Managing Social Challenges in Cross-Organizational Event-Based Systems

Laura S. Thiele[1,2] and Nico Brehm[1]

[1] University of Applied Science Jena, Department of Industrial Engineering, Carl-Zeiß-Promenade 2, 07745 Jena, Germany
[2] German Aerospace Center (DLR), Institute of Data Science, Mälzerstraße 3, 07745 Jena, Germany
[laura.thiele,nico.brehm]@eah-jena.de

Abstract. During the last decade the manufacturing industry focused on the realization of industry 4.0 aspects. Besides the implementation of new technologies, existing software structures also need to be reviewed and adapted in this context. To stay competitive in the global market, especially small and medium-sized companies need to emphasize on better cooperation with other organizations. This leads to the implementation of cross-organizational distributed software system structures. The development of distributed systems faces different challenges – technical and code-centric as well as social challenges. This paper focuses on the social challenges that appear in distributed development processes. After defining the main challenges, the paper introduces a development approach that is based on the integration of a Federated Management System (FMS). FMS is a technical approach to minimize social challenges by the generation of system transparency and the provision of a platform for communication and interaction. It facilitates a distributed system development of cross-organizational event-based systems.

Keywords: Software Development · Distributed Systems · Event-Driven Systems · Wiki-Based System Management.

1 Introduction

The manufacturing industry is subject to current trends in the market, such as increasing product variety, custom and individual fabrication, as well as reducing production and delivery times [4,12,13]. Due to these trends, the manufacturing industry started to adapt traditional structures of their manufacturing in order to implement the ideas of Industry 4.0.

Most of the Industry 4.0 approaches are based on comprehensive data acquisition and usage. By now, data is captured by several different devices and systems. It needs to be collected and stored in order to process it for further purposes. To support the manufacturing industry in its current challenges, several research projects, e.g., [5,6,7,10,15], were started to develop system architectures that enable comprehensive data acquisition and usage in the manufacturing

J. Manner, S. Haarmann, S. Kolb, O. Kopp (Eds.): 12th ZEUS Workshop, ZEUS 2020, Potsdam, Germany, 20-21 February 2020, published at http://ceur-ws.org/Vol-2575

environment. Since research emphasizes that especially small and medium-sized enterprises (SME) may increase their competitiveness through improvement of cooperation with federated companies (suppliers, purchasers, and subcontractors) [11], research projects started to consider an inter-organizational data exchange in their developed systems [5,7,10].

By analyzing the outcome of these researches, it is noticeable that all developed architectures show distributed system structures. Which means that parts of the systems are separated but work together like a single system.

Mishra & Tripathi [8] categorize distributed systems into systems where: a) only software and hardware are distributed, b) users are distributed and c) both – users, hardware and software are distributed. Since software development strategies changed from a fixed group of developers towards an open and community-based development, like in open source software projects, systems can also have distributed developers. Hence, the distributed system schema of Mishra & Tripathi can be enhanced by the category of distributed developers, as displayed in Figure 1.

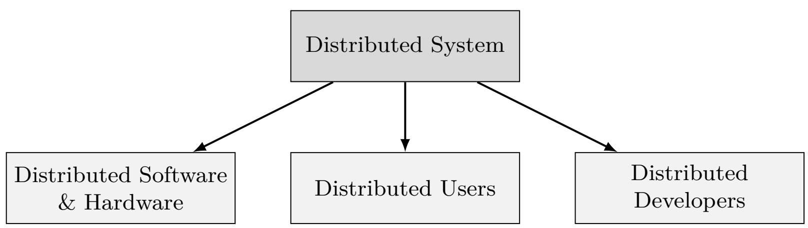

Fig. 1. Distributed Systems

Considering the schema that is displayed in Figure 1, a distributed system may belong to one up to three of the displayed categories. Each category implicates specific challenges that need to be considered [2,3,8].

Effective collaborations within distributed systems require adequate technical solutions. But especially for systems that have distributed users and/or developers, an adoption of good organizational practices and development processes is also necessary [3]. In this paper we do not focus on the technical and code-centric perspective of distributed system development, instead we are focusing on the social part of the development process and how to address it with technical solutions.

2 State of the Art

The integration of system users and developers with different knowledge in the development phase of a software is not a new aspect. All agile methods already implemented this aspect to enhance the field of knowledge and to get feedback early in the development phase [3]. In most agile software projects the participating developers are known. Therefore, it is easier to manage the cooperation between them. Software projects that are developed by distributed developers, e.g., via an open platform like GitHub, show the problem that system users and developers are unknown. This complicates the coordination and cooperation.

Allaho & Lee [1] analyzed software projects on GitHub in their research. They recognized that project documentation is very important to help developers to understand the project. It helps to create transparency of the project structure and behavior.

Transparency is a very important aspect, also for Carbot et al. [3]. They analyzed open source projects and occurring problems in their work and determined that most of the projects and its leadership miss transparency for the contributors. Furthermore, they recognized that most projects hardly ever follow any kind of democratic practices, which makes it difficult for contributors to influence the development of the project. During their research, Carbot et al. found out that many projects are only developed by a few members. It seems to be difficult to motivate others to contribute.

Promoting contribution not only refers to system developers, it also includes system users. Wikipedia is probably the most known example. There, a high amount of system users are motivated to contribute to the system [14]. Wikipedia provides a framework that makes it easy to contribute to the system even if the user is not a software developer. The principle of a wiki-based platform for communication and coordination of software projects was already implemented in different tools that support agile software development projects. It is a convenient method to exchange information between participants.

The importance of a medium for communication between system developers was also addressed in the work of Müller [9]. He pointed out that communication supports collaboration within a system's environment and motivates new contributors to get into the community.

3 Social Challenges in Distributed System Development

Based on the observations that were made, the social challenges in distributed software project can be summarized as:

C1 how to provide transparency (motivated by [1,3,14]),
C2 how to attract and support new contributors (mot. by [3,9,14]) and
C3 how to optimize collaborations (mot. by [3,9]).

Transparency (C1) needs to be established in the entire distributed system. That means, architectural aspects (relating the hardware and software) need to be transparent as well as system contributions from users (e.g., feedback) or developers (e.g., system extensions). System transparency helps to understand the systems structure and behavior. Hence, it can be seen as the main social challenge in distributed systems.

System contributions (C2) are very important as they expand and improve the system. System transparency helps to minimize entry barriers for new contributors. Furthermore, the knowledge about a system's functional and operational range may inspire people to contribute new extensions and improvements.

The optimization of collaborations (C3) is important to maintain efficiency in system development and system operation. The system, or its environment, should provide a possibility (e.g., a tool) for users and developers to manage and optimize their collaborations.

4 Context and Assumptions

This section provides an overview of the context and the assumptions under which the development strategy, presented in the following section, was developed.

Since processes in the manufacturing industry are mostly triggered by occurring events (e.g., start and finish of tasks, error announcements, attainment of states, such as temperatures) it is assumed to have a system that has an event-driven architecture: information is published to a broker that routes it to subscribing services.

To support collaborations among small and medium-sized companies, an inter-organizational system usage should be supported by the system. That may enable companies to integrate machines and devices to the system and share information with a federated group of organizations.

Furthermore, the system should enable decentralized system development, where system components are developed by autonomous groups of software developers. They may develop new system components (services) that can be added to the system.

In order to develop a strategy that supports the development of such systems, in summary, we assume that:

- the system shows a distributed system structure in all categories (software & hardware, users, and developers)
- the system is based on an event-based approach (publish-subscribe pattern)
- the system is situated and developed in a trusted domain (no security considerations)
- the system will be developed in an agile manner (continuous system development and enhancement)

5 The Federated Development Strategy

To support the development of cross-organizational event-based systems, we developed a federated system development strategy that helps to handle the social challenges as they were stated in Section 3. To manage the challenges we developed an architecture of a Federated Management System (FMS) that needs to be integrated to the systems architecture.

Figure 2 displays an architectural sketch of our federated development strategy for cross-organizational event-based systems. In the upper part, the figure displays a system architecture that exchanges event messages between two companies via a broker (the assumed system situation). In this situation, social challenges appear because of the distributed system usage.

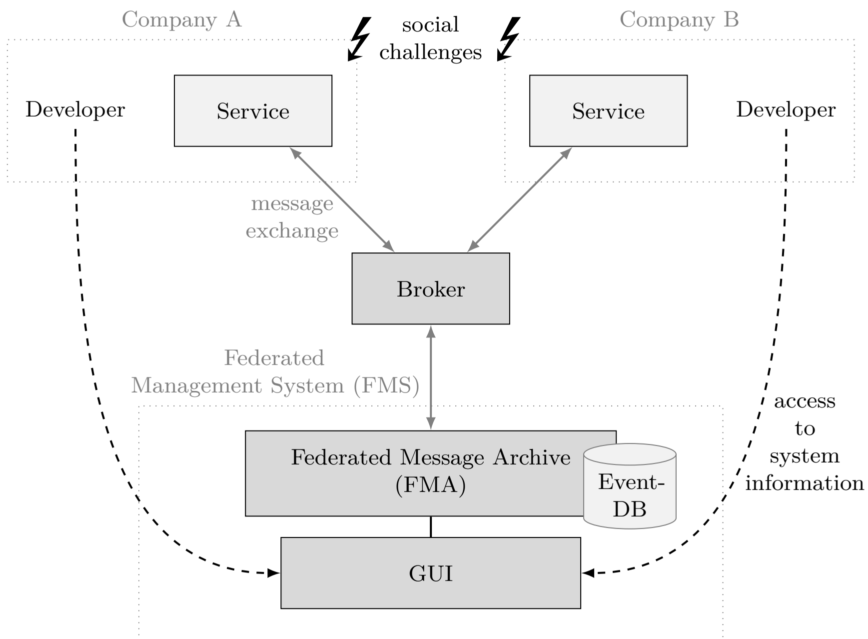

Fig. 2. Federated Management System

The lower part of the figure shows the FMS that needs to be integrated. It consists of a Federated Message Archive (FMA) component that includes an event-database and a graphical user interface (GUI). The FMA component is connected to the systems broker. Whereas other services only listen to specific events, the FMA subscribes to all messages that might be posted in the system. Consequently, the broker forwards all messages to the FMA which stores them in the event-database. In doing so, the FMA collects several information about posted events, such as message type, message type's publishing frequency, message content and more.

The GUI of the FMS, see Figure 3, provides access to the information in the FMA. It is the operative component that uncovers system information to users and enables transparency in the system environment. As displayed in Figure 3,

Fig. 3. Screenshot of an FMS GUI Prototype

a user may search for specific information that is shared via the system, e.g., information related to orders. Hence, the system displays a list of messages that are related to the search term. In this example the user can see that there exist more than one message type that is related to orders: topic '*order/new*' and topic '*order/accepted*'. Furthermore, the user can explore the messages content, e.g., the single attributes, in order to get the information he or she is locking for.

By searching for a specific message topic, the user can obtain further information about a specific message type. After forwarding a search query of a message topic to the FMA, all collected messages of the specific message type are compared and analyzed by the FMA component. Findings are formatted and sent to the GUI were they are displayed to the user. Hence, the user can obtain specific information about a message type, such as the frequency in which it is used or specific information about attributes that are held in the message content (e.g., range of attributes values or most common value).

The GUI of the FMA can be accessed by every developer that belongs to the trusted group of organizations in which the system is implemented (see Figure 1). This ensures a high degree of transparency within the whole system community (C1).

New developers that are interested in system contribution and join the cross-organizational system environment get access to the FMS. Due to transparency that is provided by the FMS, new system developers are able to get to know the systems structure and behavior. By analyzing events that were stored in the FMS, new developers can learn how messages are structured and how they can be subscribed in order to use their information for further purposes.

Besides the provision of system information, the FMS furthermore provides a wiki-based area where system developers are able exchange information. By this, system users get the opportunity to ask for special services and for system extensions within the system community. Concrete demands on system extensions may animate developers to contribute to the system (C2). Especially for new contributors it might be easier and more motivating to start contribution on a concrete task.

The wiki-based area of the FMS supports communication among the system participants. This is very important to optimize collaborations (C3). The wiki-based area may not only be used for placing demands, but also for coordinating contributions. Developers can quote on which functionality they are currently working and several developers are able to coordinate their collective work via, e.g., Kanban boards. Furthermore, new developers can ask for help within the community and senior system developers may share their knowledge and experiences.

6 Conclusion and Future Work

In this paper we presented an approach to handle social challenges in cross-organizational event-based systems. First, we presented a classification model of distributed systems which shows that distributed systems can be categorized into three categories: hardware & software distributed systems, as well as users distributed and developers distributed. Depending on their category of distribution, distributed systems face different challenges. Systems that have distributed users and/or developers have to handle not only technical and code-centric issues, but also social challenges. In this paper, we focused on social challenges and summarized them as follows: 1) provision of transparency, 2) attraction and support of new developers and 3) optimization of system collaborations. Finally, we presented FMS and how this system approach that can be used to mitigate social challenges in distributed systems.

In the future work it is planned to evaluate FMS in real system environments. To enable such evaluation, cross-organizational event-based systems will be implemented in the manufacturing environment and system developers will be asked to enhance the system by components that supports the manufacturing process. The evaluation of FMS's usability will be done in user studies that will be undertaken with the developers of the involved companies. There will be a mixture of developers with more and less experiences to establish the different needs and requirements to the FMS. Based on the evaluations outcome, FMS will be further adapted and improved.

References

1. Allaho, M.Y., Lee, W.C.: Trends and behavior of developers in open collaborative software projects. In: 2014 International Conference on Behavioral, Economic, and Socio-Cultural Computing (BESC2014). IEEE (Oct 2014)

2. Braubach, L., Pokahr, A.: Addressing Challenges of Distributed Systems Using Active Components. In: Intelligent Distributed Computing V, pp. 141–151. Springer Berlin Heidelberg (2011)
3. Cabot, J., Izquierdo, J.L.C., Cosentino, V.: Community-based software development for MDE tools. In: Joint Proceedings of EduSymp 2016 and OSS4MDE 2016 (2016)
4. Denkena, B., Dittrich, M.A., Uhlich, F., Maibaum, L., Mörke, T.: Das gentelligente Werkstück. In: Handbuch Industrie 4.0, pp. 295–321. Hanser Fachbuchverlag, Munich (2017)
5. DIN: Erweiterung des EPCIS-Ereignismodells um aggregierte Produktionsereignisse zur Verwendung in betrieblichen Informationssystemen (Jan 2016)
6. DIN: Reference Architecture Model Industrie 4.0 (Apr 2016)
7. Fuchs, J., Oks, S.J., Franke, J.: Platform-based service composition for manufacturing: A conceptualization. Procedia CIRP **81**, 541–546 (2019)
8. Mishra, K.S., Tripathi, A.K.: Some Issues, Challenges and Problems of Distributed Software System. In: (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (4) (2014)
9. Müller, M.: Agile Challenges and Chances for Open Source: Lessons Learned from Managing a FLOSS Project (Nov 2018)
10. Otto, B., Lohmann, S., Steinbuß, S., Teuscher, A.: IDS Reference Architecture Model Industrial Data Space. Tech. rep., International Data Space Association and Fraunhofer-Gesellschaft zur Förderung der angewandten Forschung e.V., Dortmund (2018)
11. Peris-Ortiz, M., Ferreira, J.J.: Cooperative and Networking Strategies in Small Business. Springer-Verlag GmbH (2016)
12. Reinhart, G., Zühlke, D.: Handbuch Industrie 4.0, chap. Von CIM zu Industrie 4.0, pp. XXXI–XL. Hanser Fachbuchverlag, Munich (2017)
13. Schuh, G., Reuter, C., Hauptvogel, A., Brambring, F., Hempel, T.: Ergebnisbericht des BMBF-Verbundprojektes PROSENSE : hochauflösende Produktionssteuerung auf Basis kybernetischer Unterstützungssysteme und intelligenter Sensorik, chap. 1 Einleitung, pp. 1–13. Apprimus Verlag Aachen, Aachen (2015)
14. van Steen, M., Pierre, G., Voulgaris, S.: Challenges in very large distributed systems. Journal of Internet Services and Applications **3**(1), 59–66 (Nov 2011)
15. Theorin, A., Bengtsson, K., Provost, J., Lieder, M., Johnsson, C., Lundholm, T., Lennartson, B.: An Event-Driven Manufacturing Information System Architecture for Industry 4.0. International Journal of Production Research **55**(5), 1297–1311 (Jul 2016)