How to Optimize Application Performance in the Cloud: Top 5 Performance Enhancing Recommendations

LOADSPRING

Table of Contents

187 Ballardvale Street, Suite B-210, Wilmington, MA 01887 +1 978.685.9715 information@loadspring.com

Introduction

The following topics will be addressed in detail in the next few pages highlighting our lessons learned while Cloud-enabling Project Management applications since 1999:

- Defining Performance. We'll explain what it means when someone talks about performance, and what it means when a member of the project team calls in and says their tools are not performing as expected. We'll share tips and tricks on how performance in the cloud for Project Management applications is truly defined.
- We'll review root cause analysis techniques for performance challenges.
- We'll make the top 5 performance enhancing recommendations for Project Management in the field.
- We'll explain how to automate performance management, what it means, and how to leverage it to improve end user and customer experience.

Defining a Performance Challenge

1. **Know your Users**

   It is critical to know your user community. More often than not, companies don't know where their users are connecting from to access their Project Management applications in the cloud, or what functions and capabilities they're using within the application. Knowing this is key to understand whether there is a true performance issue, or at least identify some of the variables we're chasing when attempting to identify performance challenges. Organizational training around best practices and hiring the right people for the job is also critical.

   A common issue is users deeming their application as being "down" when it's not, rather users are in some cases attempting to access the application from a network location that has a significant latency issue. It appears that "slow" is the new "down" for users in the field of Project Management.

   Large enterprises and capital project and program initiatives cannot afford to have tools that are either down or seem to be slow, and LoadSpring has been helping companies identify and resolve these issues since 1999.

2. **Know your Applications**

   To know your applications, it's important to understand:

   - Are they sensitive to network latency?
   - Will their users add a high volume of information into the database during peak load?
   - Will the application's performance be compromised if a user deletes a significant amount of data during the middle of the day?

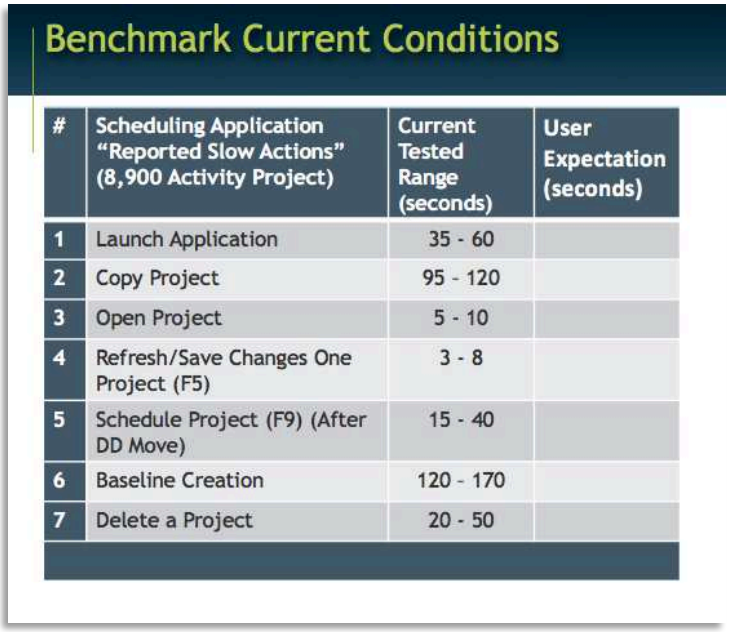
3. **Benchmark Current Conditions**

   Benchmarking current conditions helps truly define a performance balance, especially if the user perceives a challenge. Many times, it is challenging to define what "slow" means. For example, when a user says their application is running slow it's important to dive in and benchmark exactly what they're doing, how long it should take, and then compare that to the company's pre-established benchmark standards to identify the gaps. This is the best way to pursue and address latency issues.

   Benchmarking current conditions is a key process that helps define user performance in the cloud. If someone doesn't know standard capabilities and functions within the application, the key is to find a subject matter expert (SME) to learn the critical function points and how long they should take based on historical data.

   Below are 7 examples of functions that would apply to a scheduling application. When users report slow application performance, it is important to understand performance metrics between the top

seven reported slow actions for this app, and have them walk through a brief test.



#	Scheduling Application "Reported Slow Actions" (8,900 Activity Project)	Current Tested Range (seconds)	User Expectation (seconds)
1	Launch Application	35 - 60	
2	Copy Project	95 - 120	
3	Open Project	5 - 10	
4	Refresh/Save Changes One Project (F5)	3 - 8	
5	Schedule Project (F9) (After DD Move)	15 - 40	
6	Baseline Creation	120 - 170	
7	Delete a Project	20 - 50	

These results will vary substantially depending on the application, but in this example, we tested a scheduling app that is streamed through a Citrix receiver. For example, with this application in this particular scenario launch time tends to be longer as the tested range may be 35 seconds on the low end being that they are connecting to a large project schedule database. On the other hand, if this were a project cost accounting web application, the launch time would probably be between 3-5 seconds. Latencey depends on the functions and the application itself.

4. **Define Expectations**

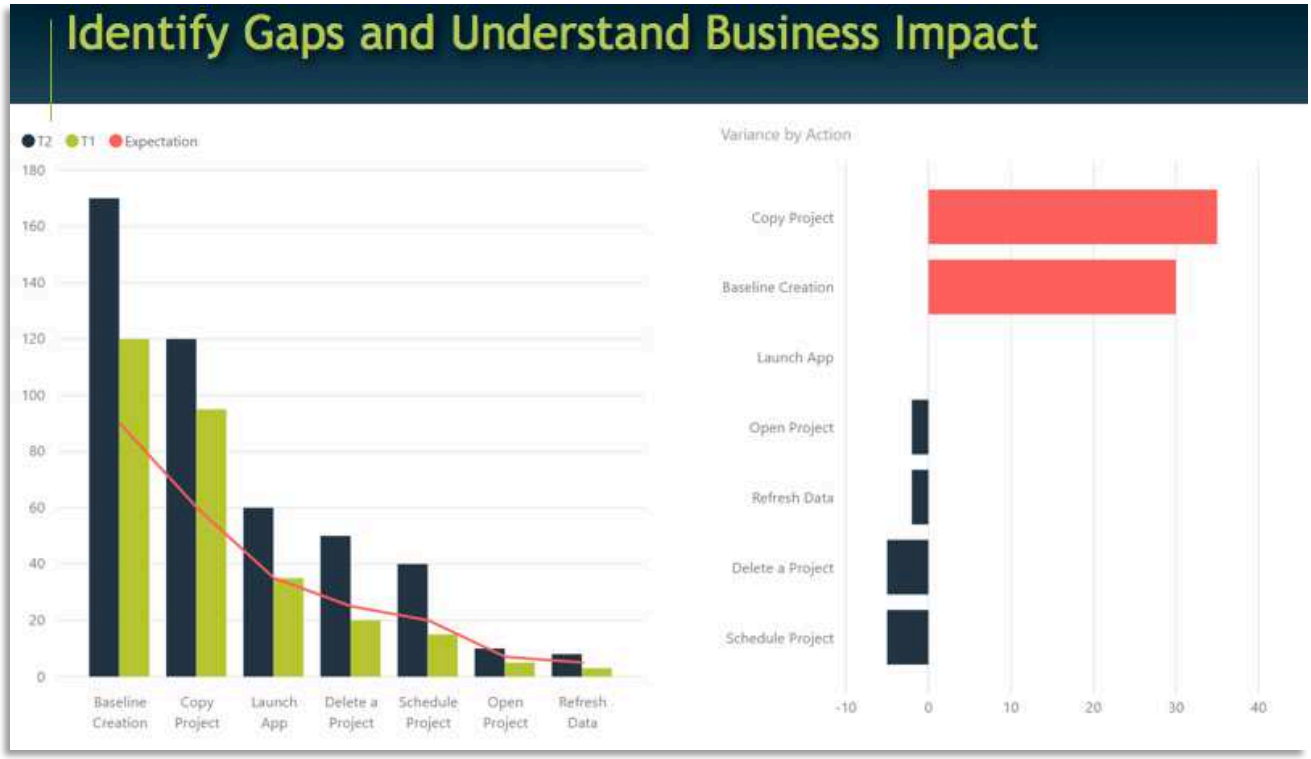
It is critical at the beginning of a project to define the expectation for "good" performance or expected standard performance, being that without defined expectations it's nearly impossible to identify whether the user is experiencing an issue or if it's only a perception challenge.

When utilizing a Cloud provider, it is important for that Cloud provider to walk through and educate the user by defining the expected or normal performance as this lays down solid groundwork to train and educate users. You will also see in the table above that in defining expectations, it is important to define the volume of data as seen with the "8,900 Activity Project" notation in the left column header.

5. **Identify Gaps and Understand Business Impact**

Once the benchmarking of current conditions has been completed, the information should be plotted and shared with the user and/or the organization to allow them to truly understand where their current performance and user experience stands compared to the expectation. This allows companies to identify any gaps that exist between expectated and actual performance.

The bar graph on the left in the image below lists the top seven long lead item capabilities in a scheduling application and their expected duration. The blue bars represent the longest it would take, the green bars represent the least it would take, and the red line represents what the user is actually experiencing. Areas where the red line is significantly above the green bar indicate where it might be taking longer for an end user to perform an action than is ideal.



The idea behind creating a graphical representation such as the one above is to enable an easy visual identification of different areas that need improvement.

The right-hand side of the above graph, shows the user experiencing Copy Project and Baseline Creation issues that are not in line with ideal conditions. Once coming to this conclusion, an analysis should be performed.

The chart below shows what a data profile looks like for a scheduling application. Projects, users and activities as well as other typical data categories that sit within a project scheduling database are detailed in the table below, along with count and volumes recorded. This information is monitored and noted as it can serve as an early indicator of the type of user experience that might be expected when the data falls within certain ranges.

It is imperative that this data is truly understood. To do so, it may be necessary to engage the software vendor to identify where the true data bottlenecks are located. Not every project manager uses project management applications exactly the same, which results in different data profiles. By examining these different data profiles companies can reach a deeper understanding of what the data means in terms of user performance.

Benchmark Current Conditions – Data Profile

Database Environment	Projects	Users	Activities	Average Activities per Project	Resources	Resource Assignments	Resource Assignments per Project	UDF Values	Activity Codes Assigned to Activities
Database 1	1,354	36	2,687,042	1,985	427	595,137	440	2,496,161	12,560,494
Database 2	550	68	23,923	43	3,622	14,114	26	66,429	123,532
Database 3	2,991	59	4,019,879	1,344	2,384	2,053,024	686	9,429,641	44,837,249
Database 4	61,107	374	46,407,752	759	12,126	15,597,861	255	174,708,573	336,956,185
Database 5	59,913	199	38,850,220	648	3,162	2,876,844	48	129,338,307	170,108,266
TOTAL	125,915	736	91,988,816	4,779	21,721	21,136,980	1,455	316,039,111	564,585,726

For instance, the chart above shows Database 4 with 61,107 projects, over 46 million activities embedded within those projects, and close to 337 million activity codes assigned to activities (listed in the far-right column). Some items that stand out when examining this data are the high number of activity codes and user-defined values (UDF values) as seen in the far-right columns, which indicate a less than optimal user experience.

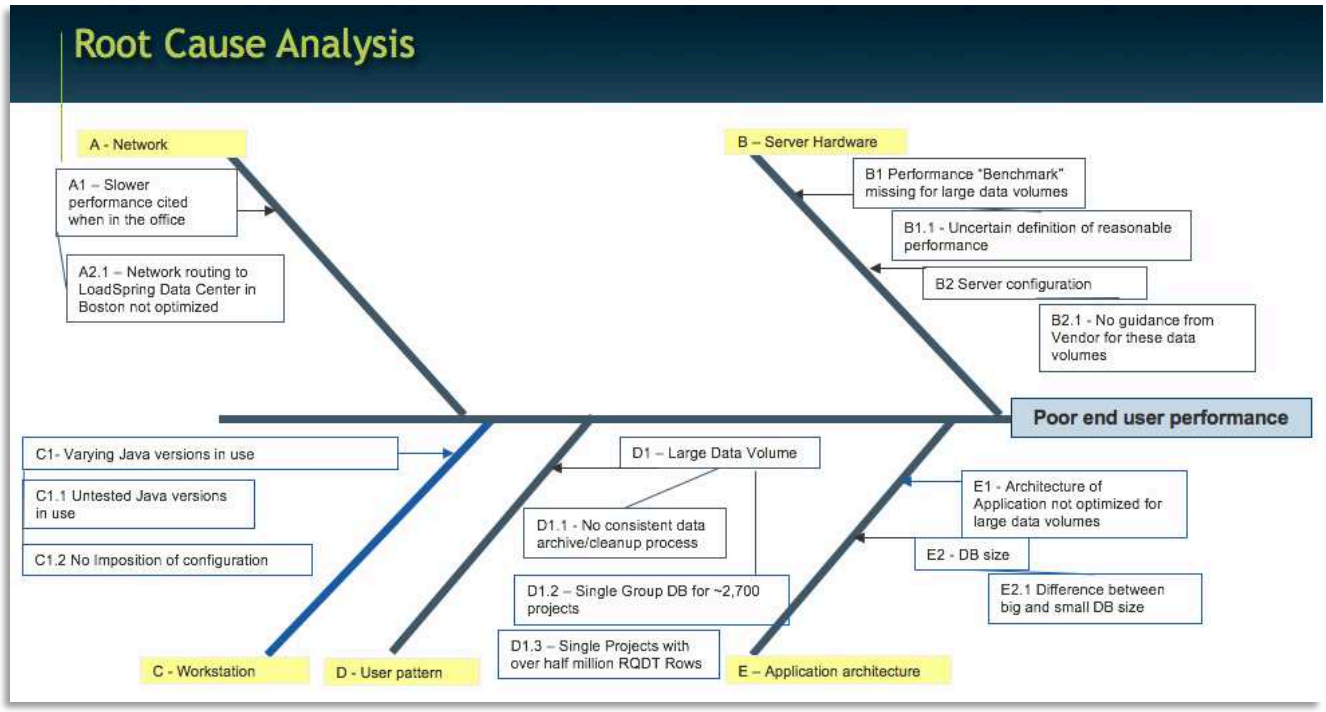
It is important to know your data and your applications in order to identify where bottlenecks begin to show up based on certain thresholds. In this case, Database 4 with more than 336 million activity code assignments has 174 million UDF values, which should ideally be below the 10 million mark for optimal user performance. This analysis is easy to conduct once there is a good foundational knowledge of application architecture constraints and the data profile.

Below is a chart that demonstrates how to identify root cause issues. This process is commonly conducted using a root cause analysis diagram, which is also known as an Ishikawa diagram, cause and effect diagram, or informally known as a fishbone diagram. When using this diagram, it is important to start at the head of the diagram, which in this case is on the far right at “Poor end user performance”, whic is what we are investigating. From there, you would move to the left through different primary categories.

In the example below, there are five different primary categories: Network, Server Hardware, Workstation, User Pattern, and Application Architecture. These categories are then expanded into potential areas of findings with possible pursuits to identify whether they are contributing to a root cause issue.

For instance, you will find under the primary category Network (A) that there are users perceiving a slower performance occurring when in the office. This means that when the user is working outside of the office, they perceive the network and applications to perform better.

Examining network routing to data centers is critical to optimize application performance. For example, if your data center is in Boston, yet your user is in your office in New York, but user traffic is first being routed through a headquarter office located in Boise, Idaho, application performance would not be optimized given the unnecessary stops for the data to go through, and therefore is a key area of opportunity for improvement.



Another example documented in the diagram above of an area with room for improvement is Server Hardware (B). Here, the issue is that there is no benchmark data for large data volumes, which leads to uncertain definition of what truly is reasonable performance. Also important to dive into is server configuration (B2) under Server Hardware (B), and Workstation (C). It's important to have guidance from a vendor to identify whether there are large data volumes, varying Java versions, untested Java versions, or different browser versions in use. It's also important to consider whether all workstations are configured the same.

These are all important considerations that can significantly alter end user performance. Under User Pattern (D), also known as use case, is an example commonly seen of a system with no consistent data archive cleanup process. The lack of data archiving allows the data to continue to grow over time as new projects are added and completed, yet never archived. This creates significant challenges as it hinders performance given the excess of unneeded data.

When examining Application Architecture (E), issues that may be contributing to poor end user performance include the vendor failing to architect the applications or database for a certain set of function capabilities with certain data profiles. Often times, they lack the information from their performance lab pertaining to data profiles that go above and beyond their tested configurations, which ultimately can contribute to poor end user performance.

The root cause analysis diagram above is a very effective tool for identifying bottlenecks in end user performance and for any quality issue in any business. It is most effective to use the diagram in a brainstorming fashion. In this case, it is best practice to gather key team members to brainstorm what is being observed, reported, and potential causes for poor end user performance.

Top 5 Performance Enhancing Recommendations

Recommendation 1: Clean Data

From experience, the number one cause of database issues or performance challenges reported in the field of Project Management is failing to keep data clean. Data management has proven to be one of the best practices any organization can do to optimize performance.

In the project management field, clean data refers to archiving old project data, archiving old resource information, and maintaining database cleanliness for production level data. We see on an ongoing basis that companies keep a high percentage of projects that are untouched for 6 months or more. In fact, watch for 90/10 in your data, which is when a small number of projects contain over half the data in your database. There was one case where we found a single project containing over 10% of the data within a company's database.

In optimizing application performance, it is commonly recommended to utilize database segregation to clean up your database. Although there are a number of ways to break up your database, most companies utilize the following structure when undergoing database segregation.

1. **Active database.** First, there should be a production or active database listing projects that are ongoing or in progress.
2. **Closeout database**. Next, there should be a closeout database for projects that are not quite done as they may still be in litigation. For these projects in progress, access to the project data is still imperative and should go into the closeout database, as they would not fit into the production database criteria.
3. **Archive database.** Finally, companies should use an archive database to house finished projects. For optimal performance, it is essential for projects to be transferred to the archive database upon completion. It is common for end users to lose focus towards the end of a project, which results in many businesses failing to transfer their finished projects to an archive database. This in turn hinders application performance tremendously.

Another easy way to optimize application performance it to utilize custom field cleanup. Many scheduling apps have custom fields that can bloat significantly over time hurting application performance. To avoid this issue, custom field cleanup is essential and should be done thoroughly, all the way down to the fields configured as being selectable by an end user and removing old unused codes. To avoid any issues when undergoing this cleanup, it is important to ensure a database administrator (DBA) monitors the process given that many applications are very sensitive to large deletions. In addition, it is best practice to schedule a database maintenance after a major cleanup, as it allows for users to reclaim whitespace resulting in improved performance. Undergoing this process provides users with a 40 to 50 percent performance improvement for long-lead data items in scheduling applications.

Recommendation 2: Commonize Use Case

Why do so many companies work to commonize use case? A lack of common data usage or data structure rules can create unnecessary data growth negatively impacting application performance in the Cloud. This can also make it extremely difficult to analyze reports across business units when there are different data structures.

Commonizing use case addresses both of these issues. Its goal is to drive global consistency while still granting business units enough freedom to also meet unique reporting and management needs. It is widely recommended to tackle the task of commonizing use case starting with a transformation program focused on what each cross-functional department needs for codes, identifying their project management application update cycles, and putting into place the useful concept of a RACI matrix.

A RACI matrix (Responsible, Accountable, Consulted and Informed) is commonly created to clarify roles and responsibilities in cross-functional or departmental projects and processes. It enables departments to take ownership of use cases and identify how to best commonize them across all their different cross-functional departments. In order for this matrix to be successful, it is important to train the user community on a change program. It is highly recommended for businesses undergoing this process to mandate that new projects immediately follow the new data rules, so that existing projects can go on as normal, while new projects adhere to the new data rules.

Recommendation 3: Pursue Technical Environment Enhancement

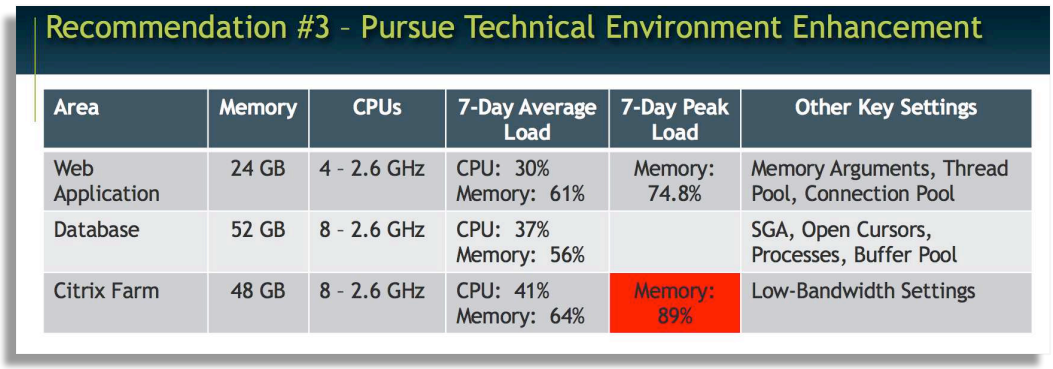
Many cloud providers have a mix of employees. Some may specialize in Cloud Project Management, while others specialize in technology experience and IT, but all employees have the mutual goal of project success with minimal hiccups.

For the technology and IT based employees, they rely on important metrics like:

- How big are the servers?
- How much memory is on the servers?
- How many central processing units (CPUs) are on the servers?
- What is the 7-day average load for those servers?
- What is their peak load?
- Is there anything starting to tap out?

But even with all that knowledge, tough lessons have been learned, such as that adding RAM to the memory does not always result in faster performance. Another lesson learned is that adding more CPUs to the server does not help improve performance, especially if the application is not built to take advantage of it. Ultimately, pursuing technical environment enhancements is not as clear-cut as many may think from an outside perspective. It is best practice to monitor and tune rather than to simply add on hardware.

Many organizations buy millions of dollars worth of hardware hoping to improve performance realizing in many cases that performance did not improve. Companies then find out that performance improvement depended instead on technical enhancements like the “Other Key Settings” in the far-right column in the table below.



Area	Memory	CPUs	7-Day Average Load	7-Day Peak Load	Other Key Settings
Web Application	24 GB	4 – 2.6 GHz	CPU: 30% Memory: 61%	Memory: 74.8%	Memory Arguments, Thread Pool, Connection Pool
Database	52 GB	8 – 2.6 GHz	CPU: 37% Memory: 56%		SGA, Open Cursors, Processes, Buffer Pool
Citrix Farm	48 GB	8 – 2.6 GHz	CPU: 41% Memory: 64%	Memory: 89%	Low-Bandwidth Settings

A perfect example of small, technical enhancements making a big difference is in the case of the Citrix Farm. When it is leveraged for projects in remote locations, it has been found that if the Citrix farm is set up for low bandwidth settings, it optimizes the data traffic and makes it significantly easier to utilize while sitting in a construction trailer in a remote location.

Proactive monitoring is imperative in pursuing technical environment enhancements. As shown in the table above, if a company proactively identifies the memory peak load nearing 100% capacity for any web application, database, or Citrix farm, they are able to quickly respond by adding a new server or more memory to helps avoid issues and enable applications to run at an optimal capacity while boosting functionality.

Recommendation 4: Pursue Network/Application Tuning

Recently a GeekWire notice came out in November of 2017 informing that many internet users were experiencing performance issues with Cloud applications. Given that these issues are so rare and difficult to address, it is best practice for a business to focus instead on what they can control like exploring the user's space and the location of the datacenter to which they are connecting.

In this case it is important to consider:

- "Hop count"
- Are they going through a proxy?
- What is the latency from their location to the data center?
- Speed testing through route tracing
- Network round trip delay

An example of network round trip delay would be when users in New York connect to a data center in Boston, but due to their office networking protocols they must first go through their home office in Boise, then back to the satellite office in New York, then back to Boston. Best practice in cases like this is to partner with a Cloud provider that works with your organization's networking personnel to whitelist your Cloud platform, so your users can go directly from New York to Boston to maximize efficiency and speed. In addition, it is important to explore different constrained network options or low bandwidth scenarios that could significantly impede application function.

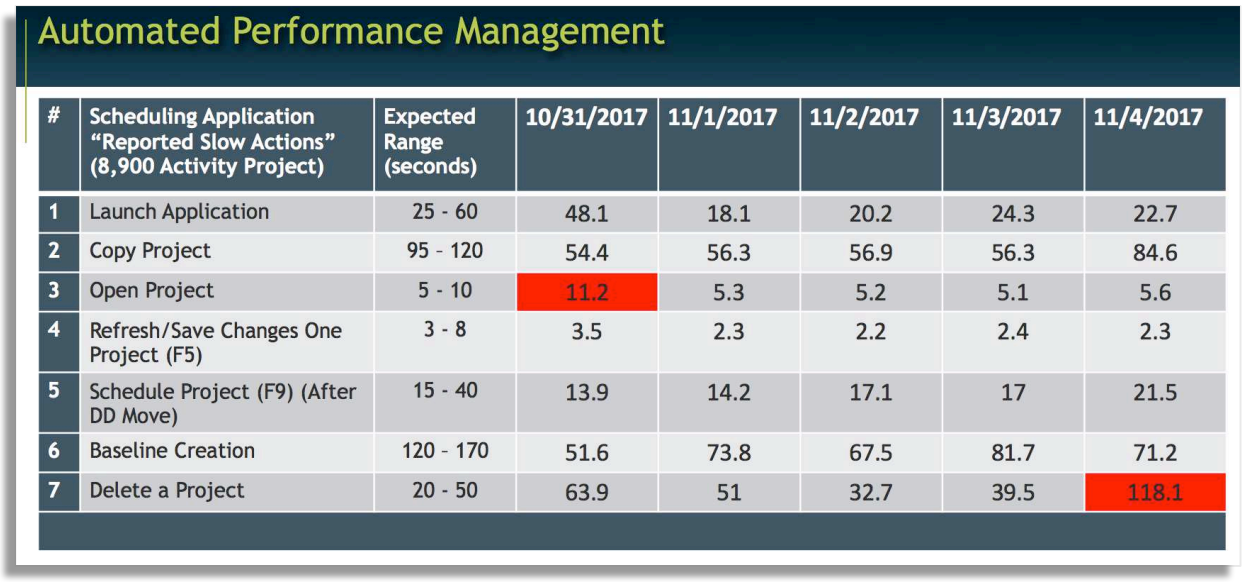
Recommendation 5: Workstation Commonization (Java and Citrix)

Java and Citrix are two common workstation add-ons in the project management space that can cause issues, which means that paying close attention to add-on consistency is vital. Project management teams must equally use the tested configuration for each application for a Citrix Receiver or a Java Runtime Engine (JRE) add-on. In addition, for users running multiple application versions on the same computer, it is important for a Cloud provider to work with your local workstation and IT community to ensure there is no conflict with the different versions. It is also imperative to look at your browser to confirm you are using a supported version, that security settings are identical across all workstations, and to ensure virus scan compatibility.

Organizations sometimes unknowingly have their virus scan software compete with their business use software, which results in a loss of speed and efficiency. In these cases, it is best practice to pause the virus scan software in order to quickly tune it up correctly. This will in turn avoid any further conflict with existing applications.

Automated Performance Management

Many companies are now seeking to automate application optimization in the Cloud. The best way to go about this is to first map out your applications by creating a table that discloses slow functions or actions within your applications, while comparing them to an expected range with day to day testing, as shown in the table below.



#	Scheduling Application "Reported Slow Actions" (8,900 Activity Project)	Expected Range (seconds)	10/31/2017	11/1/2017	11/2/2017	11/3/2017	11/4/2017
1	Launch Application	25 - 60	48.1	18.1	20.2	24.3	22.7
2	Copy Project	95 - 120	54.4	56.3	56.9	56.3	84.6
3	Open Project	5 - 10	11.2	5.3	5.2	5.1	5.6
4	Refresh/Save Changes One Project (F5)	3 - 8	3.5	2.3	2.2	2.4	2.3
5	Schedule Project (F9) (After DD Move)	15 - 40	13.9	14.2	17.1	17	21.5
6	Baseline Creation	120 - 170	51.6	73.8	67.5	81.7	71.2
7	Delete a Project	20 - 50	63.9	51	32.7	39.5	118.1

LoadSpring has found a way to address this as we have now configured day to day testing that behaves just like an end user. Our program opens the same project every morning before end users log in, it processes each of these actions through the front end, and gives LoadSpring the timing of the project applications. From there, LoadSpring receives alerts when something is beyond the threshold – this function is known as LoadSpring's Automated Performance Management (APM).

Functional application performance testing like in LoadSpring's APM service can be performed by companies themselves. In this case, LoadSpring works with companies closely to enable their environment to cater to such testing, but there are a few key items to be considered in pursuing this such as:

- **Automated functional performance testing (executed as a user via the user interface).** In these cases it is important to test and simulate as an end user to improve test accuracy. If companies opt to test certain data bits traveling through the network instead, they will not get an accurate picture of what a typical user is experiencing.
- **Consistent test actions on consistent test data.** It is important to mimic what a normal user would do and to always have consistent test actions on consistent test data. This means that companies can't simply take a 20,000 activity schedule and put it into one data base, then take a 10,000 activity schedule and put it in a different database hoping to compare those results.
  A testing protocol like this does not allow for true side to side, equitable comparison. It is best to ensure companies use the same test project and the same test functions with the only variable being the different databases or different data profiles.

- **Initially profile hourly then execute daily before users log in.** LoadSpring initially profiles hourly, as this shows where the peak load is hour-to-hour during a given workday or work week. Once applications have been optimized testing can be shifted to just once a day or every morning to check if anything is running outside the norm as far as the end-user function and to address any issues before they impact a user.
- **Early warning of user slowdown.** If any issues are spotting during the daily morning checks, LoadSpring sends out early warnings, and can tip off the end users in advance. This prevents slowdown due to the network issues. LoadSpring's automated test platform indicates when network outages are about to occur allowing us to prevent any interruptions for our customers.

The LoadSpring team thanks you for reading this educational piece. As the market leader in providing cloud solutions for Project Management and Engineering Applications since 1999, we have have become experts in maximizing software performance to provide a positive user experience for customers throughtout the lifespan of their projects.

187 Ballardvale Street, Suite B-210, Wilmington, MA 01887 +1 978.685.9715 information@loadspring.com