Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

In collaboration with Pega Pegasystems Inc.

SmartBPM vs. Eclipse IDE

A study to compare the productivity of SmartBPM and Eclipse Java IDE

Table of Contents

1. Executive Summary 1

2. About the Study 3

3. Why Eclipse IDE? 5

4. Assumptions 7

5. Team Composition 8

6. Measuring Productivity 10

- 6.1. Productivity Metrics Comparison and Interpretation 11
- 6.2. Analysis and Design 12
- 6.3. Development 12
- 6.4. Information Model 14
- 6.5. Integration 15
- 6.6. Generating Reports 16
- 6.7. Testing 17
- 6.8. Build 17
- 6.9. Deployment 18
- 6.10. Business Changes and Reverting the Changes 19
- 6.11. Process Flow 21

7. Demonstrated Advantages of Pega SmartBPM® over Java Eclipse IDE 23

8. Conclusion 25

Appendix A – About Pegasystems Inc. 26

Appendix B – About Capgemini 27

Appendix C – Time Comparison Data 28

Appendix D – Applications Developed 30

1. Executive Summary

SmartBPM Suite®, a product of Pegasystems Inc., is a business process management (BPM) solution built on the PegaRULES® automated decision engine. SmartBPM provides process and practice rules in one integrated package, enabling users to develop, execute, manage, maintain, and grow their own decision-intensive BPM applications efficiently. The product is built in Java and uses an XML data structure, generates Java at runtime, and runs on leading J2EE application servers.

Capgemini was engaged by Pegasystems Inc. to conduct a study comparing SmartBPM with Java Eclipse IDE across the entire application development lifecycle, from Analysis and Design through Deployment and Maintenance.

Pegasystems Inc. asked Capgemini to design and develop applications for vehicle-insurance quote generation and homeowner-quote request, once with SmartBPM and once with Eclipse IDE. The focus of this study was to measure the productivity gains from SmartBPM versus those from Eclipse Java IDE during development of these applications.

In this study, per Pegasystems' request, Capgemini used two independent teams made up of experienced professionals with appropriate skill sets for the two development streams. These teams were geographically spread out as in a typical onsite/offshore model to factor in real-life development realities and issues.

Capgemini measured both teams' productivity for different phases in the development cycle:

- Analysis and Design
- Coding/Development (UI and Business Logic)
- Information Model
- Integration
- Generating Reports
- Testing
- Deployment and Maintenance/Enhancements

In all these phases, SmartBPM showed higher productivity than Eclipse IDE to varying degrees.

The results have shown that application development using SmartBPM is **five and a half times faster** than with Eclipse IDE. This figure rises to **more than six times faster** if Analysis and Design activities are excluded.

The highest gain in productivity using SmartBPM was shown when Capgemini's team incorporated business changes into the application: SmartBPM was then about **seven times faste**r than Eclipse IDE.

While the reasons for the above-mentioned results are many, the key factors observed were:

- SmartBPM's ease of use, arising from Web-based development from a standard browser
- Minimal coding requirements due to built-in process and business-rules functions
- Availability of out-of-the-box tools for tracing and debugging
- Ease of rules versioning; built-in business-process flows and rules
- Object-oriented rules base, enabling reuse of rules
- Automation of database connects; Web Services integration capabilities
- Ease of deployment; real-time changes to existing rules

SmartBPM's biggest asset is that it comes with many out-of-the-box functionalities/ wizards that speed the development process and drastically reduce the need for coding and testing.

This report details the methodology adopted, team composition/structure, and study results. We have interpreted the results and identified the key reasons behind the productivity gains for each phase.

2. About the Study

Pegasystems Inc. engaged Capgemini to carry out an objective study to compare the productivity of its SmartBPM suite with that of Eclipse IDE. Capgemini decided that a vehicle-insurance quote-generation application and a homeowner-quote generation application would be developed by two independent teams in parallel, using both SmartBPM and Eclipse IDE.

The study intended to cover the entire development cycle, from Analysis and Design through Deployment, followed by making business changes to the application. The study measures developer productivity in time taken to develop similar functionalities in the two streams.

Objective or Intent of the Study

This comparison applies two different software environments to identical business-process applications to:

- Evaluate the ease of building the application with SmartBPM, versus with the Eclipse Java IDE tool
- Evaluate the effort required to make business changes to the application, and the ease of rolling back those changes. This is an important aspect of this study, as one of SmartBPM's key value propositions is the ease with which its users can initiate and roll back business changes to respond to changing business needs.
- Measure developer productivity in the two environments, in the time and effort required to:
  - Build the application and debug, test, and deploy it
  - Change/update (maintain) the application once it is developed

Rationale

With development models changing/evolving and developers/stakeholders looking at value-add rather than pure cost, the realities of IT application development have changed. Newer development tools and methodologies have focused on value addition while at the same time decreasing costs and improving ROI. Increasing focus on business process management and improvement epitomizes the new realities of the industry.

We conducted the study to verify the effects of the following realities on developer productivity:

- Traditional development tools and methodology are changing.

- The emergence of the onsite/offshore delivery model—a team of resources working onsite with the customer to elicit requirements, architect the system, and coordinate and manage the project; while offshore teams design, develop, and test applications—has changed the equation. Our study has tried to factor in the effects this model has on productivity by developing the application in the context of this model.
- Programmers are looking to add value rather than cost as software develop-ment has become commoditized, and will continue on that path in future.
- Developing components that can be safely re-used is the key to ROI, but represents unique challenges.
- SmartBPM offers a new paradigm that implies business and IT working together. The developer dividend from this built-in agile approach is new orders of productivity.
- Industry analysts report that normal BPM delivers developer-productivity gains of 60–70%. This study looks at the effect that Pegasystems' SmartBPM has on development productivity.

3. Why Eclipse IDE?

There are many reasons Pegasystems Inc. chose to compare the productivity of SmartBPM developers with that of Eclipse Java IDE. Here is a summary of some of them:

- Java remains the most popular object-oriented programming language and platform. Some put the number of Java developers at as high as 10,000,000.
- Eclipse IDE is touted as the productivity tool for Java developers. Many third-party Java companies provide Eclipse plug-ins. This is the "standard" development tool for Java projects.
- Java is supported on multiple platforms: Windows, Solaris, Linux, z/OS, HP-UX, AIX, and others. It is therefore much more pervasive than single-platform languages such as C#.
- The Java EE (previously J2EE) application server is a robust architecture that supports reliable, high-performance enterprise applications. Java is the language of choice for large, high-performance, reliable enterprise applications.
- Java is in fact SmartBPM's target language. More specifically, the various rule forms in SmartBPM are translated into Java. The SmartBPM server engine runs in a JVM as a Java applications. This helps us elucidate the productivity gains from the SmartBPM technology.

In addition to core Java Eclipse IDE, as mentioned above there are some additional productivity tools either for rule engines or business process management that have been developed as Eclipse IDE plug-ins. These have not been included for several reasons:

- There are many—too many—such tools to choose from. Furthermore, there are no plug-ins that cover both the business-process and business-rules spaces. Pegasystems' SmartBPM covers both.
- The philosophies of all these tools are very similar—after all, they are Eclipse IDE plug-ins. A plug-in, of course, will provide productivity gains over core Eclipse IDE. However, the philosophy, development lifecycle, and implementation business change will be the same. These are all in the category of thick-client IT-focused technical platforms based on Eclipse, with similar characteristics.

- As mentioned above, no single tool will be able to handle all the features captured in the robust SmartBPM thin-client platform. Therefore, it will be very difficult to isolate in the productivity study the effort that must be exerted to align or integrate different meta-models from various plug-in versus that required to use the core SmartBPM functionality versus that required for the Java Eclipse IDE approach.

4. Assumptions

In executing the project, we made the following assumptions:

- It is not necessary to consider time spent by resources on training, as training costs are not normally recovered over a single project. Therefore, this study does not include the time required to learn SmartBPM or the Eclipse IDE tool in its productivity measurements.
- Developers are well-versed in both SmartBPM and Java, as well as in the Eclipse IDE tool set used to build the sample application.
- There is no need to consider time to install and configure the tools, since the associated times and costs are not recovered over a single project. Therefore, this study does not factor these times into developer productivity.
- Productivity measurements can be extrapolated to production applications.
- The time spent researching topics and solving problems related to development is not considered, as, again, these are not recovered over a single project, and are part of the development team's learning process.
- The Java team will not develop functionalities offered out of the box in SmartBPM, such as wizards.

5. Team Composition

The development team was composed of members across locations and time zones. This was done to mimic the onsite/offshore delivery model in order to help us factor in the following realities of the current development and delivery paradigm:

- The difficulties in translating requirements from onsite to offshore
- The difficulties in managing the project from the U.S. when the development team was based in India

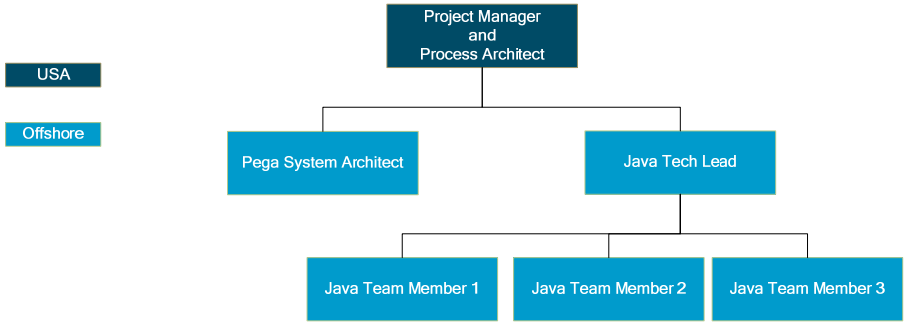
The time lag introduced in clarifying technical issues with the project manager and process architect in the U.S.

Not all the factors mentioned above affected productivity directly, but with the selected team structure, we attempted to factor these real-life issues into our study.

Project management and coordination was done from Chicago, Illinois. A Project Manager/Process Architect in Chicago facilitated interaction with key U.S. stakeholders at the client organization (Pegasystems Inc., in this case). The Project Manager worked with both the Java and SmartBPM teams, which were based out of Capgemini's offshore India development centre in Chennai, India.

The high-level team structure chosen for the study is illustrated below:

Exhibit 1: Team Structure



The project was executed from both locations, the United States and India. The activities and roles involved are given below:

Exhibit 2: Project Activities and Responsibilities

Location	Activities	Role/Ownership
United States	Project Management, Scope Management	Project Manager cum Process Architect
	Core Functional Requirements	Process Architect
	Test Plan	Process Architect
	Test Case review with SMEs	Process Architect
	Integration Testing	Tech Lead and Process Architect
	User Acceptance Testing	Process Architect
India Offshore Development Centre	Design and Architecture	Tech Architect and Tech Lead
	System Testing	Tech Lead and Developers
	Unit Testing	Developers
	System Testing	BA

6. Measuring Productivity

The time-measurement statistics for comparing the productivity of the two products had to be objective and done in a way that ensured we were measuring time for equivalent activities between the SmartBPM and Eclipse IDE streams—"apples and apples," versus "apples and oranges."

Some SmartBPM activities had a one-to-one correspondence with those in Java Eclipse IDE stream, and could be compared as is. However, some did not have a direct correspondent and needed to be classified in a broad manner for purposes of time measurement.

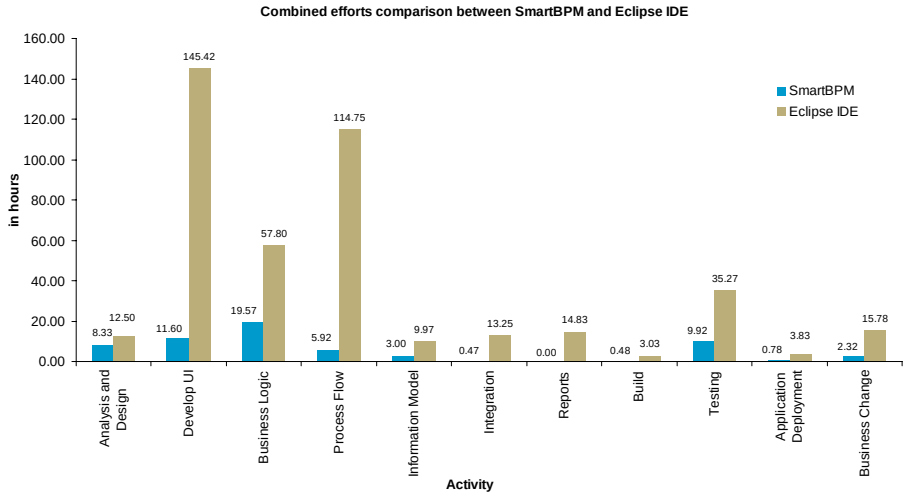
We also wanted to measure the productivity of SmartBPM's out-of-the-box features, like Reports and Integration capabilities, against that of Eclipse Java IDE's corresponding features. So, keeping all the aspects of measuring productivity in mind, we classified the activities into the following broad categories:

- Analysis and Design
- Coding/Development (further subdivided into UI and Business Logic)
- Information Model
- Integration Activities
- Report Generation
- Testing
- Build
- Deployment
- Business Changes

For each category above, we measured the time the two teams took to implement a given application functionality. We then rolled up the time taken for each task into an overall time for each category.

6.1. Productivity Metrics Comparison and Interpretation

Exhibit 3: Effort Metrics for SmartBPM and Eclipse IDE

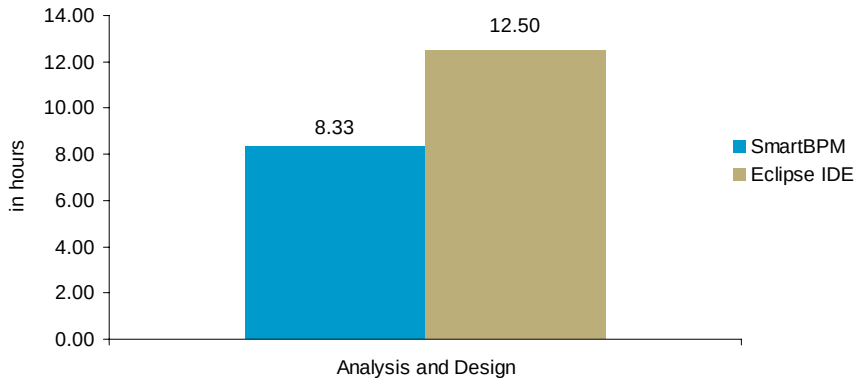


- The study has demonstrated that the entire application-development cycle was **5.5 times faster** with SmartBPM than with Eclipse Java IDE, whereas the Busi-ness Changes phase was **6.8 times faster** with SmartBPM than with Eclipse IDE.
- The analysis and design phase was **1.5 times faster** with SmartBPM than with Eclipse IDE.
- The UI-development phase has shown the highest productivity gain: UI development with Ajax functionality was **12.5 times faster** with SmartBPM than with Eclipse Java IDE. This was made possible by SmartBPM's built-in HTML editor.
- Business-logic development was **3 times faster** with SmartBPM than with Eclipse IDE.
- Developing the information model with SmartBPM was more than **3 times faster** than with Eclipse Java IDE.
- Integration activities showed that SmartBPM was **28 times faster** than Eclipse IDE.
- Testing the application using SmartBPM was **3.5 times faster** than with Eclipse Java IDE, while build and deployment were **6.3 times and 5 times faster,** respectively.

The results are discussed in detail in Section 6.2, with reasons for the above-mentioned productivity gains explained.

6.2. Analysis and Design

Exhibit 4: Metrics for Analysis and Design



In this phase, both development teams (SmartBPM and Eclipse IDE) analyzed the requirements by independently architecting/designing the system as per the best practices in their respective worlds.

The SmartBPM team took approximately 33% less time than the Eclipse Java IDE team to analyze and design the application. The productivity gain with SmartBPM can be attributed to the following factors:

- SmartBPM guardrails for design and development efforts offer best practices that speed up the design and development processes later on.
- Both business rule- and workflow-based development processes built on an easy and extensible framework
- Customizable process flows and re-usable rule base
- Features such as rules resolution and inference engine, which boost productivity

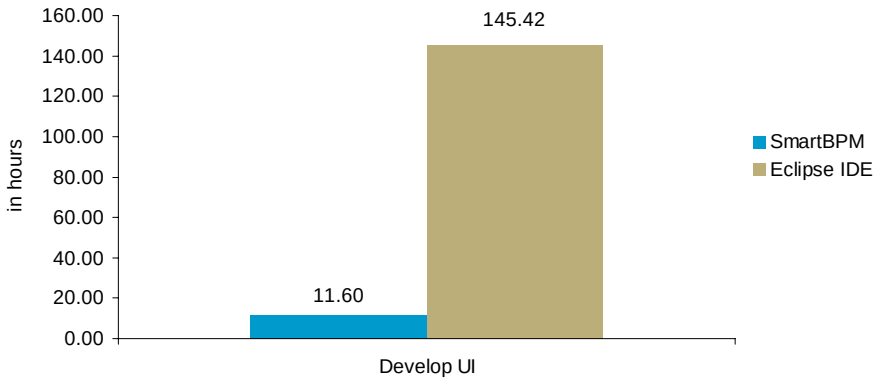
6.3. Development

This phase of the application development process consisted of building the user interface and the business logic of the customer service RMA application.

User Interface

We found that the UI—all screen layouts, including the Ajax functionality—could be developed in SmartBPM in 8% of the time it took to develop in Eclipse Java IDE. This is the highest SmartBPM vs. Eclipse Java IDE productivity gain achieved across all activities.

Exhibit 5: Metrics for User Interface Build and Enhancements



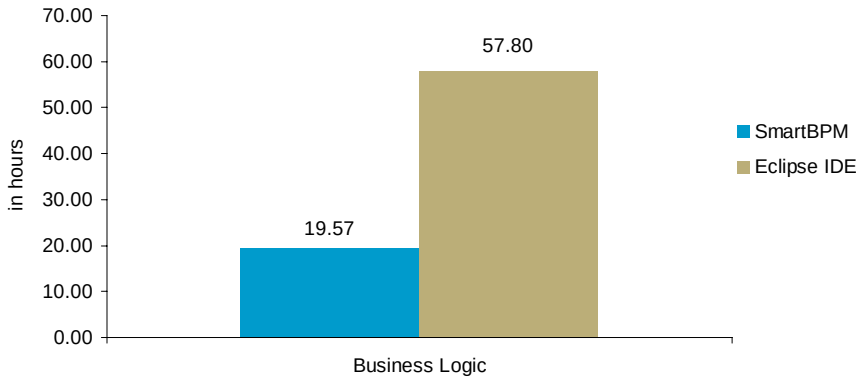
Developing the UI in SmartBPM is faster and better because:

- The new Ajax-based portal environment serves both process development and runtime delivery. The portal itself uses business rules to dynamically build the Ajax environment.
- The look and feel of the development environment has been updated and modernized. All six areas – Process, Decisions, User Interface, Integration, Reports, and Security –are under the control of the security.
- Control over the content of a pop-up in the user portal is defined declaratively using rules, and the corresponding Ajax code is generated and managed automatically.

Business Logic

The business logic forms the core of the application being developed. It includes the code for business rules, flows, and all processing logic for all the business functionalities.

Exhibit 6: Business Logic



Our study shows that the team using SmartBPM took one-third as long as the team working on Eclipse Java IDE to implement the functionality.

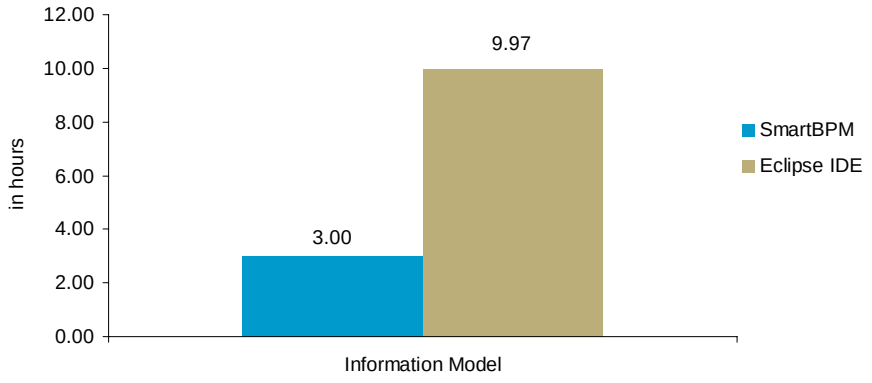
The primary reasons for this productivity gain are:

- Most of the functionality can be achieved with the OOTB activities.
- Processes and rules are dynamically assembled at runtime, based on the context of the case of work in hand.
- Supports for time- and constraint-based business logic
- Complex rules involving decision trees, constraints, "when" conditions, and decision maps are defined in the integrated development environment.
- RuleSets are developed, then used to support decision making, policy enforcement, and data transformation.

6.4. Information Model

The teams defined the entities, their properties, and relationships between them in this activity. They also designed the class structures and table structures for the data and properties that the application required.

Exhibit 7: Metrics for Developing Information Model for the Application and Subsequent Business Changes



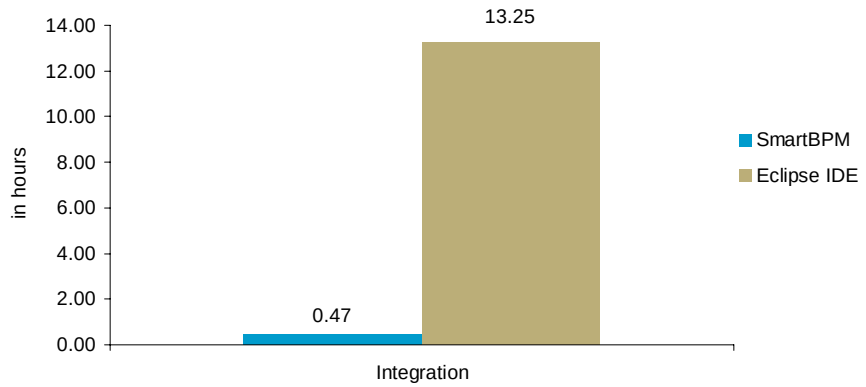
Creating the information model took about 3.5 times as long in Java Eclipse IDE versus SmartBPM. This translates into a productivity gain of 70% using SmartBPM. The reasons for this gain in developing the information model are:

- Class model consisting of a class base and operations to populate, use, and maintain it
- Easy-to-build graphical user interface helps speed up the creation of class base and definition and maintenance of properties.

6.5. Integration

SmartBPM provides wizards for Web Services development. In a few easy steps, a developer working in SmartBPM can use a Web Service without any configuration or settings.

Exhibit 8: Metrics for Integration Activity for the Application and Subsequent Business Changes



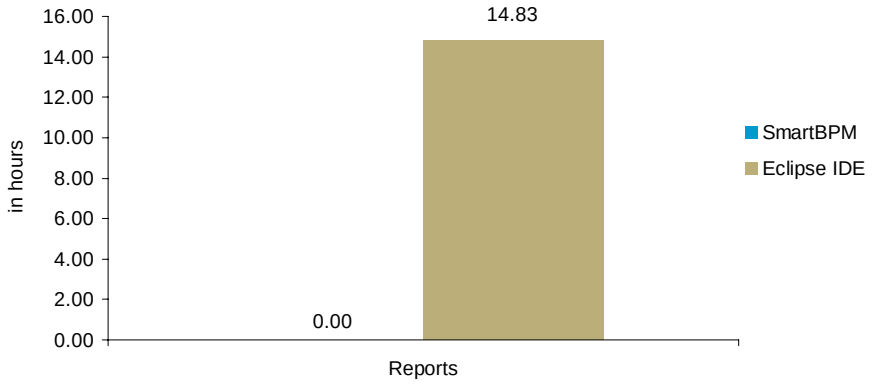
We found that it takes **two-thirds of the time** (a productivity gain of 66%) to set up and use Web Services with SmartBPM than with Eclipse Java IDE. Our tests revealed that it takes considerable experience and additional configuration time and effort when developing Web Services interfaces in custom code, as Eclipse Java IDE requires. SmartBPM also allows the developer to use other interfaces, such as EJB, SOAP, and IBM's MQSeries, with minimal development effort.

- SmartBPM Integration Services are based on industry-standard technologies for external system integration.
- Easy wizard-based steps and auto-generation of the necessary integration components
- Built-in test facility for the created activity
- SmartBPM interfaces support a wide range of technologies and standards, including HTML, HTTP, Simple Object Access Protocol (SOAP), Microsoft .NET, Sun Microsystems Enterprise JavaBeans. (EJB), IBM WebSphere MQ® messaging, Java Message Service (JMS), J2EE Connector Architecture (JCA), Business Process Execution Language (BPEL), JSR 94, JSR 168, and others. These interfaces are called **Integration Services.**

6.6. Generating Reports

We developed two reports—both available out-of-the-box in SmartBPM—using Eclipse Java IDE, and found that it took us close to 15 hours of development time to create these reports.

Exhibit 9: Metrics for Generating Reports



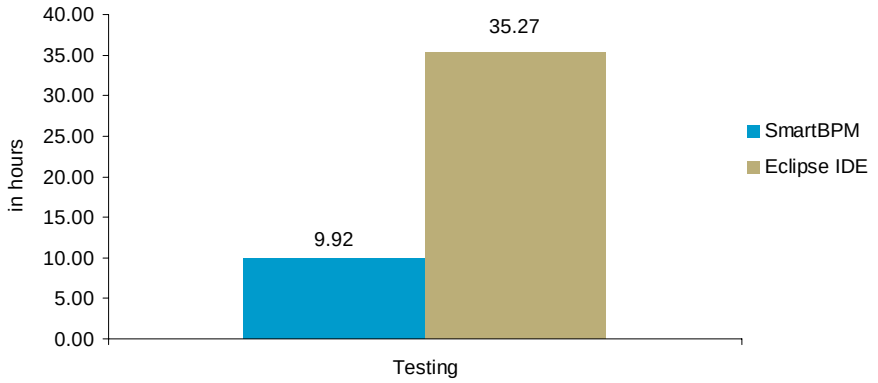
Some of the reasons SmartBPM's reporting features are more productive than Eclipse Java IDE's are:

- Easy customization
- Multiple OOTB reports with rich user-interface and graphical representation
- Report Wizard to create the report
- Rich features set:
  - When users change the data range with the **slider widget**, the slide effect clears the existing data from the report by sliding it down below the data axis and sliding the new range of data back up into position. You can use this effect for any type of chart, but it's most effective on line and area charts.
  - **Zoom** widens or contracts the focus on a segment of the data according to the position of the slider. Use this effect on any kind of chart except pie.
  - **Interpolation** inserts logically projected data points between the reported data points, creating the visual effect of data that flows horizontally across the range of data according to the slider. Use this effect on pie, line, and area charts.

6.7. Testing

We unit-tested all individual interfaces, as well as all function modules. We used the built-in debugger within SmartBPM, as well as its Clipboard facility, for detailed analysis during testing and debugging.

Exhibit 10: Metrics for Testing the Application and Business Changes



SmartBPM provides several debugging and system-monitoring tools that can be used to test applications: the Tracer, the Clipboard, the Performance Tool, the System Management application, and the log files. These tools expedite the testing process by offering an easy-to-use interface and aid detailed analyses. SmartBPM offers an immediate testing and trace mechanism for the business rules created.

We found that SmartBPM was **3.5 times as fast** as Eclipse IDE for testing and debugging the application.

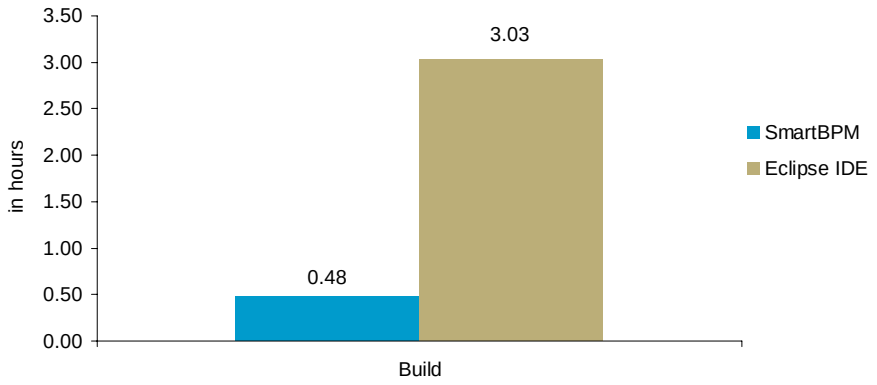
6.8. Build

The application build in Eclipse Java IDE took approximately 6.3 times as long as it did in SmartBPM, which translates into a productivity gain of 85%.

The primary reasons for this gain in productivity are:

- Automatic recognition of the change
- No need for separate build process/scripting
- Change takes immediate effect for testing

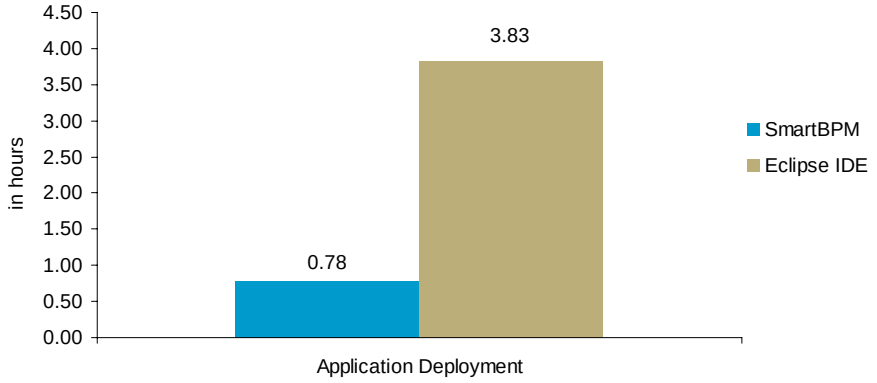
Exhibit 11: Metrics for Application and Business Changes Build



6.9. Deployment

SmartBPM provides a method to export complete applications from the system, then import them into another SmartBPM system. It uses a Move Rules facility, which downloads the application rules (in the form of a "RuleSet"), associated data, and required class structure to a .zip file. It then uses a Rule Import facility to deliver these applications to the new SmartBPM system.

Exhibit 12: Metrics for Application and Business Change Deployment



RuleSet versioning is an easy technique with which to test the newly deployed applications. During our tests, we granted a set of pilot users access to the new application by defining the RuleSet name and application to which they were to have access. When these users next logged onto the SmartBPM system, they were able to run the newly deployed application. We did not need to take the SmartBPM system down to develop the new application.

SmartBPM's Auto Refresh after Deployment is a key feature that increases application availability. The Real-Time Rule Update feature enables rapid changes in real time. By comparison, the deployment process for the Eclipse Java IDE tool set is far more complex and time-consuming, as it involves configuring a set of XML files through manual processes or wizards. SmartBPM was approximately **5 times as fast** as Eclipse IDE.

6.10. Business Changes and Reverting the Changes

One of the most important values that SmartBPM brings to the table is the ease with which users can make changes to an existing application. This is an important dimension because it gives business users the flexibility to initiate and roll out changes to the application to meet changing business requirements.

Exhibit 13: Metrics for Business Change

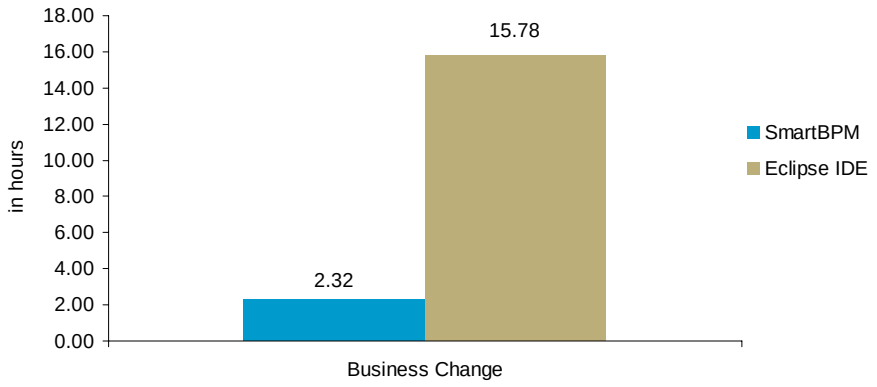


Exhibit 14: Metrics for Business Logic Change

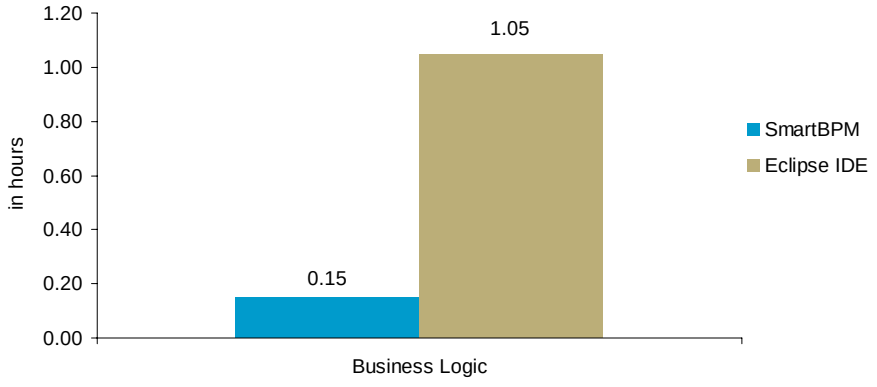


Exhibit 15: Metrics for UI Changes

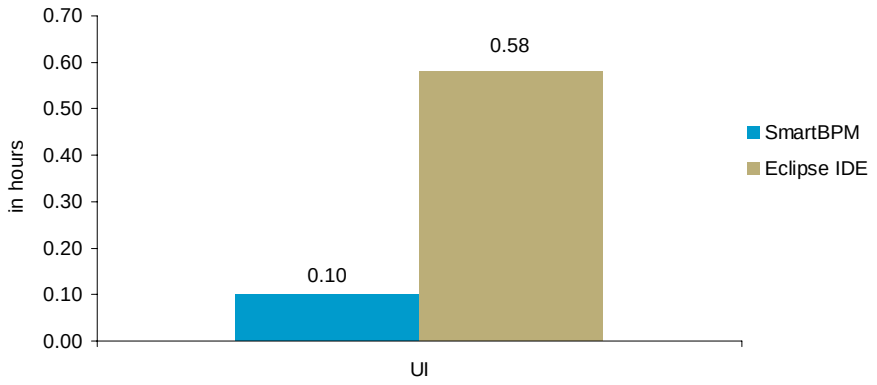


Exhibit 16: Metrics for Information Model Changes

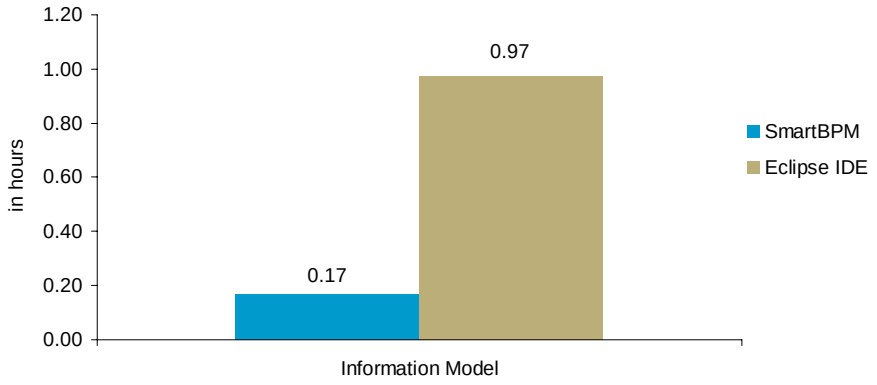
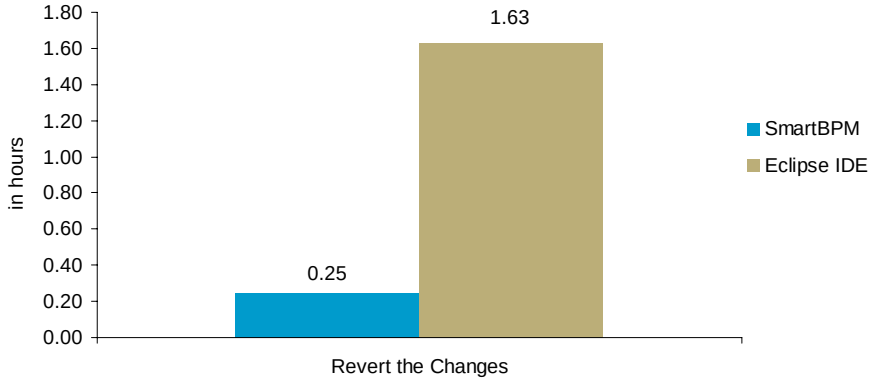


Exhibit 17: Metrics for Reverting Changes



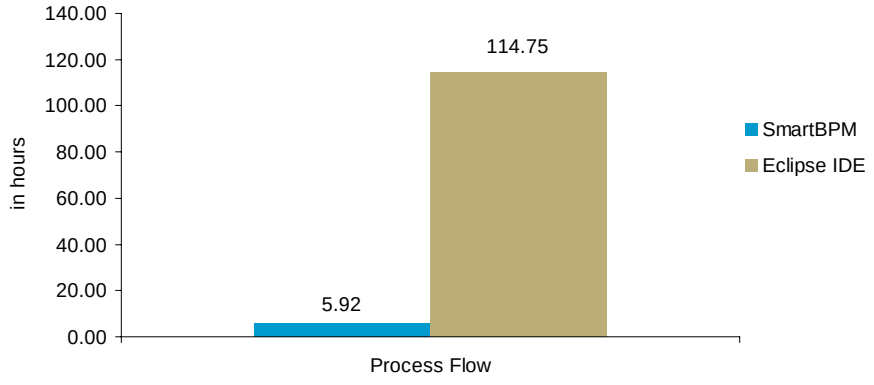
Business change, which involves changing an application after it is built, is a major challenge; organizations consume significant time and resources in this effort. Therefore, one of the critical factors in evaluating any BPM development environment is how easy it is to change/update the application once it is in production.

In order to evaluate this capability, we conducted a number of tests that involved changing the system interfaces and the business processes, such as adding a new process, changing an existing process, and discontinuing an existing process. In the Eclipse Java IDE environment, such modifications require code changes, database changes, and recompilation. By contrast, we found modifications to be very easy with SmartBPM. All that was needed was to make changes to the Visio flow diagrams, which automatically drove changes to the rules in SmartBPM.

In fact, the ease of changing and updating the application makes it possible for business analysts/users to make changes without directly involving IT resources. We find this capability to be the core strength of SmartBPM. Making business changes with SmartBPM was **6.8 times as fast** as with Eclipse IDE.

6.11. Process Flow

Exhibit 18: Estimates for Process Flow Development



SmartBPM has a visual interface to design custom flows, as well as many out-of-the-box flows that can be customized to meet the needs of the business process being modeled. While the actual development of a process-flow interface was out of the scope of the study, we have estimated how much time it would have taken to develop a similar interface, then the process flows, in Eclipse Java IDE.

This estimate includes the effort it would have taken to:

- Build the flow
- Connect the flow elements
- Connect rules/business logic to flow elements
- Assign the UI components to flow elements
- Execute the flow
- Build the ”Where am I?” feature into the flows involved in the application built

The estimated productivity gain, as seen Exhibit 13 above, is close to 95% using SmartBPM, with SmartBPM estimated to be approximately **20 times as fast** as Eclipse IDE. The reasons for this productivity gain are:

- Easy-to-use visual process-flow modeling tool
- Out-of-the-box features available in SmartBPM, such as ready-built flows that can be easily customized for a business process
- Integration with the engine that executes the flow
- UI components assigned to flow elements that are understood by the engine

7. Demonstrated Advantages of Pega SmartBPM over Java Eclipse IDE

Ease of Use

- Automatic incorporation of user-defined policy changes into existing procedures
- Autonomic diagnostics
- Intuitive Web-based interface for managing tasks, viewing processes, and accessing the Dashboard
- Online management of personal task list, which shows unassigned tasks which the user is eligible to take on
- Easily view, reassign, and escalate assigned tasks
- Interactive, graphical process image displays process details.
- Various tools support tracing, debugging, history, and performance.
- Control versions to seamlessly upgrade or roll back to previous versions
- Brings BPM to the global workforce
- Extensive library of enterprise connectors and adapters

Developer Productivity

- Intuitive user interface with drag-and-drop capabilities
- View the process graphically while developers create and monitor it
- Does not require large-footprint development tools
- Built-in processes and rules for faster development
- Collaborative development and administrative environment
- Support for various services and connectors
- Interfaces to MS Word, PDF, MS Excel, and MS Visio
- Service-oriented architecture (SOA) maximizes integration capability using wizards, as opposed to coding.
- Interactive business visualization and monitoring
- Complex business-logic support
- SOA integration testing, AJAX automation; faster time to solution

- Unique pre-flight capability enhanced with rule-referential integrity checking to develop better applications more quickly
- Built-in and auto-generated AJAX for richer end-user application
- Highly graphical and rapid solution-development environment, execution engine, and management dashboard

Deployment Features

- Internationalization capability for global organizations to develop once and deploy everywhere
- Integration of multiple image repositories, document- management systems, and content-management systems to manage global policies and processes for record retrieval and retention
- Distribute work based on process parameters such as performer skills, performer availability, performer workload, work-item priority, etc.
- Update or roll back changes to production with advanced deployment support through RuleSet management and versioning

Application Changes and Updates

- Model hierarchical business roles for automatic task updates
- Real-time updates for the processes
- Rapidly deploy and update solutions in response to changing circumstances
- Online analytic tools to continuously improve processes
- Real-time changes to business logic via business rules
- Changes to business-process flows via changes to Visio flow diagrams in real time
- No database-schema changes required to add properties or objects
- Support for SOAP, .NET, JCA, IBM's MQ Series, COM, and XML

Security

- Granular security model defines privileges by business role.
- Role-based security
- Integration with LDAP and NT authentication mechanisms
- Online user profiles allow users to manage their own preferences, such as email address, notification methods, and availability.

8. Conclusion

From our comparative study, we have observed that:

- Development is **5.5 times faster** using SmartBPM than Eclipse IDE.
- If we exclude Analysis and Design activities, development using SmartBPM is **6.2 times faster** than with Eclipse IDE.
- Making business changes using SmartBPM is **6.8 times faster** than Eclipse IDE.
- Integration activity using SmartBPM is **28 times faster** than with Eclipse IDE.
- Application build using SmartBPM is **6.3 times faster** than with Eclipse IDE.
- Deployment using SmartBPM is **5 times faster** than with Eclipse IDE.

The key finding was the ease and efficiency with which business changes were incorporated into the application and rolled out to users. This is a key parameter where SmartBPM scores over Eclipse IDE because once built, applications must be flexible to address changing business needs.

We recognize that actual production applications and scenarios will be far more complex than the one developed for our evaluation purposes. However, based on our technical and application-development experience, it is our professional opinion that enterprises will reap significant productivity improvements by using SmartBPM to build their decision-intensive business process management applications. Productivity will increase even more dramatically when SmartBPM is used in an environment that requires frequent real-time changes to rules and process flows. This is often the case in complex BPM scenarios.

The table summarizes the key differentiators between SmartBPM and Eclipse Java IDE.

Exhibit 19: Comparison of SmartBPM and Eclipse IDE

SmartBPM	Eclipse IDE
▪ OO Rules-based development	▪ OO-based library development
▪ Business process modeling	▪ No process modeling
▪ Pre-built rules and business processes	▪ No such capability beyond standard libraries
▪ Automated database connect functions and updates	▪ Manual connect set-up process
▪ Changes to rules and processes in real time and at runtime	▪ Changes to rules and processes difficult and time-consuming
▪ Web-based team development environment	▪ Web-based IDE not available

Appendix A – About Pegasystems Inc.

Pegasystems Inc. (NASDAQ: PEGA), the leader in Business Process Management, provides software to capture, automate, and change the policies and procedures that drive the world's largest organizations.

Pegasystems' SmartBPM® Business Process Management technology makes enterprise applications easy to build, use, and optimize. By automating policy manuals, system specifications, and manual coding; and making business solutions dynamically context-sensitive, Pegasystems powers the world's most sophisticated organizations to Build for Change®.

Pegasystems' award-winning, standards-based BPM suite is complemented with best-practice solution frameworks to help leaders in the financial services, insurance, healthcare, manufacturing, and government markets drive growth and productivity and achieve compliance.

Headquartered in Cambridge, MA, Pegasystems has regional offices in North America, Europe, and the Pacific Rim. For more information, visit **www.pega.com**.

Appendix B – About Capgemini

Capgemini, one of the world's foremost providers of consulting, technology and outsourcing services, enables its clients to transform and perform through technologies. Capgemini provides its clients with insights and capabilities that boost their freedom to achieve superior results through a unique way of working—the Collaborative Business Experience—and through a global delivery model called Rightshore®, which aims to offer the right resources in the right location at competitive cost. Present in 36 countries, Capgemini reported 2007 global revenues of EUR 8.7 billion and employs over 88,000 people worldwide.

We bring deep industry experience, enhanced service offerings and next generation global delivery to serve the financial services industry. With a network of 15,000 professionals serving over 900 clients worldwide, we move businesses forward with leading services and best practices in banking, insurance, capital markets and investments.

We leverage Centers of Excellence in banking, insurance, capital markets, compliance & risk management, payments, wealth management, technology services and outsourcing to consistently deliver leading solutions. Our global Centers of Excellence capture industry insights, best practices and the latest trends in techniques, tools and technology to continually upgrade solutions, help service new and existing clients and provide visionary yet practical thought leadership.

From industry trends to best practices, global or local, Capgemini Financial Services leads with four world reports: the *World Retail Banking Report*, *World Wealth Report*, *World Payments Report* and *World Insurance Report*. Featuring the latest in industry innovations and insights, Capgemini's reports help organizations be more competitive and responsive to market demands.

For more information please visit **www.capgemini.com/financialservices**.

Appendix C – Time Comparison

Exhibit 20: Time Comparison (in minutes)		
Task Name	**SmartBPM Stream**	**Eclipse IDE Stream**
Analysis and Design	500	750
Develop UI	696	8725
Business Logic	1174	3468
Information Model	180	598
Integration	29	795
Reports	0	890
Build	29	182
Testing	595	2116
Deployment	47	230
Revert the Changes	15	98
Total effort in hours	54.40	297.53

Exhibit 21: Application Build: SmartBPM Stream Effort Distribution

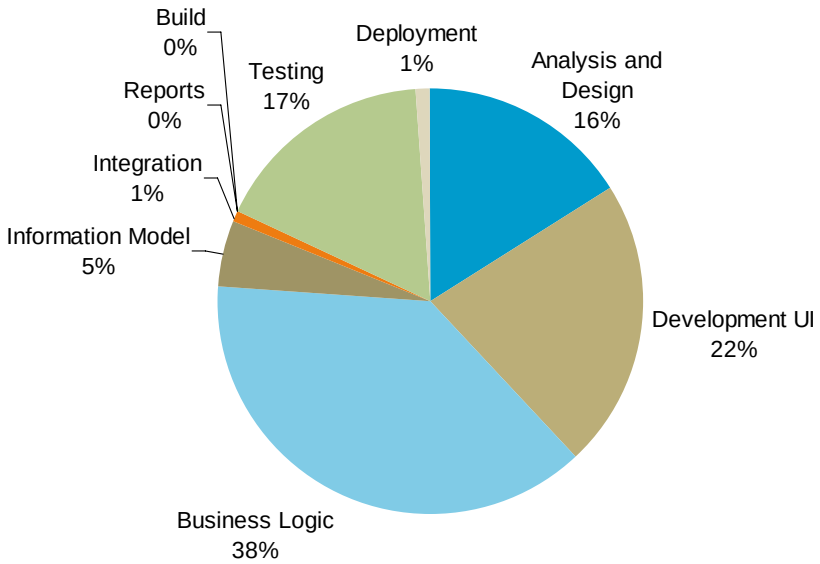
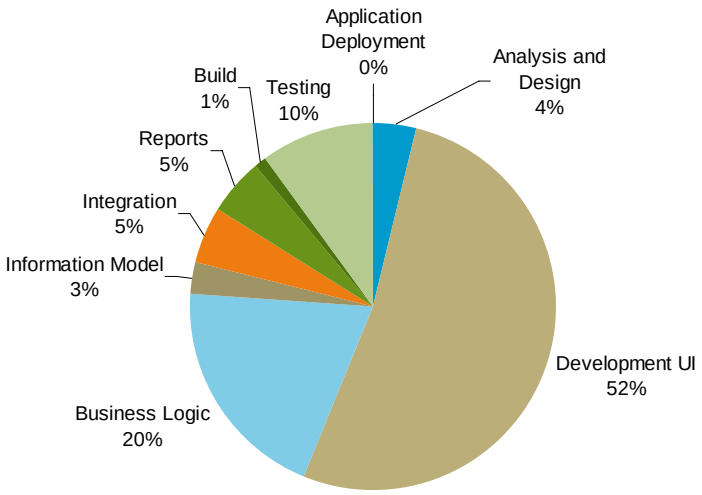


Exhibit 22: Application Build: Eclipse IDE Stream Effort Distribution



Appendix D – Applications Developed

After careful consideration, Pegasystems selected the Userv insurance application for the developer-productivity case study. There were three reasons for this selection:

- The process was representative of what we see in the field.
- The process could be implemented in the budgeted time.
- The application included SmartBPM features, demonstrating its power.

The Process was Representative

The Userv application covers a new business process. Though the application is insurance-specific, the high-level process is common across virtually all of Pegasystems' verticals.

Implementation Time was a Good Fit

The original requirements for the application were an amalgam of a number of insurance and financial services projects involving writing new business. In addition, the specific features were chosen so that the entire project could be completed in a reasonable amount of time, which the productivity study required.

Product Features Demonstrated the Value of SmartBPM

The features used in the application were good examples of SmartBPM's out-of-the-box capabilities, particularly those with very high utility for developers.

Description of the Application Developed

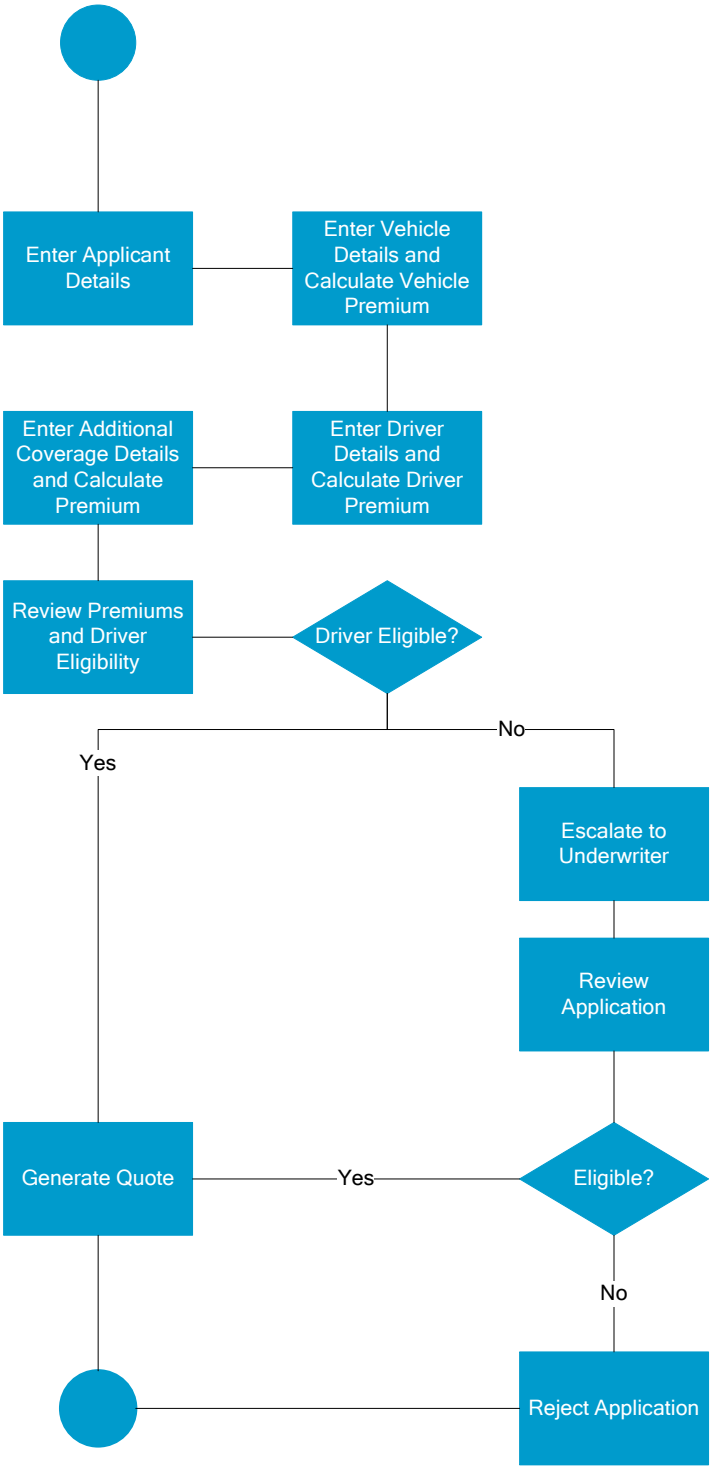
For the study, we developed two applications using SmartBPM and Eclipse IDE. The details of the application are given in the following exhibit.

Vehicle-Insurance Quote Generation

This application simulates a request for and generation of a vehicle-insurance quote. The application starts with the data-entry operator providing the applicant information on a screen, and retrieving the relationship and balance history from the database.

After the information is retrieved, if the applicant details are valid, the operator moves to the Vehicle Details screen and enters the details for the vehicles for which quotes are to be generated. The application calculates the premium based on the business rules associated with each vehicle.

Exhibit 23: Vehicle-Insurance Quote Generation



The operator moves to the Driver Details and Driver Premium screen, where he or she enters the required details of the Driver(s). The application calculates the driver premium, again in accordance with the defined business rules.

The operator moves to the Additional Coverage screen, where the details for Uninsured Motorist Coverage and Bodily Injury Coverage can be entered. Based on the coverage amounts entered, the application calculates the appropriate premium amounts for these additional coverages. The operator then goes to the Review Premium screen and can review the premium amount calculated by the system using the business rules defined. The application calculates the policy decision and displays it on the Review Premium screen.

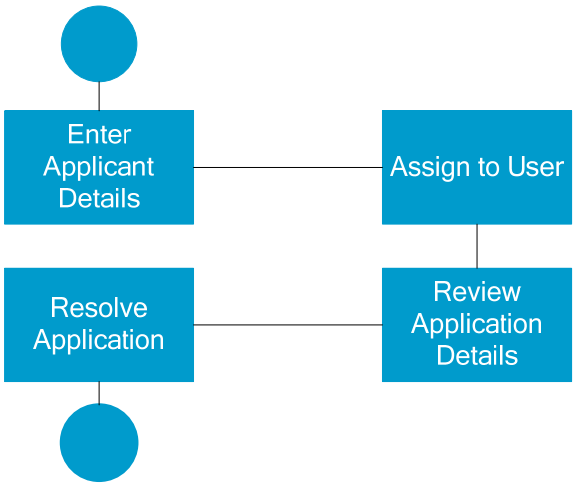
If the Policy Decision is "Eligible," the operator generates the quote; and a letter, with the premium quote, is mailed to the applicant's email ID in PDF format. If the policy decision is "Not Eligible," the operator assigns the application to "Underwriter" for review.

The underwriter reviews the quote and either rejects it, in which case the rejection note is sent to the applicant; or approves it, in which case the quote in PDF format is sent to the applicant's email ID.

Homeowner-Quote Application

This application simulates a simple process flow in which the operator creates a quote request for a homeowner in the application. The request is then assigned to another operator, who picks the item from his or her work queue and resolves or completes it.

Exhibit 24: Homeowner-Quote Application



Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

www.capgemini.com/financialservices

Capgemini Financial Services
6400 Shafer Court
Rosemont, Illinois 60018
Phone (847) 384 6100, Fax: (847) 384 0500