An Approach to Development of System Architecture in Large Collaborative Projects

Gökan May, Dimosthenis Ioannidis, Ifigeneia N. Metaxa, Dimitrios Tzovaras, Dimitris Kiritsis

An approach to development of system architecture in large collaborative projects

Gökan May[1], Dimosthenis Ioannidis[2], Ifigeneia N. Metaxa[3], Dimitrios Tzovaras[2], Dimitris Kiritsis[1]

[1]EPFL, ICT for Sustainable Manufacturing, EPFL SCI-STI-DK, Station 9, CH-1015 Lausanne, Switzerland; [2]Information Technologies Institute (ITI), CERTH Thessaloniki 57001, Greece; [3]ATLANTIS Engineering SA, Thessaloniki 55535, Greece
gokan.may@epfl.ch, djoannid@iti.gr, metaxa@abe.gr, dimitrios.tzovaras@iti.gr, dimitris.kiritsis@epfl.ch

Abstract. Innovation projects in manufacturing domain often include several end users with different use cases that require a special approach for converging to one architecture solution, which addresses the needs of all end users. The communication between end users and developers in different research and software development projects should be supported correspondingly. This paper describes an approach to development of software intensive system architecture in large collaborative projects that extends traditional approaches with different architecture viewpoints and additional iterative steps aiming to design a main platform integrating project solutions. The approach is applied and validated in a large collaborative EU-funded H2020 research project entitled Z-Fact0r, i.e. Zero-defect manufacturing strategies towards on-line production management for European factories. Based on the standard ISO/IEC/IEEE 42010 that implies a process based on a set of relevant architecture viewpoints and following the architecture development approach introduced in this study, Z-Fact0r platform is defined by the following viewpoints: conceptual, functional, information, and deployment.

Keywords: software engineering; system architecture; collaborative projects; manufacturing.

1 Introduction

In today's complex world of IT, business and manufacturing, innovation may often emerge thanks to the involvement in research and development projects of heterogeneous and interdisciplinary teams whose members are coming from different backgrounds and expertise [1]. Such projects should accommodate both the diverse user needs as well as the various interests and goals of the team development members. The communication between end users and developers in different research and software development projects should be supported correspondingly [2]. This paper describes an approach to development of software intensive system architecture in large collaborative projects that extends traditional approaches with different architecture viewpoints and additional iterative steps aiming to design a main platform integrating project solutions. The approach is applied and validated in a large collaborative EU-funded H2020 research project entitled Z-Fact0r [3]. Based on the standard ISO/IEC/IEEE 42010 that implies a process based on a set of relevant architecture viewpoints and following the architecture development approach introduced in this study, Z-Fact0r platform is defined by the following viewpoints: conceptual, functional, information, and deployment.

2 Approach to development of system architecture

Figure 1 shows the architecture development approach.

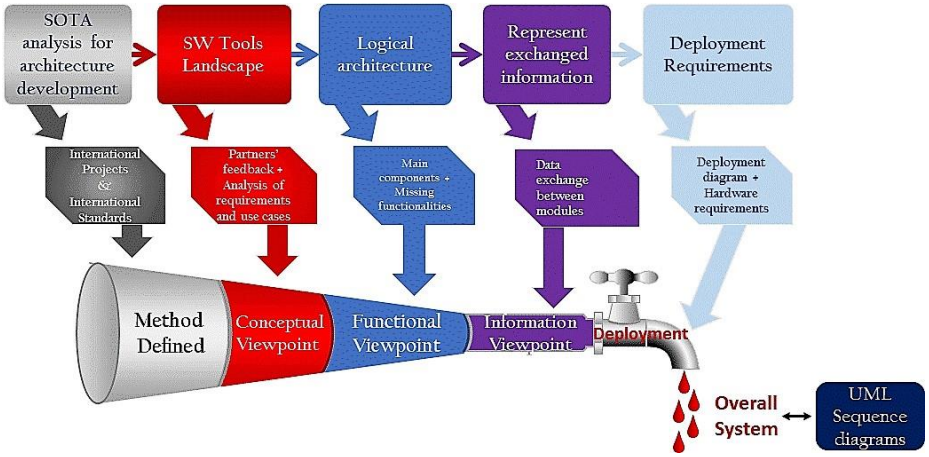

Fig. 1. Architecture development approach

In the State of the art (SOTA) analysis step, relevant large collaborative projects and international standards are analyzed from a system and architecture perspective in order to (i) define the architecture development method, and (ii) identify critical aspects for the system architecture that should be taken into account.

The design of the main system architecture is carried out based on the knowledge and insights from relevant international projects, and the documentation of the architecture is based on the standard ISO/IEC/IEEE 42010 (2011) "Systems and software engineering — Architecture description" [4]. This standard establishes a methodology for the architectural description of software intensive systems. The standard implies a process based on a set of relevant architecture viewpoints, and in this study we consider four viewpoints, i.e. conceptual view, functional view, information view, and deployment view.

The System Architecture of the main platform is described starting from the high-level architecture and tools' description down to the definition of data flow and tools' inner structure. The architectural description includes aspects related to the identification of the major system components, how they should interact and how their external interfaces should be defined.

The first high-level description that leads to the definition of System Architecture consists in identifying and classifying all the tools presented in what can be called the software tools' landscape, i.e. the conceptual view. The conceptual architecture provides an overview of the tools along with their dependencies and affiliation to the responsible leading partners. In this step, project documents, partners' feedback and the input from the analysis of the project requirements and use cases play an important role.

In the functional view the components, their functionality, and their interactions are described. Functional viewpoint contains all the functions that the system performs and the responsibilities as well as interfaces of the functional elements with respect to the

relationship between them. These functions are described using UML diagrams. In this step, a first version of the functional architecture is defined by (i) identifying the technologies and software modules to be provided by the project partners, and (ii) thinking of and identifying the missing modules and functionalities which are required for the implementation of the main platform and that will be developed and integrated along the project. Following this description of the main functional view, all technology partners provide a detailed description of the expected inputs, generated outputs and main functionalities as well as the component diagrams for each tool.

In the next step, a schema is provided to represent the exchanged information between the envisioned components. The information view describes the application domain models and the data flow as well as the distribution. Finally, the deployment requirements are collected from the partners and deployment diagram is designed accordingly. The deployment view describes how and where the system will be deployed, which physical components are needed, as well as the dependencies, hardware requirements and physical constraints. Finally, UML sequence diagrams designed for each use case should clarify how the platform will work and which components are relevant to achieve different tasks. The final system architecture thus represents a key cornerstone in setting the basis for the successful deployment of future project results by providing the means for their harmonization in a coherent infrastructure.

3 Implementation of the approach in Z-Fact0r

The EU H2020 Factory of the Future (FoF) project Z-Fact0r focuses on Zero-defect manufacturing strategies towards on-line production management for European factories. The Z-Fact0r Platform and solution introduce five multi-stage production-based strategies targeting (i) early detection of the defect (Z-DETECT), (ii) prediction of the defect generation (Z-PREDICT), (iii) prevention of defect generation by recalibrating the production line (multi-stage), as well as defect propagation in later stages of the production (Z-PREVENT), and (iv) reworking/remanufacturing of the product, if this is possible, using additive and subtractive manufacturing techniques (Z-REPAIR), and finally (v) management of the aforementioned strategies through event modelling, KPI (key performance indicators) monitoring and real-time decision support (Z-MANAGE). Accordingly, Z-Fact0r architecture will encompass the design and development of a diverse set of technologies with different specifications and requirements aligned with these 5 main Z-Fact0r strategies. Consequently, the Z-Fact0r Platform will be demonstrated in three different pilots belonging to the three end users (i.e., Microsemi, Interseals, and Durit), where each one targets at different aspect of the operation and activities.

Following the logic of the Z-Fact0r platform design and development, in the following subsections we develop the Z-Fact0r system architecture which comprise four different architecture viewpoints (i.e. conceptual, functional, information, deployment).

3.1 Conceptual viewpoint

The first high-level description that led to the definition of the System Architecture consisted in identifying and classifying all the software tools to be developed in Z-

Fact0r. In this step of the architecture development, we collect and categorize the technologies and software components that the individual partners of the Z-Fact0r project brought in with them. In addition, the partners' expertise has been quickly identified and used as best as possible in this first process. This has also helped us to identify gaps in the architecture that needed to be filled in order to achieve the platform envisioned by the Z-Fact0r project. Figure 2 presents this landscape by proposing a compact representation of the involved tools.

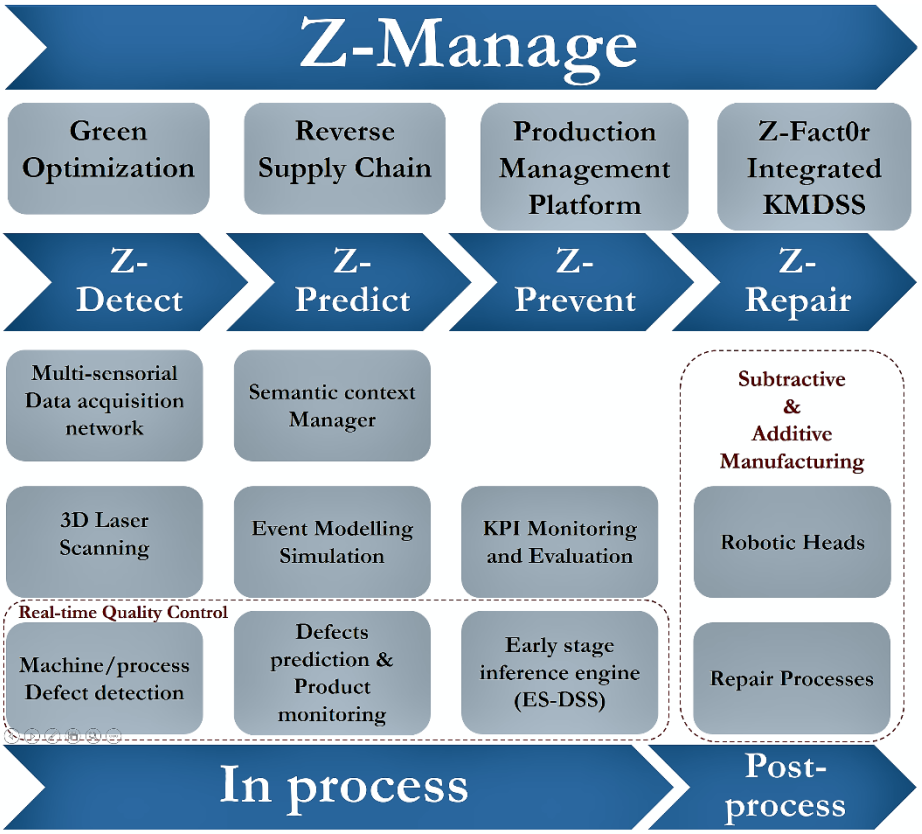

Fig. 2. Z-Fact0r conceptual architecture viewpoint

3.2 Functional viewpoint

Beginning from the conceptual architecture defined in the previous section, we have started putting the components into an initial architecture, identifying services and dependencies within the platform. We have also added new components in order to cover all the required Z-Fact0r functionalities. The result of this process is presented in this section, which includes the defined overall Z-Fact0r functional architecture. As it can be observed in Figure 3, Z-Fact0r overall architecture has been subdivided in different layers, as specified initially, where components are of different nature and offer different functionalities.

In order to define the System Architecture in such a way that it represents a usable schema for the implementation, a UML Component diagram has been created (shown in Figure 3), where each tool has been represented as a component. Component diagrams are particularly useful when applied to distributed development. In fact, the initial architectural modelling efforts during cycle 0 focuses on identifying the landscape of the system and UML component diagrams which enable the modeling of the high-level software components, and more importantly the interfaces to those components. Once the interfaces are defined and agreed to teams, it is much easier organizing the development effort between sub-teams.

Z-Fact0r functional viewpoint thus contains all the functions that the system should perform as well as the responsibilities and interfaces of the functional elements and the relationship between them. These functions are described using UML diagrams. Figure 3 shows the component diagram view of the overall Z-Fact0r architecture.

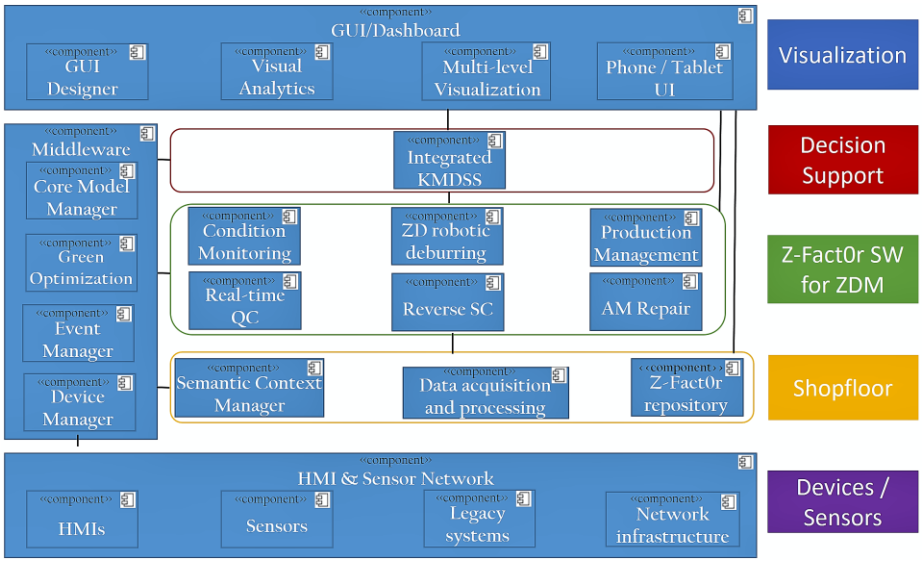

Fig. 3. Z-Fact0r functional architecture viewpoint

To sum up, the main components, their functionality, and their interactions are described in the functional view. Accordingly, the main components for Z-Fact0r architecture are:

- HMI & Sensor Network, which includes sensors, actuators, HMIs for humans to provide input to machines and thus the overall system, cameras, network infrastructure, legacy systems, etc.
- Shop-floor components which comprise semantic context manager, data acquisition and processing including 3D laser scanning, and Z-Fact0r repository.
- Middleware including device manager, event manager, green optimizer, and core model manager.
- Z-Fact0r software modules for zero-defect management in manufacturing, which builds the service layer and includes Z-Fact0r specific tools such as real-time quality

control, production management, reverse supply chain, zero-defect robotic deburring, and additive/subtractive manufacturing repair.

- Decision Support System (DSS) component, which will supervise and provide feedback for all the processes executed in the production line, evaluating performance parameters and responding to defects, keeping historical data.
- Finally, a visualization layer has been foreseen, which includes GUI/Dashboard designer, Visual Analytics Module, multi-level visualization component, and phone/tablet UI, etc.

Having identified the main functional view, we have then clarified the role of each layer and components. As first step of this phase, a template has been defined in order to collect a short description of all components brought by the partners. In particular, this template has aimed to collect the following component information: description of the main functionalities, related services, dependencies, inputs needed and outputs provided. This detailed description is introduced by a short description of the tool, pointing out the associated task and the principal user(s) of the tool. Then, the functional requirements are pointed out, meaning that for each tool a list of the main inputs, the main outputs and the main functionalities are listed and explained. After this analysis, the software structure of the tool architecture is given by the use of an UML Component Diagram allowing a first, deeper analysis on how the tool will be implemented. Eventually, a complete description of the modules included in the detailed view is provided in order to point out the responsibilities of each module and their interactions with the overall System Architecture.

3.3 Information viewpoint

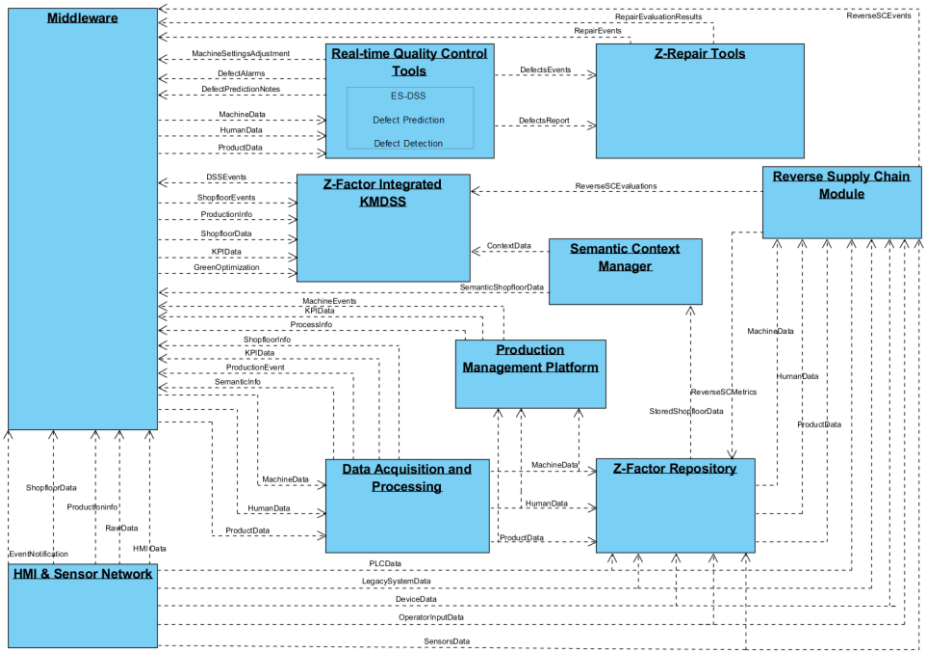

Fig. 4. Z-Fact0r information architecture viewpoint

The information viewpoint describes the way that the architecture stores, manipulates, manages, and distributes information. The ultimate purpose of virtually any computer system is to manipulate information in some form, and this viewpoint develops a complete but high-level view of static data structure and information flow. Accordingly, this section illustrates a schema to represent the exchanged information between the envisioned components of the Z-Fact0r platform. Figure 4 highlights this information viewpoint diagram.

3.4 Deployment viewpoint

The deployment view needs to document the required deployment environment of the Z-Fact0r platform, which depends on the pilot areas and their topology. In this section, a first component diagram indicating the deployment view of the Z-Fact0r components is depicted in Figure 5.

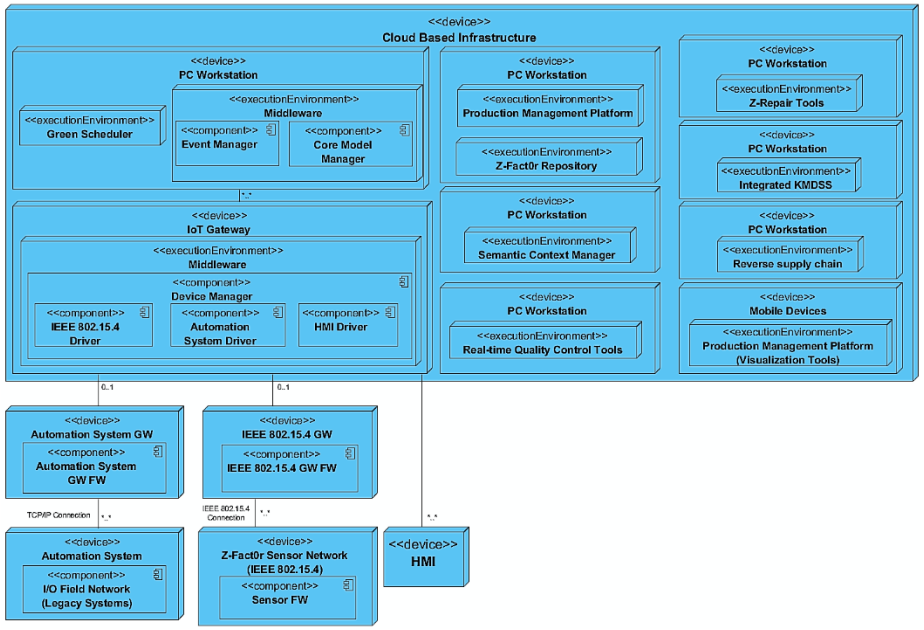

Fig. 5. Z-Fact0r deployment viewpoint

The HMI & Sensor network is comprised by a variety of heterogeneous sensors including the preinstalled factory automation system, which provides information about the production activities and the status of the factory infrastructure. Furthermore, it is comprised by the other sensors which will be used within the Z-Fact0r project. All these sensors/devices will be connected to the Middleware Device Manager through corresponding gateways, which will forward the information from the shop floor to the Middleware. The Device Manager, running in a dedicated PC named IoT Gateway, is equipped with all the necessary drivers, so as to understand and interpret the multi-sensorial information. The Device Manager is interconnected with the Middleware Event Manager, which could be located on the same PC (i.e. IoT Gateway) or even on

another workstation as it is depicted in Figure 5. The integrated Knowledge Management and Decision Support System (KMDSS) is interconnected with the Middleware Device Manager in order to acquire data stemming from the sensor network. This information is subsequently processed, generating recommendations which are then provided to users or other subsystems in the form of Events through the Event Manager.

All Z-Fact0r components, which will be connected among each other with the Middleware Event Manager, could be installed either on the same PC workstation or distributed to a number of PCs. Figure 5 illustrates the case where the Z-Fact0r components are distributed to various PC workstations. All the Z-Fact0r components compose the cloud-based Z-Fact0r infrastructure and will be interconnected among each other with a dedicated intranet, which could be either wired, wireless or even a combination of wired and wireless.

4 Conclusion

The introduced approach has proven its applicability on the realistic example of a large collaborative research project, demonstrating its ability to support the users and developers on their way from project requirements and software tools' description to the unified system architecture. Mutual analysis of the architecture by end users and software developers with different competences inspired all sides and supported the definition of new solutions. The resulting Z-Fact0r platform satisfies the identified business requirements of the industrial end-users and will act as main driver for the interpretation and explanation of the activities related to the design and management of a zero-defect manufacturing system at large. The application of this method in large innovative projects would facilitate their success.

5 Acknowledgements

The work presented in this paper is supported by the project Z-Fact0r which is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 723906. The paper reflects only the authors' views and the Commission is not responsible for any use that may be made of the information it contains.

6 References

1. Koukias, A., May, G., Vasyutynskyy, V., Nadoveza, D., McCarthy, J. C., Taisch, M., & Kiritsis, D. (2013). Approach on Analysis of Heterogeneous Requirements in Software Engineering. *IFAC Proceedings Volumes*, *46*(7), 372-377.
2. Kovács, G. L., & Paganelli, P. (2003). A planning and management infrastructure for large, complex, distributed projects—beyond ERP and SCM. *Computers in Industry*, *51*(2), 165-183.
3. Z-Fact0r Project [online] Available: http://www.z-fact0r.eu/ [Accessed: 1st March 2017].
4. ISO/IEC/IEEE 42010. Systems and software engineering — Architecture description, 2011.