Automation in Construction 85 (2018) 182–192

Contents lists available at ScienceDirect

Automation in Construction

journal homepage: www.elsevier.com/locate/autcon

Supporting smart construction with dependable edge computing infrastructures and applications

Petar Kochovski, Vlado Stankovski*

Faculty of Civil and Geodetic Engineering, University of Ljubljana, Ljubljana, Slovenia

ARTICLE INFO

Keywords:
Smart construction
Dependability
Internet of Things
Container-based systems
Edge computing

ABSTRACT

The Internet of Things (IoT) such as the use of robots, sensors, actuators, electronic signalization and a variety of other Internet enabled physical devices may provide for new advanced smart applications to be used in construction in the very near future. Such applications require real-time responses and are therefore time-critical. Therefore, in order to support collaboration, control, monitoring, supply management, safety and other construction processes, they have to meet dependability requirements, including requirements for high Quality of Service (QoS). Dependability and high QoS can be achieved by using adequate number and quality of computing resources, such as processing, memory and networking elements, geographically close to the smart environments. The goal of this study is to develop a practical edge computing architecture and design, which can be used to support smart construction environments with high QoS. This study gives particular attention to the solution design, which relies on latest cloud and software engineering approaches and technologies, and provides elasticity, interoperability and adaptation to companies' specific needs. Two edge computing applications supporting video communications and construction process documentation are developed and demonstrate a viable edge computing design for smart construction.

1. Introduction

Millions of new Internet-connected devices commonly known as the Internet of Things (IoT) have been developed in the past years. According to Gartner, Inc. [1] their number will rise up to 20 billion by 2020. These devices include sensors, actuators, smartphones, signalization, robots and so on, and are being developed to facilitate smart, automated, secure and sustainable environments and working processes in various industrial sectors including construction.

The construction process is commonly known as a very complex process which requires a lot of automation. Smart application possibilities include construction site automation, robots assisted construction, infrastructures monitoring, safety monitoring, home automation and many more.

Some applications require a lot of networking and processing support, such as coordination and logistic operations for robots, video-stream analysis for infrastructure monitoring and safety signalization at construction sites. Frequently, IoT devices run on batteries and are without a strong processor. This usually leads to the necessity to offload computations to nearby microservers and datacenters, where higher frequency processors can be obtained. Generally, the network latency, bandwidth, network throughput, round trip time and similar QoS metrics are important in the IoT applications to achieve higher dependability.

Hence, there are significant technical requirements that must be met by smart construction applications to make them of true business value. The first necessary step in the development process for smart applications is therefore the identification of their detailed technical requirements. Requirements analysis must take into consideration QoS attributes such as network related metrics and performance, and other non-functional requirements, such as reliability, availability, security, safety and similar, which are commonly known in systems engineering as dependability attributes.

A major obstacle for the introduction of advanced IoT applications in the construction sector at the moment is their unacceptable Quality of Service (QoS) hindering possibilities for real-time operation. In the past years, a variety of case studies for IoT applications in the construction sector have been published [2–5]. Indeed, such applications can only achieve their business value, if they meet high QoS standards, that is, dependability requirements.

Significant networking, processing and memory resources are required by smart applications, for example, to dynamically process data coming from cameras, sensors, robots and smartphones and to provide feedback and control loops for signalization and actuators. In order to

* Corresponding author.
E-mail address: vlado.stankovski@fgg.uni-lj.si (V. Stankovski).

http://dx.doi.org/10.1016/j.autcon.2017.10.008
Received 26 May 2017; Received in revised form 5 September 2017; Accepted 11 October 2017
Available online 05 November 2017

build an adequate approach, we can rely on existing cloud services designs in general, and dependability solutions in particular. Mell et al. [6] define cloud computing as a model, which enables universal, on-demand network access to shared pool of computing resources, such as: networks, servers, storage, applications and services that can be rapidly provisioned and managed with minimal effort. Such resources can be obtained dynamically from cloud providers.

From technology viewpoint, in order to be able to address the requirements of smart IoT applications, there are some new developments in the cloud computing domain, which may be appropriate for latency-sensitive applications. Edge and fog computing as specific areas of cloud computing intend to improve the design of cloud applications by organising processing closer to the data sources at the edge of the network [7], thus, reducing communications latency and improving reliability and other aspects. Hence, due to the nature of the edge computing concept, it may be necessary for construction companies to invest in some computing and networking infrastructures to be able to support the operation of smart IoT applications.

Recently, two new initiatives for interoperability standards have emerged – the Cloud Native Computing Foundation (CNCF) [8] and the OpenFog Consortium [9]. These initiatives rely on a variety of new technologies and projects' outcomes, including: specific operating systems for edge computing, such as CoreOS [10] and RancherOS [11] that can be installed on devices such as microservers, routers and Raspberry PIs; technologies for container based virtualization, such as Docker [12]; technologies for container orchestration, such as Kubernetes [13] and Mesos [14]; and advanced new software engineering tools, such as Fabric8 [15]. Nowadays, all these technologies can be used to build and operate elastic and interoperable IoT applications. The present study seeks to understand their utility in the context of various potential smart construction applications.

Therefore, the goal of this work is to assess the requirements of potential smart construction IoT applications and provide a practical edge computing architecture and design for their implementation. This requires the investigation of existing smart construction applications developed before the existence of the above-mentioned edge and fog interoperability initiatives.

Based on the number of ongoing activities which need smart IoT support, the requirements for underlying cloud and edge computing infrastructures may dynamically change. This, in turn, may lead to under or over occupancy of computing resources and consequently to unacceptable variations in the obtained QoS. Our design should therefore also address the requirement for cost effectiveness of the edge computing solution.

In order to demonstrate and test the dependability of the edge computing design, two IoT applications have been developed, covering construction video communications and building process documentation. Common to these two applications are their requirements to operate with large quantities of data, including data streams. This translates to high QoS requirements which must be met for their successful operation. The smart application design combines the benefits of edge computing, such as on demand self-service, broad network access, elasticity, low operation cost and adaptability to dynamically changing business needs.

The paper is structured as follows. Section 2 presents an overview of related works in the smart construction domain. Section 3 presents the approach, the system dependability requirements and the new edge computing architecture and design that can be used to support smart IoT applications. The deployment and implementation of the two applications, together with experimental results are presented in Section 4. Section 5 overviews some potential IoT application areas. Finally, Section 6 concludes the paper and reveals further development plans.

2. Related works – smart construction application opportunities

An initial review of the literature related to edge and fog computing applications in the construction domain revealed a gap. Following is a brief account of several application areas where new IoT edge computing applications would be greatly needed. Particular attention was paid to identify the technical requirements of potential applications, particularly the necessary QoS and the required computing infrastructures for their operation. Potential application areas that were investigated included construction site management, material supply management, security and safety of construction sites, real-time information sharing and communication.

Yuan et al. [16] propose a set of Cyber-Physical Systems (CPS) to prevent failure of temporary structures through real-time monitoring. Their solution is supported with a mobile application and a cloud-based database. Ren et al. [17] deal with the problem of collision accidents between cranes and immobile obstacles, which they try to reduce. Their solution is a real-time anti-collision system that warns of potential collisions and implements adequate safety strategies. The works [18] and [19] present solutions for safety at construction sites preventing collisions between building equipment as well as plant-pedestrian collisions by using different monitoring technologies. Their solution relies on the use of Radio-Frequency IDentification (RFID) tags, warning devices and is supported by a communication protocol and a user interface. This work of Jegen-Perrin et al. [20] describes a method to estimate the working area, and using RFID sensors to prevent possible collision accidents. It proposes the use of a camera-and-screen system to prevent the plant-pedestrian accidents with method to improve drivers visibility through real-time video monitoring. Time-critical requirements are common for all these applications.

In order to improve the construction process a variety of supply management and communication software prototypes have been proposed [21–23]. Prasad et al. [24] describe that material handling significantly affects the construction process, therefore they provide a software prototype for handling construction materials in a cost-effective manner with low human intervention. Wang et al. [25] provide a web-based solution that combines RFID technology and smart devices to enhance the effectiveness of data and information sharing at construction sites. Kim et al. [26] addressed in their study solutions for site monitoring, task management and real-time information sharing through a multi-tier computing infrastructure. Applications in supply management and communication may therefore offer significant improvements in the overall construction process. Common for these applications are high dependability requirements, including service availability, reliability, safety, security, portability, maintainability and scalability.

Lack of communication at construction sites often causes accidents and may lead to many severe on-site problems and delays in the construction process. There are many communication solutions available on the market, however, according to the review results presented by Shi et al. [27] the existing solutions for on-site communication between construction teams and site offices are error prone. Web-based technologies have also been considered as solutions that could improve the process. The following works [28–32] present potential benefits of using Web technologies and cloud infrastructures for such applications. The authors, however, mainly provide solutions for centralized processing and storing massive amounts of construction site data in databases in the cloud. However, real-time applications are obviously more challenging and have not been addressed.

Several studies focus on the use of various IoT devices in construction. The authors [4] presented how the IoT and related standards can improve the construction management system. Bai et al. [33] introduced tower crane safety supervising system based on IoT, which effectively supervises the tower production process safety. Most IoT applications developed so far have been optimized to run either locally or on Web-servers, not implementing any solutions to decrease the application/system downtime or decrease the amount of data-loss. The provided solutions do not fully exploit the benefits of cloud computing, such as virtualization, data replication and migration, orchestration across multiple clouds and so on.

Several of the above-mentioned applications rely on various communication solutions between different entities as well as file or information sharing. There is a great possibility to enhance them through the use of IoT. Some of these applications are designed to benefit from cloud computing, however, reaching adequate QoS, required for their operation is not always possible. While applications are developed to execute within a single data center, the geographical locations (e.g. construction sites), where automation is needed are constantly changing, so is changing the quality of the network connections. Therefore, reaching adequate level of QoS for their operation, due to centralized cloud computing solutions, is not always possible.

Based on the conducted literature review, it may be concluded that the potential of smart applications in construction is great and that none of the existing works have focused on the edge and fog computing concepts as an opportunity to address a wide range of QoS requirements.

3. Architecture and design

3.1. Methodology

The approach taken by this study is a practical one. From one side, it investigates the potential for smart IoT applications in the construction domain and their detailed technical requirements. The initial review of existing smart applications shows a gap that can be addressed in the near future by relying on more dependable edge computing infrastructures and applications. This section explains the methodology and the detailed design of smart construction applications, which is based on advanced cloud, edge and fog computing approaches and software engineering technologies.

The identified challenge is to define a set of Edge Management Services as depicted in Fig. 1 that can be used to manage Smart IoT Construction Environments. Within the smart environments various smart applications are deployed to support automation. The primary goal of the Edge Management Services is to contribute to higher QoS in comparison to existing cloud systems and therefore also to be able to meet stringent dependability requirements of every potential smart application.

The present work takes a top-down approach. Based on edge and fog computing definitions [34], exiting technology reviews [35,36], the identified technical requirements of the potential applications, and experience gained in the course of advanced cloud computing and software engineering projects, such as mOSAIC [37], SWITCH [38] and ENTICE [39], an adequate conceptual approach for edge computing in construction is prepared. It defines the main necessary services that can be used to obtain high QoS. Following the architecture and design, two exemplary edge computing applications were developed in order to demonstrate the viability of the approach and the potential benefits when meeting the required QoS. The applications that have been selected to illustrate the approach are related to video communications and file management, which are representative for various functionalities needed within smart construction environments.

3.2. Dependability requirements

Smart systems in the construction domain have to be designed as dependable. System dependability represents the system ability to deliver service that can be reasonably trusted. From a system failure perspective, the dependability can be also defined as system's ability to avoid failures that are more frequent or severe, and outage durations that are longer, than it is acceptable to the user [40], for example, to perform safety and business critical operations.

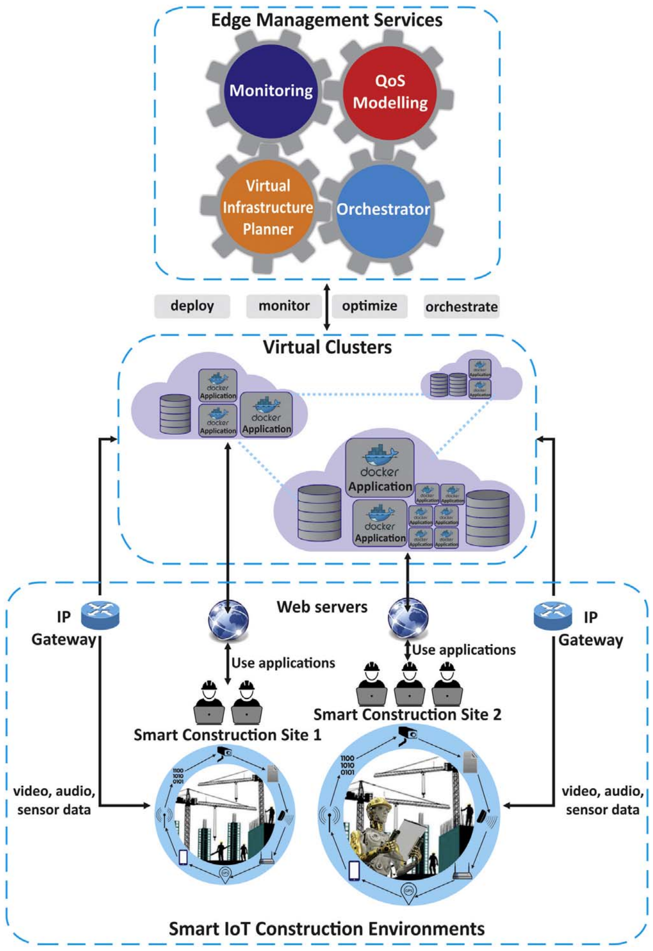

Fig. 1. Dependable edge computing infrastructure and smart IoT applications.

To measure the system dependability it is necessary to consider various dependability attributes, such as: high QoS, availability, reliability, safety, security, maintainability, integrity and many others. Addressing all of them within a single architecture is out of scope of this work. A dependable system can assume a subset of these attributes, which particularly relate to its safety and/or business critical operation, for example, the achieved QoS. Moreover, some of the dependability attributes are related, for instance, higher availability contributes to higher reliability and requires higher security, but that significantly increases the operational cost. Thus, building a dependable system in practice is a trade-off between attributes and their importance varies according to the business needs.

When the key dependability requirements are not satisfied, severe consequences are inevitable. For instance, this can cause additional financial costs, compromised safety at the construction site, construction process delays, loss of reputation and many other repercussions. In conclusion, providing the required dependability for IoT construction sites, where time-critical applications are used, certain dependability attributes have to be addressed and satisfied at all times.

3.3. Cloud, edge and fog computing approaches and technologies

Today, cloud computing is greatly used due to its properties, such as seamless scalability, configurability and the pay-per-use [41]. It has been actively researched, developed and implemented in different areas in the last decade, for example, in the course of the mOSAIC project. Past projects are now complemented with a new wave of edge and fog computing projects, one of which is SWITCH addressing software engineering for IoT and Big Data and another ENTICE addressing

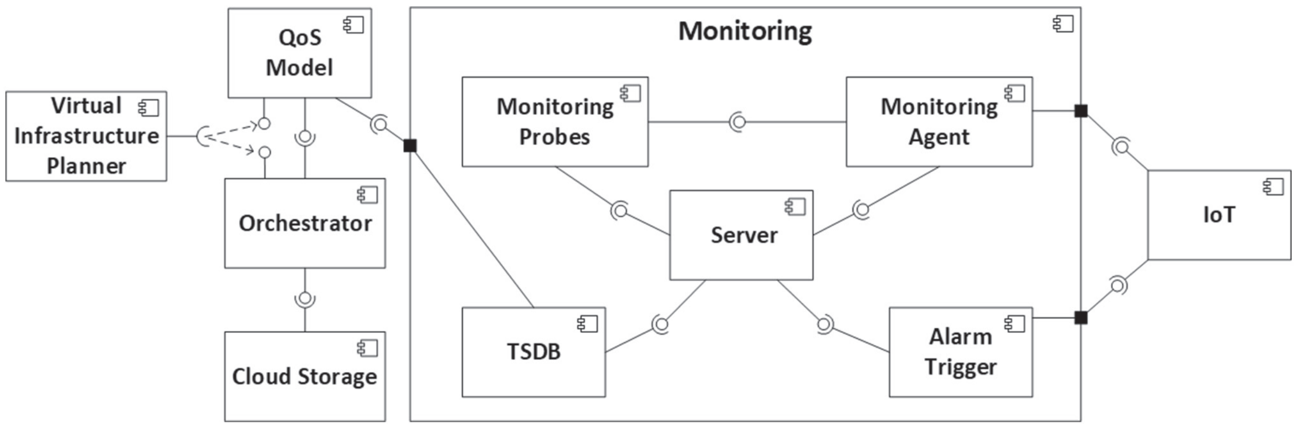

Fig. 2. Design of the edge management services.

advanced cloud computing storage solutions. Advanced new approaches, such as osmotic computing [42] have emerged. They advocate the composing applications from microservices and then executing them at the network edge. In contrast to cloud computing, which is still a centralized computing concept, new computing concepts extend the computing capacity to the network edge in a manner of decentralized computing. The fog computing focuses procession efforts within the network, between the end devices and the traditional cloud computing data centers [34]. On the other hand, edge computing pushes the processing even closer to the data sources, and it allows every device on the edge to process information [7]. Though, edge and fog are different computing concepts, they do have common goals, therefore edge and fog computing can be considered as the two different sides of the same coin.

Currently, there exist data-centric solutions [43–45] that offer near-data computing and provide strong support for IoT systems. Decentralized computing solutions deliver services closer to the users at the network edge. Their goals are to decrease delays, data transfer and processing expenses, improve scalability and so on. Furthermore, it is possible to enable the edge devices to become virtualization platforms [46]. For instance, container based virtualization can be used to achieve edge computing designs. In general, a container consists of application code, all the required libraries and, if needed, the dataset as well. A container instance therefore encapsulates both functionality and data. The implementation of an application on a container is a straightforward process, and can be used to move functionalities closer to the IoT devices that generate data.

3.4. Software engineering with modern tools and techniques

Alongside the development of virtualization technologies, software engineering approaches and tools have achieved remarkable progress in recent years as well. Modern software development and engineering practices include rapid application development, extreme programming, test-driven development, behavior-driven development and so on. Generally, such progress is facilitated through component-based software engineering. It promotes tighter cooperation between the development, business and IT operation teams. DevOps [47] is a culture, community and practice promoting continuous integration, automatic updating and scaling of applications with low failure rate and improved agility. DevOps uses a set of tools to provide for flawless software development.

The SWITCH environment is a prototypical software engineering environment consisting of three subsystems that help engineers manage application development, infrastructure provisioning and system runtime. By using the SWITCH Interactive Development Environment (SIDE) it is possible to develop IoT applications by dragging and dropping software components (e.g. Virtual Machine and container images, scripts) onto a graph canvas, to define the workflow among these software components and to assign QoS requirements to particular components that must be achieved for dependable operation of the IoT application.

3.5. Solution design for smart construction

Bearing in mind the existing set of advanced technologies for container-based virtualization and software engineering, it is possible to provide an edge computing architecture and design for smart construction applications. A set of edge computing infrastructural components necessary to support the smart applications have been identified in addition to the cloud computing resources that have to be allocated to applications dynamically. Computing resources that are usually required include processing power, memory, storage capacity as well as networking elements, such as high bandwidth and low latency communication infrastructure.

In addition to the edge and cloud computing infrastructure, the operation of the IoT applications must be facilitated with adequate middleware services. The utility of several advanced cloud computing systems (mOSAIC, SWITCH and ENTICE) was assessed against the criterion to what extent their newly developed services can be useful for building an edge computing architecture. Following this assessment, a set of particularly useful services were identified and implemented within a working environment.

Essential services (Edge Management Services) contributing to a dependable edge infrastructure and application design are schematically presented in Fig. 2 and discussed in detail in the following paragraphs.

1. The Virtual Infrastructure Planner is a component responsible for planning the necessary cloud and edge infrastructure that is required to host high QoS (time-critical) applications. In order to plan an optimal infrastructure for every smart application, it relies on a QoS Modeling component. In the proposed design, launching applications as soon as possible is not a point of interest, instead, the key requirement is to provide high QoS, stable and cost-effective performance during the operation of the IoT applications.
2. The QoS Modeling component provides technical infrastructure requirements for a particular software component implemented as a container image. The QoS model is based on benchmarking and software engineer's input. For example, an output of this tool would provide indication that an HTTPS server component requires a minimum of 1 MB memory and 80 GB storage capacity, and the maximum allowable 35 ms latency for its operation. This component also offers a functionality for the system to learn what metrics are important for achieving better and more stable QoS. QoS metrics are provided by an implemented runtime resource monitoring

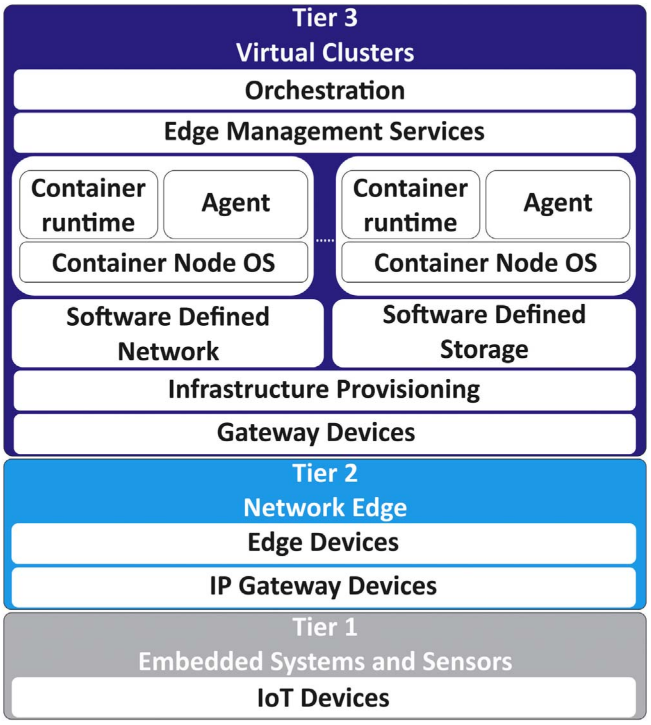

Fig. 3. Layered architecture for edge computing.

system.

3. The Orchestrator is a service for cluster management and container (that is, software services) scheduling. It can be used to address the QoS requirements for each individual smart application. Orchestration as a process automates the management of software components. It makes it possible to scale (start and stop containers) across multiple cloud and edge host systems. For example, it can be used to start a containerized HTTPS service closer to the user and thus provide for best possible file upload time. There are plenty of tools for container orchestration that can be used, such as: Docker Swarm [48], Marathon [14], Amazon ECS [49] and Kubernetes [13]. In this study the open-source Kubernetes orchestrator is used to implement a testing environment.
4. Cloud Storage can be federated storage and it can be used for storing both data and container images. When a specific functionality is requested, such as video-communications, file uploading service, sensor-data analysis service and similar, the container image can be fetched by the Orchestrator and deployed on a specific Virtual Machine running in either edge computing devices or in the data center. In such way, functionalities do not need to execute permanently. This avoids the excessive use of computing and networking resources, particularly in situations when various functionalities are not permanently needed. The ENTICE project recently delivered a set of services and a federated storage environment, which supports highly optimized, that is, fast delivery and deployment of container images. Its operation is based on Pareto optimization methods and is used in the demonstration of the present edge computing design study.
5. The Time Series Database (TSDB) is a database for storing QoS monitoring metrics. For such purposes the open-source Apache Cassandra [50] is used as it is optimized to store time series data. The monitoring of IoT based applications requires the storage and processing of large amounts of monitored QoS metrics, therefore a reliable TSDB technology is used.
6. A set of resource monitoring components (Probes, Agents, Server and Alarm Trigger) collects and processes QoS metrics [51], such as jitter and packet loss, which are collected in the runtime. These QoS metrics are used for two purposes: to build a QoS model of the software component and to facilitate an edge computing decision-making process, focused on achieving high QoS and dependable system operation. Due to the closely related work between the monitoring components, they are all implemented as one joint component. Monitoring Agents are lightweight components that address the management of metrics collection from the cloud element, like physical nodes, virtual machine instances (VMI), container instances (CI) and so on, that they are monitoring. The role of the Monitoring Agents is to register the Monitoring Probes, aggregate the measured data and distribute the data to the Monitoring Server. The Monitoring Probes are the components that address the collection of system-level and application-level metrics. The monitoring server component receives the metrics from the Monitoring Agent and forwards them to the TSDB. An Alarm Trigger, which is a rule based component, processes the incoming metrics at the server side. If a violation of the application performance level is detected, the Orchestrator is notified and it can be used to redeploy the application at another edge computing resource.
7. The IoT devices include smartphones, cameras, sensors, actuators, signalization elements and various other smart devices. They aggregate data for further processing, and in some cases convert analog to digital data. To decrease the data traffic, the collected data is first processed by the IoT devices at the edge of the network. Data is therefore preprocessed in a decentralized manner. Due to the limited amount of processing power the edge devices (e.g. router, Raspberry PI) are not capable to fully process the data. Therefore the data can be forwarded to other, more powerful edge computing devices, such as CoreOS based micro-servers in proximity of the construction site for further processing. Alternatively, if the processing of the data is not time-critical they may be forwarded to a specific public cloud provider for lower cost processing.

3.6. Tailoring the edge infrastructures

The computing infrastructure that can be used to operate IoT applications consists of two types of systems: edge computing systems (microservers, routers, Raspberry PIs and other computing capacities in proximity of the smart environment) and private and public clouds (that is, data center computing capacity). The first step in this process, that should be done by an administrator of smart applications, is to assess their edge computing needs.

Fig. 3 shows a layered architecture for edge computing, which organizes services into three tiers. Since applications can be built from reusable software components, the software components can be deployed to run on all the tiers. The first tier of software components would run in on-field IoT devices, the second tier of software components would run on edge computing nodes and the third tier of software components would run permanently on the most suitable cloud infrastructure from the virtual clusters. As an example, various IoT devices, such as video-camera equipped helmets could run on batteries and occasionally stream information to the cloud.

Time-critical software components should obviously execute in edge computing mode due to their high requirements for networking resources. Depending on the situation, the smart environment operator can decide how many edge computing devices are necessary to achieve dependable operation of the smart applications. For example, three high performance microservers deployed close to a medium size smart construction environment may be enough to facilitate high QoS and dependability. Additional parts of the applications that do not exert time-critical requirements can be deployed in a data center, which would reduce the operational costs.

The high degree of infrastructure adaptability facilitated by the previously discussed SWITCH and ENTICE services essentially means that the system can autonomously overcome the need for constant administration, when running the IoT applications. The Kubernetes-based Orchestrator can quickly start and stop containers serving particular IoT devices across the whole computing spectrum, for example, in the data center, in the edge computing devices and even in the embedded computing systems of the movable IoT devices (on which an operating system and an orchestrator are installed and running). Therefore, this is a complete system that can operate, update, extend, create, destroy and otherwise manage functionalities (software services) in real-time.

The developed Orchestrator can be effectively used to scale containers horizontally as the usage increases and therefore offers possibilities for high services availability. This, of course, means greater operational safety in close proximity of the actual smart construction environment. One can chose an availability parameter specifying how many service replicas should be simultaneously running in order to achieve dependable operation. Furthermore, the use of the Orchestrator guarantees a more stable continuous delivery of services, as a result to the rapid recovery process of the applications - applications are continuously monitored by a a set of resource monitoring components, and an Alarm Trigger issues alarms whenever QoS thresholds (e.g. latency, jitter, bandwidth) are violated.

3.7. Setting up edge computing infrastructures

Edge computing infrastructures can be composed of various gateway devices, routers, industrial computers, micro servers and micro data centers (1–5 kW energy consumption). Common to all these computing systems is that they have an Operating System. Operating Systems that can seamlessly support Docker containers are CoreOS, RancherOS and practically any Linux based system. An IP Gateway is a specific device that provides an Internet entry point to the IoT devices. Examples of IP Gateways include TA900e or Cisco-ASA, while IoT devices and sensors can include video-communications, smartphones, transmitters for temperature, air quality, humidity and so on. Obviously, a system operator would be needed to establish such an infrastructure and continuously monitor it via the Monitoring components. However, the types of the connectivity (wired, wireless) between the IoT and the software services running at the network edge has not been in the focus of the study. The designed architecture is agnostic in relation to this aspect.

The deployed IoT devices can stream audio, video and sensor data in near real time. Their data is then directly communicated and processed by containerized services running on the edge computing devices.

On-site processing of data may be necessary for time-critical situations, such as potential collisions or accidents, where an urgent response to events is needed. Moreover, edge processing allows faster further processing of data in the cloud at lower costs. Then, the edge processed data is transmitted to the network gateways, which forward the data to the most appropriate entity of the virtual data centers (clusters).

A virtual data center can consist of various Virtual Machines and physical machines on which Kubernetes is running. Services running in such a data center can include containerized databases, such as Cassandra. Generally, the virtual data center provides an infrastructure, where containerized applications can be deployed and run. Therefore, users, such as construction workers, engineers, other construction site personnel, and even robots, can use the containerized applications from various devices, such as: laptops, smartphones and tablets and at the same time monitor the data coming from sensors and react instantly, if needed.

4. Developing and deploying smart applications

Two applications, which require high QoS operation were carefully selected to illustrate the operation of the developed edge computing architecture. They are both network intensive, thus it makes a lot of sense to deploy them in an edge computing infrastructure. The first one is concerned with signalization and communications during construction and the other one is concerned with documentation of the overall building process for any future needs.

4.1. Signalization and communications

Effective communication and various signalization solutions [18–20] are essential for the success of any construction process within one or multiple construction sites. Video communication is a communication method that can be complemented with various process intensive features, such as real-time video processing to identify safety and security risks, augmented reality environments [52] improve teamwork and project collaboration, prevent delays, mistakes on the construction sites.

Construction companies, architects, site teams, engineers, surveyors and other site personnel must be in regular contact, thus videoconferencing reduces the need to meet in person to have face to face meetings. Thus, it provides face to face meetings at any time for lower costs, offering site-to-site, site-to-office and site-to-project teams communication. Apart from the meetings, the included parties in the videoconferencing can display video from the construction site if necessary to discuss and solve potential on-site issues, regardless of the geographical location. Communicating in such manner, as shown in Fig. 4, could lead to better collaboration, providing solutions faster and reducing production costs. Thus, it represents an example of an essential application.

In order to build an edge computing application open-source software can be used. Different open-source, videoconferencing applications were considered: Hubl.in [53], Talky [54], Licode [55] and Jitsi-meet [56]. Their functionalities are compared in Table 1.

Based on our analysis Jitsi-meet is an advanced WebRTC based open-source software with multiparty videoconferencing functionality for real-time communication between multiple subjects. It is a real-time multimedia streaming application based on the RTP/UDP protocols and it was selected as most suitable to be implemented as a container-based service. Its implementation in a container requires some expertise in software engineering [57]. The process consists of deploying its functionalities as services. Once the application is successfully deployed in one Docker container, it can be run on any physical device or Virtual Machine on which Docker and Kubernetes are deployed.

Fig. 4 presents the Graphical User Interface (GUI) of the WebRTC application during a demonstration of the edge computing design. A videoconference was established among five construction engineers that are discussing a construction process. All of the users connected to the videoconference session via a Web link. The video streams are managed by the containerized Jitsi Meet software in an edge computing node (Tier 2). Due to the use of the WebRTC protocol the application can run directly in a Web browser, so it is not necessary to install it on any local device (e.g. smartphone).

According to our scenario, the WebRTC application does not have a dedicated server to run constantly. Its container image is deployed and run by the Kubernetes-based Orchestrator upon users' request. The container-based virtualization and container orchestration allow creating and destroying the application instances dynamically for each individual videoconference. This helps save resources at the computing edge and run containers only when needed for particular time-critical smart operations.

4.2. Documenting the building process

Documenting is a well-known issue in many industries, and also it plays crucial part in construction. Although, this does not present a major issue for small construction projects, some modern companies try to address also this part. For larger projects, delays and data-loss are probable and documenting is considered essential. New IoT devices may help document the whole process with new formats and approaches including: RFID tags, video surveillance, sensor data and

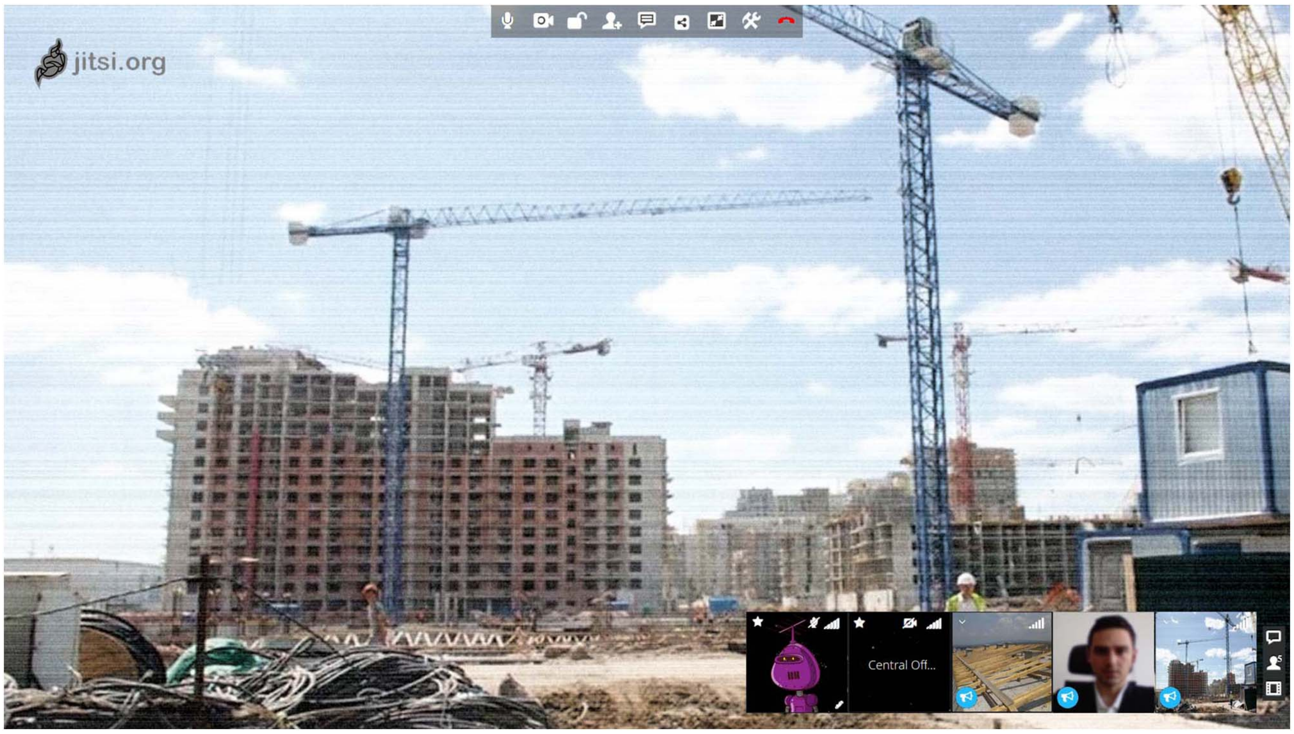

Fig. 4. Jitsi-meet based application for communication.

Table 1
Open-source WebRTC applications.

	Hubl.in	Talky	Licode	Jitsi-meet
Videoconference	+	+	+	+
Screen-sharing	–	+	+	+
Record	–	–	+	+
Chat	+	+	–	+
Encryption	+	+	–	+

many others, all of which represent a Big Data problem (volume, velocity, variety and veracity of data). In such cases, implementing cloud-based solutions could overcome the existing challenges.

Our Building Process Documentation Cloud application (BPDoC) presented in Fig. 5, is a very basic IoT application that can be used to document various on-site implementation details, such as ad-hoc solutions by using smartphones. This application has to operate in real-time to avoid any delays and it is therefore also time-critical. Generally, the goal of this application is to provide file sharing mechanism among the participants of the construction process with high QoS.

BPDoC implements a decentralized storage service. QoS for large file uploads may significantly differ according to the geographic location of the users. The fastest upload time would be achieved, if the data-receiving container runs in proximity of the smartphone. In other words, the closer the data receiving container is running, the higher the QoS. This is due to better throughput conditions, lower latency and similar.

The BDPoC application was developed as a Java Servlet. It uses Fabric8 libraries to gain access to the Kubernetes-based virtual clusters. The data-receiving container was implemented as follows. An open-source HTTPS server was deployed in a Docker container. Once the container runs, it provides a Uniform Resource Identifier (URI) of the service. This URI is forwarded to the device that is uploading the file. Therefore, the service is created, served and destroyed for each individual file upload request.

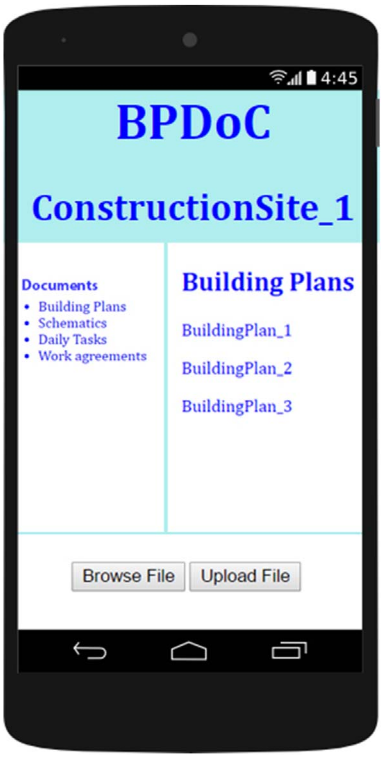

Fig. 5. BDPoC - HTTPS server based documentation.

4.3. Implementing functionalities with containers

In summary, two time-critical functionalities were implemented as software components by using Docker containers. They are as follows:

- File Upload. A container image represents an HTTPS server to which files can be uploaded.

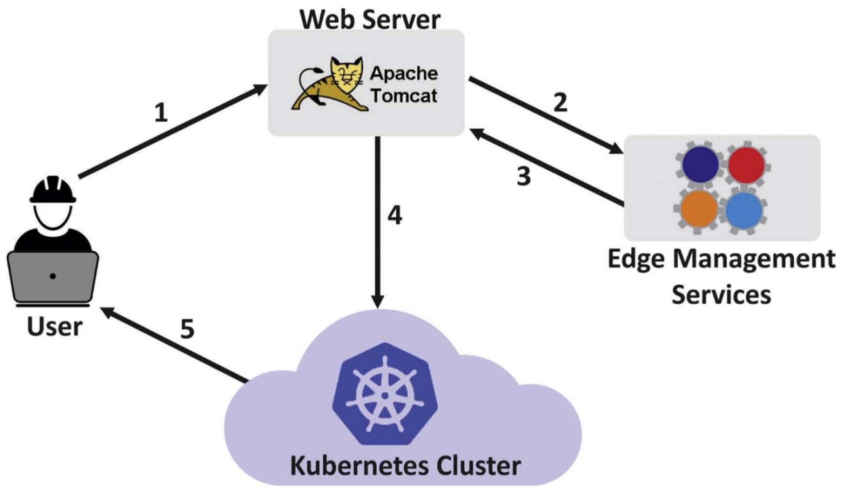

Fig. 6. Using the applications.

- Video conferencing. Developed container images are based on the Jitsi-meet open-source technology. Four separate server side components include: 1) Jitsi Videobridge, responsible for controlling audio/video media streams between participants, (2) Jicofo manages the videoconferences, (3) Prosody for the exchange of signaling messages and (4) a Web Server, which is used to serve the application via the WebRTC protocol. The application is packed onto a single container image. Four container version of the application is possible, but, it was not used in the present study.

In order to manage the created containers and operate with container instances of applications, an additional Java servlet-based Web-application that uses Fabric8 libraries to access the Kubernetes clusters was developed.

4.4. *Scenarios and testing*

The edge infrastructure, where the QoS metrics measurements were made followed the steps identified in Fig. 6: (1) A construction site engineer wants to start a video conference or upload a file, therefore he sends an HTTP request to the Web server. (2) The Web server forwards the user IP to the decision making component. (3) The decision making component evaluates the QoS requirements and decides about the placement of the software component. Then, the Orchestrator starts the container on the chosen machine. (4) The Web Server prepares a proper HTML, containing a Web form of either the file upload functionality or the videoconferencing application GUI. It consists of a created Kubernetes pod that is started by the Kubernetes-based Orchestrator service. (5) The generated HTML is sent to the client, so that the user can initiate any of the two applications at any time.

4.5. *Quality of Service metrics and measurements*

Testing procedure was designed to investigate the feasibility of the edge computing solution and the actual influence of the geographic location on the QoS of the two applications.

For the purpose of measuring the QoS metrics, it was assumed that an IoT smart construction environment is located in Ljubljana, Slovenia. Various cloud and edge computing infrastructures can be used to build a virtual data center and run containerized software components (i.e. functionalities) on the infrastructure. The Orchestrator allows the registration of both Virtual Machines and physical hardware, such as microservers and Raspbery PIs. In order to prove the concept, the testing infrastructure consisted of a set of Virtual Machines which were instantiated in different regions: Frankfurt, Germany, Singapore, Singapore, Sydney, Australia and Oregon, Canada. The Orchestrator was used to deploy, start and stop containers in all of the acquired Virtual Machines.

The IoT devices are represented by their Internet Protocol (IP) numbers. An important QoS metric that can be observed between the devices and the container on which the required functionality is deployed is the Round Trip Time (RTT), which is measured in ms. The RTT metric measures the time required for a set of TCP packets to be sent from the IoT device to the container, to confirm the packets, and then from there send them back to the IoT device. This measurement between the container and the IoT devices is facilitated by the implemented set of resource monitoring components. For the video communications (Jitsy-meet) and the BPDoC applications the following rule applies: the lower the RTT, the better the QoS. The RTT measurements are summarized in Table 2. The measurements were repeated fifty times and average values are given.

Another important QoS metric that can be observed is the jitter. It is defined as a variation in the delay of received packets. Although, the IoT device transmits packets in a continuous stream, spacing them evenly apart, the delay between packets on the receiver side can vary. This can be caused by network congestion and improper network configuration. For both applications the following rule applies: the lower the jitter, the better the QoS. In Table 3 are given the average values obtained from the jitter measurement iterations.

When designing a dependable edge computing infrastructure, which would be capable of processing large quantities of data, it is necessary to consider the data upload speed as well. It measures the speed of transferring containerized applications and data within the edge computing infrastructure. For evaluation of the results for both applications, the following rule applies: the higher the upload speed, the better the QoS. Table 4 shows the average values of the results received by measuring the upload-speed (MB/s).

In conclusion, the geographical location of the hosting services (i.e. their placement), where the containerized applications are deployed and run has an important impact on the infrastructure QoS. From the results shown in Table 2, Table 3, and Table 4, the best results were measured from the cloud infrastructures located in Frankfurt, which

Table 2
Application RTT measurements.

	RTT (ms)	
Location	Video communication	BPDoC
Frankfurt	8.186	7.145
Singapore	195.985	167.618
Sydney	314.189	349.168
Oregon	172.853	178.320

Table 3
Application Jitter measurements.

	Jitter (ms)	
Location	Video communication	BPDoC
Frankfurt	0.076	0.082
Singapore	5.102	5.953
Sydney	0.435	0.702
Oregon	1.643	1.937

Table 4
Upload-speed measurements.

	Upload-speed (MB/s)	
Location	Video communication	BPDoC
Frankfurt	8.372	8.015
Singapore	0.763	0.832
Sydney	0.867	0.893
Oregon	1.427	1.298

geographically is closest to Ljubljana where the services are provided. As a result, the closer the running containers, the better the performance.

Apart from the monitoring tools, to achieve higher system dependability it is also necessary to design the system to allow intelligent allocation and selection of resources [58]. Some applications may depend more on the processor frequency, and other applications could depend on the storage. The used set of resource monitoring components provides for the measurement of multi-level monitoring metrics and can therefore be applied to different types of software components with different resource requirements.

5. Discussion

The goal of this paper was to develop an edge computing architecture and a viable design for the development of dependable smart edge computing applications in the constructor sector. The developed design can be used to facilitate smart IoT applications, such as applications in the field of documentation, communication, safety monitoring and control, real-time sensor data monitoring and many others. Many application opportunities arise from the emerging new open-source cloud technologies. Therefore, this work presents a feasible design for smart construction environments, which is based on the latest edge and fog interoperability initiatives. Our work in this context provides technical guidance for implementation of interoperable smart construction environments. We can envisage scenarios where the edge computing approach can be used to connect multiple, remote smart construction environments to share necessary management information.

An important part of the design is its contribution towards greater dependability, particularly higher QoS. During this work, special attention was given to achieving reliability of the virtual infrastructures. Although, there exist many dependability attributes, they are closely related and commonly require high QoS. As it can be seen from the conducted experiments, the geolocation of the running software services may significantly influence the QoS. The jitter and RTT was lower and the upload speed was higher in the case of the proximal services.

Safety and security dependability attributes were also carefully studied while preparing the system design. Container virtualization arguably has the potential to improve on these two aspects. Containers start, respond on requests and can be quickly destroyed on demand. Their lifetime can be very short, thus reducing risk for an attack to be performed (compared to long-running services). Moreover, there are mechanisms to allow only authorized container-to-container traffic, so that an attacker cannot reach the specific services endpoints. In the situation when one application container is under attack, this cannot be a threat to the other running containers in the system. Moreover, the container can be rapidly destroyed, and reestablished in another location. The downtime between the destruction and restart of the container is usually small (few seconds), which contributes to higher dependability.

The reliability and security characteristics of the proposed design reduce the possibility for catastrophic smart application failure that could affect the local environment and users. For example, the implemented video communication application could be upgraded with additional functionalities, for example, to implement smart collision detection at construction sites. In this way, the safety procedures at the smart construction site could satisfy higher standards, at lower costs, and prevent human injuries, property damage or other disasters.

This paper presented two simple time-critical applications that could be branded IoT, which were used to test the dependability of the proposed design and show the benefits of edge computing infrastructures. The analysis of the literature shows huge potential for IoT applications also in other areas of construction. Some potential application areas are discussed in the following.

5.1. Supply management

Supply management of materials at construction sites, where the supplies have to be transferred between places, directly influences the flow of the construction process and its success. Real-time monitoring and control methods of the supplies are necessary for flawless construction process without delays. Thus, integrating such application in the proposed design would directly improve the construction process. Radio-frequency identification (RFID) technology implementation for supply management and material flow has found wide use in civil engineering, because it acquires real-time information and improves the dynamic control and management of supplies and materials. For that purpose, a specific use case could be developed and deployed on the proposed edge computing architecture.

RFID devices consist of a reader and a tag. The supplies at the constructions site are all equipped with RFID tags. At the entrance, every construction site has installed RFID readers. As a result, the moment the supplies enter the construction site, the RFID reader receives data from the RFID tag of the supplies and the supplies database is updated. In such a manner, the construction companies could effectively track their materials, equipment and similar belongings on multiple construction sites on different geographical locations.

Similar Web-based software solutions, such as the one proposed by Wang et al. [25] are already present for a long time. However, the software solution, if included in the current edge computing design could improve on the QoS. Moreover, the utilization of containerized visualization for databases improves the performance, reduces the infrastructure costs, improves the speed of database deployment, makes the database management easier and more portable and so on. That means that construction companies could flexibly move database containers to different environments, reduce the operational expenses and improve the QoS of the used data services.

5.2. Early disaster warning systems

Various accidents, between workers and heavy equipment may happen during the construction process. Due to the poor safety, different solutions are developed to improve the safety performance at construction sites [18,19,59]. However, to improve the safety performance, real-time monitoring devices and applications are necessary to be developed to automate this process. Therefore, integrating a time-critical, early warning system to the proposed cloud-based design, will increase the safety at the construction site, prevent disasters and save lives.

A multi-tier cloud application that represents an early warning system, has been designed by the company BEIA Consult [60]. Their system is designed to generate warnings in case of natural disasters, i.e. flood, earthquakes and similar natural disasters to save lives. This system, could be also modified and installed to control and manage data gathered from construction site sensors, because it implements light-weight virtualization and it is adapted to run containerized applications. The early warning system collects data from sensors in real-time, it processes the gathered data with various tools to detect and respond to events rapidly and provides data to the warning services. Moreover, the application should satisfy higher dependability standards for its services and enhanced scalability to the number of used sensors.

6. Conclusions

An important goal of this work was to investigate the utility of advanced edge computing concepts in the smart construction domain. High QoS is required in order to achieve dependable operation of smart applications. Edge computing infrastructures may include microservers, routers and other devices on which various orchestration services can be deployed. This can allow the execution of various container-based applications in proximity of the smart construction site. The proposed edge computing architecture and design is multi-tier and therefore flexible, so that in case of greater demand for services more computing resources can be obtained from regular data centers.

In comparison to other existing solutions for automation in construction discussed in Section 2, the novelty of this work is the new edge computing architecture and design of a dependable IoT system for construction. The developed design is implemented and two IoT applications with high QoS requirements are used to demonstrate the approach. This new design can seamlessly host many different smart applications, for different construction project needs and sizes. In addition, software services, including analytic services, such as data mining, machine learning, simulations and many other, can be implemented with containers and used in the present edge computing architecture effectively. The testing results strongly support the edge computing concept. Therefore, using cloud environments close to the construction sites (including microservers) could provide for flawless construction site management and communication, avoiding additional expenses and improving dependability.

Our plans are to extend the functionalities of the system by addressing security related challenges. We plan to integrate the present system with blockchain technology. This will provide greater degree of security in the operation of the overall system, as well as possibilities for seamless integration with services from the outside world.

Acknowledgments

The research and development reported in this chapter have received funding from the European Union's Horizon 2020 Research and Innovation Programme under grant agreements no. 643963 (SWITCH project: Software Workbench for Interactive, Time Critical and Highly self-adaptive cloud applications) and no. 644179 (ENTICE project: dEcentralized repositories for traNsparent and efficienT vIrtual maChine opErations). Funding was also received from the Slovenian Research Agency under grant agreement no. BI-RU/16-18-043 (Internet of Things and Cloud computing as support for the development of new smart approaches in the construction sector).

References

[1] Gartner,Inc, http://www.gartner.com/newsroom/id/3598917, [Online; accessed 03-May-2017].
[2] X. Jia, Q. Feng, T. Fan, Q. Lei, RFID technology and its applications in Internet of Things (IoT), Consumer Electronics, Communications and Networks (CECNet), 2012 2nd International Conference on, IEEE, 2012, pp. 1282–1285.
[3] M. Yu, D. Zhang, Y. Cheng, M. Wang, An RFID electronic tag based automatic vehicle identification system for traffic IOT applications, Control and Decision Conference (CCDC), 2011 Chinese, IEEE, 2011, pp. 4192–4197.
[4] B. Dave, S. Kubler, K. Främling, L. Koskela, Opportunities for enhanced lean construction management using Internet of Things standards, Autom. Constr. 61 (2016) 86–97.
[5] Y. Niu, W. Lu, K. Chen, G.G. Huang, C. Anumba, Smart construction objects, J. Comput. Civ. Eng. 30 (4) (2015) 04015070.
[6] P. Mell, T. Grance, The NIST Definition of Cloud Computing: Recommendations of the National Institute of Standards and Technology (2011), NIST Spec. Publ. 800145 (2012).
[7] W. Shi, J. Cao, Q. Zhang, Y. Li, L. Xu, Edge computing: Vision and challenges, IEEE Internet Things J. 3 (5) (2016) 637–646.
[8] Cloud Native Computing Foundation, https://www.cncf.io/about/, [Online; accessed 03-May-2017].
[9] OpenFog Consortium, https://www.openfogconsortium.org/about-us/, [Online; accessed 01-September-2017].
[10] CoreOS, https://coreos.com/docs, [Online; accessed 03-May-2017].
[11] RancherOS, http://rancher.com/rancher-os/, [Online; accessed 03-May-2017].
[12] Docker, https://docs.docker.com/, [Online; accessed 03-May-2017].
[13] Kubernetes, http://kubernetes.io/docs/, [Online; accessed 03-May-2017].
[14] Marathon, https://mesosphere.github.io/marathon/docs/, [Online; accessed 03-May-2017].
[15] Fabric8, https://fabric8.io/docs/index.html, [Online; accessed 03-May-2017].
[16] X. Yuan, C.J. Anumba, M.K. Parfitt, Cyber-physical systems for temporary structure monitoring, Autom. Constr. 66 (2016) 1–14.
[17] W. Ren, Z. Wu, Real-time anticollision system for mobile cranes during lift operations, J. Comput. Civ. Eng. 29 (6) (2014) 04014100.
[18] S. Chae, Development of warning system for preventing collision accident on construction site, Proceedings of the 26th International Symposium on Automation and Robotics in Construction (ISARC 2009), 2009.
[19] S. Chae, T. Yoshida, Application of RFID technology to prevention of collision accident with heavy equipment, Autom. Constr. 19 (3) (2010) 368–374.
[20] N. Jegen-Perrin, A. Lux, P. Wild, J. Marsot, Preventing plant-pedestrian collisions: camera & screen systems and visibility from the driving position, Int. J. Ind. Ergon. 53 (2016) 284–290.
[21] Y. Li, X. Zhang, Web-based construction waste estimation system for building construction projects, Autom. Constr. 35 (2013) 142–156.
[22] W.R. Jones, M.J. Spence, A.W. Bowman, L. Evers, D.A. Molinari, A software tool for the spatiotemporal analysis and reporting of groundwater monitoring data, Environ. Model. Softw. 55 (2014) 242–249.
[23] N.R. Swain, K. Latu, S.D. Christensen, N.L. Jones, E.J. Nelson, D.P. Ames, G.P. Williams, A review of open source software solutions for developing water resources web applications, Environ. Model. Softw. 67 (2015) 108–117.
[24] K. Prasad, E.K. Zavadskas, S. Chakraborty, A software prototype for material handling equipment selection for construction sites, Autom. Constr. 57 (2015) 120–131.
[25] L.-C. Wang, Y.-C. Lin, P.H. Lin, Dynamic mobile RFID-based supply chain control and management system in construction, Adv. Eng. Inform. 21 (4) (2007) 377–390.
[26] C. Kim, T. Park, H. Lim, H. Kim, On-site construction management using mobile computing technology, Autom. Constr. 35 (2013) 415–423.
[27] Q. Shi, X. Ding, J. Zuo, G. Zillante, Mobile Internet based construction supply chain management: a critical review, Autom. Constr. 72 (2016) 143–154.
[28] Q.T. Le, D.Y. Lee, C.S. Park, A social network system for sharing construction safety and health knowledge, Autom. Constr. 46 (2014) 30–37.
[29] H.-M. Chen, K.-C. Chang, T.-H. Lin, A cloud-based system framework for performing online viewing, storage, and analysis on big data of massive BIMs, Autom. Constr. 71 (2016) 34–48.
[30] T.H. Beach, O.F. Rana, Y. Rezgui, M. Parashar, Cloud computing for the architecture, engineering & construction sector: requirements, prototype & experience, J. Cloud Comput. Advances Syst. Appl. 2 (1) (2013) 8.
[31] A. Braun, S. Tuttas, A. Borrmann, U. Stilla, A concept for automated construction progress monitoring using BIM-based geometric constraints and photogrammetric point clouds, Electron. J. Inf. Technol. Constr. (ITcon) 20 (5) (2015) 68–79.
[32] M. Golparvar-Fard, F. Pe na-Mora, S. Savarese, Automated progress monitoring using unordered daily construction photographs and IFC-based building information models, J. Comput. Civ. Eng. 29 (1) (2012) 04014025.
[33] L. Bai, Y. Sun, X. Guo, Applied research on tower crane safety supervising system based on Internet of Things, Business, Economics, Financial Sciences, and Management, Springer, 2012, pp. 549–556.
[34] F. Bonomi, R. Milito, J. Zhu, S. Addepalli, Fog computing and its role in the internet of things, Proceedings of the first edition of the MCC workshop on Mobile cloud computing, ACM, 2012, pp. 13–16.
[35] R. Roman, J. Lopez, M. Mambo, Mobile edge computing, fog et al.: a survey and analysis of security threats and challenges, Futur. Gener. Comput. Syst. (2016).
[36] S. Yi, C. Li, Q. Li, A survey of fog computing: concepts, applications and issues, Proceedings of the 2015 Workshop on Mobile Big Data, ACM, 2015, pp. 37–42.
[37] V. Stankovski, D. Petcu, Developing a model driven approach for engineering applications based on mOSAIC, Clust. Comput. 17 (1) (2014) 101–110.
[38] Z. Zhao, A. Taal, A. Jones, I. Taylor, V. Stankovski, I.G. Vega, F.J. Hidalgo, G. Suciu, A. Ulisses, P. Ferreira, et al., A Software Workbench for Interactive, Time Critical and Highly self-adaptive cloud applications (SWITCH), Cluster, Cloud and Grid Computing (CCGrid), 2015 15th IEEE/ACM International Symposium on, IEEE, 2015, pp. 1181–1184.
[39] R. Prodan, T. Fahringer, D. Kimovski, G. Kecskemeti, A.C. Marosi, V. Stankovski, J. Becedas, J.J. Ramos, C. Sheridan, D. Whigham, et al., Use cases towards a decentralized repository for transparent and efficient virtual machine operations, Parallel, Distributed and Network-based Processing (PDP), 2017 25th Euromicro

International Conference on, IEEE, 2017, pp. 478–485.
[40] A. Avizienis, J.-C. Laprie, B. Randell, C. Landwehr, Basic concepts and taxonomy of dependable and secure computing, IEEE Trans. Dependable Secure Comput. 1 (1) (2004) 11–33.
[41] I. Foster, Y. Zhao, I. Raicu, S. Lu, Cloud computing and grid computing 360-degree compared, Grid Computing Environments Workshop, 2008. GCE'08, IEEE, 2008, pp. 1–10.
[42] M. Villari, M. Fazio, S. Dustdar, O. Rana, R. Ranjan, Osmotic computing: a new paradigm for edge/cloud integration, IEEE Cloud Comput. 3 (6) (2016) 76–83.
[43] J. Jin, J. Gubbi, S. Marusic, M. Palaniswami, An information framework for creating a smart city through internet of things, IEEE Internet Things J. 1 (2) (2014) 112–121.
[44] F. Khodadadi, R.N. Calheiros, R. Buyya, A data-centric framework for development and deployment of Internet of Things applications in clouds, Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2015 IEEE Tenth International Conference on, IEEE, 2015, pp. 1–6.
[45] S.K. Datta, C. Bonnet, R.P.F. Da Costa, J. Härri, DataTweet: An architecture enabling data-centric IoT services, Region 10 Symposium (TENSYMP), 2016 IEEE, IEEE, 2016, pp. 343–348.
[46] L.M. Vaquero, L. Rodero-Merino, Finding your way in the fog: towards a comprehensive definition of fog computing, ACM SIGCOMM Comput. Commun. Rev. 44 (5) (2014) 27–32.
[47] DevOps, https://devops.com/about/, [Online; accessed 03-May-2017].
[48] Docker Swarm, https://docs.docker.com/engine/swarm/, [Online; accessed 03-May-2017].
[49] Amazon EC2, https://aws.amazon.com/ecs/, [Online; accessed 03-May-2017].
[50] Apache Cassandra, http://cassandra.apache.org/doc/latest/, [Online; accessed 03-May-2017].
[51] S. Taherizadeh, A.C. Jones, I. Taylor, Z. Zhao, P. Martin, V. Stankovski, Runtime network-level monitoring framework in the adaptation of distributed time-critical Cloud applications, Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA), The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2016, p. 78.
[52] S. Meža, Ž. Turk, M. Dolenc, Component based engineering of a mobile BIM-based augmented reality system, Autom. Constr. 42 (2014) 1–12.
[53] Hubl.in, https://hubl.in/, [Online; accessed 03-May-2017].
[54] Talky, https://about.talky.io/, [Online; accessed 03-May-2017].
[55] Licode, http://licode.readthedocs.io/en/master/, [Online; accessed 03-May-2017].
[56] Jitsy-meet, https://jitsi.org/Documentation/DeveloperDocumentation, [Online; accessed 03-May-2017].
[57] J. Trnkoczy, U. Paščinski, S. Gec, V. Stankovski, SWITCH-ing from multi-tenant to event-driven videoconferencing services, 1st Workshop on Autonomic Management of Large Scale Container-based Systems co-located with the 2017 IEEE International Conference on Cloud and Autonomic Computing (ICCAC), IEEE, 2017.
[58] V. Stankovski, J. Trnkoczy, S. Taherizadeh, M. Cigale, Implementing time-critical functionalities with a distributed adaptive container architecture, Proceedings of the 18th International Conference on Information Integration and Web-based Applications and Services, ACM, 2016, pp. 453–457.
[59] S. Hwang, Ultra-wide band technology experiments for real-time prevention of tower crane collisions, Autom. Constr. 22 (2012) 545–553.
[60] BEIA Consult International, http://www.beiaro.eu/category/projects-list/, [Online; accessed 03-May-2017].