Backman, Juha; Linkolehto, Raimo; Koistinen, Markku; Nikander, Jussi; Ronkainen, Ari; Kaivosoja, Jere; Suomi, Pasi; Pesonen, Liisa

Cropinfra research data collection platform for ISO 11783 compatible and retrofit farm equipment

Published in:
Computers and Electronics in Agriculture

DOI:
10.1016/j.compag.2019.105008

Published: 01/11/2019

Document Version
Publisher's PDF, also known as Version of record

Computers and Electronics in Agriculture 166 (2019) 105008

Contents lists available at ScienceDirect

Computers and Electronics in Agriculture

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

Cropinfra research data collection platform for ISO 11783 compatible and retrofit farm equipment

Juha Backman[a,*], Raimo Linkolehto[a], Markku Koistinen[a], Jussi Nikander[b], Ari Ronkainen[a], Jere Kaivosoja[a], Pasi Suomi[a], Liisa Pesonen[a]

[a] Natural Resources Institute Finland (Luke), Production Systems, Tietotie 2, 02150 Espoo, Finland
[b] Aalto University, School of Engineering, Department of Built Environment, Otakaari 4, Espoo, Finland

ARTICLE INFO

Keywords:
ISOBUS
Document database
Development platform
Data capture
Data transfer

ABSTRACT

The agricultural machinery produces an increasing number of measurements during operations. The primary use of these measurements is to control agricultural operations on the farm. Data that describes the in-field variation in plant growth potential and growing conditions is the basis for precision farming. The secondary use for the gathered information is documentation of work and work performance for business purposes. Researcher also benefits from the increasing measurement capabilities. Biologists and agronomists can model the crops and agronomic phenomena. Work scientists can analyse the agricultural work processes. And finally, machines with additional accurate sensors can be used for agricultural machine product development and technological research purposes.

This paper concentrates on an independent research data collection platform (Cropinfra) which can be used to collect data for all above mentioned purposes. Data can be collected both from ISOBUS (ISO 11783) compliant machines as well as older and proprietary systems and stored to database for further analysis. The farm machines in Cropinfra are supplemented with extra sensors that are more accurate than existing in commercial machines. Therefore, the Cropinfra can be used as a reference measurement system to verify the correct operation of the machines as well as to produce data for biological research purposes. This paper will also present how the cloud connection of the data collection system can be realized. The solution was designed to be compatible with the existing ISO 11783-10 standard. The examples presented in this paper verify that the solution works in real farming environment. The data has been used in numerous research projects already, and in the future the data will be an important asset when machine learning and other artificial intelligence methods will be studied and utilized.

1. Introduction

Agricultural machinery has gone through significant changes during last decades. The introduction of electronic control units (ECU) has opened up new ways for carrying out agricultural operations (Stone et al., 2008). Especially the GNSS (Global Navigation Satellite System) based solutions enable position based precision farming (Thomasson et al., 2019). At the same time the sensors and other measurement devices have also developed tremendously and their unit price has come down (SACHS, 2014). Agricultural machinery produces an increasing number of measurements during operations. This information can be used for many purposes. One primary purpose is to use data gathered to control agricultural operations. Soil conditions, crop growth and health can be monitored and used to make fertilization and plant protection decisions and prescription maps for precision farming operations (Söderström et al., 2016). Another use for the information is documentation of work and work performance for business purposes. For example, an agricultural contractor can verify the work they have carried out by showing the measurement data to the customer (Suonentieto, 2019; Dataväxt, 2019). In addition to the land owners and farmers, researchers can also benefit from the increasing measurement capabilities. Biologists and agronomists can model crops and agronomic phenomena, verify hypotheses and produce new knowledge (Kaivosoja et al., 2013). Machinery data is an important source of so called big data in agriculture (Wolfert et al., 2017). Also work scientists can analyse the agricultural work processes in more detail. And finally, machines with additional and accurate sensors can be used for Product development of agricultural machinery and technological research

* Corresponding author.
E-mail address: juha.backman@luke.fi (J. Backman).

https://doi.org/10.1016/j.compag.2019.105008
Received 21 November 2018; Received in revised form 30 July 2019; Accepted 12 September 2019

purposes (Öhman et al., 2004; Suomi and Oksanen, 2015)

Data can be collected from the machinery using a number of different ways. The existing electronic control units (ECUs) can create their own proprietary logs, which can be downloaded to USB stick or, in some cases, to cloud servers. Another way is to use an independent logging unit which does the same, but is not directly bundled with any manufacturer's control units. In both cases, the data collection includes four phases: measurement devices or sensors, data capture, data transfer, and data storage. However, the last two phases of the process can be fundamentally different depending on the technology used.

Farms can be very heterogeneous, as the history of the farm, the geography of the land used by the farm, the crops and livestock grown on the farm, as well as technological capability and business strategy of the farm all affect how the farm works. The type of production and the size of the farm define the type and size of machinery required. Furthermore, farm's machinery is often a mixture of older and newer machines from a number of manufacturers. Rarely any manufacturer can provide all machines, devices and software that the farmer needs. Therefore there is a strong need for compatibility between different brands. In agriculture the communication between different components in tractor-implement system is defined by the commonly agreed standard ISO 11783 (ISO, 2017) – marketed under the name ISOBUS (AEF, 2019a). The Agricultural Electronic Industry Foundation (AEF) develops and markets the ISOBUS system.

In the previous research projects the information management of farm machinery and the data transfer from the machinery to farm management information systems (FMIS) and various on– and off-site services has been studied. Fountas et al. (2015) have proposed a Farm Machinery Management Information System based on interviews of tractor operators and farm managers. The solution was presented as a rich picture from which a conceptual model for conventional farm machinery and agricultural robots was developed. Steinberger et al. (2009) have presented a prototype implementation of ISO 11783 task controller without user interface for data logging. The data was stored in ISO XML format in a PDA and transferred to a local storage using WLAN. For telemetry purposes feasibility of OPC Unified Architecture (OPC UA) technology to transfer ISO 11783 related process data between farm machinery and the internet was studied by Oksanen et al. (2015). Oksanen et al. presented also an approach to convert ISO 11783 DDOP (Device Description Object Pool) to OPC UA information model. In the experiments the latency and bandwidth usage of communication were measured and found feasible.

Commercial solutions for data collection exist from all major farm machinery manufacturers. For example AGCO has AgCommand, John Deere has JDLink, CLAAS has TONI, etc. There exist also retrofittable fleet management and telemetry solutions from different manufacturers, for example Suonentieto AgriSmart, Wapice IoT-Ticket, etc. Currently the ISOBUS working group at AEF is working on a standard for data transfer between ISOBUS-compatible machinery and farm management systems (AEF, 2019b). The standard is expected to be ready for use during the year 2019.

This paper concentrates on an independent research data collection platform (Cropinfra) which can be used to collect data both form ISOBUS compliant machines as well as old and proprietary systems and store the data to database for further analysis. The farm machines in Cropinfra are supplemented with extra sensors that are more accurate than those existing in commercial machines. Therefore the Cropinfra can be used as a reference measurement system to verify the correct operation of the machine. The Cropinfra has developed in many national and international research projects starting from 2003 (Fig. 1). The Cropinfra platform is a concept for future farm's data collection and management structure that is capable to serve also research purposes, and which has been verified by implementing it in a real scale experimental farm for technology development. The research farm was located in Southern Finland, in Vihti. It included 151 ha of field, all necessary buildings for farm activities, four tractors, a combine harvester and all necessary implements to operate the farm. The results of Cropinfra based research projects have recently been reported in Nikander et al. (2017), Nikander et al. (2015) and Pesonen et al. (2014).

In this paper we report the methods and means used to develop Cropinfra into a framework that can be used as a general agricultural data collection and exploitation system for research purposes. The primary objective of this paper is *to depict how the existing technology on agricultural data collection can be integrated into an open research system*. A secondary objective is to *present the experiences and results of different ways to realize the cloud connection of data collection system* that is at the moment crucial improvement to be included in the ISOBUS standard.

The rest of this paper is organized as follows. In Section 2, methods, we describe how the information management and software environment in a modern farm can be organized, focusing on the parts that are vital to this paper. In Section 3 we describe how the CropInfra platform implements farm information management infrastructure, and how the system has been used for data capturing and management using examples. Section 4 contains the discussion and conclusions of our work.

2. Methods

This work is based on data collection from agricultural machinery, as well as the analysis, and exploitation of that data both in cloud systems and in the machinery. In this work, we concentrate on the four phases of data collection: *measurement*, *capture*, *transfer*, and *storage*. In the *measurement* phase, the various sensors take readings and provide values that can be stored. In the *capturing* phase, the sensor values are combined and given semantics by assigning each value to a specific variable. In the *transfer* phase the data created in the capturing phase is transferred from the field machine to a cloud service. Finally, in the *storage* phase the data is stored in a cloud service for further analysis and exploitation.

Data collection from tractors, implements, and other agricultural machinery is primarily based on previous development carried out for digital systems in heavy machinery and vehicles in general. These systems have then been adapted to use in agriculture, and extended in order to best serve the specific needs of the agricultural sector. The cloud systems and other software used outside the machinery mostly use standard data exchange methods. The data formats and analysis methods used are, of course, agriculture-specific in order to both capture the nature of the data being used as well as providing useful information to act as basis for managing farm, i.e. planning and executing specific agricultural operations.

2.1. Farm information management

The digital environment of a modern farm can already be extremely complex and is becoming increasingly so. Fig. 2 shows an example of a farm digital environment, in which the parts that are relevant to this study are emphasized. The Figure has been adapted from one originally published in Nikander et al. (2015). The elements of the Figure that are out of scope for this work are shown in order to provide context for the reader how the data transfer in this work would fit into the overall software environment of a future farm. In the bottom of the Fig. 2 are depicted the *measurement* and *capturing* phases of data collection that are carried out in the tractor-implement. The *transfer* phase is represented by the arrow between the tractor-implement and the cloud, and the *storage* phase is in the cloud environment at the top of the Figure. As can be seen in Fig. 2, the data transfer infrastructure described above is only small part of the overall digital environment of a modern farm. The rest of the farm digital environment is described in the original publication in Nikander et al. (2015).

2.2. Tractor and implement data

Currently, data management and transfer in tractors and

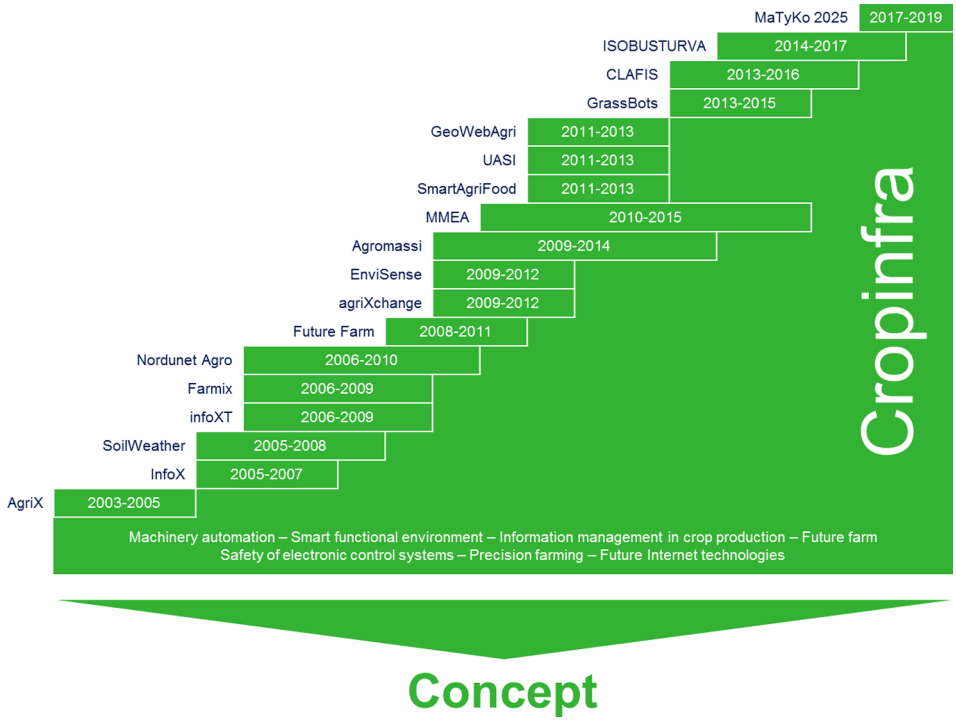

Fig. 1. The Cropinfra concept and research platform has developed in many research projects during 15 years.

implements are, in practice, based on one of three methods. The first is the *ISO 11783 (ISOBUS) standard* (ISO, 2017a,b), the second are *proprietary systems*, typically used in older machinery that do not conform to ISOBUS, and the third are *analog systems*, where the data first needs to be converted to a digital signal in order to be transferred for permanent storage.

ISOBUS defines the communication protocol used between different components of the tractor-implement system. Data transfer is in specified form, and therefore the data created by an ISOBUS system can easily be read and understood by any interface conforming to the standard. The majority of machine-related data (such as implement status, engine performance, or speed) are covered by the standard. There is also a part where manufacturers can add data not currently included in ISOBUS. Oksanen et al. (2005) has published an informative

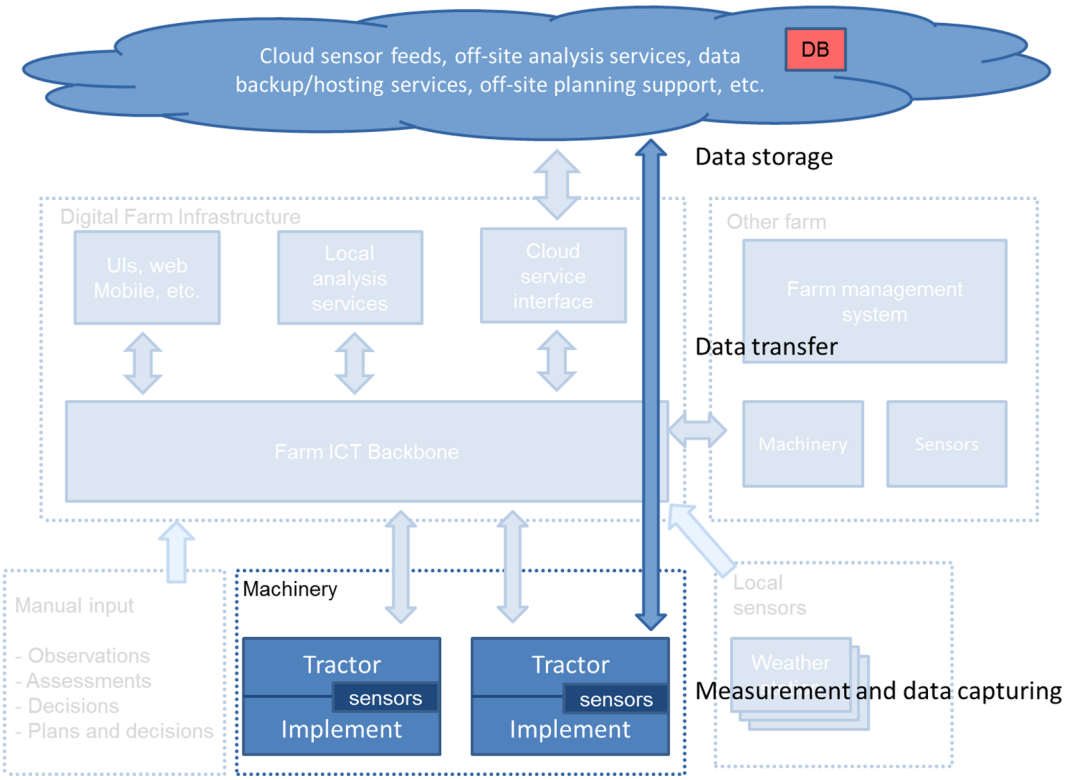

Fig. 2. The software environment of a modern farm, adapted from Nikander et al. (2015).

article about the standard in its use.

In addition, ISOBUS defines a set of functionalities and devices required from the machinery. The simplest ISOBUS system consists of a tractor ECU (T-ECU), an implement ECU (I-ECU) and a universal user interface called universal terminal (UT) or virtual terminal (VT) (AEF, 2019b). These devices form the basic structure used to control the tractor and implement combination. There can be additional devices attached to the system, such as a positioning device (GNSS), or a task controller (TC). The task controller can be used to control the implement, as well as to store the data logged in an executed field operation (ISO, 2015). A TC that is capable of location-specific control and logging is called TC-GEO and a TC that is capable only of data logging is called TC-LOG.

Not all the machines in the agriculture are compatible with the ISOBUS standard. There are also numerous proprietary solutions for implement control. There is no single solution for capturing data from these systems. Instead, each system needs an interface to adapt the data provided by the proprietary system to the farm software infrastructure. Should the data be used to control the device, the conversion needs to be done the other way around. However, it can be difficult to use farm data as input into a proprietary system unless the control software of the system has an open input method.

In the systems where digital controllers are not used, signals between controls, sensors and actuators use only voltage or current signals. In this case, the system needs a setup similar to proprietary systems, where the signal is first converted to a digital form, and then adapted to the needs of the larger farm infrastructure. As analog systems seldom are able to take input except through the analog controllers, it can be difficult to use data to control such a system.

2.3. Back-end technologies in cloud interfaces

Although the ISO 11783 standard will eventually cover the data transfer from the tractor to cloud services (AEF, 2019b), these technologies have already been used elsewhere. In essence, there are two different technologies needed: a data transfer method and a data format. Possible data transfer technologies are, for example, REST and MQTT. JSON and Protobuf are potential candidates for data format.

2.3.1. REST

REST (Representational State Transfer) is an architectural style, not a standard, that describes a set of constrains over the HTTP (Leach, 1999a) application protocol for designing distributed client/server related information systems that provide interoperability between computer systems on the Internet. Web services following the REST architectural style are called as RESTful web services. The objects provided by the RESTful web services (sometimes referred as RESTful API) are called resources. The resources are typically identified by using URIs. The role of the application server is to provide access to resources and the client accesses, and in some cases it modifies the resources (Fielding, 2000). A client uses HTTP methods for reading a resource (GET), for creating a new resource (POST), to update an existing resource or create a new resource (PUT) and for removing a resource (DELETE) (IETF Trust, 2007).

2.3.2. MQTT

MQTT (Message Queuing Telemetry Transport) is a simple and lightweight publish/subscribe-based messaging protocol running over TCP/IP, or other similar protocols like Bluetooth (ISO/IEC 20922, 2016). MQTT is designed for devices in Machine-to-Machine communication and IoT context with limited capabilities and networks with low-bandwidth, high-latency and possibly unreliable connectivity. In attempt to assure some degree of delivery in constrained environments the standard describes three qualities of service: *at most once,* where occasional message loss is allowed, *at least once,* where message delivery is guaranteed but duplicates can occur and *exactly once,* where messages are assured to arrive and only once.

The underlying publish/subscribe messaging paradigm means that the system requires a message broker. The broker operates between publishers and subscribers taking incoming messages from data provider and routing them to relevant destinations. The information (published data) is organized in a hierarchy of topics and the broker distributes the information to subscribers according to the subscribed topics. Public/subscribe model implements decoupling: on the general level publishers and subscribers have no need to know anything about each other's systems (ISO/IEC 20922, 2016).

2.3.3. JSON

JSON (JavaScript Object Notation) is a programming language independent, open-standard data-interchange file format for structuring and sharing data. It is based on subset of the JavaScript programming language and it can represent any serializable data. The format can be generated and parsed programmatically, and is human-readable at the same time. JSON is used as a data exchange format in asynchronous web applications and as a natural data exchange format in interfaces that are bound to document oriented database management system (Bray, 2015, 2017).

2.3.4. PROTOBUF

Protocol buffers offer a programming langue neutral and platform-neutral mechanism for serializing structured data as well as an interface description language in the inter-machine communication domain (Google, 2018). A data structure in Protobuf is defined using three types of variables: *required fields*, *repeated fields*, and *optional fields*. Data is compiled into data access classes for automated reading and writing for a variety of data streams. For data transfer the Protobuf text format messages are encoded into a binary format, which reduces the message size up to 40% and speeds up the parsing time up to 50 times when compared to the size and processing time of a text based format like XML. Protocol buffers are well suited to the environments with time critical computing requirements. (Google, 2018)

3. Results

Fig. 3 depicts the overall structure and the different devices and services of the Cropinfra research data collection system on a conceptual level. At the bottom of the picture there is the *measurement* level, which contains the physical sensors that are either commercial ones, or reference sensors added for research purposes. Sensors are connected to ECUs on the *data capturing* level that transform the analog signals and transmit those to ISOBUS communication (CAN) bus. Above the ISOBUS CAN in Fig. 3, there are control and logging devices that capture, utilize and store the measurements locally. Finally, at the top of the Figure are the data *transfer* and *storage* means. The data itself is stored either directly in the file system, or in a database. The different phases of the data collection process are explained in further detail in the following sections. In the following, we will first discuss how legacy machines that provide only analog readings need to be modified in order to fit into the Cropinfra framework. Then we will discuss data capturing, data transfer in both bulk and real-time modes, and data storage. After that, we will discuss the data capturing and transfer results using examples where the Cropinfra framework has been successfully used.

3.1. Measuring legacy machines and using reference measurements

To support all different implements in Cropinfra, analog sensors and I/O device (Axiomatic, 2008) are used to provide measurements from non-ISOBUS implements to the CAN-bus. The I/O device sends the machine identifier that is the same as the globally unique NAME-identifier used in ISOBUS (ISO, 2019). The identification and the measurements messages from I/O device can be read similarly as from

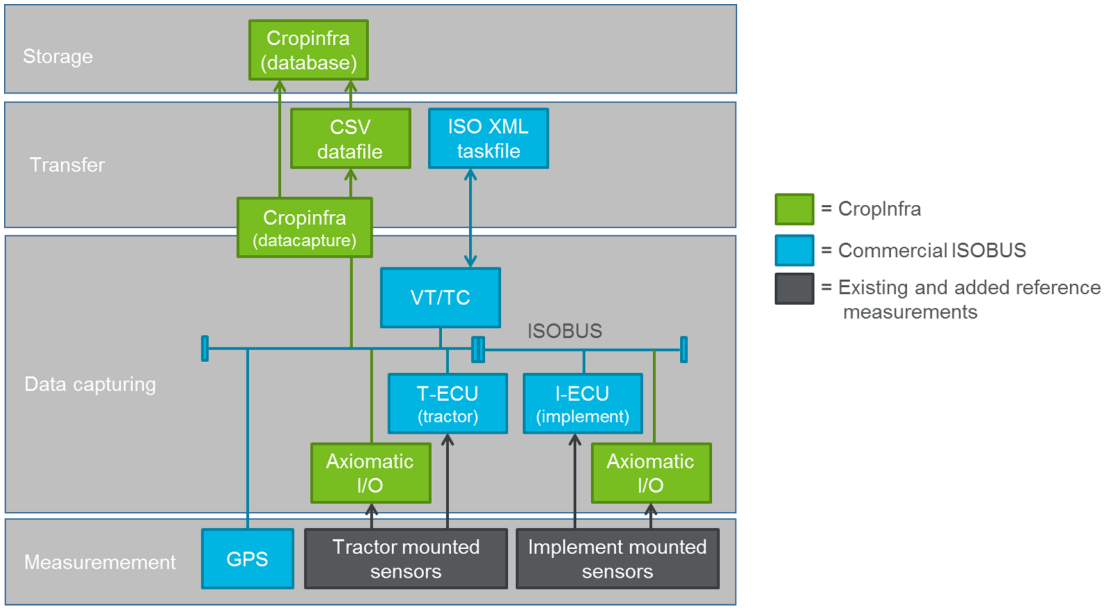

Fig. 3. The structure of the data collection system.

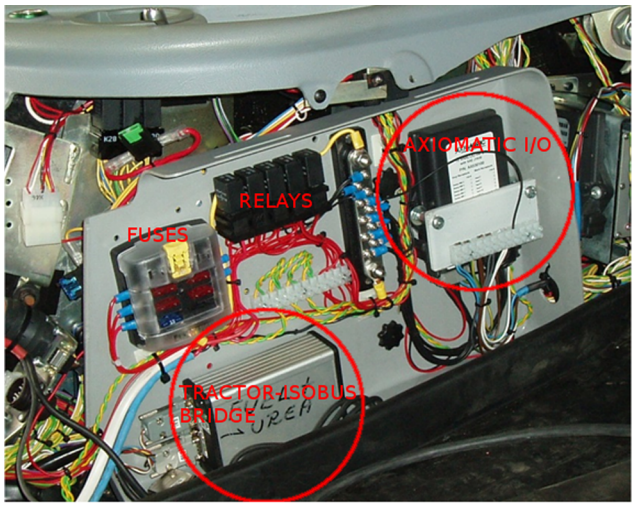

Fig. 4. I/O module and additional electronic wirings are mounted inside the side panel of the cabin.

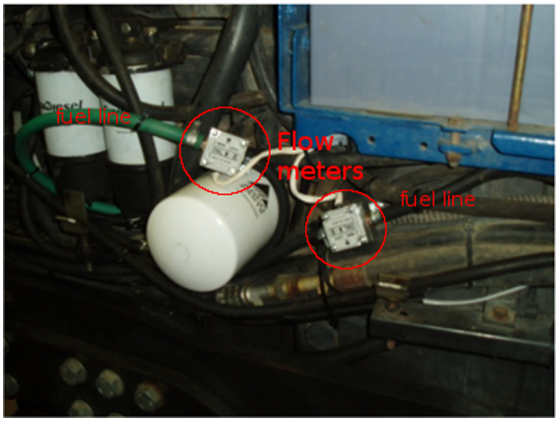

Fig. 5. The fuel consumption has to be measured with external sensors in old tractors.

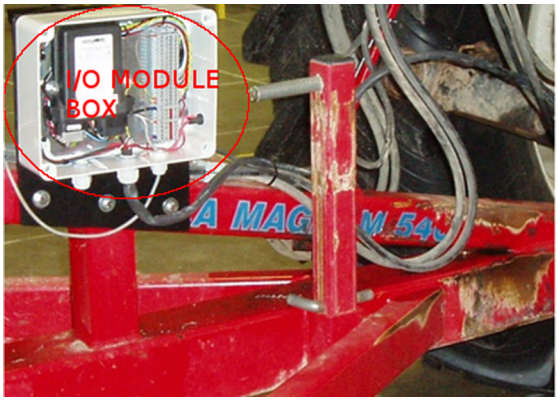

Fig. 6. In figure I/O module fitted to Potila Magnum 540 harrow.

the ISOBUS machines. In another words, the old and proprietary tractors and implements are modified to provide limited ISOBUS-like functionality (see Fig. 4).

For example Valtra 8950 tractor measurements were transmitted to ISOBUS using the I/O device which measured the signals from ISO 11786 signal connector. The ISO 11786 defines a signal connector where the ground speed, linkage position and PTO speed are provided using pulses and analog signals. The fuel consumption was measured using two FLOWMATE OVAL M-III (Oval, 2016) flowmeters (Fig. 5).

The implements without proprietary ECU were also equipped with Axiomatic I/O device. For example the Potila Magnum 540 harrow does not have any electric control (see Fig. 6). All the functions are directly hydraulically controlled. The I/O device was used to measure whether the implement is in working position or not. In other implements the I/O device was also used to provide reference measurements to validate correct operation of the commercial controllers.

3.2. Data capturing

As explained in Section 2.2, the data can be captured from the ISOBUS CAN-bus using standard ISOBUS TC-GEO or TC-LOG device. However, if such device is used for research purposes, it restricts the use of a parallel device with the same functionality. By the TC protocol definition in the standard, the implement can communicate with only one TC at a time. These kinds of conflicting situations may occur when technological research projects are conducted and ISOBUS prototype

devices are tested. Thus, a data capturing device that only listen the other ECUs and doesn't send anything to the can-bus was developed.

Furthermore, the requirements for data capturing in different research projects and in different machines can vary. To meet the requirements, the data capturing software had to modify in different research project. The National Instruments LabVIEW system engineering software (National Instruments, 2018) was selected to be used to program the data capturing software. LabVIEW based software are fast to program, flexible and self-documenting.

The data capturing software was designed to be user friendly and reliable. First, the program identifies all the devices in the ISOBUS network using the address claim functionality (ISO, 2019). All the devices have to claim their own address in the bus using globally unique ISOBUS NAME. The data capturing program is able to identify the machines that are connected based on this NAME. Next, the user is asked to complete the rest of the information before the data capturing is started. The user interface is described more detailed in the next subsection. When all the initial information is completed and user has started the data capturing, the program stores all essential data from the ISOBUS network. The data capturing system does not send any request message to the can bus or use the task controller protocol to communicate with the implement. However, the software stores the process data messages between the TC and the Implement as well as all other messages that are in the bus.

3.2.1. User interface

The user interface of the data capturing software is shown in Fig. 7 and in Fig. 8. The first picture is from the initial data input tab, which is used at the beginning of the work. The second picture is from the driving tab, which is used during the work.

As described above, the data capturing software automatically detects the machines that are connected in the bus. Those are set to be default values in the dropdown boxes in Fig. 7 ("Traktori" [tractor in English] and "Työkone" [implement]). The user can also manually select the machinery. In addition, the user has to select the correct worker ("Suorittaja"), operation ("Työtehtävä"), parcel ("Lohko"), plant type ("Viljelykasvi") and variety ("Siemenlajike") and also the targeted seed ("Siemenmäärä") and fertilization rate ("Lannoitemäärä") as well as corresponding calibration values ("Kiertokoe"). In the bottom of the page, there are also place for free comment ("kommentti") that will be added in the log file. The comment can be added any time during the operation. The user interface is slightly different when different types of implements are used. For example when capturing data from a ploughing task, there is no need to select the plant or rate values and in plant protection tasks there is a selection menu for pesticides instead of seeds and fertilizers.

Fig. 8 shows the tab used during a field operation when data capturing from the sensors has been started. During the work, the user interface is used to monitor that all ECUs in the ISOBUS network are working and sensors produce the correct data. For example, the GPS-box and TC-box are green if those produce data to the bus, otherwise they are red. The current field parcel based on the GPS-location is also show in the text box to verify that the worker is in a correct parcel. The tractor and implement measurements are show as meters for visual verification of correct measurement values.

3.2.2. Hardware for data capturing

The data capturing software was made using the National Instruments LabVIEW system engineering software, so the executable-version of the software could be run in any device that has a LabVIEW runtime. In Cropinfra, Panasonic Toughbook CF-19 rugged laptop computer was used. The system was used in tractors and in a combine harvester. A docking station was mounted in each machine to which power supply, CAN adapter and 3G/4G router (ASUS 4G-N12) was connected. The basic setup of the hardware is shown in Figs. 9 and 10. The initial device setup was selected at the beginning of the Cropinfra-

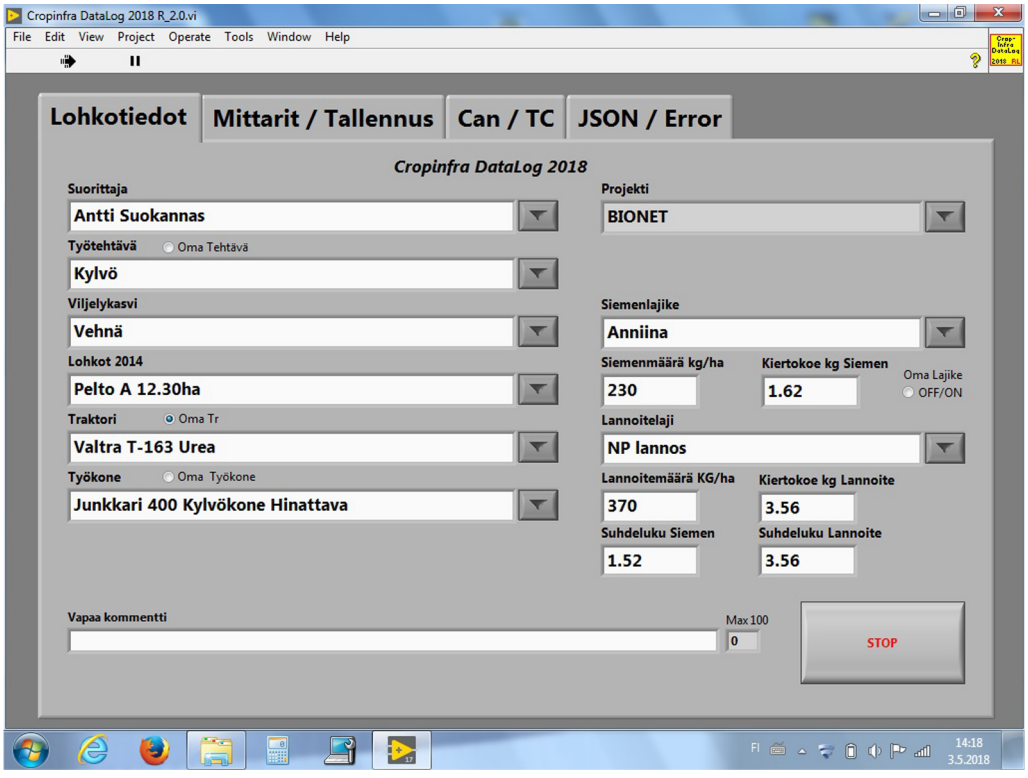

Fig. 7. Data capturing user interface at the beginning of the work.

Fig. 8. Data capturing user interface during the work.

Fig. 9. Rugged computer for data capturing is mounted in the tractor.

related projects in 2003. However, additional devices were used in different research projects.

3.3. Bulk data transfer

The Cropinfra data capturing software stores all the collected information internally in a csv-type text file. Another option would have been to use the ISO 11783-10 XML format. The example of the structure of the data file is in the attachments (Table 3). The data file consists of two parts: *header* and *data*. The *header* stores the information that describes the agricultural operation and is constant during the process (initial data input tab in user interface). In the *data* section, the process data produced by the machinery and sensors are stored using 5 Hz

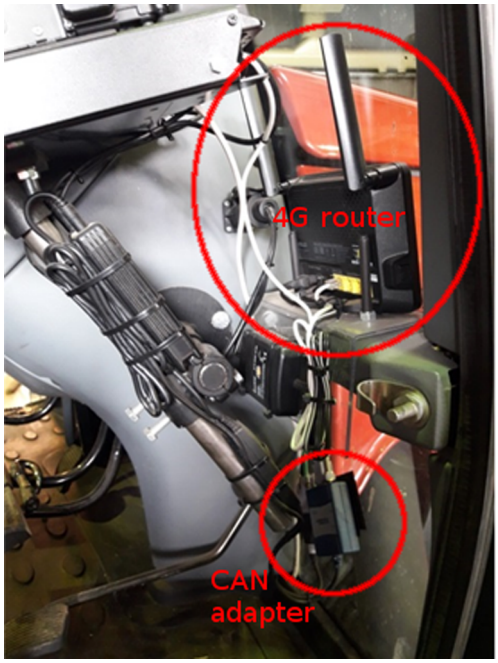

Fig. 10. Wireless 4G router is used to cloud connection and national instruments CAN-adapter for ISOBUS connection.

logging frequency. The file is created when the user starts the data capturing and the data is flushed to the disk in every iteration cycle to ensure that there will not be any loss of data even if the system crashes

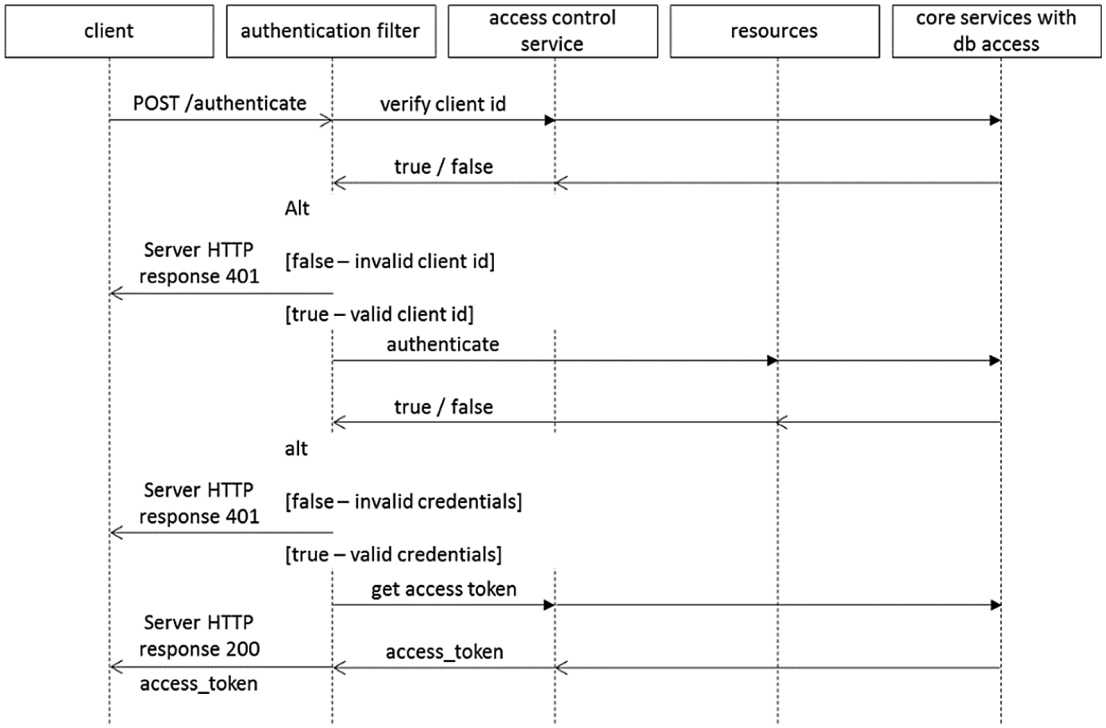

Fig. 11. Authentication process.

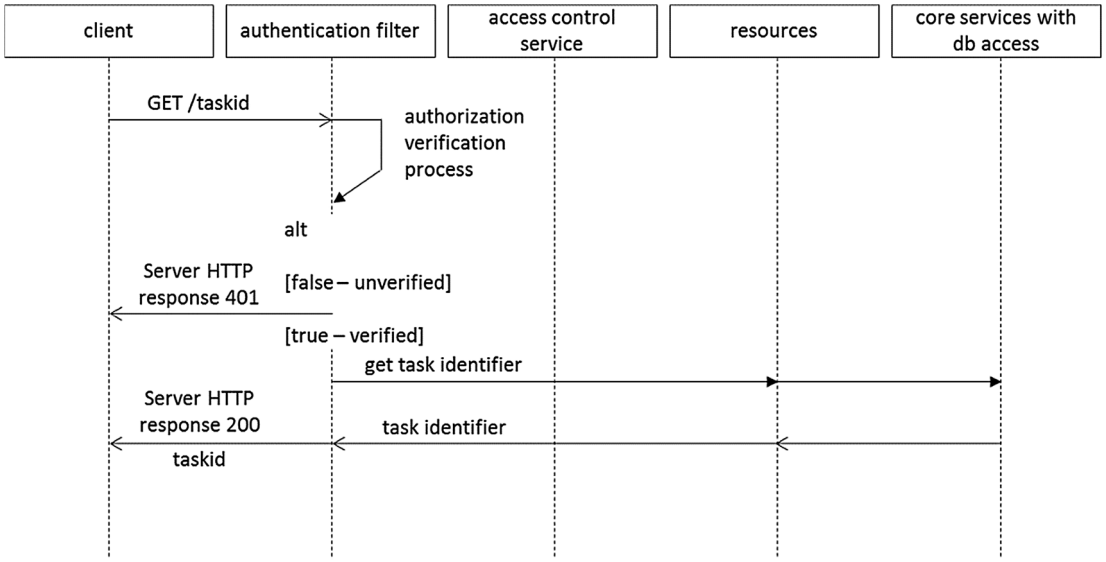

Fig. 12. Getting a task identifier.

during a field operation.

The final log file is transferred to the database either manually using an USB-stick or automatically using FTP. When using FTP, a file transfer daemon monitors the local folder that is used to store the log files. If the daemon notices that a new log file is created or data capturing software has been closed, it is concluded that previous log file is completed and the transfer of the previous log file can be started. The file transfer daemon keeps track of all the log files that are created and transferred. If the network is disconnected or the computer shutdown, the data transfer is tried again after the computer is started and connection resumed.

3.4. Real-time data transfer

The Real-time data transfer is integrated to the data capturing software. Unlike the bulk data transfer, the real-time data transfer attempts to follow the ISO 11783-10/11 standard. The ISO 11783-10 XML schema for the data is used in the system (ISO, 2015).

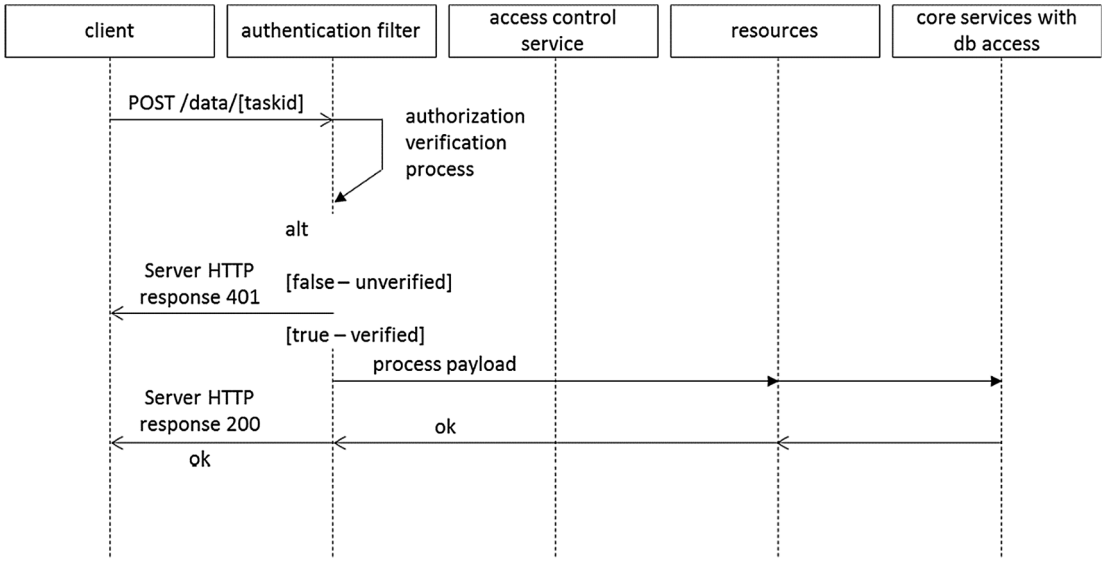

Fig. 13. Posting data.

Table 1
Examples of TSK and TLG documents in a collection.

A TSK document	{ "_id" : { "$oid" : "591557ec13f3c30cd0aa9f13"} , "system taskid" : "388fecaf-de0e-498f-8d2a-13541d4751dc" , "system datetime" : "2017-05-12 09:36:28" , "system document type" : "TSK" , "F" : "WRK99" , "E" : "PFD4" , "D" : "FRM1" , "G" : "2" , "TIM" : { "A" : " 12.5.2017 9:36:38.718"} , "OTP" : { "A" : "CPC1" , "B" : "OTQ1"} , "CTP" : { "A" : "CTP1" , "B" : " Vehnä" , "CVT" : { "A" : "CVT1" , "B" : " "}} , "PDT" : { "A" : "PDT1" , "B" : " "} , "DAN" : [{ "A" : "A00019000CC00000" , "C" : "DVC1"} , { "A" : "A00884002C400001" , "C" : "DVC2"}]}
A TLG document	{ "_id" : { "$oid" : "58ddf39a13f3c311809bffca"} , "system taskid" : "2c92c49e-9f79-4ed6-889a-d0532ee779ae" , "system datetime" : "2017-03-31 09:13:46" , "system document type" : "TLG" , "TIM_A" : 91721.8 , "A" : "91721.800000" , "DLV" : [{ "A" : "0007" , "B" : "20.000000" , "C" : "DET78" , "G" : "DVC2"} , { "A" : "0007" , "B" : "23.000000" , "C" : "DET75" , "G" : "DVC2"} , { "A" : "008D" , "B" : "1.000000" , "C" : "DET1" , "G" : "DVC2"} , { "A" : "0186" , "B" : "0.000000" , "C" : "DET4" , "G" : "DVC1"} , { "A" : "008D" , "B" : "30.800000" , "C" : "DVC1" , "G" : "008D"} , { "A" : "0095" , "B" : "13.150000" , "C" : "DET2" , "G" : "DVC1"} , { "A" : "019A" , "B" : "0.000000" , "C" : "DET2" , "G" : "DVC1"} , { "A" : "DET4" , "B" : "1530.000000" , "C" : "DET2" , "G" : "DVC1"} , { "A" : "018D" , "B" : "10.980000" , "C" : "DET5" , "G" : "DVC1"} , { "A" : "018D" , "B" : "11.019600" , "C" : "DET6" , "G" : "DVC1"}] , "PTN" : { "A" : "60.420072" , "B" : "24.373801" , "C" : "10.980000" , "D" : "12.910500" , "G" : "76.612670"} , "CAN" : { "C" : "1.000000"}}

The REST (W3C, 2011)/JSON (ECMA International, 2017) API (Application Programming Interface) was selected to be used, because the JSON is human readable and XML is easy to convert to JSON objects. There are at least two different ways to implement the online data transfer using REST/JSON communication protocol adapting the ISO 11783-10 standard:

1. Using ISO 11783-10 XML structure as it is and converting all the root elements to own JSON objects. These objects are requested and sent to and from the cloud database, in order to get all the needed ids. New task is started by sending a new TSK element (element structure is explained later) with correct references.
2. Simplify the ISO 11783-10 XML structure by removing the references inside the XML and moving all the elements inside the TSK element. All information is sent once inside the TSK JSON, which starts the new task.

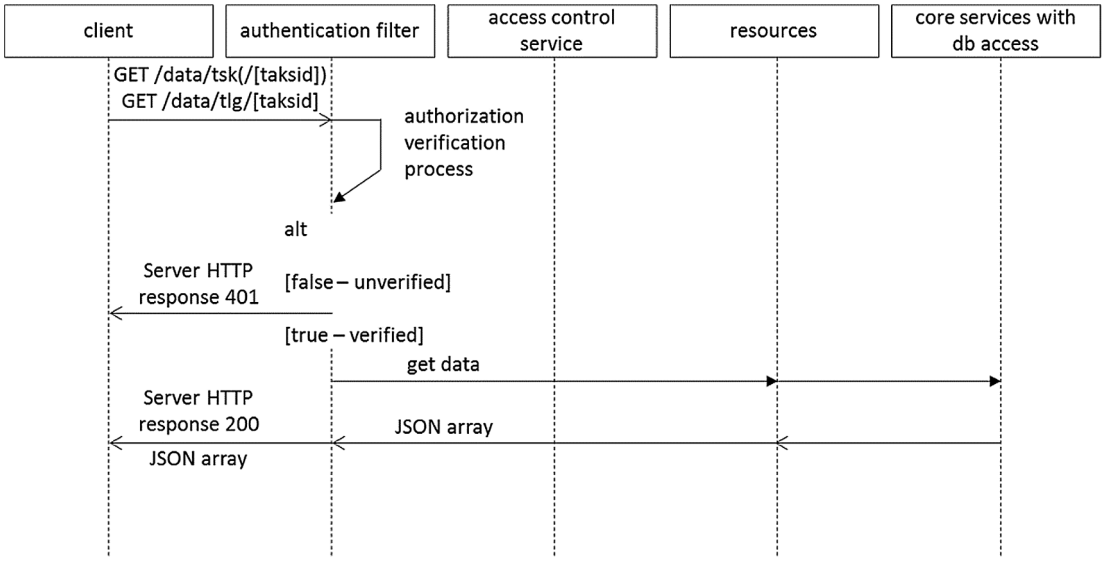

Fig. 14. Requesting data.

Table 2
The sizes of the transmission buffers in different tests.

Name of the parcel	Area (ha)	number of TLG messages	Buffer size (number of TLG messages)		
			Min	Max	AVG
Pumppulohko A	2.99	23,939	0	303	4.07
Heikkilä A	2.86	45,614	0	26	2.139
Uutela A	11.29	112,268	0	139	2.226
Kaupanpelto A	1.84	25,614	0	14	2.126
Tientaus A	4.09	44,537	0	44	2.205
Uutela A	11.29	27,443	0	14	2.150
Kirjava Pohj. A	3.09	37,734	0	35	2.194
Kirjava Etel. A	7.56	81,124	0	94	2.198
Pelto A (part 1)	12.30	134,545	0	512	4.261
Pelto A (part 2)	12.30	185,123	0	451	2.846
Pelto A (part 3)	12.30	166,510	0	503	2.927
Pelto A (part 4)	12.30	52,942	0	11,767	1938
Hovin Luhta A (part 1)	4.66	70,303	0	24	2.213
Hovin Luhta A (part 2)	4.66	32,846	0	24	2.140
Luhta A	8.02	69,209	0	20	2.150
Riihipelto A	5.26	37,251	0	24	2.176
Kirjava Länt. A	5.85	55,441	0	556	6.238
Kirjava Länt. A	5.85	55,441	0	556	6.238

Both approaches have advantages and disadvantages, but the first option was chosen because it follows the existing standard more closely.

3.4.1. Backend

The backend application stack is built on REST/JSON API with MongoDB document oriented Data Base Management System (MongoDB, 2008). The interface is implemented using Java and deployed on GlassFish application server, the open source Java EE (Oracle, 2018) reference implementation, and published using representational state transfer architectural style that provides a set of operations that a remote client can invoke over a network using the HTTP protocol (Leach, 1999a). The backend data storage solution is built on MongoDB, a document oriented non-relational database management system. MongoDB implements a flexible data model that supports storing the JSON data as it is, enabling agile prototyping and efficient client-to-backend data processing.

3.4.2. Data operations

The prototype REST/JSON API provides simple resources for online data transfer from mobile unit to data storage and for requesting data from that storage. The API implements HTTP Basic Authentication (Leach, 1999b) for authentication and authorization.

The data transfer sequence starts with authentication handshake between mobile unit and back-end (Fig. 11). The client accesses the authenticate resource using POST HTTP method and sends BASE64 (Josefsson, 2006) encoded credentials (username and password) as Authorization header. Also a valid client id is required. In case of successful authentication sequence the authorization is granted by assigning an access token to be used in further requests.

After successful authentication and authorization process the client requests a task identifier using GET/taskid resource (Fig. 12). The task identifier is used by the back-end system as a data set join key to identify the task specific documents in the database. The task identifier is required as a path parameter whenever the data is posted to the resource.

Data is sent to POST/data/[tasked] resource using the HTTP POST method (see Fig. 13). The resource accepts application/json mime type and the payload JSON array can contain TSK or TLG type documents, or both. The back-end stores each document in received array as an individual object into database (Table 1).

3.4.3. Beginning of the work

The work is started by creating a new task, including the header information of the work that is meant to be done. The database returns the task id that is used as a reference in communication with the database after the beginning of the work.

URI: POST {BASE URL}/data/[taskid]

Send:

```
{"TSK":{"B":"<TaskDesignator>","D":"FRM<FarmIdRef>","E":"PFD<ParcelIdRef>","F":"WKR<WorkerIdRe
f>","G":"2","TIM":{"A":"<StartTime>"},"OTP":{"A":"CPC<OperationIdRef>","B":"OTQ<OperationSpeci
ficationIdRef>"},"CTP":{"A":"CTP<PlantIdRef>","B":"<PlantDesignator>","CVT":{"A":"CVT<PlantVar
ietyIdRef>","B":"<PlantVarietyDesignator>"}},"PDT":{"A":"PDT<FertilizerIdRef>","B":"<Fertilize
rDesignator>"}, "DAN":[{"A":"<TractorNAME>"
,"C":"DVC<TractorIdRef>"},{"A":"<ImplementNAME>","C":"DVC<ImplementIdRef>"}],
"CAN":{"C":"<comment>"}}}
```

Receive:

```
HTTP response 200/401
```

3.4.4. Definition of captured data

It is supposed that each machine has a device description according to ISO 11783 standard. The device description defines the structure of the TLG elements. In this example two different device descriptions are used: one for the tractor and one for the implement. In the attachments, device description of the ISOBUS compatible Valtra T-163 tractor and also ISOBUS compatible Junkkari Maestro 4000 seed drill are presented. Both produce measurement messages to the ISOBUS which can be logged by the data capturing software. The tractor T-ECU sends the wheel and ground speed, engine RPM, PTO RPM, Rear hitch position and diesel consumption. For the urea consumption, a separate ECU is used to transfer the message from the tractor bus to the ISOBUS. The seed drill sends fertilizer and seed rates as a response to the TC message. An external I/O device was used to send the reference measurements from the fertilizer and seed rates. Position, speed, direction and GNSS quality information is received from the GPS.

3.4.5. Calibration data

The calibration data is send only when the task is started or when the calibration is changed. This kind of data is for example seed and fertilizer rates when prescription control is not used.

Template for the calibration data is:

URI: POST {BASE URL}/data/[taskid]

Send:

```
{"TLG":{"TIM":{"A":"<time>","DLV":[{"A":"<DDI>","B":"<value>","C":"<DET>","G":"<DVC>"}, … ]}}}
```

Receive:

```
HTTP response 200/401
```

Note! In the standard, there is not G-attribute to specify the device; instead it is supposed that device element numbers (DET) are unique.

For example, the seed rate and fertilizer rate are sent using:

URI: POST {BASE URL}/data/[taskid]

Send:

```
{"TLG":{"TIM":{"A":"<time>","DLV":[{"A":"0006","B":"<seed
rate>","C":"DET78","G":"DVC2"},{"A":"0006","B":"<fertilizer rate>","C":"DET75","G":"DVC2"}]}}}
```

Receive:

```
HTTP response 200/401
```

3.4.6. Process data

Process data is sent similarly to calibration data. However, there can be multiple TLG-elements in the same data frame to allow burst sending of measurements from different time stamps. In addition to that, the TLG-element also includes the PTN-element which defines the geographical location of the machine when the measurement has taken place.

Template for the process data:

URI: POST {BASE URL}/data/[taskid]

Send:

```
{"TLG":[{"TIM":{"A":"<time>","PTN":{"A":"<latitude>","B":"<longitude>","C":"<altitude>","D":"<
tatus 0=no GPS, 1=GNSS, 4=RTK>","G":"<number of
satellites>"},"DLV":[{"A":"<DDI>","B":"<value>","C":"<DET>","G":"<DVC>"}, … ]}}, … ]}
```

Receive:

```
{"A":["TLG<id>"}",… ]}
```

In this example the process data consist of TIME_PC, Seed_r, Fertilizer_r, Position, PTO, Hitch, Diesel, Urea, Tr_rpm, Tr_dir, Tr_W_Speed, Tr_R_Speed, LAT, LON, GPS_Speed, Dir, Alt, Qality, Sat_num. Those values are originally used in the proprietary data format and mapped to corresponding device description elements. The JSON frame is hence:

URI: POST {BASE URL}/data/[taskid]

Send:

```
{"TLG":[{"TIM":{"A":"<Time_PC>","PTN":{"A":"<LAT>","B":"<LON>","C":"<Alt>","D":"<Quality>","G"
:"<Sat_num>"},"DLV":[{"A":"0007","B":"<Seed_r>","C":"DET78","G":"DVC2"},{"A":"0007","B":"<Fert
ilizer_r>","C":"DET75","G":"DVC2"},{"A":"008D","B":"<Position>","C":"DET1","G":"DVC2"},{"A":"0
186","B":"<PTO>","C":"DET4","G":"DVC1"},{"A":"008D","B":"<Hitch>","C":"DET4","G":"DVC1"},{"A":
"0095","B":"<Diesel>","C":"DET2","G":"DVC1"},{"A":"019A","B":"<Urea>","C":"DET2","G":"DVC1"},{
"A":"0186","B":"<Tr_rpm>","C":"DET2","G":"DVC1"},{"A":"018D","B":"<Tr_W_Speed>","C":"DET5","G"
:"DVC1"},{"A":"018D","B":"<Tr_R_Speed>","C":"DET6","G":"DVC1"}]}}, … ]}
```

Receive:

```
HTTP response 200/401
```

3.4.7. Requesting data

Prototype provides simple resources for requesting data from process data collection. Client can request all the TSK documents in collection or alternatively a TSK document or a TLG document matching a given task identifier. The resources are:

- GET/data/tsk
- GET/data/tsk/[taskid]
- GET/data/tlg/[taskid]

If successful, the response mime type is application/json and the body is formatted as JSON array with 1 – n JSON documents ([{},…]).

All the POST and GET methods require client id and access token fields in HTTP request header (Fig. 14).

3.5. Results of data capturing and real-time data transfer

In the presented examples in Section 3.4.6 Process data, the payload size for the process data (TLG message) is 570 bytes if only measurements from one time instance are sent. The measurements are updated 5 times per second, which means that the bandwidth requirement is 2850 bps plus 10% overhead from HTTP-frames producing totally 3.1 kbps bandwidth requirement. The upload data rate in 2G EGDE is 60 kbps, in 3G 2 Mbps and in 4G LTE 75 Mbps. Even though the upload data rate varies according to the load of the network, the required bandwidth is only fraction of the available capacity. However, the latency of the mobile network and HTTP server makes it impossible to post messages with 5 Hz data rate. The solution was to pack several TLG messages together and send those in one http message. In the Cropinfra, the default was to pack 5 TLG messages together, and if the network is disconnected, all messages after the last successful sent are packed together.

The data transfer was tested during sowing operations in Luke's experiment farm in Vihti that is located in Southern Finland. There are no big cities near the farm, so the mobile network is typical to countryside. The sizes of the transmission buffers (TLG message queue length) in different tests are listed in Table 2 together with the field area and total number of TLG messages.

Totally, 1,202,443 TLG messages were sent during the data transfer tests, which means 653 Mbytes payload or 18,036,645 individual sensor readings. The total area of the parcels were 81.1 ha, which means that the average raw data production density is 8 Mbytes/ha. The buffer size was typically less than five TLG messages in queue and in average about two, which means that the TLS messages were usually sent successfully once per second. However, the network connection was lost occasionally and in every parcel the maximum size of the buffer is at least 14. The biggest buffer size is in last part of "Pelto A", when the buffer size was 11,767. The connection was lost twice in that test. In the first time, the buffer size grew up to 8125 before the TLG messages were successfully sent. In the second time, the connection did not recover before the computer was shut down. That was the only occasion when data was lost during the tests.

4. Discussion and conclusions

This paper has presented how data collection from field operations can be implemented to support research. The system supports both state-of-the-art ISOBUS (ISO 11783) compatible tractors and implements as well as old proprietary machinery. The provided examples verify that the solution can be used in real scale farm.

The data captured using the system has been used for example in following research; yield maps and in the combined seed drilling applied fertilizer rates were used as source data for generating new fertilizer application task (Kaivosoja et al. 2013, 2017); data from different field operations to determine the spatial overlapping of working widths (Kaivosoja and Linkolehto, 2016), where 140 different complete field operations were analysed to found out that in driving lines are overlapping 10% in average; the positioning data was also used to analyse GNSS positioning error (Kaivosoja and Linkolehto, 2015). In the work of Kaivosoja et al. (2014), application of real time web services were

adapted to the platform. In the future the collected data is even more valuable when machine learning and other artificial intelligence methods will be utilized (Kamilaris and Prenafeta-Boldú, 2018; Liakos et al., 2018).

This paper has also presented how cloud connection can be implemented in the data collection system. The solution was designed to be compatible with the existing ISO 11783-10 standard. The connection was tested and found to be working in the sowing operations of a real scale farm. There was one case of significant data loss, which happened when the computer used for data capturing was shut down before all data was transferred to the cloud service. The loss could have been prevented by saving the buffer to non-volatile memory. It was also found out that the payload size does not matter. The latency of the mobile network and the response time of the HTTP server as well as the reliability of the connection are the primary restrictions in the data transfer.

The FMIS working group in AEF is preparing the guideline to implement the cloud connection manufacturer independently. After the AEF has published the guideline, the work is started in the ISO to standardise the solution. Based on the prototypes implemented so far among the AEF community, it seems that the solution will be based on the MQTT publish-subscribe-based messaging protocol and protobuf data serializing solution developed by the Google. The solutions presented in this paper are alternative but still not conflicting solutions. The POST message used in this paper could be changed to send messages in MQTT with the topic being the same as the URL in POST. The JSON format used in this paper is conducted from the ISO 11783 standard and it is possible to convert the JSON to the protocol buffer.

Appendix A: CSV-file format

See Table 3.

Table 3
Example from the beginning of the datafile.

Data (”27.5.2015 Kirjava Länt. A 5.85 ha Vehnä Kylvö”)

```
<CropinfraData><DATE>27.5.2015 </DATE><WRK> Esko Kaskioja </WRK><PDF> Kirjava  Länt. A 5.85 ha
</PDF><CPC> Kylvö </CPC><CTP> Vehnä </CTP><DVC> Junkkari kylvökone 400 Hinattava </DVC><TRC>
Valtra T-163 Urea </TRC><CAN>
**********************************************************************************************
*************** </CAN><PJT> BIONET </PJT></CropinfraData>
Worker : Esko Kaskioja
Time: 27.5.2015 14:40:53
Task: Kylvö
Plant type : Vehnä
Plant variety : wellamo
Seedrate kg/ha : 200
Pesticide : , Water l :
Fertilizer: NP lannos
Fertilizerrate kg/ha: 370
Parcel : Kirjava  Länt. A 5.85 ha
Tractor : Valtra T-163 Urea
Implement : Junkkari kylvökone 400 Hinattava
Calibration kg Fertilizer : 3,02
Ratio Fertilizer : 3,02
Calibration kg Seed : 1,52
Ratio Seed : 1,52
Project : BIONET
Comment :
**********************************************************************************************
***************
AAA%
TIME_PC       Parcel_N      Mark          Seed_r        Fertilizer_r  Position      PTO
              Hitch         Diesel        Urea          Tr_rpm        Tr_dir        Tr_W_Speed
              Tr_R_Speed    LAT           LON           GPS_Speed     Dir           Alt
              Qality        Pos_Time      Sat_num       Empty
134232.30000000             66.00000000   0.00000000    0.00000000    0.00000000    0.00000000
              0.00000000    26.00000000   3.05000000    0.00000000    647.50000000  1.00000000
              0.00000000    0.00000000    60.44967910   24.35707830   0.07200000    209.66310000
              68.60372900   1.00000000    38554.00000000              11.00000000   0.00000000
134232.50000000             66.00000000   0.00000000    0.00000000    0.00000000    0.00000000
              0.00000000    26.00000000   3.05000000    0.00000000    648.00000000  1.00000000
              0.00000000    0.00000000    60.44967910   24.35707830   0.07200000    209.66310000
              68.60372900   1.00000000    38554.00000000              11.00000000   0.00000000
134232.70000000             66.00000000   0.00000000    0.00000000    0.00000000    0.00000000
              0.00000000    26.00000000   3.05000000    0.00000000    648.00000000  1.00000000
              0.00000000    0.00000000    60.44967930   24.35707990   0.07200000    209.66310000
              68.60372900   1.00000000    38554.00000000              11.00000000   0.00000000
134232.90000000             66.00000000   0.00000000    0.00000000    0.00000000    0.00000000
              0.00000000    26.00000000   3.05000000    0.00000000    647.50000000  1.00000000
              0.00000000    0.00000000    60.44967930   24.35707990   0.07200000    209.66310000
              68.60372900   1.00000000    38554.00000000              11.00000000   0.00000000
```

Appendix B: DDOP

See Tables 4 and 5.

Table 4
Device description of the tractor.

DVC1 : Valtra T-163 Urea

```
<DVC D="A00086000CC00001" E="12345678912301012016" F="31303231303130" G="FF000004406E65"
A="DVC1" B="Valtra Tractor" C="ApplicationVersion?">
  <DET D="Tractor" E="0" F="0" A="DET1" B="1" C="1">
    <DOR A="20101"/>
    <DOR A="20102"/>
    <DOR A="20103"/>
    <DOR A="20104"/>
  </DET>
  <DET D="Board Computer" E="1" F="1" A="DET2" B="2" C="2">
    <DOR A="10209"/>
    <DOR A="10213"/>
    <DOR A="12"/>
  </DET>
  <DET D="GNSS Sensor" E="2" F="1" A="DET3" B="3" C="7">
    <DOR A="20301"/>
    <DOR A="20302"/>
    <DOR A="20303"/>
  </DET>
  <DET D="Rear Hitch" E="3" F="1" A="DET4" B="4" C="6">
    <DOR A="20401"/>
    <DOR A="20402"/>
    <DOR A="20403"/>
    <DOR A="20404"/>
    <DOR A="11"/>
    <DOR A="13"/>
  </DET>
  <DET A="DET5" B="5" E="4" F="1" C="2" D="Wheel based speed sensor">
    <DOR A="6"/>
    <DOR A="7"/>
  </DET>
  <DET A="DET6" B="8" E="5" F="1" C="2" D="Groud based speed sensor">
    <DOR A="9"/>
    <DOR A="10"/>
  </DET>
  <DPT D="Vehicle Length" A="20101" B="009A" C="4466"/>
  <DPT D="Vehicle Width" A="20102" B="009B" C="2500"/>
  <DPT D="Vehicle Height" A="20103" B="009C" C="3140"/>
  <DPT D="DRP to Ground" A="20104" B="00B4" C="500"/>
  <DPT D="NRP GNSS Reported Position X" A="20301" B="0086" C="566"/>
  <DPT D="NRP GNSS Reported Position Y" A="20302" B="0087" C="0"/>
  <DPT D="NRP GNSS Reported Position Z" A="20303" B="0088" C="2215"/>
  <DPT D="Rear Three-Point-Hitch Mounted" A="20401" B="009D" C="4"/>
  <DPT D="CRP Rear Hitch X" A="20402" B="0086" C="-730"/>
  <DPT D="CRP Rear Hitch Y" A="20403" B="0087" C="0"/>
  <DPT D="CRP Rear Hitch Z" A="20404" B="0088" C="925"/>
  <DPD E="Actual Fuel Consumption per Time" A="10209" B="0095" C="1" D="3" />
  <DPD E="Diesel Exhaust Fluid consumption" A="10213" B="019A" C="2" D="16" />
  <DPD E="Actual Speed" A="6" B="18D" C="1" D="1"/>
  <DPT D="Speed Source" A="7" B="190" C="1"/>
  <DPD E="Actual Speed" A="9" B="18D" C="1" D="1"/>
  <DPT D="Speed Source" A="10" B="190" C="2"/>
  <DPD E="PTO RPM" A="11" B="186" C="1" D="1"/>
  <DPD E="Engine RPM" A="12" B="186" C="1" D="1"/>
  <DPD E="Actual Work State" A="13" B="8D" C="1" D="1"/>
</DVC>
```

Table 5
Device description of the implement.

DVC2 : Junkkari Maestro 4000

```
<DVC B="Seed drill controller" A="DVC-22" G="FF000000006E65" F="0000000103E6B8"
E="000000000001" D="A00884002C400001" C="3.74">
  <DPT B="009A" A="10" D="Physical Object Length 1" C="6000" />
  <DPT B="009C" A="11" D="Physical Object Height" C="2200" />
  <DPT B="009A" A="12" D="Physical Object Length 2" C="500" />
  <DPT B="00B3" A="14" D="Actual Cultural Practice 1" C="1" />
  <DPT B="009A" A="15" D="Physical Object Length 3" C="500" />
  <DPT B="00B3" A="17" D="Actual Cultural Practice 2" C="2" />
  <DPT B="0086" A="20" D="Device Element Offset X" C="4500" />
  <DPT B="0087" A="21" D="Device Element Offset Y" C="0" />
  <DPT B="0088" A="22" D="Device Element Offset Z" C="0" />
  <DPT B="009D" A="23" D="Connector type" C="0" />
  <DPT B="009B" A="74" D="Physical Object Width" C="4000" />
  <DPD B="008D" A="25" E="Working state" D="9" C="1" />
  <DPD B="0006" A="30" E="Fertilizer setting" D="0" C="2" />
  <DPD B="0007" A="31" E="Fertilizer measurement" D="15" C="1" />
  <DPD B="0006" A="40" E="Seed setting" D="8" C="2" />
  <DPD B="0007" A="41" E="Seed measurement" D="15" C="1" />
  <DET B="1" A="DET1" F="0" E="0" D="Seed drill body" C="1">
    <DOR A="25" />
    <DOR A="10" />
    <DOR A="74" />
    <DOR A="11" />
  </DET>
  <DET B="75" A="DET75" F="1" E="1" D="Fertilizer rate controller" C="2">
    <DOR A="74" />
    <DOR A="12" />
    <DOR A="14" />
    <DOR A="25" />
    <DOR A="30" />
    <DOR A="31" />
  </DET>
  <DET B="78" A="DET78" F="1" E="4" D="Seed rate controller" C="2">
    <DOR A="74" />
    <DOR A="15" />
    <DOR A="17" />
    <DOR A="25" />
    <DOR A="40" />
    <DOR A="41" />
  </DET>
  <DET B="81" A="DET81" F="1" E="7" D="Connector" C="6">
    <DOR A="20" />
    <DOR A="21" />
    <DOR A="22" />
    <DOR A="23" />
  </DET>
</DVC>
```

References

AEF, 2019a. ISOBUS. Retrieved July 29, 2019, from https://www.aef-online.org/the-aef/isobus.html.

AEF, 2019b. Think ISOBUS, AEF Presskit. Retrived July 29, 2019, from https://www.aef-online.org/fileadmin/user_upload/Content/pdfs/AEF-com-pack.zip https://www.aef-online.org/fileadmin/user_upload/Content/pdfs/AEF-com-pack.zip.

Axiomatic, 2008. Press Release. Retrieved May 5, 2018, from http://www.axiomatic.com/pr-AX030100-may08.pdf.

Bray, T., 2015. RFC 7493 – The I-JSON Message Format. Retrieved October 9, 2018, from https://tools.ietf.org/html/rfc7493.

Bray, T., 2017. RFC 8259 – The JavaScript Object Notation (JSON) Data Interchange Format. Retrieved October 9, 2018, from https://tools.ietf.org/html/rfc8259.

Dataväxt, 2019. LOGMASTER. Retrieved July 24, 2019, from https://www.datavaxt.se/logmaster/.

ECMA International, 2017. Standard ECMA-404, The JSON Data Interchange Syntax. Retrieved June 11, 2018, from http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf.

Fielding, R.T., 2000. Architectural Styles and the Design of Network-based Software Architectures. Doctoral dissertation. University of California.

Fountas, S., Sorensen, C.G., Tsiropoulos, Z., Cavalaris, C., Liakos, V., Gemtos, T., 2015. Farm machinery management information system. Comput. Electron. Agric. 110, 131–138.

Google, 2018. Protocol Buffers, Retrieved October 9, 2018, from https://developers.google.com/protocol-buffers/.

IETF Trust, 2007. Hypertext Transfer Protocol – HTTP/1.1. Retrieved October 9, 2018, from https://www.w3.org/Protocols/HTTP/1.1/rfc2616bis/draft-lafon-rfc2616bis-03.html.

ISO/IEC 20922, 2016. Information technology – Message Queuing Telemetry Transport (MQTT) v3.1.1. Retrieved October 9, 2018, from https://www.iso.org/standard/69466.html.

ISO, 2019. ISO 11783-5:2019. Tractors and machinery for agriculture and forestry – Serial control and communications data network – Part 5: Network management. Retrieved July 29, 2019, from https://www.iso.org/standard/74366.html.

ISO, 2017. ISO 11783-1:2017 – Tractors and machinery for agriculture and forestry – Serial control and communications data network – Part 1: General standard for mobile data communication. Retrieved November 21, 2018, from https://www.iso.org/standard/57556.html.

ISO, 2017. ISO 11783-7:2015 - Tractors and machinery for agriculture and forestry – Serial control and communications data network – Part 7: Implement messages application layer. Retrieved April 29, 2019, from https://www.iso.org/standard/59380.html.

ISO, 2015. ISO 11783-10:2015 – Tractors and machinery for agriculture and forestry – Serial control and communications data network – Part 10: Task controller and management information system data interchange. Retrieved November 21, 2018, from https://www.iso.org/standard/61581.html.

Josefsson, S., 2006. RFC 4648 – The Internet Society, 2006. The Base16, Base32, and Base64 Data Encodings. Retrieved June 11, 2018, from https://tools.ietf.org/html/

rfc4648.
Kaivosoja, J., Pesonen, L., Kleemola, J., Pölönen, I., Salo, H., Honkavaara, E., Saari, H., Mäkynen, J., Rajala, A., 2013a. A case study of a precision fertilizer application task generation for wheat based on classified hyperspectral data from UAV combined with farm history data. In: Remote Sensing for Agriculture, Ecosystems, and Hydrology XV. SPIE Remote Sensing, Proceedings, pp. 8887.
Kaivosoja, J., Näsi, R., Hakala, T., Viljanen, N., Honkavaara, E., 2017. Applying different remote sensing data to determine relative biomass estimations of cereals for precision fertilization task generation. In: Salampasis, Michail, Theodoridis, Alexandros, Bournaris, Thomas (Eds.), Proceedings of the 8th International Conference on Information and Communication Technologies in Agriculture, Food and Environment (HAICTA 2017), Chania, Crete Island, Greece, September 21–24, 2017.
Kaivosoja, J., Linkolehto, R., 2016. Spatial overlapping in crop farming works. Agron. Res. 14, 41–53.
Kaivosoja, J., Linkolehto, R., 2015. GNSS error simulator for farm machinery navigation development. Comput. Electron. Agric. 119, 166–177.
Kaivosoja, J., Jackenkroll, M., Linkolehto, R., Weis, M., Gerhards, R., 2014. Automatic control of farming operations based on spatial web services. Comput. Electron. Agric. 100, 110–115.
Kaivosoja, J., Pesonen, L., Kleemola, J., Pölönen, I., Salo, H., Honkavaara, E., Saari, H., Mäkynen, J., Rajala, A., 2013b. A case study of a precision fertilizer application task generation for wheat based on classified hyperspectral data from UAV combined with farm history data. In: Neale, Christopher M.U., Maltese, Antonino (Eds.), Proceedings of SPIE 8887, Remote Sensing for Agriculture, Ecosystems, and Hydrology XV, 88870H, October 16, 2013.
Kamilaris, A., Prenafeta-Boldú, F.X., 2018. Deep learning in agriculture: a survey. Comput. Electron. Agric. 147, 70–90.
Leach, P. J., 1999. RFC 2616 – Hypertext Transfer Protocol – HTTP/1.1. Retrieved June 11, 2018, from https://tools.ietf.org/html/rfc2616.
Leach, P. J., 1999b. RFC 2617 – HTTP Authentication: Basic and Digest Access Authentication. Retrieved June 11, 2018, from https://tools.ietf.org/html/rfc2617.
Liakos, K.G., Busato, P., Moshou, D., Pearson, S., Bochtis, D., 2018. Machine learning in agriculture: a review. Sensors 18 (8), 2674.
MongoDB, 2008. Introduction to MongoDB. Retrieved June 11, 2018, from https://docs.mongodb.com/manual/introduction/.
National Instruments, 2018. What Is LabVIEW. Retrieved May 5, 2018, from http://www.ni.com/fi-fi/shop/labview.html.
Nikander, J., Linkolehto, R., Jäger, M., Pesonen, L., Ronkainen, A., Suokannas, A., 2017. Prototype environment for integrating and sharing farm things and associated data. Adv. Anim. Biosci. 8 (2), 645–649.
Nikander, J., Koistinen, M., Laajalahti, M., Pesonen, L., Ronkainen, A., Suomi, P., 2015. Farm information management infrastructures in the future. In: 26th International Workshop on Database and Expert Systems Applications (DEXA) Valencia 2015, pp. 104–107.
Oval, 2016. Super Micro Flowmate General Specification. Retrieved May 5, 2018, from http://www.oval.co.jp/english/gs_home/gbb340e-14c.pdf.
Oksanen, T., Piirainen, P., Seilonen, I., 2015. Remote access of ISO 11783 process data by using OPC Unified Architecture technology. Comput. Electron. Agric. 117, 141–148.
Oksanen, T., Öhman, M., Miettinen, M., Visala, A., 2005. ISO 11783 –Standard and its Implementation. IFAC Proc. Vol. 38 (1), 69–74.
Oracle, 2018. Java EE – Documentation. Retrieved June 11, 2018, from http://www.oracle.com/technetwork/java/javaee/documentation/index.html.
Pesonen, L., Teye, F., Ronkainen, A., Koistinen, M., Kaivosoja, J., Suomi, P., Linkolehto, R., 2014. Cropinfra – an internet-based service infrastructure to support crop production in future farms. Biosyst. Eng. 120, 92–101.
Sachs, Goldman, 2014. The Internet of Things: Making sense of the next mega-trend. Technical report. Goldman Sachs.
Steinberger, G., Rothmund, M., Auernhammer, H., 2009. Mobile farm equipment as a data source in an agricultural service architecture. Comput. Electron. Agric. 65, 238–246.
Stone, M.L., Benneweis, R.K., Van Bergeijk, J., 2008. Evolution of electronics for mobile agricultural equipment. Trans. ASABE 51 (2), 385–390.
Suonentieto, 2019. Lohkokirjanpito ja täsmäviljely – AgriSmart. Retrived July 24, 2019, from https://www.suonentieto.fi/tuotteet/agrismart/.
Suomi, P., Oksanen, T., 2015. Automatic working depth control for seed drill using ISO 11783 remote control messages. Comput. Electron. Agric. 116, 30–35.
Söderström, M., Sohlenius, G., Rodhe, L., Piikki, K., 2016. Adaptation of regional digital soil mapping for precision agriculture. Precis. Agric. 17, 588–607.
Thomasson, J., Baillie, C., Antille, D., Lobsey, C., McCarthy, C., 2019. Autonomous technologies in agricultural equipment: a review of the state of the art. ASABE Distinguished Lecture No. 40 In: Agricultural Equipment Technology Conference, 11–13 February 2019, Louisville, Kentucky, USA, pp. 1–17.
W3C, 2011. REST – Semantic Web Standards. Retrieved June 11, 2018, from https://www.w3.org/2001/sw/wiki/REST.
Wolfert, S., Ge, L., Verdouw, C., Bogaardt, M., 2017. Big data in smart farming – a review. Agric. Syst. 153 (2017), 69–80.
Öhman, M., Oksanen, T., Miettinen, M., Visala, A., 2004. Remote maintenance of agricultural machines. Proceedings of the 1st IFAC Symposium on Telematics Applications in Automation and Robotics 2004, Helsinki, Finland.