What is Application Lifecycle Management ?

Increase productivity

Stop wasting your time doing things manually by automating every step in your project's Life Cycle

At lower costs

Get a 30% return on investment guaranteed and save 15% on development costs

For all stakeholders

Developers, Project Managers and Test Engineers are all part of the development process

RedBridge

Table of contents

What is Application Lifecycle Management

Initially, Application Lifecycle Management (abbreviated as ALM) was called Library Management. It was developer-oriented, limited in functionality (only the software source code was versioned) and static (collection of the different versions of the source code).

Later on, management also wanted to cover the complete development process. This interest in the process aspect made software change management a more dynamic discipline.

Modern Application Lifecycle Management not only wants to address the development process and related downstream activities, but also the production environment (deployment) and the upstream activities like requirements management and issue tracking.

In the end ALM has an enormous impact on the overall efficiency and costs of your application development process.

Moving forward with ALM 2.0

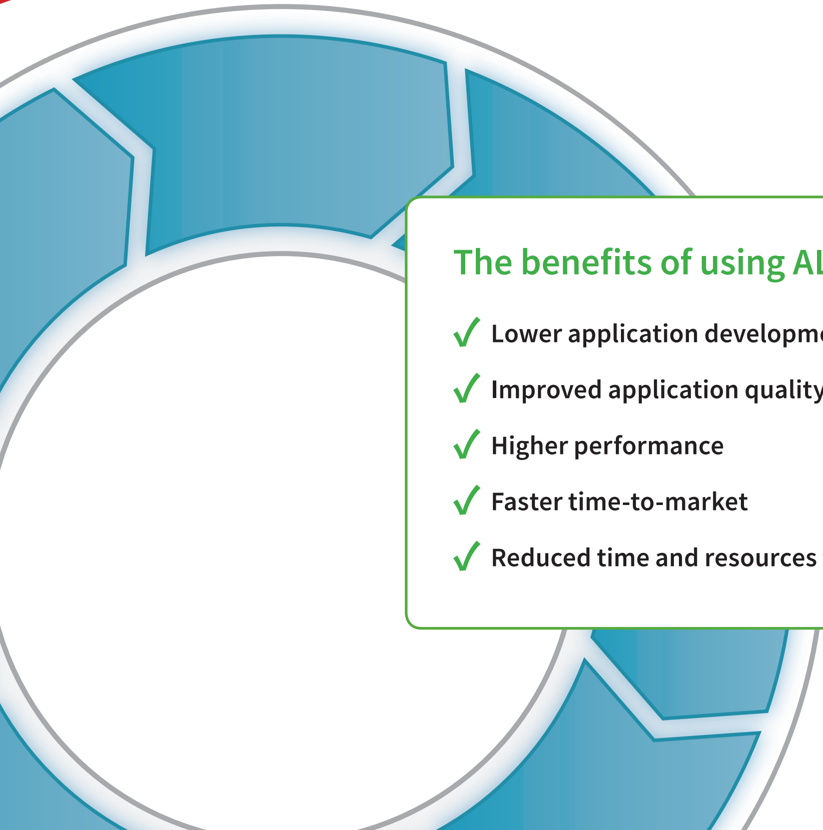
The first generation of ALM tools limited itself to collecting a set of tools used within an organization. However, in today's demanding environment, these ALM 1.0 solutions show a lot of deficiencies such as: lack of integration between the different tools, inconsistencies because of different ALM features locked in practitioner tools, lack of communication between the different stakeholders involved in the process, lack of transparency, a lot of manual intervention required to keep everything in sync, costly to maintain, …

The new generation of ALM tools, the so-called ALM 2.0 solutions, solve a lot of these deficiencies. The tools are not just collected anymore, but fully integrated to support the entire development process. The integrated tools communicate and share features, common services insure transparency (workflow, security and analytics) and all processes are automated.

ALM 2.0 solutions are **methodology independent**, **tool independent** and **repository neutral** (enabling cross-platform development).

Modern ALM 2.0 tools need to...

- Serve the complete development process
- Be independent of the development methodology used (waterfall, extreme programming,...)
- Give all stakeholders freedom of tools choice.
- Pay attention to the needs of all stakeholders (both IT and non-IT related, internal and external)



The benefits of using ALM 2.0 tools

- ✓ Lower application development, testing and deployment costs
- ✓ Improved application quality
- ✓ Higher performance
- ✓ Faster time-to-market
- ✓ Reduced time and resources to deploy multiple-environment applications

Multi-vendor approach

We can distinguish 2 major development methodologies: single-vendor and multi-vendor.

Single-vendor
Single-vendor solutions provide a proprietary ALM platform working best with the vendor's own tools. Although it is logistically easier to have to deal with only one vendor, single-vendor solutions might make you less independent and less aware of new technologies on the market.

Multi-vendor
In practice, most of the software development companies use a variety of (best-of-breed) tools produced by different vendors. The challenge will be to get all those systems to work together and to fully integrate them within the overall application lifecycle. The main advantage is that users can continue working with their preferred tools and no extra costs are involved for implementing new systems.

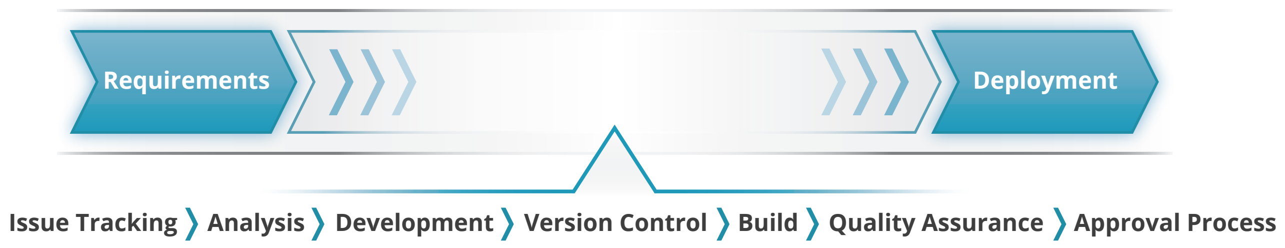
On top of offering a fully compliant ALM 2.0 tool, IKAN believes in the multi-vendor approach. No matter what development system(s) or platform(s) you are currently using, we have an ALM framework for you.

Process

Application Lifecycle Management (ALM) is a collection of processes, roles and deliverables, which is controlled and improved with each successive iteration in the software lifecycle.
Different approaches can be followed, such as:

- **The very strict, sequential "waterfall" method**, whereby the development process has been predefined as a succession of events from coding over testing to quality control and, finally, release of the application. In this approach, each phase of the project must have been completed before the next phase can start.
- **The spiral model**, combining features of the waterfall method and the prototyping model.
- **The incremental, iterative Agile approach**, whereby the project is split up in smaller workcycles, and the process can be adapted to the evolving needs of the project.

No matter which approach you folllow, IKAN ALM can be used to simplify and improve your development process.



Process phases overview

Requirements

Defining business needs and choosing matching solutions are the key issues for a successful business process. Once these issues are handled, IKAN ALM will keep track of the implemented solutions.

Issue Tracking

Tracking and managing the issues and bugs that emerge during software engineering is a critically important task. An issue tracking system will assist in creating, updating and resolving reported issues. It often also contains a knowledge base containing information on customers, resolutions to common problems, and other such data. It is a valuable asset for quality assurance and for programmers to keep track of reported software bugs and issues in their work.

Analysis

For any software engineering project – be it new application development or modification of an existing one – quality starts with analyzing the business to ensure that system requirements clearly and accurately reflect business and customer needs. Poor analysis can lead to a wide array of quality problems, including fragility, lack of scalability, and resistance to modification.

Development

This is the actual coding of a software application. Here, software developers may choose from a wide variety of software tools to help them build the applications such as the modern Eclipse platform, Microsoft's Visual Studio®.NET environment or the older C++ and Cobol development tools.

Version Control

Version control is an important part of making team-based software engineering work effectively. Version control practices help people work on the same components in parallel, without interfering with each other's work.

Build

An important part of any software development process is getting reliable builds of the software. A fully automated and reproducible build, including testing, that runs many times a day allows each developer to integrate his work on a daily basis, thereby reducing integration problems. There are several subcategories within the build process, including continuous build, rebuild, nightly build, forced build.

Quality Assurance

Too often software engineering uses quality standards that are far below those of other engineering disciplines. However, higher quality software can differentiate one company from others in its sector.

Approval Process

Reviews and approvals validate the completeness of a product and maintain consistency among its components. Controls such as to-do lists, electronic approval, document repositories and change-package based code reviews help to ensure that the necessary reviews can be done.

Deployment

At some point, a software application must be distributed to one or more servers at one or more locations. If the application is to meet or exceed service level agreements, then the IT operations team must assume responsibility for ensuring the quality of the roll-out. In this case deployment tasks and regular activities may be scheduled to start at any time, making it easier to coordinate deployments in advance of deadlines. This boosts development quality and productivity.

Improving the process

Each phase in the Application Life-cycle Management process typically attains a specific maturity level of process as shown by the chart.

IKAN ALM helps your organization to achieve a higher maturity level.

Improving quality (CMMI)

The active discipline at each phase of the lifecycle defines and emphasizes quality differently because different project teams need different approaches.

Working in a team raises the need for communication and changes the way teams would execute these approaches individually. By implementing controllable, improvable and unequivocal processes to improve the total quality, the team can achieve a higher maturity level within the overall CMMI (Capability Maturity Model Integration) framework.In the long run, it is improved quality that enables software engineering teams to deliver more projects on time, at lower cost, and with more features.

By implementing the IKAN ALM solution around best-of-breed software engineering tools, your organization's processes will evolve from chaotic and ad hoc (CMM level 1) to continuing improvement/improving processes (CMM level 5).

For more detailed information, visit the Software Engineering Institute's website.

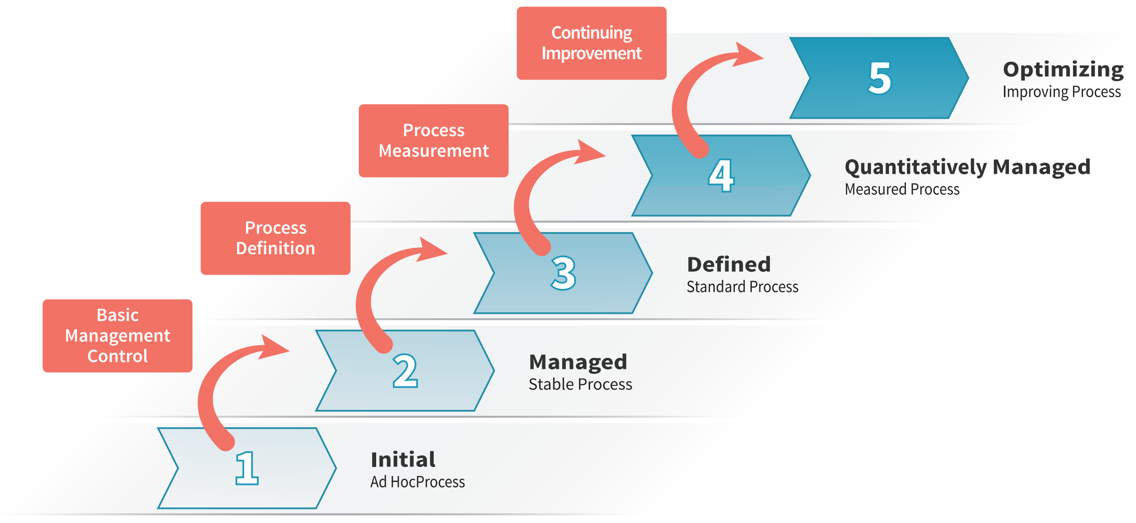
Improving traceability

It is common to have processes and controls in place for versioning.

The classic versioning systems like CVS, IBM® Rational® Clearcase (UCM and Base), IBM® Rational® Clearcase LT, Microsoft® Visual SourceSafe, Subversion, Serena® Version Manager etc. offer extensive possibilities in this field. But when building, deploying and tracking the life-cycle "pipeline" of your software applications, you lose sight of your code as versioning tools do not cover these areas. A proper ALM solution solves these issues by extending automation, control and visibility throughout the whole process, from development, through build and into production.

Overal benefits

- Fewer errors
- Faster results
- Less recursive work in QA and testing
- Predictable and shorter roll-outs
- More reliable assets in the overall IT infrastructure
- Improvement in Total Quality Management.



Benefits

Every step in the ALM process solves different problems for different stakeholders, here are some common answers.

Why use version control?

Stakeholder	Benefit
Developer	Keep track of the changes.
IT Management	Safe storage of all historic data.
Production	Easy to revert to an earlier version.
Management/Audit	No loss of data.

Why implement a continuous integration process?

Stakeholder	Benefit
Developer	Concentrate on developing software. Get early feedback on committed code.
IT Management	Get early feedback on code quality. Find weak spots.
Production	Get high-quality production code.
Management/Audit	Fewer errors. Repeatable process. Faster and shorter release cycle.

Why have an automated build?

Stakeholder	Benefit
Developer	No loss of valuable time trying to build manually.
IT Management	Allow to do more builds Give rapid feedback.
Production	Everything is coordinated by a script.
Management/Audit	Prevent mistakes.

Why approval management?

Stakeholder	Benefit
Developer	Improve communication across the project team.
IT Management	Control the evolution in the different stages of the lifecycle. Build in audit moments.
Production	Control deployment to the production servers.
Management/Audit	Traceability. Who authorized?

Why have an automated deploy?

Stakeholder	Benefit
Developer	Guarantee that production will receive the quality code that I created.
IT Management	Speed up the process and help reduce errors.
Production	No manual intervention reduces risk.
Management/Audit	Increase the release cycle frequency and productivity.

Why implement a rollback process?

Stakeholder	Benefit
Developer	More time to fix defects.
IT Management	Can always revert to the latests good release.
Production	Refuse or eliminate risk of service outage.
Management/Audit	Ensure return to exact prior state. Quickly resolve errors in production.

Why lifecycle management?

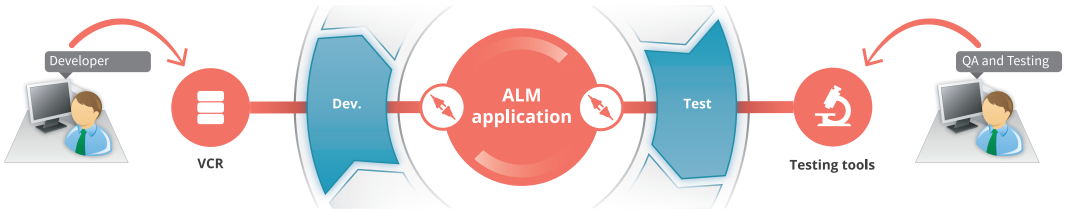
Stakeholder	Benefit
Developer	No worries of building code for the test or production level.
IT Management	Have a clear view of the development process and status.
Production	Automate production deployment. Reduce the amount of rework needed.
Management/Audit	Easily anwser the questions: Who? When? Why and What occurred?

Tools

Today there is an unknown number of tools that can be used in software application development. For each discipline, be it requirements management, issue tracking, integrated development environments (IDEs), versioning or testing, there is a wealth of choice.

We believe that a sound ALM solution should leave each stakeholder the freedom of choice on what tool he uses and the ALM solution itself should handle the communication of the different tools. IKAN ALM gives the user that freedom.

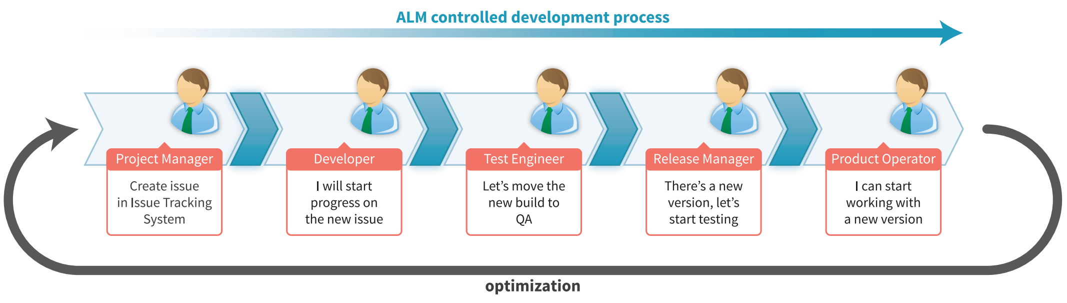
For example:



- A developer may use the IDE of his choice, as long as he can commit his code to a versioning system.
- Development teams can use any preferred methodology e.g. waterfall, spiral, prototyping, agile,...
- Different development teams within an organisation are able to use their versioning system of choice.
- IKAN ALM will communicate with existing issue tracking (or defect tracking) systems.
- Local, Decentralized or Cloud development environments are fully supported.
- IKAN ALM is multilingual and adaptable on user level.

Stakeholders

All activities of the application lifecycle involve different stakeholders, each of them focussing on different aspects of the overall development process. Implementing IKAN ALM to manage the application lifecycle, provides a series of benefits for each group of stakeholders without compromising the responsibility of the other groups.


ALM controlled development process
Project Manager
Create issue in Issue Tracking System
Developer
I will start progress on the new issue
Test Engineer
Let's move the new build to QA
Release Manager
There's a new version, let's start testing
Product Operator
I can start working with a new version
optimization

For example:

- A developer wants to have early feedback on the code committed in the trunk stream (Continuous Integration) and wants to build the project in his IDE with the correct latest sources and common libraries.
- A project manager wants to have a clear overview of the project status: is the latest code in the trunk buildable? do the unit tests run successfully? what are the guys of QA testing? which is the current production version?
- Production operators prefer an automated deploy process, where they can control the environment variables.
- The CIO and CEO of a corporation would like to see an automated and repeatable process with an audit trail.

Stakeholder	Benefit
Development	No more repetitive or unwanted costs. Cutting costs by reusing components. No changes in working environment, no need to learn new skills. 15% more productivity.
Build and Release management	Complete separation of duties. No more risky deploy mistakes. 25 % more productivity in the build and release phases.
IT Management	Increased productivity. Controllable and enforceable rules for test and approval. Cost-cutting. Overall reporting on IT level. Complete project insight.
CFO (Chief Financial Officer)	Full compliance of the whole project. Included traceability of actions. Overall reporting on business level. Control over IT ownerschip costs.
CEO (Chief Executive Officer)	High CMM level and control over budget and performance. Full compliance to regulators. Stronger and more flexible organization.
External (shareholders, govern-ment, auditors)	Complete and traceable compliance.

RedBridge Software (Belgium)
Kardinaal Mercierplein 2
2800 Mechelen, Belgium
Tel. +32 15 797306

RedBridge Software (France)
3, Rue du Général De Gaulle
28700 Aunay-Sous-Auneau, France
Tél: +33 2 37 25 31 22

info@redbridgesoftware.com
www.redbridgesoftware.com

RedBridge