Precision Agriculture (2019) 20:926–958
https://doi.org/10.1007/s11119-018-09624-8

Architecture design approach for IoT-based farm management information systems

Ö. Köksal[1] · B. Tekinerdogan[1]

Published online: 11 December 2018

Abstract
Smart farming adopts advanced technology and the corresponding principles to increase the amount of production and economic returns, often also with the goal to reduce the impact on the environment. One of the key elements of smart farming is the farm management information systems (FMISs) that supports the automation of data acquisition and processing, monitoring, planning, decision making, documenting, and managing the farm operations. An increased number of FMISs now adopt internet of things (IoT) technology to further optimize the targeted business goals. Obviously IoT systems in agriculture typically have different functional and quality requirements such as choice of communication protocols, the data processing capacity, the security level, safety level, and time performance. For developing an IoT-based FMIS, it is important to design the proper architecture that meets the corresponding requirements. To guide the architect in designing the IoT based farm management information system that meets the business objectives a systematic approach is provided. To this end a design-driven research approach is adopted in which feature-driven domain analysis is used to model the various smart farming requirements. Further, based on a FMIS and IoT reference architectures the steps and the modeling approaches for designing IoT-based FMIS architectures are described. The approach is illustrated using two case studies on smart farming in Turkey, one for smart wheat production in Konya, and the other for smart green houses in Antalya.

Keywords Smart farming · Farm management information system · Internet of things · Architecture design

Introduction

Smart farming represents the application of modern information and communication technologies (ICT) into agriculture to increase the amount of production and economic returns, often also with the goal to reduce the impact on the environment (Rains and Thomas 2009). Similar

✉ Ö. Köksal
koksal@aselsan.com.tr

B. Tekinerdogan
bedir.tekinerdogan@wur.nl

[1] Wageningen University & Research, Hollandseweg 1, 6706 KN Wageningen, The Netherlands

terms are used for the same purpose such as precision agriculture, site-specific farming, site-specific crop management, prescription farming, and satellite farming (Adamchuk et al. 2004; Zhang et al. 2002). Smart farming builds on advanced technology such as cloud computing, remote sensing, data-driven farming, big data analytics and internet of things (IoT). Several important benefits of smart farming have been provided in the literature including optimizing production efficiency, optimizing quality of the crop, minimizing environmental impact, minimizing risk, conservation of resources, reducing cost, increasing profit, and better management decisions (Sørensen et al. 2010, 2011; Rains and Thomas 2009; Zhang et al. 2002).

One of the key elements of the smart farming is the farm management information system (FMIS). Although initially FMISs started as simple record keeping systems, modern FMISs are sophisticated systems with advanced modules supporting a comprehensive set of farming operations (Fountas et al. 2015). With the introduction of IoT, FMIS and smart farming in general have gained a new momentum. The IoT helps in smart and automated information gathering and merging. It helps as well as in monitoring sensor data coming from different machines, animals, plants, other farms and greenhouses and other systems such as unmanned air and land vehicles. In this way, the decision making and planning in the agricultural domain can be further supported which can lead to even more effective and efficient farming. With the help of the IoT, farming practices such as yield monitoring, cultivar selection, pest management, irrigation, etc. can be applied more precisely. Crop yield can be monitored and precise crop maps which show high and low production areas can be obtained readily (Rains and Thomas 2009).

For developing an IoT-based FMIS it is important to design the proper IoT architecture which represents the overall gross level structure of the system. IoT based farm management information systems typically have different functional requirements such as the type of crop, the type of sensors, communication protocols, and the data processing capacity. Besides of functional requirements also quality requirements such as security level, safety level, time performance, and overall cost of development and operation are also different for different applications.

The different requirements typically require changing the IoT architecture. In the literature, several reference architectures for FMISs and IoT have been proposed that can be reused to derive the IoT application architecture. Deriving the proper architecture however is far from trivial and this can impede the success of the IoT system. The objective of this study is to contribute to the current state-of-the-art of FMISs by enhancing the current architecture design approaches for IoT-based FMIS. Hereby, the study aims to provide an architecture design method for designing IoT-based FMISs. The approach presented adopts a feature-driven domain analysis approach to model the different smart farming requirements. Further, based on FMIS and IoT reference architectures the steps and the modeling approaches for designing the IoT-based FMIS architecture are described. The approach is illustrated using two case studies on smart farming in Turkey, one for developing IoT-based FMIS for smart wheat production in Konya, and the other for smart green houses in Antalya.

Background and related work

Internet of things (IoT)

Until recent time, the internet was primarily used for interconnecting computers any time and any place but this required human interaction and monitoring. The IoT is a new

paradigm that adds a dimension to the current information and communications technologies (ICTs), whereby the dimension "Anything communication" is added to the communication capabilities. The IoT enables anytime, anyplace connectivity for anything, by linking the objects of the real world with the virtual world. In the IoT world physical things and virtual things, all interact with each other in the same space and time.

Since the IoT is the result of technological progress in many fields such as wireless sensor networks, machine-to-machine communication, mobile computing, ubiquitous computing, and embedded systems, the term "IoT" might have different meanings. Many definitions of IoT can be found in the literature, but the IoT is defined by the International Telecommunication Union (ITU) as "the network of physical objects or 'things' embedded with electronics, software, sensors, and network connectivity, which enables these objects to collect and exchange data". Here "thing" is defined as: an object of the physical world (physical things) or the information world (virtual things), which is capable of being identified and integrated into communication networks (ITU 2005). McEwen and Cassimally (2014) formulate the IoT with a simple equation as: "Physical Object + Controller, Sensor, and Actuators + Internet = IoT".

The IoT is the result of technological progress in many parallel and often overlapping fields, including those of embedded systems, ubiquitous and pervasive computing, mobile telephony, telemetry and machine-to-machine communication, wireless sensor networks, mobile computing, and computer networking. What is important is that IoT adds a dimension to the current ICTs, which already provide "any *time*" and "any *place*" communication.

Various reference architectures have been provided for the IoT. In general, IoT architecture is represented as a layered architecture. In this case a "layer" simply represents a grouping of modules that offers a cohesive set of services. Based on the literature (Al-Fuqaha et al. 2015; Pandya and Champaneria 2015; Gazis et al. 2015; Palattella et al. 2013; Sheng et al. 2013) the reference architecture is shown in Fig. 1.

The reference architecture includes the following layers: device, network, session, application, business, management and security. The device layer consists of sensors and physical devices. This layer identifies and collects data and specific information generated by sensors and physical devices. The data gathered is passed to the network layer. In essence the device layer bridges the gap between the physical world and the digital world. The network layer provides functionality for networking connectivity and transport capabilities. This layer is also called *transport layer*. This layer securely transmits data gathered from sensors to the session layer. The transmission medium can be wired or wireless. The session layer is responsible for service management and consists of functionality for setting up and taking down of the association between the IoT connection points. Several session

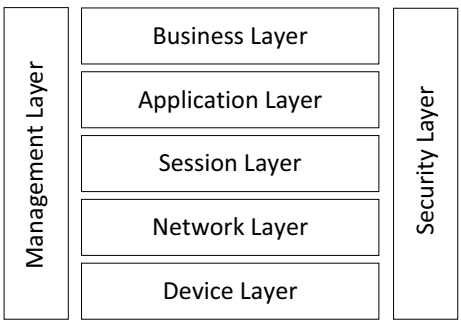

Fig. 1 IoT reference architecture

layer standards and protocols are offered by different organizations. Although most of these standards and protocols use the transmission control protocol (TCP) or the user datagram protocol (UDP) for transport, they have different architectures and characteristics for various purposes. The application layer contains the IoT services and manages the system using the data from the session layer. The implemented IoT application can be, for example, smart farming, smart city, and smart home. The business layer defines business logic and workflows. This layer is responsible for the management of all IoT systems, services and applications within the domain. The business models are defined in this layer based on the data gathered from the application layer. The data is analyzed to build the required business models and define the strategies. The security layer is a side-car layer relating to the other five layers and provides the security functionality. Similarly, the management layer is a side-car layer supporting capabilities such as device management, local network topology management, and traffic and congestion management (Köksal and Tekinerdogan 2017; Khan et al. 2012).

Architecture design

Software architecture for a program or computing system consists of the structure or structures of that system, which comprise elements, the externally visible properties of those elements, and the relationships among them (Venters et al. 2018; Tekinerdogan 2014; Bass et al. 2012; Clements et al. 2010). Software architecture forms one of the key artifacts in the entire software development life cycle since it embodies the earliest design decisions and includes the gross-level components that directly impact the subsequent analysis, design and implementation (Apel et al. 2013). It is generally accepted that software architecture design plays a fundamental role in coping with the inherent difficulties of the development of large-scale and complex software. Research on architecture design in the last two decades has resulted in different useful techniques and approaches.

Architectural drivers define the concerns of the stakeholders. A stakeholder is defined as an individual, team, or organization with interests in, or concerns relative to, a system. Each of the stakeholders' concerns impacts the early design decisions that the architect makes. A common practice is to model different "architectural views" for describing the design according to the stakeholders' concerns (Tekinerdogan 2014; Demirli and Tekinerdogan 2011; Clements et al. 2010). An architectural view is a representation of a set of system elements and relations associated with them to support a particular concern (Clements et al. 2010). Having multiple views helps to separate the concerns and as such support the modeling, understanding, communication and analysis of the software architecture for different stakeholders. Architectural views conform to viewpoints that represent the conventions for constructing and using such a representation. An *architectural framework* organizes and structures the proposed viewpoints.

A recent software architecture framework approach is the so-called Views and Beyond (V&B) approach (Clements et al. 2010). The approach distinguishes three different categories of viewpoints or styles including module, component-and-connector, and allocation styles:

- "Module" view category that is used for documenting a system's principal units of implementation.
- "Component and Connector" category that is used for documenting the system's units of execution.

- "Deployment View" category that is used to document the relationships between a system's software and its development and execution environments.

A software architecture that addresses the concerns of specific stakeholders is here referred to as "concrete architecture". A concrete architecture defines the boundaries and constraints for the implementation and is used to analyze risks, balance trade-offs, plan the implementation project and allocate tasks (Tekinerdogan 2014). Concrete architectures can be viewed as specific implementations of "reference architectures", which are generic designs. In turn, a reference architecture is derived from the knowledge and experiences accumulated in designing concrete architectures in the past (Cloutier et al. 2010; Angelov et al. 2012). The concrete architectures differ from one case to the next depending on the requirements of the stakeholders involved. Reference architectures can be used descriptively to "capture the essence of existing architectures" or prescriptively to guide the development of new ones (Cloutier et al. 2010).

Related work

Several studies discuss the adoption of internet technologies to support FMIS (Kruize et al. 2016; Fountas et al. 2015; Kaloxylos et al. 2012; Steinberger et al. 2009; Nikkilä et al. 2010; Sørensen et al. 2010; Wolfert et al. 2010; Seelan et al. 2003; Murakami et al. 2007). These studies have focused on different issues including the adoption of service-oriented architectures for FMIS (Murakami et al. 2007; Steinberger et al. 2009; Wolfert et al. 2010), the development of data exchange standards for supporting interoperability over the internet (Schmitz et al. 2009), and the adoption and implementation of geographic information systems (GIS) (Seelan et al. 2003). The main focus of these studies is integration and operation of an FMIS over the internet. The integration and adoption of IoT in the FMIS is not explicitly considered.

In (Murakami et al. 2007), a distributed service-oriented reference architecture is proposed for the development of information systems for precision agriculture. This web-based approach is focused on communication between software on a service bus. In (Schmitz et al. 2009), the so-called *AgroXML* is proposed as a standardized language based on extensible markup language (XML) to be used for data exchange in FMIS. In Nikkilä et al. (2010), a web-based approach is defined to implement connectivity requirements arising from the internet and the management of GIS data. In (Sørensen et al. 2010), a new model for FMIS is proposed to provide better information handling focusing on internal data connection, external information collection, plan generation, and report generation in FMIS. In Kaloxylos et al.(2012), an architecture is proposed to provide support and integration of different stakeholders and services, and interworking with the external services.

There are some studies in the literature related to web-based architectures (Steinberger et al. 2009; Chaudhary et al. 2004). These studies present architectures to enhance the effectiveness of web-based decision support system on which data can be requested for further use via a web portal and a web service interface.

Instead of full FMIS most architecture academic research on FMIS is restricted to individual component of an FMIS such as predicting crop yield, implementing a special sensor, and the usability of an FMIS. There are only few studies that explicitly discuss FMIS architectures in a comprehensive manner (Nikkilä et al. 2010). For example, in (Linseisen 2001), FMIS architecture is discussed by focusing on an information system gathering and storing high accuracy GPS data. In Beck (2001), an architecture, based on implementing

object databases such as common object request broker architecture (CORBA) middleware and Java languages is proposed to provide easier development, maintenance, and easier integration of information systems.

Advances in the functionality of academic and commercial FMIS are presented in (Fountas et al. 2015). This study investigates commercial and academic FMIS packages and performs a cluster analysis on them. The authors indicate that commercial packages tend to target daily farm office tasks such as budgeting, finance, recordkeeping, machinery management, and documentation. On the other hand, academic FMISs deal with compliance to standards, automated data capture, and interoperability issues.

There are also studies that discuss traditional on-site FMIS software. However, these studies mainly focus on the improvement of information integration of traditional FMIS and do not take IoT technologies into account. For example, Verdouw et al. (2016), propose an architecture to improve the standardization and integration of data, application, and process. A service-oriented architecture (SOA) based solution is proposed to improve the information integration implementing business process management (BMP).

Related to IoT is the research on wireless sensor networks which is reviewed in Jawad et al. (2017) and Aqeel-Ur-Rehman et al. (2014). These studies primarily focused on comparing sensors and communication technologies such as ZigBee, Bluetooth, Wifi, Sigfox, Wibree, long range radio and GPRS. Although these protocols might increase the number of possibilities to communicate data in IoT, these studies do not directly offer design solutions for FMIS.

This paper has focused on applying IoT for FMIS in particular. However, IoT has also been applied in different application domains. The application of IoT in agriculture has been reviewed in Verdouw et al. (2016). This review showed that the IoT concept captured the attention of the scientific community in 2010 and since then number of studies has continuously increased. In total 168 papers and books were reviewed in the paper. The identified top topics of these studies include food supply chains, arable farming, general agriculture, greenhouse horticulture, and livestock farming, and open-air horticulture including orchards. On the other hand, it is stated that IoT applications mostly focus on basic functionalities, including tracking, tracing, monitoring, and event management. It is concluded that although IoT is receiving an increasing level of attention, it is still in its infancy in the agriculture and food domain which is suffering from lack seamless integration and advanced solutions.

Case studies and problem statement

In this section, the problem statement is presented that is illustrated using two case studies of smart farming in Turkey. The case studies have been selected based on their relevance and their difference with respect to the functional and quality requirements. The case studies include the development of IoT-based FMIS for wheat production and tomato production in Turkey. In the following, first the details of each case study are presented, and subsequently the problem statement is described.

Case study: wheat production

Turkey has 23.9 million hectares of cultivated farms. Grain production occupies 49% of this area. Wheat production constitutes 67% of the total grain production (Turkish Land

Crop Office 2017). Turkey's wheat production is about 20 million tons yearly (Turkish Ministry of Agriculture 2018). As such, wheat production is one of the most important agriculture businesses in Turkey.

One of the key regions of wheat production in Turkey is the region of Konya which is far from costal area and located on a major plain near the middle of Turkey. Konya has a terrestrial climate, and big arable farms. Konya is the region in Turkey with the largest wheat production. It produces 3 million tons of wheat yearly.

Case study: tomato production in greenhouses

The second case study is of tomato production in greenhouses in Antalya. Tomato production in the world is 170 million tons yearly and almost 12 million tons of this production is produced in Turkey. Turkey exports tomatoes and tomato products to many countries. The total export is about 600 000 tons. Tomato is produced both in open fields and in greenhouses. About 51% of greenhouse production in Turkey is tomato.

Antalya is located in the south of Turkey just north of the Mediterranean coast. The typical Mediterranean climate of the region is suitable for vegetable and fruit production. Currently, specialty greenhouse farming is very common in Antalya. Some 80% of glass greenhouses and 50% of plastic greenhouses of Turkey are in Antalya.

Problem description

Generally, wheat and tomato are produced with traditional farming practices in Konya and Antalya regions. A general observation from governmental reports is that a small part of the farmers in these regions use traditional FMIS (Turkish Ministry of Agriculture 2018). Even with the use of FMISs several problems in the agricultural sector could still be identified.

Inefficient crop production

To meet the growing population in Turkey it is important to increase the crop production, which requires a more effective and efficient crop production. According to the Turkish Statistical Institute (TSI), Turkey is one of the top 10 wheat producers in the world. But this production is not sufficient for Turkey's growing internal demand. In order to compensate the need, Turkey imports more than 4 million tons of wheat each year. From the efficiency point (tons/hectare) of view improvements are required and possible. Turkey's average wheat yield is about 2.6 tons/ha.

Inefficient usage of soil

Turkey has 2.2 million farmers, and 23.9 million hectares cultivated farms but 17% of arable farms are fallow (Turkish Ministry of Agriculture 2018). One of the reasons for this is due to lack of insight and support regarding decision making on crop production, soil fertilization, and pesticide management.

Increase in cost of farming inputs

In the last years the cost of fertilizers, fuel and pesticides have dramatically increased, but the usage of these inputs has not been effectively monitored. This has adversely affected farming and greenhouse production profit. As a result of this, the number of farmers in these domains have decreased and the overall production has declined. To solve these problems, better monitoring and management of inputs is required to decrease the overall costs.

These problems can, to some extent, be tackled by focusing on improved business and logistics processes, by applying total quality and smart farming principles, and appropriate sensing and effector technologies. Yet, these solutions remain limited compared to the adoption of IoT that provides further optimization by enabling the integration of various technologies such as (wireless) sensor networks, mobile computing, cloud network, data analytics, and decision support systems.

To cope with these problems IoT can be considered a feasible solution (Dlodlo and Kalezhi 2015; Ma et al. 2011). IoT enables the use of sensors to measure the required parameters (e.g. soil quality), support the decision-making process using services such as data analytics, and use actuators to execute the proper action at the right time and right place. This is for example the case for the wheat and tomato production that has been described in the previous sections. With the introduction of IoT several benefits are envisioned. Firstly, determining the variability in yield potential might allow optimizing production at each site. With the help of smart farming practices such as site-specific soil nutrition management, quality of the soil can be improved. Also, pest management allows mapping pest populations and obtaining site-specific application maps reducing pesticide usage and minimizing environmental impact. Managing farming practices and obtaining profit maps can help reducing the risk in agriculture. Better irrigation, fertilization practices, and pest management strategies save resources. Crop production problems can be solved more precisely and in less time with smart farming. Further, long term data can be collected and analyzed, leading to better strategic management decisions. Saving input materials and resources enables reducing labor requirements and cost. Finally, reducing cost and improving quality will increase profits obtained. So far with the existing FMIS these goals could not be fully realized or are only achieved to a limited extent.

Although IoT promises to be very worthwhile, it is not easy to develop an IoT-based FMIS. In the literature, various different reference architectures have been proposed for both IoT and FMIS. Recently, the two concepts are further integrated leading to an IoT based FMIS architecture. Unfortunately, deriving a concrete application architecture for the specific farming situation is far from trivial. This is because the existing architectures are usually represented as reference architectures that are too abstract and do not consider further details that are required to derive the application architecture. To derive the concrete architecture for a particular context, the different features of FMIS and IoT should be selected. This includes for example the different management functionality, the security protocols, the device communication protocols, and the cloud services. For each of these many different selections can be made and the combination of these leads to a broad design space.

Obviously, given a description for the smart farming system many different architecture alternatives can be identified. Since the architecture has a direct systemic impact on the overall IoT based smart farming it is important to derive the proper architecture to meet the overall smart farming requirements of the stakeholders. For guiding the architect in deriving the customized concrete architecture a systematic approach is necessary.

FMIS development method

In Fig. 2, the proposed development approach for deriving an IoT-based FMIS application architecture is shown. The approach consists of two basic activities including *Domain Engineering* and *FMIS Development*. In essence, the approach is based on the product line engineering process as described in the literature (Tüzün et al. 2015; Capilla et al. 2014; Apel et al. 2013).

The *Domain Engineering* activity focuses on developing and preparing the development artefacts (e.g. design and code) for the FMIS. The first step includes the development of an IoT FMIS family feature model that defines the common and variant features of the different FMISs. The subsequent step focuses on developing the reference architecture for IoT based FMIS. The final step in the domain engineering activity aims to develop the reusable components that will be necessary to develop the FMIS based on the reference architecture. The following sections elaborate on the development of the family feature model and then present the corresponding reference architecture.

The *FMIS development* activity focuses on a specific IoT-based FMIS. The FMIS will be developed based on reuse of the artefacts in the domain engineering activity. The first step in the application engineering includes the selection of the features of the application. Further, the features will include both features for the IoT and the FMIS. These will be usually different for different FMISs in different contexts. Based on the selected features the specific FMIS application architecture will be developed starting from the reference architecture of the domain engineering activity. The final step includes the implementation of the FMIS. In this final step, the components developed earlier in the domain engineering activity will be reused. Very often a simulation system can be developed for FMIS to validate the system before deciding on the large-scale investment.

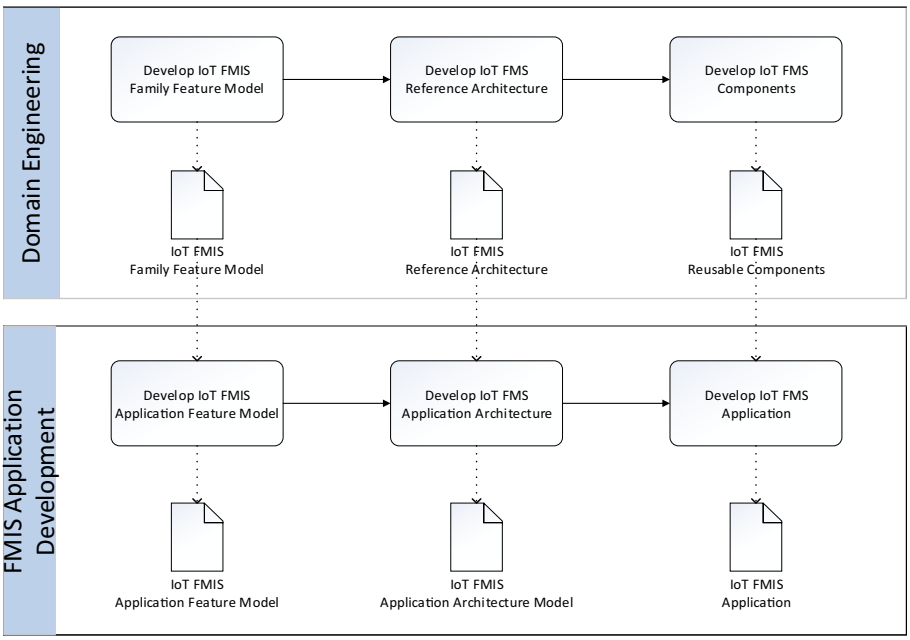

Fig. 2 FMIS development approach

Family feature model

The first step of the domain engineering activity of the proposed approach in Fig. 2 is the development of a family feature diagram for IoT-based FMIS. A feature diagram is a tree that is used to model the commonalities and differences within a specific domain or system. The feature diagram includes a root node representing the domain or system that features the essential characteristics or externally visible properties of the system (Tekinerdogan et al. 2012). Features may have sub-features which can lead to a hierarchical tree. Features can be mandatory or variant. Variant features are usually represented as optional or alternative features. Optional features can be selected or not, whereby alternative features require the selection of one of the defined features. A feature configuration is a set of features which describes a member of a communication protocol. A feature constraint further restricts the possible selection of features to define configurations.

The domain analysis consists of two basic activities including domain scoping and domain modeling. In the scoping process, the domain scope is defined and the set of knowledge sources are selected. In the domain modeling process the feature diagram is provided.

During the domain scoping process for the IoT based FMIS, not only scientific papers but also websites and white papers of the important vendors and stakeholders in the IoT and smart farming domains have been considered. The selected list of important sources that were considered for IoT are shown in Appendix 1, the list of sources for smart farming are shown in Appendix 2. It is not claimed that the set of sources is comprehensive but an analysis of these selected studies shows a convergence and agreement on the concepts. In the following section the feature diagram for IoT is described first followed by a description of the feature diagram for smart farming.

Feature model for IoT

Based on the primary studies given in Appendix 1, the top-level feature diagram of the IoT as given in Fig. 3 is obtained. In essence, the top-level figure diagram presents the design features and as such includes the mandatory features for the layers of the earlier defined IoT reference architecture in Fig. 1. The feature diagram states that all the layers are mandatory for setting up an IoT system.

For each of the layers a detailed feature diagram that represents the commonalities and variability for the corresponding layer can be derived. Among the IoT layers it appears that the decisive layer is the session layer that includes the protocols for initiating the connection and the further communication session. Figure 4 shows the feature diagram that was derived from the domain analysis to the IoT session layer communication protocols.

The top-level mandatory features in the feature diagram are protocol type, source-target, transport type and architecture. The protocol type feature defines the protocols that were identified from the selected primary studies. These identified protocols are the following:

- Message Queuing Telemetry Transport (MQTT): One of the most popular protocols to collect device data and communicate with servers (OASIS 2011).
- Extensible Messaging and Presence Protocol (XMPP): is based on exchanges of XML messages in real time that is defined to connect devices to servers (IETF 2011).

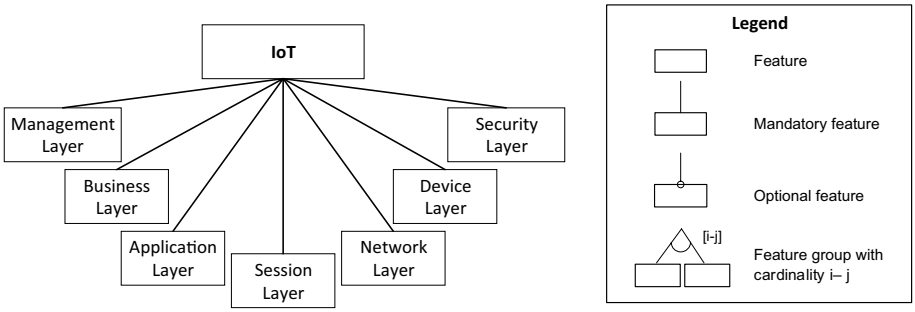

Fig. 3 Top level feature diagram of IoT

- Advanced Message Queuing Protocol (AMQP): A queuing system designed to connect servers to each other (OASIS 2011).
- Data Distribution Service (DDS): A fast data bus for integrating devices and systems (OMG 2015).
- The Constrained Application Protocol (CoAP): A specialized web-based protocol to be used in constrained nodes and constrained networks (IETF 2011).

As given in Fig. 4, there are three types of source-target relations available in session layer protocols: Device-to-Device (D2D), Device-to-Server (D2S), and Server-to-Server (S2S). In some studies, these features are also called Machine-to-Machine (M2M), Machine-to-Cloud (M2C), and Cloud-to-Cloud (C2C) respectively.

Session layer protocols are closely related to the network layer. For all communication protocols, the transport layer could be either the user datagram protocol (UDP) or transmission control protocol (TCP). Some protocols like DDS, support both UDP and TCP. The addressing scheme (unicast, broadcast, or multicast) might be important depending on the application requirements. The selection of network layer protocol is important since using TCP and/or UDP changes the characteristics of the communication from performance and security perspectives. The layer below the network layer, that is the device layer includes the IoT devices that have a direct connection with the physical devices. The physical devices have different functionality and transmission range. If low power devices and networks will be used, adoption of TCP in the network layer is generally less feasible, and likewise the UDP protocol is used instead. On the other hand, TCP is required for supporting security and the common security protocols of (SSL/TLS) are not available using UDP.

The architecture of the session layer protocols can be either publish-subscribe or request-reply. In publish-subscribe architecture, participants (e.g. sensors) send data to a

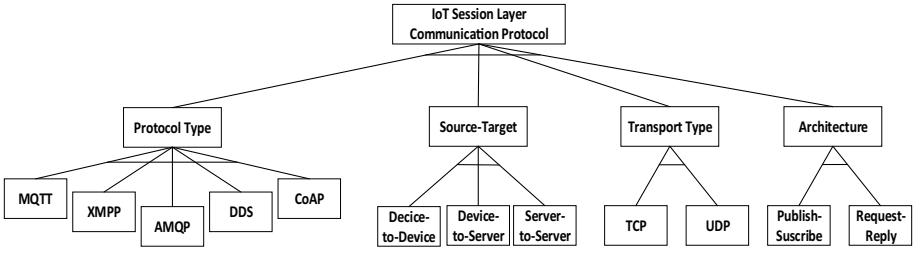

Fig. 4 Feature diagram of session level communication protocols of IoT

topic on which several subscribers (e.g. cloud software) that are registered to this topic might read data. In this architecture publishers and subscribers do not need to know each other, and do not need to be operating at the same time, i.e. this communication type provides time and space uncoupling. This type of communication is well suited for data that must flow from one producer to many consumers. On the other hand, for the request-reply architecture, senders and receivers do need to know each other. The requester sends a request message and waits for the response. When the replier (e.g. sensor) receives the request, it responds with a reply message. The session layer protocols of IoT generally use publish-subscribe architecture except in the case of CoAP in which a request-reply pattern is adopted. There are many criteria to select the right IoT session layer protocol depending on the application requirement. Further information on the selection of the proper IoT session layer protocol is provided in (Köksal and Tekinerdogan 2017).

Feature model for farm management information systems

For IoT based smart farming the other important domain is of smart farming itself. Similar to the IoT domain again a domain analysis process has been applied in which relevant primary studies on smart farming were searched and based on these selected studies a feature diagram was derived representing the common and variant features. The selected primary studies are listed in Appendix 2. Based on the literature the following sub-domains for smart farming could be identified: (1) Global Positioning Systems (GPS), (2) Geographical Information Systems (GIS), (3) Sensors, (4) Variable Rate Technology (VRT), (5) Yield Monitoring (YM), (6) Yield Mapping (YMAP), and (7) Farm Management Information Systems (FMIS).

FMIS software is a core part of the smart farming. FMISs are used to collect and process data used to manage all farming operations. The top-level feature diagram for FMIS is given in Fig. 5. The right part of the feature diagram has been derived from the FMIS functions as defined in (Fountas et al. 2015). The left part focuses on the IoT related functionality of FMIS including collection of data, processing data, visualization of data, communication with external systems, and system management. This paper will further focus on the IoT FMIS aspects.

The data acquisition feature defines the gathering of data from sensors and other systems used in smart farming such as tractors, agribots and unmanned vehicles. It includes functions for processing the gathered data whereby useful information is extracted using data mining, machine learning, and image processing. The data visualization feature defines the displaying of processed data in different forms. Apart from classical tables, reports, and monitoring tools, dedicated visualization maps are essential for smart farming applications. Maps such as yield maps, soil maps, lighting maps, and profit maps are displayed for different purposes. The system management feature defines the management of data acquisition, processing, visualization, and external system communication features of FMIS. The quality related functions such as reliability, scalability, extensibility, and security are considered in this feature. Finally, the “communication with external systems” feature defines the communication with external systems, such as a weather forecast system. Each of these features are shown in a different color in Fig. 5 to refer to these in later sections.

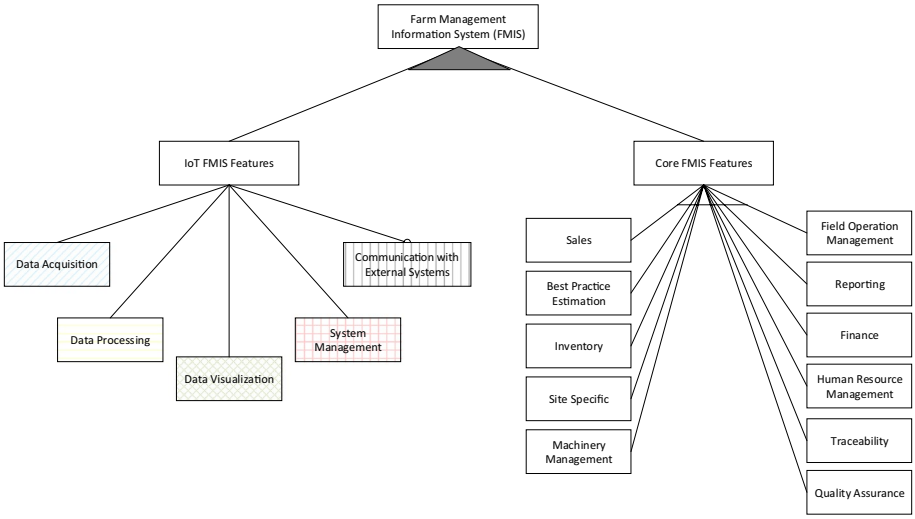

Fig. 5 Top level feature diagram of FMIS

Feature model for IoT based FMIS

In principle, IoT and FMIS are independent concepts and as such these have been modeled separately in the previous sub-sections. In principle the features from the IoT feature diagram and the features from the FMIS feature diagram could have been selected. Alternatively, the notion of an IoT based FMIS can be considered as the integration of both concepts that needs to be separately considered. To this end, Fig. 6 shows the integrated family feature diagram derived from the feature models for IoT and FMIS. The focus was on FMIS as the dominant decomposition and the IoT features were integrated in the separate leaves of the FMIS feature tree. In the figure for different type of features different fill patterns have been used. The detailed feature diagram given in Fig. 6 can in principle be further extended with respect to specific project requirements. For the context of this paper though, the provided feature models are sufficient to illustrate the approach.

In IoT based FMIS, data acquisition consists of IoT data acquisition and conventional data acquisition to support legacy systems. IoT data acquisition contains 5 alternative IoT session layer protocols as discussed earlier namely MQTT, XMPP, AMQP, DDS, and CoAP. Depending on the application one or more protocols for IoT communication can be selected for the FMIS. In the earlier section the criteria for this have been described. A traditional data acquisition feature consists of ISOBUS and controller area network (CAN) protocols. As stated before, other (legacy or non-IoT) protocols can be added to the feature diagram.

Data processing features mainly depend on the application type and include Image/Video processing, data mining, data logging, and decision-support features. One or more features might be used at the same time. Depending on the application requirements these features can be extended to use different processing features.

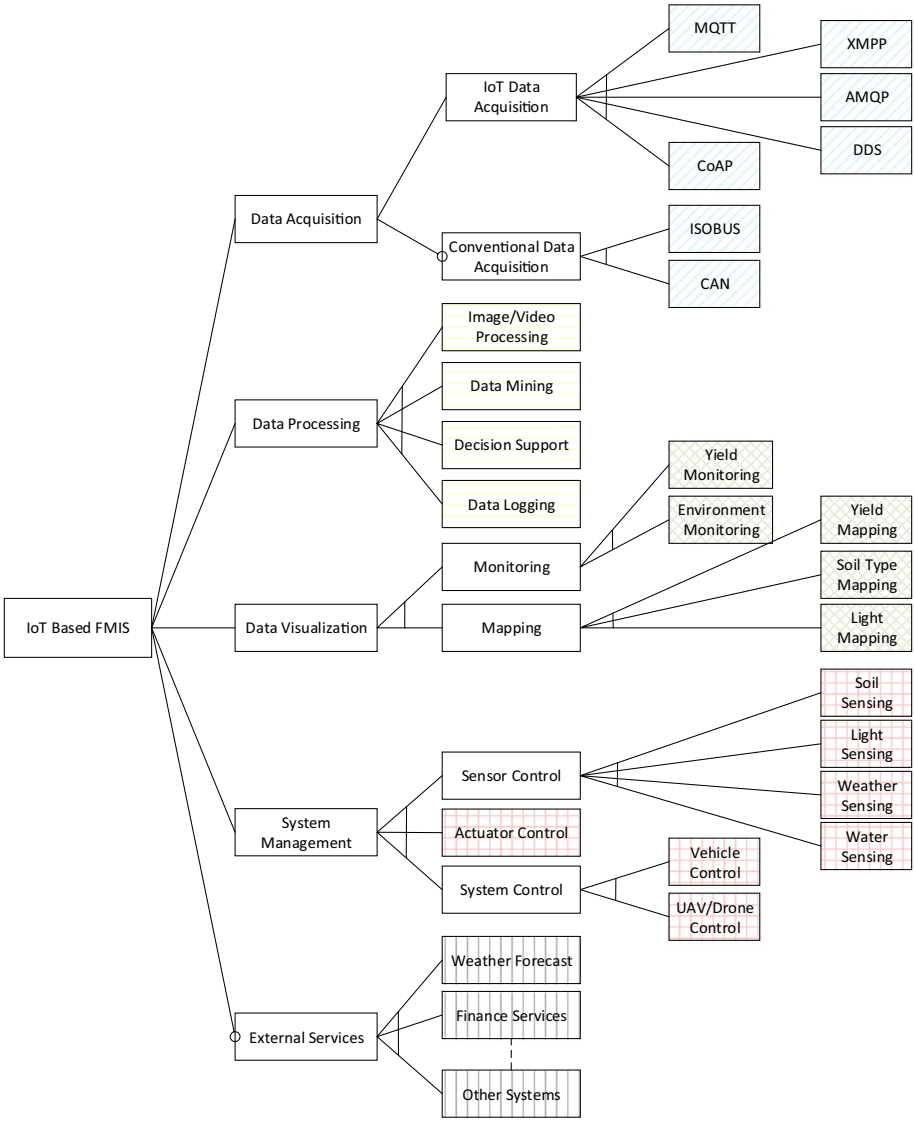

Fig. 6 Family feature diagram of FMIS software

Data visualization consists of monitoring and mapping functions. Monitoring consists of environment monitoring and yield monitoring functions. Mapping includes yield, soil-type, and light mapping features.

System management includes sensor control, actuator control, system control features such as device identification, node discovery, and directory and naming services. Sensor control consists of several sub-features such as soil sensing, light sensing, weather sensing, and water sensing. Also, system control includes vehicle control and UAV/Drone

control features. Finally, external services feature contains externally communicated systems such as weather forecast, finance services, and other external systems.

Reference architecture for FMIS

Once the family feature models for IoT based FMISs have been developed, the next step is the development of the reference architecture for the potential systems. In fact, in the literature several reference architectures have already been proposed for FMISs (Nikkilä et al. 2010; Sørensen et al. 2010, 2011; Beck 2001; Fountas et al. 2006) However, in general these reference architectures are either at a conceptual level and/or do not consider IoT aspects explicitly. Hence, in this section, the reference architecture for IoT based FMIS will be introduced. For this, selected viewpoints of the Views and Beyond architecture framework (Clements et al. 2010) will be used including the decomposition viewpoint, layered viewpoint and deployment viewpoint.

Decomposition view

The decomposition view is used to show how system responsibilities are partitioned across modules and how these modules are decomposed into sub-modules. Usually, the features in the feature diagram are realized by one or more modules in the decomposition view. The decomposition view of the architecture depicts the overall structure of the architecture which is reasonably decomposed into modular implementation units. It is regarded as a fundamental view of the architecture since it serves as an input for other views (e.g. work allocation view) and helps to communicate and define the structure of the software. The proposed reference decomposition view for the IoT based FMIS is given in Fig. 7. The modules in the decomposition view are colored to make the link with the earlier defined features in the feature diagrams of FMIS. In essence, the decomposition view includes the modules for data acquisition, data processing, data visualization, system management, and modules for communication with external services. The decomposition view includes all the possible modules for the various IoT based FMIS applications. Note that in this case for each feature in the earlier feature diagram, one module has been defined in the decomposition view. Further the lower level functionalities such as node discovery, and directory and name services, have not been depicted. The following section explains the configuration of a specific decomposition view from this reference decomposition view.

Layered view

The layered view is similar to decomposition view since it reflects the division of software into units. The difference is that in a layered view, modules are structured into *layers*, which interact based on a strict ordering relation. This means that if layer A is allowed to use layer B, layer A's implementation can use any public facilities of Layer B. However, layer B cannot use any facilities of layer A.

Figure 8 shows the layered view for the IoT-based FMIS. Here the dominant decomposition is taken from the IoT layered view as it was given in Fig. 1, and likewise it includes the same layers of the IoT reference architecture. The specific details are primarily in the higher-level layers including the business layer, the application layer and the data acquisition layer. The FMIS business layer includes all required farm management operations

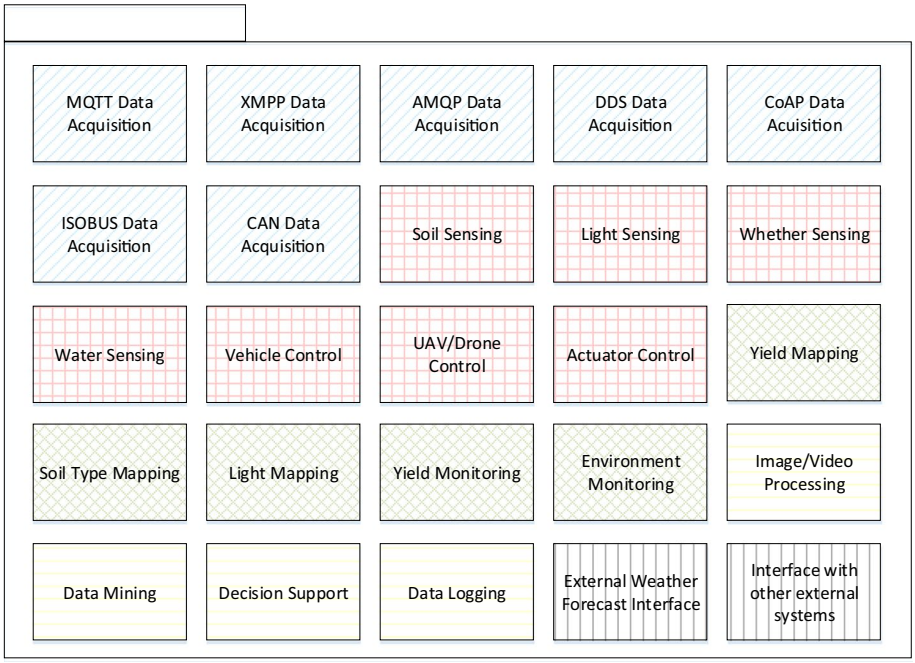

Fig. 7 IoT based FMIS—decomposition view

logic and workflows such as Fertility Management, Nutrient Management, Pest Management, Weed Management, and Irrigation Management. The FMIS application layer includes FMIS Data Processing, Data Visualization, System Management, and Communication with External Systems. The FMIS Data Acquisition Layer is for data adaptation

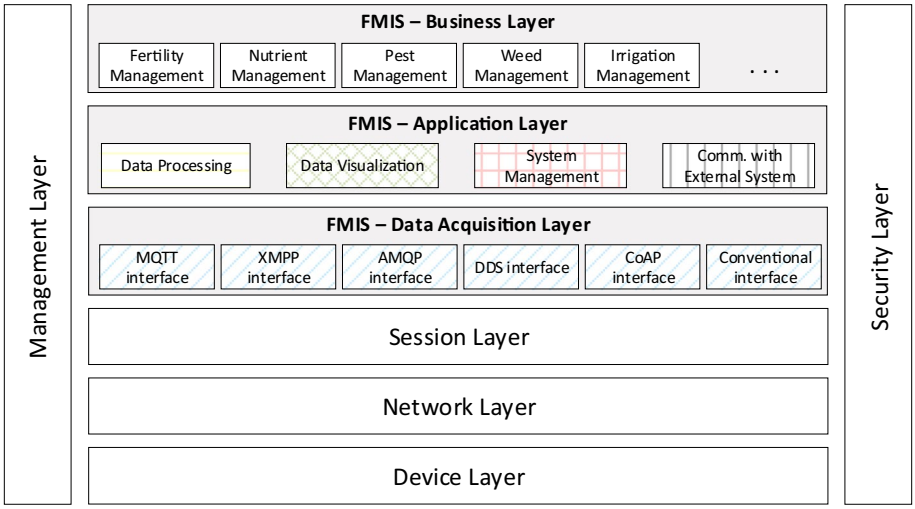

Fig. 8 IoT based FMIS—layered view

between the IoT session layer and FMIS, i.e. this layer provides the connection with the session layer protocols of the IoT. As such, this layer includes the IoT session layer interfaces such as MQTT interface, and XMPP interface. To support non-IoT systems, the module conventional interface also takes place in this layer.

Deployment view

The earlier defined views (decomposition, layered) focus on modeling the software modules of the IoT-based FMIS. The deployment view elaborates on these views and is used to show the allocation of the identified software modules to the hardware of a computing platform. The deployment view of an IoT based FMIS is given in Fig. 9. The data processing module is deployed on the Central Cloud Server and Client (Farmer) nodes. The other nodes are dedicated to sensors, actuators, and cameras. The main sensors in the UAV/Drone and Satellite are cameras. Vehicles can have their own sensors, actuators, and cameras. Hence, these items can be assigned to different nodes.

Case study evaluation

Case study protocol

The primary objective of the case studies is to evaluate the impact of the developed architecture design method for IoT based FMIS. To evaluate the above research questions, the case study research protocol as defined by Runeson and Höst (2008) has been applied. Based on this the indicated five steps have been followed: (1) case study design

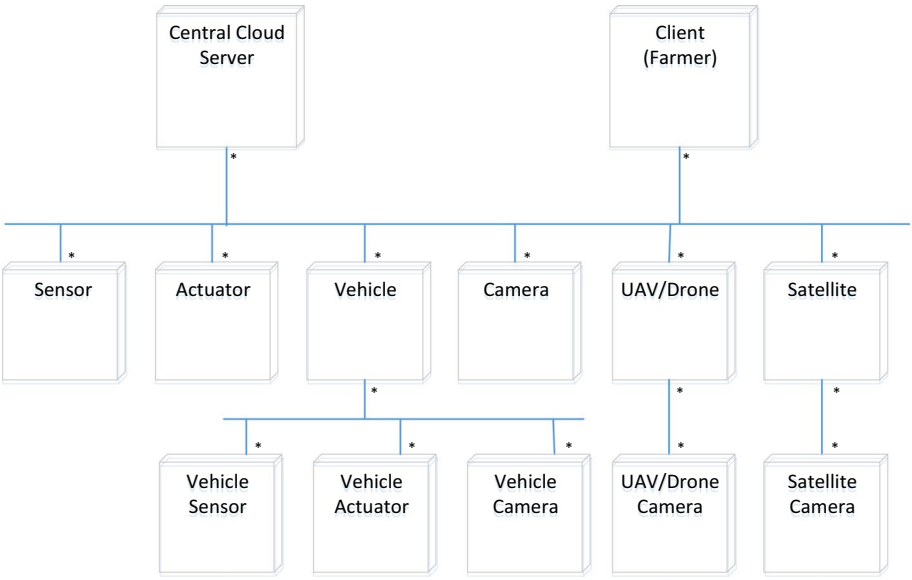

Fig. 9 IoT based FMIS—deployment view

(2) preparation for data collection (3) execution with data collection on the studied case (4) analysis of collected data (5) reporting.

Table 1 shows the case study design elements. The case study research has been applied both for a retrospective case and prospective case. The retrospective case included a system which was developed before and for which there was already an existing architecture with required design documents. The prospective case includes the system that is planned to be developed.

The goal for the retrospective study was to compare the earlier result with the result that is produced by proposed method. In this way it was aimed to evaluate the effectiveness of the method. For the prospective case study, the aim was to evaluate both the effectiveness and the practicality of the approach. The research questions were defined accordingly as it is shown in the table.

For the adopted background and sources in the case study research, official design documents have been used and interviews were held with project managers and system architects. For the retrospective case study, the requirements and design documents were available, whereas for the prospective case study only the requirements document was available. For both case studies project managers and system architects were interviewed. The project manager had more than 20 years of experience in farm management system. The two system architects had more than 15 years of experience in designing farm management systems.

For both case studies a qualitative data analysis approach has been used. For the retrospective case study, indirect data analysis has been used by analyzing the requirements documents, applying the method and comparing the results of the method with the existing architecture. For the prospective case study, a direct and indirect data analysis approaches were used. For the direct data analysis, semi-structured interviews were conducted, in which a list of predefined set of questions were asked to the project manager and software architects. The predefined questions are listed in Table 2. The questions included a 5-point Likert scale (strongly disagree to strongly agree) for the possible answers. In addition, explanation was asked for each question. Note that although Likert scales are used for the evaluation of the approach, it is not aimed to be quantitative but primarily qualitative. The interview questions with Likert scales as such were used as supporting instruments to the open semi-structured interview that were held with the stakeholders.

The interview was organized as follows:

1. First a meeting was scheduled with the project manager and system architects for the initial interview. The goal of this interview was to capture the initial thoughts and experience on IoT adoption.
2. In the second step a short presentation was provided about the goal of the developed method. Also, the operation of the method as well as the final outcome was shortly explained.
3. In the third step the method was applied both for the retrospective case (wheat production) and prospective case (tomato production in greenhouses).
4. In the fourth step, the researchers analyzed the architecture design that resulted from the application of the method to the retrospective case and the prospective case.
5. In the fifth step, the researchers held a post interview with the subjects with the purpose of identifying the impact of the method and its practicality.

Table 1 Case study design

Case study design activity	Retrospective case study (wheat production)	Prospective case study (tomato production)
Goal	Comparing and assessing the method feasibility and recommended application architecture	Assessing the effectiveness of the method Assessing the practicality of the method
Research questions	RQ1: To which extent is the derived application architecture in alignment with the decision of the case study?	RQ1. To which extent does the method support the architecture design of the IoT based FMIS? RQ2. How practical is the method for deriving the IoT based FMIS application architecture?
Background and source	Official requirements documents Official architecture design documents Project Manager and System architects	Official requirements documents Project Managers and System architects
Data collection	Indirect data collection based on document analysis (the design documents and technical reports)	Indirect data collection and direct data collection through semi-structured interviews (mix of open and closed questions)
Data analysis	Qualitative data analysis	Qualitative data analysis

Table 2 Questionnaire for the Interview

Questions
With information at hand, are you planning to increase the adoption of IoT in the future?
Do you think that this reuse-based architecture design method is more effective than the architecture design method that you adopted so far?
Do you think that the provided recommended application architecture is of high quality?
Do you think that the reference architecture is of high quality?
Is the method and the reference architecture sufficient to derive the application architecture?
Do you think that the method is practical?
Will you use the method again?
Do you think that the application of the method can provide a competitive advantage to the organization?
Has the usage of the method enhanced your knowledge on IoT based FMIS?
Do you have any suggestions for improving the method?
Do you have any suggestions for improving the family feature models?
Do you have any suggestions for improving the reference architecture?

6. In the sixth step, the researchers collectively assessed data from the initial interview, report delivered by the method, and the post interview. The assessment was carried out separately and later was discussed together to derive the lessons learned.

The following subsection discusses the results of the above process after the first two steps. Subsequently the evaluation in step 4, 5 and 6 are described.

FMIS architecture design

As stated before, the application architecture is derived from the family feature model and the reference architecture. As described earlier the FMIS development method includes the development of the application feature model and the application architecture.

Retrospective case study: smart wheat production

Figure 10 shows the feature model for smart wheat production that is derived after the analysis of the existing case study. This application feature model is obtained by reusing the feature model for FMIS given in Fig. 6 and selecting the features that are needed for this case study. As shown in the figure, for this case study, MQTT session layer protocol of IoT is chosen. The main reasons were because open source implementations of MQTT could be used. MQTT supports TCP and device-to-server communication which were considered necessary in the given context. Likewise, the MQTT feature of the IoT Data Acquisition will be used. Also, in order to support conventional data acquisition with tractors used in wheat production ISOBUS and CAN communications will be supported. Almost all data processing and data visualization features are required for smart wheat production. So almost all of these features remained in the feature diagram of Smart Wheat Production. For this retrospective case study, the FMIS will be integrated with the external weather forecast system only.

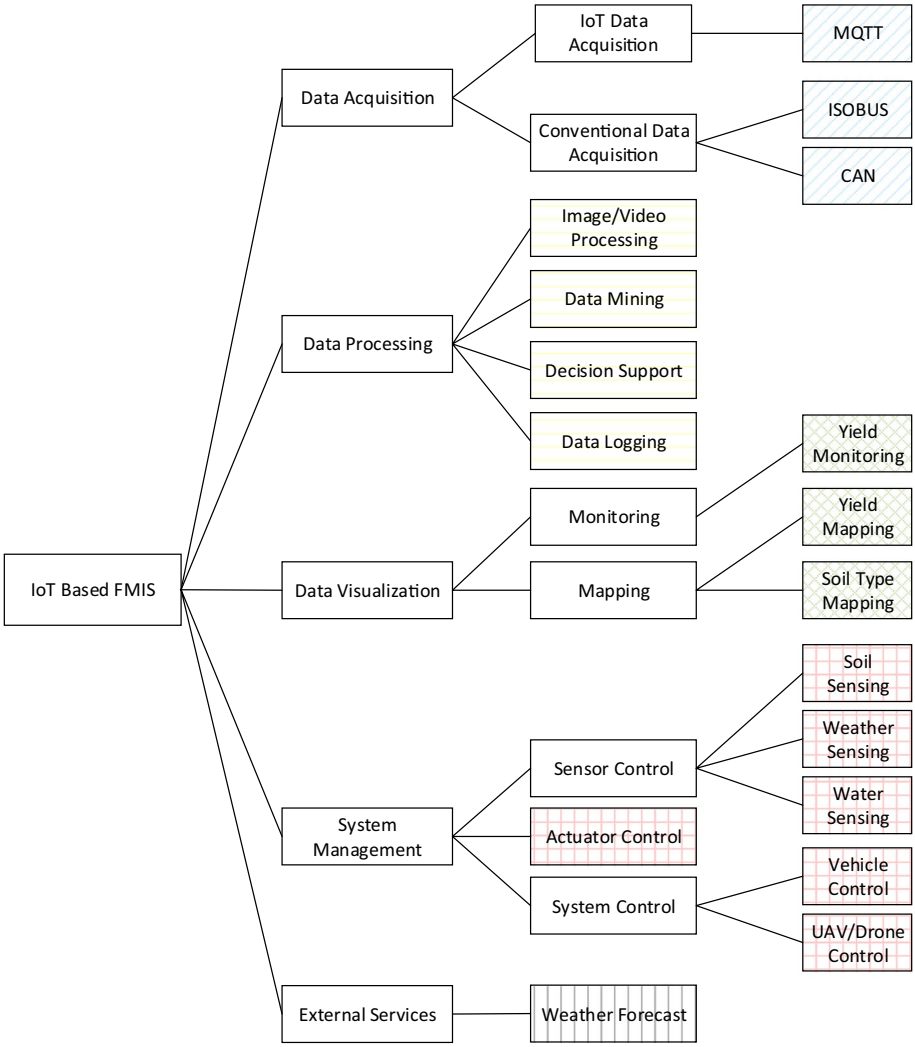

Fig. 10 Application feature diagram of the FMIS software for Iot based smart wheat production—retrospective case study

Decomposition view

Based on the selected features as defined in Fig. 10, the application architecture can now be derived. As discussed before, the architecture of a system is usually described using multiple different architecture views. For each of the required architectural views it is necessary to develop the application arhiteture view. Figure 11 shows the decomposition view of the Smart Wheat Production that is obtained using the reference decomposition view given in Fig. 7. As explained above, MQTT, ISOBUS, and CAN data acquisition modules will be used to support IoT communications. All the sub-features of the system management feature of the family feature model will be used, except the light sensing feature. The light

sensing feature is used to obtain light maps in the greenhouses. Yield monitoring, yield mapping, and soil type mapping modules will be used to implement data visualization features. All data processing modules namely image/video processing, data mining, decision support, and data logging modules will be implemented. Finally, a single external communication interface: external weather forcast interface module will be implemented.

Layered view

Figure 12 shows the layered view of smart wheat production. Similar to the other views, this view is also customized from the reference layered view diagram which is given in Fig. 8. Here the modules of the decomposition view are distributed over the layers in the layered view. The modules MQTT interface and conventional interface are allocated to the FMIS-data acquisition layer. The FMIS-application layer includes the modules data processing, data visualization, system management and communication with external system. The FMIS-business layer include fertility management, nutrient management, pest management, weed management, and irrigation management. It was assumed that the other layers and the modules in these layers are similar as defined in the reference architecture.

Deployment view

The deployment view of the smart wheat production case study is given in Fig. 13. The required software modules given in the decomposition view are deployed to a central cloud

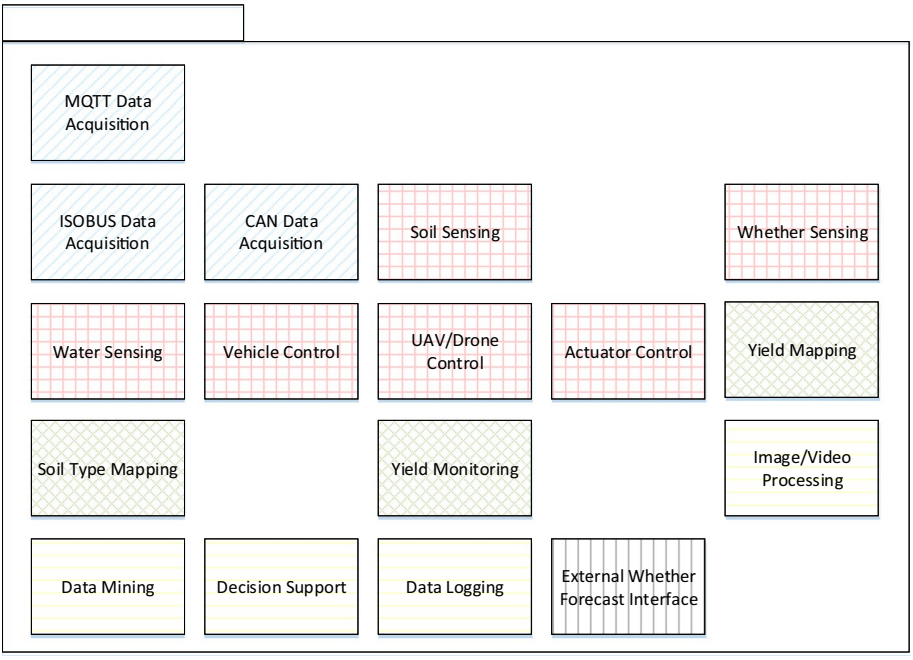

Fig. 11 IoT based FMIS—decomposition view for an IoT based smart wheat production—retrospective case study

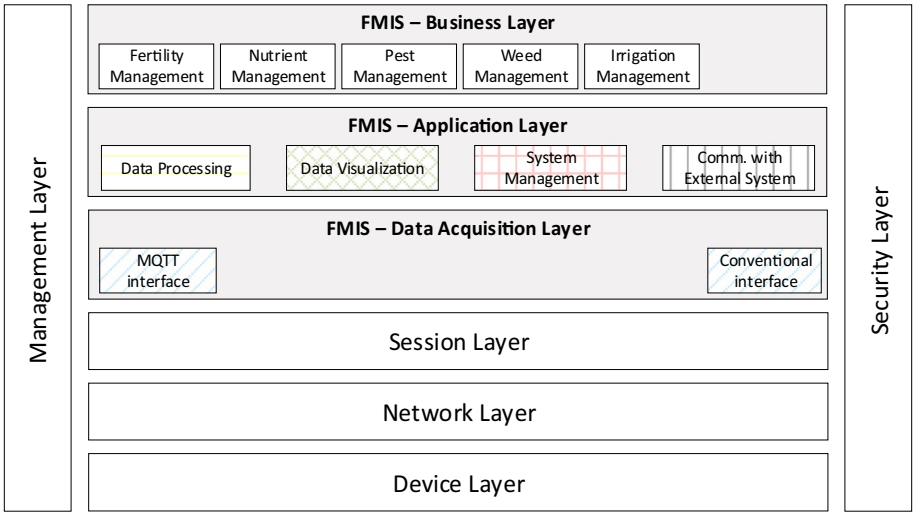

Fig. 12 IoT based FMIS—application layered view for an IoT based smart wheat production—retrospective case study

server and client (farmer). Since there is no satellite to be used it is omitted for this case. A subset of the features from the ground sensors actuators, and cameras, on-the-go sensors, actuators will be used. These will be deployed on vehicle, tractor or UAV/Drone nodes.

Prospective case study: IoT based smart tomato production in greenhouses

Similar to the retrospective case study the steps of the approach have been applied to derive the FMIS architecture. Due to space limitations the description of the steps will not be repeated and the specific diagrams will not be shown. It should be noted that unlike the retrospective case study, for the prospective case study there was no ready defined architecture. This did not have an impact on the application of the method but for the evaluation the comparison with existing architecture could naturally not be done.

Result of the evaluation for the retrospective case study

The previous subsection has shown the application of the approach. As defined in the case study protocol the effectiveness and practicality of the approach has been analyzed. The results of the interview showed that for all the asked questions a score of 4 or 5 were provided. Based on these results the effectiveness and practicality of the approach will be elaborated on.

Effectiveness of the approach

For assessing the effectiveness of the approach, the resulting application architecture was analyzed and post-interviews were carried out. For the retrospective study the application architecture was described in different document formats including MS PowerPoint and MS Visio. Further some detailed design documents could be accessed. The available architecture was

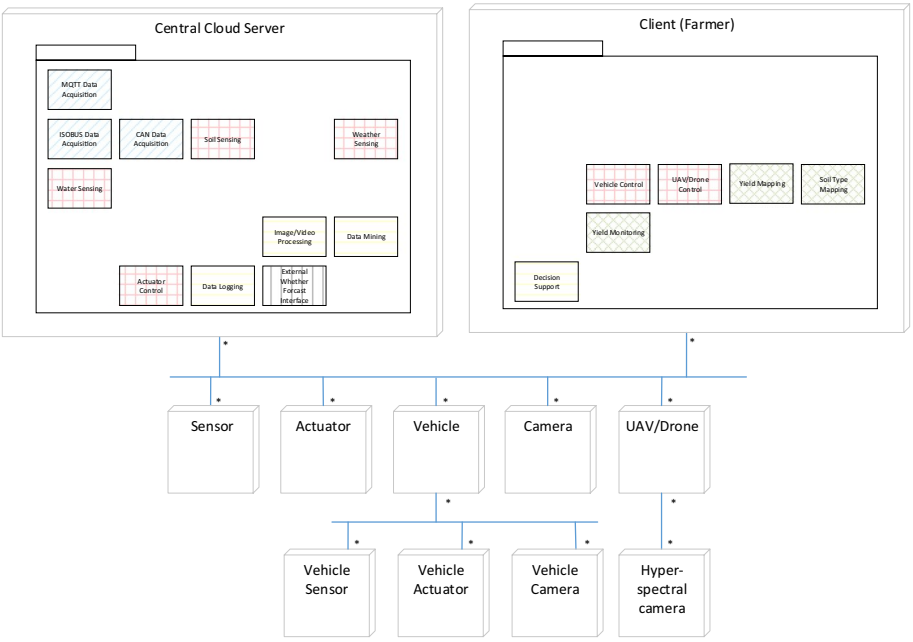

Fig. 13 IoT based FMIS—deployment view of Iot based smart wheat production—retrospective case study

not based on a well-defined viewpoint approach and no systematic approach was used. Nevertheless, it was possible to analyze the existing application architecture and compare it with the application architecture that was derived using the presented approach. For the comparison three relations were identified including (1) convergence (2) deviation (3) absence. The convergence relation implies that similar architecture elements could be identified in both the architecture designs. Deviation implies that the resulting application architecture had additional elements that were not defined in the existing architecture. Finally, absence defines that the resulting application architecture had missing elements that were defined in the existing architecture. Overall the result of the analysis showed that the resulting application architecture was quite similar to the existing architecture. In general, the convergence was very high. Several deviation and absence relations could be identified though. With respect to deviation in the resulting application architecture the modules *UAV/Drone Control* and *External Weather Forecast Interface* were not defined in the existing architecture. On the other hand, also some absence relations could be identified. For example, the resulting application architecture did not have the module *Finance Interface Module, Farmer Data Module,* and *Simulation Module* that were explicitly defined in the existing application architecture. This became also apparent in the post-interviews. In the interview, the questions 2, 3, 4 and 5 relate directly to the effectiveness of the approach. The architects provided a score of at least 4 for all these questions indicating that the approach is largely effective for the corresponding case.

Practicality of the approach

The practicality of the method was assessed though questions 6 to 9 of the questionnaire. The architects gave at least a score of 4 for all these questions indicating that they were

quite satisfied with the practicality of the method. Further it was stated that the architecture design could now be more quickly designed. Another interesting issue that was explicitly mentioned was the fact that the method helped to support communication about the design decisions early on. For the question "*Will you use the method again*?" the answers were positive again and both architects indicated that they would use this method for the subsequent project. The architects also had some suggestions for improvement including the need for allowing specific delta modules when deriving the application architecture.

Result of the evaluation for the prospective case study

Similar to the results of the retrospective study, the answers to the provided questions were positive and received a score of 4 or higher. In the following the effectiveness and practicality of the proposed approach is discussed for the prospective case study.

Effectiveness of the approach

For the retrospective case study, the resulting architecture was compared with the existing architecture to assess the effectiveness of the approach. For this prospective case study, the results could not be compared with an existing application architecture since only requirements document was provided and the application architecture had still to be designed. The requirements document was used to identify the required application feature diagram. Based on this the application architecture was derived. The effectiveness of the application architecture and the overall approach was based on the results of the interview.

The scores for the questions 2 to 5 related to effectiveness were at least 4. This indicated that the approach was effective for the given case. The architects noted that the application could be easily derived based on the defined requirements.

Practicality of the approach

For assessing the practicality of the approach, the results of questions 6 to 10 were considered. Again, it appeared that these had all a score with at least 4. Similar to the retrospective case study the approach was found practical and easy to use. Similar statements as in the retrospective case study were made. The explicitly mentioned identified benefits included the reduction of effort to design the architecture, discuss the design decisions, and the surprisingly shorter learning curve than expected.

Discussion

The introduction of the IoT has led to the notion of IoT-based FMIS to support the smart farming goals. This paper aimed to integrate the IoT systems with the FMIS to align both systems and create additional value that cannot be achieved if these are considered separately. This integration effort leads to the enhancement of the current FMIS with new modules that support the smart farming operations based on IoT. In the presented approach these required new modules have been explicitly defined in addition to the traditional FMIS modules. The overall approach as such provides an integrated view of the overall system to better support the architecture of IoT-based FMIS.

The method that was discussed can be adopted for deriving an IoT based FMIS architecture for multiple different systems. Hence, the focus is on the whole product family of IoT based FMISs rather than on a single system. The notion of product families or product line engineering and the corresponding systematic reuse is discussed in detail in the product line engineering community (Clements 2006). The presented method is inspired and customizes the product line engineering approach in which reference models are developed and applications are developed by reusing these reference models. The reference feature diagram that was shown aims to target and integrate the domains of IoT and FMIS. The focus in this paper was primarily to illustrate the overall method. The feature diagrams as well as the reference architecture design could be easily extended. The architectures for IoT and FMIS were discussed separately and the integration of both for supporting IoT based FMISs was illustrated. The architecture can be extended in two ways. First of all, the different views could be further refined to provide an even more comprehensive result. This would require for example to further detail the modules that are needed in the decomposition view. Secondly, the architecture representations could be extended with other architecture views. Three architecture views were chosen including decomposition view, layered view and deployment view. If needed other architecture views in the architecture documentation process could be used as well. The complete versions of the feature diagrams as well the detailed implemented architecture designs have not been shown due to confidentiality issues.

The reference architecture is designed in such a way that it is generic enough to derive different concrete architectures. Nevertheless, as is the case for all reference models, the reference architecture does not provide all the details. Likewise, a system which requires very dedicated features that were not anticipated would not be covered by the reference architecture. Furthermore, the focus has been on illustrating the reference architecture and the approach for deriving a concrete architecture. This appeared to be useful and practical. However, it is not claimed that the reference architecture is complete, and likewise further research can be used to refine and enhance the reference architecture. For example, the device layer and the related functionality have not been discussed in detail in this article. This could though be easily added without loss of generality and applicability of the proposed approach.

Although, the approach has been shown for two important case studies in the smart agri-food sector, the method can be also applied for the development of other FMISs. The paper did not focus on the implementation of these systems. The reason for this is because of confidentiality and the goal to prescribe the system-to-be in the prospective case study. For the prospective case study, it is decided to develop first a simulation system to evaluate the outcome of the method. This is considered as part of future work.

This paper provided both the reference architecture for IoT-based FMIS and the overall approach to derive a concrete architecture. The idea of systematic guidelines for deriving a concrete architecture could also be used for enhancing the use of existing IoT-based reference architectures.

Although the proposed method has illustrated the development of IoT based FMISs, the method could even be used for developing traditional FMISs. In that case the IoT architecture part would be omitted and just the development of reference models for FMIS would be considered.

This paper describes a domain-driven approach to design IoT-based FMIS and support the architect in deriving a concrete IoT-based FMIS architecture. Several other important issues need to be considered to realize effective smart farming. Important aspects include the acceptability of the provided IoT technology by the relevant stakeholders including the

end-users, development of cost-effective transition strategies, and farm management and agricultural economics. Detailed discussion on economics and profitability of IoT solutions in the agriculture domain have been addressed by multiple studies including (Griffin and Lowenberg-DeBoer 2005; Griffin et al. 2018; Wolfert et al. 2017; Lowenberg-DeBoer et al. 2000; Schimmelpfennig 2016; Kutter et al. 2011).

Adopting IoT-based FMIS is not trivial and usually requires large economic investments. To justify these up-front investments the return-on-investment both with respect to cost and quality should be defined. Further, IoT-based FMIS potentially requires changes to farm equipment or totally new farm equipment. Hence, it is important to analyze the acceptability and adoption scenarios, and provide clear transition strategies for the efficient introduction, usage and maintenance of smart farming. Due to space limitations and the concrete scope of the paper this is not further discussed.

A systematic case study research has been applied to validate the proposed approach. Each empirical study usually has to deal with a few potential threats to validity. In the following these are discussed for the presented case study research, and the mitigation strategy for each threat is described.

Construct validity refers to what extent the operational measures that are studied really represent what the researchers have in mind and what is investigated according to the research questions (Yin 2009). Table 3 shows various identified threats to construction validity together with the counter measures.

Internal Validity relates to a causal relationship between treatment and the outcome. The retrospective case study relied on existing design documentation and related literature. There could be missing information in both cases that would affect the outcome. To mitigate this threat several iterations were done to derive both the application feature model and the application architecture. In the prospective case, the lack of proper requirements documentation could have an impact on the derived decisions. To mitigate this threat, this has been discussed with the interviewed persons in detail and several iterations were adopted.

External Validity concerns the ability to generalize the results of the study. In the evaluation, both a retrospective and prospective case study were adopted derived from different application domains. This was done to support triangulation and likewise extend the external validity. The approach was considered effective for both case studies but due to the small number of participants a stronger statement could not be provided. In the future work a repetition of this study with multiple other case studies with an increased number of participants would further justify the claims of this paper and also support the quantitative evaluation.

Conclusion

Farm management information systems are being increasingly applied in many farming systems. Several architectures for FMISs have been proposed in the literature but these are usually abstract, and it is not trivial to derive the application FMIS architecture for the corresponding context of the farm system. This paper has provided an architecture design method for deriving application architectures for FMISs. For this, reference architectures for IoT and FMIS were adopted, and a novel IoT based FMIS has been defined. The architecture design method has been provided for deriving the customized application FMIS architecture. To support the design of the application architecture a domain driven

Table 3 Threats to construct validity and applied counter measures in case studies

Threat	Countermeasure
Inappropriate analysis of existing requirements and architecture (for retrospective case study)	To ensure that all the requirements were understood a meeting was organized. The missing artefacts were reverse engineering and discussed with the architects
Incorrect interpretation of the descriptions of the questions by the interviewed persons	The principles described in Kitchenham and Pfleeger (2002) were applied for constructing the questions and answers. To ensure uniqueness of interpretations of the questions, detailed explanations were provided
Incorrect interpretation of the description of the answers by the interviewed persons, and likewise the wrong selection of answers	The descriptions of the answers were cross-checked and discussed by both researchers. Additional detailed explanations were provided. Answers that were not clear were reformulated before the interviews
Incorrect interpretation of the open questions by the interviewed persons	To mitigate this threat, the interpretation of the questions was verified with the interviewed persons
Incorrect interpretation of the researchers to the provided answers of the interviewed persons	To mitigate this threat both researchers were present in the interview to achieve observer triangulation

approach has been adopted whereby a family feature diagram was defined representing the common and variant features of IoT based farm management information systems. The approach was illustrated using a systematic case study approach. Both retrospective and prospective case studies were implemented including IoT based wheat production and IoT based tomato FMIS. The case study research showed that the approach was both effective and practical. It appeared that both the reference architecture that was provided as well as the corresponding method appeared to be very useful in deriving the customized application FMIS architecture. Since in general developing IoT systems is not trivial adopting a systematic approach appears to be useful in not only the final results but also the intermediate steps that support the communication between the stakeholders and the overall guidance of the design decisions. The contribution of this paper can be useful for both researchers who do research on IoT based FMISs as well as practitioners who aim to architect different FMISs. The future work will apply the approach for other farm management systems. Further focus will be on the architecture design and integration of multiple different FMISs.

Appendix 1: Selected set of primary studies for IoT

Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayyash, M. (2015). Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications. *IEEE Communications Surveys and Tutorials*, *17(4),* 2347–2376. https://doi.org/10.1109/COMST.2015.2444095.

Gazis, V., Gortz, M., Huber, M., Leonardi, A., Mathioudakis, K., Wiesmaier, A., … Vasilomanolakis, E. (2015). A survey of technologies for the internet of things. In *IWCMC 2015 - 11th International Wireless Communications and Mobile Computing Conference* (pp. 1090–1095). https://doi.org/10.1109/IWCMC.2015.7289234.

IETF. (2001). Extensible Messaging and Presence Protocol (XMPP). *IETF Standard*. https://xmpp.org/about/technology-overview.html. Last accessed 03 February 2018.

IETF. (2014). Constrained Application Protocol (CoAP). *IETF Standard*. http://coap.technology/. Last accessed 03 February 2018.

Koksal, O., & Tekinerdogan, B. (2017). Feature-driven domain analysis of session layer protocols of internet of things. In *Proceedings—2017 IEEE 2nd International Congress on Internet of Things, ICIOT 2017* (pp. 105–112). https://doi.org/10.1109/IEEE.ICIOT.2017.19.

OASIS. (2012). Advanced Message Queuing Protocol (AMQP), Version 1.0. *OASIS Standard*. http://docs.oasis-open.org/amqp/core/v1.0/amqp-core-messaging-v1.0.html. Last accessed 03 February 2018.

OASIS. (2014). Message Queuing Telemetry Transport (MQTT), Version 3.1.1. *OASIS Standard*. http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html. Last accessed 03 February 2018.

OMG. (2015). "Data Distribution Service for Real Time Systems (DDS), v1.4. *OMG Standard*. http://www.omg.org/spec/DDS/About-DDS/. Last accessed 03 February 2018.

Palattella, M. R., Accettura, N., Vilajosana, X., Watteyne, T., Grieco, L. A., Boggia, G., & Dohler, M. (2013). Standardized protocol stack for the internet of (important) things. *IEEE Communications Surveys and Tutorials, 15*(3), 1389–1406. https://doi.org/10.1109/SURV.2012.111412.00158.

Pandya, H. B., & Champaneria, T. A. (2015). Internet of things: Survey and case studies. In *2015 International Conference on Electrical, Electronics, Signals, Communication and Optimization (EESCO)*, 1–6. https://doi.org/10.1109/EESCO.2015.7253713.

Sheng, Z., Yang, S., Yu, Y., Vasilakos, A., McCann, J., & Leung, K. (2013). A survey on the ietf protocol suite for the internet of things: Standards, challenges, and opportunities. *IEEE Wireless Communications, 20*(6), 91–98. https://doi.org/10.1109/MWC.2013.6704479.

Appendix 2: Selected set of primary studies for smart farming

Adamchuk, V. I., Hummel, J. W., Morgan, M. T., & Upadhyaya, S. K. (2004). On-the-go soil sensors for precision agriculture. *Computers and Electronics in Agriculture*. https://doi.org/10.1016/j.compag.2004.03.002.

Fountas, S., Carli, G., Sørensen, C. G., Tsiropoulos, Z., Cavalaris, C., Vatsanidou, A., … Tisserye, B. (2015). *Farm management information systems: Current situation and future perspectives. Computers and Electronics in Agriculture, 115*, 40–50. https://doi.org/10.1016/j.compag.2015.05.011.

Jawad, H., Nordin, R., Gharghan, S., Jawad, A., & Ismail, M. (2017). Energy-Efficient Wireless Sensor Networks for Precision Agriculture: A Review. *Sensors*, 17(8), 1781. https://doi.org/10.3390/s17081781.

Murakami, E., Saraiva, A. M., Ribeiro, L. C. M., Cugnasca, C. E., Hirakawa, A. R., & Correa, P. L. P. (2007). An infrastructure for the development of distributed service-oriented information systems for precision agriculture. *Computers and Electronics in Agriculture, 58*(1), 37–48. https://doi.org/10.1016/j.compag.2006.12.010.

Nikkilä, R., Seilonen, I., & Koskinen, K. (2010). Software architecture for farm management information systems in precision agriculture. *Computers and Electronics in Agriculture, 70*(2), 328–336. https://doi.org/10.1016/j.compag.2009.08.013.

Rains, Glen, Dan Thomas. (2000). Precision Farming: An Introduction. *University of Georgia, Georgia Extension Bulletin No. 1186*. http://hdl.handle.net/10724/12223. Last accessed 03 February 2018.

Schmitz, M., Martini, D., Kunisch, M., & Mösinger, H. J. (2009). AgroXML enabling standardized, platform-independent internet data exchange in farm management information systems. In *Metadata and Semantics* (pp. 463–468). https://doi.org/10.1007/978-0-387-77745-0_45.

Sørensen, C. G., Fountas, S., Nash, E., Pesonen, L., Bochtis, D., Pedersen, S. M., … Blackmore, S. B. (2010). Conceptual model of a future farm management information system. *Computers and Electronics in Agriculture, 72*(1), 37–47. https://doi.org/10.1016/j.compag.2010.02.003.

Sørensen, C. G., Pesonen, L., Bochtis, D. D., Vougioukas, S. G., & Suomi, P. (2011). Functional requirements for a future farm management information system. *Computers and Electronics in Agriculture, 76*(2), 266–276. https://doi.org/10.1016/j.compag.2011.02.005.

Steinberger, G., Rothmund, M., & Auernhammer, H. (2009). Mobile farm equipment as a data source in an agricultural service architecture. *Computers and Electronics in Agriculture, 65*(2), 238–246. https://doi.org/10.1016/j.compag.2008.10.005.

Verdouw, C., Wolfert, J., and Tekinerdogan, B. (2016). Internet of Things in Agriculture, CAB Reviews: *Perspectives in Agriculture, Veterinary Science, Nutrition and Natural Resources*. 1-12. http://dx.doi.org/10.1079/PAVSNNR201611035.

Zhang, N., Wang, M., & Wang, N. (2002). Precision agriculture - A worldwide overview. *Computers and Electronics in Agriculture, 36*(2–3), 113–132. https://doi.org/10.1016/S0168-1699(02)00096-0.

References

Adamchuk, V. I., Hummel, J. W., Morgan, M. T., & Upadhyaya, S. K. (2004). On-the-go soil sensors for precision agriculture. *Computers and Electronics in Agriculture*. https://doi.org/10.1016/j.compag.2004.03.002.

Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayyash, M. (2015). Internet of things: a survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys and Tutorials, 17*(4), 2347–2376. https://doi.org/10.1109/COMST.2015.2444095.

Angelov, S., Grefen, P., & Greefhorst, D. (2012). A framework for analysis and design of software reference architectures. *Information and Software Technology, 54*(4), 417–431. https://doi.org/10.1016/j.infsof.2011.11.009.

Apel, S., Batory, D., Kästner, C., & Saake, G. (2013). Feature-oriented software product lines. *Feature-Oriented Software Product Lines: Concepts and Implementation*. https://doi.org/10.1007/978-3-642-37521-7.

Aqeel-Ur-Rehman, A. Z. A., Islam, N., & Shaikh, Z. A. (2014). A review of wireless sensors and networks' applications in agriculture. *Computer Standards and Interfaces*. https://doi.org/10.1016/j.csi.2011.03.004.

Bass, L., Clements, P., & Kazman, R. (2012). *Software architecture in practice* (3rd Edn, Vol. 3nd). Architecture.

Beck, Howard. (2001). Agricultural enterprise information management using object Databases, Java, and CORBA. *Computers and Electronics in Agriculture, 32*(2), 119–147. https://doi.org/10.1016/S0168-1699(01)00162-4.

Capilla, R., Bosch, J., Trinidad, P., Ruiz-Cortés, A., & Hinchey, M. (2014). An overview of dynamic software product line architectures and techniques: Observations from research and industry. *Journal of Systems and Software, 91*(1), 3–23. https://doi.org/10.1016/j.jss.2013.12.038.

Chaudhary, S., Sorathia, V., & Laliwala, Z. (2004). Architecture of sensor based agricultural information system for effective planning of farm activities. In *Proceedings—2004 IEEE international conference on services computing, SCC 2004*, (pp. 93–100). https://doi.org/10.1109/SCC.2004.1357994.

Clements, P. (2006). Software product lines. *Software Product Lines, 3714*(3), 1–105. https://doi.org/10.1007/11554844.

Clements, P., Bachmann, F., Bass, L., Garlan, D., Ivers, J., Little, R., et al. (2010). Documenting software architectures. *Style DeKalb IL*. https://doi.org/10.1109/ICSE.2003.1201264.

Cloutier, R., Muller, G., Verma, D., Nilchiani, R., Hole, E., & Bone, M. (2010). The concept of reference architectures. *Systems Engineering, 13*(1), 14–27. https://doi.org/10.1002/sys.20129.

Demirli, E., & Tekinerdogan, B. (2011). Software language engineering of architectural viewpoints. In *Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics)*, (Vol. 6903 LNCS, pp. 336–343). https://doi.org/10.1007/978-3-642-23798-0_36.

Dlodlo, N., & Kalezhi, J. (2015). The internet of things in agriculture for sustainable rural development. In *2015 international conference on emerging trends in networks and computer communications (ETNCC)*, (pp. 13–18). IEEE. https://doi.org/10.1109/ETNCC.2015.7184801.

Fountas, S., Carli, G., Sørensen, C. G., Tsiropoulos, Z., Cavalaris, C., Vatsanidou, A., et al. (2015). Farm management information systems: Current situation and future perspectives. *Computers and Electronics in Agriculture, 115,* 40–50. https://doi.org/10.1016/j.compag.2015.05.011.

Fountas, S., Wulfsohn, D., Blackmore, B. S., Jacobsen, H. L., & Pedersen, S. M. (2006). A model of decision-making and information flows for information-intensive agriculture. *Agricultural Systems, 87*(2), 192–210. https://doi.org/10.1016/j.agsy.2004.12.003.

Gazis, V., Gortz, M., Huber, M., Leonardi, A., Mathioudakis, K., Wiesmaier, A., Zeiger, F., & Vasilomanolakis, E. (2015). A survey of technologies for the internet of things. In *IWCMC 2015—11th international wireless communications and mobile computing conference* (pp. 1090–1095). https://doi.org/10.1109/IWCMC.2015.7289234.

Griffin, T. W., & Lowenberg-DeBoer, J. (2005). Worldwide adoption and profitability of precision agriculture implications for Brazil. *Revista de Política Agrícola, 14*(4), 20–37.

Griffin, T. W., Shockley, J. M., Mark, T. B., Shannon, D. K., Clay, D. E., & Kitchen, N. R. (2018). Economics of precision farming. *Precision Agriculture Basics*. https://doi.org/10.2134/precisionagbasics.2016.0098.

IETF. (2011). "XMPP main". 2011. https://xmpp.org/. 2013. "CoAP." http://coap.technology/.

ITU. (2005). The internet of things. *ITU Internet Report, 2005,* 212. https://doi.org/10.2139/ssrn.2324902.

Jawad, H., Nordin, R., Gharghan, S., Jawad, A., & Ismail, M. (2017). Energy-efficient wireless sensor networks for precision agriculture: A review. *Sensors, 17*(8), 1781. https://doi.org/10.3390/s17081781.

Kaloxylos, A., Eigenmann, R., Teye, F., Politopoulou, Z., Wolfert, S., Shrank, C., et al. (2012). Farm management systems and the future internet era. *Computers and Electronics in Agriculture, 89,* 130–144. https://doi.org/10.1016/j.compag.2012.09.002.

Khan, R., Khan, S.U., Zaheer, R., & Khan, S. (2012). Future internet: The internet of things architecture, possible applications and key challenges. In *2012 10th international conference on frontiers of information technology* (pp. 257–260). IEEE. https://doi.org/10.1109/FIT.2012.53.

Kitchenham, B. A., & Pfleeger, S. L. (2002). Principles of survey research part 3: Constructing a survey instrument. *ACM SIGSOFT Software Engineering Notes, 27*(2), 20. https://doi.org/10.1145/511152.511155.

Köksal, Ö., & Tekinerdogan, B. (2017). Feature-driven domain analysis of session layer protocols of internet of things. In *Proceedings—2017 IEEE 2nd international congress on internet of things, ICIOT 2017* (pp. 105–112). https://doi.org/10.1109/IEEE.ICIOT.2017.19.

Kruize, J. W., Wolfert, J., Scholten, H., Verdouw, C. N., Kassahun, A., & Beulens, A. J. M. (2016). A reference architecture for farm software ecosystems. *Computers and Electronics in Agriculture, 125,* 12–28. https://doi.org/10.1016/J.COMPAG.2016.04.011.

Kutter, T., Tiemann, S., Siebert, R., & Fountas, S. (2011). The role of communication and co-operation in the adoption of precision farming. *Precision Agriculture, 12*(1), 2–17. https://doi.org/10.1007/s11119-009-9150-0.

Linseisen, H. (2001). Development of a precision farming information system. In *Proceedings of the third European conference on precision agriculture* (pp. 689–694). http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.460.5744.

Lowenberg-DeBoer, J., Erickson, K., & Vogel, K. A. (2000). *Precision farming profitability*. West Lafayette: Agricultural Research Programs, Purdue University.

Ma, J., Zhou, X., Li, S., & Li, Z. (2011). Connecting agriculture to the internet of things through sensor networks. In *2011 international conference on internet of things and 4th international conference on cyber, physical and social computing* (pp. 184–187). IEEE. https://doi.org/10.1109/iThings/CPSCom.2011.32.

McEwen, A., & Cassimally, H. (2014). *Designing the internet of things*. Wiley. https://www.wiley.com/en-tr/Designing+the+Internet+of+Things-p-9781118430620.

Murakami, E., Saraiva, A. M., Ribeiro, L. C. M., Cugnasca, C. E., Hirakawa, A. R., & Correa, P. L. P. (2007). An infrastructure for the development of distributed service-oriented information systems for precision agriculture. *Computers and Electronics in Agriculture, 58*(1), 37–48. https://doi.org/10.1016/j.compag.2006.12.010.

Nikkilä, R., Seilonen, I., & Koskinen, K. (2010). Software architecture for farm management information systems in precision agriculture. *Computers and Electronics in Agriculture, 70*(2), 328–336. https://doi.org/10.1016/j.compag.2009.08.013.

OASIS. (2011). "AMQP." 2011. http://www.amqp.org/specification/1.0/amqp-org-download. 2014. "MQTT." 2014. http://mqtt.org/2014/11/mqtt-v3-1-1-now-an-oasis-standard.

OMG. (2015). *DDS Specification V 1.4* (p. 180). http://www.omg.org/spec/DDS/1.4/.

Palattella, M. R., Accettura, N., Vilajosana, X., Watteyne, T., Grieco, L. A., Boggia, G., et al. (2013). Standardized protocol stack for the internet of (important) things. *IEEE Communications Surveys and Tutorials, 15*(3), 1389–1406. https://doi.org/10.1109/SURV.2012.111412.00158.

Pandya, H.B., & Champaneria, T.A. (2015). Internet of things: Survey and case studies. *2015 international conference on electrical, electronics, signals, communication and optimization (EESCO)* (pp. 1–6). https://doi.org/10.1109/EESCO.2015.7253713.

Rains, G. C., & Thomas, D. L. (2009). *Precision farming: An introduction*. Griffin: University of Georgia.

Runeson, P., & Höst, M. (2008). Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*. https://doi.org/10.1007/s10664-008-9102-8.

Schimmelpfennig, D. (2016). Farm profits and adoption of precision agriculture. https://www.ers.usda.gov/webdocs/publications/80326/err-217.pdf?v=42661.

Schmitz, M., Martini, D., Kunisch, M., & Mösinger, H. J. (2009). AgroXML enabling standardized, platform-independent internet data exchange in farm management information systems. In *Metadata and semantics* (pp. 463–468). https://doi.org/10.1007/978-0-387-77745-0_45.

Seelan, S. K., Laguette, S., Casady, G. M., & Seielstad, G. A. (2003). Remote sensing applications for precision agriculture: A learning community approach. *Remote Sensing of Environment*. https://doi.org/10.1016/j.rse.2003.04.007.

Sheng, Z., Yang, S., Yifan, Y., Vasilakos, A., McCann, J., & Leung, K. (2013). A survey on the IETF protocol suite for the internet of things: standards, challenges, and opportunities. *IEEE Wireless Communications, 20*(6), 91–98. https://doi.org/10.1109/MWC.2013.6704479.

Sørensen, C. G., Fountas, S., Nash, E., Pesonen, L., Bochtis, D., Pedersen, S. M., et al. (2010). Conceptual model of a future farm management information system. *Computers and Electronics in Agriculture, 72*(1), 37–47. https://doi.org/10.1016/j.compag.2010.02.003.

Sørensen, C. G., Pesonen, L., Bochtis, D. D., Vougioukas, S. G., & Suomi, P. (2011). Functional requirements for a future farm management information system. *Computers and Electronics in Agriculture, 76*(2), 266–276. https://doi.org/10.1016/j.compag.2011.02.005.

Steinberger, G., Rothmund, M., & Auernhammer, H. (2009). Mobile farm equipment as a data source in an agricultural service architecture. *Computers and Electronics in Agriculture, 65*(2), 238–246. https://doi.org/10.1016/j.compag.2008.10.005.

Tekinerdogan, B. (2014). Software architecture. In T. Gonzalez, J. Diaz-Herrera, & A. Tucker (Eds.), *Computing handbook: Computer science and software engineering* (p. 2280). London: Chapman and Hall/CRC.

Tekinerdogan, B., Sozer, H., & Aksit, M. (2012). Feature-based rationale management system for supporting software architecture adaptation. *International Journal of Software Engineering and Knowledge Engineering, 22*(7), 945–964. https://doi.org/10.1142/S021819401250026X.

Turkish Land Crop Office. (2017). Turkish grain report 2016. http://www.tmo.gov.tr/Upload/Document/hububat/hububatraporu2016.pdf.

Turkish Ministry of Agriculture. (2018). Herbal production data. http://www.tarim.gov.tr/sgb/Belgeler/SagMenuVeriler/BUGEM.pdf.

Tüzün, E., Tekinerdogan, B., Kalender, M. E., & Bilgen, S. (2015). Empirical evaluation of a decision support model for adopting software product line engineering. *Information and Software Technology, 60,* 77–101. https://doi.org/10.1016/j.infsof.2014.12.007.

Venters, C. C., Capilla, R., Betz, S., Penzenstadler, B., Crick, T., Crouch, S., et al. (2018). Software sustainability: Research and practice from a software architecture viewpoint. *Journal of Systems and Software, 138*(April), 174–188. https://doi.org/10.1016/J.JSS.2017.12.026.

Verdouw, C., Wolfert, J., & Tekinerdogan, B. (2016). Internet of things in agriculture. *CAB Reviews: Perspectives in Agriculture, Veterinary Science, Nutrition and Natural Resources*. https://doi.org/10.1079/PAVSNNR201611035.

Wolfert, S., Ge, L., Verdouw, C., & Bogaardt, M.-J. (2017). Big data in smart farming—A review. *Agricultural Systems, 153*(May), 69–80. https://doi.org/10.1016/J.AGSY.2017.01.023.

Wolfert, J., Verdouw, C. N., Verloop, C. M., & Beulens, A. J. M. (2010). Organizing information integration in agri-food-A method based on a service-oriented architecture and living lab approach. *Computers and Electronics in Agriculture, 70*(2), 389–405. https://doi.org/10.1016/j.compag.2009.07.015.

Yin, R.K. (2009). Case study research : Design and methods/Robert K. Yin. *Applied Social Research Methods Series* (p. 5). https://doi.org/10.1097/FCH.0b013e31822dda9e.

Zhang, N., Wang, M., & Wang, N. (2002). Precision agriculture—A worldwide overview. *Computers and Electronics in Agriculture, 36*(2–3), 113–132. https://doi.org/10.1016/S0168-1699(02)00096-0.