SHORTEN RELEASE CYCLES BY BRINGING DEVELOPERS TO APPLICATION LIFECYCLE MANAGEMENT

Business white paper for application team professionals

The new challenges of application lifecycle management

Today, software applications can be found embedded in almost every possible product and market, from supermarkets to cars and houses, and even washing machines. Companies with advanced technologies that can move quickly to meet customers' demands and bring new, innovative products to market find themselves ahead of the competition.

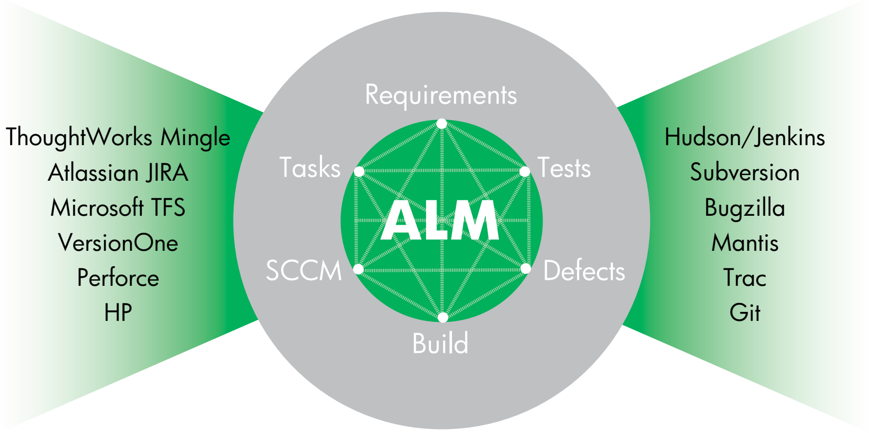
To respond to changing customer demands, organizations must shorten software release cycles. Many companies are moving to agile methods to be able to rapidly address changing business needs. To make agile and other iterative methods work, application teams need to have real-time communication and visibility across the team and across the application lifecycle. However, many organizations are using a variety of best-of-breed tools throughout the application lifecycle that cause information to be scattered and difficult to share. This makes collaboration and communication between teams difficult, which can slow down release cycles and make agile methods difficult to implement.

Cloud, mobile delivery, and service-based composite applications are market-disrupting technologies that are bringing greater complexity to the application development process. On top of these challenges, IT budgets are constantly shrinking. IT organizations are required to do more with less and hence are looking for solutions to increase productivity and operate with less cost.

To address these challenges, it is becoming more important to have a management platform for application lifecycle management (ALM). ALM has evolved to align all application teams to the process of planning, developing, and testing software applications. It increases control, visibility, and predictability for the process of building modern applications. It also enables change management, risk-based prioritization, and communication across disparate teams. But development teams are often resistant to participating in ALM, and without their cooperation, ALM cannot deliver all of its promised benefits.

This white paper explores the capabilities and benefits that the next phase of ALM must deliver in order to energize application lifecycle processes, serve the needs of the business, get developers involved, and help overcome these market disruptions. What we see at HP is that successful companies embrace governance, real-time visibility, measurement, and collaboration across application teams.

Figure 1
ALM provides a hub of information across heterogeneous tools and environments



Why ALM must change

The original ALM

When ALM first appeared in the 2000s, it was a vehicle to introduce business and management discipline to software development practices. Vendors developed integrated, single-platform tool suites. However, this integrated ALM approach didn't match the organization's needs because of:

- The costly budget required to replace existing tools and adopt a single integrated platform
- The need for a significant process change
- The large platform implementations
- The growing adoption by developers of an increasing variety of open source tools

These trends have changed the approach of standardizing all application teams across a unified platform. Rather than trying to standardize across a single platform, enterprises have acknowledged the need for application teams to use a best-of-breed set of tools of their choice. With applications that are now defined by their connectivity, ALM must become a unified platform that operates with the ecosystems of different products and solutions.

Silos on the applications team

Medium and large organizations commonly have heterogeneous ALM and application development tooling environments, often consisting of third-party, open source, and agile ALM tools. These are used by distributed teams employing hybrid delivery models, which include a mix of agile and non agile methods. This mix creates silos between development teams, project managers, and testing teams. These silos make it difficult to have visibility or traceability across the software lifecycle, resulting in delays, missteps, duplication of effort, and even project failure.

Most ALM solutions are able to connect the business analysts, testers, and project managers on a single unified platform; however this is not sufficient to ensure project success. In order to keep up with the fast pace of iterative modern delivery, it is essential to capture development work as well.

The impact of agile methods

Over the past two years, agile and agile-like methods have become the de facto standards of application delivery teams. Because many ALM tools were not designed to support agile and other iterative methods, many application teams must re-evaluate their tools and practices.

The frequent sprint release cycles in agile methods require development and testing teams to test more, test faster, and test earlier in the lifecycle. Ideally, testing should take place in parallel to development, and development should be test driven. To do this QA teams should test continuously, and they must have real-time visibility into development work in order to identify new builds to validate as early as possible.

Agile methods encourage a high level of traceability. Task management or work item management provides agile teams with a universal mechanism for traceability between change sets, tests, builds, and requirements throughout the overall software development lifecycle (SDLC). The consolidation of agile project management solutions and ALM suites creates a hub that communicates ALM artifacts to all stakeholders to increase productivity.

Distributed application teams—is it a small world after all?

Distributed application teams

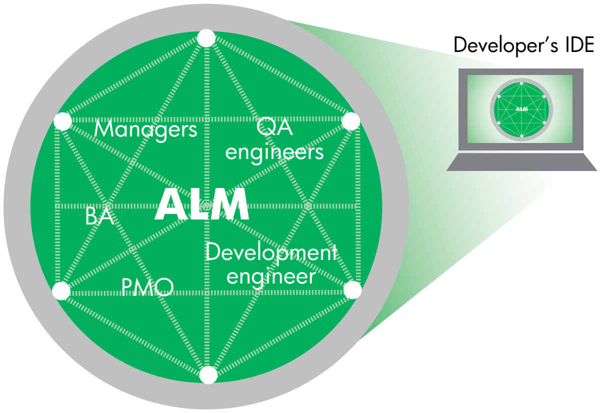
Mergers and acquisitions, outsourcing, and the trend toward globalized work teams have created new challenges for ALM. Gone are the days when all key stakeholders in the development process worked in the same physical location and used the same toolset.

In today's world, members of development teams can be located virtually anywhere in the world. Because they may use diverse toolsets and different processes, the information related to a development project can be scattered across multiple systems. This creates ongoing collaboration and traceability challenges for organizations that manage the development process and the rest of the lifecycle of software applications. In addition, this fragmentation in the development process limits agility, predictability, and change readiness.

The need to join forces closely with development

As an example, suppose that in a global company, Team A uses Java, Eclipse tools, and agile processes, while Team B uses Microsoft® Visual Studio and waterfall processes. In the absence of a comprehensive ALM system, Team A and Team B cannot keep up to date on each other's activities. Stakeholders on Team A may not be aware of the identification of a software defect or the changes made in response to it by Team B. Disjointed communications and a lack of traceability lead to lost productivity, project delays, and costly rework later in the development process.

Figure 2
Connecting silos between developers and application teams



It might seem that many of these challenges could be solved by standardization on a common set of development tools and processes. But the reality is, developers are most productive with the tools and processes they know well, and they aren't eager to change. Shifting to new tools and processes can be disruptive to both people and productivity.

What's needed is an end-to-end ALM solution that provides a common management platform that can be shared by all project stakeholders and that allows individual development teams to use their tools and processes of choice.

Getting developers to work within ALM

The SDLC doesn't represent an application's life whereas ALM looks beyond the development process to the application's time in production and the changes that occur to the application over time. While this makes ALM attractive to some stakeholders, it is challenging for organizations to convince the development team to work within ALM because developers consistently refuse to leave their integrated development environments (IDEs).

The IDE is the developers' home environment where they are the most productive and comfortable spending their time. Even when they are introduced to advanced, cutting-edge project management tools with an advanced Web-based user interface and are requested to log in to update their tasks, it does not happen. Developers don't like to follow processes and fill in forms; developers like to develop. On the other hand, keeping up with the fast pace of frequent and shorter release cycles is dependent on real-time visibility by other stakeholders into development work. Application team members must be able to quickly assess the current state of the application, including:

- **The business goal**—Why is the new content or new code being checked in? What is the functionality that it is supposed to deliver, and what is the business reason for it?
- **The motivation**—Is it aimed to fix a defect, or is it a derivative of a new requirement or a new user story?
- **Traceability**—Which build does this content belong to? Which requirement or defect?
- **Release progress**—How does this content contribute to the progress of the release or the iteration? How does the new piece of code help the team achieve its goals for the release?
- **Risk assessment**—How stable is this code? Does it have areas that are fragile that need further attention by the testing team? How complicated is it to change this requirement?
- **Governance**—How can we ensure that developers are spending their time in alignment with business priorities?

Since the IT organization would like to introduce processes that increase developers' productivity and efficiency, it makes sense to accommodate the development team's needs. By introducing light-weight processes and tools that enable developers to work in their familiar environment, it becomes possible to bring the development team into the ALM fold.

Productivity relies on IDE traceability.

Context-driven development

ALM solutions should enable this type of context-driven development by providing developers with the ability to view ALM artifacts while coding.[1] Developers can better understand the functionality that their code should deliver if they are able to view the requirements or defects that provide the motivation for their specific tasks.

Developers gain a better understanding of what was already done and can better estimate what should be added going forward if they are able to view the business requirement, history of change sets, and latest defects that are connected to their task. Ideally, the ALM system should provide visibility into linked code change sets that are traceable to requirements and defects. To increase productivity, this contextual information should be available to developers directly from their favorite IDE.

Context-driven reporting

The entire applications team better understands what is going on when the ALM system is automatically updated as the developer codes. From the developer's perspective, a task should be presented in the IDE in a way that can free the developer from the need to report progress, update status, and add a description of his work.

The integration between the developer's IDE and the ALM system enables the rest of the application team to see the context of the developer's work. The project manager, scrum master, and development manager all get accurate, real-time updates about the progress of the individual developer, the progress of the whole team, and the status of the iteration or the release. As a result, development managers and project managers gain transparency, encourage efficiency, reduce latency of inter-team communications, and increase application code quality. Because all this data is available to QA teams in real time, they can better plan their testing, accelerate testing time, and identify defects earlier in the lifecycle when they are easier and less costly to fix.

Bringing intelligence to the lifecycle

Expanding to source code change management (SCM) tools and build management tools

Risk comes first

Aiming to increase productivity and accelerate time to market, application teams should have multitasking capabilities and a quick decision making mechanism. Informed decisions should be always based on risk. In a world of increasing demand form the business to accelerate innovation while cutting costs, risk should always come first. To facilitate that, ALM solutions should expend their traceability of defects to requirements to tests to capture code changes as well and connect to build management systems to surface actionable information. Application teams must be able to understand what was changed in the code in order to focus their testing resources on the right priorities. Additionally, application teams should have a mapping of complexity factors that are associated with each ALM artifact. Based on that, the testing team can focus testing resources on the high risk components first.

Predictability into the development work

Testers should gain predictability into what is about to be transitioned from development to testing, what's the content of the new build, what's the functionality the new build is supposed to deliver, and its stability and coverage. Based on that, testers can better plan their testing, project managers, and development managers can better track the release progress and adjust resources and priorities. These capabilities can be achieved by linking ALM solutions to the various build management solutions the organization is working with. This visibility into the health, stability, and risk of the build can further assist with decisions with regards to the readiness of the build to be transitioned to production.

Apply governance

Today, most IT organizations lack the ability to control or monitor changes to code that are checked in by developers. They cannot tell whether code changes are linked to any business justification. If the changes are unnecessary or unplanned, they may lead to longer release cycles and errors in production code. Using a source code version control system does not address this issue. Although code changes are checked in, they are not linked to the defects or business requirements that motivated their development.

Allowing freedom of choice

ALM solutions should enable IT organizations to develop applications in a transparent and traceable way, regardless of development methods, platforms, tools, and processes. In addition they should ensure that any software configuration change gets reviewed and validated as quickly as possible in order to keep the project moving through the deployment phase.

[1] Source: "Time is right for ALM 2.0+," West, Dave, Forrester, 19 October, 2010.

The value of surfacing change sets in the ALM context

A single system of record that provides traceability of requirements to defects to tests and up to code changes and build metrics generates, benefits for all the application teams, management stakeholders, the testing organization, and developers:

- **Management:** ALM solutions should simplify the life of the development manager with regard to release tracking, resource management, and change impact analysis. Managing multiple projects, developed by disparate teams and while using heterogeneous tools and systems, development managers must have capabilities to aggregate the relevant data and present it in the ALM dashboards. That includes data about release progress, coverage, health and risk, including tracking team and individual velocity. Providing detailed information about the content and the stability of the new build helps the development manager and the project manager to better align resources. By analyzing the magnitude of effort required to address a requirement, user story, issue, or defect, development managers and project managers can optimize resource allocation for making the required changes. Furthermore, it allows them to understand the risks to the application associated with the changes. In addition, managers can assess the progress of an iteration and the overall release by understanding the exact functionality that the new build is aimed to deliver.
- **Testing team:** By understanding the functionality that new code is supposed to deliver, testing teams can more easily plan their testing. The information helps the team to plan effective testing that is focused on the more fragile components of the build. They can perform risk-based test planning by analyzing the level of complexity associated with the ALM artifact.
- **Developers:** In order to rapidly execute their tasks, developers need to quickly identify the pieces of code to modify in order to fix defects or fulfill requirements. To do this, developers must be able to view the change sets that are related to their tasks, and it is particularly helpful if they are traceable to the relevant ALM artifacts. Especially when dealing with composite applications and complex environments, this mapping provides the developer with intelligent understanding of the relationships between tasks, source code components, and ALM artifacts. Based on the principle of context-driven development, providing this information saves developers time and effort and shortens the release cycle.

Agile project management tool—bringing it all together

Scalability across disparate teams

Agile project management solutions should provide IT organizations and application teams the ability to create a unified, flexible, and auditable development process that works with all tools, platforms, and processes throughout the entire application development lifecycle.

On top of that, since agile methods are implemented in different ways from one organization to another, the agile project management solution should be scalable and allow IT organizations to define flexible workflows to meet their evolving requirements.

The agile project management system should provide the development manager and project manager with a solution to manage tasks and workflows. It should surface information across disparate application teams and provide actionable information about team velocity and the progress of the release as compared to the plan. This helps make sure that the application makes it to market as planned and on target.

The need for a tightly integrated solution

Gone are the days when the project manager was able to chase team updates by foot or over the phone. Since developers avoid following administrative processes such as reporting and documentation, today's agile project management solution should provide mechanisms for real-time automatic updates.

Capturing the progress of the developer's work together with the progress of the whole iteration can be provided through integration between the project management tool, the developers' IDE, and ALM repository. With this integration, application teams get end-to-end traceability between all ALM artifacts to create an orchestrated development process that delivers enterprise-wide visibility across distributed application teams.

To provide complete insight to all application teams, progress updates should capture the context of the developers' work. These updates should automatically document the linkage of a task with its relevant requirements, user stories, defects, change sets, and attachments.

An integrated agile project management solution accelerates development by removing process bottlenecks, coordinating changes, and providing real-time dashboards and updates across all projects, teams, and tools. These measurements add visibility into both development data and process metrics across all solutions in the application development lifecycle. IT organizations can simultaneously view development data from multiple tools, such as the number of defects—as well as cycle time metrics from the agile project management solution. Cycle time metrics may include how long a release has been in the testing phase, the lag time between development and testing, which areas are not on track, and the biggest bottlenecks across all projects.

HP Application Lifecycle Management

HP Application Lifecycle Management (ALM) is a unified platform that provides end-to-end traceability of application lifecycle artifacts across projects and across the lifecycle, from inception to retirement. HP ALM is a hub of information to multiple application lifecycle tools. Because HP ALM is a single repository that provides a single system of record, it increases collaboration across heterogeneous teams, roles, platforms, and ecosystems.

According to our case studies, with HP ALM, application teams can deliver applications up to two times faster, while reducing production issues by 80 percent on average.

HP Development Integration

For organizations with a variety of heterogeneous application development and lifecycle management tools used by distributed development teams, HP provides developer tool native integration and third-party ALM tool synchronization. This makes HP ALM the most developer-oriented collaboration and insight hub for application delivery teams.

The intuitive, developer-centric user interface assists development teams in viewing ALM artifacts while coding—from open-source and commercial IDEs. This unique feature makes it easier for developers to update their progress and allows application teams to capture the context of their work in real time. Because the user interface is familiar, HP ALM motivates developers to use the application lifecycle management suite. As a result, organizations gain transparency, encourage efficiency, reduce latency of inter-team communications, and increase application code quality by finding defects earlier in the application lifecycle. This results in a return on investment (ROI) of up to eight times.[2]

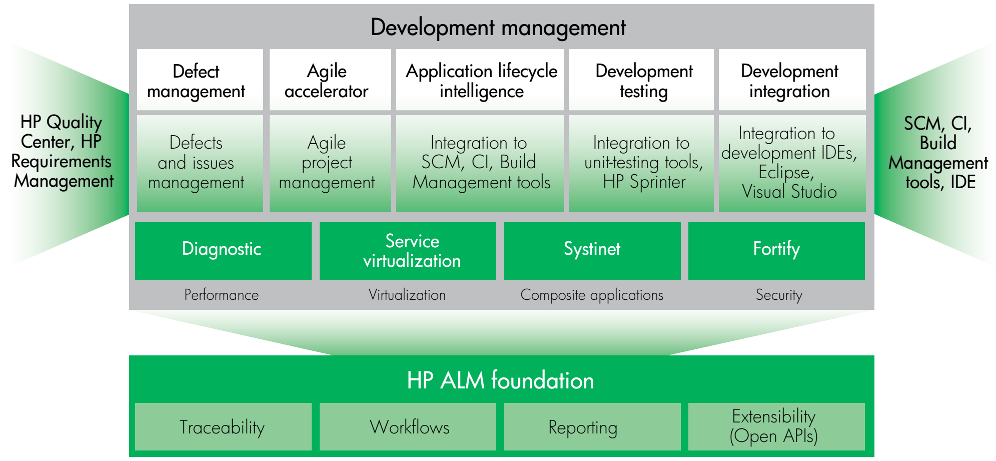
HP Application Lifecycle Intelligence (ALI)

ALM solutions should scale the application lifecycle to meet the fast pace of change in today's world. HP ALI facilitates that by linking code changes to ALM artifacts allowing testers to focus their testing efforts on changes in the code only and thanks to that dramatically cutting testing time. This outstanding traceability is provided while allowing application teams to work with heterogeneous SCM and Build Management tools. While allowing developers this freedom of choice, HP ALI aggregates data about everything that matters to application teams and provides visibility into what's happening across disparate development teams.

Thanks to that HP ALI empowers the application teams to make informed decisions based on an intelligent understanding of the health, stability, and risk of a new build. This is provided thanks to various metrics that are generated automatically by HP ALI and are constantly updated. Metrics include: unit test coverage, unit test success rate, defect density, amount of code changes per each requirement, defect and for the whole build, including build content, and coverage.

This enhances resource prioritization and increases productivity, enabling application teams to lower their operating costs.

Figure 3
HP ALMDev portfolio


Development management
Defect management
Agile accelerator
Application lifecycle intelligence
Development testing
Development integration
HP Quality Center, HP Requirements Management
Defects and issues management
Agile project management
Integration to SCM, CI, Build Management tools
Integration to unit-testing tools, HP Sprinter
Integration to development IDEs, Eclipse, Visual Studio
SCM, CI, Build Management tools, IDE
Diagnostic
Service virtualization
Systinet
Fortify
Performance
Virtualization
Composite applications
Security
HP ALM foundation
Traceability
Workflows
Reporting
Extensibility (Open APIs)

[2] Source: "Evaluating Tasktop Developer ROI," Tasktop white paper, Tasktop, 2010.

HP Agile Accelerator

HP Agile Accelerator allows HP ALM users to rapidly adopt and manage agile projects. It lets them plan their sprints, build user roles, compose user stories, and configure flexible workflows and tasks. In addition to sprint management and tracking, it provides reports and graphs, such as burn-up, burn-down, and velocity. The layout, which is available as a Kanban-style taskboard, provides an at-a-glance view of tasks, their status, and their owners as they progress through the workflow.

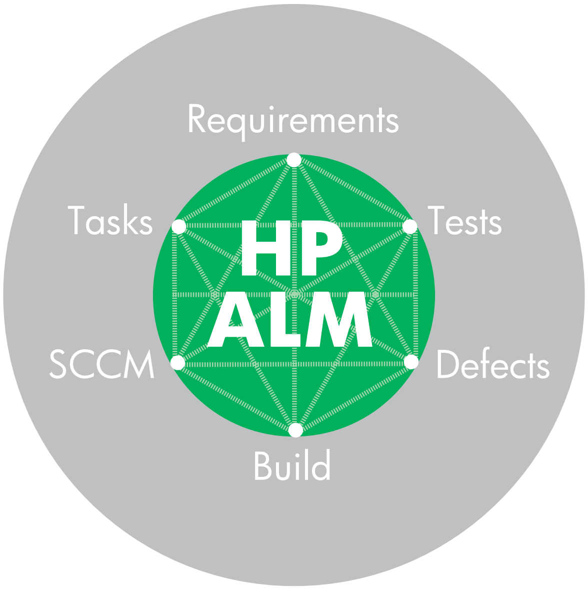
Because HP Agile Accelerator sits on top of HP ALM, you can view on ALM both agile and nonagile projects on the same platform.

Motivate application teams to participate in ALM.

HP ALM allows application delivery teams to stay in their familiar environments, where they are the most productive:

- **Developers** work in their home IDE.
- **Project managers** work in their project management tool.
- **QA and testing teams** work in their test management tool.
- **Business analysts** work in their requirement management solution.
- **Senior management** (such as the QA manager or VP of applications) can view the progress of the release with complete health metrics, including developer and defect status, in their ALM dashboard.

Figure 4
HP ALM—for complete end-to-end traceability



Each team member works in a role-based tool of choice while HP ALM integrates across teams and provides the right information to the right people at the right time. This enhances collaboration and communication between teams to increase productivity and shorten release cycles.

HP Application Lifecycle Management increases control, visibility, and predictability of software applications. To learn more, visit **HP Software Solutions Community/ALM** and **www.hp.com/go/almdev**.