Performance Engineering
Guidebook for the Systems Development Life Cycle (SDLC)

This e-book introduces the need for and the goals of adding performance engineering activities to the SDLC. This guidebook is to provide the project team with an overview of the activities.
Table of Contents

1. **Introduction** .................................................................................................................................................. 3
   1.1. Review the marketplace (outside CLIENT perspective) ............................................................................. 3
   1.2. Review the typical environment .............................................................................................................. 3
   1.3. Measurement .............................................................................................................................................. 4
   1.4. Why update the SDLC with performance tasks and activities ............................................................. 5

2. **The call to action** ........................................................................................................................................... 5
   2.1. Focus on the user experience .................................................................................................................. 5
   2.2. Benefits of measuring the real user experience ....................................................................................... 6
   2.3. Planning for data conversions ................................................................................................................. 6
   2.4. Position performance tests results to predict production performance ................................................. 6
   2.5. Consolidation of systems ......................................................................................................................... 7
   2.6. Expectations not set for performance goals ............................................................................................. 7
   2.7. Is the new system faster than the old system ........................................................................................... 7

3. **The performance engineering lifecycle** .................................................................................................... 7
   3.1. Review the risk assessment for Phase Zero ............................................................................................. 8
   3.2. Select the performance engineering activities ......................................................................................... 9
   3.3. Define the performance goals ................................................................................................................... 9
   3.4. Design and develop/integrate to achieve the goals ................................................................................. 11
   3.5. Performance testing to verify the goals ................................................................................................... 11
   3.6. Measurement of the production system to track to goals .................................................................. 12
   3.7. Communicate and feedback .................................................................................................................... 13

4. **Review performance engineering in the SDLC** ......................................................................................... 14
   4.1. Snapshot .................................................................................................................................................. 14
   4.2. Phase Zero: risk assessment .................................................................................................................... 14
   4.3. Compliance plan ...................................................................................................................................... 14
   4.4. Application ............................................................................................................................................... 14

5. **Infrastructure** .............................................................................................................................................. 16

6. **Agile** ............................................................................................................................................................ 17

7. **Performance engineering for the project manager** .................................................................................. 18
   7.1. The critical business transactions ........................................................................................................... 18
   7.2. Business volumes .................................................................................................................................... 19
   7.3. Performance risks .................................................................................................................................... 19
   7.4. Performance engineering milestones ..................................................................................................... 20
   7.5. Understand the performance test data needs .......................................................................................... 20
   7.6. Facilitate between SI’s and performance team ....................................................................................... 20
   7.7. Troubleshooting and triage ..................................................................................................................... 20
1. Introduction

This document provides guidance to project teams on implementing performance engineering tasks and project activities. Sections 1 and 2 set the stage for adoption of performance engineering activities. Sections 3 and 4 review the performance engineering lifecycle and the need for a feedback loop across the SDLC. Section 5 addresses the SDLC and performance, and section 6 discusses the agile approach. Section 7 is for the project manager: having the project manager and the business support for performance engineering are critical to success.

Many businesses have an early review phase of the project. It’s used to evaluate the technology and sometimes the business challenges, and to gain commitment before the major project formally starts. Many businesses call this the discovery phase, a risk review phase or a Phase Zero. I like the Phase Zero term and will use it through out. Phase Zero is a performance and technology risk assessment process that will help determine how many of the performance engineering activities are needed for the project based on the project risks. There are six dimensions for the review; these are presented in sections 3 and 5.

1.1. Review the marketplace

1.1.1. On-premises and off-premises systems: The marketplace is using more Software as a Service (SaaS)-based solutions. This is resulting in a more complex enterprise computing environment, where the newer systems are outside the organization; however, they still need access to the existing corporate systems. Using SaaS-type solutions requires a well-defined set of service level agreements (SLA’s) that cover availability, performance, scalability and security. Companies must put monitoring and measurement in place to enforce the SLA’s.

1.1.2. Internet speed and consumer expectations: The users of enterprise applications are greatly influenced by the speed of the web. Many IT teams, and even the business, are slow to realize this and try to minimize the frustration of slow internal systems. Enterprise software users are used to fast retail websites. This is a challenge for many enterprise systems, as they may not have a business need to drive to 3 seconds or less, and enterprise transactions often are far more complex than internet retail transactions. As enterprises continue to roll-out new systems, they must set user expectations appropriately.

1.1.3. Application performance monitoring (APM): The very real trend is to move to Real User Monitoring (RUM). Currently, many companies are using vendors to provide synthetic user monitoring, where automated scripts are created and then executed periodically (every 5 minutes, 10 minutes, etc.) to measure the performance of the system or website. Companies are starting to see the benefits of measuring every user and every transaction. This allows them to proactively see performance issues before the user may feel them.

1.1.4. Visibility: Application performance dashboards showing real-time performance are being used more often for critical business applications. Often times, these dashboards are displayed in an open area (such as a lobby) for anyone in the organization to see.

1.1.5. Internet performance benchmarks: Most of the key internet-based businesses are constantly being benchmarked by different APM vendors. Companies, such as hotels, banks, retail brokerage, etc., are measured and compared. However, for enterprise systems, benchmarks must be determined by the company. There are no common performance benchmarks for enterprise resources planning (ERP), or manufacturing execution systems, call center, distribution centers, etc.

1.2. Review the typical business environment

1.2.1. Complex and highly distributed systems

The typical business application consists of a system of systems. Each one plays a role in the supply chain of the transaction. The people and users of the systems are located across the globe. The global network plays a significant role in performance and response time of the applications, offered more and more in the cloud. For newer companies, we will see a zero premises implementation for systems very soon.
1.2.2. **Wide variety of users for the enterprise**
The wide variety of users can result in a wide variety of system response times, for example for users in a number of call centers across the globe, in plant manufacturing, and in distribution centers, as well as employees using collaboration platforms. Each type of user has a different workflow that will have specific response time goals that support the business process.

1.2.3. **Outsourced development and system integrators (SIs)**
Many large and small business unit IT systems are managed by a business or IT partner. The SI’s will design and deploy the application for either on-premises or an off-premises implementation. The need for defining the performance goals becomes more critical for SaaS-based solutions. You must communicate the performance goals to the SI and the team delivering the solution. This is reinforced by adding performance-related tasks and activities to the SDLC.

1.2.4. **Enterprise monitoring program**
Operations has an enterprise monitoring program in place for business-critical applications, and the operations team is constantly enhancing the program.

1.2.5. **Software vendor packages**
An enterprise uses many different commercial off the shelf (COTS) packages and hosted applications. The business should consider performance and scalability requirements when selecting a new package. The project team should define the response time requirements (e.g., 3 seconds, 10 seconds, etc.) of the critical business transactions, messaging throughput (orders per second), and data migration throughput. A performance and technology risk assessment is part of the SDLC process that reviews all components of the solution. A performance monitoring strategy is critical to measure the production transactions and to hold the vendor accountable to meet the SLAs.

1.2.6. **Consolidation**
Consolidation is occurring and will occur across many enterprises. For example, the consolidation of systems is taking place for distribution centers (DCs), moving the workload of multiple DC’s onto one system or platform. In this example, how will the project team determine that the system can process the new workload and still achieve the performance goals and SLAs? Consolidation projects require a clear understanding of the new additive workload, the capacity plan for the underlying computer system, and a robust performance monitoring strategy.

1.3. **Measurement**

1.3.1. **What is measured?**
Ultimately, the throughput of the business process is measured. Each process needs a set of key business metrics for performance; for example, the number of shipments processed during the day, or for the peak hour. The response time of the critical user transactions are measured. The response time of the system will impact the business process throughput. If the order search time doubles, for example from 5 to 10 seconds, the business process may slow down. What level of information does the operations team need? They may only need to know something is slowing down. If the operations team is tracking dozens or hundreds of enterprise applications, how can you help them distinguish the signal from the noise?

1.3.2. **How do we measure it?**
The critical business transactions can be measured using a number of tools and techniques such as operational reports, IT monitoring tools for the real user experience, messaging infrastructure, etc. A critical new activity in the SDLC is to define the monitoring strategy for each application. How do you make the most of a tool such as Dynatrace? One tool may not solve all your monitoring challenges, so you need to define a roadmap and understand the gaps as well.

1.3.3. **Who needs the information?**
Each project must identify the key users of the system, as part of the monitoring strategy, as well as define the proper dashboards and reports. It must also identify the business unit executives, the operational team, and the implementation team for a multiphase roll-out.
1.3.4. **What do we compare it to?**
The key metrics must be trended over time. These are required to determine if a new release causes performance issues and include: the growth of the user population, the response of key online transactions, the number of inbound and outbound messages, etc. What is the important time period for the business? Is there a daily or weekly peak to compare? If the project is a new application or system, the new system will be compared to the current production system. The project must address the question: Will the new system be faster than the current one?

1.3.5. **Forecasting and trending**
As part of the ongoing operations of the application, the business and IT owners must trend the workload on the system. You should measure and trend the user behavior hourly, daily and weekly and watch the trends for workload peaks. Does the user behavior follow a steady pattern or is it more spike-like? You can trend the volume of messages that enter the system and the number that leave the system, again, hourly, daily, weekly.

Forecasting is about the future. How will the business volumes grow over time, with the workload change overtime?

1.4. **Why Update the SDLC with performance tasks and activities**

1.4.1. **Position projects for success**
A key objective is to prevent late-in-the-lifecycle issues related to performance and scalability of the application. We focus on identifying the performance goals of the critical business processes and the workflows within the process. We want to set user expectations early about what is a satisfied online experience. The project team should also highlight critical data conversions or data loading processes that are required for conversion. We want to communicate the performance goals to all members of the project team. The performance testing strategy and critical testing scenarios must be defined early as well, so we can address the needs of the testing environment and the data.

1.4.2. **Introduce a standard approach**
Provide the teams a standard approach for performance goals and a vocabulary for defining the performance goals.

1.4.3. **Communicate performance goal**
A critical activity for the project manager and IT leaders is to communicate the key performance goals to all team members. The typical project involves many different vendors and different groups within the enterprise. The performance goals and the progress of achieving them must be tracked and communicated. Across many different industries, online transactions are measured and compared by third parties.

1.4.4. **Manage risk**
The project team must be aware of the technical and performance risks of the solution. The team should understand what critical components are new or immature, and may present a risk to the overall project. Are there critical components the solution requires? Will the new system present a large new workload pattern to an existing component? There are five key dimensions to consider:

1) User population  
2) Type of application  
3) Underlying technology  
4) Release plan  
5) Overall solution architecture

2. **What is the call to action for your business?**
Does your organization have a history of applications with different types of performance and scalability issues? Do your applications constantly have issues after each release? The goal is to introduce software performance engineering practices in the SDLC to prevent late-in-the-life cycle performance issues.

2.1. **Focus on the user experience**
Is this you? There has been and continues to be user frustration with the application performance, however, there is only anecdotal information on performance. For many systems, it is difficult getting the real facts of performance. This lack of facts results in a major distraction for the business team and the IT team.
There are many different types of users. For example, retail online customers, distribution centers, manufacturing centers, call centers, patients, healthcare providers, etc. Each type of user has its own expectations for system response time. In a distribution center, when the picking ticket is delayed by seconds or minutes, that could delay the shipment of time-critical medical devices or medicine.

In many cases, a new release or system is noticeably slower than the previous one. This will impact the workflow of the business. If the workflow suddenly goes from two minutes to four minutes, it will take twice as long to process the same amount of work.

2.2. **Benefits of measuring the real user experience**

Measurement of the user experience will provide direction and help set priorities for performance improvement. The facts will reduce the distraction factor. The measurement will show the slowest transactions and the conditions when they occur. With the right tools, the root-cause can be identified within minutes, instead of days.

Measurement will provide the details for a new release and will show whether performance changes. It will also provide data to an executive dashboard.

2.3. **Planning for data conversions**

Has this happened to you? A delayed data conversion process can delay the project, and it is often on the critical path. The team must define the window for the data conversion. The initial ROI of the project can be impacted negatively when the conversion runs long or must be re-executed. For instance, the project may require that 500,000 sales orders be converted in a 48-hour window. This key goal will drive performance requirements for the system to process just over 10,000 sales records per hour.

2.4. **Position performance tests results to predict production performance**

Are you here? Performance tests require a significant amount of resources and time to properly design and execute. During previous projects, often times the results from the performance tests provide little value to the business, because they did not match the performance of the production system. There are many reasons this can occur.

2.4.1. **Workload model**: The workload model is incomplete. Key test cases were omitted or missed. The workload model should include the online cases, interfaces or messaging, batch cases. The workload should be rated by risk, complexity and frequency.

2.4.2. **Test cases**: The workload model drives the test cases. The test cases become test scripts. The arrival rates, the transactions mix and the test script data can be problematic.

2.4.3. **Test data**: The data base is too small. For instance, historical information, ratios of orders to line items, order to customers. The data distribution is different than production.

2.4.4. **Testing tool**: A proper test may require more than one testing tool to drive the load, or to properly simulate production volumes. A real user monitoring tool may be needed while the load generating tool is running.

2.4.5. **Environment monitors**: Are you missing resource consumption information during a test? You need a complete picture of resource utilization to gain insight on how the workload and the system interact. Capturing the complete picture will give your business confidence in the results.

2.4.6. **Size of the testing environment**: This is critical miss. You must know the difference between production and test environments of every component in the test. If the environment is half the size of production, then you need to adjust the expectation.

2.4.7. **Application release**: The software vendor or SI is making frequent changes and releases, so the version that was tested is not what is going into production.
2.5. Consolidation of systems

Have you experienced this? There are a number of application consolidation projects underway and being scheduled with the enterprise. These projects are taking the work from many distribution centers, or call centers, or plants, and putting them onto one central system.

The critical factors are many, but for performance, the key question is, “How do you know the central system will support the workload of the consolidated business processes?” Are there overlapping peaks for the different plants/call centers/distribution centers? There have been instances where all the system resources have been consumed in wave two of a six wave implementation plan.

Having to add unplanned and unbudgeted hardware or software will negatively impact the ROI or the payback period of the project.

2.6. Expectations not set for performance goals

The user expectations must be set early in the project. Many times the performance goals have been left to the end of the project. As the people start to use the system, whether in the testing environment, user acceptance testing phase, or day-in-the-life-testing, it is too late to change the performance. In many cases, the location of the user often has been ignored when designing and developing the system. The location is a significant miss in corporate networking environments. The development team has no idea about the bandwidth.

2.7. Is the new system faster than the old system?

I bet you have seen this. A major discussion point for project teams has been the question; “Is the new system faster than the old one?” Very little focus has been given to the question, but it is a key one to address early. We want to enable proper measurement of new systems to help answer this question. Each project must determine if there is business value in measuring the system/application being replaced. If you don’t address this with your users, you will constantly be in a battle with them as they keep telling you the new system is slower.

3. The performance engineering life cycle

This section provides a general overview of the key activities in the performance engineering life cycle and is where you can accelerate your DevOps adoption. Both require solving the same organizational challenges. The life cycle crosses many organizational boundaries within the enterprise. The development team must know how their application is performing in production. The people involved in the development process benefit greatly from a performance feedback loop. The business and IT teams establish the performance goals, the architects and technical team design and build for the goals, the performance team verifies the goals are met, and the operations team monitors and measures the actual results. As a best practice: the information from each step must be captured and communicated to all parties involved. People need to understand how their design and development choices are working in production.

You should notice similarities with performance engineering and the DevOps approach as they both require combined teams from development, operations and testing. This performance life cycle can be compressed or extended based on your SDLC methods. These phases are required whether you are using waterfall, iterative or agile methods.
For projects that are in a Phase Zero, I have developed a three-step process to determine the type of performance engineering tasks and activities required. I have found, and I am sure you have as well, that not every project needs every performance engineering activity. The risk assessment covers six key project dimensions:

1. Understand the risks from the user population.
   a. Who is using the solution?
   b. Where are they located and how mobile are they?
   c. Review the user population, the business growth plans, and volume peaking factors

2. Understand the associated risk with type of application.
   a. What type of application is supporting the business function?
   b. For instance, is this a core messaging framework, ERP module(s), reporting and analysis, business or consumer portal, or a third-party cloud-based application?

3. Understand the associated risk with the technology the application or solution is dependent upon.
   a. Is this a new technology platform or solution for our project?
   b. Will part or all of the solution be hosted externally? What is the risk with vendor, do they have a demonstrated track record?
   c. How will information move across cloud boundaries? What is the mechanism?
   d. Has it been done before?
4. Understand the risks associated with the application release strategy.
   a. Understand the expected frequency of releases. How many components or feature will be part of a release?
   b. Degree of change per release (size of the change)
   c. Added business units at each release

5. Understand the risks associated with the entire solution architecture (logical and physical layers).

6. Understand the risks associated with the team organization and structure.
   a. How many teams from the client are involved and how are they geographically distributed?
   b. How many vendors are involved?
   c. Is there an SI with a cloud application or a cloud vendor? There can be three or more different companies involved.

3.2. **Select the performance engineering activities**

Once the risk assessment has been completed, the performance tasks and activities will be selected for the project. The performance goals include user experience, batch processing, messages, and data conversions.

<table>
<thead>
<tr>
<th>Activity</th>
<th>Selected for project</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Define performance goals (future state) for the key business transactions. Define the business or requirements phase</td>
<td></td>
</tr>
<tr>
<td>2. Measure or define performance goals (current system) for key business transactions</td>
<td></td>
</tr>
<tr>
<td>3. Define performance goals for data conversion. Example: Throughput, convert 1,000 prescriptions per hour</td>
<td></td>
</tr>
<tr>
<td>4. Conduct a proof-of-concept performance validation of vendor solution</td>
<td></td>
</tr>
<tr>
<td>5. Establish benchmark or reference for performance</td>
<td></td>
</tr>
<tr>
<td>6. Define capacity plan and the cost associated with the solution. Calculate the cost per business transaction</td>
<td></td>
</tr>
<tr>
<td>7. Define performance test strategy; don’t forget the test environment</td>
<td></td>
</tr>
<tr>
<td>8. Define application performance monitoring strategy and plan</td>
<td></td>
</tr>
<tr>
<td>9. Execute performance test</td>
<td></td>
</tr>
<tr>
<td>10. Implement application performance monitoring for feedback</td>
<td></td>
</tr>
<tr>
<td>11. Define performance dashboards that are critical during roll-out</td>
<td></td>
</tr>
<tr>
<td>12. Support root-cause analysis, people and tools ready to go</td>
<td></td>
</tr>
</tbody>
</table>

3.3. **Define the performance goals**

Everyone on the project team must be aware of the overall performance goals of the application for:

- Critical business transactions
- Key user transactions
- Batch throughput
- Messaging throughput
- Data conversions
Example for setting the overall performance context for the project:

<table>
<thead>
<tr>
<th><strong>Project Guidelines</strong></th>
<th><strong>Boundary - Satisfied</strong></th>
<th><strong>Comments</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Online performance goals are at month-end peak online user volume: 5,000 concurrent users at 4 transactions per second</td>
<td>5,000 concurrent users 4 transactions per second</td>
<td>Geographical challenges to the WAN in Asia</td>
</tr>
<tr>
<td>Interfaces performance goals are at peak month end volume</td>
<td></td>
<td>Need additional workshop on gathering the peak business volumes</td>
</tr>
<tr>
<td>Batch processing goals are at peak month end volumes</td>
<td>Complete batch cycle in four hours</td>
<td>Must keep the 4-hour window as the business grows. Throughput will need to increase (scalability)</td>
</tr>
<tr>
<td>Business growth 25% year over year</td>
<td></td>
<td>Interfaces and batch processes must be able to scale with the business growth</td>
</tr>
<tr>
<td>Data conversion – critical goal</td>
<td>1,000,000 orders converted in 18 hours. 900 converted orders per minute</td>
<td>Need to double the system for the conversion process, this is a one-time event</td>
</tr>
<tr>
<td>Online user transactions – Small</td>
<td>2 Seconds</td>
<td>Specific item look up, account code, product code, etc.</td>
</tr>
<tr>
<td>Online user transactions – Medium</td>
<td>4 Seconds</td>
<td>Lookup and present customer Auto insurance policy</td>
</tr>
<tr>
<td>Online user transaction – Large</td>
<td>8 Seconds</td>
<td></td>
</tr>
<tr>
<td>Online historical based transactions (same quarter)</td>
<td>4 seconds</td>
<td></td>
</tr>
<tr>
<td>Online historical based transactions (crossing quarters)</td>
<td>6 seconds</td>
<td></td>
</tr>
<tr>
<td>Interfaces out bound – Type 1</td>
<td>5 TPS — 6.5 TPS for growth</td>
<td>Adjust for 25%</td>
</tr>
<tr>
<td>Interfaces out bound – Type 2</td>
<td>10 TPS — 12.5</td>
<td></td>
</tr>
<tr>
<td>Interface inbound – Type 1</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>Interface inbound – Type 2</td>
<td>7 TPS</td>
<td></td>
</tr>
<tr>
<td>Batch processing – Process 1</td>
<td>60 TPS</td>
<td></td>
</tr>
<tr>
<td>Batch processing – Process 2</td>
<td>5 TPS</td>
<td></td>
</tr>
<tr>
<td>Batch processing – Process 3</td>
<td>15 TPS</td>
<td></td>
</tr>
<tr>
<td>User navigation screen to screen, tabs, pop-up windows</td>
<td>&lt;=1</td>
<td>Client desktop/browser Thick clients</td>
</tr>
</tbody>
</table>
3.4. Design and develop/integrate to achieve the goals

The performance goals or requirements are given to the design and development teams. They will provide traceability for how they achieved the performance requirements. The development team will also define how the solution will achieve the scalability goals of the business.

The best practice is to communicate clearly to the system architects and the developers the performance goal of the critical business transactions.

It is critical to success of the enterprise when there are reference architecture and implementation frameworks that are for performance. The enterprise architecture team can help collect this information for the team.

3.5. Performance testing to verify the goals

The performance testing strategy is developed to validate the performance goals and to stress test the system for unexpected workloads.

Methodology is iterative and tactical

A performance testing strategy and execution document will be created. The purpose is to align the performance tests with the critical business transactions with the high risk business transactions. The performance tests should include a mixed workload of user transactions, batch processing, and messages. The strategy should include testing scenarios of:

<table>
<thead>
<tr>
<th>Performance test scenario</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Component</td>
<td>Run tests focused on key technical components under load to measure throughput: interfaces, vendor product, application server, etc.</td>
</tr>
<tr>
<td>Stress</td>
<td>Identify the point at which the system violates its service level objectives. Then once crossed, how much more load can the system handle before it becomes unstable?</td>
</tr>
<tr>
<td>Duration</td>
<td>Run the system for 24/48 hours under average or peak to uncover instability issues.</td>
</tr>
<tr>
<td>Scalability</td>
<td>Validate the scalability of the solution; additional servers, VM's, CPU's, etc. Does the system scale as predicted to meet the business growth plans?</td>
</tr>
<tr>
<td>Distributed</td>
<td>Validate the user experience across geographies.</td>
</tr>
<tr>
<td>Failover/Failback</td>
<td>Validate the Failover/Failback solution under a load and to measure the impact to the system.</td>
</tr>
</tbody>
</table>
3.6. Measurement of the production system to track to goals

The production system performance must be measured to evaluate and compare the actual results to the original goals. The metrics are captured and can be trended over time. For instance, the growth of the user population can be trended weekly, monthly, etc. A key performance-engineering task is to define the monitoring strategy and dashboard requirements for the business team. The critical business transaction list will be the starting point for the dashboards. The business owners are typically interested in:

- **User experience**: response time of the critical transactions, order entry taking 5 seconds
- **Peak user load**: identify the peak number of concurrent users for a quarter
- **Application throughput**: identify the number of orders processed during the 8-hour day, for instance 10,000 orders processed
- **Inbound and outbound messages**: number of messages

<table>
<thead>
<tr>
<th>Real-user test</th>
<th>Measure the Real-user experience under increasing load</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integration</td>
<td>Define a complete end-to-end business process workflow, that crosses multiple systems. This can involve one system sending transactions to another system. How does the flow of one system slowdown a second system? This is becoming more critical in cloud based applications, where multiple components are in the cloud.</td>
</tr>
<tr>
<td>Performance Regression test</td>
<td>Establish a repeatable and automated test to verify the baseline performance of each release. Also, it is a must to enable retesting once a performance defect has been corrected.</td>
</tr>
</tbody>
</table>
The metrics from the dashboards must be available to the application solution architect, the development team, and the performance testing team, essentially your DevOps team. These metrics complete the cycle for defining, designing, testing and finally operational tracking of the performance goals of the application. Each person must be aware of whether the performance goals were met or not.

3.7. Communicate and feedback

The production measurements must be communicated to all parties involved in the application life cycle. Real-time, on-demand dashboards can be created for different audiences, the Business Unit IT team, the business team, and the technical team. Use production measurements to compare the performance of the application across releases. This allows you to determine quickly if the new release is slower than the previous release. I cannot emphasize this enough for release: measure and compare quickly.

Communication is critical during the execution of the performance test, the progress made during each test and the weekly progress. At the end of the performance testing phase, the results must provide the business with the information to determine if the system can go-live. Will the system support the business goals for performance and scalability?

Communicate the importance of software performance across the entire project team:

- Set expectations across teams for performance priority
- Communicate key performance and scalability goals
- Connect performance activities across the project releases
- Provide performance metrics for key business transaction for each release
4. Review performance engineering in the SDLC

4.1. Snapshot of the activities

Performance engineering enhancements to an SDLC

<table>
<thead>
<tr>
<th>Functional</th>
<th>Technical</th>
<th>Verify &amp; Deploy</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Requirements</td>
<td>Technical Requirements</td>
<td>System Test Report</td>
</tr>
<tr>
<td>Initial performance &amp; scalability non-functional requirements (NFRs)</td>
<td>Performance requirements traceability to business &amp; functional requirements</td>
<td>Report of results against targets</td>
</tr>
<tr>
<td>Initial performance volumetrics, usage scenarios and SLAs at business level</td>
<td>Define usage scenarios</td>
<td>Recommendations</td>
</tr>
<tr>
<td>Performance requirements traceability to business requirements</td>
<td>Identify applicable performance POCs</td>
<td>Production Transition Plan</td>
</tr>
<tr>
<td>Updated business volumetrics, usage scenarios and SLAs at functional level</td>
<td>Performance Test Plan</td>
<td>Design and implementation plan for performance monitoring solution. Includes dashboards, standard reports and alerts</td>
</tr>
<tr>
<td>Performance requirements traceability to business requirements</td>
<td>Scope, approach, resources and schedule</td>
<td>Updated capacity plan</td>
</tr>
<tr>
<td>Architecture Design</td>
<td>Test scenarios, response and load targets</td>
<td>Finalized production support flow and RACI</td>
</tr>
<tr>
<td>Initial capacity plan</td>
<td>Tools and environment requirements</td>
<td>Production Readiness Checklist</td>
</tr>
<tr>
<td>Architecture design traceability to performance requirements</td>
<td>Monitoring requirements</td>
<td>Supporting Documents</td>
</tr>
<tr>
<td>Release Test Plan</td>
<td>test data strategy</td>
<td>Performance Quick Reference Guide</td>
</tr>
<tr>
<td>Performance RACI complete</td>
<td>Detailed test case flows with required data inputs and validation criteria</td>
<td>Overview of PE in the lifecycle</td>
</tr>
<tr>
<td>Performance testing approach</td>
<td>Technical, Middleware, Data Migration Design</td>
<td>Guidance for use of template updates</td>
</tr>
<tr>
<td>Environment management plan</td>
<td>How volume, scale, response times, availability and other NFRs are supported</td>
<td>Performance Risk Assessment</td>
</tr>
<tr>
<td>Test data for performance test approach</td>
<td>Define performance monitoring metrics</td>
<td>Updated project plan for risk mitigation</td>
</tr>
<tr>
<td>Functional Phase Exit Review Checklist</td>
<td>Scope infrastructure requirements</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Technical Phase Exit Review Checklist</td>
<td></td>
</tr>
</tbody>
</table>

4.2. Phase Zero: Risk assessment

During Phase Zero, a performance and technology risk assessment will be delivered for the project for the purpose of selecting the appropriate performance engineering activities. The risk review and assessment is led by a senior performance engineer through a series of workshops over a few weeks, to cover the six critical areas outlined in section 3.

4.3. Compliance plan

Many regulated businesses have formal compliance protocols that must be followed. This requires additional activities and documentation to confirm the outcome is repeatable. There are additional deliverables added to the compliance plan for performance goals and work products. The list includes the performance test protocol, performance test scripts and performance test report.

4.4. Application focus

4.4.1. Functional

Business process performance goals: The functional phase establishes the performance goals of the project. The critical business processes require performance goals. Each critical business process needs a throughput goal for each key business transaction within the process. For instance, the process for order management may need to support a workload of 16,000 new orders per 8-hour day. You must define the average day and hour, peak days and hour, and growth factor for the number of orders.

- Average new orders: 16,000/day; 2,000 per hour
- Peak new orders: 48,000 busiest day in the last year, peak hour 10,000
- Growth is 10% per year.
Key business transactions (use case) within the process: each process will have critical business transactions:

- **Online workflows are assigned a performance goal.** For instance, the workflows of new order, update order, order status, customer order history, etc., must be assigned a duration. For instance, the online workflow of update order has a duration of 90 seconds. A workflow has multiple steps. There may be nine steps in the update order workflow.

- **Response time:** The time the system takes to respond to a user request must be defined, for example, the user presses a keyboard or mouse, waits for the system to respond. First, the response time must be defined in terms of: Satisfied, Tolerating, Frustrated and Abandonment. These will be mapped to actual time, 3 seconds, 10 seconds, 30 seconds, based on the project needs.

- **Batch:** Any batch processes are assigned a performance goal.

- **Message:** Inbound and outbound messages are assigned a performance goal.

- **Data:** Data conversions are assigned a performance goal.

- **Conversion processes:** Workflows that support the conversion process must be assigned performance goals.

At the end of the functional phase, all the critical business transactions will be given an initial performance goal and growth rates. These goals will be refined and negotiated during the Technical phase. The feasibility of achieving the goals within the current budget must be assessed.

### 4.4.2. Technical

The technical team must review and accept the performance goals from the functional phase. They must provide guidance and in some cases negotiate with the business team to refine the performance goals. For example; the business goals were a peak of 10,000 orders per hour with a response time of 3 seconds for the 95th percentile. The technical team may determine the capacity plan needs to be increased resulting in more investment in infrastructure, does the business want to increase the budget?

The purpose of the technical phase is to translate IT application requirements and performance goals into detailed design specifications, to build or implement the detailed design, and to test the newly implemented functions of the application.

The technical design activities document the technical application flow, specific file and data structures, data migration, program logic and data migration; the creation of the software design document that meets the requirements.

Implement the technical design, including the system development and system integrations. This is the process of translating a design into the software components and the hardware components.

The technical design team must be able to map the performance goals to the design and technical decisions. For instance, how does their design and code support the peak goals of 10,000 new orders per hour?

During this phase, the performance engineer will define the performance test strategy and the performance monitoring strategy. There are preparation activities that must occur in this phase to make the performance testing environment ready for use.

At the end of the technical phase, the team should be able to demonstrate early performance indicators to show how they achieved the performance goals. The performance testing strategy and plan is developed, and the performance monitoring strategy is defined.

### 4.4.3. Verify and deploy

The purpose of this phase is to perform independent verification of the system and to deploy it in its production environment. During user acceptance testing, the system is tested end-to-end to verify that it satisfies the user functional and non-functional requirements. This includes the performance test plan, the performance test report and the performance test script.
5. Infrastructure

This is the SDLC track that is used for designing and deploying new IT services.

IT services covers a wide range of options, where the service is being used indirectly by the business users. The application teams will use these IT services to deliver the business functionality. In some cases, the IT service may not be required to have a specific performance goal until the application team is using it. When possible for the IT service, it is a best practice to define the average and peak workload expected by each instance of the IT service. The goal is to define a targeted throughput, as in transaction per second the IT service will need to support.

5.1. Specification and design:

5.1.1. APM requirements: What are the requirements for enabling this IT service for supporting end-to-end APM (from user to data source and back)? APM is used for monitoring the end-user experience, discovering if this new IT service has a role in the user transaction(s), and how can we measure it. Some technologies require additional monitoring components or plug-ins to collect metrics.

5.1.2. Response time goals: Depending on the nature of the IT service, there may be a requirement to define a response time for the IT service under a specific workload. For instance, 100 milliseconds for a message IT service. Or a throughput rating, as in 10 messages per second.

5.1.3. Performance usage expectations: Define the workload profile an instance the IT service will support. For instance: user population, data conversion requirements, inbound and outbound messages, or batch processing. It could be a gateway service with the requirement of supporting the needs of 1,000 call center representatives.

5.1.4. Performance monitoring: Depending on the IT service, there can be different types of performance metrics to collect, such as metrics for resource utilization: CPU usage, network usage, etc., and there are workload usage metrics; transaction types and volumes. Define the mechanics of collecting the metrics.

5.2. Verify and deploy:

5.2.1. Performance test strategy and plan: Depending on the IT service, this can be a challenge to define a performance testing strategy. For example, if the IT service is a component of the overall user transaction, then component-level testing can be the strategy. End-to-end performance testing would not be an option in this case. A component test approach can be used to measure the throughput of an instance of the IT service. The strategy should also consider stress testing, scalability testing and long duration testing. In some cases, failover and failback testing under load may be appropriate.

5.2.2. System test plan for performance: This section covers the following items: define the entry and exit criteria of a successful test, environment and data requirements, critical testing scenarios for scripts, and the testing tools and monitoring tools.

5.2.3. System test scripts and reports for performance: The test scripts and results.
6. Agile

The agile approach is being considered for larger and larger software development projects. This market has responded to this with the frameworks like the Scaled Agile Framework (SAFe) and Disciplined Agile Delivery (DAD). The agile projects are moving from department-focus to enterprise-focus and the project teams are getting larger. The non-functional requirements of systems are becoming more important. The end-user experience is critical.

How do we implement performance engineering and performance testing tasks and activities in the agile SDLC? The introduction of performance engineering and performance testing cannot compromise the original intent of agile—faster and partial delivery of software. The goal of agile is to produce frequent working copies of the system, with incomplete features. Features are introduced for each release. Traditional performance testing occurs after all the features have been developed and the application is nearing production, whereas agile requires an iterative performance testing approach. Regardless of methodology, every project or application needs to define the critical business transactions and assign performance goals under a peak workload condition. The performance goals should include online, batch, messages and data conversions.

- **Performance theme for the project** – This is a project-wide message to let everyone know the project has a performance and scalability focus, and the risk associated with the project require PE tasks, activities, and people. This must convey to the architect and the lead technical person, that best practices, tips and techniques for performance and scalability must be used for this project.

- **Performance stories** – The performance tests for the project, the workload models, and the test types. Add performance requirements or performance constraints to the story.

- **Create a performance backlog or scorecard** – The performance tests that need to be executed as functionality is added. This can become your performance burndown report. You need to keep track of your performance debt. There are short-term decisions that push out your performance challenges. Performance is harder to solve the longer you wait.

- **Spike** – How will the performance tests be executed in context of the Release schedule and Sprints? Will there be a focused sprint exclusively for performance tests?

- **Definition of done** – Clearly defined exit criteria for the performance tests. The exit criteria can include the environment configuration; database size, transaction response time, stress test results.

- **New role on the agile team** – A performance engineer for architecture reviews, code reviews, database reviews, and test planning.

- **Plan and schedule iterative performance testing** – Plan for multiple rounds of testing as partial features are completed.

- **The SDLC document for agile** – Has performance activities that cover performance goals, performance testing and performance monitoring of the application.

---

![Functionality completeness and workload models](image)

**Figure 4 - Performance test burndown**
7. Performance engineering for the project manager

The project manager is critical to enabling the performance engineering activities. Many of the performance engineering activities are new to organizations. The project manager (PM) should fully understand the performance activities for the project. The PM must identify who is the performance engineer. A key activity the PM must coordinate is the workshop, “Introduction to performance engineering,” delivered by a senior performance engineer for the project team. The following list contains the key items the PM should monitor:

7.1. The critical business transactions

There should be a list of the critical business transactions and the associated performance goals. A negotiation process is required during the project to gain agreement between the business team and the technical team. Sometimes the initial business goals are not technically feasible within the current project budget. The critical business transactions become test scenarios for the performance test. The chart below outlines the four steps for establishing performance goals. The critical business transactions include online, batch processing, inbound and outbound messages, and data conversions.

Getting to performance goals in four steps. Considering three perspectives.

<table>
<thead>
<tr>
<th>Step 1</th>
<th>Business Perspective</th>
<th>User Perspective</th>
<th>Vendor Perspective</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Response Time Tolerance</td>
<td>Response Time Tolerance</td>
<td>Standard Guideline</td>
</tr>
</tbody>
</table>

**Step 2** Negotiate Response Time Tolerance

**Step 3**

**Functional Negotiated Guideline**
- Object Complexity Levels – Small, Medium and Large
- Peak and Average Response Time Tolerance Range: 0.1-10 Seconds

**Response Time Targets**

**Technical Design Guideline**
- Performance Budgets
- (Application and Infrastructure)

---

Figure 5 – Establishing performance goals
7.2. Business volumes

The business volumes should be defined for each business process, in terms of throughput. For example, the order management process handles 16,000 orders per day. The peak hourly volume must be defined with every metric. Other metrics include:

<table>
<thead>
<tr>
<th>Business metric</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total user population</td>
<td>10,000</td>
</tr>
<tr>
<td>Peak number of concurrent users</td>
<td>2,000</td>
</tr>
<tr>
<td>Number of internal users</td>
<td>50%</td>
</tr>
<tr>
<td>Number of external users</td>
<td>50%</td>
</tr>
<tr>
<td>Business growth forecast</td>
<td>10% yearly</td>
</tr>
<tr>
<td>Peak inbound messages</td>
<td>10 / second</td>
</tr>
<tr>
<td>Peak outbound messages</td>
<td>20 / second</td>
</tr>
<tr>
<td>Batch processing (orders, shipments, etc.)</td>
<td>100,000 shipments</td>
</tr>
<tr>
<td>Historical requirements and retention</td>
<td>5 years</td>
</tr>
<tr>
<td>Geography supported (user distribution)</td>
<td>LATAM, US, Europe</td>
</tr>
<tr>
<td>Database size and growth</td>
<td>500 GB's</td>
</tr>
<tr>
<td>Critical user transactions</td>
<td>Place order – 3 seconds or less</td>
</tr>
</tbody>
</table>

7.3. Performance risks

Tracking the performance technology risks from the Phase Zero assessment, for example:

<table>
<thead>
<tr>
<th>Item</th>
<th>Who and how to mitigate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peak number of users: 50% higher than previous version</td>
<td></td>
</tr>
<tr>
<td>Vendor hosted: Network volume</td>
<td></td>
</tr>
<tr>
<td>Testing infrastructure: Hosted and internal components</td>
<td></td>
</tr>
<tr>
<td>New component: All transactions flow through as yet undefined component</td>
<td></td>
</tr>
<tr>
<td>Remote pricing call: Need real time pricing</td>
<td></td>
</tr>
<tr>
<td>Data conversion: Requirement to convert 500,000 order history in 48 hours to be ready for cut-over.</td>
<td></td>
</tr>
<tr>
<td>Team composition: Do you have access to performance engineer?</td>
<td></td>
</tr>
</tbody>
</table>
7.4. Performance engineering milestones
The PM works with the performance engineer to define the key milestones, such as:

• Initial performance goals defined for the business requirements
• Technical team review and agreement to performance goals
• Critical risks identified
• Performance testing strategy
• Performance testing execution plan
• Monitoring strategy defined
• Performance testing environment implementation
• Data needs defined for performance test
• Identify performance test team

7.5. Understand the performance test data needs
Data preparation for a performance test can be a large effort on its one. The PM should be aware of the amount of data needed and the effort required to set it up. This can come from an existing production system or the data may have to synthesize for a new system. If the data is coming from a current production system, what conversion steps are required? How long do they take and will they impact the performance test schedule?

7.6. Facilitate between SI’s and performance team
The performance goals must be presented to the SI’s team for review and agreement. The performance team should be meeting with the technical team to communicate the performance goals and understand the development teams’ progress towards the goals. The project benefits when the various teams and the performance engineering team are working closely together.

7.7. Troubleshooting and triage
The need for troubleshooting can arise in the development process and the performance testing process. The PM can help define the roles and communicate during the troubleshooting exercise.

In Conclusion
Here are a few key take-a-ways:

• Please use this e-book as a checklist for your development process to include performance activities throughout your lifecycle. Evaluate your current process to see when you are engaged and how you need it to change.

• Create a technical risk assessment process and corresponding checklist in your organization. Understand why the business has selected the vendor in question and put the technical risk in the business context.

• Know that agile methods need to define the performance engineer role and make sure you do not defer performance debt to the end. Performance activities and testing need to be part of your agile process. If you wait until the end, it won’t go well. You should be getting performance measurements along way, from key Sprints.

• Know what you are measuring and how you are measuring. Make use of dashboards to get business buy-in and show the response time of critical business transactions. Plan for using an Application Performance Management tool from the beginning.

• Know your performance testing environment. You should control the environment and truly understand the differences between it and the production environment. Does your performance testing environment indicate production performance? If so, you must know why.

• Review this document with your project managers and enlist them in helping to drive the activities for performance engineering. Focus on establishing the critical business transactions and use the performance Non-Functional Requirements (NFR) approach to make sure everyone has the same expectations for performance goals.
Walter Kuketz is a Vice President of Consulting and performance engineering expert at CGI. Prior to joining CGI through its acquisition of Collaborative Consulting, he served as Collaborative’s Chief Technology Officer (CTO). Walter helped start and differentiate that company with performance engineering services. He works alongside high-profile, Fortune 100 companies to apply technology for maximum business value and advantage. This includes implementing enterprise architecture in large, complex systems while helping to align business needs with IT – ensuring new software is efficient and valuable. Walter works onsite for delivery of software performance engineering advisory services and technology risk assessment of systems which evaluate performance and scalability of new software. He also assesses and advises on how systems will meet specific business goals.

Walter created the initial draft of the Software Performance Engineering Body of Knowledge and leverages it with each client to create a Performance Engineering Center of Excellence. The Body of Knowledge has been incorporated into the Systems Development Life Cycle (SDLC) of large enterprises. He has presented on this topic on numerous occasions to the National Computer Measurement Group, of which he is an active member. He is also a frequent contributor to DZone.com.

About CGI

Founded in 1976, CGI is one of the largest IT and business process services providers in the world, delivering high-quality business consulting, systems integration and managed services. With a deep commitment to providing innovative services and solutions, CGI has an industry-leading track record of delivering 95% of projects on time and within budget, aligning our teams with clients’ business strategies to achieve top-to-bottom line results.