

Forrester Consulting

HELPING BUSINESS THRIVE ON TECHNOLOGY CHANGE

November 2007

The Business Case For Better Problem Resolution Processes

*A Commissioned Study Conducted By Forrester
Consulting On Behalf Of BMC Software*

FORRESTER®

FORRESTER®

Headquarters

Forrester Research, Inc., 400 Technology Square, Cambridge, MA 02139 USA
Tel: +1 617/613-6000 • Fax: +1 617/613-5000 • www.forrester.com

Table Of Contents

Executive Summary	3
Background And Methodology	3
What Is Application Problem Resolution?	3
Problem Resolution Processes Are Inefficient	4
The Impact Of Inefficient Problem Resolution Processes	5
General IT Costs	6
Development Costs	7
Testing Costs	8
Business Costs	9
Sources Of Problem Resolution Process Inefficiency.....	10
The Business Case For Better Problem Resolution Processes	11
Recommendations	11
Appendix A: Supplemental Material.....	12
Related Forrester Research	12

© 2007, Forrester Research, Inc. All rights reserved. Forrester, Forrester Wave, RoleView, Technographics, and Total Economic Impact are trademarks of Forrester Research, Inc. All other trademarks are the property of their respective companies. Forrester clients may make one attributed copy or slide of each figure contained herein. Additional reproduction is strictly prohibited. For additional reproduction rights and usage information, go to www.forrester.com. Information is based on best available resources. Opinions reflect judgment at the time and are subject to change.

Executive Summary

An efficient application problem resolution process is critical to the success of any IT organization. The longer problems in applications under development or under test go unresolved, the longer it takes to deliver those applications to production. And the longer it takes to resolve problems in production applications, the longer these problems disrupt normal business operations. A study commissioned by BMC Software and conducted by Forrester Consulting indicates that most IT organizations have highly inefficient problem resolution processes. This paper explores the extent of this inefficiency, its impact on the application development organization, and its impact on the business. We also identify the most common root causes of inefficient problem resolution processes and recommend remedial action.

Background And Methodology

To assess the state of application problem resolution practices in enterprise IT and software product companies, BMC Software commissioned Forrester Consulting to conduct an anonymous telephone survey of 150 managers, directors, and vice presidents in charge of application development teams and organizations at US and Canadian companies. The goal of this research was to determine the following:

- How much developer and tester time does application problem resolution consume?
- What makes application problem resolution processes more or less efficient?
- What are the IT, business, and opportunity costs associated with application problem resolution?

This primary research effort, conducted in the summer of 2007, included 100 respondents from enterprises with at least 1,000 employees and at least 50 developers, as well as 50 respondents from software product companies with at least US\$25 million in annual revenues. These respondents were all managers, directors, and executives in charge of application development or maintenance teams or organizations. We screened out respondents who were uninvolved or unfamiliar with the problem resolution processes at their companies. As an incentive to participate, Forrester promised respondents a \$10 Amazon.com gift certificate and a summary of survey results.

What Is Application Problem Resolution?

Application problems are unexpected behavior in software applications. Problems can occur in development, in testing, or after the application has been deployed to a production environment or customer site. Problems can be caused by software defects, but they don't necessarily have to be. Problems can be due to improper configuration, an infrastructure malfunction, or even user error.

The steps that IT organizations and software vendors take to determine the root cause of an error constitute the problem resolution process. Steps in a typical problem resolution process include:

- The problem is discovered by end users, IT operations, testers, or developers, depending largely on the environment in which it occurs.
- Information about the problem is documented and passed along to the parties responsible for high-level diagnostics and assignment of responsibility.

- Alternately, the party that discovered the problem is able to resolve it. For example, IT operations may discover and repair the hardware malfunction responsible for a production problem.
- If the problem appears to be because of a software defect, it may be passed to development. In this case, developers will review the information they've been provided and attempt to recreate the problem, diagnose the root cause, and address the underlying issue.
- If developers are unable to recreate the problem, they will have to gather more information.
- The changes that developers make are passed along to the environment in which the problem was discovered, whether test, production, or a customer site, for deployment.

The problem resolution process involves a large number of stakeholders scattered across the enterprise, all of whom are interested in different domains, familiar with different vocabularies, and accustomed to different tools. This inherently makes the problem resolution process prone to inefficiency, as each handoff between roles represents an opportunity for miscommunication or insufficient communication. When this happens, the result is a bottleneck in the process.

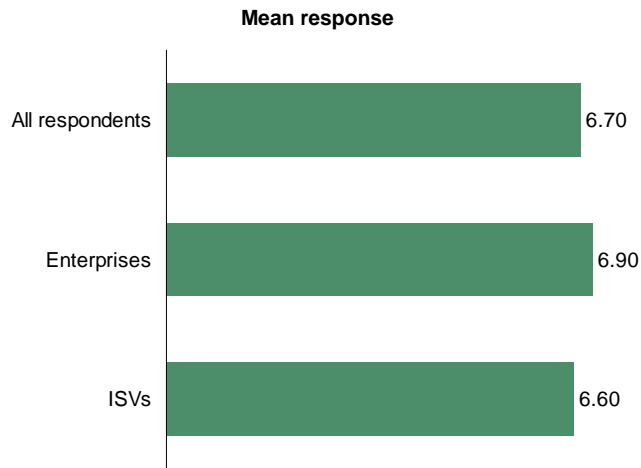
Problem Resolution Processes Are Inefficient

We asked 150 application development managers, directors, and executives about their problem resolution processes. Our findings? For most shops, problem resolution is highly inefficient. Developers dedicate nearly a third of their time to problem resolution, with much of that time spent investigating and attempting to recreate problems rather than actually implementing a fix. And testers spend hours documenting each problem they encounter, in addition to time spent communicating with developers about problems and testing their fixes.

The result is that problems take too long to resolve (see Figure 1). On average, application problems take more than six days to resolve, and 11% of problems take more than 10 days. The criticality of problem resolution depends almost entirely on the nature of the problem and the nature of the application in which it occurs. There are few problems and few applications for which six days is an acceptable amount of time to leave a problem unaddressed.

Figure 1: Problems Take An Average Of Six Days To Resolve

“In number of days, what would you estimate is your development organization's average elapsed time between application problem discovery and application problem resolution?”



Base: All respondents

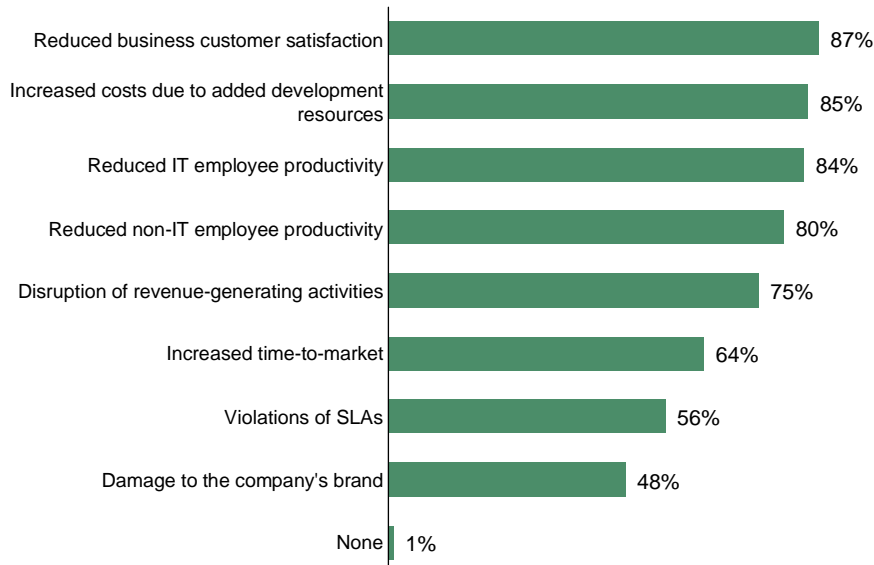
Source: Application Development Problem Resolution phone survey of 150 IT decision-makers responsible for overseeing development teams. A commissioned study conducted by Forrester Consulting on behalf of BMC Software, July 2007

The Impact Of Inefficient Problem Resolution Processes

Inefficient problem resolution has far-reaching effects, both within IT and within the business. The application development managers, directors, and executives we interviewed identified costs that are associated with the time it takes to fix problems in development, test, and production environments or customer sites (see Figure 2).

Figure 2: Slow Problem Resolution Saps Employee Productivity, Angers The Business

“What are the costs associated with the time it takes to diagnose application problems, whether the problem is in development, in testing, in production environments, or at a customer site? (Select all that apply)”



Base: All respondents
(multiple responses accepted)

Source: Application Development Problem Resolution phone survey of 150 IT decision-makers responsible for overseeing development teams. A commissioned study conducted by Forrester Consulting on behalf of BMC Software, July 2007

General IT Costs

Poor problem resolution creates inefficiencies for nearly every role in IT, decreases the overall performance of the IT organization, and prevents IT executives from fulfilling the promises they make to their business partners. The costs of these inefficiencies result in:

- **Increased costs because of the need for more development resources.** Developers and testers who spend their time distracted by problems in production have less time available for the projects that they are working on. It's not only production problems that keep them from delivering new business value; the problems that they encounter before deployment hinder the progress of development, too. As a result, development organizations with slow problem resolution processes require more development resources. Some shops have teams of developers tasked solely with problem resolution; the cost of these teams relates directly to the efficiency of the problem resolution process.
- **Increased time-to-market.** Problem resolution, whether it targets problems in development, testing, or production, pulls developers and testers away from their current workload to get software out the door. The more inefficient the problem resolution process, the more it unnecessarily extends delivery timelines. Because problems are inherently unpredictable, it is nearly impossible to predict the impact they will have on delivery timelines. For this reason, inefficient problem resolution doesn't just result in increased time-to-market; it also results in unpredictable time-to-market.
- **Lower IT employee productivity.** As we mentioned, inefficient problem resolution is especially detrimental to the developer and the tester. But the problem resolution process

involves far more roles, often including help desk representatives, operations managers, security professionals, change advisory board members, and even mid- and upper-level IT executives. Each problem that requires more involvement from IT employees lowers the productivity of the IT organization as a whole.

Development Costs

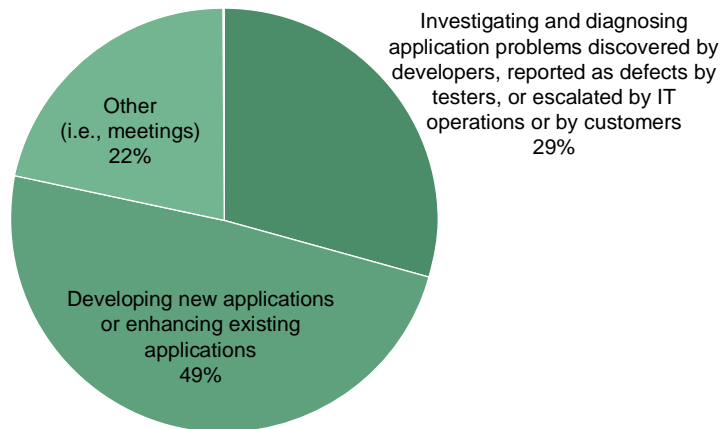
Our interviews with development managers who have visibility into problem resolution processes in their shops indicate that on average developers spend an astonishing 29% of their time on problem resolution (see Figure 3). This includes time spent on:

- **Problems developers discover themselves.** Problems don't just occur when coding is complete. Developers frequently encounter unexpected behavior in applications as they are coding. For example, functionality that is already built may stop working for unknown reasons. At this point, developers must determine whether code changes they've implemented have had an unintended impact or if there are problems with the infrastructure that they are employing.
- **Problems testers discover.** Whether working in parallel with development or performing final independent validation and verification testing, testers inevitably discover problems that developers will be asked to resolve. These may include problems with any of the areas testers address, from functionality to security to performance to usability.
- **Problems customers or operational professionals discover.** The type of problem that IT professionals are most familiar with, and that they often focus on to the exclusion of problems in development and test, is the problem encountered in deployed or shipped applications.

In all of these situations, developers are responsible for recreating, diagnosing, and resolving the problems in question. Sometimes these problems are rooted in code, but often the problem resides in aspects of the development, test, or production environment that the developer has little to no control over — for example, the server, the network, or even a third-party component.

Figure 3: Developers Spend Almost A Third Of Their Time On Problem Resolution

“What percentage of their total time do developers spend investigating and resolving application problems that are discovered by developers, reported as defects by testers, or escalated by IT operations or by customers?”



Base: All respondents

Source: Application Development Problem Resolution phone survey of 150 IT decision-makers responsible for overseeing development teams. A commissioned study conducted by Forrester Consulting on behalf of BMC Software, July 2007

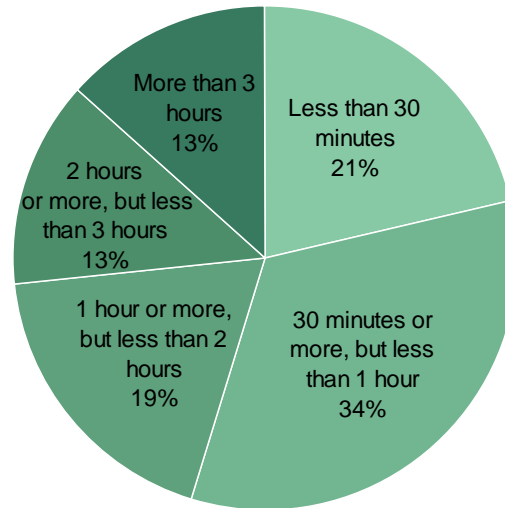
Testing Costs

Testers are inextricably involved in the problem resolution processes, and poor problem resolution processes affect the efficiency of the testing effort in the following ways:

- **Documenting problems discovered in test takes time.** Nearly half of the development managers and executives we interviewed reported that testing typically takes more than an hour to create a single problem report (see Figure 4). Problem reporting, like many other testing activities, is primarily a manual process, which also makes it especially error-prone. The more time testers spend communicating information about the problems they encounter, the less time they have for testing, making it more likely that problems will be discovered in production that could have been caught in test.
- **Testers have to approve fixes as well as find problems.** It doesn't matter if the problem is found in test or production or whether or not the problem is rooted in a software defect, testers are responsible for confirming that any changes made have the desired effect. The more efficient testing is, the faster this can occur, and the less time added to the problem resolution process. Just as development fixes to production problems pull developers off of their assigned projects, testing these fixes pulls testers off of their projects.
- **The test-fix sand trap extends testing timelines.** Not every problem fix is successful, and even those that are often create problems in other areas. The result is what Forrester calls the “test-fix sand trap,” testers find problems, developers resolve them, testers test again and find more problems, developers resolve those problems, too, and the cycle continues. This cycle is one of the major causes of delayed testing and release timelines, and it's one of the reasons why it's so hard to answer the question of how long testing will take.

Figure 4: Creating A Problem Report Often Takes More Than An Hour

“On average, how much time does it take testing, QC, and QA resources to create a report about an application problem discovered during testing?”



Base: All respondents

Source: Application Development Problem Resolution phone survey of 150 IT decision-makers responsible for overseeing development teams. A commissioned study conducted by Forrester Consulting on behalf of BMC Software, July 2007

Business Costs

It's easy for IT professionals to identify the effect of inefficient problem resolution processes on their own work, but it's often harder for them to account for the dramatic impact that slow time-to-resolution has on the efficiency and effectiveness of the enterprise as a whole. The development managers and executives who we interviewed pointed to the following business consequences of inefficient problem resolution processes:

- **Lower business customer satisfaction.** Topping the list of costs associated with inefficient problem resolution is reduced business customer satisfaction. In an enterprise IT organization, the speed with which IT responds to incidents in production is one of the primary ways that the business gauges the responsiveness of IT overall. Speedy resolution of problems is one of the primary factors that can positively contribute to customers' opinions of their software vendors, as there is little interaction with them post-sale. The speed with which problems are resolved in development and testing also contributes directly to cost and time-to-market, both of which are of concern to business customers.
- **Lower non-IT employee productivity.** Employees who rely on an internal application to perform their jobs are affected when the problems in question are in production and when problems in development and testing delay their access to new software. IT organizations frequently fail to account for these costs because they aren't billed back to the IT department. When the IT department's problems prevent employees from doing their jobs, the result is often a surge in ill will toward IT. It's harder for software vendors to ignore the impact that problems in their products have on their end users, as it frequently results in defections from their installed bases and loss of maintenance revenue.
- **Disruption of revenue-generating activities.** Inefficient problem resolution can disrupt revenue generation in many different ways. When an internal application is down, revenue-

generating employees may be unable to do their jobs. For example, a problem in an application that provides insurance quotes may prevent insurance agents in the field from selling policies. Alternately, problems in customer-facing applications may prevent customers from making purchases on their own; this is especially true of B2C applications like rental car reservation services and online shopping sites. If the application in question is a software product, the vendor's sales force may be unable to demo the application, or the services professionals may be unable to implement it to the prospect's or customer's satisfaction, thus preventing an initial sale or decreasing the chances of further sales. When inefficient problem resolution processes delay time-to-market, there is also an opportunity cost associated with a shorter window of time in which to make sales.

- **Damage to the company's brand.** When consumers find quality problems in an enterprise's software applications, they naturally draw conclusions about the caliber of the enterprise's products and services. This is not only true for production problems; problems in development and testing affect brand perception, as well. They may prevent a firm from bringing products to market in a reasonable time frame. It's tempting to think that problems in production matter only for consumer-facing applications, given the impatience of the typical consumer. But problems in partner-facing applications endanger partner relationships, too.

Sources Of Problem Resolution Process Inefficiency

Inefficient problem resolution processes have a clear and dramatic impact on both IT and the business. But what causes this inefficiency? Based on Forrester's knowledge of the problem space and the survey conducted for this study, we can point to the following as major root causes of inefficient problem resolution processes:

- **It takes too long to gather information about problems.** Although many shops have forms that must be filled out concerning each problem and automated routing of these forms, few have an automated method of collecting and assembling information about each problem. This means that customer service must spend time querying end users about their experiences, production support must spend time pulling data from various pieces of hardware and monitoring systems, and testers must retrace and record their steps and specify the exact build that they have tested. Should these steps prove to be insufficient, as is often the case, testers must try to find additional clues.
- **The information that's gathered about problems is inadequate.** Developers receive information about problems in a wide variety of media, from textual descriptions to screen shots to server logs. These information sources are rarely sufficient when taken on their own, and they often fail to be sufficient — let alone efficient — when combined. When the combination of data from all sources is insufficient, developers must scrounge for additional data, most often falling back on live conversation. But live conversation can only occur when all necessary parties are simultaneously available, which may not be possible for hours, days, or even weeks.
- **Differences in environments obstruct problem recreation.** Developers often have difficulty recreating the problems they're assigned because of discrepancies between development, test, and production environments or customer environments. These differences can be ameliorated, but in most circumstances they can't be avoided. These differences often prevent developers from recreating problems, and recreation is a

prerequisite for diagnosis and thus for resolution. As a result, our interviewees reported that on average, 25% of problems are returned marked “not reproducible.”

- **There’s too much back and forth.** There are dozens of stakeholders involved in the resolution of a single problem, and it’s rare that any one of them has access to all of the relevant information. The result is a great deal of asynchronous communication, which is less effective, as well as less efficient.

The Business Case For Better Problem Resolution Processes

All too often, IT organizations fail to recognize the true impact of inefficient problem resolution. Even when this is not the case, they may struggle to make the case for change. Based on the findings of this study, we recommend justifying investments into problem resolution tools based on time savings in development and/or testing and reductions in the time to resolve a problem. The following calculations should underpin these justifications:

- **Time savings for developers.** The findings of the survey conducted for this study indicate that developers spend an average of 29% of their time on problem resolution. Either based on this finding or on a number specific to your organization, estimate how much this number could be reduced by implementing better techniques for problem resolution. At that point, calculate the amount of time your investments will save per developer. Multiply the portion of developer time saved by your average fully burdened developer salary to determine your cost savings in this area.
- **Time savings for testers.** The calculation here is similar to the calculation of time savings for developers. Determine the proportion of time that testers spend on the problem resolution activities you propose to make more efficient, then calculate the amount of time you plan to save per tester, and multiply that by the cost of your fully burdened tester. This represents the return on your investment in better problem resolution processes for testers. These savings are likely to center around speedier problem documentation, a reduction in back-and-forth communications with development, and speedier resolution of problems discovered during testing — thus less wait time between testing cycles.
- **Reduced duration and thus cost of production problems.** The focus here is less on the cost of an inefficient problem resolution process than on the results of inefficiency in this area. Our research revealed that it takes an average of 6.7 days to resolve application problems. You may have different data about the mean time-to-resolution in your company. Regardless of the figure you have, you can estimate an average cost per application problem per unit of time, both in impact on the business and in impact on IT. Take the reductions that you estimate you can make in this time through investments, and multiply them by this cost to determine your expected savings.

Recommendations

Forrester’s recommendations for shops looking to improve their problem resolution process:

- **Determine the true efficiency — and cost — of your problem resolution process.** Use the figures presented here as a starting point or as a supplement to your own metrics. But to build consensus within your organization about the true extent of your problem resolution pains, gather your own data. How? It’s often easy to collect data from your bug tracking,

test management, and service desk applications, although it may be tricky to correlate the data from these disparate sources. Alternatively, field a survey of your own to developers and testers within your shop, asking some of the same questions posed in this study.

- **Decide what to tackle first and what returns to expect.** It may be that in your organization there are particular parts of the problem resolution process that are especially inefficient. Take a close look to determine whether this is the case, and then identify the people, process, and tool changes that could help in this area or even in multiple areas — and by how much.
- **Consider improving the problem resolution process bit by bit.** The entire problem resolution process may prove to be too much to bite off in one chew. If this is the case, adopt a more incremental approach. First focus on improving a single part of the problem resolution process, for example, the part that one role plays in it.
- **Use improvements in problem resolution to drive better apps-ops relationships.** Problem resolution is one of several disciplines that spans applications and operations; when problem resolution is dysfunctional, the apps-ops relationship usually is, too. Use improvements in your problem resolution process as the launching pad for a larger initiative to improve communication and collaboration across these two silos of the IT organization, or even across development, testing, and operations.

Appendix A: Supplemental Material

Related Forrester Research

See the August 20, 2007, “Tradeoffs And Tactics For Test Environments” report.

See the February 28, 2006, “Performance-Driven Software Development” report.