# Welcome to This Guide

## Introduction

Welcome to HP Performance Center of Excellence Best Practices.

This guide provides concepts, guidelines, and practical examples for implementation of Performance Centers of Excellence in various organizations.

## About HP Center of Excellence

HP is the market leader in Automated Performance Testing. This is a discipline that leverages products, people, and processes to minimize risks during deployment of applications, upgrades, or patches. At its core, automated performance testing is about applying production workloads to pre-deployment systems while simultaneously measuring system performance and end-user experience. A well-constructed performance test answers questions such as:

- Does the application respond quickly enough for the intended users?
- Will the application handle the expected user load and beyond?
- Will the application handle the number of transactions required by the business?
- Is the application stable under expected and unexpected user loads?
- Are you sure that users will have a positive experience on go-live day?

By answering these questions, automated performance testing quantifies the impact of a change in business terms. This, in turn, clearly predicts the risks of deployment. An effective automated performance testing process helps you make more informed release decisions, and prevents system downtime and availability problems.

One of the approaches to achieving maximum value of automated performance testing is to build an internal Center of Excellence. Many IT organizations have adopted the Center of Excellence (CoE) model as a practical way to consistently and continuously improve IT operations. In the CoE model, the QA infrastructure, processes, and best practices are consolidated and centralized. This provides a management and automation platform for application delivery, as well as leadership, consulting, and support services to help companies optimize application quality.

A CoE also provides the entire organization with visibility into standardized quality metrics and key performance indicators (KPIs). This helps to keep all stakeholders informed, and keep applications aligned with business objectives.

**HP Performance Center** is the automated performance testing product which allows native implementation of a Performance CoE. It is a global cross-enterprise performance testing tool that you install on your organization's own infrastructure.

- Performance Center enables managing multiple, concurrent load testing projects across different geographic locations without any need to travel between them.
- Performance Center administers all internal load testing needs.
- With Performance Center, you can manage all aspects of large-scale load testing projects, including resource allocation and scheduling, from a centralized location accessible through the Web.
- Performance Center helps streamline the testing process, reduce resource costs, and increase operating efficiency.
- Performance Center helps to pinpoint performance bottlenecks.
- Performance Center enables you to determine the number of users the application under test can scale up to. This information gives clues as to what can be done to increase the application's load capacity.

The purpose of this book is to assist IT organizations to assess their current situation and successfully build and maintain their own Performance Center of Excellence. These guidelines are based on in-depth research using real-life best practice data and expertise from a wide range of sources, including:

- HP operating system administrators
- HP professional services organization
- Technical documentation and books written by industry experts
- Many customer testing organizations

Using these guidelines will help make the initial creation process efficient and help achieve maximum value from the CoE.

## Audience

This guide is intended for:

- Performance CoE Managers
- Performance Engineers
- QA Managers

## Prerequisites

Before using this book, you should understand the performance validation field, either from a performance engineer perspective or from a developer perspective. Knowledge of the major phases of the Software Development Life Cycle (SDLC) is an advantage. You should also be familiar with the typical business processes in real IT organizations.

# Structure

This guide is organized as follows:

- *Welcome to This Guide*
- *Introduction to Performance CoE*
- *Build Successful CoE*
- *Operate CoE Successfully*
- *Get People for CoE Success*
- *Conclusion*
- *Appendix – PCoE Interview's Questionnaire*

**Note:** Wherever the term "CoE" is used, it is assumed it is used in the context of the Performance Center of Excellence. Therefore, both CoE and PCoE are used throughout the book.

# Feedback

If you have questions, comments, or valuable best practice information to share, send a message to the following email address:

*lt_cust_feedback@hp.com*

# 1 Introduction to Performance CoE

Performance Center of Excellence is both an approach and the resulting entity in mature IT organizations. While there are many ways to conduct performance testing, customer experience has proven that establishing a centralized, procedures-based body greatly improves business results in less time.

## Importance of Performance Engineering

Software engineering has been defined as the procedures, methods and tools that control the software development process and provide the foundation for building high-quality software in a productive manner. There are many dimensions to "software quality", including, but not limited to, functionality, ease-of-use, flexibility, scalability, security and performance. Many of the software engineering methodologies focus on ensuring that the software meets functional requirements while the development adheres to a strict schedule and budget.

The performance requirement, however, is becoming both increasingly difficult to manage and more necessary. The move to client / server software architectures deployed on distributed systems has made traditional, queuing theory-based analysis obsolete. This is compounded by Internet services in which information is sent to a user on a "subscription" rather than a "transaction" basis. Just as the nature of the software and computer systems has changed, so has the user community.

In many cases, system performance is evaluated far too late in the system life cycle. In the worst case, performance issues are discovered as the telephone rings; irate users or customers complaining about response time. In a slightly better scenario, the organization may undertake benchmarking prior to deployment. At the stage prior to deployment, however, the application has already been designed, coded and tested. This results in increased expenditures for computation and communication resources.

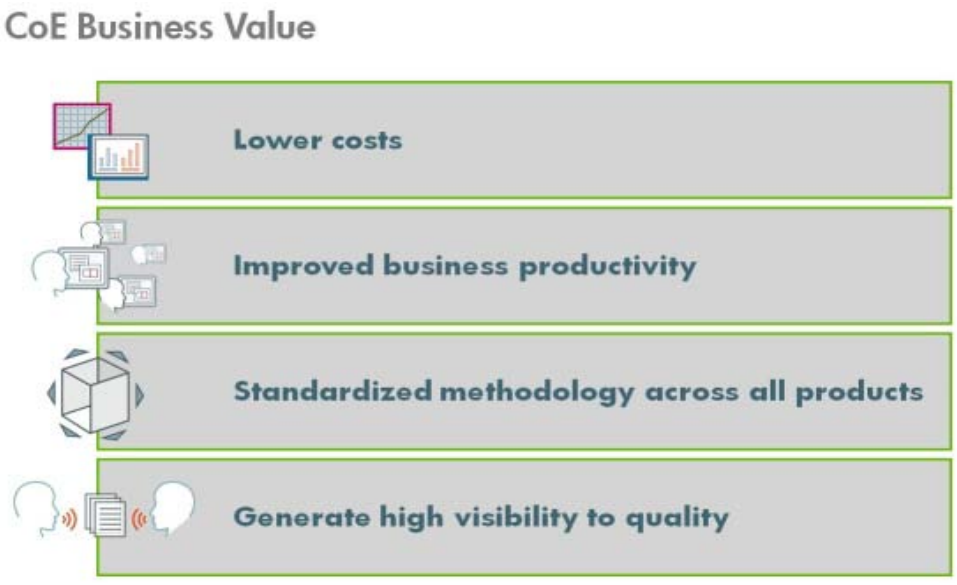
While the pressure to deliver high-quality applications continues to rise, shrinking development and deployment schedules, geographically distributed organizations, outsourcing, and high turnover rates for skilled employees make application testing more challenging. Even if there is a proper skill set

in one business unit, its achievements are not propagated to the entire organization. Many companies' IT departments face the common challenges of no sharing, no common processes, and no performance awareness.

Faced with the reality of having to do more with less, juggling multiple projects and managing diverse and distributed project teams, many organizations are adopting test management methodologies and are turning to automated test management tools to help centralize, organize, prioritize, and document their testing efforts.

This document explores the challenges and rewards of test management and provides practical ways to help you implement an organized and structured testing process.

One such approach that has been embraced by key customers and analysts alike is the Performance Center of Excellence (CoE) model—essentially a logical or physical entity that drives standardization and processes across an organization, to improve quality, consistency, effectiveness, and efficiency.




A Performance CoE eliminates dissimilar tool usage, inefficient performance testing, and inaccurate analysis. A Performance CoE can also provide the entire organization with a central platform for performance management that will create visibility into critical performance parameters of the delivered application, keeping everyone informed and keeping applications aligned with business objectives.

# Why CoE?

The simple answer to this not-so-simple question is because a Center of Excellence drives valuable results. Numerous analyst reports attribute the importance of CoE to the significantly enhanced ability of an organization to meet or exceed the goals that the center supports. When governance, a support structure, guidance, measurements, and shared learning exist across an organization, success is far more likely. A successful organization can focus on specific projects goals. The need to obtain results should be the primary motivation for creating any center of excellence. Without this, the organization cannot reach its full potential.

## Why Performance Testing is Not Enough

Performance testing is a specialized skill set. It requires knowledge of the applications, the hardware, and the third party systems. It involves understanding application behavior from an architectural standpoint, modeling skills, and a systematic approach. Not all testers have these qualities, and it takes years to fully develop the proper skill set. If performance testers stay in separate project testing groups, it is hard to ensure that all applications are being properly performance tested to the same standard. Furthermore, when new technologies appear, these disparate groups will not all have the necessary expertise to support those new technologies.

In many cases, each project team develops its own performance optimization processes, using whatever tools happen to be favored by individual team members. The result is an unmanageable mix of workflows, practices, tools, and techniques that are not well documented, not well integrated, and not transferable to other groups. This leads to more than inefficiency - it significantly increases the cost and complexity of practices and has the potential to affect the morale and limit the IT staff's skill development. Different silos of the development and delivery process use their own approach to performance optimization, resulting in miscommunication, misalignment, and ultimately sub-optimal performance. This patchwork way of testing detaches project teams from IT and business management, and blurs the projects' objectives.

When it comes to planning for production deployment, many companies benchmark their systems late in the development process, if at all. As a result, systems are deployed in sub-optimal configurations that still contain undiscovered performance and availability issues—and the disparate toolsets used to enhance performance do not work together.

## Why You Cannot be Left Behind

Application performance has traditionally been overlooked because it is usually not owned by any one silo of business, despite the key fact that:

> *No matter how well applications are functionally designed, no matter how well they meet business requirements, they are virtually useless to end-users if performance is slow or unreliable.*

Unfortunately, users are more likely to remember the worst experience with a company rather than the best experience. Therefore, performance testing is a crucial component of any deployment because application performance has a direct impact on the end-user experience, which directly influences your revenue stream. Performance testing can make the difference between a satisfied customer and a customer who switches to a competing provider.

Performance problems often have a significant effect on the user experience because of the extended time and effort required to remediate performance issues. Performance issues don't always get resolved overnight, and performance fixes often require a total interruption of service.

Enhancing the speed and availability of applications can be complex, and often requires expertise that takes time to develop. At the same time, every phase of application performance optimization has become specialized due to greater complexity of platforms like SOA, virtualization, Web 2.0 and other IT trends. These complex platforms come with their own specific skill sets, tools, formats, protocols, and processes. Very often, the skills developed and processes created by individual project teams for load testing, code optimization, usage analysis, tuning, and capacity planning—are not shared within the IT department or across the lines of business.

This problem is compounded with business and IT budget constraints and constant business changes like divestures, mergers and acquisitions, and outsourcing. The modern corporation cannot continue to present an image of “business as usual” while they scatter resources and money across the company to duplicate work at best.

The proliferation of Internet and the fact that the majority of the end users are familiar with performance of web applications put enormous pressure on IT departments to improve the user experience inside the corporation. Customers are less likely than in the past to wait for transaction to complete, even a more complex transaction. They demand strict SLA compliance which usually includes clauses of expected performance. Unsatisfactory performance thresholds lead to low customer satisfaction and may even cause users to abandon certain applications.

## Project Testing

This type of testing may be described as a welcomed first step towards real automated testing. It is usually departmentally based and targeted at specific application in use at the department level. Each department purchases its own set of tools and licenses, and procures its own infrastructure to conduct tests. One department may have expertise in a specific area, for example, J2EE applications, and another one may have a resident database expert, but the groups do not share knowledge. In most of the cases, the tests are performed by developers who do not have a direct communication channel with business customers and therefore do not have an insight into business objectives and criticality of the applications.

This testing leads to limited results because of:

- No reuse of infrastructure and testing tool licenses
- No reuse people and skills
- Lack of alignment between practices and business goals

## Standardization

This is the phase on the way to a true Performance CoE. When an organization moves to combining hardware assets and software licenses, it is usually because they understand that the cost to build and maintain a production-like environment for performance testing is prohibitive. This cost is even higher when testing labs are duplicated between lines of business. On the other hand, the consequences of failing to have an accurate performance testing environment may be catastrophic.

The Performance CoE can initially serve as a catalyst for consolidating the software and hardware used for performance testing, resulting in cost savings through better asset utilization and lower administration and maintenance costs. Here the CoE transforms itself from a departmentally focused service organization to a cross-line-of-business resource that essentially manages the infrastructure and supports the users.

At this point, the organization is still not recognizing the developing performance engineers' skill sets and sharing knowledge. As project teams and management begin to recognize the advantages of the Performance CoE model, the CoE can gradually expand its services to broader performance issues.

## Centralization

This is the next step towards deeper involvement of performance testing team in various stages of the software development and deployment lifecycle.

Here the Performance CoE team begins to provide actual performance testing services including planning, scripting, and execution of tests. Traditionally, most lines of business are limited in their performance testing knowledge and use of industry best practices and processes, but with the CoE model they have access to the experience and recommendations of performance experts.

This stage, sometimes called *Service Utility* model, is very popular at many customer organizations who claim to have an implemented CoE. CoE establishment gives an opportunity to nurture specialized and rare knowledge of performance engineering. It enables staff to work across multiple lines of business, resolve problems in intersection of integrating technologies and raise the understanding of how architectures work together. All of this is an additional factor in attracting and retaining talents.

## Performance Authority

The highest level of maturity is when the CoE becomes a *Performance Authority*, in which the CoE becomes a routine part of application development, deployment, and operation, contributing to an organizational culture focused on performance and cost efficiency. Under the Performance Authority model, no application reaches the production stage without going through all-encompassing, consistent processes, focused on enhancing performance.

Performance Authority serves the basic principles below – some of which were covered during the earlier stages of CoE development:

- Support

  For its area of focus, a Performance CoE should offer support to the business lines. This may be by providing services or subject matter experts.

- Guidance

  Some typical approaches to providing guidance are standards, methodologies, best practices, tools and knowledge repositories.

- Shared Learning

  To encourage shared learning, provide training and certifications, skill assessments, team building and formalized roles.

- Measurements

  A CoE should be able to demonstrate its value and justify its creation through the use of output metrics. Measurements are the foundation for CIO and business visibility.

- Governance

  Efficiently utilizing resources (money, people, etc.) and realizing all of their potential is an important function of a CoE. This should ensure that organizations invest in the most valuable projects and create economies that scale to match their service offering. In addition, coordination across other corporate interests is needed to enable the CoE to deliver value.

The proper implementation of these principles allows IT organizations to build a more effective CoE, contribute to overall customer satisfaction, and meet or exceed organizational service level agreements (SLAs). This in turn leads to better ROI and wider acceptance of the Performance Authority.

# 2 Build Successful CoE

This chapter describes best practices for planning a Performance Center of Excellence and taking steps towards establishing such an entity inside your current IT organization.

Performance testing is about measuring the performance of the overall architecture under load. The resources that should be applied within the performance testing space and the overall capacity of your organization to perform this type of testing are dependent on the answers to the following questions:

- How often will performance testing occur?
- How rigorous should the performance testing be?
- How much risk and/or cost is incurred when performance issues occur in production?
- How many and how often are technical resources assigned to performance testing?

If your answers to these questions show that your application team or company is making serious ongoing efforts to carry out performance testing, and this effort will or does consume several resources within your organization, you should seriously consider forming a Performance Center of Excellence (PCoE).

As mentioned in the previous chapter, the Performance CoE model allows you to start small, use existing resources, and achieve tangible benefits almost immediately. In time, you can expand to higher levels of maturity and services as you demonstrate valuable business results and achieve wider acceptance among project teams and management.

A PCoE promotes performance testing best practices, defines and implements performance testing processes, rolls these processes out to the operational team, and helps set the strategic direction of the organization.

The PCoE model supports a group of performance specialists and technical performance testing components leveraged across all projects. The performance testing specialists educate and supplement the resources that exist within each project team to ensure consistency. This allows for maximum project penetration with minimum staffing changes.

The PCoE builds a base of shared best management and performance testing practices across projects to more efficiently use resources. PCoE promotes reusability of performance test deliverables such as test plans, test cases, and fault reports. This group maintains a repository of deliverables as they are created for each project and provides them as a framework and samples, for future efforts.

In general, a Performance CoE can help you achieve the following performance objectives:

- Increased confidence of success when rolling out new code, new software and new hardware configurations by allowing you to identify capacity constraints, concurrency problems, and other performance bottlenecks before going to production.
- Improve the performance of existing code through application tuning, reducing overall resource requirements while decreasing overall response times.
- Improve ongoing capacity planning/management by gaining a better understanding of the performance profile of your architecture and its scalability characteristics.

# Two Approaches to Building a CoE

Organizations usually understand the need to establish a Performance CoE based on pressures coming from two directions:

- top-down direction, where the task to build a center of excellence is made obligatory by the management
- bottom-up direction, where the urgency of testing consolidation is raised and communicated by the testing community itself

Our own surveys of selected customers and industry sources indicate that the first type of influence is much more widespread than the second.

## Top-down

In the top-down approach, a company's vice president of application development, or a similar role, mandates the testing group to come up with a proposal to create a process-oriented center for nurturing and concentrating testing resources and knowledge. Such an initiative often surfaces when a number of complaints about poor performance in a mission critical application reach the business managers such as the CIO or even CEO. Sometimes the mandate may be a result of an outside consultant claiming that a CoE may improve customer satisfaction, leverage accumulated knowledge, and provide a better ROI of hardware and software resources. In some cases, the CIO or VP Apps is approached by the performance tool vendor, or returns from an industry expo event equipped with the notion that establishing a center of excellence will greatly benefit the company.

Whatever the trigger, the management should clearly articulate the objectives of the Performance CoE, its initial coverage, and anticipated go-live date. It is recommended to start with defining a mission statement for the initiative. The phrase *mission statement* has suffered serious dishonor over the years, as organizations, often prompted by consultants, have emitted myriad examples lacking any substance. A useful mission statement is one that actually corresponds to what we intend to do and what we indeed do. In our case, the mission of the test team is what the team accomplishes on a daily basis.

# Communication

When establishing Performance Center of Excellence, one of the most important (often overlooked) factors is communication with multiple parties within the organization such as business, finance, and IT.

Keeping stakeholders informed is always advisable but it becomes more critical when building a CoE. We already mentioned the importance of executive sponsorship for the entire duration of activities. This sponsorship should be regularly fed by targeted reports. The reports should include the data that will specifically interest each stakeholder according to their role.

Unfortunately, performance testers are mostly:

- bad at understanding what stakeholders want to know
- bad at editing out information that is not necessary or relevant (more is *not* better)
- bad at translating technical problems into their likely business impacts
- bad at presenting data in a way that is easy to digest

To improve the widespread attitude towards a Performance Center of Excellence and effectively communicate its value, invest a serious effort in crafting information for numerous stakeholders in the company. The following sections present communication “dos” and “don'ts” based on the stakeholder's role:

- Executive sponsor
  - Do
    - report problems that may potentially delay go-live date
    - report issues that can occur after go-live
  - Don’t
    - mention problems that you can deal with on your own
    - list any technical details or tests that were run
- Project Manager
  - Do
    - report risks to the project schedule
    - report risks to the project budget
    - alert about any issues that can prevent business acceptance

- While written reports are crucial, it is also recommended to hold a meeting to discuss completion of the project to make sure that its results are properly understood
- Advertise the value achieved by testing team and always be ready to explain and show off the benefits of running a CoE

# Tool Standardization

As we have shown *earlier* in this document, tools standardization is one of the first steps on the path to a full-blown Performance CoE. At the very basic level, it includes making an inventory of hardware resources and software testing tools in various lines of business to find a common ground for standardization. There are many possible results of this inventory.

You may find that most of the projects use the same testing software because the company purchased the software with a volume discount. While this means that there is accumulated expertise in the tool, you should still verify the tool's ability to support the objectives of all lines of businesses. Make sure that it can provide an integrated set of applications designed to automate key performance optimization processes, such as testing and tuning, bottleneck identification, capacity planning, and diagnostics.

On the other hand, the inventory check may reveal that the company uses two or three different software packages from different vendors, and none of these is clearly superior to the others. In this case, it may make sense to conduct a competitive analysis of the performance testing software on the market and initiate a process of replacement.

Even in cases where a company decides to keep the existing toolset for financial reasons, it is advisable to move licenses to a centrally responsible group. This will optimize utilization and reduce costs.

HP Performance Center software may serve as an ultimate performance software package because of these benefits:

- It generates consistent, measurable, and repeatable load tests from a single point of control
- Allows to replace real users with thousands of virtual users, and it isolates performance bottlenecks across all tiers and layers
- Manages all of your performance testing assets and schedules your tests, all through a single Web interface

Coupled with HP Diagnostics software, it allows for technology-specific products that delve all the way down to the code level to resolve performance issues in J2EE, .NET, Oracle, PeopleSoft, SAP, Siebel, and so on.

HP Performance Center comes with the installation package and license for HP SiteScope, the leading monitoring solution of production and testing environments.

As for the hardware resources, consolidating most or all of the servers, routers, and other network equipment in one place encourages collaboration between previously dispersed teams supporting intensive testing schedules. Instead of each testing group keeping a testing environment for one deployment area of the product landscape, test environments can be rolled out as required based on the overall deployment schedule. This not only leads to efficiencies in hardware utilization, but it also reduces the overall resource cost of supporting testing environments.

In a summary, tool standardization:

- Increases utilization
  - Load testing available to all projects
  - Visibility promotes ubiquitous load testing
  - More flexible access to assets promotes use
- Reduces costs
  - Lower acquisition costs through pooled licenses model
  - Lower maintenance costs via central infrastructure
  - Lower support costs since resources are centralized and support contract covers all instances of the hardware or software
  - Lower staff costs because training can be unified

# 3 Operate CoE Successfully

There following core set of test deliverables is required for any software testing phase:

- performance test strategy
- performance test plan
- test cases
- defect collection
- status reports

This set of deliverables takes the performance testing team through all phases of the process, from planning to testing and finally to defect solving and status reporting. This does not represent a definitive set of test deliverables, but it may help any test organization begin the process of determining an appropriate set of deliverables.

One common misconception is that these must be presented as a set of "physical" documents. There are applications available that achieve the goals of these deliverables without creating a physical document or set of documents. The objective is to capture the required content in a useful and consistent framework, as concisely as possible.

Before we get into details of various testing deliverables, it is worth mentioning the problems commonly encountered by Performance CoE staff when producing these deliverables (see the *Communication* section).

Performance engineers need to know that non-testers have difficulty understanding testing concepts. They typically don't understand the toolset specifics used to accomplish testing. All they know is that they want results – reliable high-quality software that they can sell with confidence. To ensure this result, testers need to know how to communicate with clients, stakeholders, developers and managers who are not fluent in software testing.

These writing tips might help you successfully document the process and communicate with various performance testing parties:

- Determine who is your audience
- Understand what they want to know
- Do not include information that your audience does not need to know

- Get to the point. Make it clear. Don’t dance around issues
- Your audience will probably not read the entire document - highlight the important parts
- Specify clear action items

# Project Plans

## Performance Test Strategy

Probably the first document in a collection of testing documents is the Performance Test Strategy. While usually applicable to big projects in big organizations, this document may prove useful in defining the overall approach to performance testing. This document usually describes the "big picture" activities that performance testing team perform during the test phase that impact schedule, resourcing, and budget. Ideally, the document should answer some of these core questions about the testing cycle:

- Where does the performance test phase occur in relation to other test phases?
- Which environment will be used for performance testing?
- What tools will be used for testing?
- What is the size of the testing team?
- Who will the team need help from?
- What dependencies does this test phase have?
- What are the entry and exit criteria?

## Performance Test Plan

At a minimum, the Performance Test Plan presents the test itself: objectives, scope, approach, assumptions, dependencies, risks and schedule for the appropriate test phase or phases. Many test organizations also use the test plan to describe the software testing phases, testing techniques, testing methods and other general aspects of any testing effort. It is advisable to retain this information in a "Best Practices" repository. This avoids redundant and conflicting information from being presented to the reader and keeps the test plan focused on the task at hand –- planning the testing effort. In most cases, the content of test plan should address the following questions:

- How does the test environment compare to the production environment?
- How should performance test results be interpreted in each of the above environments?

- What transaction volumes should be used?
- What transaction response time (TRT) is considered satisfactory?
- What is the schedule for data creation and overall environment preparation?
- When should they start script development?
- How will the tests be executed?
- How many cycles of test execution are required?
- When and how will analysis and reporting take place?

The following sections provide more details about parts of the test plan.

## Objectives

The objective of the current testing effort needs to be clearly stated and understood by the software testing team and by any other organization involved in the deployment. This should not be a generic and therefore empty statement on testing the "whole application" - unless that is actually the goal. Instead, the primary testing objectives should relate to the purpose of the current release. If this was an ecommerce system, and the purpose of the current release was to provide additional payment functionality, then the objective (sometimes called mission statement) could be:

*To ensure the enhanced payment functionality performs to overall specification and to verify any existing functionality deemed to be in scope.*

The test objective describes the "why" of the testing effort. The details of the "what" are described in the scope portion of the test plan. Once again, any general testing objectives should be documented in the "Best Practices" repository. General or common objectives for any testing effort could include expanding the test case regression suite, documenting new requirements, automating test cases, and updating existing test cases.

## Scope

In the Objectives section above, we mentioned that generic clauses of testing the whole application typically do not have practical substance. A practical testing scope should include a list of the items that are meant to be tested (in scope), and those that are not supposed to be tested (out of scope).

- In scope

  These are the components of the system that should be tested. This can be in the form of an itemized list of those "in scope": requirements, functional areas, sub-systems, business functions or any aspect of the system that clearly delineates the scope to the testing organization and any other organization involved in the deployment. The "What is to be tested?" question should be answered by this portion of the test plan - the aspects of the system that will be covered by the current testing effort.

- Out of scope

  The components of the system that will not be tested should be clearly defined as being "out of scope." This does not mean that these system components will not be executed or exercised; it just means that test cases will not be included that specifically test these system components. The "What is NOT to be tested?" question should be answered by this portion of the test plan. Often neglected, this part of the test plan begins to deal with the risk-based scheduling that all test organizations must address: What parts of the system can I afford not to test? The testing approach section of the test plan should address that question.

## Approach

This section defines the testing activities that will be applied against the application for the current testing phase. This addresses how testing will be accomplished in relation to the "in scope" aspects of the system and any mitigating factors that may reduce the risk of leaving aspects of the system out of scope.

The approach should be viewed as a to-do list that will be fully detailed in the test schedule. The approach should clearly state which aspects of the system are to be tested and how: environment testing, interface testing, parallel testing, regression testing, longevity testing, stress and load testing, and any other testing approach that is applicable to the current testing effort. The justification for using any given set of approaches should be described, usually from the risk perspective.

## Assumptions

Assumptions are facts, statements and/or expectations of other teams that the test team believes to be true. Document the assumptions specific to each testing phase. These are the assumptions upon which the test approach was based. If any of the assumptions prove not to be true, there may be a negative impact on the testing activities. In any environment, there is a common set of assumptions that apply to any given release. These common assumptions should be documented in the "Best Practices" repository. This assumption document should contain only assumptions unique to the current testing effort and perhaps those common assumptions critical to the current situation should be documented.

## Dependencies

Dependencies are events or milestones that must be completed in order to proceed within any given testing activity. These are the dependencies that will be presented in the test schedule. In this section, list the events or milestones that are deemed critical to the testing effort, and any potential impact or risks to the testing schedule.

## Risks

Risks are factors that could negatively impact the testing effort. Create an itemized list of risks and describe their potential impact on the testing effort. Risks that have been itemized in the project plan need not be repeated here unless the impact to the testing effort has not already been clearly stated.

## Schedule

The test schedule defines when and by whom testing activities will be performed. The information gathered for the body of the test plan is used here in combination with the available resource pool to determine the test schedule. Experience from previous testing efforts along with a deep understanding of the current testing goals will help make the test schedule as accurate as possible. There are several planning and scheduling tools available that make the plan easier to construct and maintain. For example, HP Performance Center includes advanced scheduling capabilities that allow effective sharing of testing resources within a project or between projects.

## Test case

Test cases are the formal implementation of a test case design. The goal of any given test case or set of test cases is to detect defects in the system being tested. A test case should be documented in a manner that is useful for the current test cycle and any future test cycles. As a bare minimum, each test case should contain the author, name, description, step, expected results, and status.

- Test case name

  The name or title should indicate the essence of the test case, including the functional area and purpose of the test. Using a common naming convention encourages reuse and helps prevents duplication. Consider defining naming conventions in the “Best Practices” repository.

- Test case description

  The description should clearly state the sequence of business events to be exercised by the test case. The test case description can apply to one or more test cases; it will often take more than one test case to fully test an area of the application.

- Test case step

  Each test case step should clearly state the navigation, data, and events required to accomplish the step. Using a common descriptive approach encourages conformity and reuse. Keywords offer one of the most effective approaches to test case design and can be applied to both manual and automated test cases.

- Expected results

  The expected results are the expected behavior of the system after any test case step that requires verification or validation. This could include screen pop-ups, data updates, display changes, or any other discernable event or system transaction that is expected to occur when the test case step is executed.

- Status

  This is the operational status of the test case. Is it ready to be executed?

# Artifacts

When performance testing execution is complete, the next step is to analyze the data gathered during the monitoring and diagnostics phases. The results of the tests are represented in various artifacts such as detected defects and status reports.

## Defects

The primary purpose of testing is to detect defects in the application before it is released into production. Furthermore, defects are arguably the only product the testing team produces that is seen by the project team. Document defects in a manner that will be useful in the defect correction process. At minimum, each defect should contain the author, name, description, severity, impacted area and status.

- Defect name

  The name or title should contain the essence of the defect, including the functional area and nature of the defect.

- Defect description

  The description should clearly state what sequence of events leads to the defect. When possible include a screenshot or printout of the error. Make sure to make the business impact clear in non-technical language

- Indicate frequency or likelihood of a problem so that severity and priority can be determined by stakeholders

- How to replicate

  The defect description should provide sufficient detail for the triage team and the developer fixing the defect to duplicate the defect.

- Defect severity

  The severity assigned to a defect is dependent on the phase of testing, impact of the defect on the testing effort, and the risk the defect would present to the business if the defect was rolled-out into production.

- Impacted area

  The Impacted area can be referenced by functional component or functional area of the system. Often both are used.

The defects found in performance testing are different to those a functional tester finds:

- Defects often do not have an explicit requirement
- Defects usually cannot be reproduced by the development team without assistance
- Performance defects are technical in nature and difficult for non-technical managers to understand. Caveat: these are the people who allocate developers to fix these defects.

In most organizations, defect reporting is done using some sort of bug tracking software tool such as HP Application Lifecycle Management (ALM), Quality Center Edition which allows for all the above characteristics to be entered and provides seamless integration with HP ALM, Performance Center Edition.

## Status Reports

A variety of reports are generated by the members of Performance Center of Excellence team. While not every organization produces all types of reports, it is a good idea to document testing results on a regular basis, be it daily or weekly or at the end of the project cycle.

The content of any status report should remain focused on the testing objective, scope and scheduled milestones currently being addressed. It is useful to state each of these at the beginning of each status report and then publish the achievements or goals accomplished during the current reporting period, as well as those that will be accomplished during the next reporting period. Any known risks that will directly impact the testing effort need to be itemized here, especially any "showstoppers" that will prevent any further testing of one or more aspects of the system.

- Reporting period

  This is the period covered in the current status report. Include references to any previous status reports that should be reviewed.

- Current scope

  Specify the components of the system being tested (hardware, software, middleware, etc.), "in scope," and any related components that are not being tested, "out of scope."

- Schedule milestones

  Identify any schedule milestones being worked on during the current reporting period and state their current status. Milestones that were scheduled but not addressed during the current reporting period need to be raised as risks.

- Risks

  Risks are factors that could negatively impact the current testing effort. Provide an itemized list of risks that are currently impacting the testing effort and describe their impact on the testing effort.

It is recommended to keep reports and their raw data accessible to stakeholders – for example, on shared drive:

- This gives a feeling of being hands-free in performing own analysis
- This increases trust because there is nothing to hide
- This encourages involvement as everyone can manipulate the data using familiar tools like Excel

## Test Execution Report

This is a brief, informal text that is created after each test is executed, usually sent as an email, rather than a Word document. It is typically targeted at technical stakeholders and meant to be kept for reference by the CoE team.

This document should present the most important facts about test execution:

- purpose of the test
- configuration changes that were made, if at all
- the business result of the test
- possible root cause of the bottleneck when tuning or investigating a defect
- reference to results files and scenario file

Keeping notes on tests execution and sending them to parties of interest forces tester to do a real analysis of the test results.

## Periodic Report

Even if not required to produce a periodic status report, it is still recommended. In most of the cases, the periodic report – daily, weekly, monthly – is intended for project managers and should ideally cover the following information:

- How schedule tracking is done
- Any issues that may affect the schedule, budget, or project resourcing
- What was done at the high level this reporting period
- What is planned to be done next week
- There is no need to list problems already solved
- Avoid including unnecessary information that makes a report less readable

## Summary Report

This is probably the most important report that summarizes the system performance and stability at the end of the project or test cycle. All stakeholders should get a copy of the report even though for some it may deem too technical or too high level.

The report deals with multiple aspects of the testing cycle such as:

- Which test were executed and whether they passed the success criteria

  In majority of performance testing cycles, the results are more complex than a simple pass or fail status. The main purpose of the tests is to predict behavior after a product "goes live"

- Clearly present the major KPIs

  These should include response times, errors, transaction volumes, system resource utilization etc.

- Changes to scope

  The report should clearly indicate any differences between the scope outlined in the test plan and the final testing activities

- Outstanding problems

  As defects are the only material result of the testing, major defects and their business impact should be listed and classified according to priority if possible

- AUT configuration

  All relevant information such as versions, key settings, file locations etc. should be mentioned in the report to accompany scope definitions

When preparing a Summary Report, follow these guidelines:

- Provide conclusions and analysis assumptions, not just the raw test results
- Introduce the final settings that were used in the test. There is no need to list all steps that led to their tuning
- Mention the origin and status of the test data and other preparation activities without getting into much detail
- Highlight most important response times that prove your conclusion and leave the rest in appendix

Even if you follow the guidelines and include information relevant to specific stakeholder interest, most people will not read past the summary section. Therefore, in addition to a written summary, it is advisable to hold a meeting where most important results and conclusions are communicated to stakeholders in visual form such as PowerPoint. This will help you to distribute a better understanding and acceptance of your performance testing efforts.

# 4 Get People for CoE Success

Probably most important motivation for establishing a Performance Center of Excellence is to leverage accumulated knowledge in load testing, scripting, and expertise in various application technologies and fields to maximize the available human resources.

The PCoE is first of all a group of testing specialists and technical testing components that can provide testing knowledge, technology, methodology and resources to multiple engagements. The testing specialists educate and supplement the testing resources that already exist within each project team. This group usually maintains the deliverables as they are created and provides them as a framework and examples for future efforts. The CoE builds a basis of best testing practices that can be shared. The CoE promotes reusability of test deliverables such as test plans, test cases, test automation, and defects. This process increases the efficiency of the testing effort as reusable components grow.

## Roles

Depending on company size, organization structure, maturity level and even industry the company operates, a Performance Center of Excellence may be divided among several departments and teams. In most cases, the center includes the following core roles:

- CoE manager
- Solution architect
- Testing practice lead
- Test automation engineer
- Business analyst
- Operations manager
- Testing lab specialist

*These are roles and responsibilities, not resources.* Therefore, one resource could fulfill multiple roles within the PCoE structure.

The environment specialist role is included in this list, but in this case we refer only to those activities specifically related to PCoE. We do not include any more general environment activities or roles that are for the benefit of the entire project or organization, such as configuration management.

## CoE manager

The role of CoE manager is to effectively lead the testing teams. To accomplish this, the manager must understand the discipline of testing and how to effectively implement a testing process combined with the traditional leadership roles of a manager. This means that the manager must be qualified to manage and implement or maintain an effective testing process. This involves creating a test infrastructure that supports robust communication and a cost-effective testing framework.

Responsibilities:

- Promote the center of excellence
- Define and promote the importance of the role PCoE plays within the organizational structure
- Define the scope of testing within the context of each release or delivery
- Enforce policies of testing methodology usage
- Assist in establishing and maintaining Service-Level Agreements (SLA)
- Provide SLA knowledge to project teams
- Escalate test issues for resolution
- Implement and continually develop appropriate KPIs that measure the effectiveness of the PCoE
- Plan, deploy and manage the testing effort for any given engagement
- Manage and continually update the testing assets required for meeting the testing mandate:
  - Team members
  - Testing tools
  - Testing process
- Manage CoE human resources and budget
- Provide appropriate training to team members
- Retain skilled performance testing personnel

Managing or leading a testing team is probably one of the most challenging positions in the IT industry. The team is usually understaffed and lacks appropriate tooling and financing. Motivation and retention of key testing personnel under these conditions is critical. There are some tips on how to meet this challenge:

- Update the scope of the test plan to accommodate changing timelines
- Set clear expectations with both the testing team and project management team
- Keep constant lines of communication with the developers and project managers
- Promote the importance and contributions of the testing team whenever relevant
- Ensure that each center of excellence team member has a clearly defined role and a well-defined career path
- Measure and communicate the PCoE return on investment
- Explain testing expenditures in terms of investment (ROI), not cost

## Solution Architect

The role of the testing solution architect is to determine and maintain the test architectural vision for the PCoE. This is probably the most critical role within any center of excellence and should be undertaken by senior testing personnel who have proven experience across several industries in the fields of project management, testing practice leadership, or test automation engineering. The testing solution architect is responsible for selecting and integrating the appropriate set of tools, processes and procedures to ensure overall testing efficiency. The test architect works closely with the CoE manager, testing practice lead, senior test automation engineers, software vendors and quality assurance team to formulate and implement the testing framework.

Responsibilities:

- Be the keeper of the test architectural vision
- Formulate the short and long term test architectural goals
- Ensure that appropriate software testing tools are selected to meet these goals
- Integrate tools, processes and methodologies into a unified structure

- Provide testing frameworks and templates to projects
- Promote the evolution of the PCoE architecture

## Testing Practice Lead

The role of the testing practice lead is to provide the processes, procedures and templates that support effective test design and testing. These should cover test design, test construction, test execution, capturing test results, defect reporting procedures, and test coverage analysis.

Responsibilities:

- Provide testing methodology training
- Work with QA to continuously improve the testing methodology
- Provide guidance on practical application of the methodology
- Provide guidance on organizing the testing project
- Evaluate and recommend approval of test strategies
- Ensure proper methodology coverage of the technical testing processes

## Test Automation Engineer

The role of the test automation engineer is to design, build, test and deploy effective test automation solutions. To fulfill this role, the test engineer applies appropriate automation technologies to meet the short- and long-term goals of the testing organization. The objective is to automate as much of the testing effort as possible using a minimum set of code/scripts. The focus should be on optimizing the testing effort rather than on testing coverage. If one manual test case or manual test preparation process consumes a large amount of test resources, then this manual process should be the first to be automated.

Responsibilities:

- Be skilled in the practical use of testing tools
- Train end users to use testing tools
- Administer the testing tools
- Provide information on upgraded and new testing tools

- Perform various types of testing, such as performance, stress, and capacity

## Business Analyst

Business Analysts (BA), sometimes called Subject Matter Experts (SME), may or may not exist within the PCoE hierarchy. In most cases, Performance Center of Excellence is an internal organization, where the business analysts are probably part of a distinct entity within the company or divided between lines of business. In cases where the business analyst is not part of the PCoE organization, the PCoE must partner closely with the BA/SME to ensure that the business processes to be tested are clearly understood. If the PCoE is an external outsourced organization, then the PCoE should contain BAs who communicate with the client and act as business points of contact within PCoE to ensure that the business objectives are clear and testing is meets the client's requirements.

Responsibilities:

- Possess in-depth knowledge of the realities of a particular industry
- Adapt methodology to include trend-specific processes
- Advise on test process and applicability of test stages to support industry trends

## Operations Manager

The role of the operations manager is to schedule the consumption of PCoE resources, including software, hardware and human resources. The operations manager works closely with the CoE manager, solution architects and lab specialists to ensure that the schedule can handle any planned or unplanned overlaps of resources. This is an implementation of a resource matrix, which must be visible to and understood by all PCoE partners in all lines of business and IT departments. This is especially true when multiple clients or partners share a common resource pool.

Responsibilities:

- Schedule shared project resources - human and environmental
- Manage reusable test business components, such as plans, cases, data, schedules, and scripts
- Publish testing schedules

## Testing Lab Specialist

The role of the testing lab specialist is to ensure that the hardware – and to a certain extent the software – of the PCoE can support all planned testing activities.

Responsibilities:

- Administer hardware, software and networks, including applications and systems
- Perform technical capacity planning
- Support various types of testing, such as performance, stress, and capacity
- Provide the primary interface to technical support groups
- Provide technical consulting to project teams
- Manage reusable technical testing components
- Monitor the technical environment during testing

Remember that many of the PCoE staff will fill a combination of these roles.

# Load Testing Skills

Effective testing delivers a critical business advantage by providing accurate information on the status of a product, or a product enhancement, before it reaches the market or supports an internal business function. This allows organizations to take appropriate remedial actions before implementation, and plan appropriate post-implementation support.

However, there is no effective testing without proper load test design and execution. Performance testing is a specialized skill set. It requires knowledge of the applications, the hardware, and the third party systems. It involves understanding application behavior from architectural standpoint, modeling skills, and a systematic approach. Not all testers possess these qualities, and it takes years to fully develop the proper skill set.

Therefore, one of the most important challenges that a Performance Center of Excellence faces is how to train and retain performance specialists.

The first part of this challenge can be achieved through extensive and ongoing education. Here is the sample learning path for Test Automation Engineer, the core role of each PCoE:

- Basic knowledge
  - Programming: familiarity with at least one object oriented language or C
  - Operating system tuning skills
  - Network analysis and tuning expertise
  - Database analysis and tuning proficiency
  - An architectural view of systems
  - The ability to monitor an infrastructure for bottlenecks
  - The ability to discover and address system bottlenecks independent of the tool
- Load testing abilities
  - The ability to test effectively, independent of the software testing tool
  - The ability to view tests abstractly, and correctly identify poor testing practices for both software testing and non-software testing

# Application Knowledge

In the topics above we explored the issue of responsibilities of each role within a Center of Excellence, starting from the CoE manager all the way down to the performance engineer. There are also specialized load testing skills applicable to members of a CoE.

Beyond basic professionalism, there are other specific categories of knowledge that deal with understanding the Application Under Test (AUT), its technologies, APIs, UI and more:

- Technology

  Since bugs often are technology-dependent—or at least influenced by the system's technology— performance engineers should understand how systems are built, including having a grasp of the underlying programming languages, system architectures, operating system features, networking, presentation layers, database functionality and implementation, and so forth.

- Application domain

  Other bugs originate less from how the system was built, rather from what the system is supposed to do. It is especially difficult to predict what a correct result should look like. Therefore, engineers need to understand the business, technical, or scientific problem that the system solves.

While having technology and application domain knowledge sounds obvious, possessing this knowledge may be the determining factor for a performance validation project's success. For example, if the AUT is mostly a J2EE application running on industry leading application servers, it is imperative to have Java expert on the testing team. Such a person’s knowledge may help to pinpoint Java-specific bottlenecks, tune the system’s lesser-known parameters using a Java profiler and so on.

Another common type of knowledge required virtually on every CoE team is a database performance specialist. Most of the modern applications make extensive use of data stored in relational databases either by direct SQL calls or through application server objects encapsulating the so called data access layer. Identifying database-related problems is one of the hard to find but necessary skills every Center of Excellence should posses. Database administrators, whether for Oracle or IBM DB2 or MS SQL Server, are always in demand, but very few of them also deal with performance issues. If you're lucky enough to have one, we suggest that you try to encourage them to develop an attractive career path that will help retain them on the team.

# 5 Conclusion

Centers of Excellence, sometimes called competency centers, have existed for years but were mainly focused on specific technologies such as J2EE or .Net. Performance Center of Excellence focuses on performance validation and deals with processes and tools to enhance testing effectiveness.

The PCoE model helps to create a group of testing specialists and technical testing components that can be used to leverage testing knowledge, methodology and resources across all projects. The testing specialists educate and supplement the testing resources that exist within each project team. This allows maximum project coverage using minimum human resources and equipment.

Setting up and running a PCoE increases both short- and long-term returns on any investment. With a structure in place, the testing process can now become a mature established process within the organization. The benefits of a mature Performance Center of Excellence include the following:

- It provides a means of leveraging environmental resources, knowledge, and human resources across projects.
- It provides all project teams with access to testing specialists.
- It provides testing consistency across projects.
- It enables reusability of artifacts.
- It increases the quality of end products.
- It allows establishment of key metrics and their analysis for process improvements.
- It adopts risk-based testing, requirements traceability and knowledge management techniques to increase testing organization business value.
- It helps you to meet test schedule and budget commitments.
- It reduces cross-project testing budgets by sharing human and technical resources.
- It makes testing efforts predictable and accurate.

# A PCoE Questionnaire

This appendix lists the questions used to survey customers who are at various stages of building or running a Performance Center of Excellence. The questions may help you perform a self-assessment of your business maturity.

1. Chart the general organization structure and where the performance team fits into the organization.
2. Describe the structure of your performance testing organization. What role (if any) do geography and outsourcing play in this organization?
3. What are this organization's responsibilities?
4. Do you consider your testing organization a Performance CoE?
    a. What is your definition of PCoE?
    b. Who initiated the creation of the CoE?
    c. Do you have projects that are not currently performance tested and why?
    d. Did you use a standardized toolset for load testing, monitoring, analysis, diagnostics etc.? If so, which tools do you use?
    e. Do you have projects that do not use the standard tools?
    f. During which phase of the development life cycle do you start getting involved?
    g. Do you have a common methodology for performance testing? Is there more than one such methodology?
    h. Does the organization consider it mandatory to validate performance prior to releasing products to production?
5. Is there a standard method for working with all LOBs or does it vary according to LOB? If it varies, what is the main difference?
6. What services do you provide to various LOBs (performance validation, performance optimization, capacity planning, and workload analysis)?
7. Is there any formal relationship between PCoE and LOB? Do you have or enforce SLA?
8. When you execute performance validation, are you required to present formal reports or artifacts?

9. What happens after a performance project is delivered? Is there any interaction with development teams?
10. What are the main difficulties you face when running the CoE, from both technical and soft managerial perspective?
11. What are your goals for the upcoming 2-3 years regarding the CoE management and operation?