OpenLogic
Open Source: Choice. Confidence. Control.

Open Source Return on Investment: Achieving the Financial Promise of Open Source Software

A Joint Whitepaper by Navica and OpenLogic

Contents

This whitepaper, a joint effort of Navica and OpenLogic, describes how organizations can achieve the highest possible ROI from their open source efforts. The whitepaper begins by describing Navica's open source ROI framework and how several deployment scenarios provide different potential ROI results. The whitepaper then goes on to present how OpenLogic's solution can improve open source ROI by making an organization's use of open source more efficient and effective.

The Value of Open Source

By selecting high-ROI open source deployment scenarios, and increasing organizational efficiency through the use of OpenLogic's products, every company can ensure high financial payoff from their use of open source software.

Open Source ROI: Navica's Scenario-Based Analysis Framework

No two open source deployment scenarios are alike. In one scenario, an organization may consider replacing existing Microsoft Windows servers with Linux machines; in another scenario, a different organization may wonder whether it makes sense to create a new website with a commercial or an open source content management system.

It is obvious that every open source deployment situation differs and there is no single open source ROI answer. The question, then, is how to assess open source ROI for differing deployment scenarios and for different open source products.

Navica's scenario-based open source ROI methodology provides an analysis framework that enables organizations to assess the potential ROI of any deployment scenario. The Navica methodology provides four different deployment scenarios that use as their evaluation criteria the level of familiarity an organization has with the envisioned product technology and also whether the open source product will replace an existing product or is being newly introduced to the organization.

By applying the Navica framework, organizations can quickly determine if the scenario is likely to have a high ROI. In addition, the framework provides the factors that must be assessed to calculate detailed ROI figures.

Open Source ROI Fundamentals

What are the fundamental issues that organizations should keep in mind when thinking about open source ROI? In other words, what are the ground rules that underpin open source ROI assessments? Here are five key realities about open source ROI:

License Fees: Free Software Does Not Guarantee High ROI

While the absence of license fees is obviously attractive, it's not sufficient for high ROI. For example, if an enormous amount of work is required to implement an open source solution, the overall costs will outweigh the license fee savings of the product. Consequently, the lack of license fees is but one element in an overall financial assessment of an open source system. To fully assess open source ROI, all aspects of the system must be taken into account to ensure a full financial evaluation.

Not Everything is Free with Open Source

Even though open source software comes without a price tag, other aspects of an open source system carry a cost. For example, employee time spent installing, configuring, and integrating an open source product imposes salary costs. If an outside consultant is used, they will expect to be paid for their services. If new hardware is purchased to run the open source system, the machine will come with an invoice. Consequently, it's vital to fully account for all costs associated with an open source system to ensure an accurate ROI figure is calculated.

Hard and Soft Costs are Both Important

Whether you are using commercial or open source software, it is important to understand both the hard and soft costs. As the previous paragraph notes, some system costs come with an invoice. These types of costs, usually referred to as hard costs since they require explicit payment, are easy to calculate. One can simply add up the invoices presented by outside companies to calculate the total hard costs of a system.

Costs associated with use of internal company personnel are typically referred to as soft costs, since they are more difficult to accurately calculate. Many organizations do not have an explicit internal cost assigned for employee time. Moreover, most organizations do not accurately track employee time by task. Clearly, this makes it difficult to establish how much employee time and cost is associated with a given task.

Nevertheless, it's important to calculate the amount and cost of employee time used in implementing an open source system, since these soft costs can represent a significant portion of the overall project cost.

Software Lasts Forever

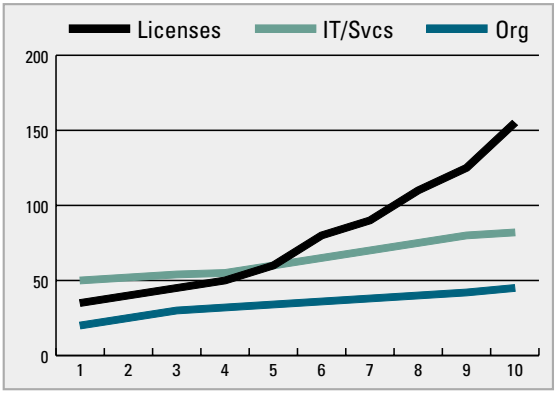
Many projects have financial calculations performed for initial implementation or for the first full year of operation. This form of financial analysis ignores a fundamental IT reality: systems have extended lifespans, since organizations are reluctant to replace a working system, given the cost and disruption associated with replacement.

Consequently, it is vital to use a realistic timeframe for project ROI calculations. It is particularly important to do so for open source-based systems. Since much of the cost of these systems is in early investment for internal personnel to learn and customize the open source product, a

short project time horizon can make it seem that an open source system is more expensive than a commercially-licensed product. However, because commercial products carry yearly maintenance fees, using a longer time horizon can more accurately calculate true project costs for the realistic time frame of the system.

Figure 1 illustrates the effect that project duration can have on the relationship of the different cost elements of a project. While license fees during the first year of the project represent a smaller percentage of project costs than do IT/Services costs, because of their recurring maintenance fees (typically 20% of original licensing fee), they eventually become the preponderance of total project cost.

Figure 1: Open Source Project ROI Cumulative Costs



Consequently, when making an open source project ROI calculation, it is vital to use a realistic project life projection, since this will enable a more accurate ROI assessment. A common practice for ROI project planning is to create a table with ROI calculations based upon three, five, and 10 year project durations. This allows project planning to proceed with a broader range of ROI information available to assist the decision process.

There is No Blanket Open Source ROI Story

The foregoing factors should make it obvious – while open source holds the potential for significant project ROI, it's vital to take all costs factors into account when calculating open source ROI. Because no two open source systems are alike, every project will differ in ROI.

Given this reality, Navica has developed a scenario-based ROI framework that provides guidance in calculating open source ROI.

The Components of Open Source ROI

While many specific costs go into calculating open source ROI, all of them may be assigned to three primary categories of cost:

- License/support/maintenance/subscription costs,
- IT/Service provider costs,
- Organizational costs.

Each of the categories captures a type of cost and simplifies ROI comparison analysis.

License/Support/Maintenance/Subscription Costs

These costs, referred to as License/Maintenance in the rest of this whitepaper, reflect the payment made to an outside entity in order to gain access to use of software. License fees are well understood as the up-front payment made before a vendor delivers software, while Support/Maintenance refers to the yearly payments required so that customers may gain access to patches and enhancements for software that has already been licensed. Subscription may be a less familiar term, but it is often used for both commercial and open source products. Subscription refers to a yearly fee paid to maintain access to the regular releases of a software product; in a sense, subscription may be thought of as being analogous to a magazine subscription in which a yearly payment is made to ensure regular delivery of updated product by the product's publisher.

These costs are hard costs, since they always are invoiced by an outside entity, thereby ensuring access to the software product.

IT/Service Provider Costs

These costs are those required for technical personnel and are associated with the work of installing, configuring, customizing, and operating the software. The factors affecting the size of this cost are how difficult a piece of software is to install and configure as well as how much custom programming and integration are necessary to modify the product to meet the organization's functional requirements.

Depending upon whether the technical work is done by internal or external people, this cost can be either soft or hard, or, indeed, can be a mix of soft and hard. For this reason, it is important to carefully assess the amount of work necessary to fully implement an open source system.

Organizational Costs

Organizational costs are those borne by the end users of a new system. Often referred to as "cognitive load," these costs reflect the learning curve imposed by new software. When a user begins to utilize new software, productivity goes down, since he or she is unfamiliar with the new system. The overall cost experienced due to this temporary reduced productivity is captured in Organizational Costs.

Open Source ROI Scenarios

In this whitepaper, we will analyze the relationship of the three project cost components for four different ROI scenarios:

- Open Source vs. Custom Systems
- Open Source vs. Commercial Systems: Unfamiliar vs. Unfamiliar
- Open Source vs. Commercial Systems: Familiar vs. Unfamiliar
- Open Source vs. Commercial Systems: Familiar vs. Familiar

Open Source ROI Scenario One: Custom Systems

Many organizations develop custom software themselves to meet internal needs. The benefits of custom development are clear: the organization gets software that meets its exact requirements, rather than using an externally developed package that forces it to compromise on its requirements. However, the downside of custom development is also clear: it's expensive to write custom systems from scratch.

The benefits of using open source rather than custom-developed software stem from the fact that open source software is distributed with binaries and source code rather than binary format alone, which is typical of commercial software products.

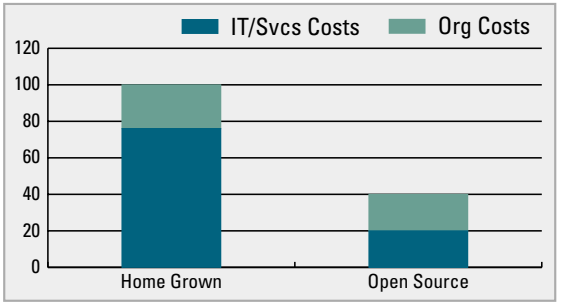
A company can use an open source product as components within its internal system and add its organization-specific code to the core product. In this fashion, it can still achieve its aim of creating a system that meets its exact requirements while avoiding the cost of developing the entire system top to bottom. Figure 2 illustrates the cost savings possible by using open source in a home-grown custom system.

As the figure indicates, IT/Services costs are much lower when using open source software. This reflects the fact that, rather than absorbing all development costs itself, the company is able to take advantage of the investment by other developers and organizations in the open source product.

In addition to the cost savings associated with development, there are ongoing savings as well. Because open source software is created by a community of developers, many bug fixes and enhancements will be written by outside developers. By contrast, should a company develop a custom system internally, it will take on the entire expense of bug fixing and product enhancement. Therefore, there are both initial and ongoing IT/Services savings available through the use of open source software when compared to custom development.

Less obvious, perhaps, is the fact that organizational costs may also be lower when using open source software. This is due to the fact that many other organizations have offered feedback about the software to the open source developers; this feedback has enabled them to improve the usability of the product. By contrast, a custom-built piece of software will have to begin receiving and incorporating feedback upon initial release, which means that the software will not be as user-friendly as an open source counterpart.

Figure 2: Custom System ROI



Open Source ROI Strategy

As Figure 2 indicates, replacing custom development with use of open source software offers significantly lower costs and raises the ROI of the open source option well above custom software development. Therefore, it is recommended that organizations should prefer use of open source software in this scenario, carefully assessing whether the requirements of the company mandate internal development.

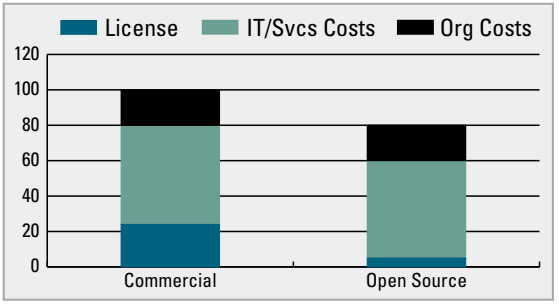
Open Source ROI Scenario Two: Unfamiliar Versus Unfamiliar

If an organization is considering implementing a system that utilizes technology the organization has no experience in using, it may be said the organization is unfamiliar with the product technology. Since the organization will be learning this new technology regardless of whether it selects a commercial or open source product, this scenario is referred to as "Unfamiliar versus Unfamiliar."

Figure 3 outlines the relative costs of using open source or commercial software in this scenario. Because the organization has no background in the technology, it will need to invest in training and skill development whether it uses a commercial or an open source product. For this reason, the IT/Services costs are similar for both open source and commercial alternatives. Similarly, because the system is new, the Organizational costs experienced upon system introduction are equal. After all, whether the system uses commercial or open source software, users will be learning a new application; therefore the Organizational costs will reflect the learning curve costs experienced by the user base.

The primary ROI difference in this scenario is the cost of the software itself. While the organization may choose to engage a commercial open source vendor to provide a subscription or support for the selected open source product, it is still likely to realize significant savings in comparison to the license fees associated with the commercial counterpart to the open source product. Typical savings available from open source subscription versus commercial license range from 75% to 90%, indicating the potential savings by selecting open source.

Figure 3: Unfamiliar vs. Unfamiliar ROI



Open Source ROI Strategy

As can be seen in Figure 3, the general strategy for this scenario is to consider open source as the default choice, and examine the detailed financial situation in order to make a final decision.

Open Source ROI Scenario Three: Familiar Versus Unfamiliar

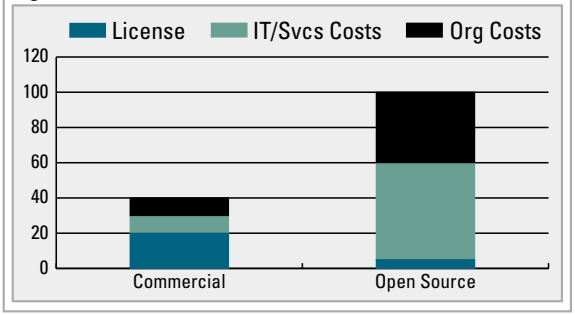
Sometimes an organization has an existing commercial software system that it wants to consider replacing with an open source alternative that is based on a different technology. An example of this is the potential replacement of a Windows server with a Linux machine.

While it is clear that this scenario has the potential to save License/Maintenance fees, the savings may not be as dramatic as they would appear at first glance. Since the initial license fee for the commercial software was paid in the past, it is a sunk cost that is unrecoverable at the current point in time. In this case, the ROI would depend on the cost of ongoing maintenance fees compared to the subscription or support fees possibly associated with the open source alternative available to the organization.

However, the IT/Services and Organizational costs for the open source alternative may be significantly higher if an open source choice is made. This is because the organization has already made a significant investment in skill development and overcoming the user base cognitive load associated with the existing system; however, should a new open source alternative be selected those costs would need to be borne once more.

Consequently, this scenario often poses ROI challenges in making an open source choice. Figure 4 outlines the relative costs of using open source or commercial software in this scenario. Rather than absorbing such a significant cost all at once due to a cutover to the new software, organizations often adopt a "Surround and Extend" strategy, in which the existing commercial software products are left in place and new open source products are implemented to provide complementary functionality. In this way, the organization can begin to realize the benefits of open source, while gradually investing in building IT/Services skills as well as user skills. A common example of this approach is the use of Linux as a file/print server in a Microsoft Windows Server infrastructure. The existing Windows Servers continue to provide application functionality, while the company begins to use Linux in a low-investment, low-risk fashion.

Figure 4: Familiar vs. Unfamiliar ROI



Open Source ROI Strategy

Figure 4 compares the commercial and open source costs in the Familiar vs. Unfamiliar ROI scenario. In this scenario, a good strategy is to "Surround and Extend" the legacy infrastructure, with an eye toward carefully evaluating open source ROI opportunities to ensure that open source is selected where it can deliver neutral or better ROI. The duration of the project is critical in this scenario, since the initial costs of building IT/Services skills, implementing the system, and educating the user base must be compared against the total amount of commercial software maintenance fees to be paid during the lifetime of the project. Different project duration assumptions may dramatically change the ROI of the project, so a complete analysis of the project is important for this scenario.

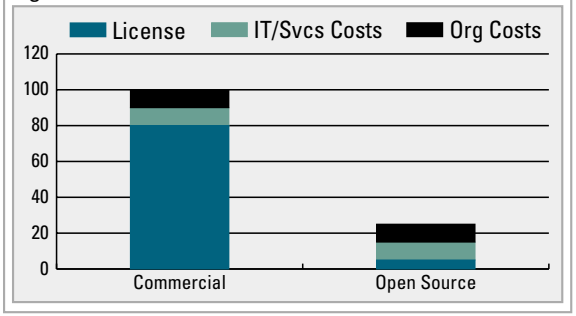
Open Source ROI Scenario Four: Familiar Versus Familiar

The previous scenario discussed the potential ROI when an organization considers implementing an open source product to replace an existing commercial product when the two products use dissimilar technology. The Familiar Versus Unfamiliar Scenario is, perhaps, the most difficult open source ROI scenario. The organization must absorb skill development costs which must be compared to the ongoing costs of commercial maintenance fees.

However, the circumstances are significantly different when the two alternatives share a common technology basis. Figure 5 outlines the relative costs of using open source or commercial software in this scenario. In the Familiar Versus Familiar scenario, no additional investment in skill development is necessary. The organization has already made that investment for the existing commercial product, which means that little additional investment is necessary when shifting to a new open source product. This is a common experience when comparing standards-based products, because the two products are bound to resemble one another, given that they both implement the same standard. For example, if an organization already has a significant installed base of a commercial Java application server like WebSphere or WebLogic, the cost of moving to a comparable open source product like JBoss is likely to be quite small, since most of the skills necessary for the commercial products will transfer quite easily to the open source product.

In this scenario, the IT/Services and Organizational costs are the same for both the commercial and open source alternatives; the primary difference between the alternatives is the cost of ongoing commercial maintenance compared to any subscription costs for the open source product.

Figure 5: Familiar vs. Familiar ROI



Open Source ROI Strategy

As demonstrated by Figure 5, the Familiar versus Familiar Scenario can be an extremely high ROI scenario, since open source subscriptions tend to be far lower than their commercial counterparts' maintenance fees. The recommended strategy is to seek out these opportunities very aggressively, since they present very high ROI potential.

Open Source ROI: A Summary

Open source is a relatively new phenomenon for most organizations, and its potential financial benefits are unclear because of its novelty. Even though its lack of license fees is immediately attractive, there are other variables to take into account when assessing the ROI potential of a particular project.

A number of ground rules should be taken into account when assessing open source ROI, including recognizing

that overall project costs include a mix of license fees, IT and services costs, as well as organizational costs. Furthermore, a realistic assessment of the likely lifespan of the project is important, since most systems are used well beyond the original estimates made at project inception.

It is critical to recognize that potential open source ROI can vary enormously, depending upon the individual circumstances of projects. A useful ROI assessment tool is to use scenarios to allow rapid ROI calculation. Navica's open source ROI scenarios provide a methodology to enable organizations to examine the potential financial benefits of their open source projects.

OpenLogic: Improving Open Source ROI

As the preceding part of this whitepaper makes clear, open source offers significant financial benefits for organizations that wisely choose deployment scenarios. However, common to all of the scenarios is the fact that open source ROI depends upon balancing commercial license fees against organizational costs, particularly IT/Services costs. Clearly, a solution that makes organizations more efficient in using open source will raise the ROI potential of its open source efforts. OpenLogic provides an open source management solution to significantly reduce the costs—and risks—of open source use, and thereby improves potential open source ROI.

The OpenLogic Solution: Addressing the Costs of Open Source Use

In order to gain the best possible ROI from deploying open source, it is important for organizations to select, deploy and use open source as efficiently as possible. OpenLogic enables the highest potential open source ROI by providing a commercial-grade level of support for hundreds of open source packages. Its solution provides all of the software and services that companies need to select, deploy, manage and support open source. OpenLogic minimizes the following cost elements that organizations experience when they use open source:

Sourcing and Selection

There are well over 100,000 different open source products. As can be imagined, the viability, functionality and quality of these products vary enormously. So a constant question raised regarding open source is "what are the right products for me to use?"

Assigning internal personnel to sift through the myriad of open source products is expensive in terms of direct employee time. Even worse is the time-to-market cost that the selection and evaluation process causes – each week that employees are involved in selection is a week that new solutions are not being delivered to meet business needs.

OpenLogic reduces selection cost for organizations by delivering a certified library of hundreds of the most popular enterprise open source packages. OpenLogic chooses the best open source products in each category and uses a rigorous 42-point evaluation process to certify that each package in the OpenLogic Certified Library meets the needs of enterprises in 5 key areas: viability, licensing, functionality, technology and support. OpenLogic's screening process ensures that companies can be confident they are using the very best open source products to address their business requirements.

The OpenLogic Certified Library enables organizations to bypass an extended evaluation process, reducing time from internal personnel and aiding all-important time-to-market considerations.

OpenLogic ROI Savings

Companies ordinarily save approximately 50% on their evaluation and selection costs by using the OpenLogic solution. While difficult to estimate, the time-to-market advantage enabled by OpenLogic is certainly significant.

Productivity Drains

Even after an organization has narrowed the universe of open source products down to a manageable number and created its approved list for deployment, it faces the issue of how to efficiently deploy and manage open source in the enterprise. Open source presents two technical challenges. First, it is highly componentized, requiring the integration of multiple disparate components to create a solution. Second, open source is frequently updated with new patches and releases. This constant flow of new products creates unending work to install and configure each new product release and poses a combinatorial nightmare of integration and testing to ensure the organization's infrastructure continues to work properly. In short, there is a huge amount of technical work presented by the rapid evolution of open source.

The level of effort can be easily understood with a simple example. With an infrastructure containing just 10 open

source products, when a new version of a product is released, a company must create a testbed of the existing infrastructure. Once that task is complete, the new version of the updated product must be installed and configured. After the stack is brought up to date, the overall application must be tested to ensure it still functions reliably with the newly-released product within the stack. After thorough testing, the stack may be moved into production. However, there are nine other products used as part of the stack. As soon as a new version of another product is released, the entire cycle begins again. This effort often takes days, or even weeks, for many companies.

As can be easily imagined, this task can seem endless. With minimal documentation provided by open source products, it can be time-consuming and error-prone to get open source packages on varying release cycles to work together. This tedious effort means that valuable technical personnel are being used to repetitively execute a configure-and-test cycle –increasing the risk that talented employees will leave for other opportunities that provide more challenge.

OpenLogic Enterprise eliminates these productivity drains by automating the process of installing, configuring, integrating, and testing open source packages. The end result is a customized, ready-to-run stack that the organization can install and use as a coherent whole, instead of a piecemeal collection of components. Instead of using valuable technical personnel to manually install, configure, and integrate open source components, organizations can direct their resources to more valuable business objectives. And with OpenLogic, the entire process can be reduced to hours, instead of days or weeks.

OpenLogic also eliminates the need for organizations to drain valuable resources on frequent open source updates. The OpenUpdate subscription service continually monitors the flow of new open source releases – evaluating and re-certifying each new version and notifying users when it is available.

OpenLogic ROI Savings

The delivery of an automated solution to deploy and manage open source provides significant benefits. Typical organizations realize a nearly 70% reduction in productivity drains when they move to the OpenLogic solution. Beyond the obvious economic benefits is increased employee satisfaction and reduced time-to-market available through use of the product.

Compliance and Legal

There are well over sixty open source licenses approved by the Open Source Initiative (OSI). In addition, there are literally hundreds of other open source licenses that have never been submitted to OSI. Each license carries rights, responsibilities, and some amount of risk with it. Tracking licenses, ensuring license compliance with company standards, and mitigating risk all require investment of expensive company time and talent. As new product releases become available, and licenses change over time, the time and talent investment goes on and on. Without a central, convenient method of assessing risk, approving licenses and auditing for compliance, the benefits of open source to the organization may be reduced due to added work and concern regarding risk exposure.

The OpenLogic solution reduces both the workload and risk associated with open source licenses and compliance. It enables companies to define and enforce license policies, ensuring that only products with company-approved licenses are available for developers and users.

The OpenLogic solution also provides a knowledgebase about open source licenses, giving detailed information about each license so that organizations may make their license selections based upon thorough knowledge about licenses and their implications.

Furthermore, the OpenLogic solution creates an audit trail of what products have been deployed, ensuring that the organization has a record of what licenses and versions have been used in its software infrastructure.

Finally, OpenLogic provides indemnification to its users for all products in its certified library. This simplifies the issue of risk reduction enormously, since the organization does not need to maintain individual indemnification agreements for each product. The OpenLogic indemnification also offers consistent conditions and coverage, reducing the challenge of tracking varying terms and conditions and protection according to the conditions offered by different vendors.

OpenLogic ROI Savings

By using the OpenLogic solution, a typical organization can save over 50% of the costs associated with legal and compliance efforts.

Support

Open source products provide unique support challenges. Open source communities typically provide mailing-list

support. Users of an open source solution can post a question, and other mailing list participants can respond. Unfortunately there are no guarantees of timeliness or accuracy of the response. In addition, for the handful of top open source products, there are vendors that provide commercial support for open source products. The cost of this support varies by vendor. For most organizations, however, the main source of support are internal resources. Although many organizations do not track the effort formally, internal support can create significant hidden costs. The major challenge for most organizations is how to efficiently obtain support and avoid wasted and duplicated effort in support situations. The numerous support options can actually become a hindrance to companies, as support becomes diffused through redundant channels and conflicting advice.

This problem is compounded due to the complexity of integration both among open source components and between open source, commercial and custom developed solutions. When a problem surfaces, it may manifest itself in one product, but actually be created by another. There may, in fact, be several components in between the product in which the problem manifests and the true root cause of the problem.

Attempting to solve support issues that involve products provided by several different organizations is a traditional problem for IT organizations. The highly granular nature of open source magnifies this problem, since more products may be part of the software stack. For technical personnel, trying to track down a problem and then isolate it, while being forced to manage several different support mechanisms representing the various components in the stack, represents an enormous challenge.

Beyond the inefficiencies of multiple and conflicting support mechanisms is the fact that support is a key resource for effective technical development and deployment. Every minute spent identifying appropriate support channels, trying to make sense of conflicting recommendations, or trying to get up-to-speed on unfamiliar products is time not being devoted to solving the problem at hand, not to mention the unproductive time wasted as other technical personnel or end users wait for the issue to be resolved.

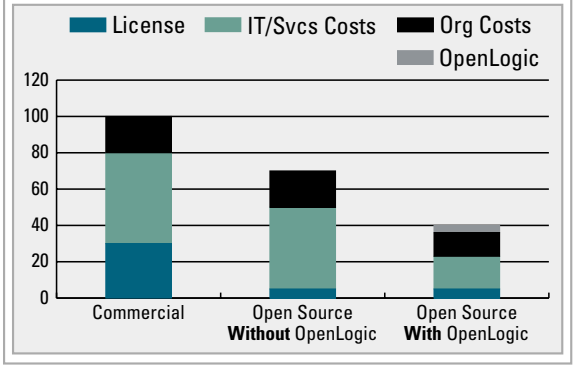
OpenLogic significantly reduces the challenge of open source support. As an end user, rather than tying together multiple support channels yourself, you can rely on OpenLogic to provide a single point of contact and responsibility for solving open source support issues across hundreds of open source products.

Beyond offering a single point of contact for all support needs, OpenLogic has ensured it can solve the most challenging open source support issues. OpenLogic has created the OpenLogic Expert Community, made up of source code contributors to the products contained in the OpenLogic Certified Library. When a problem is referred to OpenLogic, should the solution require code changes or in-depth troubleshooting, OpenLogic can turn to its community of experts for a fix. Since these experts are already committers and contributors to the open source products they support, they can ensure that any changes are incorporated in future releases. By maintaining relationships with many open source developers located worldwide for round-the-clock support and advice, OpenLogic ensures that it can meet the service levels that organizations demand.

OpenLogic ROI Savings

By serving as a consolidated support resource, OpenLogic can offer a far more complete support mechanism than any company could create on its own. Typical OpenLogic users save nearly 60% of the cost of the traditional open source support mechanisms, while receiving support with quicker response, more consistent quality, and deeper technical knowledge.

Figure 6: Open Source ROI with OpenLogic



OpenLogic ROI Summary

As Figure 6 illustrates, OpenLogic further improves the ROI potential of open source, reducing the IT/Services and Organizational costs of open source significantly.

Example: Raising Open Source ROI with OpenLogic

As a hypothetical example (see Table 1), consider a typical company with 100 different open source users using an open source software stack comprised of 50 different products. They will probably be organized into teams of up to a dozen individuals, with varying responsibilities – Architect, Programmer, DBA, Web Designer, and Support. Each team, and possibly even different members within teams, will use a different combination of open source components.

Table 1: Comparing Open Source Without OpenLogic To Open Source With OpenLogic

Costs	Use OSS **Without** OpenLogic	Manage OSS **With** OpenLogic	Savings
Sourcing and Selection	$660,000	$342,000	48%
Productivity Drains	$1,100,000	$339,000	69%
Compliance and Legal	$493,000	$216,000	56%
Support	$1,600,000	$675,000	58%
TOTAL	**$3,853,000**	**$1,572,000**	**60%**

In every phase of a project, OpenLogic makes open source use more efficient, and therefore less expensive. In this example, savings range from 48% to 69%, with the greatest percentage savings achieved in the highest cost elements of the project: technical overhead and support.

Overall savings were 60%, reducing the original $3.8 million project cost to only $1.6 million. Obviously, the ROI of this solution is raised considerably through the use of the OpenLogic solution.

Unseen in this ROI analysis, however, are the qualitative benefits of using OpenLogic. Because the OpenLogic solution makes using open source more efficient, it reduces the duration of projects as well as their cost. Reducing the time spent in selecting products, assessing their licenses and legal conditions, installing and configuring them, and locating support resources means that project teams operate more effectively and bring solutions into production more quickly.

OpenLogic: The Higher ROI Open Source Solution

Open source holds tremendous potential for making IT spend more effective. Much lower license fees offer the opportunity to reduce the overall cost of applications. Just as important, lower license fees make it much more feasible to avoid the "buy before you try" dilemma common to proprietary software solutions.

Every open source-based project is different, which means that there is no single open source ROI answer. Depending upon the circumstances of the individual project, the ROI of an open source solution can be enormous or negligible.

As companies make more significant commitments to open source, the challenges of open source become clear: selecting the right products; ensuring that legal risk is minimized; reducing the time spent installing, configuring and updating the products; and obtaining high quality, responsive support.

The OpenLogic solution addresses those challenges and can make organizations much more effective in their open source use. By providing a certified set of open source products, including OpenUpdate for certified new releases, reducing risk through an overall indemnification program, and ensuring that technical personnel focus their time on creating and delivering applications, rather than investing hours and days in installation and support, OpenLogic ensures that its customers attain the highest possible open source ROI.

OpenLogic is a leading provider of solutions that enable enterprises to easily customize, deploy and manage open source environments. OpenLogic's offering includes an open source management platform, a library of 200+ open source products, flexible stacks and consolidated technical support. With OpenLogic, enterprises can reduce the total-cost-of-ownership for open source technologies. OpenLogic is currently used by 700 customers worldwide.

OpenLogic, Inc. Phone: 1-888-OPENLOGIC
www.openlogic.com