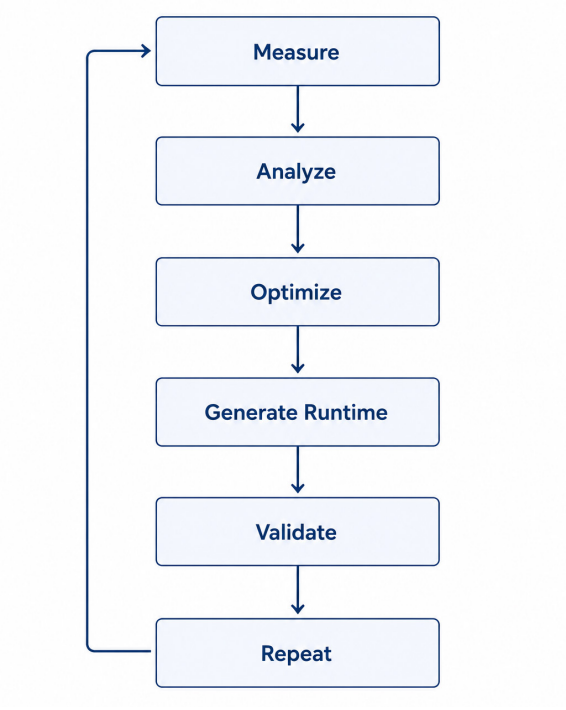

Figure 6 – Continuous Optimization Lifecycle

Major benefits include:

- Complete visibility into runtime behavior
- Objective identification of execution bottlenecks
- Thread-level performance analysis
- Profile-guided optimization
- Application-specific runtime generation
- Reduced memory footprint
- Improved execution speed
- Enhanced deterministic behavior
- Automated optimization of OS Abstractor® APIs
- Quantitative validation using differential reports
- Seamless integration with AppCOE and OTCG

Key Takeaways

The MapuSoft Application-OS Profiler extends far beyond traditional runtime performance analysis. By integrating directly with AppCOE and the Optimized Target Code Generator (OTCG), it transforms runtime measurements into actionable engineering intelligence that improves both application performance and software modernization outcomes.

Comprehensive Runtime Visibility

Gain complete visibility into how embedded applications utilize operating system services under actual operating conditions, replacing assumptions and estimates with objective runtime measurements.

Profile-Guided Optimization

Use measured execution data to drive application-specific optimization. Runtime profiling enables the Optimized Target Code Generator (OTCG) to generate target code that reflects the application's actual execution characteristics rather than relying solely on static analysis.

Intelligent OS API Optimization

Identify frequently executed OS Abstractor® APIs, infrequently used services, and unused operating system functionality. AppCOE optimizes critical APIs while eliminating unnecessary runtime components, reducing code size and execution overhead.

Improved Performance and Deterministic Behavior

Generate smaller, faster, and more deterministic runtime environments through profile-guided optimization. Reduced execution latency, lower memory consumption, and predictable real-time performance are especially valuable for embedded, industrial, aerospace, automotive, medical, and safety-critical applications.

Accelerated Performance Tuning

Rapidly identify execution bottlenecks through detailed timing analysis, API usage statistics, thread-level profiling, execution timelines, and graphical performance reports, allowing engineering effort to focus on the areas with the greatest performance impact.

Objective Performance Validation

Validate optimization efforts using quantitative differential reports that compare multiple profiling sessions across software revisions, compiler optimizations, operating systems, hardware platforms, and runtime configurations.

Seamless Integration with AppCOE

The profiler operates as a fully integrated component of the AppCOE ecosystem, providing a unified workflow for runtime analysis, visualization, differential reporting, and automated generation of optimized application-specific target code.

Complementary to Static Analysis

Static analysis identifies which operating system services an application requires, while runtime profiling reveals how those services are actually used. By combining both perspectives, AppCOE achieves a level of optimization that neither technique can provide independently.

Transform Profiling into an Optimization Engine

Unlike conventional profilers that simply report performance statistics, the MapuSoft Application-OS Profiler serves as the intelligence engine behind AppCOE's profile-guided optimization framework. Runtime measurements are automatically transformed into application-specific optimization decisions, enabling the Optimized Target Code Generator (OTCG) to produce highly optimized target code with improved performance, reduced memory footprint, and enhanced deterministic behavior.

Conclusion

The MapuSoft Application-OS Profiler redefines the role of profiling in embedded software modernization. Rather than functioning solely as a diagnostic tool, it serves as the intelligence engine behind a data-driven optimization workflow.

By combining comprehensive runtime analysis with AppCOE's Optimized Target Code Generator, the profiler enables engineering teams to move beyond performance measurement toward automated, application-specific optimization.

The result is embedded software that is smaller, faster, more deterministic, objectively validated, and specifically optimized for the way the application executes.

This combination of runtime intelligence, automated optimization, and seamless integration with AppCOE makes the MapuSoft Application-OS Profiler a key differentiator in embedded software modernization and one of the platform's most powerful engineering capabilities.

By transforming runtime profiling from a diagnostic activity into an intelligent optimization process, MapuSoft enables organizations to preserve software investments while delivering measurable improvements in performance, determinism, and long-term maintainability.