Java Development Productivity and Quality Using Eclipse:

A Comparative Study of Commercial Eclipse-based IDEs

The productivity benefits of using commercial Eclipse-based Java IDE products from IBM (IBM Rational Application Developer), Genuitec (MyEclipse), and CodeGear (JBuilder) compared to the freely downloadable baseline Eclipse configuration.

Report Prepared by CostXperts

The Cost Xpert Group, Inc. (www.CostXpert.com) specializes in software metrics and predictive models. Considered one of the top experts in the field of software development cost analysis our services are focused on helping clients substantially increase the probability of completing software development projects successfully.

Our staff combines real-world software project management experience with technical training in areas such as parametric cost analysis, system dynamic modeling of software processes, knowledge based modeling of risk, and both stochastic and deterministic optimization of project operations. Many of our consultants are world-renowned leaders in their field of expertise. The CostXpert Group has over 5,000 customers including Boeing Corporation, Chevron Information Technology, Ernst & Young, Hewlett-Packard, and Unisys Corporation

Executive Summary

Eclipse is both a phenomenon and success story in the Java eco-system and in the overall software development field. More than just a software development tool, Eclipse represents an open source community dedicated to building a development platform and to offering a wide range of extensible frameworks, tools and runtimes for building, deploying and managing software across the application lifecycle.

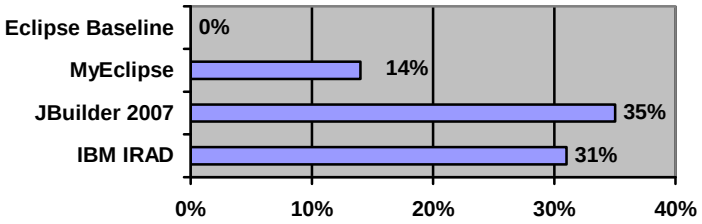
The attraction of Eclipse can be attributed to many factors including its open source model, flexibility, expandability, extensive commercial industry support, and of course low cost; a baseline Eclipse configuration for Java can be downloaded for free. Many vendors enhance this base configuration with value-added technologies and services for which they charge a license or support fee.

For instance, companies such as IBM, Genuitec, and CodeGear (Borland's Developer Tools spin-off) have developed new Java IDE solutions based on Eclipse. Each of these three Eclipse-based solutions has taken a different strategy and approach to enhancing the baseline Eclipse configuration, delivering unique value to Java developers. By comparison, some companies, such as Sun Microsystems and JetBrains, license development tools based on proprietary technologies developed independently from Eclipse's open framework.

The goal of this study was to objectively measure the benefits of using commercial Eclipse-based Java IDE products from IBM® (IBM Rational Application Developer®), Genuitec® (MyEclipse®), and CodeGear (JBuilder® 2007 Enterprise Edition). These benefits are compared to the freely downloadable baseline Eclipse configuration.

In this study, team configurations and projects of varying sizes and purposes were modeled and measured under two scenarios: (1) building new Java software and (2) enhancing/maintaining existing Java applications. The study measured development cost, time to completion, and resulting application quality. In all situations, all three commercial IDEs (MyEclipse, JBuilder, and IRAD) were found to offer substantial development cost savings and project quality improvements over the baseline free Eclipse distribution.

Average Organization Net Cost Savings



For typical software development organizations, these percentages translate into substantial net hard dollar savings in terms of software development personnel, time and quality. For the representative organizations used in this study, the return on investment (ROI) of acquiring JBuilder ranged from 90:1 to 165:1. That is, for every dollar spent on JBuilder, an organization can expect a return of $90-165 in savings through developer productivity and improved quality.

For every dollar spent on JBuilder, an organization can expect a return of $90 to $165 in savings through greater developer productivity and improved quality

Background

Since its introduction to the Java world slightly over five years ago, Eclipse has quickly grown to a position of dominance. According to a BZ Research report, Eclipse has a 70% market share for Java development[1], and IDC estimates that there were 2.27 million Eclipse users as of August 2006[2]. Borland's CodeGear business unit has been part of the Eclipse consortium through Borland since the beginning, and CodeGear has fully embraced the Eclipse platform with the release of JBuilder 2007. JBuilder 2007 enhances and extends Eclipse and open source technologies by simplifying installation and management, integrating "best of breed" products and plug-ins from the Eclipse community, and integrating additional lifecycle management tools.

In the last several years commercial Java IDEs based on Eclipse have begun to ship from established vendors such as CodeGear (formerly the Borland Developer Tools Group) and IBM, as well as from early stage companies such as Genuitec. These commercial Eclipse-based IDEs can deliver increased value to developers and development organizations with enhanced functionality offered through the Eclipse plug-in architecture**.**

Objectives

The overall objective of this study was to measure the productivity gains from using a commercially available Eclipse-based IDE when compared to the standard Eclipse distribution (version 3). The study evaluates three different commercially available Eclipse-based IDEs: IBM Rational Application Developer v. 7, Genuitec MyEclipse v. 5, and CodeGear JBuilder 2007 Enterprise Edition.

Our overall objective is to compare the productivity of development teams using Eclipse 3, My Eclipse 5 and IBM's RAD (IRAD), and CodeGear's JBuilder 2007 Enterprise Edition

All three commercial IDE products are designed to provide a turnkey Java development experience based on Eclipse. Each company takes a different approach to building on top of the Eclipse open source framework to create their solutions.

With Rational Application Developer, IBM takes a "unified platform" approach tuned for teams of developers all using the IBM/Rational environment (i.e.: WebSphere application server, Rational Unified Process, and Rational Lifecycle tools). IBM's approach is weighted toward increased developer and team productivity for organizations that have already standardized on IBM's development and deployment platforms, backed by a range of enterprise support options.

Genuitec takes a bundled function approach by including a mix of open source Java components along with proprietary and non-Eclipse components in a "turnkey" Java IDE package. Genuitec's approach provides a variety of Java feature components in an integrated distribution with a value price point. Genuitec's approach is weighted toward good value at a low cost per developer with minimal support options.

1 http://www.sdtimes.com/article/LatestNews-20070515-06.html

2 IDC, Worldwide Adoption of Eclipse: The Framework Resonates with Java Developers in All Regions, Doc #203080, Aug. 2006

CodeGear takes an “integrate, enhance and extend approach”. In JBuilder, CodeGear has integrated popular open source components as part of the company's Eclipse-based IDE. CodeGear has also included value-added features that deliver improved productivity, quality, and team collaboration for Java development. JBuilder’s approach is weighted toward increased developer and team productivity as well as application quality in heterogeneous development and deployment environments (e.g.: supporting multiple technologies for SCM, bug tracking, project management, web server, and application server).

Project Analysis Results

The study evaluates each of the Eclipse-based tools/environments across four different categories of projects:

1. **New-Large**: A new, multi-tier enterprise application with multiple middle-tier servers consisting of approximately 2,000 function points worth of delivered functionality;

2. **New-Small**: A new desktop application consisting of approximately 100 function points worth of delivered functionality;

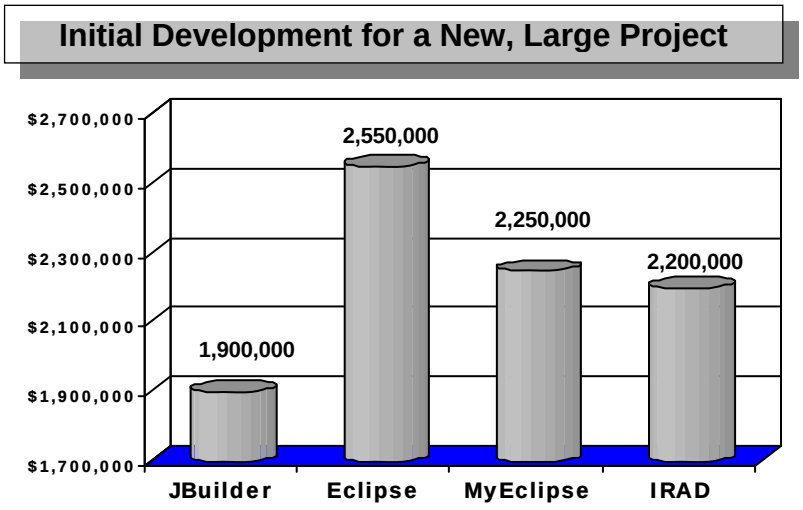
3. **Enhance-Large**: A project to enhance a multi-tier enterprise application with multiple middle-tier servers consisting of approximately 2,000 function points worth of delivered functionality;

4. **Enhance-Small**: A project to enhance a desktop application consisting of approximately 100 function points worth of delivered functionality.

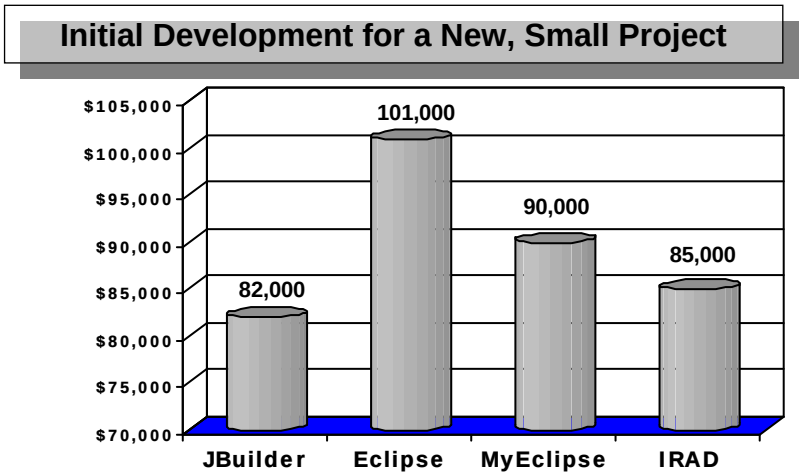
For each of these categories, the study estimates the overall return on investment (ROI) for three different types of business entities: (1) a large consulting organization, (2) a multi-billion dollar global corporation, and (3) a young, rapidly growing company.

It must be stressed that the costs shown are total development lifecycle costs and not just coding effort. Many of the differences that we measured among the IDEs were driven by advantages in areas such as design, configuration management, testing, environment definition and maintenance, and quality assurance.

For new projects, as shown in the following figures, JBuilder offers a statistically significant cost advantage over the baseline Eclipse configuration and MyEclipse. The cost advantage ranges from 10% to 34% depending on the size of the project and on the particular IDE. JBuilder also offers a 4-16% cost advantage over IBM's RAD (IRAD), the low end of which is within our error tolerance but should still be considered statistically significant.



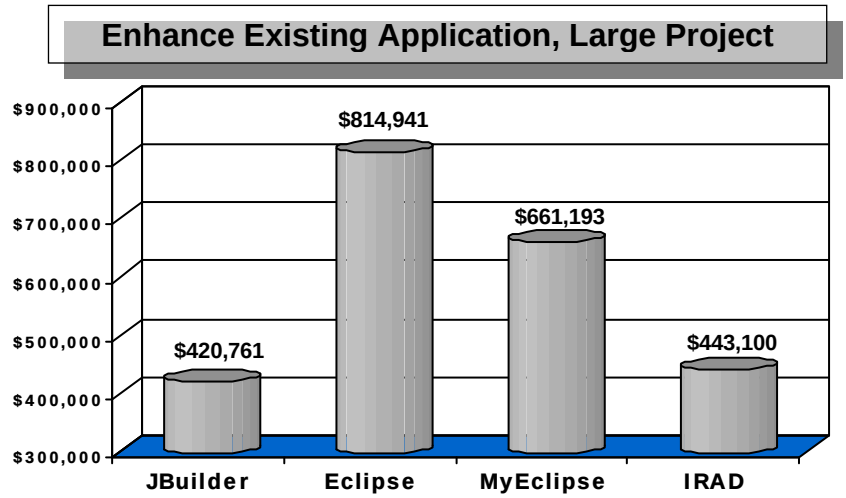
Cost to Build a New 2,000 Function Point Enterprise System



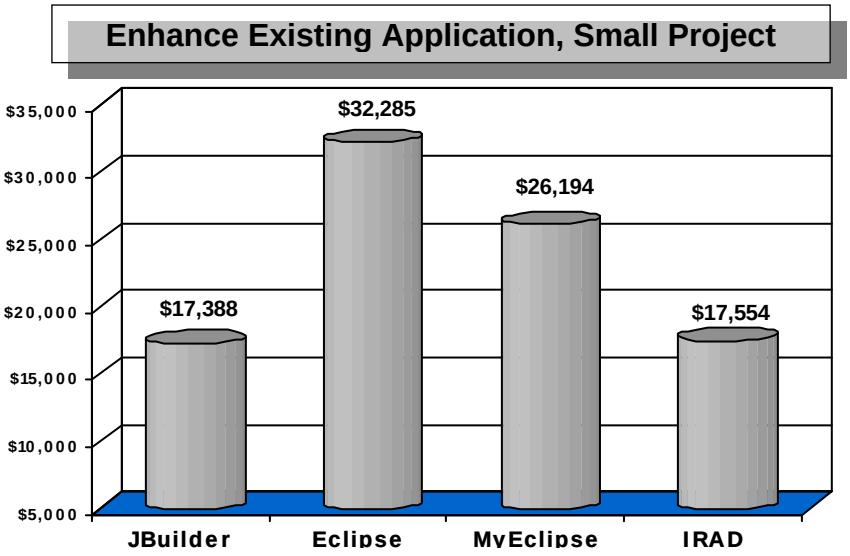
Cost to Build a New 100 Function Point Enterprise System

The distinct JBuilder advantage is not driven by improvements during the coding phase as much as it is by advantages in areas such as design, configuration management, testing, environment definition and maintenance, and quality assurance.

For enhancement projects JBuilder outperforms the baseline Eclipse configuration and MyEclipse by a range of 51% to 94% depending on the size of the project and the IDE. JBuilder also performs better than IRAD on enhancement projects, ranging from 1.7% (small projects) to 5.2% (large projects). These numbers fall within our error of tolerance and for small projects represent a tie from a statistical point of view.



Cost to Enhance a 2,000 Function Point Enterprise Application



Cost to Enhance a 100 Function Point Desktop Application

In sum, for both large and small projects, and for new or existing applications, the advantages of IRAD and JBuilder are indeed significant. We measured similar benefits across all other dimensions of analysis including development schedule, residual defects, and maintenance costs. For additional details, please see Appendix B

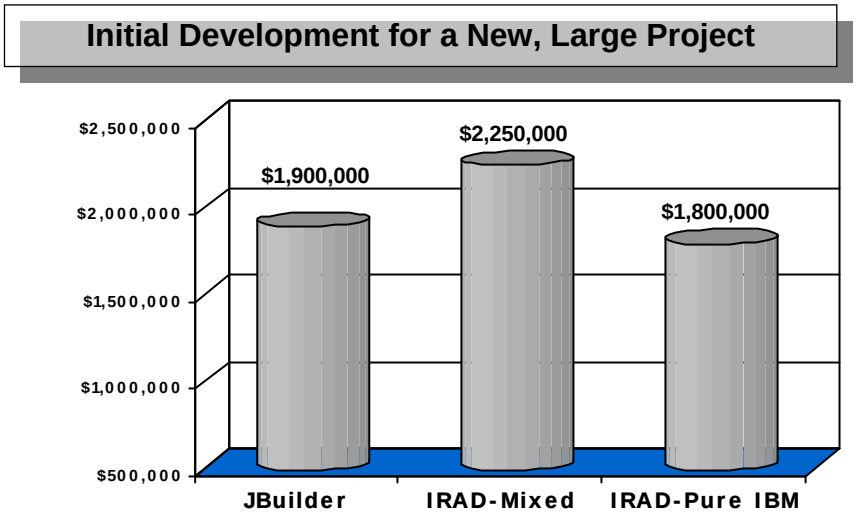
Many of the JBuilder advantages over other Eclipse-based IDEs result from:

- Better management tools
- Enhanced tools for managing risk and resolving software architecture issues
- Static and dynamic code analysis
- Improved tools for reverse engineering code into UML documentation, thereby reducing time required to assess, assimilate and understand the legacy code

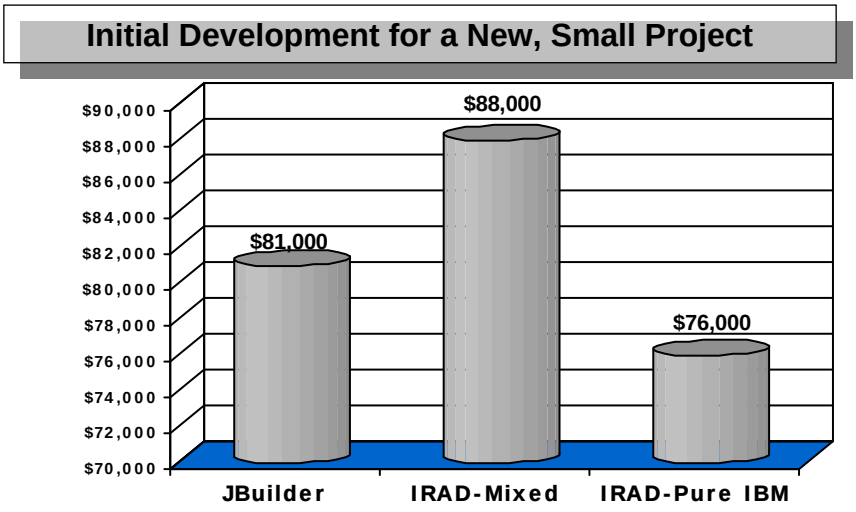
Added Dimension: Homogeneous Versus Cross Vendor Environments

JBuilder and IRAD offer similar capabilities and performance, with JBuilder slightly outperforming IRAD when looking at Java projects in general. Based on our experience working with customers supporting diverse IT environments, we decided to extend our analysis to compare projects using a pure IBM solution (WebSphere, etc.) versus those projects using IRAD with other third party components (such as Apache or JBoss). The results were quite interesting.

We found that for new large development projects in a pure IBM environment, IRAD outperformed JBuilder by 5%, while in a mixed tool/server environment JBuilder outperformed IRAD by 15%. The results were similar for new development small projects, although JBuilder's advantage over IRAD in this situation was reduced to 10%. These results are summarized in the following figures.

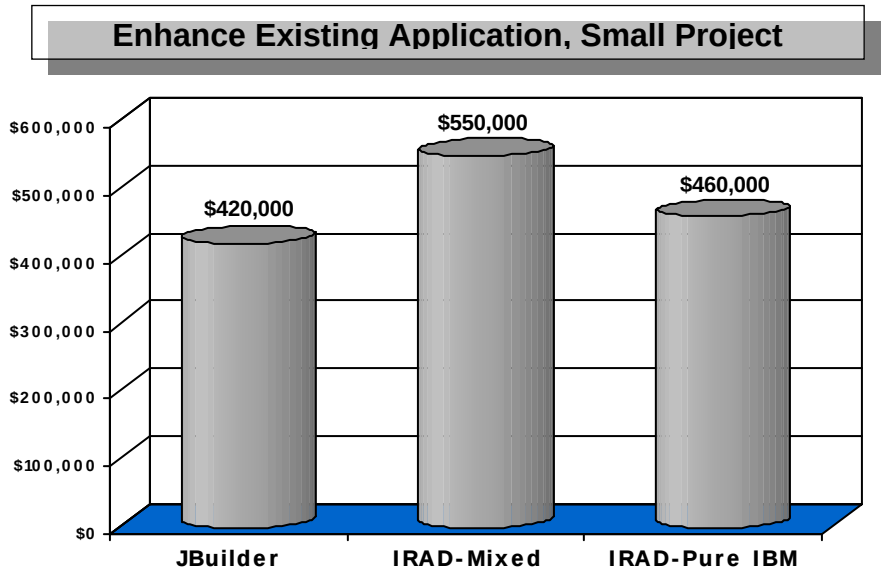


JBuilder versus IRAD for 2,000 Function Point Project



JBuilder versus IRAD for 100 Function Point Project

We expected to see similar results for enhancement projects, but were surprised to find that the JBuilder tools allowed the CodeGear solution to outperform IRAD on enhancement projects even in a pure IBM Environment. The results for a large enhancement project are as follows:



JBuilder versus IRAD for 2,000 Function Point Enhancement Project

Representative Company Calculations

Having completed our cost/benefit analysis on the basis of an individual project (e.g.: "New, Large" development project), our next step was to perform a cost/benefit analysis for three example companies each having multiple concurrent projects.

In this section we'll be looking at the impact of a decision to standardize on the baseline Eclipse configuration, MyEclipse, JBuilder 2007 Enterprise Edition, or IRAD for Java development work. We used Eclipse as the baseline for this comparison, as the Eclipse software/environment is free. Return was then defined as the cost savings associated with improved productivity and quality given the investment in a development and maintenance support license on top of Eclipse.

We assumed an average Java software project staff salary (including developers, analysts, testers, documentation staff, and so on) of $65,000 per year and a labor loading of 2.1, resulting in a loaded cost of $66 per Java development hour.

For this analysis we used approximate retail pricing and therefore assume the per developer cost to acquire licenses as:

Development Tool	First year cost to acquire licenses, support and maintenance (estimated)
Baseline Eclipse Configuration	$0
MyEclipse	$50
JBuilder 2007 Enterprise Edition	$2,000
IRAD	$3,500

The companies were defined as follows:

***Custom Development, Inc**.:* Custom Development, Inc. is a large consulting organization with a focus on software development. Approximately 70% of their current tasking involves Java, and the projects include a mixture of new development, and enhancement projects. The Java projects/tasks anticipated for the next year are as follows:

Project Type	Number of Projects
New-Large	250
New-Small	500
Enhance-Large	750
Enhance-Small	250

IT Intensive, Inc.: IT Intensive, Inc. is a multi-billion dollar global organization in which large IT systems are central to their operations. The software is a mixture of languages, including Cobol, C++, and Java. Roughly 30% of their work involves Java. Most of the New projects are add-ons to existing applications, and a high percentage of their work involves maintaining and enhancing existing legacy applications. The Java projects/tasks anticipated for the next year are as follows:

Project Type	Number of Projects
New-Large	100
New-Small	100
Enhance-Large	1,000
Enhance-Small	50

Young and Growing, Inc.: Young and Growing, Inc. is a relatively new company with revenues of $10-25M and experiencing rapid growth. Software is critical to their success, and time to market for new development and enhancements is a primary concern. Approximately 80% of their projects involve Java, and most of the work is new development. The Java projects/tasks anticipated for the next year are as follows:

Project Type	Number of Projects
New-Large	5
New-Small	20
Enhance-Large	5
Enhance-Small	2

Results

We have seen that limiting developers to use the baseline Eclipse configuration distribution is significantly less efficient than purchasing any of the three alternate environments from CodeGear, IBM, or Genuitec. The benefit of these commercial solutions translates into an ability to improve productivity and quality significantly. For our three sample companies the results in terms of software development staff required are as follows:

Total Java Project Staff Required	**Custom Development, Inc.**	**IT Intensive, Inc.**	**Young and Growing, Inc.**
Baseline Eclipse Configuration	6,775	5,618	98
MyEclipse	5,865	4,701	87
JBuilder 2007 Enterprise Edition	4,402	3,249	70
IRAD	4,648	3,428	74

Of course, there is an obvious relationship between head-count and costs. Using the previously discussed $66 per labor hour, the payroll related costs (including overhead) are as follows:

Total Java Project Costs	***Custom Development, Inc.***	***IT Intensive, Inc.***	***Young and Growing, Inc.***
Baseline Eclipse Configuration	$ 930,072,000	$ 771,239,040	$ 13,453,440
MyEclipse	$ 805,147,200	$ 645,353,280	$ 11,943,360
JBuilder 2007 Enterprise Edition	$ 604,306,560	$ 446,022,720	$ 9,609,600
IRAD	$ 638,077,440	$ 470,595,840	$ 10,158,720

These cost savings do not come without a price. All environments (except the baseline Eclipse distribution) carry license acquisition costs, maintenance and support. If we assume that 30% of the staff on an average project require a Java software license we can use total headcount to determine the number of licenses required, and multiplying this number by the cost per license, we arrive at:

License, Maintenance, and Support Cost	***Custom Development, Inc.***	***IT Intensive, Inc.***	***Young and Growing, Inc.***
Baseline Eclipse Configuration	$ 0	$ 0	$ 0
MyEclipse	$ 88,000	$ 70,500	$ 1,300
JBuilder 2007 Enterprise Edition	$ 2,642,000	$ 1,950,000	$ 42,000
IRAD	$ 4,879,000	$ 3,598,000	$ 77,000

For the sake of simplicity, we have chosen to treat all these costs as annual recurring fees (no change from year to year), which in some cases may overestimate the total cost. For example if the license fee is an annual subscription, then the costs will not vary from year to hear. However, in the case of a one time license fee plus annual support/maintenance, the actual annual cost will be less after year one because the license fee does not apply in subsequent years.

The net cost savings over pure Eclipse is simply the total project cost savings minus the license, maintenance and support acquisition costs. These results were as follows:

Net Cost Savings Over Eclipse	***Custom Development, Inc.***	***IT Intensive, Inc.***	***Young and Growing, Inc.***
Eclipse	$ 0	$ 0	$ 0
MyEclipse	$ 124,836,800	$ 125,815,260	$ 1,508,780
JBuilder	$ 323,123,440	$ 323,266,320	$ 3,801,840
IRAD	$ 287,115,560	$ 297,045,200	$ 3,217,720

Or expressed as percentages:

Net Cost Savings Over Eclipse	***Custom Development, Inc.***	***IT Intensive, Inc.***	***Young and Growing, Inc.***
Eclipse	0%	0%	0%
MyEclipse	13%	16%	11%
JBuilder	35%	42%	28%
IRAD	31%	39%	24%

Conclusions

In sum, for organizations using Eclipse for their Java software development, this report highlights the added value provided by commercial Eclipse-based IDEs. Our study evaluated three specific commercial products: MyEclipse from Genuitec, IBM Rational Application Developer, and CodeGear JBuilder 2007 Enterprise Edition.

Using any of these commercial tools, developers can expect to improve productivity and quality of their projects compared to using the baseline Eclipse distribution, corresponding to a net benefit anywhere from 11-42% of the total development budget. More specifically, the study included three example companies for which the net annual savings ranged from $1.5M to $323M (depending on the size of the development staff).

We also investigated some of the differences between the three commercial Eclipse-based IDEs. In general, we found that each of the three products has unique strengths. MyEclipse, for example, offers a solution for organizations with limited budget, lower support requirements, and fewer lifecycle management requirements. On the other hand, IRAD and JBuilder offer more robust solutions with enhancements for team collaboration, productivity, and quality. IBM's solution excels in development/deployment environments built entirely on IBM technologies. JBuilder provides a solution particularly well suited to heterogeneous development and deployment environments based on technologies from a variety of vendors.

Appendix A

Overview of Cost Xpert Model and Approach

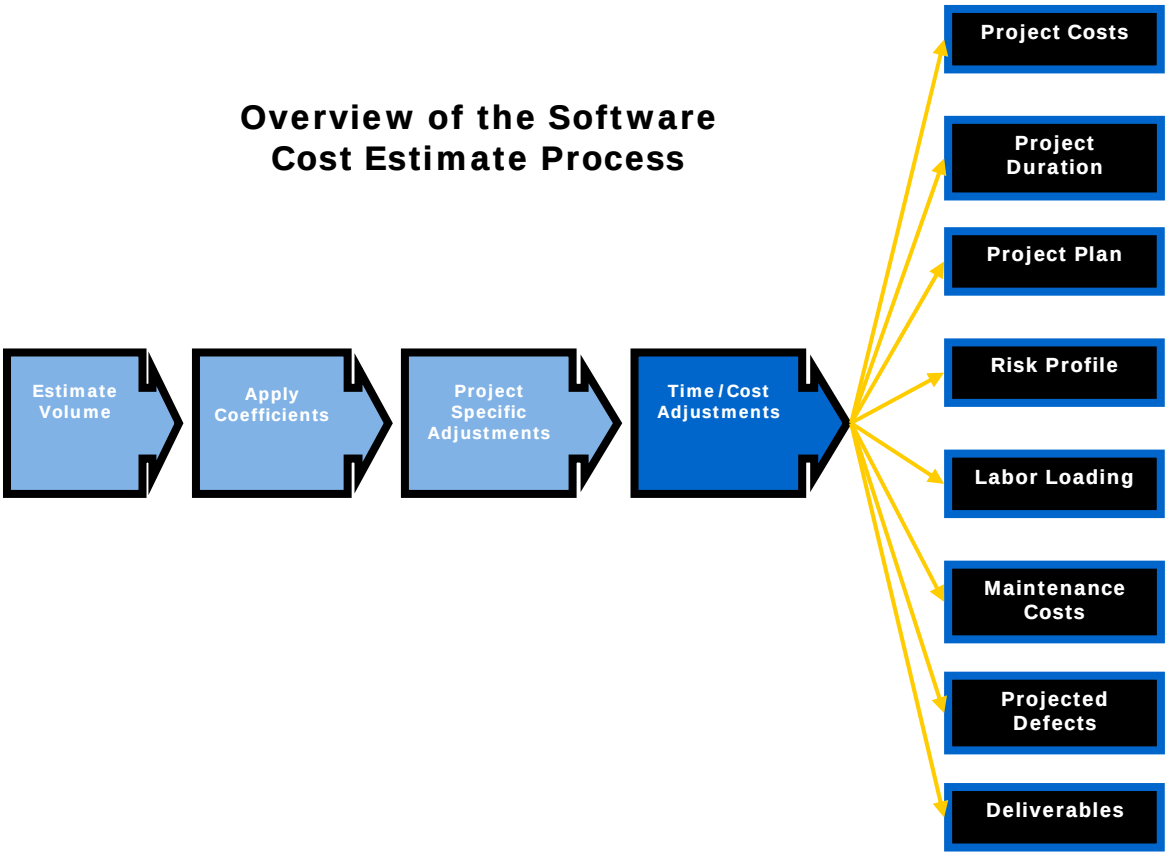
For this study we combined queries of proprietary databases containing industry data along with our own cost analysis models. The resultant models incorporated over 70 parametric sub-models to accurately predict software project data. Empirical results using the models demonstrate an accuracy within plus or minus 7% of actual project data, and in this white paper differences in project data greater than 3.7 percent may be considered statistically significant.

Empirical results using the models demonstrate an accuracy within plus or minus 7% of actual project data, and in this white paper differences in project data greater than 3.7 percent may be considered statistically significant.

Parametric Modeling uses roughly 100 input parameters that define a project and over 70 models of project behavior to forecast the project outcomes. This technique is often used in estimating software project costs and schedules and in preparing optimum project plans. A free sample tool can be downloaded from www.costxpert.com.

Parametric modeling of software cost, schedule, and so on has been in existence since the early 1980s. It is employed by most major organizations to help estimate and manage software development projects. The Standish Group, Software Productivity Research, and others have found that the use of these techniques double the probability of projects reaching a successful conclusion. The parametric models used in this study have a design goal of a plus or minus 10% accuracy and are currently achieving an accuracy of plus or minus 7% in the field.

The models cover all software lifecycle phases through completion of user acceptance, including strategy, requirements, design, coding, and testing. As shown in the attached figure, from a high-level perspective the models begin with program volume (function points, lines-of-code, and so on), apply cost coefficients to calculate effort, adjust effort using project specific factors and the time-cost tradeoff, and then output costs, duration, a project plan, a risk profile, labor loading charts, maintenance projections, defect projections, and project deliverable descriptions with page counts.

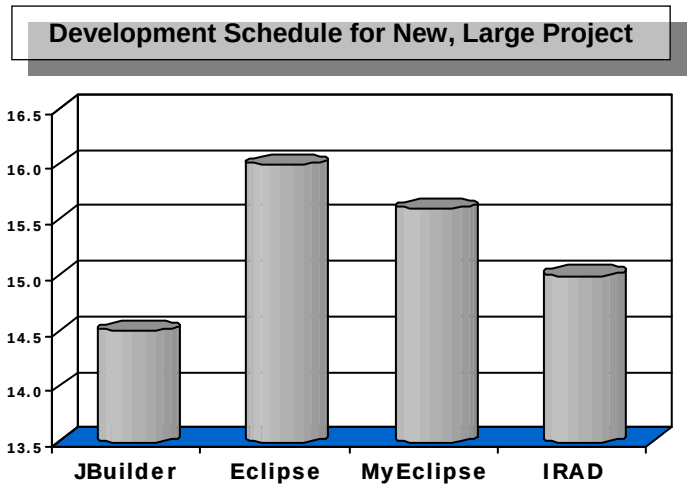


All study projects were defined based on a development team with 3 or more years experience in the application domain and 12 months experience with Java. For the initial analysis we assumed an average hourly rate of $100. We assumed a 55th percentile team in terms of skill. The development was assumed to be performed at a single site with a local area network. We assumed a language mix of 90% Java code and 10% SQL code.

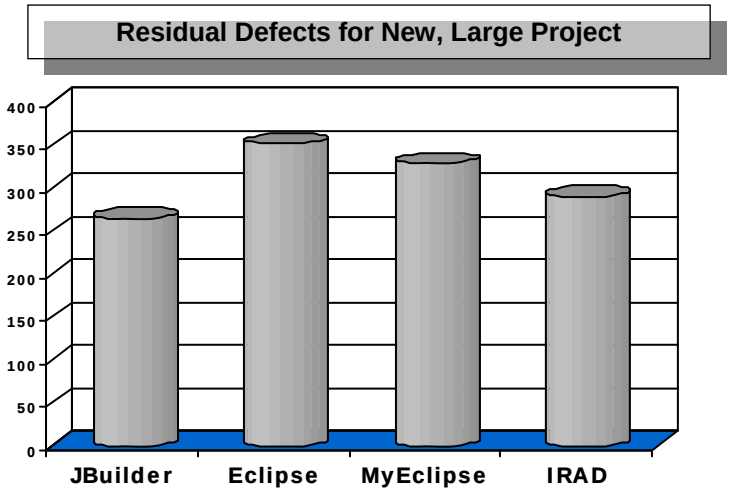
Appendix B

The Cost Xpert model evaluates development projects across a number of different dimensions. Figures presented in the main body of this paper represent total project costs, including (but not limited to) costs associated with development, residual defects, and maintenance.

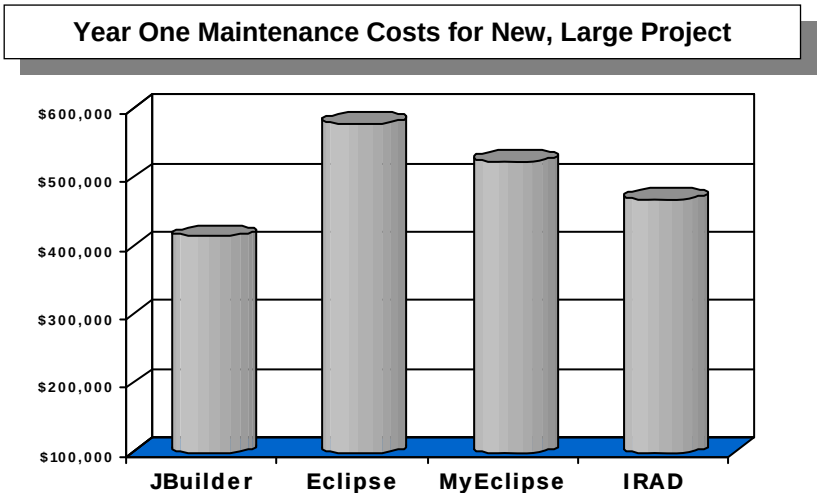
In order to provide the reader with more detailed analytical results, the graphs below map out the specific savings across each of these three dimensions for a new, large project. In each case, the JBuilder and IRAD products demonstrate a measurable advantage over the other Eclipse-based tools. These results quantify the advantage of JBuilder to provide a consistently more cost effective solution for Java developers.



Development Time (Calendar Months) for New 2,000 Function Enterprise Application



Year One Residual Defects for New 2,000 Function Point Enterprise Application



Year One Maintenance Costs for 2,000 Function Point Enterprise Application