IBM Business Process Manager V8.5 Performance Tuning

IBM Redbooks Solution Guide

This IBM® Redbooks® Solution Guide provides valuable information for IBM Business Process Manager application development architects and technical leaders. It describes a holistic approach to performance issues that range from establishing initial goals and application design through the implementation phase and finally to long-term application maintenance and planning for future capacity. These tasks require a coordinated effort for all of the teams involved in a complex project to ensure that performance objectives can be met. These teams include business owners, infrastructure architects, database administrators, and test leaders. This solution guide serves as a reference to establish these conversations early and to make sure that all critical topics are covered. The typical application development lifecycle for IBM Business Process Manager follows a model described in Figure 1.

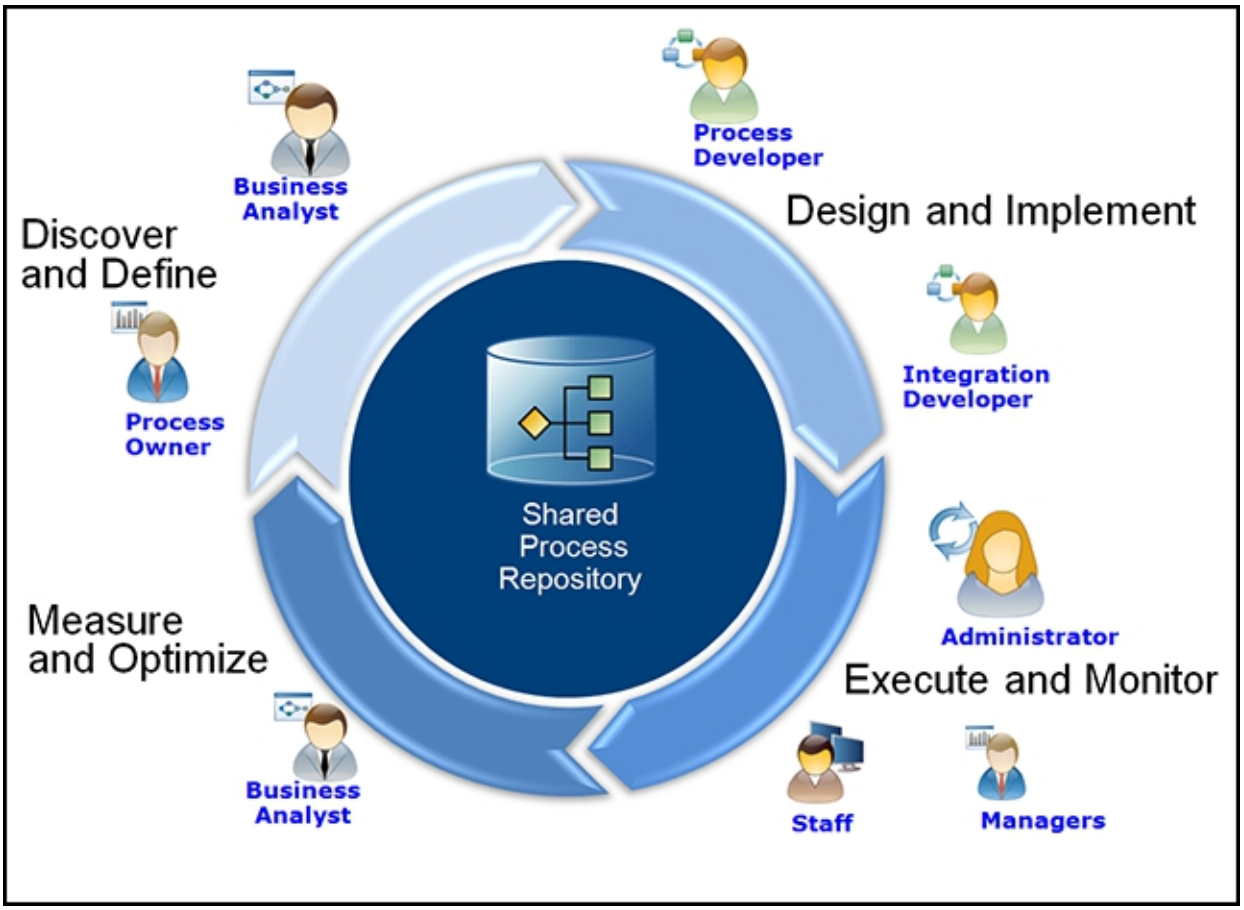

Figure 1. IBM Business Process Manager application development lifecycle

Many of the participants in the overall application development lifecycle play an important role in the success of the project from a performance perspective. For example:

- The performance architect is ultimately responsible for the performance of the solution from end to end and drives performance objectives throughout the application development lifecycle.
- The business analyst is responsible for providing accurate non-functional requirements for the project.
- Application development architects are responsible for designing solutions that are capable of meeting the non-functional requirements.
- Application developers are responsible for following preferred practices and for avoiding anti-patterns.
- The test team lead or manager, together with the performance architect, is responsible for test case design and for translating the non-functional service level agreements (SLAs) that are received from the business analyst into concrete performance objectives.
- Testers and script developers are responsible for building and debugging the measurement harness that simulates the user activity as required for driving the tests.
- Test team members are responsible for executing the measurements and for collecting the resource consumption data that explains the results.
- The infrastructure team, including the system administrator, is responsible for IT monitoring.
- The database administrator is responsible for database tuning and monitoring.

Of course, these roles can vary from one organization to another, and, in some cases, the same person or team of people might fulfill multiple roles.

Did you know?

As a business process development and execution platform, IBM Business Process Manager manages business transactions across a wide variety of industries and functional domains. Even within a single organization, some business processes can constitute straightforward orchestration of a set of web service calls, and other processes coordinate the activities of complex teams of knowledge workers. Clearly, the lifecycle of a process instance can vary dramatically from one business process to another. Similarly, the performance characteristics of one process application can be quite different from those of another. The key to a successful production deployment is to match the actual behavior of the application to the requirements of the business.

With every new IBM Business Process Manager release, performance engineers optimize the code to ensure it can support the most strenuous throughput and response time objectives. However, success is measured in the experience of business users when a new application is deployed into production. To achieve success, the Business Process Manager code must work together with the process application, the services it integrates, and the infrastructure on which it runs. This work begins as soon as the new application is conceived and continues through the performance verification cycle.

One set of challenges common to all system-wide concerns, such as performance, is communication and collaboration across the organizational lines within a business. Separation of responsibility among business leaders, application development teams, infrastructure teams, and database management teams is a common practice and does lead to crisp definition of ownership. However, achieving success on system metrics requires a degree of nimbleness in working across these boundaries. Occasionally, we see progress delayed because of formal structures that result in a period of waiting, for example if an application development team needs to request a configuration change within a database server or if a database administrator needs additional physical resources (processor, memory, or storage) allocated to a database server.

Business value

The most common costs of poor application performance are easy to imagine. Business users might experience slow response times from their user interface. The slow response time leads to dissatisfaction, which can threaten the overall adoption of new applications. Similarly, miscalculation of the performance metrics of a fully-automated application can lead to resource exhaustion (processor or memory), which might threaten the stability of the entire system. When a process application spans multiple business units, the consequences of a resource failure can be substantial.

Although establishing and verifying non-functional requirements early in the development cycle can add to the cost of application development, failing to do so can be even more expensive. Redesigning an application to meet performance objectives after the fact often requires removal or replacement of substantial amounts of code, exposing the project to wasted effort. Building performance objectives into every phase of application development pays off in the long run through reduced risk of system failure and reduced maintenance cost. Also, applications that are well tuned and well measured are less likely to require over provisioning of hardware, which might reduce software licensing costs.

Solution overview and architecture

Now, let us describe IBM Business Process Manager performance measurement and analysis activities by breaking them down among the application development lifecycle from initial requirements gathering through long-term maintenance and capacity planning.

Performance planning and the importance of setting objectives

Just as the functional requirements for a business process management solution originate with the business community who will be using the application and are refined and validated through early discussion and playback, so are performance requirements for the application. Make sure to use the opportunity presented through the regular cadence of playback meetings to gather information about the performance expectations that the business community brings to requirements gathering process. Starting early has the advantage that it forces the Performance Architect to think about the objectives in terms of the end to end business experience from the outset. Often times, it is tempting to think of performance objectives one component at a time. However, it is much easier to build a comprehensive performance plan and gather consensus across all of the teams that will contribute to its execution when the objectives can be described based on end to end scenarios that the Business Analyst and, ultimately, Business Sponsor of the project, understands well and can discuss easily.

As a performance architect and engineer, you can help business users articulate their expectations using metrics that are easy to validate within the performance lab. These expectations allow an initial performance plan for the project to be generated at the same time that the functional test plan is generated. Experience shows that some metrics, such as page load response time for a user interface element, are easy for business users to visualize and discuss. Alternatively, some metrics of extremely high value to the performance analyst, such as total system throughput, are more difficult to discuss. Try to conduct these conversations in a context that is natural to the business. Try to avoid talking about the average number of times a particular API invocation occurs per second. Instead, start with the number of mortgage applications (or whatever business process you are modeling) processed per month, the total number of teams that need to contribute to the processing, and the relative sizes of those teams. After these numbers are established, it is easier to guide the conversation to a discussion of the load that they system might expect during a peak hour.

Focus on establishing performance objectives that accurately reflect the needs of the business. Although it is usually easy to understand the risks that are associated with under estimating the performance objectives, over estimating the objectives can be just as detrimental to the overall success of the project. For example, a response time objective of 1 or 2 seconds for a particular user interface element might be perfectly reasonable. Increasing the objective to 500 milliseconds might seem like an innocuous change and allow for some headroom "just in case." However, in many cases, it is unlikely that this decrease in responsiveness would be of value to the application users, and the engineering costs of achieving this

tighter tolerance might be high. Sometimes, through poor communication, additional headroom accumulates when establishing an objective (with each team building in its own buffer) to the point that the engineering ultimately does not match the business need.

Over time, you should expect to notice patterns that will simplify the process of gathering requirements. For example, perhaps your organization will standardize on 2 second response times (measured at the 90th percentile) for all simple user interface elements, 5 seconds for complex elements, and custom response times only for exceptional operations. At this point, the response time conversation becomes the simpler activity of identifying the complex and exceptional interactions with the system. Similar patterns might emerge for the throughput metrics. Look for them, and use them to your advantage. Perhaps the business users will not have a good idea of the throughput requirements for a totally new application, but they suspect that it might be similar to rates that they are already observing in another application (perhaps that you collaborated on in the past). Beginning the performance requirements conversations early allows those requirements (and the likelihood that they will be met) to be tracked as application developments and playbacks proceed. If performance requirements will force changes in application design or implementation, surfacing those early in the development cycle benefits everybody.

Ultimately, the performance requirements gathering process will result in a performance plan that describes the metrics of importance, the goals or objectives that the application should achieve for each metric and a description of the simulation infrastructure required to evaluate the application. Simulation for some metrics is straightforward. Single user response time measures can be evaluated using a common user interface (like a web browser) by a person with a stopwatch executing the test manually. More extensive load tests required to evaluate throughput objectives for IBM Business Process Manager applications require a large scale simulation capable of generating http traffic. Tools, such as JMeter, IBM Rational® Performance Tester, and LoadRunner, are usually used to fulfill this role. Also, because most IBM Business Process Manager user interface applications communicate with the server via a series of REST interactions, it is possible to construct a stand-alone Java application that simulates server load. In every case, it is important to execute the performance measurements against server hardware that matches that to be used in production environment as closely as possible. Planning early can ensure that hardware is available when needed.

Architecture considerations

Because most business process management solutions are complex systems made up of many parts that must work well together to meet functional and non-functional requirements, an architectural approach to performance measurement and analysis is important. Decomposing the system into its components makes analysis easier simply because small things are easier to understand than large ones. End-to-end measurement is critical to ensure that the system works well as a whole when all of the components are working together. In this section, we look at a few architectural techniques that help.

Start small and scale up. The requirements determination phase might have revealed that your application needs to support thousands of concurrent users starting and completing tens of thousands of business process instances per hour. This does not mean that your performance measurements (or stress and load tests) need to begin with this degree of load. You can learn a lot about your application through repeated measurement, tracing, and analysis of single user scenarios. When these are working as expected, measurements with a small concurrent population will help to verify your load simulation infrastructure and also the scaling capabilities of your new application. Data gathered from measurements with small populations and load will help to predict the results you will see under heavier load and help with verification of the actual measurements. Furthermore, small scale measurements are cheaper and easier to implement, allowing them to be executed earlier and more frequently in the development cycle and giving more time to react in the case that the application needs to be adapted.

Similar scale-up techniques apply to other parts of the infrastructure. When the performance plan calls for measurements against a large, complex IBM WebSphere® ND topology, it is a good idea to start with a smaller system, perhaps a single cluster member with a minimum hardware configuration. As analysis progresses, the system can grow to ensure that assumptions regarding vertical scalability (increasing the number of processing cores per cluster member) and horizontal scalability (increasing the number of cluster members) are met.

Pay attention to the interfaces, especially where the business process management application interacts with external services, such as a web services endpoint. Concurrency or throughput limitations in a service provider frequently limit the performance characteristics of the entire solution. Because web service traffic is relatively easy to simulate (compared with human generated traffic), it is natural to decompose an application at the service implementation boundaries to verify their capacity before re-composing the application for end to end measurements. Of course, when re-composing the application, don't forget to consider the glue that holds it all together--the network connections among the components themselves. A lean user interface that pulls data from a well-tuned backend server might still show poor responsiveness if the network connection between the two has high latency or limited capacity. It is especially important to design the physical topology of the solution to ensure good connection between the process designer and its process center and between a process server and its database server.

Application development

Business process modeling plays an important role in the performance characteristics of a business process management application. Throughout the design and implementation phases of application development, options will arise that can have a significant impact on overall system performance. Chapter 3 of the IBM Redbooks publication, *IBM Business Process Manager v8.5 Performance Tuning and Best Practices*, SG24-8216, describes many common pitfalls and suggests alternatives. This section addresses the topic in a more philosophical manner, looking at some high level design choices and a generalized, data-driven methodology that we find useful during the implementation phase.

IBM Business Process Manager is a flexible business process modeling and execution environment that offers a variety of structures to meet a variety of business needs. At its core, however, all business process management applications manage and persist the state of some business activity. Matching the execution and persistence model to the needs of the business provides an excellent opportunity to optimize for performance. For example, when constructing processes for the BPEL engine, application designers have a choice between short-running processes (microflows), with simple and efficient transactional characteristics, and long-running processes (macroflows), which are capable of coordinating the most complex transactional workloads. Data shows that the simpler state management model used for short-running processes allows execution at much higher throughput. Many processes allow for short-running execution in the common case, but need more complex handling for exceptional cases. So a compound implementation model fits these applications well. All process instances begin with microflow execution and many complete within that microflow. For exception processing, the microflow invokes a macroflow that models the more extensive processing that these instances require.

Similar considerations apply to designing for straight through processing. Computers execute work at a much faster rate than humans. However, business process management usually (and appropriately) begins with an analysis of human workflow. As the process is optimized over time, opportunities arise to automate activities that are carried out by human workers that can be more efficiently handled automatically. This analysis might lead to a shift from a human-centric application that is authored with *Business Process Modeling Notation* (BPMN) to a more automated application that is authored with *Business Process Execution Language* (BPEL) or a hybrid of the two that uses an Advanced Integration service to link the two. Because the engine that executes BPMN processes was optimized for handling human centric applications, it is not nearly as fast at executing fully automated workloads as the BPEL engine is. For this reason, it is best to build Straight Through Processing applications using BPEL to get the maximum resource utilization efficiency.

Applying an iterative, data-driven performance measurement and analysis methodology is a great help in maximizing the efficiency of the performance optimization exercise. Many application development engineers have ideas that can improve the overall efficiency of an application. You can channel these enhancement ideas through a scientifically sound evaluation process. Establish a performance simulation environment early, and extend it as the capabilities of the application expand. This method allows early collection of performance baseline data, which is valuable in tracking application performance as new function is added. Measure the effect of one potential performance enhancement at a time in order to build confidence that the enhancement delivers the expected gain. If it does not, it might be abandoned.

Sometimes, application development teams retain performance "best practices" that incur some development cost but that actually deliver minimal or no performance gain. Use resource monitoring data (such as CPU, memory, and network consumption statistics as well as tracing and thread dumps) to inform your search for new performance enhancements. It might not be worthwhile to pursue an improvement to an area of the application that consumes only 10% of the total resource consumed by the system. Finally, remember Alexander's Law: it is faster not to do something than to do it fast. Sometimes, there is just no need to include the expensive activity in the application at all.

The importance of performance planning

Actual execution of performance measurement and analysis takes time, attention, and physical resources. Sometimes, due to lack of planning ahead of time, a dedicated hardware environment is not available for performance measurement and data collection activities. In these cases, performance activities might be executed on a physical environment that is shared with system integration test or on the production environment as it is being prepared for use. Each case presents challenges to successful execution of the Performance Plan.

By its nature, performance measurement and analysis are disruptive activities, making them unsuitable for sharing a system integration test environment with other activities. One goal of performance measurement is to drive the system under test to complete saturation of physical resources in order to measure maximum sustainable load. Similarly, sensitive response time measurements can be polluted by other test activities running on the machines at the same time. For this reason, it is common to see time-sharing policies used for sharing performance measurement hardware among multiple activities. Time sharing is better than simultaneous use but limits the rate at which progress can be made. Performance analysis is an iterative process, requiring painstaking attention to detail and changing only one parameter at a time. Time sharing reduces the total number of iterations that can be completed within a day.

In some cases, it appears better to execute pre-production performance measurement and analysis activities on the actual production hardware as it is being prepared for use. This might seem like a good idea because it is desirable that the performance test environment match the production environment as closely as possible. Using the same physical machines certainly delivers a good match in this regard. However, after the production environment is deployed to the business, it is no longer available for continued performance measurement activities. Continued measurement as new versions of the application are developed is critical for avoiding release to release regression. Continued measurement is also critical for long term capacity planning activities.

For these reasons, plan early for allocation of a dedicated performance measurement and analysis environment for use during initial application development and longer-term use.

The importance of database design

As a business process state management system, IBM Business Process Manager applications make heavy use of their database, writing state to ensure it is preserved. Furthermore, IBM Business Process Manager allows a flexible infrastructure for describing business data to be stored with the business process and then retrieving those process instances from storage. From a database perspective, this means that IBM Business Process Manager applications are both read-intensive and write-intensive. Close collaboration between the business process management administrator and the database administrator is recommended, as performance analysis must be approached as an end-to-end activity, considering the entire system.

Traditional database analysis using operating system and database tools to identify memory needs, I/O behavior, slow executing SQL statements and similar issues apply to business process management applications and work well. It is critical to ensure that the database server has sufficient physical memory available and that it is backed by a fast disk subsystem (RAID or SAN are common enterprise solutions). Optimization of database indexes is another important activity for the database administrator (DBA) and the business process management administrator to address. IBM Business Process Manager does create a comprehensive set of indexes at install time. However, because there is performance overhead associated with index maintenance, it only creates indexes that are beneficial to the majority of IBM

Business Process Manager applications. Every application will benefit from additional index analysis and this analysis is often critical to get the best performance from a system. Again, an iterative approach works best – use your performance simulation and load test to identify long running SQL statements, use the database query plan advisor to identify indexes that might be beneficial and conduct measurements with and without these new indexes to evaluate their effectiveness, only accepting those that deliver the expected benefit.

Verification and release to production

In addition to the ongoing performance analysis and optimization that occurs during the application development cycle, most enterprises require formal performance verification, including a successful load test as one of the requirements before releasing the application to production. Executing this part of the plan requires simulating full production load (or perhaps greater than full production load) against a fully functional system that is running the application. However, this type of simulation does not mean that the first measurements must be executed at full load, even at the final verification stage. Returning to a small load and monitoring resource consumption as the load increases continues to be good practice and aids in discovery of performance tuning requirements or potential bottlenecks. Final verification should be executed on hardware that matches the production servers as closely as possible. This final verification applies to the servers that host external services and to the servers that host IBM Business Process Manager. In some cases, the final performance verification can call out to the actual production servers hosting web services endpoints, but this is often possible. In either case, the concurrency and throughput characteristics of the endpoints should match those of the production servers.

In addition to validation of the performance characteristics of the business process application, it is worth spending a moment to think about validation of the performance objectives themselves. One common issue observed in performance plans associated with IBM Business Process Manager applications is that the objectives are actually too high. How does this come about? Typically, performance engineers look at the user interface elements that make up a solution and design think times based on a reasonable person completing the form. Perhaps this form can be completed in 30 seconds. So the performance engineer builds a simulation with a think time of 30 seconds per human task completion. Next, the total user population is analyzed, and it is determined that the system needs to support a few thousands of concurrent users. Putting these two pieces together creates a performance plan that requires the system to support the completion of thousands of tasks and many hundreds of business process instances per minute. It is a good idea to validate the throughput implied by the performance objectives against actual business requirements in addition to the number of concurrent users. In some cases, performance verification exercises actually simulate several months' worth of business activity in a period of a few hours. When this is intended, it is a fantastic result; when unintentional, it can lead to some confusion.

Production application management and capacity planning

Good performance practice does not end when the application is released to production. The same resource monitoring tools and skills built up for performance evaluation during application development are extremely useful long term and should be a part of the IT monitoring practice for any enterprise application. Observation of patterns in CPU or memory utilization on IBM Business Process Manager and database servers can help identify approaching capacity issues before they become apparent to the business. In particular, collecting verbose garbage collection statistics (**-verbose:gc** on the Java command line) and using a graphical visualization tool (such as PMAT: https://www.ibm.com/developerworks/community/groups/service/html/communityview?communityUuid=22d56091-3a7b-4497-b36e-634b51838e11) to inspect them is an inexpensive way to monitor the pulse of the application. Many IT departments depend on automated systems of monitoring key resources and sending alerts when thresholds are exceeded.

Long-term performance health of a business process management solution requires a *maintenance plan*. The databases that contain operational data for IBM Business Process Manager are optimized for rapid execution of business process activities, not for long term retention and access of that data. So, it is recommended that a retention period be established for the data associated with business process instances that have completed. The actual amount of time that these completed instances need to be retained will vary from one business to another, but after that period expires, those instances should be removed from the server using the appropriate tools. Business monitoring tools, such as the Performance

Data Warehouse and IBM Business Monitor, are optimized for processing this data for long-term storage and retrieval.

After an application has been available in production for a while, a time will come when capacity planning is needed. Perhaps an application is popular and business users interact with it more frequently or the application is rolled out to a larger community. The experience gained from performance verification exercises and IT monitoring of the application in the production environment is the best input to planning future capacity. Resource consumption models usually follow predictable patterns, so it is often possible to project tomorrow's needs based on a thorough understanding of today's behavior. Also, many enterprises use their load test to ensure enough head room is available to provide a buffer against capital expenditures. Load testing at 150% of projected peak load gives some confidence that the system will be able to handle a growing business and might provide lead time in case additional capacity is required.

Usage scenarios

A wholesale company provides an online parts sales operation where it handles direct orders that are placed by their business partners. Consider a business process application, MyAuditApp, whose goal is to review order data submitted by a business partner for correctness. After the review is complete, the order is sent to another application (perhaps an existing system, not implemented in IBM Business Process Manager) for processing. In this case, the audit process contains two stages. The first stage in the audit is relatively simple, is fully automated, and is required for all orders. The second stage is more complex and requires human oversight, but it is needed only if the first stage of the audit reveals a potential issue.

Performance objectives for the two stages of the application are distinct but are linked. This company knows that it needs to be able to process 30,000 orders per day, distributed throughout a 12-hour period, but with peaks of 4,000 orders in any given hour. These requirements establish a throughput objective for the overall application of about 1.1 process instances per second. To allow some head room, a formal performance objective of 1.5 process instances per second is established for the stage of the application that handles the automated processing.

Because the business knows that 90% of all orders are handled successfully by the automated stage of processing, a formal objective of 0.15 process instances per second is established for the manual stage of the application. In addition, the corporate standard of a 2-second response time for all UI elements (measured at the 90th percentile) is accepted for all stages of the processing, except for initial login and final submission of the audit completion form. These operations are expected to complete in 5 seconds (again, measured at the 90th percentile).

Even though MyAuditApp is a single business process application implemented in IBM Business Process Manager, it clearly has two separate facets. So the performance analysis engineer decides to model the performance simulation as though it were two separate applications. The first half of the simulation is easy to construct. Because all the interactions are automated, perhaps with communication occurring via Java Message Service (JMS) and WebService interactions, software components can be written to drive the application using exactly the same interfaces that will be used in production. Furthermore, because this phase of processing is the most straightforward, the application development team decides to implement it first while requirements determination and initial playbacks of the human interface elements of the application take place. In this way, early performance analysis of the first stage of the application begins before the entire application is fully developed.

Performance simulation for the manual stage of MyAuditApp proceeds differently, because it must account for human interactions, which occur through a web browser and communicate with the IBM Business Process Manager server via REST interactions. Response time evaluation will be executed using the same corporate standard web browser that the business users will use. Evaluation of actual response times when interacting with the application will begin early in application development and will continue as new functionality is added. Any new function that regresses response times beyond acceptable limits will be re-engineered during the next development iteration to ensure that this type of work does not accumulate.

In parallel, work begins to construct the load test simulation that will be used to evaluate throughput metrics for the human-centric stage of the application. An HTTP simulator tool (such as JMeter, LoadRunner, or Rational Performance Tester) is used to record the browser interactions that are required to start and complete a process instance. Then, the tool can simulate many users interacting with the system in the same way. Data pools are used to allow each simulated user to log in to the IBM Business Process Manager server under distinct credentials. The HTTP simulation tool also allows retention of user tokens so that correlation of HTTP session data can be handled by the IBM Business Process Manager server, just as though the load were supplied by a collection of distinct web browsers. After the application has been certified functionally complete, and concurrent with system integration test of the application, load tests begin. The IBM Business Process Manager server is "pre-loaded" with a number of process instances available to work on and the database is backed up so that future measurements can begin from a controlled state. Measurements begin with a single user, ensuring that response times observed within the simulator are consistent with actual browser interactions and that resource consumption on the IBM Business Process Manager server, Database server, LDAP server and all external service provider machines appear reasonable. After this, load is gradually increased until the system successfully processes 0.15 business process instances per second, while maintaining 90th percentile response times faster than 2 seconds. Any issues observed as load increases are resolved before increasing load. In this case, the sizes of thread pools and database connection pools need to be increased from the default in order to allow the desired concurrency. Also, two new database indexes are identified based on the data access patterns of the application.

Performance measurement and analysis continues in this way, systematically increasing the capability of the system under test and of the applied load, until the performance architect is satisfied that the application and the infrastructure upon which it will be executed are capable of achieving the goals.

Integration

Generation and execution of a successful performance plan requires integration across many different types of boundaries, both physical and social. Business Process Integration itself is also a key objective for many IBM Business Process Manager solutions, automating and orchestrating activities across a wide array of software systems connected via REST, JDBC, LDAP, and web service protocols. The software solution’s end-to-end performance will be limited by the slowest element within the execution chain. Similarly, effective communication across all of the teams that contribute to the application--from the business community through the application development team to the infrastructure and administration teams--can play a significant role in the success of the project.

Supported platforms

The supported platforms for IBM Business Process Manager V8.5 are:

- IBM AIX® (32- and 64-bit)
- Linux on x86 (32- and 64-bit)
- Linux on zSeries® (32- and 64-bit)
- Solaris on SPARC (32- and 64-bit)
- Windows (32- and 64-bit)

For detailed system requirements, see "IBM Business Process Manager Advanced detailed system requirements" at:
http://www.ibm.com/support/docview.wss?uid=swg27023005

Ordering information

Ordering information is shown in the following table.

Table 1. Ordering part numbers and feature codes

Program name	PID number	Charge unit description
IBM Business Process Manager Advanced	5725-C94	Processor Value Unit (PVU)
IBM Business Process Manager Standard	5725-C95	PVU
IBM Business Process Manager Express	5725-C96	PVU
IBM Business Process Manager Tools and Add-ons	5725-C97	Authorized User Application Instance

Related information

For more information, see the following documents:

- IBM Redbooks publication *IBM Business Process Manager Version 8.0 Production Topologies*
  http://www.redbooks.ibm.com/abstracts/sg248135.html
- IBM Redbooks publication *IBM Business Process Manager V8.5 Performance Tuning and Best Practices*
  http://www.redbooks.ibm.com/abstracts/sg248216.html
- IBM Redbooks publication *Business Process Management Deployment Guide Using IBM Business Process Manager V8.5*
  http://www.redbooks.ibm.com/abstracts/sg248175.html?Open
- IBM Redbooks publication *Leveraging the IBM BPM Coach Framework in Your Organization*
  http://www.redbooks.ibm.com/abstracts/sg248210.html?Open
- Blog post, "5 Things to Know About Systematically Deploying IBM Business Process Manager"
  https://www.ibm.com/developerworks/community/blogs/5things/entry/5_things_to_know_about_systematically_deploying_ibm_business_process_manager?lang=en
- IBM Offering Information page (announcement letters and sales manuals):
  http://www.ibm.com/common/ssi/index.wss?request_locale=en

  On this page, enter *Business Process Manager*, select the information type, and then click **Search**. On the next page, you can narrow your search results by geography and language.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service. IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you. This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products. This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

This document was created or updated on October 29, 2014.

Send us your comments in one of the following ways:

- Use the online **Contact us** review form found at:
  ibm.com/redbooks
- Send your comments in an e-mail to:
  redbooks@us.ibm.com
- Mail your comments to:
  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400 U.S.A.

This document is available online at http://www.ibm.com/redbooks/abstracts/tips1171.html .

Trademarks