Process Automation Solutions for Software Development

The BuildForge Solution

By Doug Fierro, Director of Product Management

March 2006

Introduction

The BuildForge® software is a process automation system that can make it possible to centralize, automate, and accelerate your software development while leveraging your investment in existing tools. The system can allow you to connect multiple applications and hardware platforms to enable global reporting, centralized tracking, and distributed access to hardware resources. This paper describes the system's approach to the problems of integrating processes and systems, then details how the system can provide significant improvements in development team productivity, product quality, and regulatory compliance.

Integrate and Track your Progress

The BuildForge system offers advantages for bringing together procedures, people, and tools into cohesive and efficient systems. It addresses three key development challenges: application lifecycle management, tool standardization, and compliance.

The Challenge of Application Lifecycle Management (ALM)

Getting a product from initial coding all the way into production involves a complex network of people, processes, and technologies that need to be integrated. Typically, development teams face some common challenges:

- The groups involved in critical phases—development, configuration management, QA, release, and customer support—are separated by organizational boundaries, disconnected toolsets, or vast geographies.
- Each team has its own processes, which are often manual and rarely documented.
- Essential tools such as bug-tracking databases and source code control systems are often disconnected and contain silos of critical information, but making them work with each other is difficult and time-consuming. As a result, the organization never fully knows what was delivered to the customer...until it breaks in production. To cope with ever-increasing demands for high-quality

products at more frequent intervals, development teams need a solid foundation of repeatability, reliability, and tracking. Communication and integration throughout the development lifecycle is essential, but mandating consistency across teams can create political turf wars and these initiatives often fail. The BuildForge system offers a powerful alternative by providing the ability to automate, integrate, and report on any tool in your organization's development environment.

Solution for the Standardization Problem

Many organizations standardize on a set of tools or a product suite to reduce complexity and improve communications and productivity. But often, an event occurs that requires a transition to the standard set of tools such as the acquisition of another company, or outsourcing part of a project to an organization that uses a different set of tools. Ultimately, development teams need a flexible architecture that allows them to support heterogeneous toolsets during these types of transitions, and that helps them to adapt to future development requirements as required.

The BuildForge system leaps the hurdles involved in setting up an integrated development ecosystem: it is designed to work with all of your current tools and to allow you to add new ones to your environment over time. It provides a consistent interface to functionality across operating systems and hardware. With it, you can launch a Microsoft® Windows® program or Unix command from the same interface...or start a process that runs tasks on several Macintosh, Microsoft Windows, and Linux® machines simultaneously.

Enabling Auditing and Compliance Management

More and more organizations are encountering the need to prove that their processes comply with government mandates. To pass an audit, development teams must demonstrate that they have repeatable end-to-end development processes that have sufficient access controls and consistently document why systems changed and who changed them. Without an automated ALM system, teams can spend countless hours aggregating data from each of their applications to provide the complete development view to auditors.

The BuildForge system can provide a comprehensive, self-documenting record of your development processes so you can be ready for an audit at a moment’s notice. As a bi-product of executing your daily development activities, the software captures critical data about exactly what was released, what changed (by whom and why), and what tests were performed to provide a detailed Bill of Materials that may be used as evidence for compliance and auditing tasks.

How the System Works

At its core, the BuildForge system is a development automation and process management solution. Think of BuildForge as a database of processes that can be run, tracked, scheduled, and distributed to various computers. This section describes the scope of development automation that BuildForge provides, and explains how the various pieces of the system work together.

Defining and Running Processes

Within the BuildForge system, you define a process as a series of tasks called a project. Each task within a project contains a set of command lines and environment variables which can be passed to a computer on your network for execution. Once they are defined, the system can run projects at scheduled times or when an appropriate user launches them. After a project starts, the system runs the tasks on one or several servers, and records the results in its database. Tasks may launch other projects, and finish by notifying appropriate members of your organization as to the success or failure of a project. A project's activities can involve the compilation of a software executable, but the general nature of the BuildForge system means that they can include much more. Projects can involve a comprehensive development process, extending from source checkout all the way through the build, test, and deployment phases. A typical project might do all of the following tasks:

1. Check out a set of source code files
2. Compile the code, reporting on progress along the way
3. Run automated unit tests against successful compiles
4. Create an installer
5. Publish the installer to a download site, and notify teams that the installer is available
6. Run the installer to create an installed executable
7. Run automated tests against the executable
8. Report the results of the tests
9. Launch a subordinate project to update standard libraries
10. Promote executables and other files to QA for further testing
11. Deploy finished releases to production environments, such as web servers or CD manufacturing.

Many other types of projects and activities are possible. With BuildForge you can:

- Create PDF files from documentation source files, copy them to a download location on your web site, update the web page listing the files, and update the web site search index.

- Allow individual developers to launch builds of individual components, and run a master build daily to compile the entire product, calling all of the individual builds.

- Run the same process on five or twenty different computers...serially or in parallel.

- Allow a project team to run its own builds, while restricting web server updates to the IT team and the marketing team (for example).

Products in the BuildForge System

The heart of the BuildForge system is BuildForge FullControl™, which provides the comprehensive process automation, distributed server access, and process tracking. For additional power, add BuildForge FullThrottle™ to provide process threading, server pooling and extended reporting.

How the System's Components Work

The BuildForge system started with a simple concept – to provide development teams with a reliable process engine without mandating the use of specific tools. That idea has emerged into a flexible and robust automation infrastructure that is used by distributed development teams around the world. At its core is the ability to run arbitrary command lines on various machines from a web-based interface, regardless of the target machine's location or operating system. The system contains the following components:

- The *Management Console* provides a user interface to the system; it is a web-based PHP application that runs on an Apache HTTP server. Through it, you can organize commands into projects, and manage the server resources and environment variables those commands need.

- The *BuildForge Engine* uses information entered via the Management Console and stored in the database to communicate with Agents, execute project tasks, and perform notifications; it acts on instructions from the Management Console.

- The *Database* stores comprehensive project information and tracks each run of a project. User and system actions are stored in it, so that auditing and reporting data can be extracted and analyzed.

- Individual computers run *Agents* compiled for their operating systems, allowing them to respond to instructions from the system. Agents construct appropriate environments for projects dynamically, drawing information from the database so that each command executes in an environment fully configured for its needs.

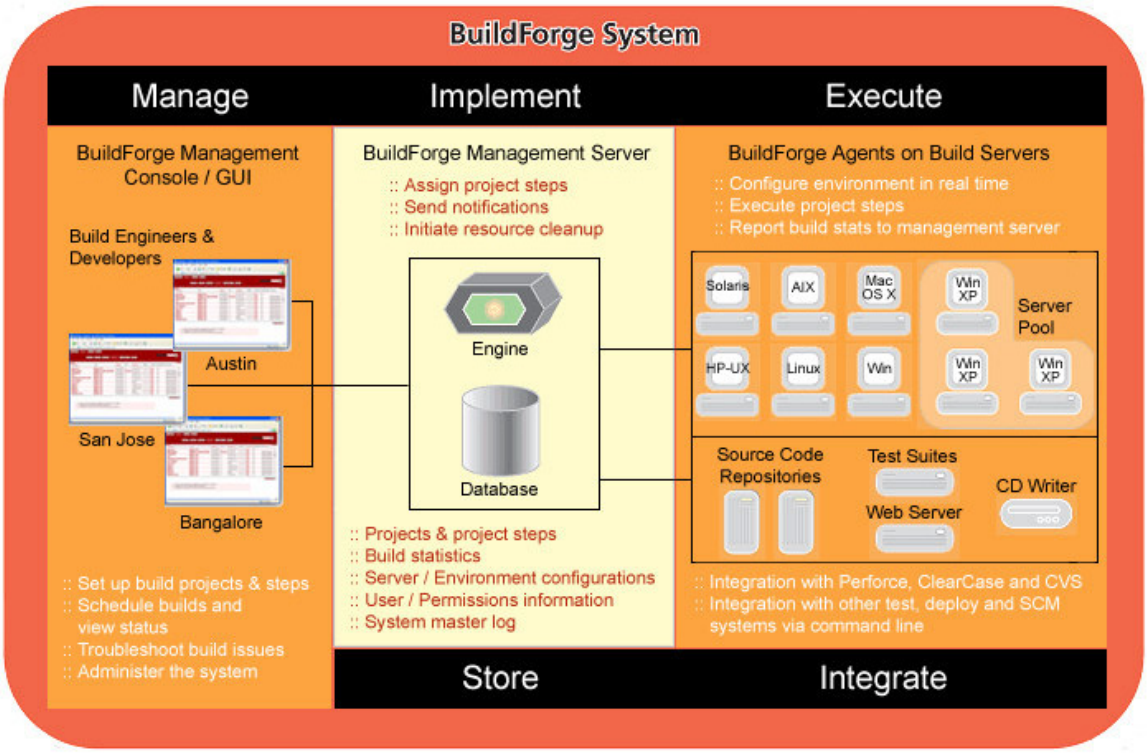

Figure 2: BuildForge system components

Some Examples

With these components in place, the system acts as a framework that ties your development resources together. Consider the following development scenarios that can be automated and tracked with BuildForge:

- A configuration manager uses the Management Console to schedule a daily run of a process that compiles the most recent development version of your flagship product for multiple operating systems. The system knows each step required to complete the project, and it creates the appropriate environment at run-time based on your specifications.

  After running some common steps, the project forks into two subprocesses, one for Red Hat Linux and one for Microsoft® Windows®, which each check out their own source code and start compiling the product in parallel. Throughout the entire process, the CM team has a real-time view of the project's progress. If one of these processes fails, the subprocess notifies the developer that was responsible for the work. Once the project is successfully completed, the finished executables are posted to an internal web server for testing and QA is automatically notified.

- A developer launches an interim build of the product she is working on. Unbeknownst to the developer, the machine she normally uses is down. The system automatically shifts the process to a different server and notes the actual server used. The developer receives an e-mail message when the process completes, noting the location of the most recent build artifacts. Neither the developer nor the configuration management team has to respond immediately to the outage.

- A webmaster creates a process that collects updated web page files from a staging server and copies them to appropriate locations on the public server several times a day. Through a single

automated process, he is able to update the files, restart the web server, and archive the old versions of the files to another machine.

- A project team needs to stress test their product using many machines. Working with configuration management, they schedule processes to run at night using servers belonging to many departments. The project team receives restricted access to the servers for their scheduled tests, so the departments can safely share their resources without compromising the server's environment. Any files installed or generated during the tests may be automatically cleaned up by the system, restoring the server to its original condition.

Advantages of the BuildForge System

The BuildForge system's unique approach to automation offers clear advantages over other methods. These advantages derive from:

- Anchoring the system on the true deliverable of any software project, the executable
- The ability to report based on what really happened in development

Built on the Key Reliable Artifact of Development – The Executable

The effort required to write code tends to overshadow the process that produces a final product from that code. Processes like building software have typically been seen as necessary evils that occur just before the product is ready to ship, and handled by understaffed configuration management or build teams within the organization—yet the executable is the only real deliverable that the product development team produces. Consider these examples:

- Developers can write code for months, only to find that an improperly built executable fails to function correctly for customers.
- Your source control system tells you that a defect is fixed, but that branch never makes it into the final release. As valuable as the SCM system is for tracking code, it can't show you what really made it into the product when you are trying to track down whether a particular issue was resolved.
- Your bug database relies on human input when it insists that a problem is fixed. The executables that your team produces become the definitive product of record. To be accurate, you need a way to manage your development that derives its data from the executables. When you use the BuildForge system to create your products, the system can report on the bug fixes that went into the executable and the code used to make it. It ties your development process together, and allows you to track backward from problems to the code that produced them.

Reporting What Really Happened

The BuildForge system bases its reports on detailed records obtained from specific process runs, whether that process is a software compilation, an automated test, or a website update. This differs from other document-based development systems that require manual input as their information source. Consider the benefits of an integrated development environment using BuildForge:

- Your source control system can tell you what code was checked in, but the BuildForge system can tell you
- what code was actually used to make a release.

- Your defect tracking system can tell you which fixes were supposed to make it into a release; the BuildForge system can tell you which fixes actually made it in, and can run automated tests to validate them.

- When a customer calls with a problem in a specific software release, you can refer to the BuildForge system to determine what was included in that release.

- When you encounter a problem on your web site, the BuildForge system can tell you exactly what changed, and can help you roll back to the earlier version if needed.

Solutions for Software Development

Whether you want to speed up your development process, enhance the quality of your products, enable distributed teams, or fulfill auditing and compliance requirements, the BuildForge system can help.

Accelerating Your Processes

The BuildForge system excels at speeding up your development process with features that address typical bottlenecks, including:

- Handoffs of a project from one group to the next
- Processing speed
- System downtime
- Underused hardware
- Opaque batch processes

This section describes how the features of the BuildForge system address these problems.

Faster Handoffs and Fewer Fumbles

The BuildForge system's self-documentation features help ensure better coordination...whether people work across a building or across an ocean from each other. Teams need not worry about passing the appropriate information to the next team downstream from them; they can concern themselves with handling their

portion of the work appropriately and encapsulating the information in the system, trusting the system to store it and pass it along. The following features facilitate this process:

- Bill of Materials (BOM): The system organizes a description of the contents of the process run and notes about it into a package which can include checkpoints to show file changes during the run.

- Logs: The system logs activity and command output, and makes those logs available across your organization according to the security settings you specify for your projects.

- Notes: The system allows users to enter notes relevant to projects and tasks within projects.

These features combine to provide a clear picture of exactly what happened during a process: the system tracks who did what, on which machines, what the results were, and how long the process took. Such a comprehensive set of tracking features can make it possible to efficiently comply with auditing rules. For more on this topic, see "Enabling Auditing and Compliance Management."

Increased Speed and Uptime

When you use the BuildForge system, you can take advantage of increased speed and uptime from concurrent processing and server pooling:

- You can group several machines into a server pool, then start several process runs. As soon as one server is fully occupied, the system redirects processes to other servers in the same pool.

- A single process, when started, can launch tasks on several machines concurrently for greater speed.

- You can launch processes on different machines to build different versions of an application, such as for several different operating systems, in the appropriate native environments.

- If the default machine specified for a process is down when the BuildForge system attempts to run it, and the system was a member of a pool, the system automatically hands the process to another member of the pool and the process continues uninterrupted.

- A process can run the same task on every member of a pool at once. For example, if you want to deploy a code update to all of your pooled Linux servers, a single task can do it, and the updates happen in parallel.

More Speed with Less Hardware

When you implement the BuildForge system, you make it possible for your departments to share their hardware in a controlled fashion. Without these safeguards, a department or even an individual project tends to purchase hardware that is dedicated for a single purpose, creating idle, underutilized hardware over time. But with the BuildForge system in place, the following scenarios become possible:

- A department that needs to occasional run some processes on a Macintosh system can be provided with limited access to one...without needing to physically access or modify the system. Access can be restricted to specific tasks and cleanup can be automated. If the use becomes too frequent, access can be removed.

- One department's servers can be set up as backups for another's, so that if a key server in a department is down, the system automatically uses the secondary machines until the preferred machines come back online.

- A group of machines can be designated as a pool of available servers, ready for any process that needs them, regardless of the department or project involved.

- A machine can be set up to offer only a portion of its processing power to other departments.

These scenarios can translate into increased processing speed and lower costs for your organization.

Process Clarity and Analysis for Continuous Improvement

Because the BuildForge system tracks all of the activities you perform during the development process, it contains a wealth of information to help teams improve their processes. BuildForge generates a collection of pre-built reports that you can use it to analyze your processes and continuously improve them. In contrast to using an extensive batch file or script that provides little visibility to your processes, with BuildForge you can:

- Immediately pinpoint errors without searching through extensive log data

- Review individual processes to see how long they typically take, looking for anomalies or trends

- Track down where problems originate, across departments or within departments

- Track server utilization over time

- Identify areas of the code base that are changing most frequently, and pinpoint error “hot spots’ to improve project planning

- Alternatively, you can generate your own reports from the system's database using the documented schema provided with the system.

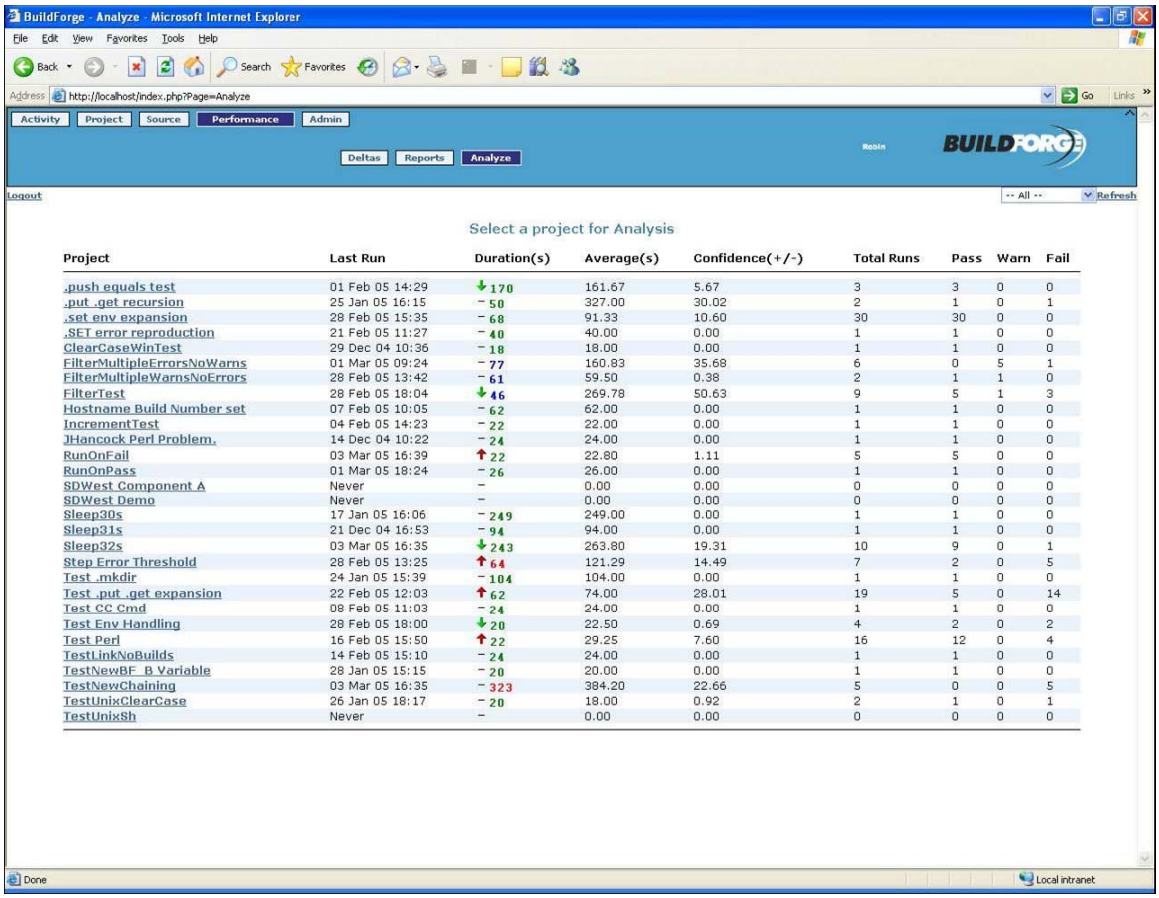

Figure 3: Historical project analysis using BuildForge

Improving Quality

Speed alone is not the whole story. When you implement the BuildForge system in your organization, you may experience a variety of quality improvements that result from having a consistent automation system, standardized development processes, and better team communication.

Repeatable Processes

With the self-documenting BuildForge system, you can repeat a process exactly as it was run the first time. When you drive your activity from the BuildForge database, the system can repeat a run of a project using the values fed to the system in the previous build...or display those values for you, allow you to change them, and then run the build. It's ideally suited for iterative development and for replicating past processes to test them or duplicate problems. You can even create shared libraries of processes that can be reused from project to project: this provides valuable quality controls through standardization and reduces the time required for new project setup.

Knowledge Retention

Turnover is a fact of life in all organizations, but the BuildForge system helps you survive it. The system's centralized knowledge base automatically stores process information within the company's infrastructure, protecting that investment for the company and dramatically lowering the learning curve for each stored process. With it you can:

- See the current definition of a process
- See the results of previous runs of a process
- See the changes in a process definition over time
- View notes that explain why a process was changed

The system drives the creation of a repository of process knowledge, automatically archives it, so that processes are executed in a standardized, consistent, and repeatable fashion.

Reducing Human Error with Automation

The BuildForge system gives you the tools you need to implement a comprehensive development automation system. After you design a process, you can easily automate it, and then share the automated process with the rest of your organization.

The system supports automation by allowing you to:

- Schedule regular process runs
- Drive activity based on the success or failure of a run, or a portion of a run
- Drive activity based on changes to source code
- Deliver automatic reports based on the success or failure of a process
- Provide an automatically-generated Bill of Materials (BOM) with each run, so that you know immediately what makes up a run, why the process was run, and who is making the changes.

With these features in place, each team becomes more productive:

- The development team gets more frequent code builds, providing them with immediate feedback on the success or failure of their coding efforts to keep them on the right track.
- The quality assurance team gets details on the availability and contents of each release and can start from the results of automated tests to judge problems in a release. Through the BOM, the QA team knows what changed and what tests should be run. With automated tests, many routine problems can be eliminated from consideration.
- The configuration management team spends less time managing day-to-day operations, leaving more time for optimizing development processes.

- The information technology team can use BuildForge reports to determine system needs.

Centralized Control and Distributed Access

Communication between teams is often a major impediment to timely releases. Without an effective build and release automation system, the configuration management team's ability to respond to demands for new builds, new tests, and new projects gates the development process. In addition, developers who are disconnected from the build process often lose productivity while waiting for results. Similarly, if a project is non-repeatable and undocumented, information about varying product versions is difficult and time-consuming to collect, and reproducing a customer's problem can become the most time-consuming part of fixing it.

BuildForge supports distributed development teams by providing centralized control and secure, role-appropriate access at the same time. You can bring processes from many departments into one central reporting system, then spread access to the processes (and the information they generate) across your organization as widely as desired. A centralized CM group can create and test a process, then grant the ability to use that process to appropriate groups of people who may not all belong to the same department. And individuals not part of the formal configuration management team can create their own processes, and share them companywide. At the same time, you retain control over who can run an automated process, view a report, or define a new process.

With the BuildForge system in place, a developer can launch a build project whenever it is needed, to provide instant feedback and test a new feature or a fix...but he can only launch those processes that your organization has decided he should have access to. This kind of self-service makes developers more productive and reduces the time your configuration managers spend responding to requests. You can extend this idea to any process that you automate.

Providing Agility Through More and Better Test Cycles

Though automated testing can never replace the insight of a quality assurance team, the BuildForge system makes it possible to add efficient automated tests to your build processes and share those tests across your organization. And the self-documenting nature of the BuildForge system makes each test cycle more productive, as each build contains information about what happened in it and what its executable contains. Any iterative development process (such as Agile Development and other methodologies) demands ongoing code-build-test activity to provide frequent feedback to the development and testing teams. The BuildForge system provides powerful automation, scheduling, and source control integration that enables Continuous Integration.

Through the BuildForge system, a tester can start to evaluate a product with many details already determined:

- Problems that can be detected by automated tests can be fed back to development before QA sees the release, so that manual testing begins only after automated tests have passed.

- Previously encountered errors can be quickly retested, speeding and improving regression processes so that developers and testers can focus on fixing new problems with less concern over introducing new ones.

- Stress testing and loading can be tested via automation.

- The BuildForge system provides infrastructure for kicking off the tests on all the target machines and collecting the results.

- The BuildForge system's filters can pre-process test results, enhancing reporting with detailed lists of warnings or failures.

- The BOM report details the bug fixes that a build contains, so each can be tested or matched against automated test results.

- The BuildForge system can be integrated with your source control system and bug tracking database to feed results back to them.

Possible uses for the BuildForge system for a test group include:

- Smoke testing: after every build, launch a subset of basic tests and report on their success or failure to determine whether the build still works after changes.

- Automate installations of completed builds on multiple target platforms. You can use BuildForge to set up your test environment, and notify testers that machines are ready for test scripts to be run.

- Projects in the BuildForge system are tagged with different classes; you can use the project class to determine which tests to run:

    - When a developer just wants a quick scratch build, skip the testing for speed.
    - For a scheduled build, run a smoke test to make sure nothing major is broken.
    - For a production build, run a full test suite, perhaps including scaling or load testing, a suite that may require a significant part of your system resources.

These features make it possible to build more and more efficient testing into your processes, and thus more quality into your product releases.

Designed for Integration

At heart, the BuildForge system is a solution for tying applications together to accomplish larger tasks. With BuildForge, you can easily mix applications, operating system commands, and batch files or shell scripts within your projects.

With its unique ability to encapsulate command lines, the BuildForge system creates the necessary environment for each command on the target computer system before sending it the command. And you can define the environment for a command to integrate it with the other pieces of software you want to use. Beyond its basic function of tying together applications, you can integrate with the BuildForge system in the following ways:

- You can integrate your existing user data via the LDAP standard, so that you can avoid creating an additional set of user logins for the new information system.

- BuildForge offers out-of-the-box adaptors for major source code control systems that can detect changes and launch process runs based on them.

- An application programming interface (API) allows you to drive BuildForge functionality from other applications within your information system, giving you fine-grained control over your processes.

With these features, the BuildForge system remains an integral part of your development environment and a well-behaved member of your corporate information system.

Enabling Auditing and Compliance Management

As discussed earlier in this paper, compliance is a topic that is on the minds of most development teams. Even if your team is not responsible for the company's financial systems, if your group delivers products or services that are critical to the company revenue streams then you should prepare to be audited. In the aftermath of Sarbanes Oxley, many companies have decided that standardized, repeatable, and documented development processes are simply an essential best practice for doing business. The key is to make the collection of this audit data as effortless as possible. The BuildForge system captures critical data and provides end-to-end traceability to use as evidence for compliance and auditing tasks:

When you use the BuildForge system to run a process, the system automatically tracks your process from start to finish. The more processes you build into the BuildForge system, the clearer your development picture becomes.

The system retains version information on all of your processes so you can tell what changed, who changed it, and why for each product iteration.

The system provides a Bill of Materials (BOM) with each completed process run.

The sections that follow detail these features.

Automatically Tracking Your Processes

Even the simplest of scripts becomes a valuable compliance tool when you run it through the BuildForge system, because the system stores data about every run of a project within it. A batch file or shell script that copies files from a source location to a web server, when run within the BuildForge system, quickly becomes much more than a script:

- The system reports success or failure via e-mail as well as on a web-based dashboard.
- Attempted commands and resulting output or error messages are stored in the logs.
- The system can schedule the script for repeated runs, guaranteeing the standard process is executed the same way, every time, and occurs as often as needed.
- The system can find available server resources to run the script, rather than relying on the availability of a single machine.
- The system logs who started the script.
- Other user actions, such as changes to the script, are logged.

User notes help document the reason for each change that occurs.

When integrated with your existing tools, the BuildForge system translates information silos into useful compliance data that is available at your fingertips.

Versioning Your Processes

When you add a process to the BuildForge system, the system stores many pieces of information about that project.

This includes the steps in the project, the environment variables needed by the project, and the server (and/or pool) that the project should run on). The project record can be updated at any time as you change your process; the system retains the current version of the process as the default set of instructions for that process.

Each time you run the project, the system also stores a copy of this information. You can review the record of the project run to see exactly what steps were performed on a particular occasion, though the process definition may have changed in the meantime. Further, the system can recreate an earlier run, following the instructions from that run and ignoring intervening changes, when desired.

Thus, any process added to the BuildForge system immediately becomes a repeatable and traceable one. If changes to a process result in quality problems, you can return to the last good state.
Although the BuildForge system automatically stores process records within its own database, you can archive the process instructions in your source control system as well. This allows you to store the instructions for a process along with the files used to create the product. You can use the BuildForge system to automate the archiving process as well.

Self-Documenting Systems via the Bill of Materials

The BuildForge system includes a configurable Bill of Materials (BOM). The system automatically produces a bill of materials (BOM) for every process that it runs. The BOM provides a concise package of information about a process run that can be used to review the results of the process, and serves to document the process for various consumers of the product.

The system automatically includes certain items of interest about every build in its BOM.

- You can add tasks to a process that cause additional information to be written to the BOM, using provided commands. For example, you can store information about the files in the process working directory, then show how the process changed those files at various checkpoints during the process.

- The BOM acts as a ticket passed from one group to the next in the process. Without one, a build is a mystery package; with it, a team can quickly evaluate what a new build means to them.

- The BOM can even be used as a working outline for the technical publications team as they start to document a release.

The BOM acts to encapsulate the process and its end-to-end results in one convenient package suitable for auditing or verifying compliance.

Realizing the Benefits

The BuildForge system does not transform your organization's processes by itself. You must implement the system within your organization to see the benefits. BuildForge FullControl and FullThrottle are easy to install and implement, and can be readily evaluated in a customer's environment.

Evaluating the System

The best way to find out whether the BuildForge system is for you is for one of your team members to download, install, and test the software. Though the system is designed to be installed on many machines which work in parallel, you can test most of its functionality on a single machine after running a pair of installation programs.

The *BuildForge Implementation Guide* includes instructions for setting up an evaluation system, as well as a tutorial for learning how to use it. You can try out representative projects from your own development process, or implement a single process end-to-end within the system, and make your own evaluation checklist. That way, you can validate that the product works in your environment. Pilot implementations can typically be completed in two to four weeks.

Product evaluations require prequalification. If you are interested in conducting a product trial or learning more about BuildForge, contact your BuildForge Account Executive or e-mail sales@buildforge.com.

Implementing the System: Phased Deployment

When you implement the system, you won't need to change your processes overnight. It's neither necessary nor desirable to implement it all at once. Instead, you can start by adding a single process to the system. As soon as you are happy with the results, you can select key processes to add, then prioritize others.
In a large organization, a single department might adopt the system, then spread it to others after demonstrating its worth. The system can thus grow organically as users see its value, rather than seeing the resistance that a mandate from the top of the organization can cause.

Each process that you add to the system becomes a repository of controlled knowledge about your organization that you can hand to the people who need it, while restricting it from those who don't need it.

Conclusion

The BuildForge system provides a complete process management and automation system that enables development teams to be more productive and rapidly deliver high-quality products to market. Customers using the BuildForge system begin by adding key processes into it; after they analyze the return on their investment, they are motivated to add more processes to the system. The system's ability to coexist with existing tools makes adoption far easier than implementing an ALM suite. As your teams use the system, they find that they retain control over processes and machines, but the safeguards built into the system enable them to disseminate information and tools further than ever before. The system makes it possible to

integrate teams and tools of varying origins yet bring the information created by them into a central repository so that the scattered components of a complex development process can be brought into sharp focus.

If you are interested in conducting a product trial or learning more about BuildForge, e-mail sales@buildforge.com.