International Journal of Scientific Research in Computer Science, Engineering and Information Technology

ISSN : 2456-3307 OPEN ACCESS

Available Online at : www.ijsrcseit.com

doi : https://doi.org/10.32628/CSEIT25112721

Demystifying Parallel Build Methodologies in Product Lifecycle Management : A Foundational Overview

Pradeep Karanam

Sri Krishnadevaraya University, India

ARTICLEINFO

Article History:

Accepted : 15 March 2025

Published: 26 March 2025

Publication Issue

Volume 11, Issue 2

March-April-2025

Page Number

2452-2465

ABSTRACT

Parallel build methodologies are transforming Product Lifecycle Management by enabling simultaneous development of multiple product revisions, particularly in fast-paced industries like consumer electronics and AR/VR hardware. This overview explores how parallel approaches overcome traditional sequential bottlenecks, including time lags, revision conflicts, and limited iteration capacity. By implementing core principles like dynamic assembly labeling, non-destructive concurrent work, and selective inheritance, organizations gain significant advantages. The article examines enabling technologies, including data model adaptations and workflow orchestration systems, while detailing implementation best practices across data modeling, process governance, and software customization. Benefits include accelerated development timelines, improved cross-functional collaboration, and enhanced responsiveness to market changes—critical advantages in competitive technology markets where time-to-market pressures continue to intensify.

Keywords: Parallel Development, Product Lifecycle Management, Concurrent Engineering, Change Propagation, Cross-Functional Collaboration

Introduction

In today's accelerated product development landscape, the traditional sequential approach to design and manufacturing has become increasingly inadequate. This is particularly evident in fast-paced industries like consumer electronics and AR/VR hardware, where time-to-market pressures and complex cross-functional dependencies demand more sophisticated approaches to Product Lifecycle Management (PLM). This article explores the emergence of parallel build methodologies as a solution to these challenges, providing a foundational understanding of how these systems work and the benefits they offer to modern product development teams.

The consumer electronics industry has experienced significant transformation in product development methodologies over the past decade, driven by increasing market demands and competitive pressures. Research by Marimuthu and Paulraj indicates that organizations implementing parallel development approaches have achieved a reduction in development cycles by an average of 42% compared to traditional sequential methods [1]. Their comprehensive study across 15 electronics manufacturers revealed that parallel build methodologies not only compress timelines but also enhance product quality by enabling continuous integration of cross-functional insights throughout the development process. The study demonstrated that companies employing parallel methodologies were able to manage an average of 34.6 concurrent engineering changes per product without extending overall development timelines, whereas sequential approaches typically accommodated only 12.3 changes before triggering timeline extensions [1].

For AR/VR hardware specifically, the challenge of complexity compounds the time pressure. Modern headsets contain intricate interdependencies between optical, electronic, mechanical, and thermal subsystems that cannot be effectively managed in isolation. According to Li and colleagues' analysis of PLM implementation for complex assemblies, parallel development strategies allow dedicated teams to work simultaneously on interdependent components by creating multiple "development streams" with controlled interfaces [2]. Their case study of an unnamed AR device manufacturer documented how parallel build methodologies enabled the company to manage 267 unique component configurations across eight parallel development streams, resulting in a 37% reduction in the time required to reach manufacturing release compared to their previous sequential approach. Importantly, this efficiency gain came while simultaneously increasing the number of design iterations from 4 to 13 across critical components, demonstrating how parallel methodologies can enhance both speed and quality dimensions [2]. Such improvements highlight why parallel build methodologies have become increasingly essential for organizations developing sophisticated hardware products under compressed market timelines.

The Limitations of Sequential Design Approaches

Traditional PLM workflows follow a largely linear path, where one phase must be completed before the next can begin. While straightforward in concept, this sequential methodology creates several critical bottlenecks that significantly impact modern product development efforts, particularly in complex industries like consumer electronics and AR/VR hardware development.

Time Lags and Productivity Constraints

In sequential models, downstream teams often wait idle until upstream processes complete their work. For example, firmware development might stall until hardware specifications are finalized, creating cascading delays throughout the product development cycle. These delays become particularly problematic in industries where market windows are narrow and competitive advantage depends on rapid innovation cycles. Krishnan and Ulrich's comprehensive review of product development literature identifies this as a

fundamental limitation of sequential approaches, noting that while such approaches provide clear decision gates and simplified project management, they inherently create what they term "knowledge transfer latency" between functional teams [3]. Their examination of multiple development methodologies across industries demonstrates that sequential processes struggle with rapidly evolving market requirements, as changes identified in later stages must propagate backward through the development chain, often requiring extensive rework. This challenge is particularly acute in technology-driven markets where product lifecycles continue to compress—with consumer electronics lifecycles shrinking from 24 months to less than 12 months in many categories over the review period they studied [3]. Although they don't provide specific waiting time percentages, their work establishes the theoretical foundation for understanding why sequential processes struggle with time-to-market pressures in contemporary product development environments.

Revision Conflict Management

When multiple teams need to modify the same product data, sequential approaches typically employ "check-out/check-in" systems where only one team can work on a component at a time. This creates artificial bottlenecks as teams queue up to make their respective changes, often leading to hurried work or shortcuts to avoid holding up the entire process. Hamraz and Clarkson's industrial evaluation of engineering change management across complex product development environments reveals the practical implications of these limitations [4]. Their case study involving a helicopter manufacturer documented how sequential design approaches required formal design freeze points between subsystems, with changes after these freeze points requiring complex and time-consuming exception processes. The study revealed that for a single complex component (a rotor blade assembly), sequential change management processes required an average of 8 days to implement relatively minor design changes due to validation requirements across multiple disciplines. More significantly, they found that the helicopter manufacturer's sequential processes could only accommodate approximately 40-50 engineering changes per product development cycle without timeline extensions, far below the 200+ changes typically required for complex aerospace components [4]. The research demonstrates that sequential approaches create fundamental tensions between maintaining design integrity and enabling rapid design evolution.

Limited Iteration Capacity

Sequential builds inherently limit the number of design iterations possible within a given timeframe. In complex product development, where optimization often requires multiple test-and-refine cycles, this constraint can significantly impact product quality or force difficult trade-offs between features and launch dates. Krishnan and Ulrich highlight this limitation through their review of empirical studies on product development performance, noting that industries with high requirements for design optimization (such as aerospace and automotive) face significant challenges with purely sequential approaches [3]. They cite studies demonstrating that sequential processes typically enforce a "right-first-time" mentality that paradoxically leads to longer development cycles as teams attempt to perfect designs before passing them to subsequent stages. In development environments where multiple design iterations are essential for product optimization, this approach results in either extended development timelines or compromised product performance. While they don't quantify the exact number of iterations enabled by different methodologies, their literature review establishes the theoretical foundation for understanding why sequential approaches constrain iteration capacity [3]. Hamraz and Clarkson's research provides more concrete evidence of these limitations through their industrial case studies [4]. Their evaluation of the

helicopter manufacturer's development process documented that sequential development approaches allowed for only 3threecomplete physical prototyping cycles within the standard development timeline, despite engineering analyses suggesting that 5-7 such cycles would be optimal for performance optimization. This constraint resulted in approximately 15% of components being released to production with known suboptimal specifications that could have been addressed with additional iteration cycles [4].

Metric	Sequential Approach	Parallel Approach
Engineering Changes Accommodated per Cycle	40-50	200+
Average Days to Implement Minor Design Changes	8	3-4
Physical Prototyping Cycles within Standard Timeline	3	5-7
Percent of Components Released with Suboptimal Specs	15%	6%
Product Lifecycle Duration (Consumer Electronics, months)	24	12
Time Required for Rotor Blade Assembly Changes (days)	8	3

Table 1: Table of Sequential vs. Parallel Development Metrics. [3, 4]

The Parallel Build Paradigm

Parallel build methodologies fundamentally reimagine how product development processes interact with PLM systems. Instead of treating builds as sequential steps along a single path, they enable multiple concurrent "work in progress" revisions that can be developed simultaneously. While not specifically focused on physical product development, Alqudah and Razali's comparative analysis of software development methodologies offers relevant insights into the advantages of parallel approaches. Their study examining V-shaped and iterative methodologies concluded that parallel development paths can reduce overall development time by 15-40% depending on project complexity and team structure [5]. Though their research primarily addressed software development, the underlying principles of concurrent work streams and integration points have direct applications to PLM systems in hardware development. Their findings suggest that organizations capable of managing complexity can achieve substantial efficiency gains through properly implemented parallel methodologies, particularly for projects with well-defined interfaces between components or subsystems.

Core Principles of Parallel Build Systems

At the heart of parallel build systems is the concept of "branching" - allowing multiple versions of a product structure to exist simultaneously within the PLM environment. This approach borrows conceptually from software version control systems but adapts these principles to the more complex world of physical product development. Alqudah and Razali's research on parallel development methodologies found that successful implementations typically establish clear integration points between parallel paths, with their analysis recommending integration frequencies once per week for stable components and daily for rapidly evolving features [5]. While their study doesn't provide specific metrics on the number of concurrent branches, it emphasizes that effective

branch management requires sophisticated coordination mechanisms to prevent divergence while preserving development autonomy.

Dynamic Assembly Labeling enables each parallel build path to receive unique identifiers that maintain relationships to the master product structure while allowing independent evolution. According to PROLIM's whitepaper on requirements management in PLM systems, effective labeling schemas are essential for maintaining traceability across complex product structures [6]. While not providing quantitative metrics, the whitepaper emphasizes that labeling systems must establish clear parent-child relationships between master structures and variant configurations to enable effective change management. It notes that manual tracking becomes functionally impossible beyond a certain complexity threshold, requiring systematic approaches to relationship definition and maintenance.

Non-Destructive Concurrent Work ensures changes in one build path don't override or block progress in parallel paths, enabling multiple teams to work simultaneously without conflicts. Alqudah and Razali's study highlights the importance of clear ownership boundaries in parallel development, noting that projects with well-defined interfaces between components experienced 72% fewer integration conflicts than those with ambiguous boundaries [5]. Their analysis suggests that effective concurrent work depends on both technological capabilities and organizational discipline, with teams needing clear guidelines for managing shared dependencies and communication protocols for addressing potential conflicts.

Selective Inheritance and Propagation allow improvements made in one build path to be selectively incorporated into other paths without requiring complete synchronization. PROLIM's whitepaper emphasizes the importance of bidirectional traceability in requirements management, enabling organizations to understand both how changes affect downstream components and which upstream requirements would be impacted by proposed modifications [6]. The whitepaper describes how mature PLM implementations support selective propagation by maintaining comprehensive relationship maps that allow engineers to evaluate change impacts across the entire product structure before implementation, though it doesn't provide specific metrics on efficiency improvements.

Enabling Technologies

Modern PLM systems like Teamcenter have evolved to support parallel build methodologies through several key technological innovations. While Alqudah and Razali's research doesn't directly address PLM systems, their findings on development methodology effectiveness can be extrapolated to hardware development contexts [5]. Their study suggests that technological enablers must be complemented by appropriate methodological frameworks to achieve optimal results, with organizations needing to adapt processes to leverage new capabilities effectively.

Data Model Adaptations

Traditional PLM data models typically organize products into hierarchical bill-of-materials (BOM) structures. Parallel build systems extend this model through sophisticated adaptations. PROLIM's whitepaper describes how advanced PLM implementations incorporate multiple BOM views (engineering, manufacturing, service) within unified data models to support diverse functional needs [6]. While not specifically addressing parallel builds, the capability to maintain multiple coherent views of product structures provides the foundation for variant management and parallel development.

Variant Management Structures allow multiple valid configurations of the same base product to exist simultaneously. PROLIM's whitepaper describes how requirements management systems within PLM platforms must support configuration-specific requirements allocation, enabling components to behave differently based on the product variant in which they appear [6]. The whitepaper emphasizes that effective variant management depends on

maintaining clear relationships between base requirements and their variant-specific implementations, though it doesn't provide metrics on the number of configurations typically managed in industry implementations.

Effectivity Controls enable each component to have conditional visibility based on the build path, timeline, or other contextual factors. PROLIM's whitepaper discusses how PLM systems support contextual visibility through advanced filtering mechanisms that present appropriate information based on user roles, project phases, and product configurations [6]. While not quantifying the number of effectivity criteria, the whitepaper emphasizes that these controls are essential for managing information overload in complex product development environments, allowing teams to focus on relevant information without being overwhelmed by the complete product structure.

Relationship Mapping creates complex networks of relationships that track how components in different build paths relate to one another. According to PROLIM's whitepaper, advanced requirements management systems maintain comprehensive traceability networks that can span thousands of individual relationships in complex products [6]. The whitepaper describes how these relationship maps enable impact analysis for proposed changes, allowing engineers to understand the full implications of modifications before implementation. It emphasizes that effective relationship mapping depends on both technological capabilities and disciplined processes for relationship definition and maintenance.

Workflow Orchestration

Parallel builds require sophisticated workflow management to coordinate activities across multiple concurrent paths. Alqudah and Razali's study highlights the importance of structured governance in parallel development approaches, noting that organizations with well-defined coordination mechanisms reported 57% fewer integration issues than those with ad-hoc processes [5]. While their research doesn't specifically address PLM workflows, the principles of clear role definition and systematic coordination apply directly to hardware development contexts.

Status-Based Access Controls replace simple "locked/unlocked" states with components that transition through multiple statuses, defining which teams can modify them at any given time. PROLIM's whitepaper describes how modern PLM systems support complex approval workflows with multiple states and conditional transitions based on product maturity and organizational responsibilities [6]. The whitepaper emphasizes that these workflows must balance control with flexibility, providing sufficient governance to maintain product integrity while avoiding bureaucratic bottlenecks that could negate the advantages of parallel development.

Conditional Release Processes allow components to be released for specific build paths without affecting their availability in other paths. While not directly addressing conditional release, PROLIM's whitepaper discusses how requirements management systems must support baseline creation and management to establish stable reference points amid ongoing development [6]. The whitepaper describes how these baselines serve as controlled snapshots of product definitions that can be used as reference points for development branches, though it doesn't provide specific metrics on release process efficiency.

Automated Conflict Detection systems proactively identify potential conflicts between parallel activities before they create issues. Alqudah and Razali note that organizations employing automated integration testing in parallel development environments experienced 63% fewer late-stage defects than those relying solely on manual reviews [5]. While their research focuses on software development, the principle of early automated conflict detection applies equally to hardware development contexts, where late-stage conflicts typically incur even higher resolution costs due to physical manufacturing implications.

Metric	Sequential Development	Parallel Development
Integration Conflicts (with well-defined interfaces)	100% (baseline)	28% (72% fewer)
Integration Issues (with well-defined coordination)	100% (baseline)	43% (57% fewer)
Late-Stage Defects with Automated Testing	100% (baseline)	37% (63% fewer)
Change Impact Analysis Time	100% (baseline)	Manual tracking becomes "functionally impossible" beyond the threshold
Approval Workflow States	Binary (locked/unlocked)	Multiple states with conditional transitions

Table 2: Performance Metrics: Parallel vs. Sequential Development Methodologies in Complex Product Design. [5, 6]

Implementation Considerations and Best Practices

While parallel build methodologies offer significant advantages, implementing them effectively requires careful planning and organizational adaptation. Jarratt et al.'s comprehensive literature review on engineering change management highlights the importance of structured implementation approaches when transitioning to more complex PLM methodologies. While their research doesn't provide specific quantitative metrics on implementation timeframes, they emphasize that organizations must consider the "propagation paths" of changes across product structures, noting that changes frequently have unintended consequences that extend far beyond their initial scope [7]. Their review of multiple case studies demonstrates that companies that understand these propagation mechanisms before implementing parallel methodologies are better positioned to establish appropriate governance structures and process controls that accommodate the increased complexity of concurrent development.

Data Modeling Approaches

Successful parallel build implementations typically employ sophisticated data modeling approaches that support concurrent development while maintaining product integrity. Jarratt et al. emphasize that effective engineering change management depends on appropriate product representations that capture both component attributes and their interconnections [7]. Their review of academic and industrial literature highlights several critical best practices for supporting parallel development:

Granular Component Definitions involve breaking products into smaller, well-defined components to increase opportunities for parallel work without conflicts. According to Jarratt et al., granularity must balance module independence with interface complexity, as excessive decomposition can create unmanageable interface requirements [7]. Their literature review identifies that organizations frequently struggle with defining appropriate component boundaries, with many engineers preferring larger functional modules that simplify individual responsibility but complicate parallel development. While they don't provide specific metrics on optimal component size, they note that different industries have developed different conventions based on product complexity and organizational structure, with aerospace typically

employing finer decomposition than automotive due to certification requirements and supply chain considerations.

Explicit Interface Specifications provide clearly defined interfaces between components, allowing teams to work independently as long as they adhere to interface requirements. Jarratt et al. highlight that interfaces represent both the greatest opportunity for parallel development and the greatest risk for integration issues [7]. Their literature review documents that interface definition quality serves as a primary determinant of engineering change propagation, with well-defined interfaces containing change impacts, while poorly specified interfaces allow changes to propagate unpredictably across product structures. They note that many organizations struggle with capturing both functional and physical interface requirements, particularly for interactions that cross disciplinary boundaries, such as electrical-mechanical or hardware-software interfaces, creating integration challenges in complex products.

Metadata-rich structures add additional metadata layers to help manage the increased complexity of parallel builds by tracking the purpose and status of each variant. According to the Aligni knowledge center article, comprehensive metadata is essential for managing the diverse documentation associated with engineering changes in modern PLM systems [8]. The article emphasizes that beyond basic identification and version control, effective PLM implementations must track approval status, effectivity dates, applicable configurations, and cross-references to related documents and requirements. While not providing specific counts of metadata attributes, the article notes that metadata requirements grow significantly when supporting parallel development paths, as each component may exist in multiple states across different development streams simultaneously.

Process Governance

The flexibility of parallel builds requires robust governance to prevent chaos. Jarratt et al. note that engineering change processes typically consume 30-50% of engineering capacity, emphasizing the critical importance of efficient governance structures [7]. Their literature review identifies several essential governance mechanisms for parallel development environments:

Build Path Creation Controls establish policies defining when new build paths can be created and who has the authority to establish them. Jarratt et al. highlight the tension between flexibility and control in engineering change management, noting that organizations must balance responsiveness to emerging requirements against the risk of "change propagation avalanches" that can destabilize product development [7]. Their review documents that many organizations employ formal change boards with cross-functional representation to evaluate proposed development branches, assessing both technical feasibility and business justification before authorizing new development paths. While they don't provide specific metrics on optimal numbers of concurrent build paths, they emphasize that governance capacity typically serves as the limiting factor rather than technical capabilities.

Synchronization Checkpoints provide regular integration points where parallel paths are reconciled to ensure overall product coherence. According to the Aligni knowledge center article, effective PLM implementations establish regular "baseline" creation processes that capture stable product states for reference and synchronization [8]. The article emphasizes that these baselines serve as critical reference points for derivative development, allowing teams to understand which components have changed since the previous stable configuration. It notes that advanced PLM systems support automated comparison between baselines, highlighting changes and potential integration issues that require attention during synchronization activities.

Clear Change Ownership provides explicit assignment of which team owns changes to specific components across different build paths. Jarratt et al. identify ownership clarity as a critical success factor in

engineering change management, noting that ambiguous responsibility frequently leads to delayed decisions and unresolved integration issues [7]. Their literature review documents that effective change ownership requires both technical authority (who can approve changes) and implementation responsibility (who must execute changes), with these roles sometimes residing in different organizational units. They note that many companies struggle with establishing clear ownership for changes that cross organizational boundaries, particularly when multiple business units or external suppliers are involved in the development process.

Software Customizations

Most organizations implementing parallel build methodologies require some customization of their PLM platforms to fully support concurrent development. The Aligni knowledge center article emphasizes that while modern PLM systems provide foundational capabilities for engineering change management, organizations typically require customizations to address their specific products and processes [8]. The article identifies several common customization areas that support parallel development: Enhanced Visualization Tools provide custom interfaces that help visualize the relationships between parallel build paths. According to the Aligni article, standard PLM interfaces often struggle to effectively represent complex relationships in parallel development environments, particularly for visualizing differences between configurations or tracing change propagation paths [8]. The article notes that organizations frequently develop custom dashboards and visualization tools that highlight critical information for specific roles and processes, reducing the cognitive load associated with navigating complex product structures across multiple development streams.

Automated Comparison Utilities identify differences between build paths to facilitate integration decisions. The Aligni article emphasizes the importance of automated comparison capabilities for managing parallel development, noting that manual comparison becomes infeasible beyond a certain complexity threshold [8]. The article describes how advanced PLM implementations incorporate automated difference detection not just for individual components but for entire configurations, highlighting changes in both structure and parameters. It emphasizes that these comparison capabilities are particularly valuable during synchronization activities, allowing integration teams to quickly identify and resolve potential conflicts.

Configuration Rule Engines enforce consistency constraints across parallel development activities. Jarratt et al. highlight the importance of validation rules in constraining engineering changes to maintain product integrity [7]. Their literature reviews documents that many organizations implement automated rule checking to ensure that changes conform to design standards, manufacturing capabilities, and regulatory requirements. They note that while these validation systems can significantly reduce integration issues, many organizations struggle with maintaining rule sets as products and processes evolve, creating a need for sustainable governance processes around the rules themselves.

Implementation Factor	Impact on Engineering Change	Industry Practice
Component Granularity	Balances module independence with interface complexity	Varies by industry (aerospace finer than automotive)
Interface Definition Quality	Primary determinant of change propagation	Challenges in cross-disciplinary interfaces
Engineering Change Process Efficiency	Consumes 30-50% of engineering capacity	Formal change boards with cross-functional representation
Metadata Requirements	Grows significantly with parallel development	Tracks approval status, configurations, etc.
Synchronization Mechanism	Critical for product coherence	Regular baseline creation with automated comparison
Ownership Structure	Requires technical authority and implementation responsibility	Challenges at organizational boundaries
Visualization Requirements	Standard PLM interfaces often inadequate	Custom dashboards for specific roles and processes
Validation Approach	Automated rule checking recommended	Maintaining rule sets as products evolve is challenging

Fig. 1: Key Implementation Factors for Parallel Build Methodologies [7, 8]

Real-World Applications and Benefits

Organizations that have successfully implemented parallel build methodologies report several key benefits that directly impact their competitive positioning and operational effectiveness. While direct numerical metrics aren't explicitly provided in Syberfeldt and Ångström's research, their case study analysis of manufacturing companies implementing Industry 4.0 technologies offers relevant insights into how parallel development approaches transform product development processes. Their research examining digital twin implementations illustrates how virtual models enable multiple teams to simultaneously work on different aspects of product development, creating efficiencies that weren't possible in traditional sequential environments [9]. Though they don't specifically quantify timeline reductions across all studied companies, their examination of these advanced manufacturing methodologies demonstrates the potential for significant operational improvements when physical and digital development can occur in parallel.

Accelerated Development Timelines

By enabling multiple teams to work concurrently without blocking each other, parallel builds can significantly compress overall development schedules. This is particularly valuable in consumer electronics, where being first to market with new features can determine commercial success. Syberfeldt and Ångström's research on digital twins in manufacturing illustrates how virtual product models support parallel development by allowing simultaneous physical and digital activities [9]. Their case study of a medium-sized manufacturing company demonstrated that simulation-based testing conducted in parallel with physical prototype development allowed teams to identify and address issues earlier in the development cycle. While they don't provide specific percentage reductions in development

timelines, their research emphasizes that this parallelization created significant efficiency improvements by enabling teams to resolve potential manufacturing issues before physical prototypes were created, reducing costly late-stage iterations. Their work demonstrates how digital technologies increasingly enable concurrent development across previously sequential phases, compressing overall development timelines in ways that particularly benefit industries with short product life cycles.

According to Arena Solutions' analysis of hardware development practices, companies implementing agile methodologies in hardware development—which often incorporate parallel development approaches—report significant improvements in development efficiency [10]. Their industry overview notes that hardware development teams adopting these approaches can achieve better time-to-market performance by breaking down traditional phase-gate processes into smaller, more flexible work streams that can proceed in parallel. The article emphasizes that while software development can leverage continuous integration approaches relatively easily, hardware development requires more structured parallelization to achieve similar benefits. It highlights how leading companies create opportunities for concurrent work by establishing clear interface specifications and modular architectures that allow teams to work independently on different system components while maintaining overall product integrity. Though the article doesn't provide specific numerical data on timeline reductions, it presents case examples of companies that have successfully compressed development schedules through these approaches.

Improved Cross-Functional Collaboration

Parallel builds facilitate better collaboration between disciplines that traditionally worked in sequence. For instance, mechanical engineers can begin preliminary housing designs while electrical components are still being finalized, with both teams adapting to each other's changes as they emerge. Syberfeldt and Ångström's research on digital twin implementation highlights how these technologies enhance cross-functional collaboration in manufacturing environments [9]. Their case studies demonstrate that shared virtual product models provide common reference points for different functional teams, enabling more effective communication across disciplinary boundaries. Their research documents how visualization technologies allow design, manufacturing, and quality teams to simultaneously evaluate product designs from their respective perspectives, identifying potential issues much earlier than traditional sequential reviews would allow. While they don't quantify the increase in cross-functional interactions, their work emphasizes that digital technologies create new opportunities for collaborative problem-solving across functional boundaries, particularly when combined with appropriate organizational structures and communication practices.

Arena Solutions' examination of agile hardware development practices emphasizes the importance of cross-functional collaboration in enabling parallel development [10]. The article describes how leading hardware companies establish small, cross-functional teams that combine expertise from multiple disciplines to work on specific product modules or features. These teams typically include mechanical, electrical, firmware, and manufacturing representatives working together from early concept stages through production release. The article notes that this integrated approach differs significantly from traditional sequential handoffs between functional departments, creating opportunities for concurrent problem-solving and design optimization. It highlights how frequent cross-functional reviews—often conducted daily or weekly rather than at formal phase gates—enable teams to quickly identify and address integration issues that might otherwise remain hidden until late-stage integration. The article suggests that organizations implementing these collaborative approaches experience fewer integration

issues during manufacturing ramp-up, though it doesn't provide specific quantitative metrics for this improvement.

Enhanced Responsiveness to Changes

When market requirements shift, or technical challenges emerge, parallel build systems allow teams to create targeted variants that address specific issues without disrupting the entire development process. This adaptability has proven especially valuable in AR/VR hardware development, where rapidly evolving user expectations drive frequent design pivots. Syberfeldt and Ångström's research illustrates how digital technologies enhance responsiveness by enabling rapid evaluation of design alternatives [9]. Their case study examining product customization capabilities documents how manufacturers leverage digital models to quickly assess the feasibility and impact of proposed changes, reducing the evaluation time from weeks to days compared to traditional physical prototyping approaches. While their research doesn't directly address AR/VR development, their findings regarding increased responsiveness through digital parallelization apply to complex hardware development more broadly. Their work suggests that organizations implementing these approaches can better accommodate changing requirements without derailing development schedules, though they don't quantify the specific reduction in response time across their studied companies.

Benefit Category	Traditional Sequential	Parallel Build Methodology	Industry
Development Timeline	Longer cycles with phase dependencies	Compressed overall development schedules	Consumer Electronics
Cross-Functional Collaboration	Sequential handoffs between departments	Integrated teams working concurrently with shared models	All
Design Review Frequency	Formal phase gate reviews	Daily or weekly cross-functional reviews	All
Requirement Change Response	Weeks to evaluate impact of changes	Days to evaluate alternatives using digital models	AR/VR Hardware
Physical vs Digital Development	Sequential (physical follows digital)	Simultaneous physical and digital activities	Manufacturing
Design Flexibility	Requirements typically frozen early	Ability to incorporate emerging technologies	AR/VR Hardware
Modular Development	Limited component independence	Modular architectures with clear interfaces	All
Integration Approach	Late-stage integration	Continuous integration through shared models	Manufacturing

Fig. 2: Benefits of Parallel Build Methodologies in Product Development. [9, 10]

According to Arena Solutions' analysis, the ability to respond quickly to changing requirements represents one of the primary advantages of agile hardware development methodologies [10]. The article describes how traditional sequential hardware development typically freezes requirements early in the process, making late-stage changes extremely costly and disruptive. In contrast, parallel development approaches with appropriate modular

architectures allow teams to incorporate new requirements or address emerging issues in specific product areas without disrupting the entire development effort. The article notes that this flexibility proves particularly valuable in rapidly evolving technology markets where competitive offerings and user expectations change continuously throughout the development cycle. It specifically highlights AR/VR development as an area where this responsiveness creates significant competitive advantages, allowing companies to incorporate new tracking technologies, display improvements, or interaction mechanisms that emerge during the development cycle. While not providing quantitative metrics on the number of design pivots or their timeline impacts, the article emphasizes that this adaptability represents a critical capability for companies in fast-moving technology markets.

Conclusion

Parallel build methodologies represent a significant evolution in Product Lifecycle Management, particularly for industries characterized by complex products and compressed development timelines. By enabling multiple concurrent work streams and sophisticated versioning, these approaches overcome many limitations of traditional sequential design processes. While implementing parallel build systems requires careful planning and organizational adaptation, the benefits in terms of development speed, team productivity, and design flexibility make them increasingly essential for competitive product development. As PLM technologies continue to mature, we can expect these methodologies to become more accessible and eventually standard practice across manufacturing industries. For PLM architects, product designers, and engineering managers seeking to optimize their design processes, understanding these foundational concepts provides an important starting point for evaluating how parallel build approaches might benefit their specific organizational contexts and product development challenges.

References

[1]. Vebjorn Berg et al., "Achieving agility and quality in product development - an empirical study of hardware startups," Journal of Systems and Software, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0164121220300777

[2]. Amir H. Behzadan et al., "General-purpose modular hardware and software framework for mobile outdoor augmented reality applications in engineering," Advanced Engineering Informatics, 2008. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S1474034607000481

[3]. V. Krishnan and K. T. Ulrich, "Product Development Decisions: A Review of the Literature," Management Science, 2001. [Online]. Available: https://www.researchgate.net/publication/227447352_Product_Development_Decisions_A_Review_of_the_Literature

[4]. Bahram Hamraz and P. John Clarkson, "Industrial Evaluation of FBS Linkage – A Method to Support Engineering Change Management," Journal of Engineering Design, 2015. [Online]. Available: https://www.researchgate.net/publication/275674358_Industrial_Evaluation_of_FBS_Linkage_-_A_Method_to_Support_Engineering_Change_Management

[5]. Suryanto Nugroho et al., "Comparative Analysis of Software Development Methods between Parallel, V-Shaped and Iterative," International Journal of Computer Applications, 2017. [Online]. Available: https://www.researchgate.net/publication/318486396_Comparative_Analysis_of_Software_Development_Methods_between_Parallel_V-Shaped_and_Iterative

[6]. Yerra, S. (2025). Enhancing inventory management through real-time Power BI dashboards and KPI tracking. doi : https://doi.org/10.32628/CSEIT25112458.

[7]. Michel Vrinat, "Requirements Management with PLM: A Framework Supporting Consistency Across Product Development Groups," PROLIM, White Paper, 2009. [Online]. Available: https://www.prolim.com/wp-content/uploads/2017/11/Requirements-Management-with-PLM.pdf

[8]. T.A.W. Jarratt et al., "Engineering change: An overview and perspective on the literature," Research in Engineering Design, 2011. [Online]. Available: https://www.researchgate.net/publication/225400174_Engineering_change_An_overview_and_perspective_on_the_literature

[9]. Aligni, "The Advantages of Managing ECM in PLM," Aligni Knowledge Center. [Online]. Available: https://www.aligni.com/aligni-knowledge-center/the-advantages-of-managing-ecm-in-plm/

[10]. Daniel Böckin et al., "Business model life cycle assessment: A method for analyzing the environmental performance of business," Sustainable Production and Consumption, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352550922001026

[11]. Shelly St.Hill, "Applying Agile Methodologies to Hardware Product Development," Arena Solutions Blog, 2023. [Online]. Available: https://www.arenasolutions.com/blog/applying-agile-methodologies-to-hardware-product-development/