

**SIEMENS**

*Ingenuity for life*

# Why ALM and PLM need each other

This paper analyzes the differences and similarities between application lifecycle management (ALM) and product lifecycle management (PLM). It examines why ALM cannot be used for PLM and why PLM cannot be used for ALM, and details the reasons why the two solutions need to work together.

# Contents

<b>Executive summary</b> .....	<b>3</b>
<b>PLM and ALM – Defining the systems</b> .....	<b>4</b>
What is PLM?.....	4
What is ALM?.....	4
Comparing ALM and PLM lifecycle phases .....	5
ALM phases.....	5
PLM phases .....	5
<b>Similarities and differences of PLM and ALM</b> .....	<b>6</b>
Parts versus files .....	6
Browser-based collaboration .....	6
Lean manufacturing and agile software development .....	6
PLM traceability versus ALM traceability.....	6
Version control versus change management.....	8
ALM and PLM similarities and differences .....	8
Key takeaways.....	9
<b>Software in the product manufacturing process</b> .....	<b>10</b>
Key takeaways.....	10
<b>The need to marry PLM and ALM</b> .....	<b>11</b>
<b>Summary</b> .....	<b>12</b>

# Executive summary

Software is quickly surpassing hardware's dominance in the product development process, particularly within technologically complex products and industries such as automotive, aerospace and defense, and medical device manufacturing. Technology manufacturers have typically turned to product lifecycle management (PLM) solutions to accelerate time-to-market, make the process more efficient, improve collaboration and adhere to regulatory compliance requirements.

However, traditional PLM systems have typically managed software as a "part" in the context of the product manufacturing process and have struggled with the management of software's complex development processes. Software has its own distinct lifecycle – with different information to be managed, different collaboration processes and methodologies, different specifications and items – a lifecycle that has historically been insufficiently addressed by traditional PLM solutions.

To successfully deliver high-quality software-driven products to the marketplace, manufacturers must go beyond their conventional standalone PLM or ALM system. They must actively seek out an integrated solution that allows for a complete set of product requirements including hardware and software, and supports the multi-disciplinary collaboration needed to ensure end-to-end management of software components as well as the hardware components.

# PLM and ALM – Defining the systems

## What is PLM?

PLM is the process of managing the entire lifecycle of a product from its conception, through design and manufacture, to service and disposal. PLM integrates people, data, processes and business systems and provides a product information backbone for companies and their extended enterprise.

PLM systems help organizations cope with the increasing complexity and engineering challenges of developing new products for global competitive markets by shortening and simplifying each phase of the product development process. Using PLM solutions, companies can get product to market more quickly, meet increasingly rigorous compliance requirements and industry standards, and achieve greater collaboration and communication across the product development process.

The market category for PLM first emerged in 1985, driven by American Motor Corporation (AMC), as a way to speed up the product development process and better compete in the automotive marketplace.









PLM continues to be highly leveraged in the automotive industry today, but adoption has broadly expanded to other industries such as aerospace and defense, healthcare, medical devices, process manufacturing, and energy.

## What is ALM?

“Application lifecycle management (ALM) is not a product but a process,” states industry analyst firm Ovum, in its Software Lifecycle Management report. Ovum goes on to further define ALM as the process by which information technology (IT) and software development organizations create, deploy and operate software over its full lifecycle.<sup>1</sup>

## Comparing ALM and PLM lifecycle phases

### ALM phases





- 1  **Application project and portfolio management**  
An investment analysis is performed and business case developed prior to the inception of a software project.
- 2  **Project inception and requirements gathering**  
Marketplace information is gathered, potential users/customers of the application are interviewed, and data is gathered to form documented requirements.
- 3  **Requirements management**  
As requirements evolve or change, the requirements document also must evolve to analyze impact on development schedules, delivery date, resources and so on.
- 4  **Design and use-case analysis**  
The underlying architecture of the software code is defined, and various use cases are developed to model the possible user interactions with the final system.
- 5  **Coding**  
Application code is written, or in the case of an enhancement, extended or revised.
- 6  **Testing and QA**  
The software is systematically debugged, performance, load and stress tested, with necessary revisions made to the code.
- 7  **Build release, deploy**  
The final release is compiled, the release is finalized, and the application is deployed to production.
- 8  **Application performance**  
The ongoing maintenance of the application – including enhancements and defect correction throughout the application’s life-cycle until the end-of-life phase.



#### Key takeaway

While software development can follow a cascading waterfall approach, the popular and broad acceptance of the agile development methodology now means that software is created more iteratively - in short, rapid “sprints,” with requirements changing frequently and many ongoing revisions.

### PLM phases

- 1  **Conceive**  
Information is gathered from the marketplace, customer requirements are determined and the product is imagined and technical specifications based on this information are created.
- 2  **Design**  
The product’s initial design is created, refined, tested and validated using tools such as CAD and CAE analysis. This step involves a number of engineering disciplines including mechanical, electrical, electronic, software (embedded) and simulation, as well as domain-specific expertise, for example automotive engineering.
- 3  **Realize**  
At this stage, the product design is complete and the manufacturing method is determined, with this phase addressing tool design, analysis, simulation, and ergonomic analysis.
- 4  **Service and end-of-life**  
In this final phase of the product lifecycle, we enter the service phase, which may involve repair and maintenance, waste management and end of life (disposal, destruction) of the product.



#### Key takeaway

PLM processes are typically waterfall processes with iterative work patterns that accommodate incremental change within the process.

# Similarities and differences of PLM and ALM

To make an effective business case for PLM and ALM integration, it is important to understand the similarities and differences between the two systems. This comparison will allow us to better comprehend the basic orientation of each system, understand how the systems can benefit from each other and identify the key points for a truly effective integration of the two systems.

## Parts versus files

The first area of difference lies in the fact that a PLM system is oriented around management of “parts” used to manufacture a physical product that have requirements, design attributes and changes associated to them. Whereas an ALM system is oriented around the management of software files/items such as requirement documents, pieces of software code, or test cases and the changes to those files as they occur.

## Browser-based collaboration

Software creation and the ALM tools have traditionally resided within the software development organization. However, in recent years ALM data has become valuable to an increasingly wide number of stakeholders. ALM coverage is extending to new disciplines beyond software development, and with a greater number of stakeholders comes an increased need for easy, real-time access to data, typically through a browser interface. An increasing number of ALM vendors today provide browser-based access to their systems, and modern Web 2.0 products such as Polarion® ALM are natively web-based, offering cloud-based and mobile device access.

Leveraging the Polarion ALM browser-based approach users can perform a detailed 3D rendering and bill of materials (BOM) including the embedded software with traceability to a vehicle’s transmission displayed in China from a server located in Stuttgart. Without the browser-based integration and the ALM and PLM interoperability provided with Polarion ALM this kind of capability is extremely difficult.

## Lean manufacturing and agile software development

In the 1990s, the Toyota Production System introduced the concept of lean manufacturing as a management philosophy to improve the value for the customer. Toyota was able to grow from a small company to the largest car automaker in the world by following such a philosophy.

In a nutshell, lean is any production practice that creates value for the customer, that is, something for which the customer is willing to pay. Anything else goes in the wastebasket.

We can argue the agile software development methodology is derived from lean manufacturing, at least from a philosophical perspective: in agile methods, users are at the center of the development universe and they define what is worthy of implementation.

The need to integrate ALM and PLM systems for the sake of integrating product and software development methods is of critical importance for all manufactures. It is a natural evolution for lean manufacturing and agile development to come together in a unique approach for application and product development.

## PLM traceability versus ALM traceability

Another area of difference between PLM and ALM lies in the way the systems define traceability. In a PLM system, traceability is defined as the “part of” decomposition of a complete system. A car is composed of a frame, an axle, four tires and so on. In an ALM system, traceability is defined as the links between files/items belonging to different phases. A change to a requirement may impact a line of code, or require a new test case to be developed to validate the new requirement.

A PLM system will relate and link information to PLM items such as requirements, design objects, materials, tolerances, changes and more. An ALM system, on the other hand, will relate and link information related to software code such as requirements, change requests, test cases, and commit comments.

The existence of software as a “part of” a complete system is a significant driver for bringing PLM and ALM together. Software exists in the traditional PLM world as a single part, but the level of management ends there. Traditional PLM systems’ software management does not push down to the point where that “part” has its own lifecycle complete with a multitude of files/items and changes to those files/items.

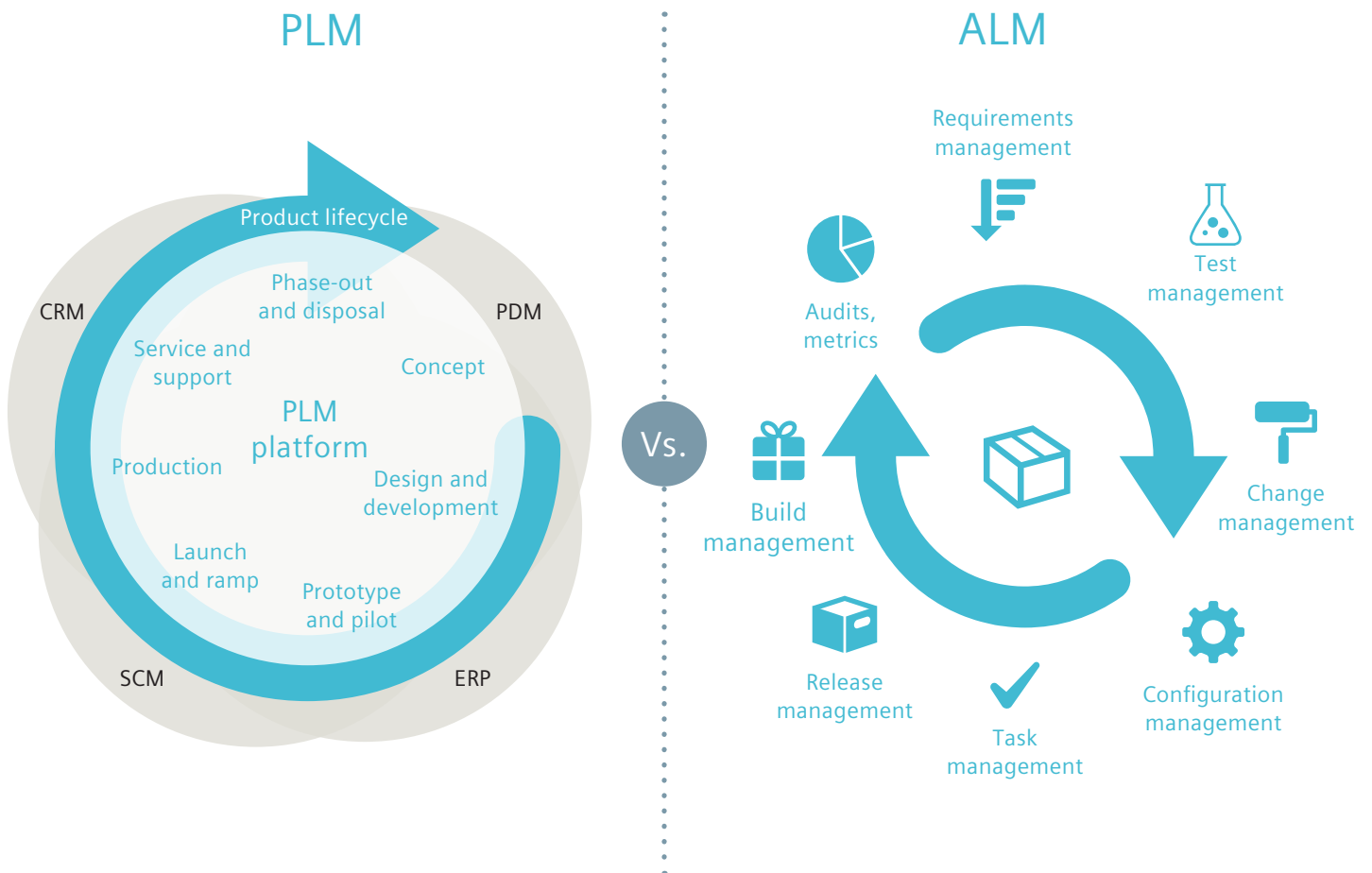
Increasingly, the lack of lifecycle management control over the software “part” creates problems for product manufacturers. Software quality issues lie at the bottom of many costly product failures and drive a significant percentage of product recalls, but traditional approaches to PLM and even PLM and ALM integration frequently lack the ability to identify all the

relationships between hardware and software to get to the bottom of the product defect and software-related issues.

Consider a situation in the automotive industry where in release 3.4.5 of a software component, when parameter X is valued at "3.5" a false signal is created. That software component is leveraged across multiple vehicles. So how does a manufacturer figure out which vehicles have this problem and need to be fixed?

In a recent blog post, Forrester Research application development and delivery analyst Tom Grant points to the gulf that currently exists between many PLM and ALM systems and makes a case for PLM and ALM vendors to close this gap:

"Business processes, such as crafting requirements that encompass both the hardware and software components, are one reason why ALM and PLM need to be stitched together. While product teams already know how to do this (for example, by framing the requirements in terms of 'systems of systems'), the tools they use don't always share the same level of understanding. PLM tools that in theory should accommodate both hardware and software usually fall short when dealing with the digital part of the product. Some elements of ALM, such as source control management, don't even exist in the PLM world."<sup>2</sup>



### Version control versus change management

Traditional PLM systems have a notion of version control – recognizing that a part may change over time – but rarely record or are able to retrieve the reason for change. Because many systems also lack ALM’s notion of traceability, they cannot identify how a change to a software part impacts all other connected items within the system.

### ALM and PLM similarities and differences

Similarities	Differences
Both systems are built around process and core disciplines	<p>ALM is centered on software “files” and prescribes a process to create software applications. The applications consist of multiple item types and complex relationships between software item types which create impact trees.</p> <p>PLM is oriented around “parts” which form a tree structure of “part-of” relationships. The item types and complex relationships between these hardware item types create impact trees.</p>
Both systems incorporate workflow, variant management, test management, requirements, and specification management	<p>ALM deals in the abstract. PLM deals in the concrete. In ALM, software engineers envision, elicit, define, implement, test, and maintain abstract functions.</p> <p>PLM focuses on the delivery of a complete bill of materials with the understanding of product configuration to the production chain. The function of the components in PLM is the product itself.</p>
Both systems allow linking components to each other	<p>In ALM there are many different types of relationships and link types creating dependency hierarchies between software items.</p> <p>In PLM there are many different relationships and link types between hardware items creating dependencies and decomposition hierarchies.</p>
Both systems allow linking information to components	<p>In PLM this information is generally quantitative and can include requirements, targets, design objects, materials, tolerances and more.</p> <p>In ALM the information linked to items is descriptive: textual, mock-ups, user stories, test scenarios, etc.</p>
In both environments there is a wide usage of models	<p>In PLM models follow the “part-of” decomposition and define product design items. PLM models are frequently segmented into different product subsystems: electrical layout, braking subsystem, transmission, interior, etc.</p> <p>In ALM a model follows the functional decomposition by means of diagrams like Entity-Relationship or Object-Oriented.</p>



At the most fundamental level, PLM and ALM systems are different, but they complement each other very well. They were designed at different times to manage very different types of information. They are built to serve different processes, and different types of users. The time has arrived for PLM and ALM to come together. Software plays too great a role in today's technologically advanced products, and the risks of not managing that software are too great for these systems to continue to exist in separate silos.

**Key takeaways**

- Software now plays a critical role in products and is often at the root of product failure. In 1999, a software error in the NASA Mars Climate Orbiter caused the \$125 million spacecraft – key to the Mars exploration program – to enter into the Martian atmosphere too low and too fast. The craft has never been heard from again.
- PLM and ALM systems are designed to manage very different things and include vastly different features for their users. If a manufacturer is building a product with a significant software portion, both PLM and ALM are needed and these systems must work together.
- PLM and ALM systems define traceability differently. In PLM traceability describes the decomposition of a product into various parts and components across the lifecycle. In ALM, traceability describes the linkages between items across the various stages of the software development process. Here again, both PLM and ALM are needed and these systems must be integrated.

# Software in the product manufacturing process

Today it takes dozens of microprocessors running 100 million lines of code to get a premium car out of the driveway. And this software is only going to become more complex.<sup>3</sup>

Software engineering is increasingly becoming the dominant force in consumer and industrial product manufacturing. Siemens now employs more software engineers in its high-tech business than Microsoft, Oracle, or SAP.

The rising and critical importance of software within products adds new complexity to the product development process. No longer are manufacturers simply responsible for the hardware; they now must develop additional processes and procedures for the development of complex, embedded software systems. As the quality and performance of this software in some circumstances (such as within a medical device, or in the control of an aircraft or vehicle) can mean life or death for the user, the software process is rigorously controlled and regulated with compliance monitored through government bodies. Failure to meet these standards can result in hefty penalties, or shut-down of manufacturing operations.<sup>4</sup>

“The importance of system-centric product development and system engineering is continuing to grow, as manufacturers increasingly include software to deliver product function. The shift from loosely orchestrated mechanical design, electronics design and software development to a more tightly executed system-centric approach requires time-consuming changes to new product development activities, processes, organizations and culture. Manufacturers that do not make the shift in a timely way will struggle competitively.”

Gartner - Marc Halpern, Janet Suleski: *Predicts 2013: Product Design and Life Cycle Management*.

## Key takeaways

- The US Air Force's F-35 Joint Strike Fighter incorporates approximately 5.7 million lines of code.
- The average medical device now has one million lines of code and that number is doubling every couple of years.
- Software is an increasingly dominant driver of industrial product innovation.

# The need to marry PLM and ALM

Earlier in this paper, we discussed the increasing amount of software within manufactured products – and in particular, within technologically complex products, such as automobiles, medical devices and aircraft.

This increasing – and increasingly dominant – presence of software adds a new dimension of complexity within the product engineering and manufacturing process. That increased complexity must be managed.

On the surface the methods for managing a product's lifecycle (PLM), and the lifecycle of a software application (ALM) seem to be quite similar. Both PLM and ALM systems are built around an integrated process and set of core disciplines. But the similarities end there.

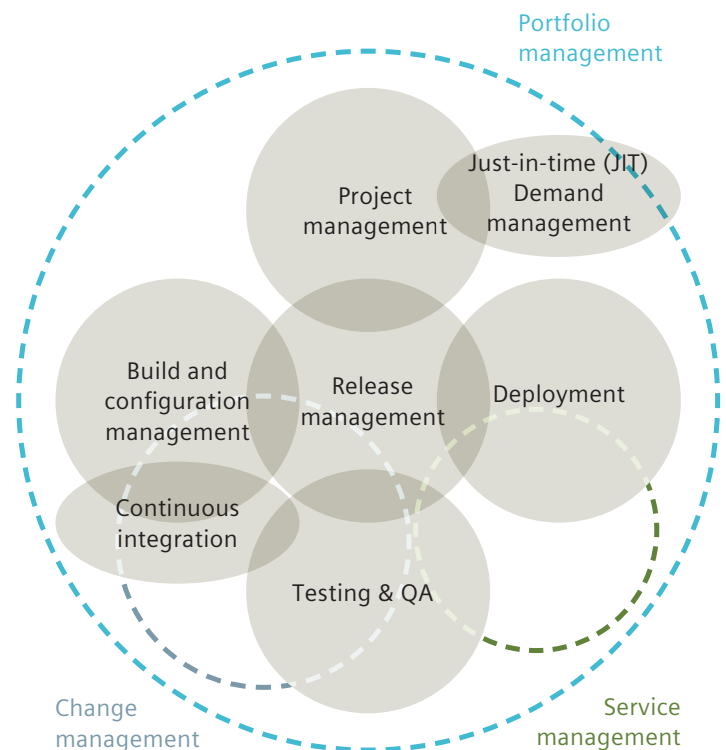
Some CIOs and IT managers with significant investments in PLM systems have tried to leverage a PLM system to manage software. A PLM system can manage product-related workflows, specifications, designs, and versions – so why not software as well?

But requiring a PLM system to manage the complexity and file management of the software development process – inclusive of iterative development cycles, changing requirements, traceability of items and relationships between items and so on – pushes the limits of many PLM systems well beyond their boundaries. The management of software development processes is a job better suited to ALM, a software-centric discipline specifically oriented around exactly the tasks outlined above.

After exploring the many differences between ALM and PLM approaches and toolsets, we can state:

1. The need to manage software and product lifecycles in an integrated way is now more urgent than ever
2. Traditional PLM toolsets are not well suited to manage software development
3. ALM toolsets are not well suited to manage product development

So the question remains, how does a manufacturer best manage the growing amount of software that now exists as a component within a part within a product? Ideally that problem is solved through integration and interoperability between a PLM system and an ALM system.



Source: The Forrester Wave™: Application Life-Cycle Management, Forrester Research, Inc.

# Summary

Software is increasingly overtaking hardware's traditional dominance in product development. This is particularly true for technologically sophisticated products (such as automotive vehicles, aircraft, medical devices and smartphones).

Systems and product engineers need to actively seek out tools beyond their conventional PLM system that allow for multi-disciplinary collaboration – especially with software engineering counterparts – that ensure end-to-end management of software components as well as hardware components.

As software and product development disciplines are significantly different, there is no way to use just PLM or just ALM in systems engineering. Manufacturers have to use both, so PLM and ALM must come together in an integrated fashion that allows all disciplines and all design processes to share and link product and software requirements, collaborate more closely by establishing the cross-domain relationships needed to fully assess the impact of change and to view and access that information using the system and tools they are most comfortable with. Successful manufactures will require the type of integration and interoperability that exist with Teamcenter and Polarion ALM.

## References

1. Ovum, Software Lifecycle Management 2011.
2. Forrester Research, Inc., Blog post, August 2012 – "ALM and PLM: Make it Work People", Tom Grant.
3. This Car Runs on Code – IEEE Spectrum, February 2009.
4. Getting Better Software into Manufactured Products – McKinsey Quarterly, March 2006.

## Siemens PLM Software

### Headquarters

Granite Park One  
5800 Granite Parkway  
Suite 600  
Plano, TX 75024  
USA  
+1 972 987 3000

### Americas

Granite Park One  
5800 Granite Parkway  
Suite 600  
Plano, TX 75024  
USA  
+1 314 264 8499

### Europe

Stephenson House  
Sir William Siemens Square  
Frimley, Camberley  
Surrey, GU16 8QD  
+44 (0) 1276 413200

### Asia-Pacific

Suites 4301-4302, 43/F  
AIA Kowloon Tower,  
Landmark East  
100 How Ming Street  
Kwun Tong, Kowloon  
Hong Kong  
+852 2230 3308

## About Siemens PLM Software

Siemens PLM Software, a business unit of the Siemens Digital Factory Division, is a leading global provider of product lifecycle management (PLM) and manufacturing operations management (MOM) software, systems and services with over 15 million licensed seats and more than 140,000 customers worldwide. Headquartered in Plano, Texas, Siemens PLM Software works collaboratively with its customers to provide industry software solutions that help companies everywhere achieve a sustainable competitive advantage by making real the innovations that matter. For more information on Siemens PLM Software products and services, visit [www.siemens.com/plm](http://www.siemens.com/plm).

For more information about Polarion ALM, please visit [www.siemens.com/polarion](http://www.siemens.com/polarion)

## [www.siemens.com/plm](http://www.siemens.com/plm)

© 2016 Siemens Product Lifecycle Management Software Inc. Siemens and the Siemens logo are registered trademarks of Siemens AG. ALM, D-Cubed, Femap, Fibersim, Geolus, GO PLM, I-deas, JT, NX, Parasolid, Polarion, Solid Edge, Syncrofit, Teamcenter and Tecnomatix are trademarks or registered trademarks of Siemens Product Lifecycle Management Software Inc. or its subsidiaries in the United States and in other countries. All other logos, trademarks, registered trademarks or service marks belong to their respective holders.

55241-A11 9/16 H