Data Visualization of A Cyber-Physical Systems Development Toolchain:
An Integration Case Study

Didem Gürdür*, Jad El-khoury
Department of Machine Design,
KTH Royal Institute of Technology,
Stockholm, Sweden
e-mail: {dgurdur, jad}@kth.se

Tiberiu Seceleanu, Morgan Johansson, Stefan Hansen
ABB AB,
Sweden
e-mail: {tiberiu.seceleanu, morgan.e.johansson,
stefan.hansen}@se.abb.com

***Abstract*—Development of Cyber-Physical Systems (CPS) requires various engineering disciplines, artifacts, and areas of expertise to collaborate. There are powerful software tools, which are used during CPS development, but it is often challenging to integrate these tools with each other. This paper proposes a data visualization approach to understanding current interoperability status and the integration needs in CPS development toolchains, and make decisions on potential integration scenarios accordingly. To this end, a case study is introduced based on a toolchain for the development of an embedded application at ABB Corporate Research. The node-link diagram (NLD) data visualization technique was used to understand integration needs and priorities. The study showed that the NLD visualization has the potential to inform toolchain architects about the interoperability situation and help them to make decisions accordingly, especially for small toolchains. Moreover, the integration solution is implemented and the result has been compared with the non-integration study.**

Keywords-toolchain interoperability; tool integration; interoperability visualization; toolchain visualization; data visualization; node-link diagram.

I. Introduction

Cyber-Physical Systems (CPS) rely on the tight interaction of real-time computing and physical systems [21]. CPS development involves the integration of computation and physical processes [1]. Moreover, this development process requires software tool support for the tasks associated with different engineering disciplines throughout the different phases of the Product LifeCycle (PLC) (see Figure 1). These tools are used to complete different PLC stages and they produce artifacts and data. Furthermore, there is a necessity to support intricate relationships between different stakeholder viewpoints at the people, model and tool levels [2].

The interoperability of these software tools is required to improve productivity and efficiency in a consistent manner for CPS development. Yet, the integration of these tools is especially challenging due to their heterogeneous nature. Even though the tool integration research field is progressing, there still are no well-defined methods to guide the toolchain architect to understand the current interoperability status of toolchains [3]. And yet, without understanding the current interoperability situation of the development toolchains, it is difficult to identifying the priorities, dependencies, and correct decisions necessary to improve the development process.

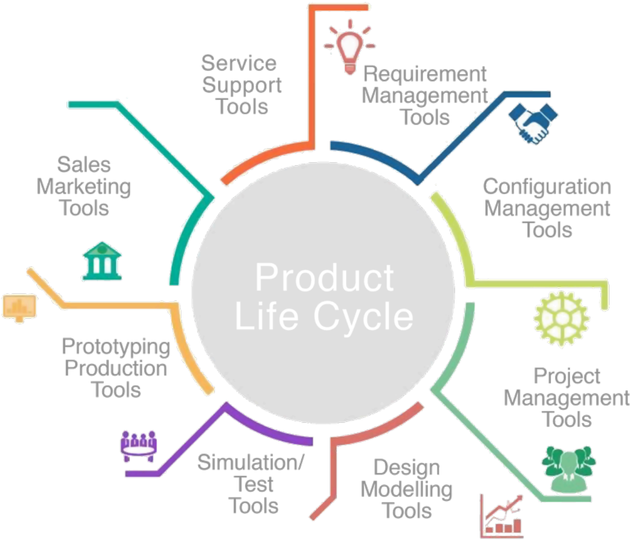

Figure 1. Product life cycle and various software tool categories [24].

Recent advances in computing and data storage technologies have made the existence of vast volumes of data possible, offering a powerful opportunity to discover new insights from the data. However, finding the valuable information in these vast data sets is not easy. Bendre and Thool [4] mention that data analytics and decision support tools in the manufacturing industry would handle, integrate, and analyze collected data and provide appropriate solutions “to improve manufacturing processes, control over production, market-oriented business avenues, and efficient customer service at lower costs, to increase profit and help manufacturers to stay in healthy competition”. Visualization and visual analytics offer the opportunity to help understand interoperability with the added ability to promptly gain insight into the current interoperability of the toolchain. Visualization allows the extraction of patterns concerning relevant issues, such as workflow and tool usage, whilst visual analytics allows iterative work with these patterns [5]. In this way, the complexity and heterogeneity can be handled by analyzing the data visualizations of the development toolchains, allowing toolchain architects to focus on important aspects of integration. This study illustrates the application of visualization and visual analytics to help

ISBN: 978-1-61208-552-4

toolchain architects understand interoperability and support the decision making process in CPS development toolchains.

The study specifically looks at the use of the node-link diagram (NLD) [15] visualization technique. The proposed NLD visualization illustrates the toolchain *before* any integration is introduced in order to show the needs and thus supports the toolchain architect in making decisions according to the collected data about tool usage. Furthermore, this paper summarizes the integration method used to overcome the identified interoperability needs and highlights the change in tool interactions *after* the integration was introduced to underline the success of the integrated toolchain.

This paper is organized in six sections: Section 2 explains the background. Section 3 presents the case study. Section 4 discusses the rationale for the choice of the NLD visualization technique and the application of the technique in the context of toolchain interoperability, as well as summarizing the integration method and presenting the user activity data after integration. Section 5 summarizes the future direction of the study. Finally, the paper concludes with a summary of the study in Section 6.

II. BACKGROUND

Interoperability is the ability of two or more systems, components or tools to exchange information and to use that information effectively [25]. Ford et al. [6] disclose at least 30 different definitions of interoperability from the last 30 years. Interoperability is a multidimensional concept, consisting of several perspectives and approaches from different domains. Although the definitions may differ, they all emphasize the importance of understanding interoperability. Our case study used in this paper is about the CPS development toolchains, focusing on the interactions between software tools to understand interoperability.

Assessing interoperability with well-chosen measures is essential for identifying priorities in product development and production. Many researchers have studied interoperability assessment models and proposed different approaches in literature [7-11]. In our earlier study [5], we examined interoperability assessment models and extrapolated that the literature mainly:

- Uses either complex metrics, separate levels, or combinations of these with little guidance on how such metrics can be used.
- Concentrates on selective aspects of interoperability.
- Focuses on structure and content, providing little guidance on how to deal with interoperability improvements.

Given these findings, we concluded/argued that a more flexible, data-oriented method to increase the understanding of interoperability is needed. Data visualization techniques were in the end chosen for the following reasons:

- Data visualization of toolchains provides an overview of the real situation of interoperability, where data can be filtered to ensure analytics for different stakeholders. Thus, the holistic, dynamic and bridged analysis could be possible to provide a better interoperability understanding for the stakeholders
- Data could be collected for different aspects of interoperability to extend the visualizations to cover more than one selective aspect, and to facilitate anaylsis of the interactions between different aspects. This is an important opportunity when addressing the overall interoperability status.
- Data analytics aims to guide the user towards better interoperability by allowing the toolchain architecture to see the big picture. Furthermore, this approach could be combined with some metrics such as cost, performance, and sustainability of the toolchain to guide the toolchain architect to take decisions according to these metrics.

Visualizations and visual technologies have also been pointed out by well-known initiatives that aim to contribute to better-integrated engineering environments, such as the Industrial Internet Consortium, the Advanced Manufacturing Partnership, Industrie 4.0, and La Nouvelle France Industrielle. These initiatives consider visual computing as a promising technology to be used to improve development environments. For instance, Industrie 4.0 mentions visual computing as a valuable support for acquiring, analyzing, and synthesizing data [12], while the Advanced Manufacturing Partnership organized workshops with the visualization, informatics, and digital manufacturing work groups to define fundamental research opportunities for these technologies with respect to smart manufacturing [13].

This study summarizes the work done in [22] where the Open Services for Lifecycle Collaboration (OSLC) [23] framework has been used to integrate software tools used in this case study. OSLC is an OASIS standard consisting of members from both industry and academia with a goal to standardize how tools should interact and share data. The OSLC standard is organized in work groups that each addresses a specific domain of tools such as requirements management, test management, change management or configuration management. Moreover, deriving all domains from the OSLC core specification ensures compatibility between domains. The earlier work [22] presents the details of how integration solution is developed by defining a version control domain based on the OSLC core specification, and describes how to represent versioned artifacts and perform version control operations. The study in [24] presents different visualization techniques and exercises their applicability before implementing any integration solutions. In this paper, we concentrate on the most successful data visualization approach from [24] and repeated the same development work by using the integrated toolchain and compare the non-integration and integration scenarios through data visualizations.

III. Case Study

The case study is about the development of a prototype application targeting the Cooling System for Transmission Plant (CSTP) (Figure 2) at ABB. The application is a closed loop control system where a number of sensor elements and actuators are connected by various interfaces. The system performs relevant actions depending on the input signals, the internal system state, the configurable logic, and possibly on operator commands. The system is required to perform a variety of computation-intensive operations, with very high real-time requirements, on data coming in concurrent streams.

This paper does not focus on the application of CSTP but rather on the creation and execution of a toolchain to support its development. Therefore, we collected data about the toolchain activities. During the development of the CSTP four different tools used such as Team Foundation Server 2005, Team Foundation Server 2015, HiDraw and Internet Explorer. Firstly, we installed user activity tracking software on one of the developers' computer. The tracking application worked on a dedicated computer for a period of time and saved information about tool usage. Secondly, we cleaned and filtered the data collected by the tracking application to remove unrelated tool accesses and to be able to understand tool interactions easier. As a third step, we used this data to develop data visualization of the toolchain. The aim was to visualize user activity during the development of CSTP to find out how much time was spent on each tool, and to see any patterns that might support a toolchain architect in making any decisions on integration scenarios. One important factor was the switching between tools, which can be explained as changing between tools. The data visualization was used to improve the understanding about the current interoperability situation. As a next step, integration need is identified and tools are integrated using the OSLC standard. The same developer was tracked again for the same amount of time on the same project in order to compare the toolchain performance before and after integration.

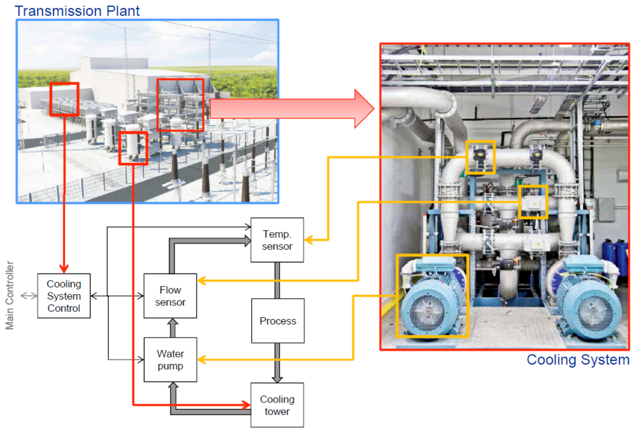

Figure 2. Cooling System for Transmission Plant [24].

During the development of the CSTP application, four main software tools are used. These tools are:

- Microsoft Visual Studio | Team Foundation Server 2005 (TFS 2005): used for storage of requirements, development artifacts and supporting documents, in addition to performing version control.
- Microsoft Visual Studio | Team Foundation Server 2015 (TFS 2015): used for storage of requirements, development artifacts and supporting documents, in addition to performing version control.
- HiDraw (HD): a proprietary graphical design tool, used to model the structure and functionality of the control application from which code can be generated, deployed and monitored. The generated code is then stored in TFS.
- Internet Explorer (IE): used as a support tool to access the TFS web interface to view and edit work items. The main reason for its usage, as explained by developers, is that ease of use of the IE, as compared to the TFS, especially for localizing work items.

The case study was designed to collect data about one developer's tool usage activity for a period of one month. A developer's activity was recorded by tracking the application to create data visualizations. This data is deliberately defined in a compact format in order to collect minimum information about the tool usage. In this way, we aimed to make the data collection, cleaning, and filtering process as simple as possible. The data only includes tool name, start time (defined as the time the developer activated a particular software tool) and end time (when the developer completed using the tool and switched to another tool). During the cleaning process, we merged the start and end time by introducing a new attribute called duration. We also filtered the data by combining some rows where the developer stopped using one tool but then start to use it again without switching to another tool. Table I is generated to summarize the total usage of each tool and the switching percentages between them (Table I).

TABLE I. FINAL DATA ABOUT THE TOOL USAGE AND INTERACTIONS DURING THE DEVELOPMENT OF CSTP.

	Total Usage	**HD**	**IE**	**TFS 2015**	**TFS 2005**
HD	53%	0%	48%	47%	5%
IE	33%	65%	0%	34%	1%
TFS 2015	13%	72%	28%	0%	0%
TFS 2005	1%	57%	43%	0%	0%

The same process was repeated after the integration of tools by tracking the same developer's activity for a similar amount of time. This second phase of the study observed the effect of integration on the developer's activity. The next section offers a discussion of the data visualization approach and comments on the understanding of toolchain interoperability needs, along with a brief summary of the integration details.

Copyright (c) IARIA, 2017. ISBN: 978-1-61208-552-4

IV. Discussion

In preparation for the collection and visualization of data, we organized meetings where different stakeholders of the toolchain discussed what factors are important to them in understanding interoperability better. These meetings concluded by identifying two main needs for understanding interoperability better:

- In any visual representation, each tool needs to be easily distinguished from the others. Each tool should be represented as a first-class element in the diagram. Different colors for each tool representation are used for this purpose.
- The most important information to be represented is the time spent using the tool by the developer For this reason, the size of tool representation should be proportional to this property. Interactive tooltips are also added to the graphical representations to provide more information about each tool.
- The interactions between tools are the main focus of current interoperability assessment methods. There is hence a need to reveal the interaction patterns in the studied case, which will then be a basis to prioritize mostly interaction tools for increasing interoperability. For this purpose, the interactions are added to the visualizations as an arc or link shape. The opacity of lines represents the interaction frequency between tools. In addition, the size of the shapes is proportional to the interaction rate.

To visualize the development toolchain we chose to use an NLD visualization technique. The two reasons behind this choice are:

Readability: NLDs are the most familiar representation of graphs in general.

Understanding: NLDs are intuitive, compact, and good at showing the overall structure of information. They are especially effective for small graphs.

An NLD is a tree-type data visualization that captures entities as nodes and relationships. The layout has the potential to use the entire two-dimensional space, offering a number of ways to represent interactions. This large variety of possible layouts allows different aspects of the data to be focused on, especially useful for large graphs. Battista et al. [14] presented an extensive collection of possible layout algorithms for drawing a graph of data using the NLD. This bibliographic survey attempts to encompass both theoretical and application-oriented papers from disparate areas. We refer the interested reader to this study for a detailed assessment of these algorithms.

Figure 3 shows the NLD visualization of the data we collected for the case study. Five data variables are used in this visualization - nodes, node labels, links, a qualitative attribute and a quantitative attribute. The mapping between data variables and visual variables in NLDs is as follows:

- The nodes are shown as circles to represent different tools;
- Each node has a label which is the name of the tool;
- There are links between nodes that are represented by line segments that show the interactions of tools.

A qualitative attribute is shown by the color of each circle and it is used to distinguish different tools. Lastly, the quantitative attribute is indicated by the size of the circle, which represents the usage frequency of the tool. In other words, the size of the circle is proportional to the time the user spent using in this tool. A link between two circles represents the switching behavior between the corresponding tools performed by the developer, where the opacity of the links is proportional to the switching frequency. A darker link color encodes higher interactions between tools. Thus, Figure 3 shows that the tool named HD is the most used tool during the development process, since the corresponding circle is the largest. Moreover, most of the tool switches occur either between TFS 2015 and HD or between IE and HD.

The toolchain architect can easily distinguish the tools using the labels next to each circle. Since the visualization is interactive there is a possibility to include more information. There are tooltips, which inform the architect about the links and nodes. For instance, a toolchain architect can get more information about the time each tool was used, by hovering over the circles, or they can learn about the switching behavior by hovering over the links between tools.

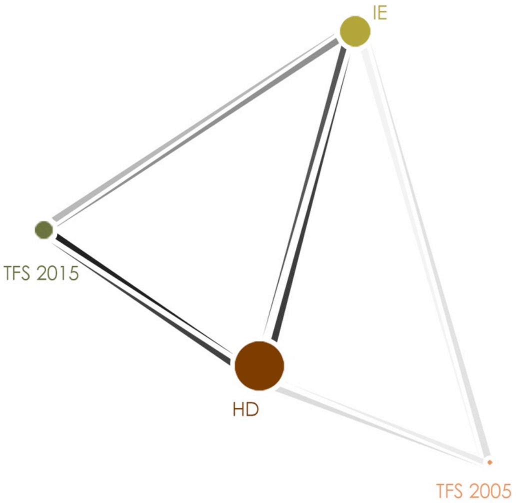

Figure 3. Node-link diagram of the development toolchain before integration [24].

NLDs help to observe global patterns of interactions in a toolchain. They make it easier to spot unexpected connections and understand the switching behavior between tools. Moreover, visual features such as color and size reveal the heterogeneity and time spent using each tool in the development process. One can make decisions about the toolchain interoperability according to the visualization and the graphic can be used to explain the need for interoperability in CPS development environments. Once the data was collected, and the visual diagrams were prepared, a meeting with the stakeholders was organized again. The toolchain architects found the visualization easy to read, and they were directly able to point out which parts of the toochain can best benefit from better interoperability.

The resulting NLD visualization of the CSTP shows the toolchain architect how much unnecessary time has been spent on IE to find information about the task. This visualization further used to show other stakeholders the necessity of integration, and supports communication about the problem. After having meetings and discussing the NLD visualization, it was decided that HD needs to be integrated with TFS 2005 and TFS 2015.

As a next step, ABB developed an integration solution based on the OSLC standard. In OSLC, data is represented as a Resource accessible and identified by a uniform resource identifier (URI). Other tools can look up, reference and interact with the resource by accessing the URI via RESTful services. OSLC resources are exposed by services through creation factories and query capabilities. An OSLC service is accessible via a service provider. Moreover, OSLC tool adapters are specialized tool extensions, which allow sharing data, signals or even parts of a user interface [22]. The integration solution requires building two tool adapters for the HD design tool and the TFS2015 version control tool. These adapters allow tools to integrate using an OSLC-compliant web service. The HD tool does not contain any version control specific implementation and can be integrated with any tool providing version control in the same OSLC manner. Moreover, through its OSLC adapter, TFS2015 can also offer version control functionality to any other tool which implements a client to the OSLC service.

This mentioned integration implementation "was applied on a set of project files and compared to existing direct tool-to-tool integration. The functionality of the OSLC tool adapter for TFS2015 and HD extension matched all existent functionality, demonstrating the integration of a design tool with a version control repository using the OSLC domain. The proposed approach also removes all TFS2015-specific code and functionality from the HD tool, eliminating the need to manage and update HD installation in case of changes to the version control system" [22]. The integration also addresses traceability between versioned items and items from an external requirements management tool. Introducing traceability is enabled by the fact that versioned items are exposed as OSLC resources, which can, in turn, be referenced by any other OSLC resource. The HD's OSLC extension allows selecting an OSLC requirement during the check-in operation, and attaching the newly created versioned item as the implementation of this requirement.

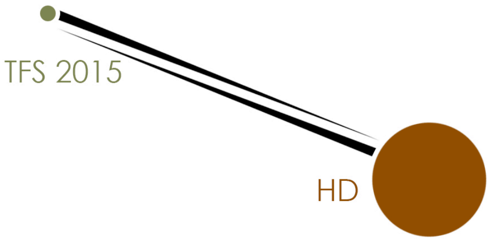

Figure 4. Node-link diagram of the development toolchain after integration.

Once the integration solution was implemented and deployed, we have used tracking software again to understand the effect of integration on the development toolchain. The same tracking application is used for this purpose with the same data collection format, for the same task. Moreover, the same developer has been tracked to minimize the effect of different user behavior. The results showed that the developer used HD 94% of the time and changed to TFS2015 only 6% of the time. Table II summarized the usage data after integration where developer only uses these two tools. Figure 4 shows the new NLD data visualization for the integrated toolchain. This new data shows that the developer does not need to switch between tools as much as the non-integrated scenario. One obvious reason behind this is the usefulness of the integration solution. Now, the developer uses version control through the HD adapter without a need to use IE to search for the information about the requirements. This also illustrated that the productivity of the developer increased since, after integration, the developer needed less time to complete the same task.

TABLE II. DATA ABOUT THE TOOL USAGE AND INTERACTIONS DURING THE DEVELOPMENT OF CSTP AFTER INTEGRATION.

	Total Usage	**HD**	**IE**	**TFS 2015**	**TFS 2005**
HD	94%	0%	0%	100%	0%
IE	0%	0%	0%	0%	0%
TFS 2015	6%	100%	0%	0%	0%
TFS 2005	0%	0%	0%	0%	0%

The case study included only one developer and this is one of the limitations that retain us to generalize the finding of the study. However, it still inholds valuable information about the importance of the understanding of current interoperability situation in development toolchains. This case study also illustrates how important the data is for improving the understanding of complex CPS development toolchains. Even with small data, the toolchain architect could have a better understanding and take decisions according to real data. Moreover, the study exemplified how this visualization can be used to develop common understanding about the interoperability state with other stakeholders.

V. FUTURE WORK

Although NLDs are a very successful way to show the overall picture, they do have some disadvantages. For instance, different layouts could create ambiguity when the node number increases [16]. Ghoniem et al. [17] showed that density has a strong impact on readability in these diagrams.

One way to approach this problem and increase the readability of NLDs is to use algorithms to obtain clustered graph layouts that optimize certain aesthetic criteria. For this reason, we suggest using a balloon layout [14, 15, 18, 19] for larger toolchain. Authors in [24] present different data visualization techniques including balloon layout to visualize the development toolchain. In future, repeating a similar case study for a larger toolchain and including more developers' activity would be beneficial to be able to further generalize the results.

VI. Conclusion

In this paper, we explained the interoperability challenge in CPS development environments, and presented data visualization as a promising approach for developing a better understanding of interoperability of CPS development toolchains. The studied development toolchain and the tools are described, in addition to the data collection and visualization process. The study showed that the NLD visualization has the potential to inform toolchain architects about the interoperability situation and help them to make decisions accordingly, especially for small toolchains. In the case study, this understanding lead to the integration of the two predominantly used tools, an HD design tool and a TFS2015 version control tool. This integration positively affected the performance of a developer and helped them to stay focused on one tool. The developer's tool usage data shows that integration eliminated the need for IE and increased the abilities of the HD tool. Last but not least, the study underlines the importance of data in the development environment and motivates the CPS industry to collect and use data in the decision-making process for better interoperability.

Acknowledgment

The research leading to these results has received funding from the ARTEMIS JU under grant agreement no 621429 for "EMC2 - Embedded multi-core systems for mixed-criticality applications in dynamic and changeable real-time environments", and from Vinnova under DIARIENR 2014-01225. The authors would like to acknowledge Vicki Derbyshire who performed language editing and proofreading.

References

[1] E. A. Lee, "Cyber-physical systems-are computing foundations adequate". In Position Paper for NSF Workshop On Cyber-Physical Systems: Research Motivation, Techniques and Roadmap (Vol. 2), October 2006.

[2] M. Törngren, A. Qamar, M. Biehl, F. Loiret, and J. El-Khoury, "Integrating viewpoints in the development of mechatronic products," Mechatronics, pp.745-762, 2014.

[3] D. Gürdür, F. Asplund, J. El-khoury, F. Loiret, and M. Törngren, "Visual analytics towards tool interoperabilty: A position paper," In Proceedings of the 11th Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, 2016a, pp. 139-145, ISBN 978-989-758-175-5.

[4] M. R. Bendre, and V. R. Thool, "Analytics, challenges and applications in big data environment: a survey," Journal of Management Analytics, pp.1-34, 2016.

[5] D. Gürdür, F. Asplund, J. El-khoury, "Measuring Toolchain Interoperability in Cyber-physical Systems," In Proceedings of the 11th International Conference on System of Systems Engineering, 2016b.

[6] T. C. Ford, J. M. Colombi, S.R. Graham, and D.R. Jacques, "Survey on Interoperability Measurement," Air Force Institute of Technology, 2007.

[7] G. E. Lavean, "Interoperability in defense communications," IEEE Transactions on Communications , 1980, pp.1445-1455.

[8] D. Mensh, R. Kite, and P. Darby. "A methodology for quantifying interoperability," Naval Engineering Journal, 1989.

[9] C. Amanowicz, and C. P. Gajewski. "Military communications and information systems interoperability," Military Communications Conference, 1996.

[10] T. Clark, and R. Jones, "Organisational interoperability maturity model for C2," In Proceedings of the 1999 Command and Control Research and Technology Symposium, June 1999.

[11] J. A. Hamilton Jr, J. D. Rosen, and P. A. Summers, "An interoperability road map for C4ISR legacy systems," Space and Naval Warefare Systems Center, 2002.

[12] J. Posada, C. Toro, I. Barandiaran, D. Oyarzun, D. Stricker, R. de Amicis, E. B. Pinto, P. Eisert, J. Döllner, and I. Vallarino, "Visual computing as a key enabling technology for industrie 4.0 and industrial internet," IEEE computer graphics and applications, 35(2), 2015, pp. 26-40.

[13] Smart Manufacturing Leadership Coalition. *SMLC-NSF Workshop* [Online]. Available from: https://smartmanufacturingcoalition.org/smlc-nsf-workshop

[14] D. G. Battista, P. Eades, I. G. Tollis, and R. Tamassia, Graph drawing: Algorithms for the visualization of graphs, 1999.

[15] I. Herman, G. Melançon, and M. S. Marshall, "Graph visualization and navigation in information visualization: A survey," Visualization and Computer Graphics, IEEE Transactions, 2000, pp. 24-43.

[16] R. Keller, C. M. Eckert, and P. J. Clarkson, "Matrices or node-link diagrams: Which visual representation is better for visualising connectivity models?," Information Visualization, 2006, pp. 62-76.

[17] M. Ghoniem, J. D. Fekete, and P. Castagliola, "On the readability of graphs using node-link and matrix-based representations: A controlled experiment and statistical analysis." Information Visualization, 2005, pp. 114-135.

[18] M. Dickerson, D. Eppstein, M. T. Goodrich, and J. Y. Meng, "Confluent drawings: Visualizing non-planar diagrams in a planar way," In Graph Drawing, September 2003, pp. 1-12

[19] D. Holten, "Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data," IEEE Transactions on Visualization and Computer Graphics, 2006, pp. 741-748.

[20] M. Baur, and U. Brandes, "Multi-circular layout of micro/macro graphs," In Graph Drawing, September 2007, pp. 255-267.

[21] S. Emgell, "Cyber-physical systems of systems—definition and core research and innovation areas," Working Paper of the Support Action CPSoS, 2014.

[22] L. Lednicki, G. Sapienza, M, E. Johansson, T. Seceleanu, T. and D. Hallmans, "Integrating version control in a standardized service-oriented tool chain," In IEEE 40th Annual Computer Software and Applications Conference, June 2016, pp. 323-328.

[23] OSLC community, *Open Services for Lifecycle Collaboration*, [Online]. Available from: http: //open-services.net/

[24] D. Gürdür, J. El-khoury, T. Seceleanu, and L. Lednicki, "Making Interoperability Visible: Data Visualization of Cyber-Physical Systems Development Tool Chains," Journal of Industrial Information Integration, 2016, doi: 10.1016/j.jii.2016.09.002.

[25] M. Javanbakht, R. Rezaie, F. Shams, and M. Seyyedi, "A new method for decision making and planning in enterprises," In 3rd International Conference on Information and Communication Technologies: From Theory to Applications, April 2008, pp. 1-5.