INTERNATIONAL DESIGN CONFERENCE – DESIGN 2024
https://doi.org/10.1017/pds.2024.280

DESIGN 2024

Interdisciplinary system lifecycle management – a systematic literature review

Fabian Wyrwich [1,✉], Aschot Kharatyan [1] and Roman Dumitrescu [2]

[1] Fraunhofer IEM, Germany, [2] Heinz Nixdorf Institute, Paderborn University, Germany

✉ fabian.wyrwich@iem.fraunhofer.de

Abstract

The increasing proportion of software in technical products means that both the products and the associated development processes are becoming more complex. An integration of the existing lifecycle considerations Application Lifecycle Management and Product Lifecycle Management into an interdisciplinary System Lifecycle Management promises to make the complexity manageable. To obtain an overview of the current benefits, challenges, requirements, approaches and open research gaps in the context of an ALM-PLM integration, this contribution presents the results of a Systematic Literature Review.

Keywords: systems engineering (SE), product lifecycle management (PLM), application lifecycle management (ALM), system lifecycle management, systematic literature review

1. Introduction

Modern products and systems are increasingly digitalised and networked. They are no longer mechatronic systems consisting only of mechanics, electrics/electronics, and control engineering. The increasing proportion of software in technical systems is turning them into complex Cyber-Physical Systems (CPS) or Intelligent Technical Systems (ITS). The associated increase in interdisciplinarity and complexity poses challenges for companies. Modern system development can no longer be driven from the perspective of a single discipline but must include a holistic system view to conduct a multidisciplinary approach. The approach that addresses these challenges is Systems Engineering (SE). The goal of Systems Engineering is to communicate a unified and comprehensive understanding of the system to all stakeholders involved. All processes and methods that are necessary to develop a ITS are considered and aligned with this goal (Walden et al., 2015).

Topics, that relate to Systems Engineering and to the increasing amount of Software in technical systems is Product Lifecycle Management (PLM) and Application Lifecycle Management (ALM). PLM is the approach to manage processes and methods, which are applied or generated through a products or systems lifecycle. Its scope is mainly focused on the hardware parts of a systems. Therefore the main parts, managed in PLM are CAD drawings, product data such as bills of materials (BOM) and project management processes (Stark, 2020). ALM on the other hand is focused on the lifecycle of software. Hence it is made for monitoring, controlling, and managing the artefacts and processes, which occur during the lifecycle of an application. These includes requirements engineering, source code management, test management, release management and many more (Schwaber, 2006).

Historically, ALM and PLM have been handled differently. While hardware is subdivided into parts, which are divided into hierarchical structures, software consists of individual files, which are combined into an entire solution via complex relationships. Furthermore, the lifecycle phases of software and hardware are also different, even if there are also similarities (Rizzo, 2016). Since complex Cyber-

Physical Systems or Intelligent Technical Systems consist of both software and hardware, it is necessary for ALM and PLM to work closely together to ensure efficient and traceable system development. For this reason, an integration of ALM and PLM into an interdisciplinary System Lifecycle Management is necessary. To capture the current state of research on an ALM-PLM integration, this paper proposes a Systematic Literature Review (SLR). This will reveal current benefits, challenges, requirements, approaches and open research gaps in the context of an ALM-PLM integration.

2. State of research and related work

2.1. Product Lifecycle Management

During a product lifecycle, a lot of information and artefacts accumulate that need to be managed as efficiently and effectively as possible (Stark, 2020). The approach for this management is PLM - Product Lifecycle Management. This is a business concept that accompanies a product from the initial idea through development and production to disposal, recycling or retirement (Saaksvuori and Immonen, 2008). This involves both the management of data, such as requirements, BOMs, test specifications, etc., as well as the processes, methods and tools that are used during the different lifecycle phases. Even though PLM can theoretically be applied without tools, the specific PLM tools facilitate the digital implementation of the aforementioned concept. In this context, PLM is often referred to as the backbone of a virtual product development (Eigner and Stelzer, 2009). This means that the PLM tools are regarded as the centre in which all data (e.g. CAD, simulation or architecture data) flow together and are also managed. Even if the term Product Lifecycle Management gives the impression that any data of any product can be managed in it, the focus is on the data, methods and processes that arise in the hardware lifecycle. This is due to the fact that PLM has grown out of CAD (Computer Aided Design) and PDM (Product Data Management). For this reason, PLM is only suitable to a limited extent for mapping and accompanying a software lifecycle.

2.2. Application Lifecycle Management

All activities that occur within a software lifecycle are coordinated within Application Lifecycle Management (ALM). This includes, among other things, requirements management, modelling, development, the creation of source code and the testing of the software. Therefore, all necessary activities must be managed, trace-links between development artifacts must be transparently displayed and the progress of all activities during the software lifecycle must be reported. It should be mentioned that ALM, just like PLM, is a concept, which can theoretically be used without a tool. This concept is mainly used to integrate all processes, methods and data that are used and generated within software development (Schwaber, 2006). To summarize ALM in three key aspects, it would be governance, development, and operations. Governance is the development of the business case of the software, the management of the project portfolio and the management of the application portfolio. Development is the generation of the software, from the requirements definition, the design, the source code to the testing and release. In other words, the development phase of ALM corresponds to the Software Development Lifecycle (SDLC) (Gorrod, 2004). The operations phase is the monitoring and the management of the software while it is e.g. running in ITS as embedded software or as an application (Chappell, 2010).

2.3. System Lifecycle Management

While PLM is hardware-based and ALM focuses on the lifecycle of software, the aim of integrating both lifecycle approaches is an interdisciplinary and holistic information and process management. System Lifecycle Management integrates different authoring, production, and development systems along the entire lifecycle of an interdisciplinary system. It is therefore to be understood as the technical and organisational backbone of all development artefacts created and the tools used for them. This enables clear traceability across all system elements. Interdisciplinary System Lifecycle Management is therefore the basis for comprehensive Digital Engineering (Eigner, 2021).

3. Research questions

As described in the introduction, the lifecycle of complex ITS includes the hardware lifecycle of the mechanical or electronical parts as well as the lifecycle of all software parts, which runs the systems. Only an interdisciplinary and holistic view and management of a system lifecycle enables efficient and successful product development as well as operation of the systems. As a solution approach, the integration of the two previously described approaches ALM and PLM, which are already anchored in their respective disciplines, promises to achieve this goal. To achieve this integration, the following research questions must be answered:

1. What are the benefits of integrating ALM and PLM?
2. What are challenges of integrating ALM and PLM?
3. What are requirements for an ALM and PLM integration?
4. What are current approaches to integrate ALM and PLM?
5. What are open research gaps to achieve an ALM and PLM integration?

4. Research method!

To answer the research questions, this contribution uses the Systematic Literature Review (SLR) approach. The aim is to explore the current state of the literature on the topic of ALM-PLM integration to explore interdisciplinary System Lifecycle Management in the context of Systems Engineering. The process and the structure of the SLR is divided into the three phases planning, conducting and reporting the review (Kitchenham and Charters, 2007). In the planning phase the goals and the needs for this research are defined, so that the result of this phase is the search strategy, which includes the research questions mentioned above. The search strategy is executed in the conducting phase, where the research questions are translated into a search string and search engines are selected. The search string is used to generate a list of literature. This list can be filtered using include and exclude criteria to reduce the list to literature that is relevant to the research question. In addition to the include and exclude criteria, the backward search can be used to expand the literature list with relevant work cited by the result documents of the SLR (Webster and Watson, 2002). In the reporting phase the results of the SLR are purified and presented for further use or combination with other findings. The documentation of the execution of the Systematic Literature Review, presented in this paper, is shown in Figure 1.

Firstly, a search string was created with the aim of obtaining literature results that answer the research questions. The approach to setting up the search string was aimed at obtaining the largest but at the same time most useful and relevant literature as possible. Since it is impossible to consider all terms used for the research topic, the search string must consist of as few, but very meaningful terms as possible. For this reason, the search string consists of ALM and PLM terms, which covers the most relevant areas. To begin, the first part of the search string consists of the term "Application Lifecycle Management" and its abbreviation "ALM". In addition, the term "Software Management" has been added to cover all aspects of software development, which are not named ALM, but address the same topic. In the second part of the search string, the term "Product Lifecycle Management" and its abbreviation "PLM" represents the hardware lifecycle management. Within the two parts of the search string, the terms are linked with the Boolean operator OR to widen the search results. The two parts itself are linked with the Boolean operator AND to ensure search results, which use at least one of the terms of the two parts.

Even though the literature search is oriented towards an ALM-PLM integration, terms such as "integration", "combination" or "configuration" have been avoided, as attempts with such extended search strings have only achieved very limited results. Furthermore, the results should deliberately not be limited to these terms, as many terms can describe the integrated view of both lifecycle management approaches and not all of them can be covered in the search string. In addition, the terms CPS and ITS are neglected, because the SLR is supposed to cover all systems consisting of software and hardware components, although they are not explicitly called ITS or CPS.

Search string:

(*ALM* OR *Application Lifecycle Management* OR *Software Management*) AND (*PLM* OR *Product Lifecycle Management*)

This search string was used in the databases of Scopus, Web of Science and IEEE as these search engines are three of the most important literature databases for Systems Engineering. The result list contains 241 documents in total (27.09.2023). Distributed among the databases used, the Scopus results list consists of 54 documents, the Web of Science results list 35 documents and the IEEE results list 152 documents. Since different databases were used, the first step of filtering the results is to remove the contained 32 duplicates. As a next review, all documents, which are assigned to a subject area other than Systems Lifecycle Management are excluded. Especially literature from the medical and physical disciplines was sorted out, as it did not contribute to the research questions. After the topic filtering step, the titles, and abstracts of the remaining 131 documents are examined. Mainly the documents whose title and abstract do not correspond to the research topic or do not match the research questions are excluded. In the final step, the remaining 44 documents are part of the full-text review. Sources that do not contain content about an ALM-PLM integration in the main text are sorted out. Beside the exclusion of further 29 documents, three sources are added via the backward search, because they provide valuable content but are not included in the SLR result list. Conclusive, the final list of unique and important documents contains 18 sources, which represent the current state of research regarding an ALM-PLM integration.

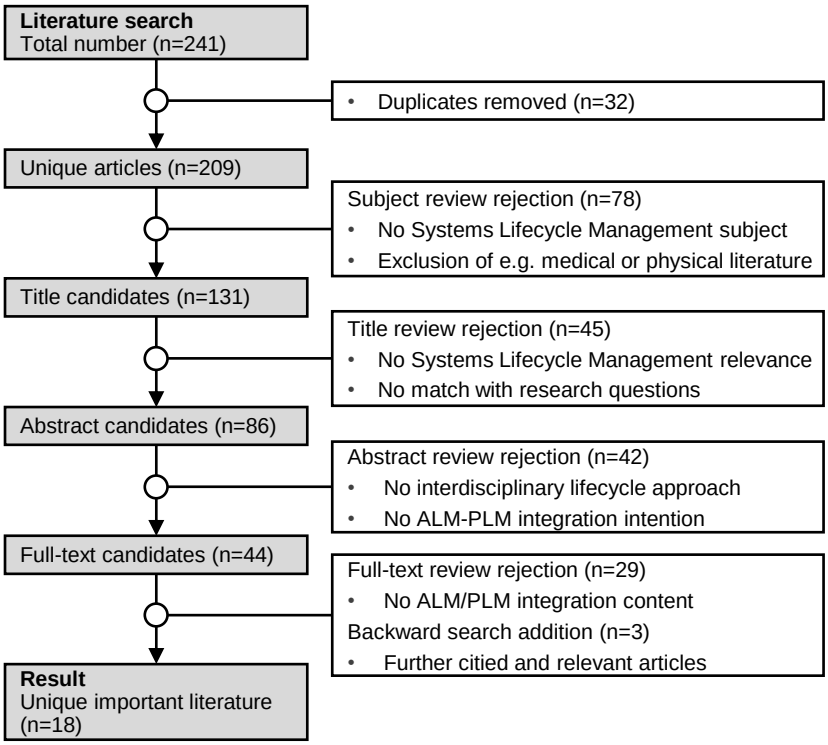

Figure 1. SLR research design to analyse the current research for an ALM-PLM integration

5. General findings

The fundamental results of the Systematic Literature Review are on the one hand the confirmation of the need for further research in the field of ALM-PLM integration and on the other hand the identified advantages and the necessity of an interdisciplinary System Lifecycle Management. The list of results of the literature research includes a Systematic Literature Research carried out in 2016, which describes the status of ALM-PLM integration at that time (Deuter and Rizzo, 2016). One of the results of this research is the classification of the publications into the categories: 1) General importance of ALM-PLM collaboration without detailed suggestion, 2) Collaborative use-cases, 3) Architecture and technology, 4) Using ALM to develop PLM solutions. Using the results contained therein with the help of the backward-search and the self-identified documents, the benefits (see 5.1) and the challenges of an ALM-PLM integration (see 5.2.), requirements for an integration (see 5.3.), current approaches for an ALM-PLM integration (see 5.4.) and open research needs in this area (see 5.5.) are presented in the next subchapters.

5.1. Benefits of an ALM-PLM integration

Although the differences between the software and hardware lifecycles described above are always addressed, it also becomes clear that ALM and PLM complement each other and that it is indispensable due to the steadily increasing share of software, as management in separate silos carries too great a risk. Furthermore, the fact that PLM is not suitable for managing software and ALM is not suitable for managing hardware forces research institutions and companies to develop an efficient and effective integration (Rizzo, 2016). Moreover, lessons learned have already been reported. Concrete benefits such as increased quality, reduced cycle time, improved engineering flexibility, improved communication, optimised coordination between the different processes and tools, faster ramp-up time and qualification management are listed. In summary, the efficiency, effectiveness, and quality of the engineering processes are increased (Ebert, 2013).

5.2. Challenges of integrating ALM and PLM

The challenges that ALM-PLM integration brings with it are primarily based on the fundamental differences between the two lifecycle approaches, which have already been mentioned. These include the fact that neither PLM is able to manage the software lifecycle nor ALM can map the hardware lifecycle (Duda et al., 2022). This fact is mainly due to the different backgrounds of the two approaches. Both approaches have a high number of different processes, various tools and build on a different knowledge base. In addition, the integration of the two disciplines has been very weak up to now. In summary, this results in a high level of organisational complexity. In addition, each discipline would like to keep its familiar tools, processes, artefacts and ways of working, as they have been using them for years and are not necessarily willing to exchange them for another (Feichtinger et al., 2022). One reason for this is primarily the different language that the various disciplines speak in their own developments. ALM and PLM are based on different data models and use different artefacts. Therefore there are incompatibilities and data intransparencies between the disciplines that represent a hurdle for an ALM-PLM integration (Rao and Palaniappan, 2020; Gaurav et al., 2022). A possible solution is a cross-disciplinary ontology that brings together the different data models. Another approach is a sufficient connection between the different modelling approaches of the disciplines to ensure direct traceability and to make the development as transparent as possible (Feichtinger et al., 2022). In addition to the technical challenges that ALM-PLM integration brings with it, organisational challenges must also be solved. Essential for the development or introduction of an ALM-PLM integration is, on the one hand, a good understanding of the ALM and PLM approaches themselves and, on the other hand, an understanding of the effects and possibilities of an integration of the two approaches (Deuter et al., 2018). Furthermore, the initial introduction of an ALM-PLM integration requires extensive, complex and professional change management, as this change requires far-reaching and comprehensive interventions in product development. A wrong approach in this case would be to provide only a tool with the necessary IT interfaces. A tool alone is not able to solve the organisational challenges, which can be proven by the high percentage of failed ALM and PLM projects (Ebert et al., 2015). In summary, three key challenges can be identified: Complexity, Multidisciplinarity and Variability (Feichtinger et al., 2022). Complexity consists of the technical integration of ALM and PLM on the one hand and change management in the respective organisation on the other. Multidisciplinarity is due to the involvement of all disciplines in an ALM-PLM integration. Variability is on the one hand due to a possible high variation of products or systems that must be covered by an ALM-PLM integration and on the other hand due to the high diversity of ALM and PLM processes, procedures and data, since there is no ALM or PLM solution that can be used for all use cases.

5.3. Requirements for a ALM and PLM integration

Since the development or research of an ALM-PLM integration is a relatively young field of research, it makes sense to think about possible use cases and requirements for a linkage of the two lifecycle considerations in advance of an implementation. These requirements can be formulated at different

levels of detail and abstraction. For example, Ebert (2013) formulates the following business requirements, that the environment of an integrated systems engineering must fulfil:

1. Integrated business logic and one comprehensive data model for the entire E/E development process from requirements, through system, SW-, and HW-design to test
2. Architecture design and management
3. Collaboration environment for requirements and test engineering, modeling of functions, components, networks and communication
4. Variant management and product line engineering for effective reuse
5. Support for quality requirements such as functional safety from requirements to concept, models, realisation to validation, proof and acceptance

In terms of implementing an ALM-PLM integration, concrete use cases and system requirements are necessary in addition to the business requirements. Deuter et al. (2017) have developed a methodical procedure to systematically identify these use cases and requirements in the sense of a Systems Engineering approach. Based on the V-model of the VDI 2206 (VDI, 2021), four phases with ALM-PLM integration requirements were identified. These include the requirements phase, system design, system integration and property validation. Due to the very large scope of all four phases, the requirements phase was focused on. As an assumption, Deuter et al (2017) defined, that the PLM is the leading system for capturing the product requirements and that the ALM system only manages the software-relevant requirements. With this preparation, the following seven use cases for an ALM-PLM integration were formulated as part of the requirements analysis:

1. Enter product requirements in PLM
2. Transfer product requirement from PLM to ALM
3. Release product requirement in PLM that is linked to ALM
4. Change product requirement linked to ALM in PLM
5. Delete product requirement linked to ALM in PLM
6. Release product requirement linked to ALM in PLM
7. Create traceability report for all requirements across all systems

For each use case, the characteristics Actor, Trigger, Goal, Postcondition and Normal Flow as well as a sequence diagram were also created (Deuter et al., 2017). This provides a comprehensive insight into the necessary features that an ALM-PLM integration must provide. In addition, it is described that the other use cases of the subsequent phases are in coverage, but not yet fully described. Based on this, the previously identified use cases were expanded in connection with the implementation and validation with the help of an industrial case study, so that the concrete requirements could be derived on the basis of a total of thirteen use cases (Deuter et al., 2018). Although the requirements derived from the use cases were not explicitly listed, they deal exclusively with the topic of traceability between an ALM tool and a PLM tool. In consultation with the tool vendor of the associated tools, it was confirmed that the identified requirements are meaningful and valuable, but that only two requirements are fulfilled at this time. In addition to the industrial case, the authors have also presented the general six-step approach used to identify the requirements of an ALM-PLM integration. In summary, overarching requirements for ALM-PLM integration are characterised by a multidisciplinary and transparent engineering environment that brings together both approaches and links their artefacts. It is important that this environment is fitted into the corresponding corporate framework. This results in a seamless integration of both lifecycle approaches.

5.4. Current approaches to integration ALM and PLM

The presentation of the identified approaches to implement an ALM-PLM integration is divided into three sections. The first section deals with the implementation of partial solutions or the solutions of partial problems in the context of an ALM-PLM integration. These concepts are generally tool focused but enable a change of the methods and processes of ALM and PLM. The second section introduces known concepts that enable the linking of different artefacts from ALM and PLM tools. The third section presents a holistic concept for sleek lifecycle integration & management, which is generally tool-independent and is intended to introduce a theoretical integrated lifecycle management.

Branch & merge concept in PLM

The first approach to solve a sub-problem in the context of an ALM-PLM integration deals with the different procedures of version control respectively of engineering change processes in ALM and PLM. In software management, the branch & merge approach is followed. This allows several versions (branches) of the same software to exist at the same time, so that it can be further developed by several developers at the same time without affecting each other. Or different software versions exist for different products that originate from the same original software version. If different versions or branches are to be synchronized again, this is called a merge. The two versions are compared with each other and combined into a common, current version. This procedure enables transparent and agile version management in the software. In PLM, changes are typically made to a single version via an engineering change process, which is then promoted to a new version, but there are never two versions of the same part. The downside of this approach is that it is not possible to work collaboratively and simultaneously on the same part. A solution to this problem is to apply the branch and merge concept to PLM. The base text or binary files of CAD files, for example, are used to compare two files and allow different changes to be merged (Bricogne et al., 2012). This enables the developers to work on the same part at the same time (different branches) and to merge their changes afterwards (Bricogne et al., 2014). Another approach is to assign a hash function to each part, which contains all the information, such as the meta data, the color, etc., of a STEP file. This can be used for an unambiguous comparison of two parts (Fresemann et al., 2021). This approach saves predecessor and successor versions, so that version management according to the branch and merge concept can be used for e.g. CAD files.

Software classification in PLM

Another sub-problem is software classification in PLM systems. While in current PLM systems software artefacts are often managed as "parts", this approach is not sufficient for companies who want to classify their software and firmware in the same way as hardware parts. However, since on average only 1% of all product classes in PLM systems are intended for software artefacts, this classification must be done by the company itself. For this case, Gaurav et al. (2022) have developed an end-to-end framework that contains all the necessary processes, concepts, and tools to classify software parts and define this data in a standardized way. The four overarching parts of the framework are Awareness and Persuasion, Design Decision, Transformation and Sustenance and Continuous Improvement (Gaurav et al., 2022).

Linking ALM and PLM artefacts with the Asset Administration Shell and OSLC

In the area of linking, the focus is primarily on the linking of concrete ALM and PLM tools. These links between the various ALM and PLM artefacts are mainly set via two well-known concepts: the Asset Administration Shell and OSLC. The concept of the Asset Administration Shell is used in the case of Deuter and Imort (2020), for example, in the area of requirements linking. For this purpose, a separate submodel was created for the ALM and PLM data, into which the data is exported. The PLM data is exported in XML data format and the ALM requirements with the help of the ReqIF data format. These elements can now be linked to each other via a relationship element in the asset administration shell (Deuter and Imort, 2020). In this case, design elements from a PLM system were linked with requirements from an ALM system to establish traceability. In theory, any linking would be conceivable with a suitable export format for the desired data.

Another approach to link ALM and PLM artefacts is the use of OSLC (Open Service for Lifecycle Collaboration). In principle, OSLC is suitable for linking any product development tools that have a corresponding interface. During the Systematic Literature Review, various use cases of OSLC linking were identified. As in the case of the Asset Administration Shell, the linking of artefacts in the PLM system with requests in the ALM system via OSLC is conceivable (Shani et al., 2017; Brusa et al., 2018). However, it is also possible to link ALM-PLM in the context of test management or quality management, e.g. in the context of functional system testing (Nardone et al., 2020). But also linking ALM-PLM tools with Systems Engineering tools, systems monitoring tools and infrastructure management tools is conceivable and possible (Kennedy and Jiu, 2013).

Sleek lifecycle integration & management (SLIM)
As a approach, Rao and Palaniappan (2020) present an innovative lifecycle management called sleek lifecycle integration & management (SLIM) (Rao and Palaniappan, 2020). This is a concept that is geared towards the integration of any PLM with any ALM as a kind of federational layer between the systems. SLIM uses a combination of integration platforms and VAUSE (Vault of Archetypes for Unified Systems Engineering). The goal of this concept is to establish a practical and elegant process that enables a seamless exchange between PLM and ALM systems. For this purpose, SLIM is located between the ALM, PLM, engineering, and non-engineering tools to connect them as a kind of "glue". For this, the focus is on both the abstraction of multiple tools, data formats and standards and the simple integration of new tools into the platforms. SLIM also promises to enable the modelling of business logics and data exchange rules in reusable templates. Likewise, the efficient transformation and traceability of heterogeneous technical artefacts as one of the most important parts of an ALM-PLM integration is part of SLIM. In summary, sleek lifecycle integration & management (SLIM) enables the synchronization of lifecycles, the migration of data and processes, the exchange of software, an integration of development partners and an interconnected toolchain (Rao and Palaniappan, 2020).

5.5. Open research gaps to achieve a ALM and PLM integration

As another result of the Systematic Literature Review, open research gaps were identified. An overarching outlook or requirement for future research in this area is a better **linking of knowledge management with business**. This means the connection and linking of business objectives and metrics with the lived processes in order to uncover awareness of grievances and potential for improvement (Ebert, 2013). Building on this, the exchange of knowledge between the two approaches ALM and PLM is another need. The design of an integrated or joint **ALM-PLM data model** pays to uncover the differences between hardware and software development and to reduce them as much as possible (Deuter and Imort, 2020). Furthermore, the previously described requirements/use cases and solution approaches must be further elaborated. In this context, the use cases and requirements for an ALM-PLM solution should be **extended to other lifecycle phases** to create a better user experience and cover all integration needs. In terms of the elaboration of the already achieved solution approaches the further **detailing and elaboration of the presented approaches** is meant. Furthermore the extension by further approaches around new methods, ideas, and processes to support the software and hardware development needs to be done. In addition, a definition of **key performance indicators (KPI)** for ALM-PLM integrations would be useful. This would allow the approaches to be evaluated and monitored in use in order to assess their success or potential for improvement (Deuter et al., 2017; Deuter and Rizzo, 2016). In the context of assessing and evaluating solutions, it is also interesting to **evaluate the approaches** in the various user areas. In this way, software developers would come into contact with PLM concepts and hardware developers with ALM concepts and can evaluate them or give feedback regarding implementation and deployment maturity (Deuter et al., 2018).

6. Conclusion and outlook

The scientific contribution of this research is to systematically analyse the field of an ALM-PLM integration to raise the potential for an interdisciplinary System Lifecycle Management approach for ITS or CPS. The results of the Systematic Literature Review (SLR) are giving an overview of the benefits, the requirements, the challenges associated with this integration, current approaches to implement an integration and open research gaps to achieve a holistic ALM-PLM integration. Therefore, the results summarize the basic and recent work on an ALM-PLM integration.
As the publications found have shown, there is already research work and approaches in the field of ALM-PLM integration done. These range from a fundamental consideration of the challenges and requirements to concrete approaches and proposals for solutions. While the requirements and challenges of an ALM-PLM integration address the basic challenges on a high level of abstraction, the integration approaches already propose solutions for some sub-areas. In addition, the research needs that are still open are identified and disclosed. The central point is to create a successful knowledge management to create a uniform understanding between the different domains. One approach mentioned in the literature is to build a common data model to uncover the differences between the ALM and PLM approaches and

to identify the commonalities and connecting points in the sense of integration. Furthermore, KPIs were addressed that numerically justify the added value and maturity of an ALM-PLM integration. The KPI validation could take place through an application of integration approaches in applying companies.
The analysed literature sources as well as the benefits, requirements, challenges, approaches and research gaps of an ALM-PLM integration mentioned in chapter five lead to the conclusion that an initial and fundamental analysis of the two different approaches is necessary for the implementation of a holistic ALM-PLM integration. To fully exploit the potential of the interdisciplinary System Lifecycle Management or an ALM-PLM integration mentioned in this article, the differences must be successively identified and divided into sub-challenges. For this purpose, a common data model and an overarching knowledge management between the software and hardware lifecycle are feasible solution approaches. To do this, it is necessary to evaluate whether a common data model is sensible and achievable, or whether a common understanding is established via a mapping between the different data models and elements. Possible mappings could be a direct linking of the software and hardware data elements or an indirect connection via an intermediate level. The development and elaboration of these approaches is part of the following research work. Furthermore, the solution approaches must be transferred into tools to guarantee the industrial applicability of the integration. In addition, KPIs of an ALM-PLM integration are to be developed to make the maturity of the implementation and realisation as well as the added value measurable.

Acknowledgement

This article is part of the research project SLE - Sustainable Lifecycle Engineering. This project is funded by the Ministry of Economic Affairs, Industry, Climate Action and Energy of North Rhine-Westphalia and the technology-network "Intelligent Technical Systems OstWestfalenLippe" (it´s owl) and is managed by the Project Management Agency Jülich (PTJ). The authors are responsible for the content of this publication.

References

Bricogne, M., Rivest, L., Troussier, N. and Eynard, B. (2012), "Towards PLM for Mechatronics System Design Using Concurrent Software Versioning Principles", *Product Lifecycle Management: Towards Knowledge-Rich Enterprises / IFIP WG 5.1 International Conference, PLM 2012, Montreal, QC, Canada, July 9-11, 2012*, Springer, Heidelberg, pp. 339-348. https://doi.org/10.1007/978-3-642-35758-9_30

Bricogne, M., Rivest, L., Troussier, N. and Eynard, B. (2014), "Concurrent versioning principles for collaboration: towards PLM for hardware and software data management", *International Journal Product Lifecycle Management* Vol.7, No.1, pp. 17-37. https://doi.org/10.1504/IJPLM.2014.065457

Brusa, E., Calà, A. and Ferretto, D. (2018), "Systems Engineering and Product Lifecycle Management (PLM)", In Brusa, E., Calà, A. and Ferretto, D. (Ed.), *Systems Engineering and Its Application to Industrial Product Development*, Springer International, pp. 327–341. https://doi.org/10.1007/978-3-319-71837-8_11

Chappell, D. (2010), *What is Application Lifecycle Management?*. [online] David Chappell and Associates Available at: http://davidchappell.com/writing/white_papers/What_is_ALM_v2.0--Chappell.pdf (accessed 12.12.2023)

Deuter, A. and Imort, S. (2020), "PLM/ALM Integration With The Asset Administration Shell", *Procedia Manufacturing, Vol. 52 / 5th International Conference of System-Integrated Intelligence, Bremen, Germany, 11-13. 2020*, pp. 234–240. https://doi.org/10.1016/j.promfg.2020.11.040

Deuter, A., Otte, A., Ebert, M. and Possel-Dölken, F. (2018), "Developing the Requirements of a PLM/ALM Integration: An Industrial Case Study", *Procedia Manufacturing, Vol. 24 / 4th International Conference on System-Integrated Intelligence, Hannover, Germany, June 19-20, 2018*, pp. 107–113. https://doi.org/ 10.1016/j.promfg.2018.06.020

Deuter, A., Otte, A. and Höllisch, D. (2017), "Methodisches Vorgehen zur Entwicklung und Evaluierung von Anwendungsfällen für die PLM/ALM-Integration", *Wissenschaftsforum Intelligente Technische Systeme (WInTeSys), Paderborn, Germany, May, 11-12, 2017*, pp. 211-222. https://doi.org/ 10.17619/UNIPB/1-93

Deuter, A. and Rizzo, S. (2016), "A Critical View on PLM/ALM Convergence in Practice and Research", *Procedia Technology Vol. 26 / 3rd International Conference on System-integrated Intelligence: New Challenges for Product and Production Engineering, SysInt 2016, Paderborn, Germany, June 13-15, 2016*, pp. 405–412. https://doi.org/10.1016/j.protcy.2016.08.052

Duda, J., Oleszek, S. and Santarek, K. (2022), "Product Lifecycle Management (PLM) in the Context of Industry 4.0", *Advances in Manufacturing III: Volume 2 - Production Engineering: Research and Technology Innovations, Industry 4.0 / 7th International Scientific-Technical Conference MANUFACTURING 2022,*

Poznan, Poland, May 16-19, 2022, Springer International, pp. 171-185. https://doi.org/10.1007/978-3-030-99310-8_14
Ebert, C. (2013), "Improving engineering efficiency with PLM_ALM". *Software and Systems Modeling*, Vol. 12, pp. 443-449. https://doi.org/10.1007/s10270-013-0347-3
Ebert, C., Hoefner, G. and Mani, V.S. (2015), "What Next? Advances in Software-Driven Industries", *IEEE Software*, Vol. 32 No.1, pp. 22-28. https://doi.org/ 10.1109/MS.2015.21
Eigner, M. (2021), *System Lifecycle Management: Digitalisierung des Engineering*, Springer Vieweg, Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-62183-7
Eigner, M. and Stelzer, R. (2009), *Product Lifecycle Management: Ein Leitfaden für Product Development und Lifecycle Management*, Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-62183-7
Feichtinger, K., Meixner, K., Rinker, F., Koren, I., Eichelberger, H., et al. (2022), "Industry Voices on Software Engineering Challenges in Cyber-Physical Production Systems Engineering". *IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA), Stuttgart, Germany, Sep 6-9*, IEEE Industrial Electronics Society (IES), pp. 1-8, https://doi.org/10.1109/ETFA52439.2022.9921568
Fresemann, C., Falbe, M. and Stark, R. (2021), "Hash functions supporting mechatronic design evolution". *Proceedings of the International Conference on Engineering Design (ICED21), Gothenburg, Sweden, August 16-20, 2021*, The Design Society pp. 1697-1704. https://doi.org/10.1017/pds.2021.431
Gaurav, A., Prakash Ila, B., Mehata Kondamudi, N. and Deshwal, P. (2022), "Software Parts Classification for Agile and Efficient Product Lifecycle Management", *Product Lifecycle Management - PLM in Transition Times: The Place of Humans and Transformative Technologies / 19th IFIP WG 5.1 International Conference, PLM 2022, Grenoble, France, July 10-13, 2022*, Springer, pp. 15-24. https://doi.org/10.1007/978-3-031-25182-5_2
Gorrod, M. (2004), *Risk Management Systems: Process, Technology and Trends*, Palgrave Macmillan, London. https://doi.org/10.1057/9780230510296
Kennedy, S. and Jiu, L. (2013), "Facilating Collaboration and Interaction across the Enterprise with OSLC", *Proceedings of the CASCON 2013 / Conference of the Center for Advanced Studies on Collaborative Research*, November 18-20, IBM Corp., Ontario, Canada, 2013, pp. 374–375
Kitchenham, B. and Charters, S.M. (2007), *Guidelines for performing Systematic Literature Reviews in Software Engineering*. [online] Elsevier. Available at: https://www.elsevier.com/__data/promis_misc/525444 systematicreviewsguide.pdf (accessed 12.11.2023)
Nardone, R., Marrone, S., Gentile, U., Amato, A., Barberio, G., et al. (2020), "An OSLC-based environment for system-level functional testing of ERTMS/ETCS controllers", *Journal of Systems and Software*, Vol. 161 No. C, https://doi.org/10.1016/j.jss.2019.110478
Rao, P. and Palaniappan, K. (2020), "Continuous Engineering Through ALM-PLM Integration", *Product Lifecycle Management - Enabling Smart X / 17th IFIP WG 5.1 International Conference PLM 2020, Rapperswil, Switzerland, July 5-8, 2020*, Springer, pp. 798-811. https://doi.org/10.1007/978-3-030-62807-9_62
Rizzo, S. (2016), *Why ALM and PLM need each other*. [online] Siemens Whitepaper. Available at: https://polarion.plm.automation.siemens.com/resources/download/why-alm-and-plm-need-each-other-whitepaper (accessed 12.11.2023)
Saaksvuori, A. and Immonen, A. (2008), *Product Lifecycle Management*, Springer, Berlin-Heidelberg. https://doi.org/10.1007/978-3-540-78172-1
Schwaber, C. (2006), The Changing Face of Application Life-Cycle Management. [online] Forrester Research Inc. Available at: https://www.yumpu.com/en/document/view/13866040/download-the-changing-face-of-application-life-cycle-mks (accessed 12.11.2023)
Sh`ani, U., Franke, M., Hribernik, K.A. and Thoben, K.-D. (2017), "Ontology mediation to rule them all: Managing the plurality in product service systems". *Proceedings of the 2017 Annual IEEE International Systems Conference (SysCon), Montreal, QC, Canada, April 24-27*, IEEE, https://doi.org/10.1109/SYSCON.2017.7934810
Stark, J. (2020), *Product Lifecycle Management (Volume 1)*, Springer International, Cham, https://doi.org/10.1007/978-3-030-28864-8
VDI (2021), VDI 2206: Development of mechatronic and cyber-physical systems, Verein Deutscher Ingenieure
Walden, D.D., Roedler G. J., Forsberg, K.J., Hamelin, R.D. and Shortell, T.M. (2015), *INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities, Vol. 4*, John Wiley & Sons, New Jersey, https://doi.org/10.1002/j.2334-5837.2015.00089.x
Webster, J. and Watson, R.T. (2002), *Analyzing the Past to Prepare for the Future: Writing a Literature Review*. [online] Management Information Systems Research Center, University of Minnesota. Available at: https://www.jstor.org/stable/4132319, https://doi.org/10.2307/4132319