

# Software Architecture Quality Attributes of a Layered Sensor-Based IoT System

Zeljko Stojanov<sup>1</sup>, Dalibor Dobrilovic<sup>1</sup>

<sup>1</sup>University of Novi Sad, Technical Faculty "Mihajlo Pupin", Zrenjanin, Serbia

## Abstract

Quality of complex technical systems is highly dependent on the quality of integrated software systems. Software system quality is determined with their functionalities and quality attributes that describe how the functionalities are performed. Although there exists a large list of software architecture quality attributes that technical systems can satisfy, they do that to a greater or lesser extent. Internet of Things (IoT) systems are complex socio-technical systems that include a variety of software elements distributed on specific hardware components and servers. Selecting and fulfilling of the most suitable quality attributes during system design is a challenging task. In this paper, we present our subjective experience with scalability, maintainability, security, availability and portability quality attributes during design of a layered sensor-based IoT system for monitoring industrial environmental conditions. Further research directions are also presented.

## Keywords

software architecture, system architecture, IoT system, quality attributes, layered architecture

## 1. Introduction

Production of high quality software products is challenging demand for software industry in highly dynamic and competitive market [1]. This assumes continuous improvement software characteristics in order to satisfy users and market demands. Software architecture is essential for the design of effective and usable software systems, their integration into complex socio-technical systems, and their easy and controllable evolution to adapt to changing business needs. Since technical systems should respond to proposed business requirements and constraints, software architecture should help in mapping them to specific design of a system [2]. From the technical point of view, software architecture reflects fundamental organization of a software system, its components and their relationships, as well as relations of a software system with its environment. Creation of a loosely coupled architecture by minimizing dependencies between components is the key point in architecture design. With this approach with minimized dependencies, changes during software evolution are localized and do not propagate through the architecture, which significantly reduces costs and complexity of maintenance.

Functionalities of software systems reflect functional requirements and constraints derived from business objectives. However, nonfunctional requirements define how identified functionalities will be provided to system user, and should be considered together with functional

---


*Workshop Information Technologies: Algorithms, Models, Systems (ITAMS), September, 2021, Irkutsk, Russia*

✉ zeljko.stojanov@uns.ac.rs (Z. Stojanov); dalibor.dobrilovic@uns.ac.rs (D. Dobrilovic)

🆔 0000-0001-6930-5337 (Z. Stojanov); 0000-0002-3083-5725 (D. Dobrilovic)



© 2021 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

requirements. The following types of nonfunctional requirements can be distinguished [3]: (1) *technical constraints* relate to selection of the appropriate software technologies for development, (2) *business constraints* relates to decisions driven from business that affect development (e.g. reduction of costs by moving to open source platforms), and (3) *quality attributes* relates to issues of concern to application users and other stakeholders, while affecting the way of system functioning. In the *Software Architect's Handbook* [4] the following definition of quality attributes is proposed:

*Quality attributes are properties of a software system and a subset of its non-functional requirements. Like other requirements, they should be measurable and testable. Software quality attributes are benchmarks that describe the software system's quality and measure the fitness of the system.*

Design of effective software architectures is based on common patterns or styles that facilitate specific organization and communication of software components [3]. Each architecture pattern has specific and well-known characteristics, which makes it appropriate for specific business and technical requirements. Large and complex systems usually are based on multiple patterns, which are combined to respond to specific requirements. Architecture patterns describe component types and their properties, their relations and topology, but also outline benefits and drawbacks of using that pattern. Detailed overview of contemporary architecture patterns, such as layered, event-driven, service-oriented or microservices architecture, is presented in [5, 6], while [7] presents reference architecture for for Industrial Internet of Things (IIoT) systems.

Considering quality attributes lead to attribute based architecture design, which incorporates predictability in architecture design [8]. Using certain attributes requires understanding them and implementing the most suitable way for measuring their effects on architecture. Since software architecture is considered in the early phase of software production cycle, determining the right architecture that satisfy proposed quality requirements rendered as quality attributes is of great importance for the success of the whole project [9]. The use of quality attributes provides support for evaluation of proposed software and system architecture. During software system design several quality attributes are considered. The level of quality attribute satisfaction during system design affects the overall performance and quality of the system. Quality attributes should be considered in the early phase of eliciting software requirements, but their testing and evaluation should be carried out during the whole system life cycle [10, 11, 4].

Based on the above discussion and our experience with different types of software architecture patterns in data intensive enterprise software systems [12] and smart manufacturing systems [13], the objective of this paper is to present discussion of specific quality attributes during the design of layered sensor-based IoT system for monitoring the industrial environmental conditions in manufacturing settings.

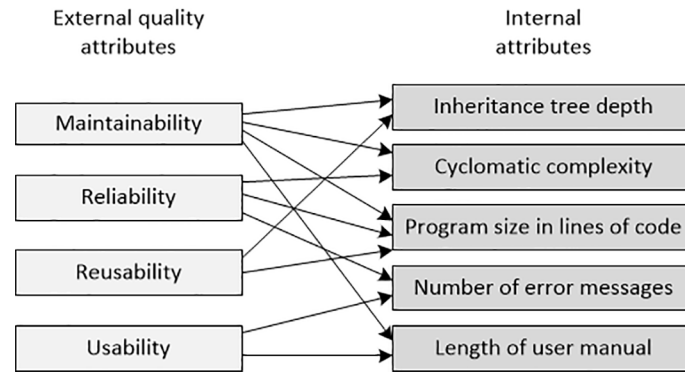
## **2. Software architecture quality attributes in IoT systems**

Software architecture quality attributes were recognized more than 40 years ago as very important for optimizing and saving software life cycle costs [14]. Since then, they attracted significant attention by researchers and practitioners from software industry. Quality attributes,

together with the proposal of quality assessment process should be included in the relevant project documentation (e.g. quality plan). A comprehensive overview of the most commonly used architecture quality attributes in the selected software engineering literature focused on software architectures [3, 11, 4, 2] is presented in Table 1.

**Table 1**  
The most commonly used architecture quality attributes

<b>Quality attribute</b>	<b>Description</b>
<i>Performance</i>	Defines the amount of work software product should perform in a proposed time for correct operation (e.g. increased throughput, shorter response time).
<i>Scalability</i>	Defines the software product capability of increasing in size, which usually means increasing processing capability with existing hardware (e.g. additional connections, increased amount of requests for processing, increased data).
<i>Maintainability</i>	Defines how easy is software product for maintenance, including frequent modification based on changed functional requirements (e.g. easier modifiability, extensibility and flexibility).
<i>Security</i>	Defines mechanisms to support security requirements such as authentication, authorization, encryption and integrity.
<i>Availability</i>	Defines software product quality to be able to be continuously used (e.g. reliability and easy recovery from failures, easy replication of the most critical elements of the software product).
<i>Integrability</i>	Defines how easily software product can be integrated into more complex technical systems.
<i>Interoperability</i>	Defines how easily software product can exchange data with other software systems within more complex technical systems.
<i>Portability</i>	Defines how easily software product can be run on different software or hardware platforms (it is based on using platform independent technologies).
<i>Testability</i>	Defines how easily software product can be tested in a given context, which depends on the software complexity, the controllability and isolability of specific software components, and automatability of test actions and processes.
<i>Usability</i>	Defines how easily software product can be used for accomplishing typical tasks by users. It is closely related to <i>Learnability</i> that defines how easily users can learn to effectively use software product.
<i>Energy efficiency</i>	Defines how software product consumes energy, which becomes very important in contemporary distributed IoT based systems. This attribute should be balanced with other attributes such as reliability, availability or usability.



**Figure 1:** Internal and external attributes of software product (adapted from [11])

In practice, it is not feasible to measure majority of quality attributes, like maintainability or usability, since they present subjective opinion affected by person education, experience and knowledge. In order to measure these attributes, some internal attributes of software product that can be easily measured are determined. These internal attributes relate to measurable properties of software products, such as cyclomatic complexity, size of files or modules, number of error messages or failures. Somerville [11] proposed the relationships between internal attributes and external quality attributes as it is presented in Figure 1.

High complexity and variety of implementation scenarios of IoT systems brings several challenges in designing architecture of the whole system and software components. Since IoT and embedded systems are used in many domains such as critical systems, fault-tolerance systems or real-time systems, considering their quality characteristics is essential for ensuring their efficient and safe usage [15].

Muccini and Moghaddam [16] presented a systematic mapping study on IoT architectural styles and patterns. Based on the review of scientific literature the following architectural concerns were identified: styles, distribution patterns, reference architectures, architectural platforms, architecture activities, and quality attributes. Based on literature review the following quality attributes were identified as the most important for IoT architecture: scalability, security, interoperability, and performance. The following quality attributes were identified as less important: privacy, availability, mobility, reliability, resiliency, and evolvability. Regarding the architecture styles, mostly used were layered, cloud based and service oriented.

Kim [17] proposed quality model of IoT applications based on the following characteristics of IoT systems: participation of hardware devices, collaboration of software and hardware components, mobility and connectivity of devices, monitoring of devices, and limited resources. The following quality attributes were proposed in the model: (1) *functionality* – functional coverage with functional requirements (suitability) and accuracy of functionalities (accuracy), (2) *reliability* – maintaining a specific level of performance, which includes fault tolerance and recoverability from undesired situations, (3) *efficiency* – ensuring acceptable performance regarding speed and resource usage, and (4) *portability* – capability to adapt to changed environment, which includes installability, replaceability. Proposed model was evaluated within a virtual environment and showed acceptable characteristics.

Ghasemi et al. [18] proposed the architecture of a system based on smart home technology for the elderly health-care. The proposed system architecture contains three main components: home server, data center, and physician. Wifi/3G/4G was used for data transfer between components, while sensed data were transferred via Bluetooth or Zigbee technology from the sensors to the home server. The proposed system architecture was design to meet the following quality attributes: availability, performance, modifiability, security, usability and interoperability. The proposed architecture evaluation was done by using the Architecture Tradeoff Analysis Method (ATAM) scenario-based approach [19]. The scenarios were produced through brainstorming and later ranked by stakeholders (user, system architect, and software developer) via voting. Based on the architecture evaluation, the authors discussed risks related security and availability issues, and their influence on system performance.

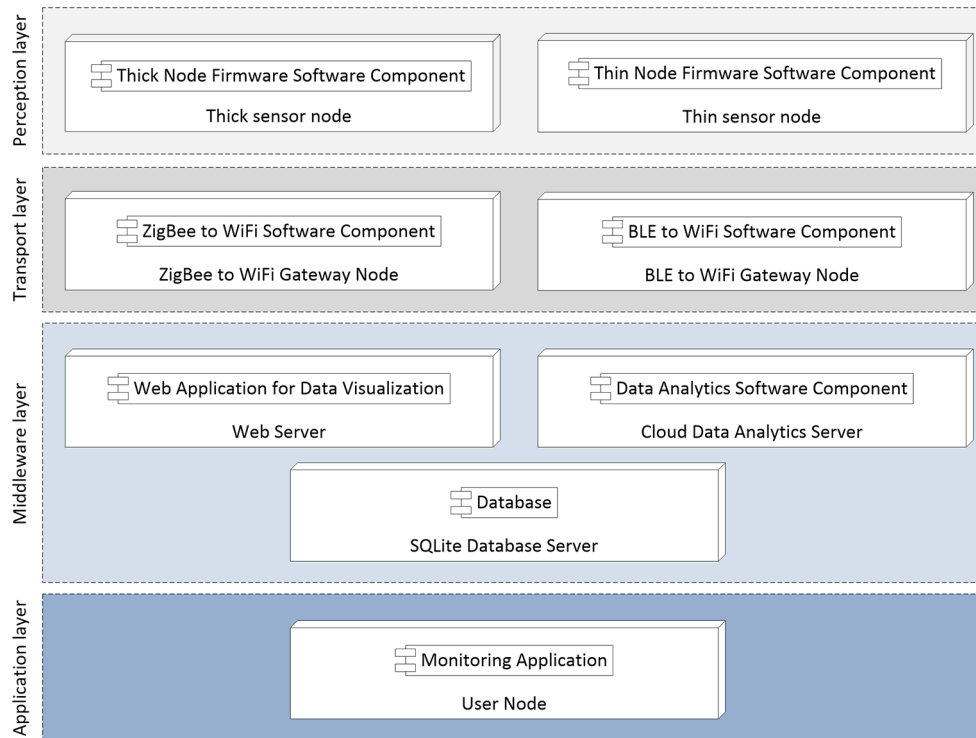
Temkar and Bhaskarb [20] presented a study with evaluation of the quality of a system for agriculture field monitoring based on Wireless Sensor Network (WSN) and IoT. The system architecture contains components that enable data collection and monitoring, data processing, execution, and feedback. Quality evaluation of the system was done by using Analytic Hierarchy Process (AHP). Quality model of the system considers the following quality attributes: functional suitability, compatibility, maintainability, usability, performance efficiency, security, reliability, and portability. Evaluation of software components of the system revealed that the proposed software quality attributes are suitable for ensuring the quality of software part of the system.

### 3. Quality attributes of a layered sensor-based IoT system

In this section, a model of an IoT system for monitoring environmental conditions within industrial settings is briefly presented, as a basis for discussing quality attributes of software architecture. The system was initially developed as a student project based on open-source software and hardware components [21], while its software architecture is presented in the paper [13]. The system is designed with a layered architecture which has many positive characteristics regarding the fulfilment of typical quality attributes (see Table 1). Software architecture of the system at the highest level is presented in the deployment diagram in Figure 2. Presented architecture shows deployment of software components at nodes and their distribution in four layers (perception layer, transport layer, middleware layer, and application layer).

Design of a system with diversity of software and hardware components is highly challenging regarding the conformance to quality attributes of the system, especially for systems used in dynamic and evolving industrial settings. The selection of the quality attributes for software architecture is based on subjective experience of the authors with software architecture and IoT systems, which can be classified as expert based qualitative evaluation of software architecture quality [22, 9]. The following quality attributes were considered in the system design: scalability, maintainability, security, availability, and portability.

*Scalability* of the system relates to both hardware and software components in terms of their capacity to fulfil all possible requirements posed to the system. We consider scalability of the system based on the three dimensions proposed in [23]: (1) *processing capacity* relates to increasing the processing capability of software components from perception layer to servers and user services, (2) *information capacity* relates to increasing the storage in database component



**Figure 2:** Software architecture model of a IoT system for monitoring industrial environmental conditions

of the system and local storage of individual software components, and (3) *connectivity* relates to increasing the number of access points for users and number of sensing elements in the perception layer. These dimensions were built into the system regarding the selection of layered architecture pattern, selection of hardware components, selection of technologies and design of individual software components distributed in all layers.

*Maintainability* is essential for ensuring proper and continuous functioning of the system regardless of problems that may occur during its use. It should be built into the system architecture from the early design decisions, and consider software complexity, technology, tools and human factor [24]. Maintainability of the proposed architecture is supported by design in which components are loosely coupled, while the propagation of potential modifications is straightforward between components in different layers. For example, adding new sensors in perception layer leads to appropriate modifications in all three components in the middleware layer, and similarly in the application layer.

*Security* focuses on enabling reliable and secure IoT systems. The primary concern in the system we propose is the authenticity of the IoT devices. The IoT devices authentication can be achieved with various approaches applicable in variety of applications [25, 26]. This aspect is important for ensuring the originality of data send by trusted devices. The second concern is the encryption of collected data send for ensuring the confidentiality of data send. Considering of extensive usage of messaging protocols such as MQTT and CoAP in IoT nowadays their support for security issues is important as well [27]. This attribute is focused on the security of

the system core elements (servers, DB, and network devices), ensuring the operability of the system, and confidentiality of data stored.

*Availability* of the system ensures that core elements of the system enable continuous real time access to the services for end users. This characteristic relates to fault tolerance and minimizing time for repairing the whole system or specific components [28]. Availability of the software components is slightly different depending on the layer in which they are deployed, since it depends on how easy is to access hardware components were software components were located. Availability of software components was considered through the selection of open source software solutions and modular architecture design with clear relationships between components, which enable easy, fast and cheap maintenance and minimization of potential failures.

*Portability* defines how easily software product can be run on different software or hardware platforms [29]. The right way for ensuring this is in usage of standardized technologies and protocols, thus giving the opportunity for implementation of open-source multiplatform software. Such software for the core of the system can be messaging protocol brokers such as Mosquitto and RabbitMQ, and database servers such as MongoDB, Cassandra, MySQL etc. At the client side, the development of web-based applications ensures the platform independence, and service accessibility from computers, tablets and smart-phones.

## 4. Conclusions

Quality of software systems is essential for performance of complex socio-technical systems. IoT systems have been widely used in different domains, and software part of these systems is quite complex regarding used technologies and business processes. This requires considering quality attributes of software architecture in order to ensure proper functioning of the whole systems and minimizing risks and costs in the system life cycle. In this paper, subjective evaluation of software architecture quality attributes of a layered IoT system for monitoring working conditions is presented. Scalability, maintainability, security, availability and portability were discussed as the most important quality attributes of the presented system.

Further work will be directed to development of metrics suitable for measuring the selected quality attributes of software architecture of the whole system, as well as individual software components distributed in layers. Considering other quality attributes, such as interoperability, usability and energy efficiency in further improvements of the system are promising development and research options.

## Acknowledgments

Ministry of Education, Science and Technological Development, Republic of Serbia, supports this research under the project "The development of software tools for business process analysis and improvement", project number TR32044.



## References

- [1] F. N. Colakoglu, A. Yazici, A. Mishra, Software product quality metrics: A systematic mapping study, *IEEE Access* 9 (2021) 44647–44670. doi:10.1109/ACCESS.2021.3054730.
- [2] L. Bass, P. Clements, R. Kazman, *Software architecture in practice*, SEI series in software engineering, 4th ed., Addison-Wesley Professional, Upper Saddle River, NJ, USA, 2021.
- [3] I. Gorton, *Essential Software Architecture*, 2nd ed., Springer-Verlag Berlin Heidelberg, 2011. doi:10.1007/978-3-642-19176-3.
- [4] J. Ingeno, *Software Architect's Handbook*, Packt Publishing, Birmingham, UK, 2018.
- [5] D. Duggan, *Enterprises software architecture and design: entities, services, and resources*, John Wiley & Sons, Hoboken, NJ, USA, 2012.
- [6] M. Richards, *Software Architecture Patterns: Understanding Common Architecture Patterns and When to Use Them*, 1st edition ed., O'Reilly Media, Sebastopol, CA, USA, 2015.
- [7] S.-W. Lin, B. Miller, J. Durand, G. Bleakley, A. Chigani, R. Martin, B. Murphy, M. Crawford, *The Industrial Internet of Things Reference Architecture*, Technical Report of Industrial Internet Consortium (IIC) Technology Working Group and its Architecture Task Group IIC:PUB:G1:V1.80:20170131, Milford, MA, USA, 2017.
- [8] M. Klein, R. Kazman, *Attribute-Based Architectural Styles*, Technical Report CMU/SEI-99-TR-022, Pittsburgh, PA, USA, 1999.
- [9] S. Moaven, J. Habibi, A fuzzy-ahp-based approach to select software architecture based on quality attributes (FASSA), *Knowledge and Information Systems* 62 (2020) 4569–4597. doi:10.1007/s10115-020-01496-7.
- [10] M. Svahnberg, C. Wohlin, L. Lundberg, M. Mattsson, A method for understanding quality attributes in software architecture structures, in: *Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering, SEKE '02*, Ischia, Italy, 2002, pp. 819–826. doi:10.1145/568760.568900.
- [11] I. Sommerville, *Software Engineering*, 9 ed., Addison Wesley, Boston, MA, USA, 2011.
- [12] Z. Stojanov, D. Dobrilovic, J. Stojanov, Extending data-driven model of software with software change request service, *Enterprise Information Systems* 12 (2018) 982–1006. doi:10.1080/17517575.2018.1445296.
- [13] Z. Stojanov, D. Dobrilovic, G. Jotanovic, D. Perakovic, G. Jausevac, V. Brtko, Software architectures in smart manufacturing: Review and experiences, in: I. Bychkov, Z. Stojanov, A. Tchernykh, A. Feoktistov (Eds.), *Proceedings of the 1st International Workshop on Advanced Information and Computation Technologies and Systems (AICTS 2020)*, volume 2858 of *CEUR Workshop Proceedings*, Irkutsk, Russia, 2020, pp. 160–168.
- [14] B. W. Boehm, J. R. Brown, M. Lipow, Quantitative evaluation of software quality, in: *Proceedings of the 2nd International Conference on Software Engineering, ICSE '76*, San Francisco, CA, USA, 1976, pp. 592–605. doi:10.5555/800253.807736.
- [15] Z. Tamrabet, T. Marir, F. MOKHATI, A survey on quality attributes and quality models for embedded software, *International Journal of Embedded and Real-Time Communication Systems* 9 (2018) 1–17. doi:10.4018/IJERTCS.2018070101.
- [16] H. Muccini, M. T. Moghaddam, Iot architectural styles, in: *Proceedings of the 12th European Conference on Software Architecture*, volume 11048 of *Lecture Notes in Computer Science*, Springer, Cham, Madrid, Spain, 2018, pp. 68–85. doi:10.1007/978-3-030-00761-4\_5.



- [17] M. Kim, A quality model for evaluating iot applications, *International Journal of Computer and Electrical Engineering* 8 (2016) 66–76.
- [18] F. Ghasemi, A. Rezaee, A. M. Rahmani, Structural and behavioral reference model for iot-based elderly health-care systems in smart home, *International Journal of Communication Systems* 32 (2019) e4002. doi:10.1002/dac.4002.
- [19] R. Kazman, M. Klein, P. Clements, ATAM: Method for Architecture Evaluation, Technical Report CMU/SEI-2000-TR-004, Pittsburgh, PA, USA, 2000.
- [20] R. Temkar, A. Bhaskarb, Quality assurance of iot based systems using analytic hierarchy process, *Turkish Journal of Computer and Mathematics Education* 12 (2021) 6759–6767. doi:10.17762/turcomat.v12i10.5542.
- [21] N. Petrov, D. Dobrilović, N. Mirosavljev, N. Grujić, Wireless sensor networks for monitoring living and working conditions, in: *Proceedings of the 8th scientific-professional conference Information technology for e-education*, Banja Luka, Bosnia and Hercegovina, 2016, pp. 170–176.
- [22] T. Rosqvist, M. Koskela, H. Harju, Software quality evaluation based on expert judgement, *Software Quality Journal* 11 (2003) 39–55. doi:10.1023/A:1023741528816.
- [23] G. Brataas, P. Hughes, Exploring architectural scalability, in: *Proceedings of the 4th International Workshop on Software and Performance, WOSP '04*, Redwood Shores, CA, USA, 2004, pp. 125–129. doi:10.1145/974044.974064.
- [24] B. C. D. Anda, Assessing software system maintainability using structural measures and expert assessments, in: G. Canfora, L. Tahvildari (Eds.), *Proceedings of the 23rd International Conference on Software Maintenance (ICSM 2007)*, Paris, France, 2007, pp. 204–213. doi:10.1109/ICSM.2007.4362633.
- [25] C. Jung, J. Choi, R. Jang, D. Mohaisen, D. Nyang, A network-independent tool-based usable authentication system for internet of things devices, *Computers & Security* 108 (2021) 102338. doi:10.1016/j.cose.2021.102338.
- [26] T. Shah, S. Venkatesan, Authentication of iot device and iot server using secure vaults, in: *Proceedings of the 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, New York, NY, USA, 2018, pp. 819–824. doi:10.1109/TrustCom/BigDataSE.2018.00117.
- [27] V. Seoane, C. Garcia-Rubio, F. Almenares, C. Campo, Performance evaluation of coap and mqtt with security support for iot environments, *Computer Networks* 197 (2021) 108338. doi:10.1016/j.comnet.2021.108338.
- [28] V. Cortellessa, R. Eramo, M. Tucci, From software architecture to analysis models and back: Model-driven refactoring aimed at availability improvement, *Information and Software Technology* 127 (2020) 106362. doi:10.1016/j.infsof.2020.106362.
- [29] S. Pennycook, J. Sewall, V. Lee, Implications of a metric for performance portability, *Future Generation Computer Systems* 92 (2019) 947–958. doi:10.1016/j.future.2017.08.007.