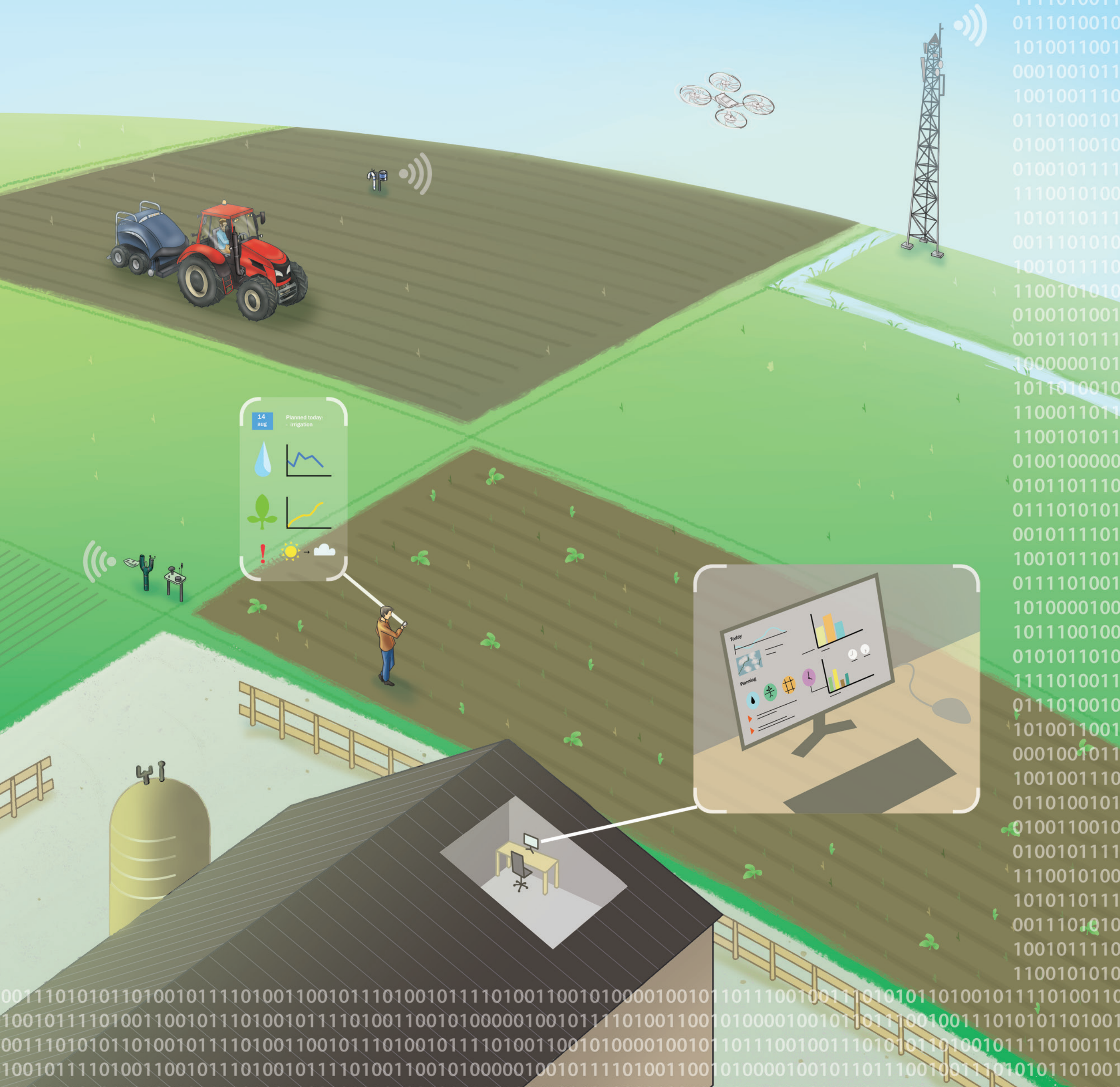
Advancement of farming by facilitating collaboration

Reference architectures and models for farm software ecosystems

Jan Willem Kruize



Advancement of farming by facilitating collaboration

Reference architectures and models for farm software ecosystems

Jan Willem Kruize

This research was conducted under the auspices of
the Wageningen School of Social Sciences (WASS)

Advancement of farming by facilitating collaboration

Reference architectures and models for farm software ecosystems

Jan Willem Kruize

Thesis
submitted in fulfilment of the requirements for the degree of doctor
at Wageningen University
by the authority of the Rector Magnificus
Prof. Dr A.P.J. Mol,
in the presence of the
Thesis Committee appointed by the Academic Board
to be defended in public
on Friday 20 January 2017
at 4 p.m. in the Aula.

Jan Willem Kruize
Advancement of farming by facilitating collaboration
Reference architectures and models for farm software ecosystems
242 pages.

PhD thesis, Wageningen University, Wageningen, NL (2017)
With references, and summaries in Dutch and English
ISBN 978-94-6257-966-8
DOI 10.18174/392940

Acknowledgements

In 2009 I got the opportunity to start a PhD with a focus on information management in agriculture. Although, I was not sure if such a position would fit me, I knew it would be a once in a lifetime opportunity and decided to apply. A main reason to apply was that I enjoyed working in-between IT and the users of IT, an aspect of my work that I still love. Moreover, I enjoyed working in agriculture and wanted to contribute in making it more sustainable. I was traveling when I heard the good news that the funding for the PhD was arranged, which meant that I could start a new journey at the beginning of 2010.

In this journey, to acquire a PhD, I have been supported by many people who I all want to thank. First, I want to thank my promoter, Adrie en my co-promoters, Sjaak and Huub. All three helped me with different aspects while working on my PhD research. Adrie challenged me at every appointment we had, which often resulted in me returning to my office with more questions than answers. Luckily this changed during the years and all these conversations helped me to improve my thinking and to understand more abstract concepts, a skill that I can use for the rest of my life. Sjaak has been the initiator as he arranged this PhD position. Without him I would never have been able to start this research. Additionally, during my PhD he helped me to improve my papers and writing, which was certainly needed at times. Finally I want to thank Huub. Huub has helped me tremendously, especially during the first years of my PhD. He helped me bring order to my thoughts and his door was always open for me to ask questions. Therefore, I never had to struggle too long.

Next to my promotor and co-promotors I want to thank my other co-authors Cor, Robbert, Ayalew and Daan for the nice collaboration that resulted in the different papers. Beside the co-authors I want to thank the participants of the PPL program, the farmers that participated in my research, the experts that validated the models and the students that supported me in my research. I especially want to thank the student Valentijn who helped with modelling farm business processes and Timon and Ivor who validated models and supported improving them.

Next to these people I want to thank my colleagues from the Operations, Research and Logistics group, Information Technology group and Wageningen Economic Research. All colleagues have always showed their interest in me and my work and were open for nice conversations. These small talks and nice coffee breaks gave a good atmosphere that I enjoyed. Between my colleagues I especially want to thank my roommates at the university, Elise, Nick, John, Marlies and Floor. I had nice conversations with all of you and during my PhD, we challenged each other and had good fun. Especially I want to thank Nick and John who supported me with writing some papers. Furthermore I want to thank all my other PhD colleagues Willem, Agatha, Bing, Alex and all former and new colleagues that are part of this group of PhD students. It has always been good fun to hang out with you after work and it still is when we go out for drinks.

Besides al my colleagues and friends from Wageningen University and Research I want to thank my friends from Groningen and the friends I made during my studies. All of you have always been interested and compassionate. I've known many of you for a long period, some even from early childhood. A group of friends I can always rely on. I am already looking forward to the annual weekend trip I always make with my study friends, the trip to the grand prix spa francorchamps, the motor cycle weekend and all other trips we will do the coming years. From this group of friends I especially want to thank Robbert and Erik, my paranymphs, who have supported me during my PhD research and on stage during the defence.

Additionally I want to thank Anouk. Anouk thank you for always listening to me, observing me and asking the right questions that helped me to be able to take the next step. Without this support I do not know if I have been able to finalize this PhD and manage my job and all others things that are part of life.

Finally, I want to thank my family for supporting me in all possible ways they could. My family has always been very supportive, each in their own way. They always encouraged me to challenge myself but never made me afraid to fail. A habit that enabled me to finalize this PhD research and that gives me energy to start a new journey.

Table of Contents

Chapter 1. General introduction 3
1.1 Background and problem definition.......................................4
1.2 Solution directions for improving the adoption of advanced farm management styles 6
1.3 Hypothesis and research questions.......................................9
1.4 Methodology...10
1.4.1 Design science and case study 10
1.4.2 Approach 11

Chapter 2. Improving arable farm enterprise integration - Review of existing technologies and practices from a farmer's perspective...15
2.1 Introduction 17
2.2 Background 19
2.3 Methodology...22
2.3.1 Finding bottlenecks in arable farm enterprise integration....22
2.3.2 Description of the characteristics of the case studies 27
2.3.3 Expert validation 28
2.4 Reference Architecture of Agricultural Enterprises 29
2.4.1 The structure of RAAgE 29
2.4.2 Farm process groups and farm business processes 31
2.4.3 The process group product management..........................33
2.4.4 Describing the arable farm ICT components......................37
2.4.5 Validation of RAAgE 1.0 40
2.5 Presentation of the bottlenecks 41
2.5.1 Bottlenecks found in the farm enterprise of case study 2....41
2.5.2 Other bottlenecks..48
2.5.3 Validation of the bottlenecks..49
2.6 Discussion and conclusions 51

Chapter 3. A Reference Architecture for Farm Software Ecosystems..55
3.1 Introduction 57
3.2 Software Ecosystems and software development for farming..60
3.3 Materials and methods..62

3.3.1 The future internet program and existing Farm Software Ecosystems 62
3.3.2 Methodology 63
3.4 Requirements analysis 65
3.4.1 Object system 65
3.4.2 Functional requirements for a Farm Software Ecosystem 67
3.4.2.1 Data handling and seamless data exchange supported by Standards 68
3.4.2.2 Configuration of ICT components 70
3.4.2.3 Interoperability between ICT components 71
3.4.2.4 Organization 73
3.4.2.5 Technical/non-functional requirements 74
3.4.3 Specifications for Farm Software Ecosystems 74
3.5 Basic design of a Farm Software Ecosystem Reference Architecture 76
3.5.1 High-level description of the reference architecture 76
3.5.2 Platform 77
3.5.3 Actors and their relationships 80
3.5.4 ICT components, configuration and orchestration 81
3.5.5 Business services and contracts 83
3.5.6 Open Software Enterprise 86
3.5.6.1 Governance of a Farm Software Ecosystem 86
3.5.6.2 Research and development 87
3.5.6.3 Software product management and configuration support documentation 88
3.5.6.4 Marketing and sales 88
3.5.6.5 Consulting and support services 88
3.5.7 Example of a Farm Software Ecosystem 89
3.6 Evaluation 91
3.6.1 Requirements verification 91
3.6.2 Validation of the reference architecture by mapping existing Farm Software Ecosystems 91
3.7 Discussion and conclusions 94

Chapter 4. A reference information model for configuring farm business processes and supporting application services 97
4.1 Introduction 99
4.2 Materials and methods 102
4.2.1 RAAgE 1.0 102

4.2.2 Methodology: Design Oriented Research 104
4.3 Requirements analysis 105
4.3.1 Object system 106
4.3.2 Stakeholders 107
4.3.3 Functional requirements for RAAgE 2.0 108
4.3.3.1 Representation of the farm object system 108
4.3.3.2 Configuration and re-configuration of farm business processes 109
4.3.3.3 Reuse of ICT components and configuration knowledge 110
4.3.3.4 Collaboration of multiple vendors in a Best-of-Breed approach... 110
4.3.4 Design specification 111
4.4 Design of a reference information model: RAAgE 2.0 111
4.4.1 Meta model of RAAgE 2.0 112
4.4.2 The components of RAAgE 2.0 114
4.4.2.1 Product reference model 114
4.4.2.2 Resource reference model 115
4.4.2.3 Process reference model 121
4.4.2.4 Configuration tree and the business rules................ 125
4.4.2.5 Templates 126
4.5 Evaluation 126
4.5.1 Exemplar for verification: Usage of RAAgE in configuration time 126
4.5.1.1 The use of the configuration tree 127
4.5.1.2 Product definition 127
4.5.1.3 Business service definition.................................... 128
4.5.1.4 Resource definition... 128
4.5.1.5 Farm-specific process model.................................. 131
4.5.1.6 Business process re-configuration 134
4.5.2 Requirements verification 136
4.6 Discussion and conclusions 137

Chapter 5. Configuring a farm information system from multi-vendor apps - A case of late blight protection 141

5.1 Introduction 143
5.2 Materials and methods 145
5.2.1 Materials 145

5.2.1.1 Case study description: advanced late blight protection in potatoes ... 145
5.2.1.2 Farm Software Ecosystem reference architecture ... 147
5.2.1.3 A reference architecture for business process configuration and the related application services (RAAgE 2.0) ... 151
5.2.1.4 Ontologies and standards used ... 153
5.2.2 Methods ... 154
5.2.2.1 Design Oriented Research using a case study approach ... 154
5.2.2.2 Approach ... 155
5.3 Results ... 156
5.3.1 Requirements definition by analysis of the business processes ... 156
5.3.2 The developed or adapted atomic application components ... 160
5.3.2.1 The atomic application components ... 160
5.3.2.2 Capabilities and messages of the atomic application components ... 162
5.3.2.3 Evaluation - app development ... 166
5.3.3 App configuration: Late Blight protection ... 166
5.3.3.1 App configuration ... 166
5.3.3.2 App interaction diagram ... 168
5.3.3.3 Evaluation – configuration ... 170
5.3.4 App usage - run-time example ... 171
5.3.4.1 The application usability ... 171
5.3.4.2 Evaluation – app usage ... 172
5.4 Discussion and conclusions ... 173

Chapter 6. General discussion ... 177
6.1 Introduction ... 178
6.2 Answering the research questions ... 180
6.3 Major contributions to practice and science ... 187
6.4 Directions for further research ... 193

References ... 197

Appendices **209**
Appendix a – Ontology 209
Appendix b - Requirements Table Farm Software Ecosystem......... 216
Appendix c - Business Rules RAAgE 2.0 220

Summary **223**

Samenvatting **231**

About the author **239**

Completed training and supervision plan **240**

1

General introduction

1.1 Background and problem definition

“Since time began, mankind has been threatened by the combination of growing populations and diminishing resources” (Harris, 1977). This continuous threat results in decreasing food security over time. As history shows, this threat can be deferred by innovative solutions developed by mankind.

Agricultural production has been increasing by innovative technologies for centuries. A first “green revolution” started in 1870 in Europe in which production and productivity grew (Zanden, 1991). The use of fertilisers, new seeds and machines increased yields while requiring less labour. In the 1960s the introduction of new crop varieties, greater input of fertilisers, water, pesticides and other technologies resulted in a second green revolution which increased production tremendously (Evenson and Gollin, 2003; FAO, 1996; Tilman et al., 2002; WHO, 1990).

Due to a growing world population that is expected to exceed 10 billion in 2050 (Crossette and Kollodge, 2011) and diminishing resources, a third green revolution is required to increase agricultural production on less acreage. At the same time, increase of food production should be accomplished in a sustainable manner taking the carrying capacity of the earth into account because society requires food to be produced environmental-friendly (Seuring and Müller, 2008). Additional to these requirements, consumers require food to be safe (Beulens et al., 2005; Grunert, 2005; Trienekens et al., 2012). Therefore, numerous policies, regulations and laws are deployed that require compliance of all actors in these networks, including farm enterprises. Due to these rules and regulations and the demands to produce more food in a sustainable safe and transparent manner, farm management has become more complex and knowledge-intensive.

In many sectors, Information and Communication Technologies (ICT) have proven they can improve management of enterprises, especially in information- and knowledge-intensive environments. Therefore, ICT has also been introduced to farm enterprises over the last 30 years. Nowadays, a broad spectrum of state-of-the-art computer-based systems, here referred to as ICT Components, are available that can support the whole farm management cycle of monitoring, planning and control processes (Figure 1) (Aubert et al., 2012; Cox, 2002; Lamb et al., 2008; Wolfert et al., 2010). Examples of such technologies are satellites to create field images, soil sensors, weather sensors, advanced planning and decision support systems, farm management systems and machines and robots that can operate with high precision.

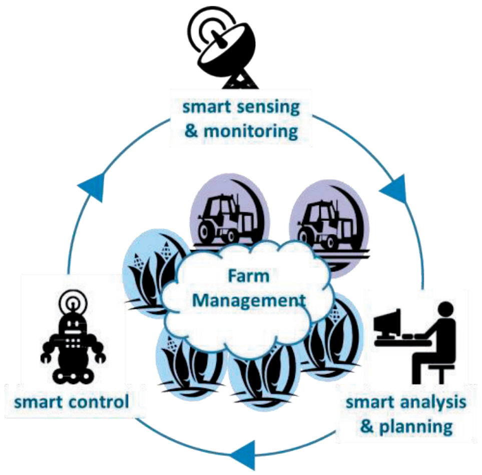


Figure 1: Monitoring and control cycle.

These technologies can be used in agriculture to realise advanced farm management styles that can contribute to a third green revolution in which yields increase while food is produced in a more sustainable, safe and transparent manner. A well-known advanced farm management style which is both knowledge- and information-intensive is precision agriculture (Bongiovanni and Lowenberg-Deboer, 2004; Fountas et al., 2006). Precision agriculture increases the profitability of crop production while simultaneously reducing the negative environmental impact by tight monitoring and control in which application rates of agricultural inputs are adjusted to local needs (Pierce et al., 1999). With precision agriculture, farm enterprises practice tight control of temporal and spatial conditions by monitoring crops, fields and environment. Practicing precision agriculture is complex because it involves large amounts of data, collected by sensors, about multiple objects such as weather, soil crops and yield. These large amounts of data need to be analysed and transformed into control operations. This requires a challenging combination of skills in agronomy, soil science, information technology, spatial statistics, and Geographical Information Systems (GIS) (Bramley, 2009).

Due to the large amounts of data, knowledge intensity and the spatial and temporal aspects related to crop production, ICT has become inevitable to realise advanced farm management styles. Currently, a wide range of ICT Components is available that can support farmers' business processes. These ICT Components often need to exchange data as they are neither part of a single farm enterprise nor developed by the same software vendor. However, data exchange and integration of ICT Components used in agriculture is cumbersome, which hampers the adoption of more advanced management styles such as precision agriculture and the use of available ICT Components (Aubert et al., 2012; Jochinke et al., 2007; Lamb et al., 2008; McBratney et al., 2005; Pedersen et al., 2004; Reichardt and Jürgens, 2009). Consequently, fewer farmers have been able to operationalise precision agriculture than was predicted 5 or 10 years ago (Bramley, 2009; Cook et al., 2000; Fountas et al., 2005; Lowenberg-DeBoer, 2003; McBratney et al., 2005). Therefore, the integration capabilities of ICT Components used in agriculture need to be improved (Aubert et al., 2012; Kaloxylos et al., 2012; Kaloxylos et al., 2014; McBratney et al., 2005; Sørensen et al., 2010a; Wolfert et al., 2010).

1.2 Solution directions for improving the adoption of advanced farm management styles

In information- and knowledge-intensive management environments, researchers focus on integrating business processes and data (Lee et al., 2003). This line of research is called Enterprise Integration, which is defined as the process of ensuring the interaction between enterprise entities necessary to achieve domain objectives (EN/ISO 19439).[1] Enterprise integration can refer to integration between different enterprises (inter-enterprise integration) or within an enterprise (intra-enterprise integration). It can be achieved by coordination of processes, interoperability between applications, standardisation of data and connectivity between devices and systems (Giachetti, 2004).

A key enabler for enterprise integration in various industries has been the development and implementation of Enterprise Resource Planning (ERP) systems (Jacobs, 2007). ERP is defined as a framework for

[1] ISO 19439:2006 Enterprise integration - Framework for enterprise modelling.

organising, defining, and standardising the business processes necessary to effectively plan and control an organisation so that the organisation can use its internal knowledge to seek external advantage (Blackstone and Cox, 2005). Current ERP systems enable enterprises to integrate business functions into a single system and share data with external actors. Examples of these functions are customer relationship management, finance, accounting and material resource planning. These ERP systems are often developed by an organisation (e.g. SAP, Lawson and Microsoft) and implemented by partners that customise the ERP system to the customer-specific demands. In agriculture, traditional ERP systems lack integration of all required functions to enable farm enterprise integration (Verdouw et al., 2015). The available ERP systems or other integral business solutions are not able to support the complex processes (e.g. rolling planning and the space-time continuum of crop production) involved in agriculture (Akkermans et al., 2003; Rettig, 2007; Van Wezel et al., 2006). Moreover, these ERP solutions do not focus on supporting operations to optimise crop production (e.g. precision agriculture, specific decision support). To support farm management, domain-specific solutions have been developed in a national or regional context, named Farm Management Information System (FMIS). These FMISs are able to support specific aspects of farming but lack support for whole-farm management. Therefore, researchers are working on the development of more advanced farm management systems (Nikkilä et al., 2010; Sørensen et al., 2010a). Therewith, development of a global, overarching system, by one single vendor that can support all business functions of farmers is neither a feasible nor, from a competitive point of view, desirable solution. Then the key question remains how to develop integrated solutions that can seamlessly support farmers' business processes for advanced farm management styles.

Development of integrated farm management systems is difficult due to the specific characteristics of agriculture (e.g. small enterprises, remote locations, horizontal and vertical cooperation in supply chains, little investments in software and enterprise integration) and the complex biological processes that need to be controlled by farm business processes. Advanced farm management styles require a variety of ICT Components of multiple vendors such as sensors, terminals, implements and FMISs. Additionally, each specific farm enterprise requires particular types of ICT Components. Development of farm-specific solutions for each individual farm for all types of different business processes would be time-consuming and therefore far too expensive. A combination of approaches that caters for this problem is ICT Mass Customisation (Verdouw et al., 2010a), in combination with Best of Breed (Light et al., 2001). ICT Mass Customisation combines advantages of standard and customised software by enabling on-demand configuration of information systems from components with standardised interfaces (Verdouw et al., 2010a). These components (for farming e.g. sensors, decision-support systems, FMISs, machines) could be supplied by different software vendors, which allows Best-of-Breed solutions. Realising ICT Mass Customisation requires the use of reference models (Verdouw et al., 2010a) to streamline and accelerate the design of specific models by providing a generic solution (Fettke and Loos, 2003; Rosemann and van der Aalst, 2007a). Reference models are sometimes called universal models, generic models or model patterns and represent a class of domains (Fettke and Loos, 2003). The application of reference models is motivated by the "Design by Reuse" paradigm. Most of the major ERP vendors use reference models to make enterprise-specific information systems by facilitating parametrisation and instantiation (van der Aalst et al., 2006).

Currently, there are (reference) models available that can be used for agriculture (e.g. SCOR,[2] IMOT,[3] EDITeelt,[4] AgroXML,[5] ISA-95[6]). Furthermore, information and data flows in agriculture are systematically described in several publications (Fountas et al., 2006; Nash et al., 2009; Sørensen et al., 2010b). These flow diagrams describe farmers' decision-making processes and how farmers manage their controlled objects (e.g. (sub)fields, crops, farm). However, all of these reference models are either outdated (IMOT), do not focus on farm enterprises and agriculture-specific characteristics (SCOR, ISA-95),

[2] Supply Chain Operation Reference (SCOR) model. (http://supply-chain.org/scor)
[3] Informatie Model Open Teelten, a Dutch information model developed around the 1990s.
[4] EDITeelt is a data standards that contain standardised messages for data exchange between multiple systems and is currently used by Dutch software vendors. (www.agroconnect.nl)
[5] AgroXML is a data standard that contains standardised data messages and is currently used by some German software vendors. (www.agroXML.de)
[6] https://www.isa.org/isa95/

or focus on a single view such as actor model, business control model, business process models (Fountas et al., 2006; Nash et al., 2009; Sørensen et al., 2010b) or data model (AgroXML, EDIteelt). As a result, these models do not sufficiently address the integration problems required to realise advanced farm management styles. They need to be extended and adapted to increase their usability for software engineers and to provide configuration support. Therefore, development of a reference model, to enable configuration of ICT Components into integrated farm management information systems, is required. However, to enable farm enterprise integration, focus should be both on technical aspects related to configuration and organisational aspects to enable aligned software development by different software vendors (Wolfert et al., 2010).

1.3 Hypothesis and research questions

Following the aforementioned solution directions, the hypothesis of this thesis is that farm enterprise integration to realise advanced farm management styles can be improved by designing a reference model for the development of ICT solutions based on ICT Mass Customisation with a Best-of-Breed approach. As a consequence, the adoption of advanced farm management styles is expected to increase, which can result in producing more and safe food in a sustainable way. The main research question can be put as follows:

How can reference architectures and models help to develop farm information systems that improve enterprise integration for advanced farm management styles?

This question can be split up into the following sub-questions:

1) *What is the cause and nature of integration problems at farm enterprises?*
2) *How can integration problems at farm enterprises be solved by a framework with reference architectures and models that include both technical and organisational aspects?*
3) *How can we substantiate that the framework will enable a solution for integration problems at farm enterprises?*

To solve these questions, a design-oriented research approach will be used that is described in the next section. Because this thesis is connected to the Dutch Program on Precision Agriculture the design will mainly focus on arable farming.

1.4 Methodology

This thesis follows a design-oriented research approach using case studies. First, the methodological background of this approach is provided in Section 1.4.1. Then, in Section 1.4.2, we explain how this methodology was applied to this thesis.

1.4.1 Design science and case study

A relatively new scientific discipline is Design Science including Design-Oriented Research (DOR). DOR primarily aims to produce an innovative design and applies typical concepts from design methodology (Hartog, 2012). The purpose of DOR is to create innovative artefacts that extend the boundaries of human and organisational capabilities (Hevner et al., 2004). These artefacts are developed based on a design process.

In Information System Research these artefacts are constructs, methods, models and instantiations (Hevner et al., 2004; March and Smith, 1995). To develop information system artefacts, guidelines of an Information System Research Framework can be followed (Hevner et al., 2004). In this Information System Research Framework three concepts are distinguished; the Environment, Information System Research and Knowledge Base (Hevner, 2007; Hevner et al., 2004). Within this framework, the Environment concept provides requirements regarding the designed artefact. Within the Information System Research concept artefacts are designed. These artefacts can be assessed by verification (does it fit the requirements?) and validation (is it useful for the Environment?). The design process within the Information System Research concept should use foundations or methods from the Knowledge Base concept. After completion of the Information System Research, the artefact should be an addition to the knowledge base and be applicable in the appropriate Environment. An artefact is an addition to the knowledge base when it proves that it provides in its environment

(1) a new solution to an existing problem that has not been solved before (2) a better solution to a problem that has already been solved. The proof of the applicability of an artefact in its environment can be tested by quantitative and qualitative research. In this research we do qualitative research with case studies to understand relationships between our artefacts and the Environment (Yin, 2009).

Case studies are conducted to test generality aspects of the artefacts and to validate its usability. A case study may be understood as the intensive study of a case where the purpose of that study is, at least in part, to shed light on a larger class of cases (a population) (Gerring, 2006). A case study method refers to a research strategy which focuses intensively on individual cases to draw insights about causal relationships in a broader population of cases (Poteete et al., 2010). Case research is particular appropriate for certain types of problems: those in which research and theory are at their early, formative stage (Benbasat et al., 1987) and "sticky practices based problems where the experiences of the actors are important and the context of action is critical" (Bonoma, 1985). The characteristics make a case study research seamlessly suitable for DOR in Information System research because a design is a new phenomenon that must the tested in its natural setting (Environment) (Hevner, 2007; Hevner et al., 2004). In a case study, data to formulate the requirements and test the applicability of a design can be based on multiple data collection methods to gather information from one or more entities (people, group or organisation) (Benbasat et al., 1987; Bonoma, 1985; Yin, 2009).

A case study research approach has been chosen as a research method in this thesis. The cases that we selected in this research are chosen in such a manner that they shed light on a larger class of cases. For each case we aim to explain how the properties of the selected cases are representative for the rest of the population.

1.4.2 Approach

Following the three research questions, this thesis goes through three steps of a design cycle, starting with (i) a problem identification and validation, (ii) design of a framework and (iii) testing and evaluating this framework in order to proof it has contributed to solving the problem

(Figure 2). However, in designing you can usually distinguish 'wheels within wheels' (Simon, 1977) so in every step the same cycle is reiterated using specific use cases. Each step uses existing knowledge, contains a design and evaluation of the artefacts in its environment. As indicated in Figure 2, four basic artefacts have been developed that contribute to the knowledge base and that help to answer the research questions of this thesis. For each artefact a description is provided including the interactions with the environment.

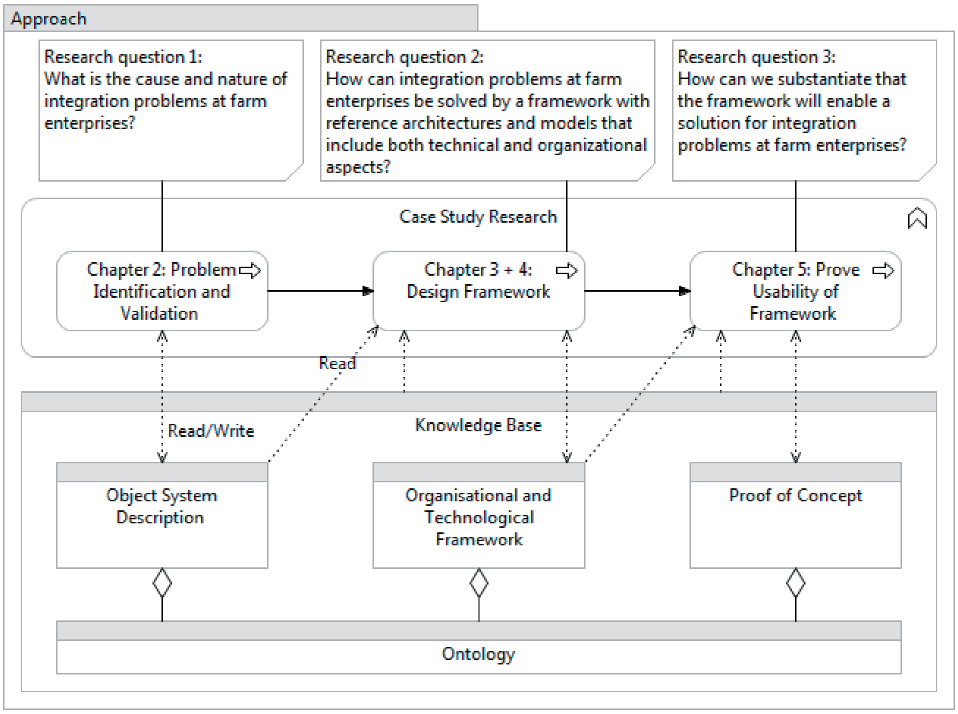

Approach
Research question 1: What is the cause and nature of integration problems at farm enterprises?
Research question 2: How can integration problems at farm enterprises be solved by a framework with reference architectures and models that include both technical and organizational aspects?
Research question 3: How can we substantiate that the framework will enable a solution for integration problems at farm enterprises?
Case Study Research
Chapter 2: Problem Identification and Validation
Chapter 3 + 4: Design Framework
Chapter 5: Prove Usability of Framework
Read
Read/Write
Knowledge Base
Object System Description
Organisational and Technological Framework
Proof of Concept
Ontology

Figure 2: Approach in this thesis.

<u>Ontology:</u> An ontology is created that provides a concise and precise, formal specification of the object system (farm, farm management, business process, resources including software) that is necessary for a shared understanding and effective communication between all stakeholders. This ontology is used to describe the object system, organisational and technological framework and the proof of concept and is extended and used in different steps of the research. This ontology can be found in Appendix A.

Object System Description: The object system description (Chapter 2) is described using a method and a reference model called 'Reference Architecture for Agricultural Enterprises' (RAAgE 1.0) that uses the ontology. The method provides a structural manner to identify problems related to farm enterprise integration. In this method RAAgE 1.0 is used to provide insight into the enterprise architecture of farm enterprises. With this method and RAAgE 1.0 problems in farm enterprise integration were systematically identified and analysed. The case study to identify these problems included three arable farm enterprises. All three farm enterprises cultivate potatoes and each farmer was interested in improving farm operations by implementing precision agriculture. With the farmers semi-structured interviews were conducted in different steps of the method. The identified case-specific problems were analysed and described in a generic manner. These generic problems were validated with national and international experts. Based on this discussion we concluded that the found bottlenecks were generic enough to shed light on other cases. These problems are presented and validated in Chapter 2. Based on the identified problems solution directions are determined that were used as requirements for the organisational and technological framework to be developed.

An Organisational and Technological Framework: This framework provides insight into technical and organisational aspects to enable ICT Mass Customisation using Best-of-Breed for farm enterprise integration. The organisational part of the framework is called a Farm Software Ecosystem which enables multiple organisations to collaborate on developing components that can be configured into an integrated farm information system (Chapter 3). The framework constitutes technical artefacts, such as a reference model for farm enterprise architectures (Chapter 2). To validate the design two Dutch initiatives in which Farm Software Ecosystems are being established were selected as use cases. The Chief Technology Officers of both initiatives have been individually interviewed with semi-structured interviews to validate the design. Moreover, the design was used to map these current initiatives. As there were not many Farm Software Ecosystems present at that time, these two were representative to provide new insight into this new organisational structure.

The framework was further extended by designing the technological framework for configuration of ICT Components[7] in Chapter 4 (RAAgE 2.0). In its second version the Reference Architecture can (i) help farmers in creating business process configurations that reflect their practices and help them to improve their business processes (business process re-configuration), (ii) enhance modular based software development and (iii) support farm enterprises in the selection and configuration of the resources that are needed to execute their (improved) business processes. This design was validated by a use case on late blight protection in potatoes. This specific case is representative for other cases in agriculture as it requires ICT Components of multiple vendors to support the business process.

<u>A proof of concept (Instantiation):</u> This proof of concept instantiates the previous artefacts by developing prototype software aligned with ICT Mass Customisation and Best-of-Breed (Chapter 5). This proof of concept has been developed by a case study again on late blight protection in potatoes but now applied at a specific farm in the Netherlands. Weather data and crop protection data (date and time) were used to validate the usability of the prototype software. In this proof of concept the design, configuration and usage of ICT Components are presented and show how farm enterprise integration can be improved. In this way it is expected that substantial evidence was provided that the developed artefacts effectively contribute to developing ICT Components that improve farm enterprise integration for advanced farm management styles, which is the main research question of this thesis.

These artefacts are presented in detail in the remainder of this thesis. In the final Chapter 6 we present a general discussion in which we revisit the research questions.

[7] An *ICT Component* is an *Application Component (Composite Application Component* or *Atomic Application Component)* that is deployed on a *Node* and which supports one or more *Business Processes* of a Business *Actor*. An *ICT Component* can act as a *Farm Information System* or be part of a *Farm Information System.* (more definitions, including definitions of the words in italic, can be found in Appendix A – Ontology)

2

Improving arable farm enterprise integration

Review of existing technologies and practices from a farmer's perspective

This chapter is based on the published journal article:

Kruize, J.W., Robbemond, R.M., Scholten, H., Wolfert, J., & Beulens, A.J.M., Computers and Electronics in Agriculture, 2013, Volume 96, Pages 75-89.

Abstract

Current consumers are demanding food that is produced more sustainably, safely and transparently. To meet these demands farm enterprises need to improve production. To support this, a variety of high-tech tools (ICT Component) are available. Despite this availability, farmers face difficulties in adopting and integrating them effectively. This chapter presents a method to identify bottlenecks that hinder arable farm enterprise integration, in which a reference model serves as a base for models describing arable farm enterprise architectures. This reference model, described in a standard modeling language, shows the interrelations between the business, application and technology layers of farm enterprises. It was validated by two experts with expertise in the creation of reference models of arable farms. This reference model was used to create enterprise architectural descriptions of three arable farms in the Netherlands, showing a series of bottlenecks. The bottlenecks that we found are: nonexistence of services and interfaces to support farm processes, (partly) overlapping applications, inability of applications to share a data repository, skills required of farmers to exchange data and unavailability of business services to configure ICT Components of different vendors. These bottlenecks, applicable for potato production in the Netherlands, were validated with national and international experts who have expertise in arable farming. Based on this validation, we conclude that these bottlenecks are valid for other cultivations and in other countries. This chapter shows that our approach, i.e. describing the arable farm enterprise architecture, is also a valuable method to gain insight in different aspects of arable farm enterprise integration and can be used to make the first steps towards prioritizing and removing bottlenecks.

Keywords: precision agriculture; farm enterprise integration; reference model; architectural description; enterprise configuration

2.1 Introduction

Agri Food Supply Chain Networks (AFSCNs) face global challenges that require a re-evaluation of current practices. Their first challenge is to feed a rapidly expanding world population. According to the United Nations, the seven billionth inhabitant arrived on the 31st of October 2011 (Crossette and Kollodge, 2011), and by about 2050 the expected world population will exceed 10 billion. In parallel to the challenges this raises, AFSCNs have others. For example, consumers are demanding high quality food that is produced more sustainably, safely and in a production chain that is transparent (Grunert, 2005; Seuring and Müller, 2008). These demands have impact both on food production operations, their management and on information exchange among AFSCN actors[8], who must cooperate and change. This will require farm enterprises to change considerably, because production agriculture always has performed and always will perform essential tasks in the web of the food industry (Kinsey, 2001).

Farm managers, as a result, must address new requirements, for example around improving quantity and quality while reducing environmental impact. Therefore, they will need more control over their production system. Farm managers must also be able to guarantee that their enterprises conform to numerous rules and regulations, and be able to monitor their operations and products and to share this information in AFSCNs for tracking and tracing purposes.

Our research aims to support arable farm enterprises in improving production and meet the previously mentioned requirements. We focus on arable farming, both because it is the basis for food production, including meat and dairy production, and because over the few last years, arable farm enterprises have begun to improve both tracking and tracing and food safety. However, adapting arable farm business to meet requirements related to sustainability, improved quality and quantity, reducing the environmental impact remains a huge challenge. These tasks are likely to be facilitated by a management style that takes into account in-field variability of soil and crop. This kind of field

[8] Actor is defined as an organizational entity that is capable of performing behaviour. (more definitions can be found in Appendix A – Ontology)

management, also known as precision agriculture, aims to increase the profitability of crop production while simultaneously reducing the negative environmental impact by adjusting applications rates of agricultural inputs according to local needs (Pierce et al., 1999). Precision agriculture has already shown its potential to increase yields and/or reduce environmental impacts (Bongiovanni and Lowenberg-Deboer, 2004). Practicing precision agriculture requires an increase in the granularity of decision making in time and space. Therefore, detailed information regarding crops and fields is needed.

To practice precision agriculture, a configuration of various ICT Components, such as sensors, Global Positioning Systems (GPSs), terminals, applications, and variable rate implements, is required within each farm enterprise. Moreover, even pictures from satellites are used to provide data about the crops on the field. All these ICT Components, which are developed by different organizations, provide (geographical) data about the crops or require (geographical) data for variable rate application. Consequently, precision agriculture is intrinsically information intensive. Managing this enormous amount of (geographical) data is difficult and therefore time consuming (Fountas et al., 2006). This can be addressed through information technologies, which have the potential to support farmers with this challenge (Schiefer, 2004).

The adoption of precision agriculture as an arable farm management style is not common. While several enterprises have a set of ICT Components that support precision agriculture, most enterprises are unable to manage in-field variability. Researchers have found multiple explanations for the low adoption rate of precision agriculture (Jochinke et al., 2007; Lamb et al., 2008; McBratney et al., 2005; Pedersen et al., 2004; Reichardt and Jürgens, 2009). One of the main reasons is the lack of compatibility and interoperability[9] between ICT Components. Another

[9] *Interoperability* is understood as the ability of two implementations of a communication protocol (e.g. a transfer protocol, an interface) to communicate properly. (https://wiki.oasis-open.org/tab/InteropGuide)

is that actual business processes[10] are insufficiently (or not at all) supported by decision support tools. However, problems related to the integration of available ICT Components in the arable farm enterprise have been insufficiently described and analyzed in literature so far.

In this chapter we aim to give a detailed description of current technological problems related to arable farm enterprise integration. We develop a reference model to be used to create an architectural description of the object system (arable farm), describing the relations between the business layer and advanced ICT Components that (should) support these processes. We test the validity of the reference model by case studies to create detailed descriptions of bottlenecks in arable farm enterprise integration. Our results can be used to make the first steps towards prioritizing and removing these bottlenecks for arable farmers.

Background information related to arable farm enterprise integration can be found in section 2.2. A detailed description of the method to detect current bottlenecks can be found in section 2.3. To detect current bottlenecks, using case studies, we needed a clear and coherent description of the arable farm to identify these bottlenecks. The reference model that enabled the detection of these bottlenecks is presented in section 2.4. In section 2.5 a detailed description of a bottleneck, the identified bottlenecks and the validation of these bottlenecks is presented. In section 2.6 we discuss our results, conclude what hinders arable farm enterprise integration and provide recommendations how enterprise integration at arable farm enterprises can improve.

2.2 Background

Wolfert et al. (2010) proposed a method for organizing information and enterprise integration that consists of three steps: 1) analysis of the existing state ('as-is') of integration at various levels, 2) basic design of an integration framework and 3) iterative implementation by living labs.

[10] Business Process is defined as a B*ehavioural Element* that groups behaviour based on an ordering of activities. It is intended to produce a defined set of *Products* or *Business Services*. (more definitions, including definitions of the words in italic, can be found in Appendix A – Ontology)

For all three phases, reference information modeling was considered as an important mean to apply the method to different sectors. These reference models can support model users saving time and cost, while quality of the model to be constructed can be increased by the use of a reference model (Bussler et al., 2006). A related remaining question was how business process models can become more easily configurable in order to enable re-use and rapid adaptation of services to a changing, dynamic environment. A follow-up paper addressed this question for the fruit industry and focused on the third phase of iterative implementation (Verdouw et al., 2010b). This chapter applies to arable farming and mainly focuses on the enhancing role of reference (process) models to analyze the existing state (as-is) of arable farm enterprise integration.

Enterprise integration can be approached in various manners (Chen and Vernadat, 2004; Giachetti, 2004; Vernadat, 2007). First, enterprise integration can refer to integration between different enterprises (inter-enterprise integration) or within an enterprise (intra-enterprise integration) (Giachetti, 2004). Second, different integration levels can be distinguished, and here we consider physical system integration (hardware), application/data integration (software) and business integration. Integration on these levels can be achieved by unification (the possible standards are methods, architectures, constructs and reusable partial models) or by federation (the possible standards are interfaces, references models or ontologies) (Chen and Vernadat, 2004). To enable this integration, at enterprise level, a uniform and clear description of the enterprise is required, including the enterprise Information System (IS). The description of an Enterprise Architecture can facilitate this task.

In ISO/IEC 42010:2007, architecture is defined as: "The fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution." Enterprise architectures are used to facilitate the process of changing an organization from a baseline (*as-is)* to a target (*to-be)* architectural state. The baseline architectural description provides insight in the current (*as-is*) state of an enterprise. For example, it describes the relations between the current business processes applications and technologies of an enterprise. The target (*to-be*) architectural description provides insight in the required future state. As foundations of enterprise systems engineering, enterprise architectures have emerged as tools to help stakeholders manage system engineering and change, and can additionally be seen as complementary to software architecture, and as a way to document system wide organization and the business context in which software operates (Chen et al., 2008). To integrate the enterprise on a physical, application and business level over time, an enterprise architecture aggregates one or more models that describe a system in a certain state and is organized by one or more views. The models and views in this chapter are designed that provide insight into the *as-is* and *to-be* architectural descriptions of arable farm enterprises that can be used to detect bottlenecks hindering internal integration.

Recent papers related to arable farm enterprise integration generally describe aspects of application/data integration (software) and business integration. Available data integration papers aim to improve the exchange of data between ICT Components (Iftikhar and Pedersen, 2011; Nash et al., 2009; Steinberger et al., 2009). These papers focus on the data exchange between ICT Components, but do not describe other aspects of the ICT Components or the supported business processes. Moreover, some papers make standards and ontologies available that describe aspects of farm enterprises, to facilitate data exchange between ICT Components (ISO 11783,[11] AGRO XML,[12] AGROVOC,[13] EDI-teelt[14]). Those that relate to application integration investigate ICT Components that support farm management, Farm Management Information Systems (FMISs) (Fountas et al., 2009; Nikkilä et al., 2010; Sørensen et al., 2010a; Sørensen et al., 2011). These studies provide detailed insights in the requirements of FMIS and (conceptual) models of future FMISs. However, they do not describe the relation between the business processes and these ICT Components. In relation to business integration, available papers are related to arable farm business process modeling. These papers describe the information flows and decision aspects at the arable farm (Fountas et al., 2006; Sørensen et al., 2011; Sørensen et al., 2010b). However, they do not

[11] http://www.iso.org
[12] http://www.agroxml.de/
[13] http://aims.fao.org/standards/agrovoc/about
[14] http://www.agroconnect.nl/

describe the relation between these information flows and decision aspects and the ICT Components. Based on this literature we conclude that relevant aspects regarding arable farm enterprise integration are investigated. Still, current research has not provided models that have a whole farm focus regarding arable farm enterprise integration. Moreover, existing models lack representational power to include all relevant integration aspects of arable farm enterprises. Our contribution to the literature combines relevant aspects of integration and identifies problems in arable farm enterprises for the integration of advanced ICT Components.

2.3 Methodology

2.3.1 Finding bottlenecks in arable farm enterprise integration

To understand current problems in farm enterprise integration, we followed a design-oriented research approach based on a case study. This approach is modeled in Figure 3, which describes processes, actors, results and the relationships between these concepts.

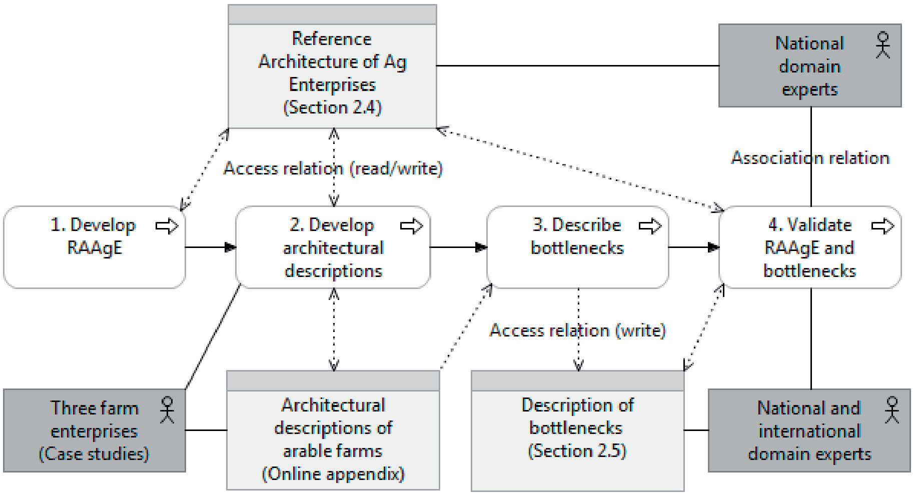

Reference Architecture of Ag Enterprises (Section 2.4)
National domain experts
Access relation (read/write)
Association relation
1. Develop RAAgE
2. Develop architectural descriptions
3. Describe bottlenecks
4. Validate RAAgE and bottlenecks
Access relation (write)
Three farm enterprises (Case studies)
Architectural descriptions of arable farms (Online appendix)
Description of bottlenecks (Section 2.5)
National and international domain experts

Figure 3: The approach, modeled in four process steps, to acquire insight in the current state of arable farm enterprise integration. The approach is modeled in the language ArchiMate (TheOpenGroup, 2012). See text for explanations.

To be able to detect bottlenecks in a systematic way, we developed: RAAgE, Reference Architecture of Agricultural Enterprises. The process *Develop RAAgE (1)* writes to the 'object' RAAgE, which is one of the results of this research. The next process, *Develop architectural descriptions (2*) is triggered, resulting in *Architectural descriptions of arable farms*. These architectural descriptions contain case-specific bottlenecks. Subsequently, the *Describe bottlenecks (3)* process is triggered, in which we describe the bottlenecks found in a more generalized manner. These generalized bottlenecks and RAAgE are validated by national and international experts in the *Validate RAAgE and bottlenecks (4)* process. The four steps are explained in more detail in the following subsections.

Develop RAAgE (Process 1)
Operations management is concerned with the design, execution and improvement of planning and control processes (Slack et al., 2010). Relevant insights into these processes can be acquired by specifying the architectural description of the farm, using a reference model. Although various reference models are available and have been reviewed (Verdouw et al., 2010a) and various architectural frameworks are available to support the design of the enterprise architectural description,[15] to the best of our knowledge no reference model was available to support designing consistent farm enterprise architectural descriptions. Hence, we created RAAgE.

Our first version of this reference model, i.e. RAAgE 1.0, modeled the architectural description and created views of arable farm enterprise architectures. RAAgE uses ArchiMate[16] as a standard language capable of representing enterprise architectures over time and closely linked to The Open Group Architectural Framework[17] (TOGAF) standard. The free and open source tool Archi[18] was used to develop RAAgE and the models describing the enterprise architectures and bottlenecks.

Conform ArchiMate, RAAgE distinguishes between the business layer, the application layer and the technology layer, and is able to describe

[15] http://www.iso-architecture.org/ieee-1471/afs/frameworks-table.html
[16] http://www.opengroup.org/subjectareas/enterprise/archimate
[17] http://www.opengroup.org/togaf/
[18] http://archi.cetis.ac.uk/

the interrelations between these layers. The business layer offers products and services to external customers, which are realized in the organization by business processes performed by business actors (TheOpenGroup, 2012). The application layer supports the business layer with application services that are realized by (software) applications (TheOpenGroup, 2012). The technology layer offers infrastructure services (e.g. processing, storage, and communication services) needed to run applications, realized by computer and communication hardware and system software (TheOpenGroup, 2012).

Reference models from different domains were used to create the business layer of RAAgE. Inspired by the functional modules in ISA95,[19] this layer categorizes farm processes into process groups.[20] These process groups are divided into management control and operating control layers. According to Anthony and Govindarajan (2006), management control is the implementation of strategies enterprises use. Operating control (task control) is defined as efficient and effective performance of individual tasks. Business processes related to the cultivation of crops, grouped in the process group *product management* were described using the Information Model Field Crops[21] (IMOT), a reference model from the 1990s. To extend the description of the process group product management and to create more up to date descriptions, we conducted interviews and meetings with farmers to create process descriptions.

The applications layer can be used to describe the applications part of ICT Components used for arable farming. Applications can be linked with business processes. In this method the business layer provides context about the appliance of an application. This has been demonstrated in a previous report (Robbemond and Kruize, 2011) in which a selection of state-of-the-art applications used at arable farm enterprises were modeled based on an earlier version of RAAgE.

The technology layer describes the technological part of ICT Components used in arable farming. Examples are e.g. servers,

[19] http://www.isa-95.com (ISA-95 part 1)
[20] In Archimate a *process group* is named a *function*.
[21] http://tinyurl.com/gwbpdco

terminals, tractors, implements and sensors. The technology layer offers among others, the structure and services that are used by applications.

The development process of the content of RAAgE was iterative. While developing descriptions of the arable farm enterprise architecture of the case studies, the content such as the process groups and processes, was constantly updated and renewed. Additionally, the validation of RAAgE changed the content of RAAgE, finally resulting in RAAgE version 1.0.

Develop architectural descriptions (Process 2)
A greater understanding of arable farm enterprise integration was achieved through three in-depth case studies of arable farm enterprises in the Netherlands. Data was gathered by conducting interviews with each farm manager that participated in the case study. Our research was delineated to processes and ICT Components in potato cultivation, in order to identify bottlenecks that hinder arable farm enterprise integration. Potato cultivation is one of the largest cultivated crops in the Netherlands (CBS, 2012) and is knowledge intensive and requires advanced ICT Components.

The method, based on TOGAF, used to develop the architectural descriptions of each case study is presented in Figure 4. It can be used to detect bottlenecks related to the cultivation of potatoes by first performing the *develop baseline (as-is) architectural description*, *developed target (to-be) architecture description* process, and then executing *the perform gap analysis* process between the *as-is* and *to-be* state. In this process, we created a detailed description of the difference between the baseline (*as-is*) architectural description and the target (to-be) architecture description. In one case we analyzed the baseline architectural description and suggested improvements to the farmer, identifying gaps or bottlenecks applicable to a larger group of farmers. Once gaps were identified, we searched for state-of-the-art ICT Components or other support available to fill these gaps in the *definition of roadmap components* process. Where no roadmap components (ICT Components) or support (services) exist that could fill a specific gap, we defined that as a bottleneck in arable farm enterprise integration. The identified bottlenecks describe the ways in which the state-of-the-art ICT Components were unsatisfactory for the farmer. A detailed description of bottlenecks was developed in this way, based on architectural descriptions. These bottlenecks, together with the architectural descriptions were discussed in the formal stakeholder review with the farm managers.

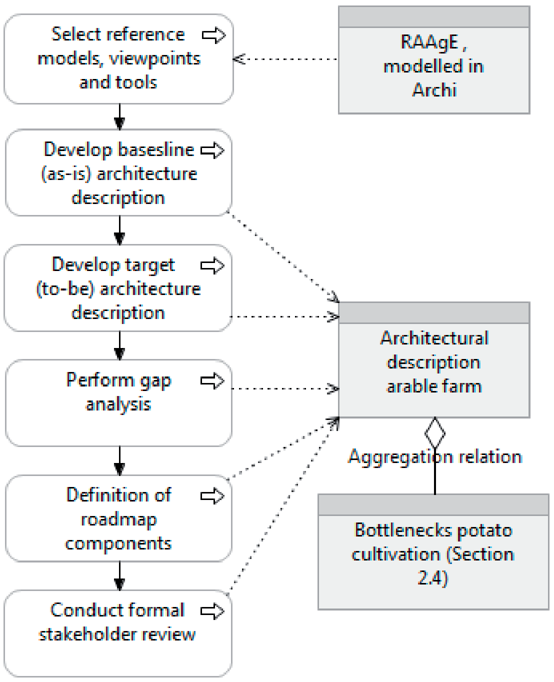

Select reference models, viewpoints and tools
RAAgE , modelled in Archi
Develop basesline (as-is) architecture description
Develop target (to-be) architecture description
Architectural description arable farm
Perform gap analysis
Aggregation relation
Definition of roadmap components
Bottlenecks potato cultivation (Section 2.4)
Conduct formal stakeholder review

Figure 4: Method to develop the architectural description of each arable farm based on TOGAF.

Describe bottlenecks (Process 3)

Based on the case studies, bottlenecks in enterprise integration for potato production in the Netherlands were identified. To be able to validate the bottlenecks found, we generalized and described these bottlenecks by grouping them and provide descriptions of these groups.

Validate RAAgE and generalized bottlenecks (Process 4)

Both RAAgE and the generalized bottlenecks are validated. In this section the validation approach is described. A more detailed description of the experts that have validated RAAgE and the generalized bottlenecks can be found in section 2.3.3.

To validate RAAgE 1.0 we used semi-structured interviews using a standard question form with qualitative questions. Each expert was interviewed separately. The questions provided answers about the most important elements of RAAgE (the structure and content). Additionally questions were asked about the added value of RAAgE, which provides the ability to create architectural descriptions of arable farms, in comparison to models that are known by these experts. These interviews can be found on the website on arable farm enterprise integration.[22]

To validate the generalized bottlenecks, we discussed these with each expert using semi-structured interviews and a standard question form. First, we explained each bottleneck and asked if the bottleneck was understood. Then we asked whether the bottleneck also occurs at arable farms that cultivate potatoes in the country/countries in which they have expertise. We then asked if they thought that the bottleneck applied for other cultivations. Finally, we asked if they are aware of bottlenecks not found in this research. These interviews can be found on the website on arable farm enterprise integration.

2.3.2 Description of the characteristics of the case studies

All three in-depth case studies were arable farm enterprises cultivating potatoes and interested in improving farm operations by implementing precision agriculture. The case studies were selected from our network based on variation in farming characteristics. First, they differ with regard to the cultivating purpose of the potatoes, i.e. either seed, ware or starch potatoes. Furthermore, each is located in a different region and cultivates on a different soil type. Finally, the farmers have different ICT Components and levels of enterprise integration. The arable farm manager of case study 1 just started using new ICT Components, such as GPS and section control on the sprayer. The arable farm manager in case study 2 has been interested in precision agriculture for more than 10 years and already uses some related ICT Components. The arable farm manager in case study 3 has detailed knowledge of precision agriculture and is able to manage in-field variability. A summary of the case studies is described in Table 1.

[22] http://tinyurl.com/gwbpdco

Table 1: Conducted case studies.

Case	Age farm manager	Primary crop	Cultivated Ha	Location[23]	Soil type
Case 1	55	Potatoes (starch +seed)	35	Groningen	clay + sand
Case 2	48	Potatoes (seed)	35	Groningen	Clay
Case 3	31	Potatoes (ware)	400	Noord-Brabant	Sand

2.3.3 Expert validation

Validating RAAgE

Of the two experts in modeling of arable farms who have validated RAAgE, one works for a research institute in the Netherlands. He has more than 25 years of experience in precision agriculture and has been involved in several projects, in which models are created that describe arable farms. Furthermore he is involved in the continuation of the ISO-11783 standard. The other expert has been working as software architect for more than 5 years, and for the last 2 years has been leading a group of software developers working on a new type of FMIS.

Validating the bottlenecks

Similarly, the bottlenecks were validated by national and international experts with expertise in arable farming.

The first (national) arable farm expert who validated the bottlenecks is employed at one of the largest arable farm in the Netherlands. This arable farm, with 1300 cultivates ha, has been involved in arable farm automation for many years. In this enterprise, automation and precision agriculture is continuously discussed and to some extent implemented. This expert has knowledge about arable farming in the Netherlands.

The second (international) arable farm expert is employed by a large machinery factory for potato production. The factory sells machines to various countries. The expert is employed as a System Engineering and Simulation and Instrumentation Manager. This expert has knowledge about potato production in Germany.

[23] All are in The Netherlands.

The third (international) arable farm expert is an internationally renowned researcher who is working at a university in Greece. During his research career he has been located in different countries. This expert has knowledge about arable farming in Greece, the United Kingdom, Denmark and the United States (Indiana).

The fourth (international) arable farm expert is an internationally renowned researcher currently employed by a university in Denmark. In prior research he has been involved in international projects such as the European future farm project. This expert has knowledge about arable farming in Denmark, Germany and Finland.

2.4 Reference Architecture of Agricultural Enterprises

The Reference Architecture of Agricultural Enterprises (RAAgE) is designed to model the arable farm enterprise architectural description in a clear and coherent manner. RAAgE 1.0 is presented here. First, we will clarify the structure of RAAgE by explaining the relations between the business layer, application layer and technology layer. Then, we concentrate on describing the business layer. The description of the farm process groups and farm business processes can be found in section 2.4.2. Next, a more detailed description of process group product management is described in section 2.4.3. Following, how we modeled ICT Components of farms using objects from the application layer and technology layer. This can be found in section 2.4.4. Finally, the results of the validation are presented in section 2.4.5.

2.4.1 The structure of RAAgE

RAAgE enhances the modeling of the architectural description of an individual arable farm enterprise. RAAgE is based on the modeling language ArchiMate (TheOpenGroup, 2012), in which three main types of elements are defined: active structure elements, behavioral elements and passive structure elements. An active structure element is an entity capable of performing behavior. A behavioral element is a unit of activity performed by one or more active structure elements. A passive structure element is an object on which behavior is performed. Additionally, Archimate distinguishes between an external view and the internal view of systems. Hidden within the internal view are services, which are a unit of functionality that a system exposes to its environment, which hides internal operations. The external view includes the interface, a point of access to one or more services.

In Figure 5, the structure of RAAgE can be found, and a detailed description of version 1.0 is publicly accessible online and can be found on the website on arable farm enterprise integration. The structural concepts are the arable farm enterprise and the ICT Components. The arable farm is modeled as an actor, that is, as organizational entities capable of performing behavior (TheOpenGroup, 2012). The arable farm is a composition of process groups, business processes, and ICT Components. A farm process group is defined as an element that groups a certain type of behavior based on required skills and resources, for example farm processes that require skills and/or resources related to inventory management, product management or accounting. The farm business process is a behavioral element that groups behavior based on an ordering of activities. The intent of business processes is to produce a defined set of products or business services, examples are *Design crop plan*[24] and *Cultivate crop*. These business processes in turn can be supported by ICT Components, for example terminals, software applications and implements.

[24] A crop plan describes what crops will be grown on what fields in a certain time period. (more definitions, including definitions of the words in italic, can be found in Appendix A – Ontology)

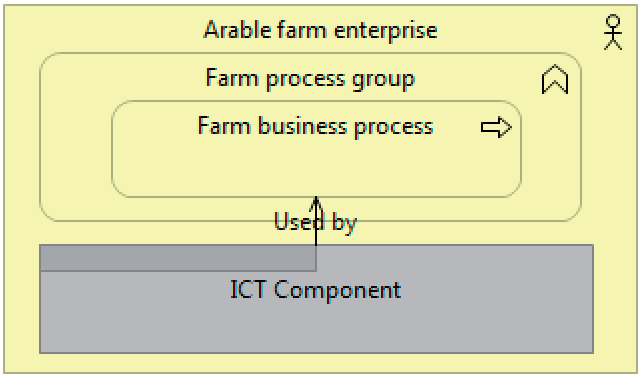


Figure 5: The structure of RAAgE showing that an arable farm enterprise consists of a Farm process group (Figure 6 and Figure 7) which includes Farm business process, described for the process group Product management in section 2.4.3. Farm business processes use ICT Components, described in section 2.4.4.

RAAgE is not intended to have complete descriptions of entities within an arable farm, nor to describe all business processes, process groups and ICT Components. It provides a model to structure present farm entities and includes a description of common farm process groups and farm business processes.

2.4.2 Farm process groups and farm business processes

The farm process group and farm business processes relevant to running an arable farm enterprise are modeled at a high level of abstraction. An arable farm enterprise has to execute farm business processes, part of a farm process group, to manage the farm enterprise. In the reference model the farm business processes are grouped on required skills and resources. In total, ten different specializations of the generic *farm process group* are identified (see Figure 6). A more detailed description about the specializations of the farm process group can be found on the website on arable farm enterprise integration or in Robbemond and Kruize (2011).

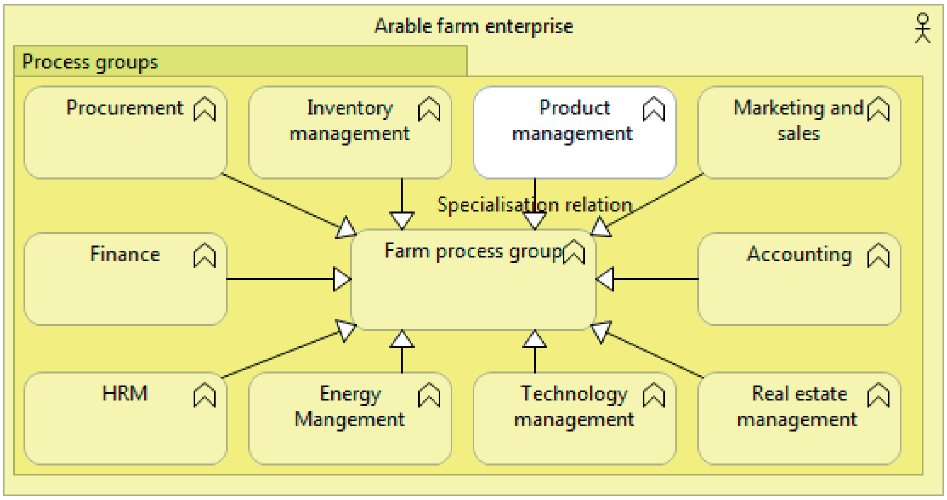


Figure 6: Specializations of the Farm process group. The generic structure, which is inherited by each of the farm process group specializations, can be found in Figure 7.

The farm processes group consist of farm business processes that can be further divided into management control and operating control. Management control processes concern the implementation of strategies and the evaluation and optimization of these strategic goals. The process descriptions included in RAAgE aim to describe, not complex processes for long term (strategic) planning, but tactical planning processes. Within the management control layer the generic farm processes are *Design management control plan*[25] (tactical planning), *Evaluate management control plan* and *Optimize management control plan* (see Figure 7). The operating control processes focus on the efficient and effective performance of individual operations and tasks. Within the operating control layer, the generic *Operating control process* includes the generic farm sub-processes; *Design operating control plan* (operational planning), *Execute operating control plan* and *Check operating task(s)*. These can be specialized as processes that are part of a specific process group. This move from generic to particular within these process groups and business processes allows the depiction of planning, monitor and control cycles. The processes related to

[25] The management control plan describes strategies enterprises will use to achieve its goals. (more definitions can be found in the Appendix A – Ontology)

cultivation, relevant for the production of crops, all included within the process group product management, are described in relatively greater detail (see section 2.4.3). This process group describes both generic and more specialized entities. With RAAgE we have focused on describing the management control and operating control processes of the process group product management.

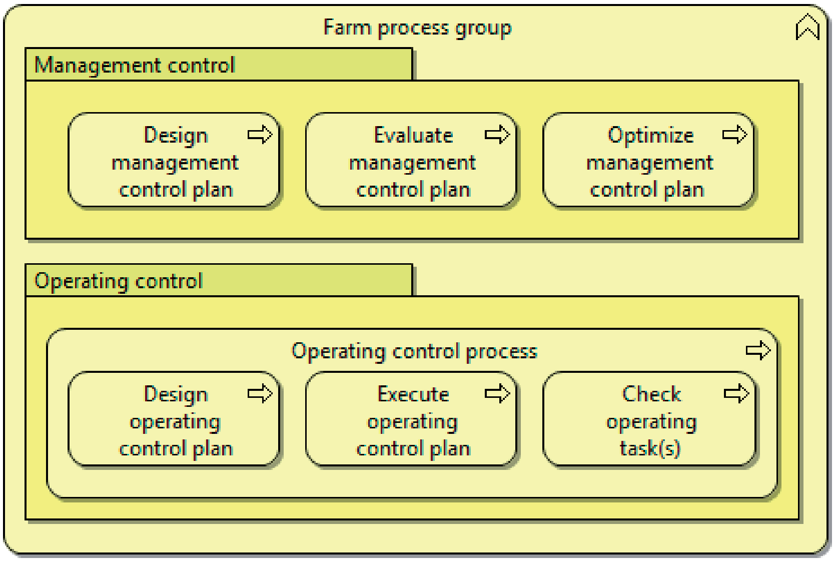

Farm process group
Management control
Design management control plan
Evaluate management control plan
Optimize management control plan
Operating control
Operating control process
Design operating control plan
Execute operating control plan
Check operating task(s)

Figure 7: The *farm process group* and each of its ten specializations are divided in a management control layer and operating control layer. The relations between processes in the management control and operating control layer of the process group Product management are depicted in Figure 11.

2.4.3 The process group product management

The process group product management is a specialization of the generic farm process group and consists of a management control layer and an operating control layer. The management control layer consists of the processes *Design product management plan*, *Evaluate product management plan* and *Optimize product management plan*. These processes aggregate sub-processes that are related to the planning, evaluation and optimization of the cultivation and processing of crops (see Figure 8). The operating control layer consists of the processes *Cultivate crop* and *Process crop*. These processes aggregate planning, executing and checking processes. The sub-processes of the processes related to cultivation of crops can be found on the website on arable farm enterprise integration.

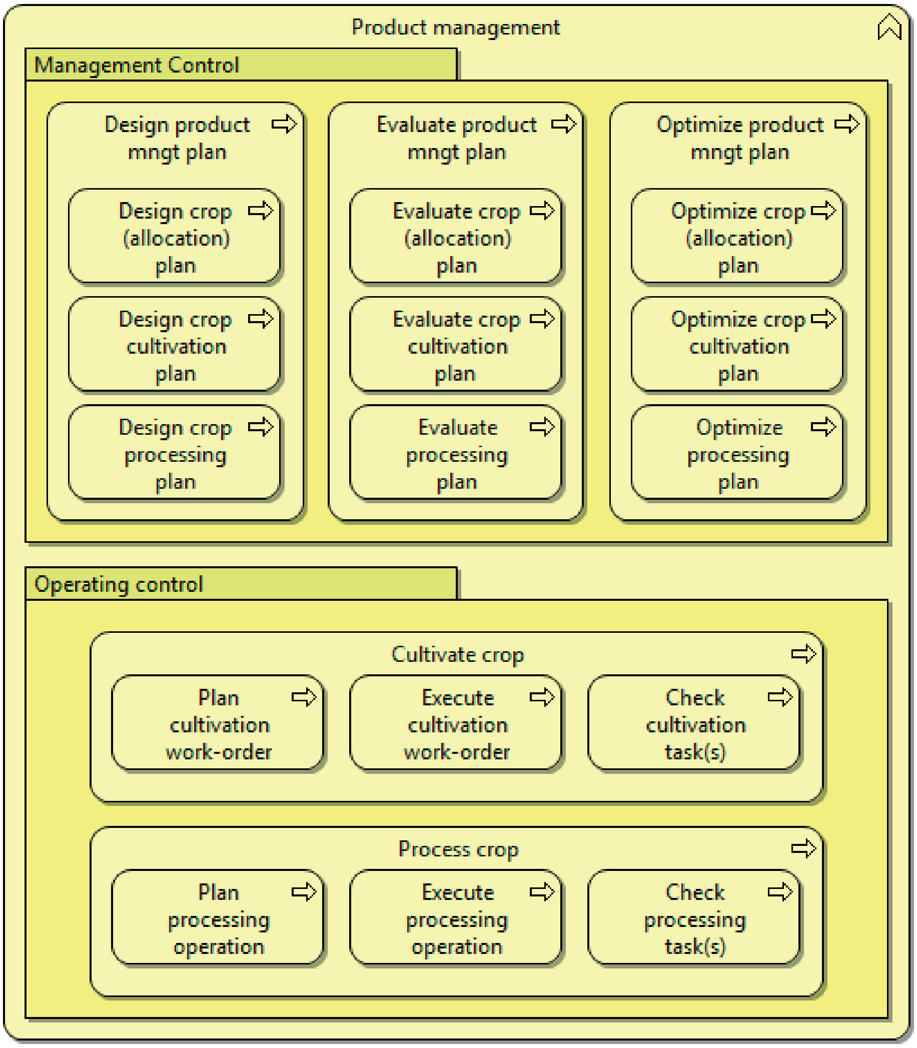

Product management
Management Control
Design product mngt plan
Design crop (allocation) plan
Design crop cultivation plan
Design crop processing plan
Evaluate product mngt plan
Evaluate crop (allocation) plan
Evaluate crop cultivation plan
Evaluate processing plan
Optimize product mngt plan
Optimize crop (allocation) plan
Optimize crop cultivation plan
Optimize processing plan
Operating control
Cultivate crop
Plan cultivation work-order
Execute cultivation work-order
Check cultivation task(s)
Process crop
Plan processing operation
Execute processing operation
Check processing task(s)

Figure 8: The process group Product Management and the included processes and sub-processes.

The identified operating control processes in the process group product management are the Cultivate crop and Process crop processes. Both of these processes can be specialized in the cultivation and processing of different crops. Specializations of the Cultivate crop process is depicted in Figure 9. Thus, the sub-processes of the *Cultivate crop* process (*Plan* *cultivation work-order,*[26] *Execute cultivation work-order* and *Check cultivation task(s))* can be related to different cultivated crops. This enables RAAgE to model processes specific for a cultivation (e.g. Cultivate grain, Cultivate potatoes, Cultivate onions).

Additionally, the *Plan cultivation work-order*, *Execute cultivation work-order* and *Check cultivation task(s),* and *Plan processing operation, Execute processing operation* and *Check processing task(s)* can each be specialized. The specialization relation of the *Execute cultivation work-order* is depicted in Figure 10.

For example, the Cultivate ware potatoes process requires a Plan crop protection work-order specifying the resources used within a specified time. This Crop protection work-order consists of one or more Jobs assigned to a Operator. A Job describes the Task to be executed and the order (route) of these Tasks. This Crop protection work-order is executed by the Execute crop protection work-order process. The data collected during the Execute crop protection work-order process can be checked by the Check crop protection task(s) process. A Task is the execution, within a 'part field', of one or more Operations. All these specializations of the sub-processes, including definitions, of the Cultivate crop process can be found in RAAgE, published on the website on arable farm enterprise integration.

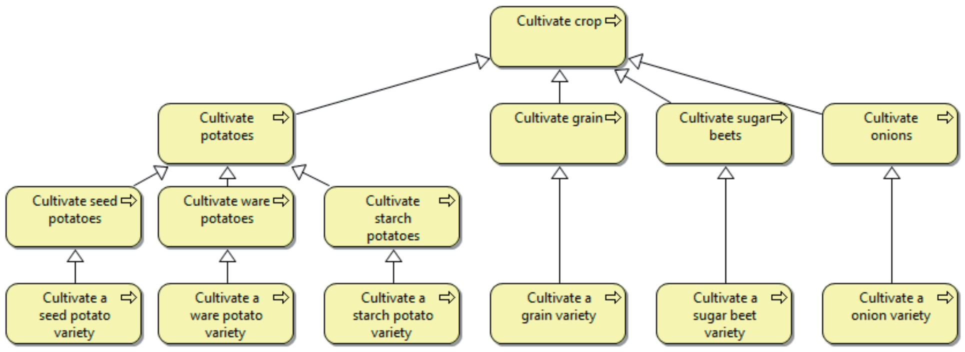

Cultivate crop
Cultivate potatoes
Cultivate grain
Cultivate sugar beets
Cultivate onions
Cultivate seed potatoes
Cultivate ware potatoes
Cultivate starch potatoes
Cultivate a seed potato variety
Cultivate a ware potato variety
Cultivate a starch potato variety
Cultivate a grain variety
Cultivate a sugar beet variety
Cultivate a onion variety

Figure 9: The specializations of the Cultivate crop process.

[26] A *Work-Order* specifies what resources are used within a specified time for a specific crop to achieve a certain goal. A Work-Order consists of one or more *Jobs* that are allocated to a *Operator*. (more definitions, including definitions of the words in italic, can be found in the Appendix A – Ontology)

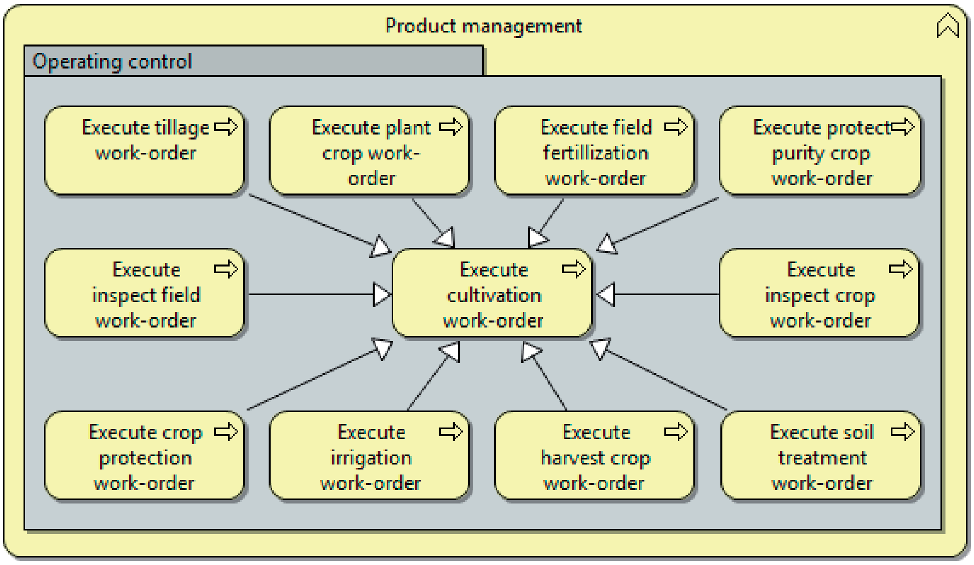

Product management
Operating control
Execute tillage work-order
Execute plant crop work-order
Execute field fertillization work-order
Execute protect purity crop work-order
Execute inspect field work-order
Execute cultivation work-order
Execute inspect crop work-order
Execute crop protection work-order
Execute irrigation work-order
Execute harvest crop work-order
Execute soil treatment work-order

Figure 10: Specializations of the Execute cultivation work-order.

To provide more insight in the process group product management, a planning, monitoring and control cycle is described (see Figure 11). This cycle is depicting the relation between the management control layer and the operating control layer of the process group *product management*. Within this cycle, the sub-process *Design crop cultivation plan* results in the *Crop cultivation plan* that determines the *Plan cultivation work-order* process used as an input for the *Execute cultivation work-order*. During the *Execute cultivation work-order, Work-order information* is gathered, which is then checked with *Task benchmark data* to produce the *Processed task information* read by the *Evaluate crop cultivation plan.* This completes the *Optimize crop cultivation plan.* This optimization can be done within the cultivation year or can be used to improve the *Design crop cultivation plan* in a next *Cultivate crop* year. Within the overall cycle, planning, monitoring and control cycles can be delineated. Such cycles are especially clear in the *Execute cultivation work-order*, but not all planning, monitoring and control cycles are currently described in RAAgE 1.0.

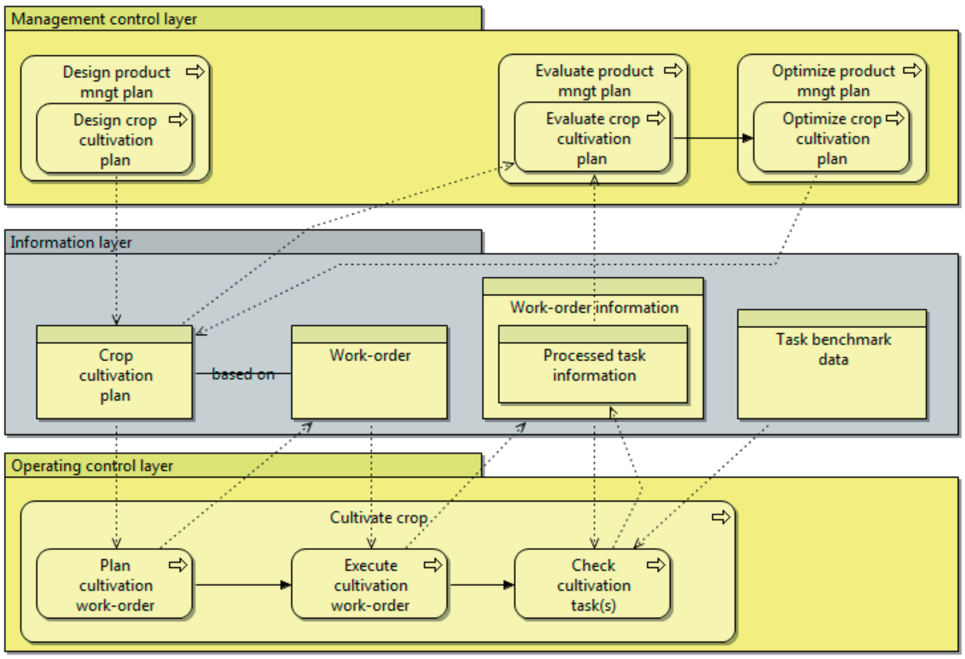

Management control layer
Design product mngt plan
Design crop cultivation plan
Evaluate product mngt plan
Evaluate crop cultivation plan
Optimize product mngt plan
Optimize crop cultivation plan
Information layer
Crop cultivation plan
based on
Work-order
Work-order information
Processed task information
Task benchmark data
Operating control layer
Cultivate crop
Plan cultivation work-order
Execute cultivation work-order
Check cultivation task(s)

Figure 11: Relation between the management control layer and the operating control layer of the processes related to the cultivation of a crop. The processes can read or write information to information objects.

2.4.4 Describing the arable farm ICT Components

At the arable farm enterprise different types of ICT Components are used to support the Farm business processes. Examples are software applications executed on a desktop computer, software applications executed on a server, implements, tractors, terminals, etc. These ICT Components, and the electronic information exchange between these ICT Components, can be modeled using RAAgE (see Figure 12). The context within which the ICT Components are used is provided by the farm process group and farm business process (business layer).

The ICT Components and electronic information exchange between ICT Components can be described in more detail using elements from both the application layer and the technology layer described in ArchiMate (TheOpenGroup, 2012). The ICT Components concept is introduced to improve communication with farmers about the ICT Components used at the farm. The concepts depicted in Figure 12 can be used to model the enterprise architectural description of individual arable farm enterprises.

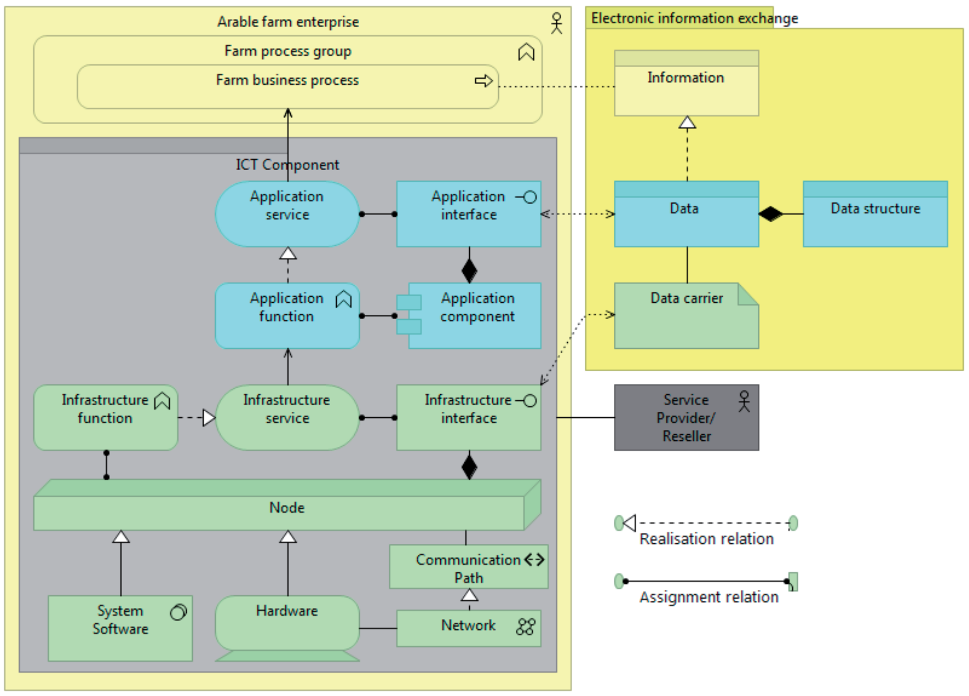


Figure 12: View on the elements describing arable farm ICT Components and elements for electronic information exchange. The context in which the ICT Components are used can be described by elements from the business layer (top), the ICT Components are described by elements from the application layer (middle) and the technology layer (bottom).

Figure 12 consists of three important parts: the *farm enterprise* (including the elements *Farm process group*, *Farm business process* and *ICT Components)*, elements to describe ICT Components in detail and *Electronic information exchange* elements. The elements *farm process group* and *farm business process* have been presented. Now, we focus on the elements to describe ICT Components and to exchange electronic information.

Since the ICT Components modeled here are able to exchange data, they are described by elements from the application layer (middle) and from the technology layer (bottom). The concepts in the application layer are; *Application service*, *Application function*, *Application interface*, *Application component*, *Data* and *Data structure*. An *application service* is an externally visible unit of functionality, provided by one or more components, exposed through well-defined interfaces, and meaningful to the environment (TheOpenGroup, 2012). The service concept provides a way to explicitly describe the functionality that components share with each other and the functionality that they make available to the environment. An *application function* is a behavior element that groups automated behavior that can be performed by an application component (TheOpenGroup, 2012). An *application component* is a modular, deployable, and replaceable part of a software system that encapsulates its behavior and data and exposes these through a set of interfaces (TheOpenGroup, 2012). An *application interface* is the point of access where an application service is made available to a user or another application component (TheOpenGroup, 2012). The *application interface* can read/write data to the *data* object. The *data* object is a passive element suitable for automated processing (TheOpenGroup, 2012). The *data structure* describes the *data* and is the so called metadata.

The concepts from the technology layer are; *Infrastructure function*, *Infrastructure service*, *Infrastructure interface*, *Data carrier*, *Node*, *System software*, *Hardware*, *Network*, *Communication path.* An *infrastructure function* is a behavior element that groups infrastructural behavior according to the node by which it can be performed (TheOpenGroup, 2012). An *infrastructure service* is an externally visible unit of functionality, realized by an *infrastructure function*, exposed through well-defined interfaces, and meaningful to the environment. An *infrastructure interface* is a point of access where *infrastructure services* are made available to another *node.* An *infrastructure interface* can read/write to a *data carrier.* This *data carrier* is a physical piece of data that is used or produced in a software development process or by deployment and operation of a system (TheOpenGroup, 2012). *A node* is a computational resource upon which artifacts may be stored or deployed for execution (ArchiMate, 2012). Examples of *nodes* are the electronic control unit (ECU) of an implement and the hardware and system software of a terminal. The *node* can be seen as a specialization of *system software* and *hardware* (device), including a *connection* (network) which realizes a *communication path*.

ICT Components that can be modeled with RAAgE are able to exchange electronic information. Examples of information are Crop growth information or a Crop cultivation plan. This information has relevance from a business perspective. This information is realized by *data*. The *data* object is composed of a *data structure* and associated with a *data carrier.* Both the data structure and the data carrier should be standardized, to enable automated processing of messages between ICT Components. Standardization would also allow application components of different tool vendors to use a shared repository to access data.

2.4.5 Validation of RAAgE 1.0

RAAgE was validated by two experts with expertise in arable farm modeling using a questionnaire, which can be found website on arable farm enterprise integration. RAAgE was then adapted based on the recommendations of these experts. This validation, together with improvements, is described below.

The structure

The experts answered questions about the structure and potential improvements. While results indicated the structure matches with its intended purpose, more extensive description of the management control layer and operating control layer seemed desirable. Suggested improvements were to compare the structure of IMOT with the structure of the RAAgE business layer and to extend the current descriptions by creating more relational diagrams. For example, while currently the relational diagram of the process group *product management* is provided, new versions will include relational diagrams for the other process groups as well.

The content

The experts were also questioned about the content of the process groups, the completeness of the process groups/processes and the possibility of RAAgE to describe arable farm ICT Components. Based on their answers we conclude that the wide range of processes described seems rather complete. Especially the process group product management has been described in detail, and the ICT Components used in arable farming can be described clearly.

Added value

The added value of RAAgE was validated by questioning the experts about figures describing the architectural descriptions of the case studies. This determined whether the figures supported understanding, whether RAAgE supports modeling of enterprise architectural descriptions and whether the figures improve communication related to arable farm enterprise integration. Based on the answers we conclude that RAAgE supports understanding the architectural description and can improve communication about arable farm enterprise integration. The experts concluded that RAAgE enables modeling of (commercial) ICT Components in relation to arable farm business processes. Therefore, these models offer common ground for different parties to discuss ICT Component integration.

Improvements
The experts were asked to provide a judgment of RAAgE and to suggest improvements. It was suggested to extend RAAgE by describing more arable farm processes and be more compliant with IMOT, which describes the business layer (including the planning stages) more comprehensively. Additionally, since the intent is to improve integration by sharing data between ICT Components, data should be described or linked with data standards. Although RAAgE could be improved, the experts conclude that RAAgE and the chosen approach is a good step towards a common ontology in arable farming and can enable arable farm enterprise integration.

2.5 Presentation of the bottlenecks

Three case studies were conducted to detect enterprise specific bottlenecks that hinder enterprise integration.

2.5.1 Bottlenecks found in the farm enterprise of case study 2

This section presents how we found bottlenecks in case study 2. A detailed description of these selected and case specific bottlenecks are provided because comparable bottlenecks are found in the other case studies. Not all bottlenecks can be described here in detail, but these can be found on the website on arable farm enterprise integration. By presenting a detailed description how we found bottlenecks the application of RAAgE and the method is demonstrated.

The baseline (as-is state) architecture of the arable farm

At the arable farm of case study 2 different applications are supporting the process group *product management*. The two applications are installed on a PC and are named 'CROP basis',[27] with additional application modules, and 'FarmWorks Mapping'.[28] These applications are used by the operating control process *Plan cultivation work-order* and *Check cultivation operation(s)* of the *Cultivate a seed potato variety* process and the management control process *Evaluate product management plan*, see Figure 13. Both applications read and write data to separate databases. These application components are used to support other process groups and farm business processes as well but these relations are not modeled in this enterprise architectural description.

[27] 'Crop basis' is a product of Agrovision. (http://www.agrovision.nl/uploads/media/CROP-Basis-leaflet_02.pdf)

[28] 'FarmWorks Mapping' is a product of FarmWorks software. (http://www.farmworks.com/products/mapping)

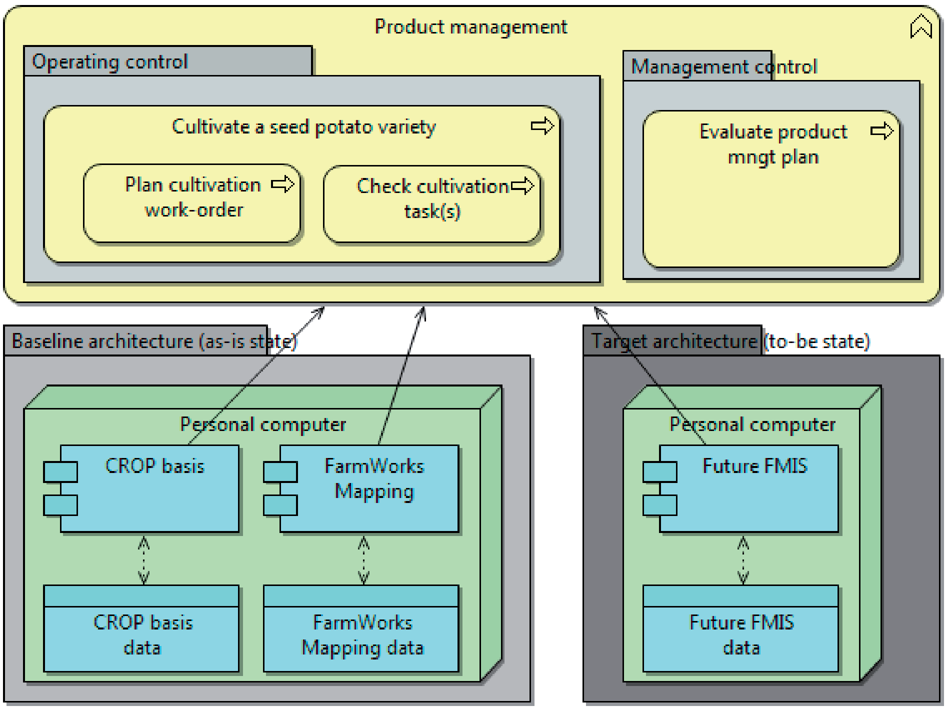


Figure 13: Application components in the as-is and to-be state that are used by processes of the Product management process group.

At the farm, the 'CROP basis' application component is associated with the additional application components 'LAP', 'GeoCrop' and 'Gewis' because these applications are able to share data based on a shared data repository. These components offer various functions such as crop registration and advice when to destruct the haulm of the potato plant (see Figure 14). The 'CROP basis' application has a focus on crop registration and to provide advice.

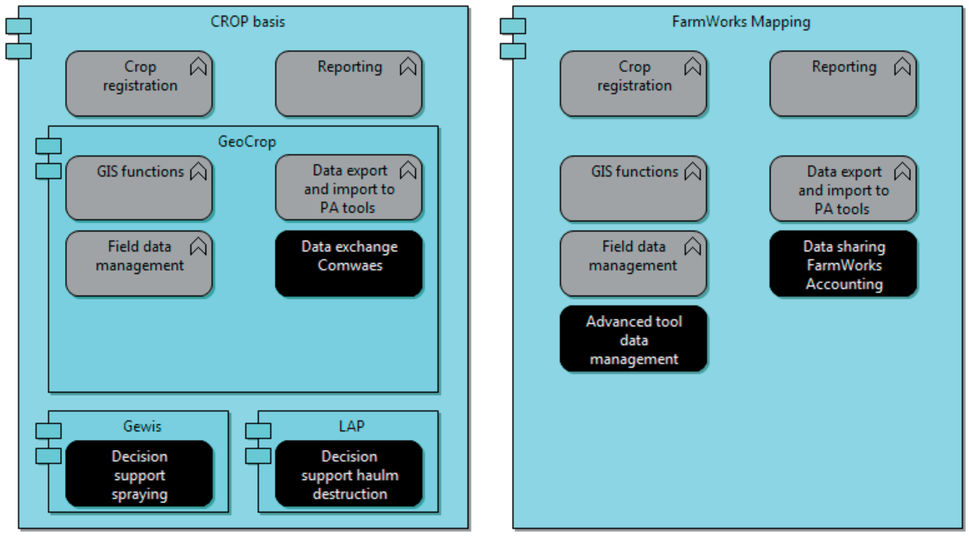

CROP basis
Crop registration
Reporting
GeoCrop
GIS functions
Data export and import to PA tools
Field data management
Data exchange Comwaes
Gewis
Decision support spraying
LAP
Decision support haulm destruction
FarmWorks Mapping
Crop registration
Reporting
GIS functions
Data export and import to PA tools
Field data management
Data sharing FarmWorks Accounting
Advanced tool data management

Figure 14: The application functions of the 'CROP basis' application component with additional application components and 'FarmWorks Mapping'. Comparable application functions are grey, different application functions are black.

In addition, various functions of the 'FarmWorks Mapping' application component are used (Figure 14). The 'FarmWorks Mapping' application focuses on displaying GIS data recorded by ICT Components on fields and to create variable rate prescription maps, to be used in the business process *Plan cultivation work-order* and *Execute cultivation work-order*. 'FarmWorks Mapping' offers a various set of interfaces to exchange data with ICT Components used for precision farming.

The target (to-be state) architecture of the arable farm

Currently, at the arable farm in case study 2 the processes in the process group product management are supported by 'Crop basis', with additional modules and 'FarmWorks Mapping'. Therefore, the farm enterprise has to pay for two application components that support the process group product management. Furthermore, the farmer needs to log data into both applications and to manage this data separately. This separation is not optimal, and the farm manager would prefer a single application supporting processes related to the cultivation of potatoes. The manager would prefer an application retaining the same functions and services currently offered by both, e.g. same GIS functions, services and interfaces as 'FarmWorks Mapping' and the 'Gewis' and 'LAP' application components of 'CROP basis'. This target (to-be) architecture is depicted in Figure 13.

Gap analysis
Based on the description of the *as-is* and *to-be* state we can conclude that two applications offered by two vendors are supporting the same processes. A switch to only one of these applications sacrifices specific services and interfaces that are used by business processes (see Figure 15). For example, if the additional module GeoCrop is no longer used, the farmer will lose historic information. Therefore, both applications are required to support the processes at the arable farm.

The application components currently have different focusses. The 'CROP basis' application module offers a various set of services related to the registration of cultivation operations. This data can easily be shared with processors of the harvested crops. Some of these services are not available in 'FarmWorks Mapping'. Meanwhile 'FarmWorks Mapping' application also offers unique services and interfaces. Consequently, both application components offer partly overlapping and partly unique services and interfaces.

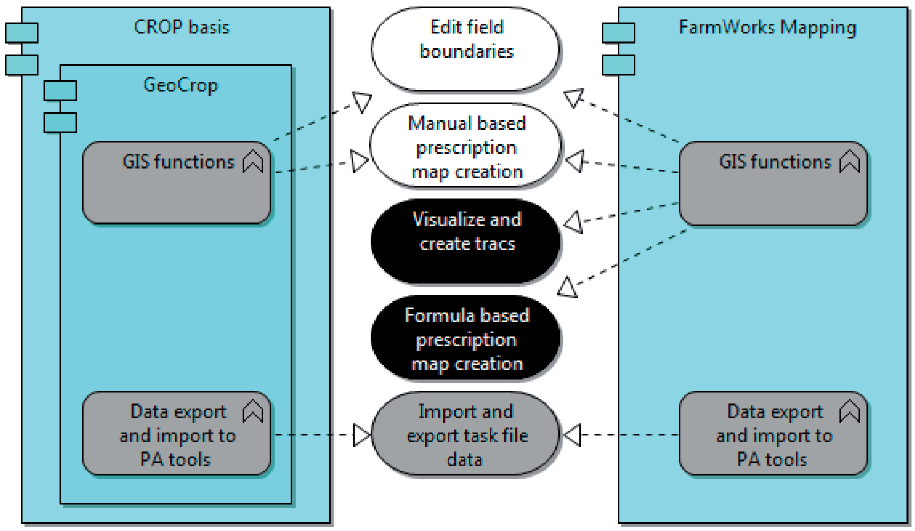

CROP basis
GeoCrop
GIS functions
Data export and import to PA tools
Edit field boundaries
Manual based prescription map creation
Visualize and create tracs
Formula based prescription map creation
Import and export task file data
FarmWorks Mapping
GIS functions
Data export and import to PA tools

Figure 15: The 'CROP basis' and 'FarmWorks Mapping' application components with similar functions realizing overlapping (white), unique (black) and comparable (grey) services. Not all services offered by the applications are depicted.

Looking into the application interfaces used by the 'Crop basis' and 'FarmWorks Mapping' application we can see that some are unique and others overlapping (see Figure 16).

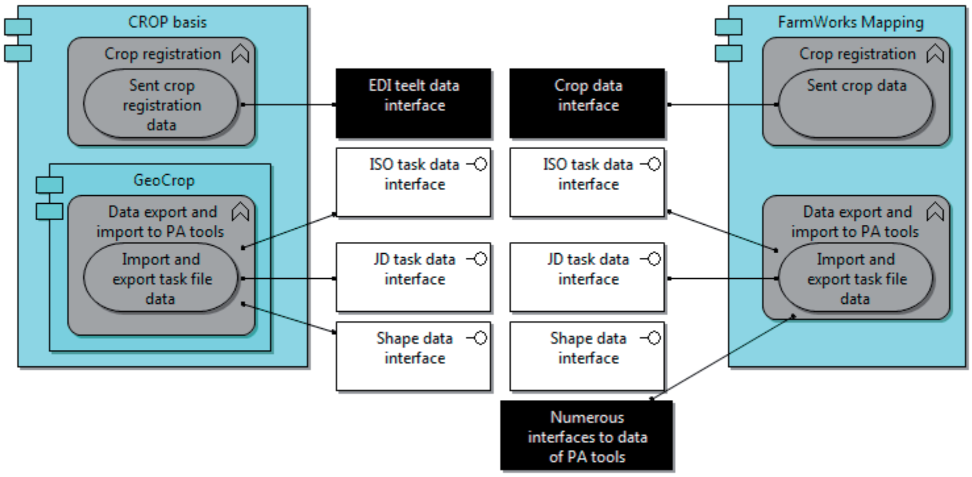


Figure 16: Application interfaces assigned to application services of 'Crop basis', 'Geo-crop' and 'FarmWorks Mapping'. Overlapping interfaces are white. Unique interfaces are black. Not all interfaces of the applications are depicted.

Since the application components currently have separate data repositories, the management of data in one repository accessed by multiple application components is not currently possible.

Roadmap components

We investigated whether other application components (roadmap components) are available that offer all required functionalities, services and interfaces or are able to have a shared data repository. This meant reviewing the application components listed in Table 2, a review, partly based on Robbemond and Kruize (2011), but also covering additional applications.

Table 2: Reviewed applications.

Vendor	Application component(s)
AgJunction	'AgJunction'
AgroCom/Claas	'AGROCOM Net'
LAND-DATA EuroSoft	'AO Pflanze'
Progis	'Docuplant LT', 'Docuplant professional', 'Docuplant soil manager'
SST Software	'SST Summit professional'
Fairport	'PAM QA Plus'
AgLeader	'SMS Software Advanced'
FarmPlan	'Gatekeeper'

We concluded that a current problem in arable farm enterprise integration is that no vendor offers a single application with all the functions, services and interfaces required by the arable farm manager in case study 2. Moreover, none of these application components of different vendors exchange data based on a shared data repository.

Bottlenecks

Based on the presented analysis we have found the case specific bottlenecks (CSB) 2 and 3, presented in Table 3. Based on another analysis CSB1 has been identified. This analysis can be found in the document describing case study 2 on the website on arable farm enterprise integration.

Table 3: Bottlenecks found in case study 2.

Case specific bottleneck	Bottleneck description
CSB1	An essential application component (FMIS) that can coordinate the cultivate seed potato processes and the process potato process is not available. This application component should support the tracking and tracing of multiple potato varieties over multiple years over multiple fields and in multiple batches. Its absence means current business processes are inadequately supported or coordinated.
CSB2	Multiple application components (FMISs) are required to support *plan cultivation work-order* and *check cultivation tasks(s)* processes. The available FMISs partly overlap but each FMIS has also unique functions, services and interfaces. This means multiple application components need to be bought and maintained, increasing the cost to the farmer. It furthermore produces usability problems, because the user interfaces of both application components must be understood.

Table 3: Continued.

Case specific bottleneck	Bottleneck description
CSB3	The variety of application components (FMISs) required to support the *plan cultivation work-order* and *check cultivation task(s)* processes are unable to exchange all data and have no shared data repository. This requires retyping/re-entering of data, indicating island automation and a vendor lock-in.

2.5.2 Other bottlenecks

This section summarizes the specific bottlenecks found in the other two case studies. The bottlenecks found in case study 1 are presented in Table 4. The bottlenecks found in case study 3 are presented in Table 5.

Table 4: Bottlenecks found in case study 1.

Case specific bottleneck	Bottleneck description
CSB1	An essential application component on a personal computer, which offers services to create, read, use or delete guidance reference lines of the 'JD 2630 terminal' , is not available. Application components exist that offer these services, but their interfaces are not compliant with the interface of the 'JD 2630 terminal'. Therefore the guidance reference lines of the JD 2630 terminal cannot be managed on a personal computer.
CSB2	Multiple application components (FMISs) will be required to support variable rate potato planting. This bottleneck will occur because available application components have partly overlapping and partly unique functions, services and interfaces. This creates higher costs, because multiple application components need to be bought and maintained. It also can create usability problems because the user interfaces of both application components must be understood.
CSB3	No business services are offered that support configuring a set of ICT Components (e.g. FMISs, terminals and implements) to enable precision planting of potatoes. In this case the configuration of a 'JD 2630 terminal' with a (to be bought) potato planting implement was not supported by an independent enterprise. This creates difficulties for farmers when they must choose and configure ICT Components that can be integrated with the ICT Components that are already used in the enterprise.

Table 5: Bottlenecks found in case study 3.

Case specific bottleneck	Description
CSB1	No application component and node (terminal/tool) exists to log data during the irrigate field task (e.g. fuel use data, amount of irrigated water). This data therefore cannot be logged.
CSB2	This bottleneck is identical to CSB2 of case 2, presented in Table 3.
CSB3	This bottleneck is identical to CSB3 of case 2, presented in Table 3.
CSB4	Available application components realized by a single node (terminals) and used to control implements offer similar functionality and services but have different interfaces. This means a variety of terminals are required to log data and control implements during execute cultivation operation processes.
CSB5	Data structures are missing for data objects that realize crop growth information. This requires that farmers be skilled at manually exchanging data (e.g. sensors data manually) between the application component on the tractor (terminal) and the application component on the personal computer (FMIS) and/or between the FMIS and application components of other organizations (e.g. through the exchange of satellite data).

Thus bottlenecks that describe the gap between the baseline (*as-is*) and target (*to-be*) enterprise architecture are present in all three case studies. These bottlenecks prevent each farm manager from achieving a self-defined *to-be* target enterprise architecture. As we have now determined, they are not addressed by current state-of-the-art ICT Components (application components realized by a node) or business services.

2.5.3 Validation of the bottlenecks

National and international experts with expertise in arable farming validated the bottlenecks found in the case studies. The main goal of this validation was to determine, if these bottlenecks were recognized and perceived as relevant for other crops and in countries other than the Netherlands. Such recognition of these bottlenecks would indicate the relevance of our results for a much larger population of farmers, who are currently not being served by available ICT Components.

Because validation requires less specific descriptions of the bottlenecks, five more generic bottlenecks were described (see Table 6) based on the case-specific bottlenecks (see Table 7). These generic bottlenecks and descriptions were validated by the experts in arable farming using semi-structured interviews. These documents can be found on the website on arable farm enterprise integration. Each expert answered the questions, presented in Table 8, for each generic bottleneck. The result was that all experts understood the generic bottlenecks and recognize that they occur in other countries and for other cultivations.

Table 6: Description of more generic bottlenecks.

Bottleneck	Description
Bottleneck 1	Services or interfaces are not realized by ICT Components that are required to support farm processes
Bottleneck 2	Application components and nodes (FMIS's and Terminals) have partly overlapping and partly unique services and interfaces
Bottleneck 3	The variety of application components (FMIS's) that are required to plan work-order and check cultivation task(s) process are not able to exchange all data or have a shared data repository.
Bottleneck 4	Data structures are missing which are required for (unskilled) data exchange.
Bottleneck 5	No business services are offered that support arable farm managers in configuring a set of ICT Components (e.g. FMIS's, terminals and implements)

Table 7: The relation between the generic bottleneck and the case specific bottlenecks (CSB).

Generic Bottlenecks	Case 1	Case 2	Case 3
Bottleneck 1	CSB1	CSB1	CSB1
Bottleneck 2	CSB2	CSB2	CSB2, CSB4
Bottleneck 3		CSB3	CSB3
Bottleneck 4			CSB5
Bottleneck 5	CSB3		

Table 8: Question to validate the bottlenecks.

	Question
Q1	Do you understand the bottleneck?
Q2	Do you recognize the bottleneck for potato production in your country/countries?
Q3	Do you think this bottleneck is applicable for other crops in your country/countries and if so for which?

Based on this validation we conclude that the bottlenecks we found are applicable more broadly than only for cultivation of potatoes in the

Netherlands. Experts suggested additional case specific bottlenecks which could be grouped into the generic bottlenecks. These generic bottlenecks, we have concluded, are present-day issues not resolved with current state-of-the-art ICT Components or business services available to arable farmers. During the validation interviews, certain vendors were mentioned as working on possible solutions already. These upcoming solutions should be the subject of future research. Solving the bottlenecks found, will enhance arable farm enterprise integration, increase arable farmer efficiency and could improve responses to consumer demand.

2.6 Discussion and conclusions

Based on existing models we developed RAAgE, a Reference Architecture of Agricultural Enterprises. RAAgE has been improved by using it for the development of architectural descriptions of three arable farm enterprises in the Netherlands. The architectural descriptions of these three case studies, focusing on potato cultivation, showed bottlenecks that hinder arable farm enterprise integration. These bottlenecks were generalized into five key bottlenecks which have been validated with national and international experts. Additionally, RAAgE has been improved based on a validation by two experts in arable farm modeling.

RAAgE 1.0 can be used as a reference to model arable farm enterprise architectural descriptions in a coherent and uniform manner. With RAAgE, a reusable model is created to describe aspect of the business layer, application layer and technology layer of an arable farm enterprise. In its current state it does not support the creation of descriptions of all entities within an arable farm, nor all business processes. However, RAAgE provides a structure and content that can be used to design different views on the architectural description of arable farm enterprises. It is a good start for describing the arable farm in a uniform manner in relation to the configuration and integration of various ICT Components at the single enterprise level.

To the best of our knowledge, no reference model was available that enables creation of arable farm enterprise architectural descriptions.

Available reference models such as IMOT and SCOR version 8.0[29] were, from our perspective, insufficient, but relevant aspects of these models were incorporated in RAAgE. For example, SCOR can model operational processes of supply chains, although not at a sufficient level of detail for arable farm enterprises. Additionally, because SCOR does not focus on designing architectural descriptions of enterprises, it cannot model the application layer and the technology layer. Similarly, IMOT provides useful insights into business aspects of arable farms. The experts in arable farm modeling, who validated RAAgE, suggested more alignment between IMOT and RAAgE, e.g. information flows, process descriptions. We will adopt this suggestion in the future. Compared to RAAgE, IMOT currently provides more insight into the business layer. At the same time, the IMOT structure of the business layer is insufficiently flexible, while RAAgE is more flexible and allows extensions. Moreover, RAAgE enables modeling of the business layer, but also of the application and technology layers.

In current literature relevant aspects regarding arable farm enterprise integration have been presented. For example, literature provided detailed insight in data flows and required datasets for precision agriculture (Nash et al., 2009), information flows within a field operation (Sørensen et al., 2010b) and the farm decision making process (Fountas et al., 2006). These details are not available in RAAgE, although can be incorporated when required. Current literature and models provide detailed insight. However, the provided models do not have a whole farm focus regarding arable farm enterprise integration. Therefore, our research is an addition to literature because available models did not describes enterprise architectures of farm enterprises neither bottlenecks that hinder data and information flows of these farm enterprises.

Thus, in creating RAAgE we have built on top of available reference models and other prior research. Using ArchiMate, RAAgE has more representational power than current models in literature related to arable farm enterprise integration. We enriched the representational power of ArchiMate by adding (specializations of) a farm process group including farm business (sub-)processes, facilitating the design of

[29] Supply Chain Operation Reference (SCOR) model. (http://supply-chain.org/scor)

architectural descriptions of arable farm enterprises. This makes RAAgE an innovative addition to currently available reference models.

In addition, our research contributes by providing insights in the architectural description and revealing our key focus: bottlenecks hindering farm integration. The bottlenecks we detected are more related to intra-enterprise integration than to inter-enterprise integration, because our focus was on detecting bottlenecks related to cultivation of crops. They also do not arise from the integration aspects of ICT Components supporting processes in process groups other than product management, since these were not investigated. Together with the small number of case studies, this means that other bottlenecks may also be present. These could be found by including more cases or by future modeling of other process groups.

However, a single focus was necessary for brevity, and considering only the ICT Components used in the process group product management allowed us to examine a rich source of data. Firstly, the processes in the process group product management and the ICT Components used to support these processes are unique in comparison to other business processes, and thus tested our modeling capabilities. Secondly, the processes and ICT Components we examined are essential to enable precision agriculture. This justifies the small number of case studies and limited focus on the processes and ICT Components used, since the bottlenecks found and validated are meant to be illustrative.

The identified bottlenecks are not all new or unknown, but we have been able to describe them in a structured manner, which helps to gain insight into the problem of arable farm enterprise integration. The structure can also be beneficial for ICT Component developers and vendors to identify the current problems farm managers face and improve their ICT Components accordingly. Structured and coherent modeling of bottlenecks, not in text but also visually using RAAgE, should facilitate communication between different stakeholders with different roles. We believe that this approach, i.e. describing arable farm enterprise architectures, is a valuable move towards gaining insight into arable farming and ICT Components used. It helps to improve arable farm integration and could contribute to adoption of precision agriculture. Furthermore, RAAgE can improve communication about arable farm enterprise integration between stakeholders, e.g. arable farmers, software developers, vendors and researchers.

Future work is needed in three areas. Firstly, RAAgE needs to be extended to enable the description of arable farm enterprise architectures in more detail. Secondly, delineating the applicability of the modeled bottlenecks requires additional case studies. Thirdly, we expect that developers of state-of-the-art ICT Components face difficulties in resolving current bottlenecks. Therefore, we suggest research should investigate these difficulties and develop an arable farm enterprise integration framework. This framework could result in the development of improved ICT Components that contribute to a more sustainable, safe and transparent food production.

3

A Reference Architecture for Farm Software Ecosystems

This chapter is based on the published journal article:

Kruize, J.W., Wolfert, J., Scholten, H., Verdouw, C.N., Kassahun, A., & Beulens, A.J.M., Computers and Electronics in Agriculture, 2016, Volume 125, Pages 12-28.

Abstract

Smart farming is a management style that includes smart monitoring, planning and control of agricultural processes. This management style requires the use of a wide variety of software and hardware systems from multiple vendors. Adoption of smart farming is hampered because of a poor interoperability and data exchange between ICT components hindering integration. Software Ecosystems is a recent emerging concept in software engineering that addresses these integration challenges. Currently, several Software Ecosystems for farming are emerging. To guide and accelerate these developments, this chapter provides a reference architecture for Farm Software Ecosystems. This reference architecture should be used to map, assess design and implement Farm Software Ecosystems. A key feature of this architecture is a particular configuration approach to connect ICT components developed by multiple vendors in a meaningful, feasible and coherent way. The reference architecture is evaluated by verification of the design with the requirements and by mapping two existing Farm Software Ecosystems using the Farm Software Ecosystem Reference Architecture. This mapping showed that the reference architecture provides insight into Farm Software Ecosystems as it can describe similarities and differences. A main conclusion is that the two existing Farm Software Ecosystems can improve configuration of different ICT components. Future research is needed to enhance configuration in Farm Software Ecosystems.

Keywords: Farm Management Information Systems, Software Ecosystems, Open Software Enterprise, Interoperability, Precision Agriculture, Smart Farming

3.1 Introduction

Agri-food supply chain networks are confronted with a growing world population and increasing prosperity and associated changing demands. These developments are challenging because the demand on food is increasing while there are stricter requirements regarding food safety, sustainable food production and transparent supply chains. Therefore, farm enterprises[30] are pushed to improve their production processes by smart monitoring and control. Smart monitoring, -planning and -control of production processes, which can be referred to as smart farming, can be supported by a broad spectrum of technologies, ICT components, and their constituent hard- and software systems (Aubert et al., 2012; Cox, 2002; Lamb et al., 2008; Wolfert et al., 2010). Examples of these ICT Components are all kinds of sensors, terminals, implement assemblies, computers and software applications. For smart monitoring and control an integrated information system is required that enables seamless interaction and sharing of data between different ICT Components. However, a lack of interoperability is currently severely hindering smart farming because ICT Components of multiple vendors do not operate as one integrated farm information system (Aubert et al., 2012; Fountas et al., 2005; Pedersen et al., 2004; Pierce and Nowak, 1999).

To overcome this, Wolfert et al. (2014) identified five main challenges (i) handling the increasingly large amounts of data, especially from all kind of agricultural equipment, (ii) interoperability between various systems at farm level and in the whole supply chain network surrounding the farm, (iii) standardization of data, (iv) go beyond the small scale and the regional focus of farm software development while at the same time (v) comply with national or regional differences in farming practices. More specifically for interoperability, the systematic analysis of Kruize et al.(2013) showed that ICT Components used within the same farm enterprise (i) have partly overlapping and partly unique services, functions and interfaces, (ii) are missing required application services, functions and interfaces, (iii) have separated data repositories

[30] A farm enterprise can be an arable farm, livestock farm or horticultural farm. In this chapter we focus on arable farm enterprises however it is expected that the concept Farm Software Ecosystem can address software integration challenges for the other type of farms as well.

and (iv) have inadequate and incomplete data exchange. In conclusion, most of the available ICT Components are lacking both technical and semantic interoperability, resulting in data sharing issues and non-coherent user interfaces (Kruize et al., 2013). Consequently, current ICT Components often hamper farm enterprise integration as they do not sufficiently support the monitoring, planning and control processes to enable smart farming. Supporting these processes by making a combination of multiple ICT Components is currently challenging. In addition, the creation of one overarching system developed by one software vendor that overcomes all mentioned challenges is neither a feasible nor – from a competitive point of view – a desirable solution. Hence, a promising method to achieve such integrated solutions is a best of breed approach, which allows users to configure customized software systems from standardized components that are supplied by multiple vendors (Light et al., 2001; Verdouw et al., 2010a). As a consequence, software systems are not supplied by single companies, but by a set of independent actors which collaborate and can compete via an integration platform (Light et al., 2001). This integration approach requires an advanced infrastructure that covers both organisational and technological aspects (Wolfert et al., 2010). An organisational infrastructure is required that enables and facilitates both collaboration and competition between actors. In such infrastructure, actors collaborate in their development to provide interoperable ICT Components that are based on their core competences and compete with ICT Components that provide similar functionalities. A technological infrastructure is required that can support the linkage of ICT Components into integrated FMISs. Both the organisational and technological infrastructure should enable and ensure a sustainable collaboration and competition in which all actors, including software developers, farm enterprises, contractors, technology providers and others, can flourish.

A concept that addresses such an infrastructure is nowadays called a Software Ecosystem. Currently, Software Ecosystems are becoming more widespread as they are increasingly considered to provide an effective way to construct large software systems on top of a software platform by combining components, developed by actors that are part of different organisations (Bosch, 2009; Manikas and Hansen, 2013; te Molder et al., 2011). Examples of current Software Ecosystems are, amongst others, Eclipse, Linux/Linux kernel and Android (Manikas and Hansen, 2013). At the moment there are no well-established Software Ecosystems for farming available, although several developments go into this direction. Large agricultural machinery vendors have setup their own proprietary platforms (e.g. John Deere's Farmsight[31] or AGCO's Fuse Technology[32]). With these platforms it is still difficult to establish interoperability with other components that come from other manufacturers. Several multi-vendor platforms (e.g. 365FarmNet,[33] Crop-R, AgroSense, FIspace) are recently introduced, but these are still in an early stage of development and sometimes regionally oriented lacking a large international user base.

To gain deeper insights into these developments and to support further development of Farm Software Ecosystems, this chapter proposes a reference architecture that can be used to map, assess, design and implement Farm Software Ecosystems that contribute to integrated FMISs. The purpose of the reference architecture is to improve communication and collaboration between multiple actors that are part of real-world Farm Software Ecosystems. It will help them to understand Software Ecosystems and enable them to join, form or improve Farm Software Ecosystems that lead to integrated farm information systems.

The remainder of this chapter first introduces literature about Software Ecosystems and the relation to software development for farming. Second, the methodology for designing the reference architecture for Farm Software Ecosystems is described. Next, the requirements for the reference architecture, the reference architecture itself and an example farm information system that can result from a Farm Software Ecosystem is described. This is followed by an evaluation to verify the Reference Architecture based on the requirements and to validate if it can map existing Farm Software Ecosystems to provide insight how it matches and in what extend. This chapter concludes with a discussion and outlook for future research and development.

[31] www.myjohndeere.deere.com
[32] www.agcotechnologies.com
[33] www.365farmnet.com

3.2 Software Ecosystems and software development for farming

In the Internet of Services (IoS) software components are available as interoperable services on the internet. The IoS allows to decouple the possession and ownership of software from its usage and thus to use Software as a Service (Turner et al., 2003). Users do not need to buy and install a large software system, but required functionality is delivered as a set of distributed web services that can be configured and executed when needed. In contrast to traditional non-modular software systems, it is no longer necessary that components are delivered by the same software vendor. Software companies can concentrate on the development of components that fit best to their core competences. Users can configure customized software systems from standardized components that are supplied by multiple vendors that interact via a common technological platform. Such collaborative environments are nowadays referred to as Software Ecosystems. Software Ecosystems are defined as the interaction of a set of actors on top of a common technological platform that results in a coherent set of ICT Components or services (Manikas and Hansen, 2013). These components include hardware, software and service modules, along with an architecture that specifies how they fit together (Eisenmann et al., 2008).

In practice, a Software Ecosystem is usually started by a single- or a group of software producing organisations that open up their business processes to become an open software enterprise (Jansen et al., 2012). Such an open software enterprise provides a technical platform and additional (collaboration) artefacts that are essential for the coherence of the software components and for collaboration between multiple actors (Seichter et al., 2010). There are various reasons why actors with different perspectives would like to collaborate in such an environment (Bosch, 2009; Wolfert et al., 2010):

(i) It increases the value of the core offering to existing users and increases the attractiveness for new users.

(ii) Increase “stickiness” of the technology platform, i.e. it is harder to change the platform when it is widely used (cf. PC operating systems e.g. Windows, iOS, etc.).

(iii) It creates and facilitates a structural and independent environment, developed by partners in the ecosystem that potentially offers a large critical mass of users (once success has been proven).

(iv) Share the costs of innovation by collaborating with other actors and accelerate innovation through open innovation in the ecosystem.

(v) Decrease total costs of ownership and risks for commoditizing functionality by sharing the maintenance with networking partners.

The concept of Software Ecosystems is new for the agricultural domain. Related literature focuses on the integrating capabilities of farm ICT Components by proposing a standardized infrastructure that supports the integration of ICT Components of multiple vendors (Iftikhar and Pedersen, 2011; Kaloxylos et al., 2012; Nash et al., 2009; Steinberger et al., 2009; Wolfert et al., 2010). Most of these papers focus on semantic aspects of the data to improve interoperability of application components (Iftikhar and Pedersen, 2011; Nash et al., 2009; Steinberger et al., 2009). Examples of available standards for the agricultural domain that facilitates data exchange between application components are the international ISO-11783 standard,[34] the Dutch EDI-Teelt standard,[35] and the German AgroXML standard.[36] Most papers also focus on application integration in which there is a focus on the design of an integrated FMIS (Kaloxylos et al., 2012; Kaloxylos et al., 2014; Nikkilä et al., 2010; Sørensen et al., 2010a; Sørensen et al., 2010b). A recent implementation of a platform that can be used to develop an integrated Farm Management Information System (FMIS) is described in Kaloxylos et al. (2012) and Kaloxylos et al. (2014). Yet, this literature misses a specification on how to operationalize and organize a Farm Software Ecosystem. The reference architecture in this chapter will address this shortcoming.

[34] http://dictionary.isobus.net/isobus/
[35] www.agroconnect.nl
[36] www.agroxml.de

Furthermore, the Software Ecosystem literature is in an early stage since the concept is coined relatively recently (Messerschmitt and Szyperski, 2003). There is still little consensus on what precisely constitutes a Software Ecosystem, a few analytical models of Software Ecosystems exist, and little research is done in the context of real-world Software Ecosystems (Manikas and Hansen, 2013). Hence, this paper also aims to contribute to the theoretical basis of Software Ecosystems in general.

3.3 Materials and methods

3.3.1 The future internet program and existing Farm Software Ecosystems

The research presented in this chapter was carried out as a part of SmartAgriFood and FIspace project which are part of the European Future Internet Public-Private Partnership programme (FI-PPP)[37]. In SmartAgriFood, the needs from the agri-food sector for Future Internet ICTs were identified while at the same time the capabilities of Future Internet were described by potential use case scenarios in agri-food (Kaloxylos et al., 2012). This has resulted in a conceptual platform architecture and several prototype applications. The FIspace project is currently implementing these concepts into a software platform for business collaboration for the agri-food, transport and logistics domain. This platform and its architecture will enable collaboration of application components and can be a basis to form several Software Ecosystems. More detailed background information about these projects can be found in Kruize et al. (2014), Verdouw et al. (2014) and Wolfert et al. (2014).

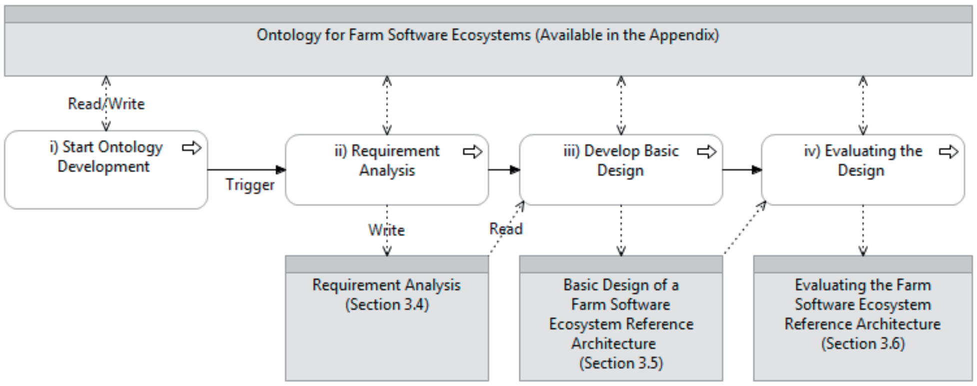
As a conceptual validation to test the mapping functionality of the reference model two Dutch initiatives in which Farm Software Ecosystems are being established were selected: Crop-R and AgroSense. Crop-R is a Dutch organisation developing an online platform offering GIS-based crop-recording applications on the web, smartphones and tablets. The system has currently more than one thousand users such as

[37] www.fi-ppp.eu

farm enterprises and contractors. AgroSense is an open source platform on which a modular and open source FMIS can be configured. The modules can come from different independent organizations.

3.3.2 Methodology

The reference architecture for Farm Software Ecosystems was developed by a design-oriented research approach (March and Smith, 1995). Aligned with the guidelines and framework of Hevner et al. (2004), the reference architecture was designed in four steps: (i) ontology definition, (ii) requirements analysis, (iii) development of a basic design and (iv) evaluation of this design within case studies (see Figure 17).



Ontology for Farm Software Ecosystems (Available in the Appendix)
Read/Write
i) Start Ontology Development
Trigger
ii) Requirement Analysis
iii) Develop Basic Design
iv) Evaluating the Design
Write
Read
Requirement Analysis (Section 3.4)
Basic Design of a Farm Software Ecosystem Reference Architecture (Section 3.5)
Evaluating the Farm Software Ecosystem Reference Architecture (Section 3.6)

Figure 17: Research Approach.

Ontologies are important tools to enable seamless communication and mutual understanding about a domain and can reduce misunderstanding between people and enable interoperability between software components, e.g. apps, data, etc. (Scholten et al., 2007). In this research an initial basis for an ontology has been laid down and is used to describe the requirements analysis, basic design and the application of the design. The ontology has been iteratively developed within this research of which the basis was the architectural language ArchiMate (TheOpenGroup, 2011). The ontology is available in the Appendix A.

The requirements analysis started with the identification and definition of the scope the object system for Farm Software Ecosystems based on Sørensen et al. (2010b) and Wolfert et al. (2010). Next, the requirements for Farm Software Ecosystems were derived from the research that was carried out in the SmartAgriFood and FIspace projects. In SmartAgriFood a survey was carried out to assess the user's expectations on future internet Functions and Services. In total 135 questionnaires in 6 countries and 8 focus group discussions with 69 participants in 5 countries have been performed and analysed, collecting feedback on the interpretation of the future internet capabilities from the user's perspective, the current use of internet applications in their daily work and today's problems or limitations resulting in expectations and requirements for the future internet. These were compared to the Future Internet's capabilities that were identified in the FIWARE project.[38] FIWARE provides enhanced OpenStack-based cloud hosting capabilities and a rich library of components – so-called Generic Enablers (GEs) – implementing a number of added-value functions offered 'as-a-service'. The Generic Enablers concern amongst others Context Management, easy connection to the Internet of Things, Open Data support and Big Data processing and analysis. A technical team consisting of a number of software architects from leading ICT companies and technical universities - including the authors - developed the conceptual architecture into the FIspace platform. A detailed documentation of this platform can be found online.[39] Beside the empirical results from these projects, a literature analysis was done on the development of Software Ecosystems in relation to smart farming. Based on the ontology and the requirements analysis, the reference architecture for Farm Software Ecosystems was designed.

The evaluation of the Reference Architecture contains two parts. First the Reference Architecture was verified based on the requirements. Second, as a conceptual validation the mapping functionality of the Reference Architecture was tested using two existing Farm Software Ecosystems. These mappings are based on semi-structured interview with the CTO's of AgroSense and Crop-R.

[38] www.fiware.org
[39] https://bitbucket.org/fi-space/doc/wiki/Home

3.4 Requirements analysis

3.4.1 Object system

To define the scope for Farm Software Ecosystems, the object system of this study should be defined. For that purpose Farm Enterprises are considered as the Business Systems[40] with basic inputs and outputs for which agricultural software is developed that virtualize the objects (Figure 18). Virtualization allows to decouple physical flows from information aspects of operations (Clarke, 1998; Verdouw et al., 2013). The core of the object system is formed by a number of Farm Enterprises that consist of production units (e.g. fields and its crops), resources (e.g. humans, Devices, buildings, etc.) and its management. To manage the production units all kind of resources are used for the production that take place in these fields ranging from big tractors, combine-harvesters to small sensors. Farm management aims to control crop production using resources that should operate as an integrated system. Raw materials (seeds, fertilizers, pesticides, etc.) are primary inputs for the production processes. Support services can be all kind of advice that helps the farmer in managing the whole Farm Enterprise (e.g. weather information, crop status). At the output side products come from the fields that can be temporarily stored in buildings. Farm management results into administration about the products produced and the usage of resources needed in supply chains (e.g. certificates), but also more general administration at farm level that is needed for public administration and other purposes.

[40] In this chapter a word that starts with a capital can be found in Appendix A – Ontology.

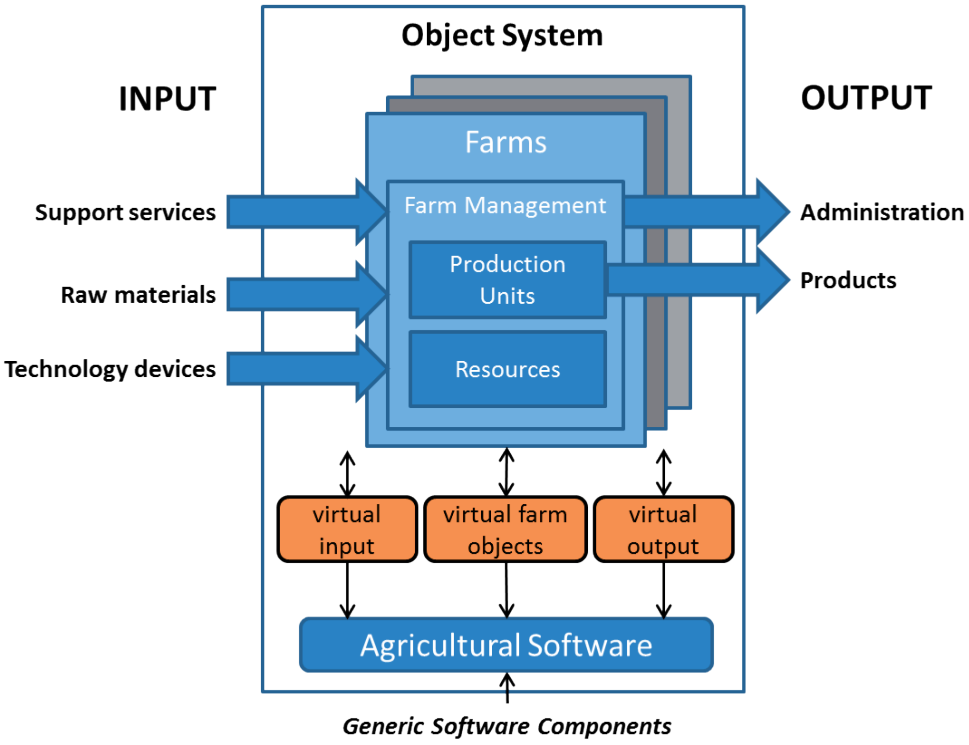


Figure 18: The object system that is identified for Farm Software Ecosystems. Input, Output and generic software components are considered to come from outside the system.

It can be observed that virtualization plays an important role to represent input, output production units and resources. Providers of inputs such as raw materials, technology devices and support services are trying to add value to their products by selling software and information in combination with their products. Processors of products and retailers try to get more control on the outputs from Farm Enterprises (e.g. certification information) by introducing their software at the farm. Similarly, public administrations are introducing software for farmers to streamline the information they need (e.g. for subsidies). At this moment the software for farm management is usually produced by companies to whom the farming domain is their core business. The software that is involved at the input- and output side is usually provided by third parties ranging from big established software companies to small, innovative start-ups. All this software is specifically focussing on the Farm Enterprise and thus included in the farm object system. Software producing companies are generally using generic software components that according to Figure 18 are considered to come from outside the farm object system(e.g. FIWARE Generic Enablers, Operating Systems etc.). Companies that are involved at the input/output side (e.g. chemical companies, processors) also have software to support their own production processes. These software applications can have Application Interfaces to enable data exchange with software applications used at the farm.

3.4.2 Functional requirements for a Farm Software Ecosystem

A Software Ecosystem was previously defined as the interaction between Actors on top of a common technological Platform resulting in a coherent set of ICT Products or Services. The previous section described the object system that a Farm Software Ecosystems has to focus on. The main question is how to operationalize and organize a Farm Software Ecosystem around a common technological Platform in order to meet and overcome the challenges intrinsic to the farm object system and mentioned in the introduction. To answer this question functional requirements are derived within the SmartAgriFood project, the FIspace project and an additional literature analysis. In SmartAgriFood the users' needs and expectations were derived and mapped onto the FIWARE GEs. This resulted in a number of use case scenarios in particular for smart farming that are previously published by Kaloxylos et al. (2012). Based on these results, literature both on Software Ecosystems and smart- and precision farming and the development of the FIspace platform the following main functional requirement categories for Farm Software Ecosystem can be identified, which are:

(i) smooth data handling and seamless data exchange between ICT Components (Kaloxylos et al., 2012; Kruize et al., 2013);
(ii) a configuration approach to link ICT Components to each other in a meaningful and coherent way (Kaloxylos et al., 2012; Verdouw et al., 2014);
(iii) interoperability of different ICT Components (Kaloxylos et al., 2012; Kruize et al., 2013);
(iv) an open software enterprise that smoothly facilitates the previous points (Jansen et al., 2012).

The next subsections will describe these functional requirement categories in more detail.

3.4.2.1 Data handling and seamless data exchange supported by Standards

In general, but also in agriculture, the amount of available data is exploding due to the introduction of all kind of sensing and monitoring Devices. One aspect of this increasing amount of data in agriculture is its storage and the transport capacity of the network. Especially this transportation of data in agriculture is challenging due to lack of connectivity and bandwidth in the remote areas and the mobile ICT Components in farm environments. The storage of farm data is mostly done in data repositories located at multiple sites (e.g. at the farm or in different cloud repositories). Locating all these data to one central place to add intelligence is difficult because this data is often acquired by different ICT Components (e.g. sensors, monitoring Devices or Application Components). Still, farmers need integrated solutions in which access to data located in distributed repositories with multiple Application Components is required. This requires a Farm Software Ecosystem to support the development of Application Components that are able to exchange messages that contain data between distributed data repositories. For seamless data exchange data lifecycle considerations have to be taken into account (e.g. data collection, processing, sharing).

To exchange data between different applications the Application Programming Interfaces (API's) must enable sending and receiving messages. These messages must be based on technologies and semantics that are known by the other API's. Technology standards including the syntax are currently well standardized (e.g. web-service technology using XML or Json as a syntax). Furthermore, semantic standards are available although in agriculture semantic standards or its implementation are often missing, making data exchange between ICT Components cumbersome (Kruize et al., 2013). The result is that farmers are not able to exchange data between different ICT Components (e.g. data exchange between FMIS and sensors) and are affected by a vendor lock-in hindering farmers changing ICT Components (e.g. to change their FMIS) (Kruize et al., 2013). In some cases, farmers would need advanced data handling skills to make data exchange between components work, which is a not desirable situation. Hence, Application Components developed within a Farm Software Ecosystem should be able to share data in an automated manner (Kruize et al., 2013; Sørensen et al., 2010b). The sharing of data between different Application Components should be organized in a robust manner because new Application Components emerge over time, needing data from existing ones. Therefore transfer of data between Application Components should be facilitated based on a shared Implementation of a communication protocol. Such a communication protocol is based on both technical and data semantic agreements. To enable the re-use of data from a semantic perspective Application Interfaces require data that is tagged with metadata or require certain data attributes. These metadata or data attributes enable that data, saved in a data repository (e.g. database) can be reused by multiple Application Components to support multiple Business Processes. This metadata should additionally describe for what kind of purposes the data can be used to resolve the fit for use aspect of data. From a technical perspective the integration of Application Components and their distributed data repositories requires a aligned technical architecture (e.g. a Service Oriented Architecture) (Wolfert et al., 2010).

Currently, there are multiple standardisation organisations that organize data exchange between Application Components by providing technical and semantic standards. Examples are ISO Standards, ISA Standards,[41] OGC Standards[42] and farm specific Standards such as EDI-Teelt or AgroXML. The use of these standards should be stimulated by Farm Software Ecosystems to enable data exchange between Application Components of multiple vendors. When existing standards are not proficient ad-hoc standards should be implemented to enable data exchange between the Application Components. The Farm Software Ecosystem should provide documentation about these ad-hoc standards. The documentation of these ad-hoc standards can be provided to software developers within the Farm Software Ecosystem to enable data exchange. Furthermore, the documentation can be used to adapt existing standards.

[41] www.isa.org
[42] http://www.opengeospatial.org/standards

The exchange of data between different Application Components is related to the next main requirement for Farm Software Ecosystems: a configuration approach to link ICT Components to each other that fit to the needs of the supported business process.

3.4.2.2 Configuration of ICT Components

Every Farm Enterprise is unique in its specific Business Processes and in the connections with other Actors at the input and output side (see Figure 18). Moreover, farm business can be very dynamic because of changing situations (e.g. fluctuating markets, weather changes, resources, etc.). These factors are influenced by regional differences and different farming practices. At the same time, at higher abstraction levels there are many similarities between Farm Enterprises and their Business Processes. The challenge for Farm Software Ecosystems is to deliver customized ICT Components based on both generic and specific Application Components. For Farm Enterprises the configuration of an aligned and integrated system, consisting of components of multiple vendors, is required (Pierce et al., 1999). Farm Software Ecosystems should therefore provide Artefacts that support software developers to develop interoperable ICT Components that can be configured by farmers - or their service providers - into an aligned Composite Application Component that dynamically supports certain Business Processes and can be customized for their specific circumstances. Such a configured ICT Component should have a coherent look and feel regarding the User Interface to enable farmers to participate in dynamic business networks and support a variety of farm Business Processes and management goals.

A known configuration approach in software development that fits to this requirement is called ICT Mass Customisation. ICT Mass Customisation combines efficient standard software and flexible customised software allowing the configuration of standardized ICT Components into a customer-specific assembly (Verdouw et al., 2010a). To enable such a mass customisation approach the Farm Software Ecosystem must fulfil multiple functional requirements (Verdouw et al., 2014):

(i) *Software Modularity* – ICT products consist of loosely coupled modules for which policy, input-output data, and Application

Interfaces are well defined and that can be easily substituted by other modules;

(ii) *Information Integration Platform* – deployed environment that enacts the execution of modules and enables/manages the exchange of information between them;

(iii) *Component Availability* – all required components should be readily available to configure the right Products that are required by the customer;

(iv) *Configuration Support* – adequate tools that guide users interactively through the Product specification process at different abstraction levels accounting for the fact that different configuration steps can take place at different moments in time by different people;

(v) *Reference Information Models* – standardized taxonomies that represent all possible configuration options of Product Instances and the interdependencies that exist between Components or features, including rules for permitted combinations.

In the current situation some ICT Components can exchange data. However, services that support farmers in configuring ICT Components into an integrated Farm Information System are hardly available (Kruize et al., 2013). In a Farm Software Ecosystem a common platform should enable integration of ICT Components into an integrated system by dynamic orchestration of Application Services, i.e. organizing, maintaining and managing. This requires substantial knowledge about the customer that will use it. An overview of required Application Services -grouped as Functions - are for example automated advisory, task plan analyser, crop availability, etc. (Kaloxylos et al., 2012). To operationalize ICT Mass Customisation, ICT Components need to be interoperable such that the components collaborate as they are one aligned and integrated system.

3.4.2.3 Interoperability between ICT Components

Interoperability of ICT Components is understood as components that have a shared Implementation of a communication protocol (e.g. a transfer protocol, an Application Interface) to communicate properly.[43] Interoperability enables that components are able to share data and can collaborate as if they were components of one aligned and integrated system. Such an integrated system should be configured in which the actual farm Business Processes are the foundation of the configuration process and the selected ICT Components (Wolfert et al., 2010).

To create an aligned and integrated system that can cover and support Business Processes of multiple Farm Enterprises a large variety of Application Services is required. However (i) currently, not all required Application Services are available and can be realized by the ICT Components and (ii) ICT Components currently used in a single arable Farm Enterprise have partly overlapping and partly unique Application Services and Application Interfaces (Kruize et al., 2013). Therefore, Farm Software Ecosystems should enable that multiple vendors can develop interoperable Application Components offering Application Services. These Application Components can offer similar Application Services to stimulate competition within a Farm Software Ecosystem or different Application Services to offer enough functionality. The best fitting Application Components can be selected in a configuration process. To enable configuration each component should have a detailed supplementary description in what kind of configuration the Application Component can be used, what kind of input data is required and what output data it provides. Additionally, the description of the Application Component should describe the performance (e.g. idle time) to ensure that the configuration works as a coherent system. Such descriptions should show which Application Components are interoperable and for what kind of configurations they can be used. The Farm Software Ecosystem should provide guidance to enable interoperability between ICT Components and provide a format for the description of each Application Component.

An overview of general requirements regarding seamless interoperability in collaborative-competitive economic networked environments can be found in Chituc et al. (2009). These requirements focus on establishing collaborations with external Actors regarding data exchange (e.g. weather data, input data).

[43] Based on https://wiki.oasis-open.org/tab/InteropGuide

Besides these aspects the Farm Software Ecosystem should provide an organisational structure that supports the development of interoperable ICT Components.

3.4.2.4 Organization

To develop interoperable ICT Components that can be used by Farm Enterprises different Actors, with aligned incentives, performing different roles are required to collaborate. To facilitate collaboration Farm Software Ecosystems should provide an organisation (Open Software Enterprise)(Jansen et al., 2012). This Open Software Enterprise should enable the development of interoperable Application Components, the configuration process and the operation of the configured Application Components in run-time.

For this run-time environment a technological (cloud) infrastructure should be available to host the platform and which is able to connect to all Application Components. Furthermore, it should provide a revenue and cost sharing model as software developers, infrastructure providers and configuration service providers are using each other’s components and services. To facilitate this, basic support for e.g. Contracts, payments, etc. is required. Furthermore, when possible disputes arise the governance structure should provide a resolving mechanism.

For the development of interoperable Application Components in a distributed environment a basic level of governance is required and the collaborating Actors should agree on the use of both technical standards as semantic standards that are publicly available or specific for a Software Ecosystem.

Overall, it is important that the Open Software Enterprise of a Farm Software Ecosystem enables that costs are reduced so that End-Users (e.g. farmers, contractors) can buy software functionalities that make their enterprises more advanced. Especially as farms are relatively small enterprises and are not able to invest large amounts of money in software. Therefore it will be required that all kind of Actors, having aligned incentives, are able to become part of a Farm Software Ecosystem to stimulate competition within the platform. Hence the Open Software Enterprise of a Farm Software Ecosystem should be open and avoid domination by large players that could cause vendor lock-ins, which ultimately will hamper innovation (Gawer and Cusumano, 2002). According to Eisenmann et al. (2008) a Software Ecosystem is ‘open’ when (i) no restrictions are placed on participation in its development, commercialization or use and (ii) any restrictions are applied uniformly to all potential platform participants (e.g. requirements to conform to technical standards or pay licensing fees are reasonable and non-discriminatory). Still, other forms of governance of Open Software Enterprises, for example with a more dominant player, can as well result in successful Software Ecosystems.

3.4.2.5 Technical/non-functional requirements

Beside the functional requirements that are mentioned so far, there are several non-functional requirements to be addressed, such as user management, data security, routing of information or machine-to-machine communication. Although these requirements are very important in the end, they are considered to go beyond the scope of this chapter because they are not very specific for Farm Software Ecosystems.

3.4.3 Specifications for Farm Software Ecosystems

The requirements from the previous section are summarized in Table 19 which can be found in the Appendix B. Based on these requirements specifications for Farm Software Ecosystems are derived that can improve farm enterprise integration. These specifications supported designing the reference architecture for Farm Software Ecosystems and are:

- The ecosystem should provide functional **ICT Components** for farming that
  - can be based on Application Components developed by various Actors independently;
  - allow for distributed data exchange in an automated, seamless manner;
  - use existing standards or be able to exchange data with ICT Components using other standards to enable data exchange.

  - support a configuration approach of the aforementioned interoperable Application Components that can be deployed in a distributed manner
- The ecosystem should enable Actors to perform different roles to enable that different **Business Services** for farming are offered that support:
  - software configuration
  - software development
  - software hosting
- The main categories of **Actor roles** are providers of ICT Components (Software Vendors), Agricultural Service Providers, Providers of the Infrastructure and users of these Services and the ICT Components. These Actor should be able to:
  - join the ecosystem;
  - influence the ecosystem and enable innovation
  - form a critical mass of users and providers to make the ecosystem efficient and effective;
- A common and open **Platform** is needed:
  - to facilitate collaboration between various Actors using the ICT Components and its Application Services;
  - that provides consistent standards for this collaboration that in the future will not create restrictions for exploitation (backwards- and forwards compatibility) so that multiple ecosystems around the same platform are possible;
  - that supports:
    - development of ICT Components according to the platform standards;
    - development of cohesive User-Interfaces to improve the user experience;
    - configuration of ICT Components into integrated systems in an easy but consistent manner, using reference information models.
- An **Open Software Enterprise** should:
  - provide the actual (cloud) infrastructure to make the ecosystem possible;
  - orchestrate the whole collaboration process and resolve possible disputes;

- be neutral towards the other Actors in the ecosystem or at least transparent in case they also participate as provider of services and/or ICT Components;
- not be dominated by a single organization;
- manage the platform in such a way that it is affordable for SMEs to participate.
- ensure that Application Components are developed according to the Platform Architecture (e.g. a service-oriented architecture) to facilitate configuration and collaboration.
- ensure that Application Components contain an description to enable the use of it in various configurations
- ensure that the data shared between Application Components are tagged

3.5 Basic design of a Farm Software Ecosystem Reference Architecture

A reference architecture for a Farm Software Ecosystem describes the generic structure (concepts and relations) of specific Farm Software Ecosystems. In this section a Farm Software Ecosystem reference architecture is described according to the requirements and specifications that were defined in Section 3.40. At the end an illustrative example is provided to show how the different components fit together in practice. A table containing all the components and sub-components that are part of the Farm Software Ecosystem reference architecture can be found in Section 3.6.2 Table 9. This table is used to map the existing Farm Software Ecosystems.

3.5.1 High-level description of the reference architecture

Figure 19 provides a high-level view of the Farm Software Ecosystem reference architecture design. The architecture comprises five main components: (i) Actors, (ii) Platform, (iii) Open Software Enterprise, (iv) Business Services and (v) ICT Components. Actors provide or use ICT Components and Business Services. The Platform includes ICT Components for End-Users. The relation between the Actors and the

Platform is managed by the Open Software Enterprise (organisation). The following subsections will describe various components of the design in more detail.

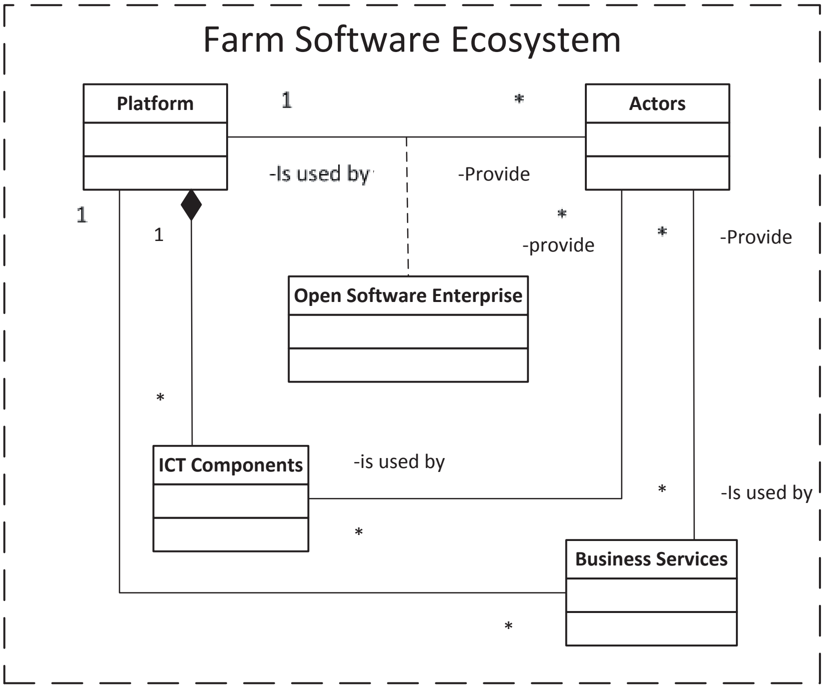


Figure 19: High-level view of the Farm Software Ecosystem Reference Architecture.

3.5.2 Platform

A platform is a set of stable components that supports diversity and evolution in a system by constraining the linkages among the other components (Baldwin and Woodard, 2008). These components are integrated and work as an integrated system. These components include software and service modules, along with an architecture that specifies how they fit together (Eisenmann et al., 2008). A Platform used within Farm Software Ecosystems must be able to support four Actor roles which are; end-users (e.g. farmers, contractors), software vendors/developer (e.g. app developer), agricultural services providers (e.g. a configurator of different systems) and the Platform orchestrator that amongst others runs the Platform. With such a Platform a configurator should be enabled to configure an ICT Component for a farmer using Application Components of multiple vendors. More details regarding these roles can be found in Section 3.5.3.

According to the requirements and specifications that were defined in Section 3.4 platform modules are defined. Each of these modules has independently value for the actor roles using the Platform. These modules can be found in Figure 20 that provides the basic architecture. In this platform architecture five modules are defined; Operating System, Development Kit, Orchestration, Security, Privacy & Trust framework and System & Data Integration.

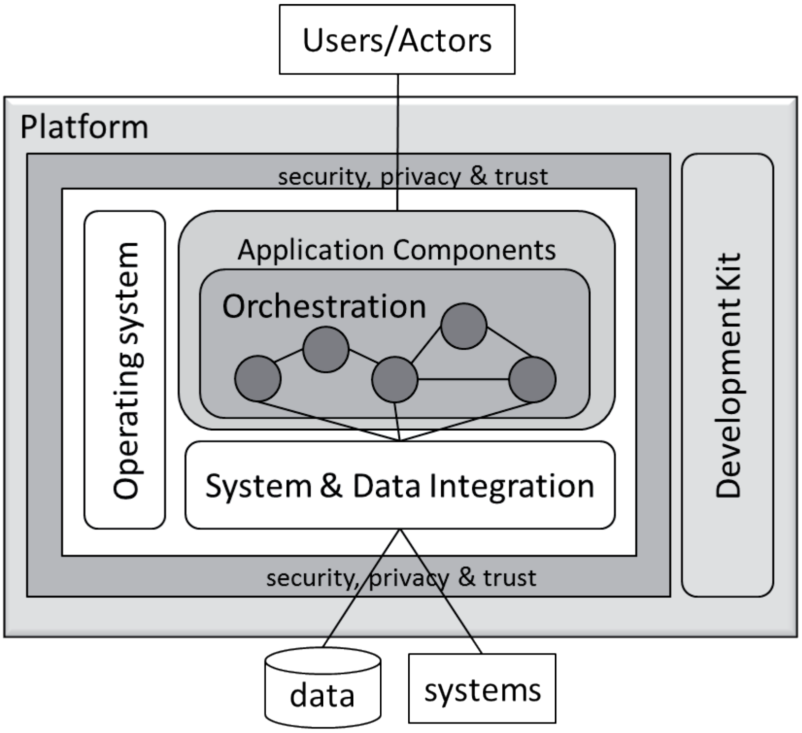


Figure 20: Basic architecture of a platform for a Farm Software Ecosystem (adapted from the FIspace platform).

The system & data integration module must provide API's to enable smooth data exchange between Application Components and enables access to distributed data repositories. To enable smooth data exchange it should contain mechanisms for data mediation to be able to handle heterogeneous data from various sources. Additionally, Payment of Application Services should be handled within this module.

The security privacy & trust framework manages all the connections. These connections can be with external data and systems or with human users. To manage these connections secure authentication and authorization methods that meet required levels of security assurance are used.

An orchestration module enables configuration. In this module linkages between different Application Services of Atomic Application Components (or Platforms) can be defined.[44] After configuring Application Services of Atomic Application Components the module intermediates between status and events of each individual Application Component. The technical specifications of this configuration are not provided in this chapter and will be presented in future research. In this chapter there is a focus on the organisational aspect of configuration which will be described in more detail in Section 3.5.4.

A software development kit is available to enable software developers to develop Application Component that can become part of the Farm Software Ecosystem. It supports the development of Application Components that can be connected to the orchestration module and should stimulate that the Application Components have a coherent look and feel regarding the User Interface.

The operating system module is needed to ensure the technical interoperability and communication between the different Platform components so that it operates as a consistent whole. One aspect is the execution of the configured ICT Components.

Regarding the deployment of the Platform and the configured ICT Components we do not provide a detailed description. Some Platforms are instantiated on a cloud Node, others on a server that is part of an enterprise and others are instantiated on a client such as a personal computer at a farm. The deployment of such a Platform and the associated ICT Components can differ for each Farm Software

[44] Within a Farm Software Ecosystem one or more Platforms can be present. In the case of configuration, one Platform will be in charge of the orchestration by connecting Application Services offered by Atomic Application Components and/or Platforms. In the case of competition between Farm Software Ecosystems and its Platforms, data from one Platform should be exported to another. In such case a data broker could be useful to exchange (and maybe store) data used by both platforms.

Ecosystem. A detailed description of a specific Platform for farming can be found in Kaloxylos et al. (2012) and Kaloxylos et al. (2014). A description of how the platform in the FIspace project was developed can be found in Verdouw et al.(2014).

3.5.3 Actors and their relationships

In a Farm Software Ecosystem multiple Actors collaborate having different organisational and operational roles. The five organisational roles are (Manikas and Hansen, 2013):

(i) *Orchestrator* - manages the software ecosystem and is responsible for its overall functioning and performance; runs the Platform, creating and applying rules, processes, business procedures, setting and monitoring quality standards and/or orchestrating the Actor relationships determining the openness;

(ii) *Niche Player* - develops or adds ICT Components to the technical Platform, producing functionality that customers or End-Users require;

(iii) *External Actor* - makes use of the possibilities of the Farm Software Ecosystem and providing indirect value to the ecosystem (e.g. by testing the platform and Application Components or by providing business services to End-Users);

(iv) *Vendor/Value Added Reseller* - makes profit from selling the ICT Components to End-Users or other vendors/value added resellers;

(v) *End-User/Customer* - purchases or obtains a complete or partial Product which is a service and/or a configuration of ICT Components (e.g. farmers, agronomists).

From an operational point of view Actors part of Farm Software Ecosystems can be classified into four main roles (Handoyo et al., 2013):

(i) *Software Vendor (Software Developer)* - is developing the technical platform, the Application Components and the Devices/Nodes.

(ii) *Agricultural Service Provider* - provides organisational Business Services including selling, customization, deployment and maintenance of ICT Components and operational services including End-User oriented functions and data. Examples of Business Services are requirements engineering, configuration and orchestration support of ICT Components.

(iii) *Infrastructure Provider* - provides a technical infrastructure (e.g. servers, networks).

(iv) *Customer/End-User* - uses ICT Components or Services that are offered by the Software Ecosystem (e.g. farmers, agronomists).

In practice, organizational and operational roles can be played by one and the same company.

3.5.4 ICT Components, configuration and orchestration

The end-user roles (e.g. farmers, contractors, agronomists) are supported in their Business Processes by ICT Components. These ICT Components will usually be a configuration of several sub-components that should collaborate seamlessly and support dynamic business collaboration processes. Therefore it is necessary to define these components in a more detailed way. The relationship between several components is presented in Figure 21.

An ICT Component is an Application Component that is deployed on a Node (e.g. local computer, internet, etc.) which supports one or more Business Processes of a company. An example of an ICT Components is the hard- and software of a terminal or PC that supports a spraying process. An Application Component is a modular, deployable, and replaceable piece of software system that encapsulates its behaviour and data and exposes these through a set of Application Interfaces (Wiederhold, 1992), for example a software module for spraying that needs to be deployed on a Node. ICT Components can be either based on Atomic Application Components or Composite Application Components. An Atomic Application Component is a piece of software, not able to share data automatically with other Application Components, e.g. a loose App on a smartphone that provides weather information. A Composite Application Component is a configuration of Atomic

Application Components that performs collective behaviour and has a coherent user-interface, in which data is automatically exchanged between (Atomic) Application Components. The orchestration module of the Platform makes them work together seamlessly as if it was one system. Therefore, the Composite Application Components require that each individual component keeps to be maintained and stays operational. When one component fails the whole composite component might fail. This requires the Farm Software Ecosystem organisation to provide functionalities that covers these issues.

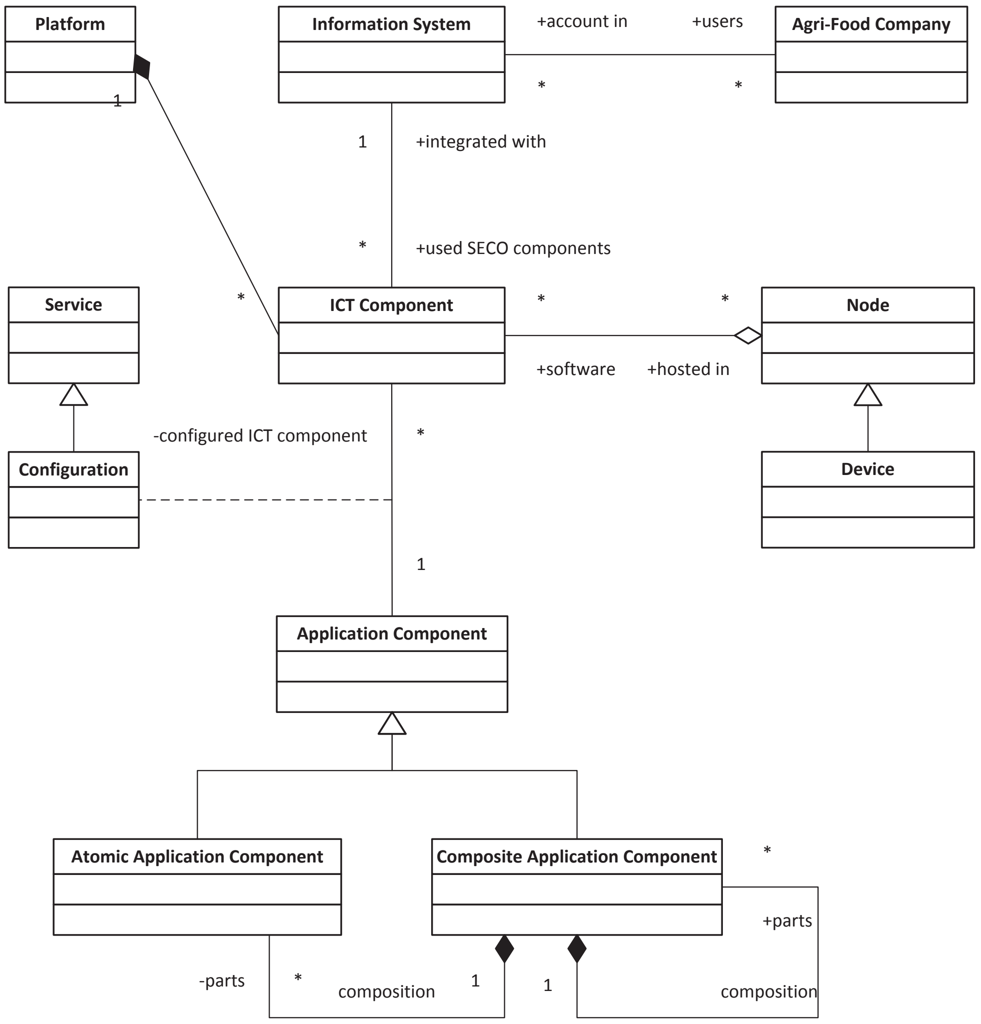


Figure 21: A UML Class diagram describing the relationships between various components in the Farm Software Ecosystem. Further explanation can be found in the text.

In a configuration process Application Services of different Application Components are selected and configured that can support Business Processes of a specific farm. The ICT Components can be offered by actors within the Farm Software Ecosystem roles software vendor or agricultural service provider and are based on various (Atomic or Composite) Application Components. A configured ICT Component can be instantiated multiple times and, in this way, support similar Business Processes. For example, an actor having the agricultural service provider role configures a Composite Application Component that can support crop protection processes. This Composite Application Component can then be instantiated on different Devices (e.g. a computer or the hardware of a Terminal) so that the resulting ICT Component can support similar Business Processes at different Farm Enterprises. Additionally, an instantiation of the Application Component part of an ICT Component can take place several times at the same Farm Enterprise. In this case, a Farm Enterprise can be supported in the crop protection Business Process, basically returning every season, with the same Application Component that is instantiated and customized to the actual situation (e.g. field, crop, etc.) of each season.

Maintenance support - a role that can be played by an agricultural service provider - is needed because (Atomic) Application Components will usually be offered by different vendors or Agricultural Service Providers. If one (Atomic) Application Component stops working, a whole (Composite) Application Component and ultimately the ICT Component and Service might malfunction as well. Therefore, a monitoring service is required that checks if each component is working correctly and if not repair or replace it with another one. These situations should also be carefully described in agreements between the users and providers of components. That will be discussed in the next section.

3.5.5 Business services and contracts

Farmers require a variety of Business Services for support. An example of an Business Service that should be offered by an agricultural service provider in a Farm Software Ecosystem is the support of a configuration process (Kruize et al., 2013). Configuration processes are knowledge intensive because the functionality of multiple Application Components and possible configurations need to be known. Therefore, a farmer might not be able to configure a Composite Application Component himself, requiring an agricultural service provider to help. In Figure 22 the relationship between the agricultural service provider role, the Business Service 'Configuration' and the use of the Platform to enable a configuration process is described.

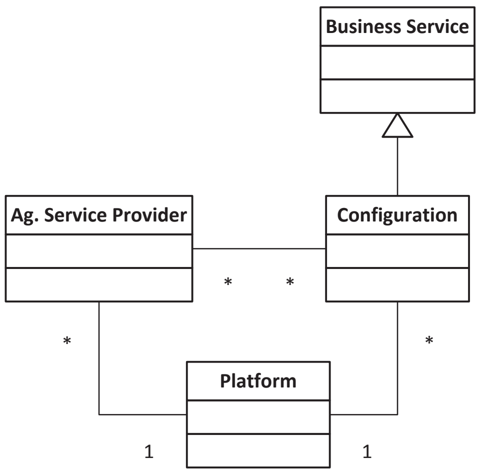


Figure 22: Relationship between the role Agricultural Service Provider, and its Business Service (Configuration). Further explanation is in the text.

To enable collaboration between Actors that perform different roles contractual arrangements are required to support collaboration using a Contract. Contracts can be formal when two roles collaborate that are part of different legal entities or less formal when two roles collaborate that are part of the same legal entity. The relationships between Actors in a certain role in which a Contract forms the formal connection are described in Figure 23.

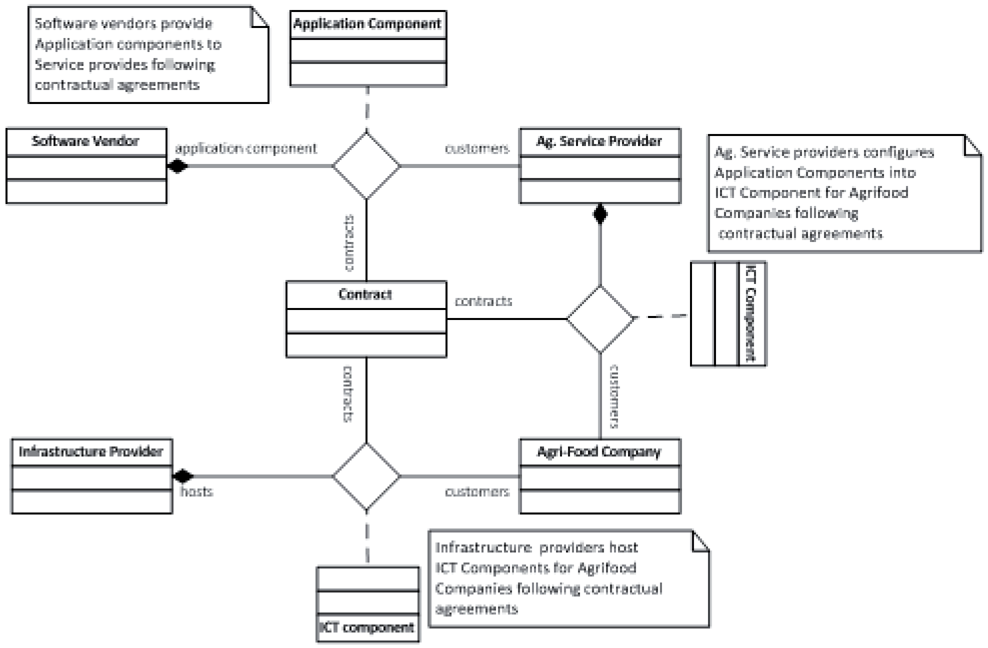

Software vendors provide Application components to Service provides following contractual agreements
Application Component
Software Vendor
application component
customers
Ag. Service Provider
Ag. Service providers configures Application Components into ICT Component for Agrifood Companies following contractual agreements
contracts
Contract
contracts
ICT Component
contracts
customers
Infrastructure Provider
hosts
customers
Agri-Food Company
Infrastructure providers host ICT Components for Agrifood Companies following contractual agreements
ICT component

Figure 23 Relationships between several Actors of a Farm Software Ecosystem in which a Contract plays a central role.

Figure 23 describes three possible contractual agreements; in reality there can be more. The first one is between a software vendor that provides a licence to an agricultural service provider to use an Application Component in a configuration Service. The Contract describes e.g. how much is paid on what basis, use and ownership of the data that is involved, etc. The second one is about collaboration between an agricultural service provider and an agri-food company. The agricultural service provider configures an Application Component which is bought by an agri-food company. This Contract contains agreements about the costs, service level, data use, etc. In the third example an agri-food company collaborates with an infrastructure provider. The agri-food company pays an infrastructure provider to host an ICT Component and eventually to provide data storage. Beside collaboration of Actors with different operational roles Actors can collaborate with Actors having the same operational role. For example, an agricultural service provider, offering a data services, can collaborate with another agricultural service provider that aggregates data into new data.

3.5.6 Open Software Enterprise

In a Farm Software Ecosystem Actors, which are part of various legal entities at different geographical locations, need to collaborate and develop software across organisational boundaries. The Open Software Enterprise that has to facilitate and orchestrate the ecosystem should fulfil this role. Different Farm Software Ecosystems can implement such an Open Software Enterprise in different ways. For example, an Open Software Enterprise can be orchestrated by a single company or a joint venture of companies that facilitates the ecosystem and possibly also the platform infrastructure. Because of these variation in implementation of Open Software Enterprises we cannot provide an exact blueprint of how they should be organized, but at least they should enhance innovation and collaboration by covering the following aspects, based on Jansen et al. (2012): i) Governance, ii) Research and Development, iii) Software Product Management, iv) Marketing and Sales and v) Consulting and Support Services. The following subsections will describe this in more detail.

3.5.6.1 Governance of a Farm Software Ecosystem

An Open Software Enterprise governs a Farm Software Ecosystem that involves the assignment of roles and decisions rights, as well as the measures and policies that enable continuation of the ecosystem. The processes that are part of the governance process group will be performed by Actors having an orchestrator role. The ability of other Actors to participate in the ecosystem and the decision making aspects of the governance determines the openness of each individual Farm Software Ecosystem. Independent of the openness of each ecosystem, an orchestrator must allow multiple Actors to develop and build ICT Components and Services using the Farm Software Ecosystem Platform. Two important aspects should be taken into account:

(i) *Partnership Model* - describes the organisational model of the Farm Software Ecosystem and makes governance policies explicit.

(ii) *IP Strategy Documentation* - describes how to use and reuse source code, data libraries, etc. which is an important basis for the contracts between Actors.

3.5.6.2 Research and development

Farm Software Ecosystems will develop continuously and should follow the state of the art in technology developments. A research and development strategy is therefore important in which the direction of the Platform, the Application Components and the ICT Components is determined and should address:

(i) *Technology Vision* – identifies the upcoming challenges that are relevant to the domain on which the software business focusses;

(ii) *Technology Research Vision* – defines the research priorities that the Actors of the Farm Software Ecosystem will focus on for the next two to three years;

(iii) *Documentation of the architecture* – describes the platform architecture as the fundamental organization of a system embodied in its components, their relationships to each other and to the environment, and the principles guiding its design and evolution (IEEE 1471:2000);

(iv) *Farm information model* – describes the organization of Farm Enterprises providing a systematic representation from different viewpoints and at various abstraction levels (Verdouw et al., 2010a). The most important information model levels are actor- business control-, business process- and data models that describe the semantics (Wolfert et al., 2010);

(v) *Application programming interfaces documentation*– defines and describes how a particular Application Component could and should be used by other components, including semantic specifications;

(vi) *Development of Collaborative Tools* - required for collaborative software development (e.g. communication, project management, issue tracking, bug tracking, globally accessible backlog, burn down chart tools etc.) (Hossain et al., 2009).

3.5.6.3 Software product management and configuration support documentation

Software Product Management is a process of managing Products (ICT Components), taking lifecycle considerations into account. Software Product Management focusses on the details of the Products such as the requirements, quality, development and marketing (Jansen et al., 2012) and will be mostly performed by Actors that have an agricultural service provider role. Actors collaborating in software product management processes should ensure that a variety of modular Application Components is available that can be configured into farm specific ICT Components, which contain Composite Application Components.

Because configuration of different Application Components is a key asset of Farm Software Ecosystems, it should be carefully documented how to do this. The available Application Components and their additional descriptions should be documented. This documentation can be supported by the farm information model that describes – or possibly *pre*scribes – certain processes or workflows in farming and the type of Application Components that could – or should – be used. Examples and tutorials will help software developers to adopt these principles quicker.

3.5.6.4 Marketing and sales

Processes of the marketing and sales process group focus on the marketing of the Products. Actors with a software vendor role or agricultural service provider role that sell Composite Application Components will perform most of these Business Processes. Marketing and sales of a Farm Software Ecosystem is essential to attract a large amount of end-users. A large amount of end-users makes the ecosystem more viable and attracts vendors to offer new Application Components.

3.5.6.5 Consulting and support services

Consulting and support service focuses on supporting end-users with implementing and using their ICT Components. These processes will be mostly performed by Actors having an agricultural service provider- or infrastructure provider role. They should provide implementation services, in which they support other Actors in configuring a Service or

ICT Component e.g. by providing documentation, reference information models, tutorials, example configurations, etc.

3.5.7 Example of a Farm Software Ecosystem

In the example Farm Software Ecosystem 'Alfa' different fictive actors collaborate. This collaboration results into an illustrational implementation of which the layout is presented in Figure 24. In this example a farm enterprise is supported in crop protection by a FMIS that is configured, using a platform, from different components that are provided by different Actors.

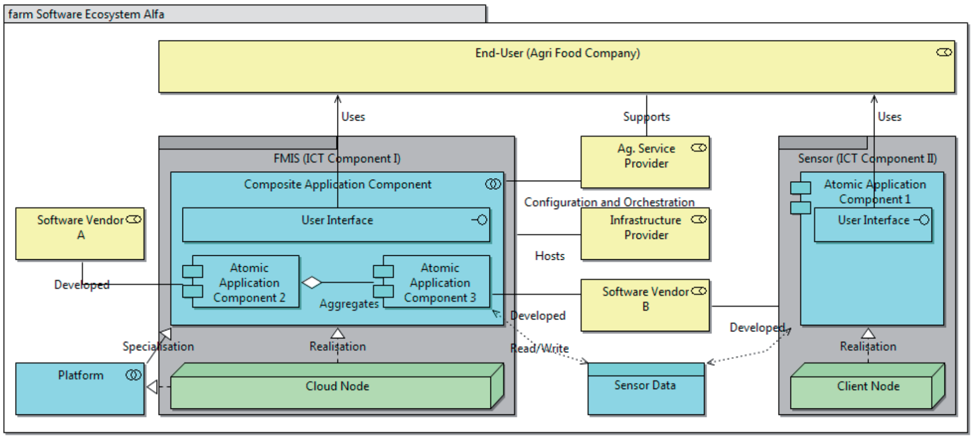


Figure 24: **Layout of the farm software ecosystem 'Alfa' showing how different components and Actors roles are related to each other. Further explanation is in the text.**[45]

The Farm Information System contains three Atomic Application Components, each offering specific Application Services. Atomic Application Component 1 offers a sensor-based field monitoring service that provides the climate conditions of a specific field. Atomic Application Component 3 can receive these sensor data and is able to analyse this data to provide insight into the risk of some diseases. Atomic Application Component 3 requires basic crop data (e.g. crop, planting time, soil type of field) that is provided by Atomic Application Component 2. The disease analysis is presented to the farmer using a coherent user interface in a Composite Application Component that is configured from

[45] The model can be downloaded using the following link: http://tinyurl.com/gwbpdco

Atomic Application components 2 and 3. The components are deployed through the Internet by a cloud Node into ICT Component I that can be used by the farmer as an FMIS to support a crop protection process. As indicated in Figure 24, ICT Component I is offered to the farmer by an agricultural service provider. This agricultural service provider has configured Atomic Application Components from Software Vendor A and B into a Composite Application Component. It should be noted that only ICT Component I is part of a platform that is hosted by an infrastructure provider, but through an Application Interface it can use the sensor data from ICT Component II. As indicated, the Sensor (ICT Component II) can also be used stand-alone by the farmer if necessary or in similar ways to support other processes (e.g. fertilization) that requires the same sensor data.

If one of the three Atomic Application Components is not working according to the requirements of the farmer it can be replaced by a similar one that is offered within Farm Software Ecosystem Alfa. This process is supported by an independent agricultural service provider. In such case a reconfiguration is needed. This can prevent a vendor lock-in as an Application Component from a vendor can be replaced by an Application Component from another vendor. Moreover, the farmer can not only change the software vendors that offer Atomic Application Components, he can change the infrastructure provider or agricultural service providers as well. This enables the Farm Enterprise to configure an Information System that is flexible and can fulfil specific requirements over time.

The Farm Software Ecosystem concept can prevent vendor lock-in as multiple vendors can compete within the same Ecosystem and specific Application Components can be exchanged. Still, the farmers could be locked into a specific Farm Software Ecosystem. To reduce the risk on a Farm Software Ecosystem lock-in the use of data standards within Farm Software Ecosystems is recommended. This would enable transferring data from one Farm Software Ecosystem to another.

3.6 Evaluation

This section describes the evaluation of the Farm Software Ecosystems reference architecture. First, the reference Architecture is verified by checking the requirements with the design. Second, a conceptual validation is presented in which the reference architecture is used for mapping the existing Farm Software Ecosystems AgroSense and Crop-R. The evaluation misses a verification regarding the usefulness of the reference architecture to asses, design and implement Farm Software Ecosystems.

3.6.1 Requirements verification

In Section 3.4.2 requirements regarding Farm Software Ecosystems are presented. Software Ecosystems fulfilling these requirements should resolve current integration challenges in farming and enable Farm Enterprise integration. To contribute to Farm Enterprise Integration this research has designed a Reference Architecture for Farm Software Ecosystems. This architecture can be used to map, asses, design and implement real worlds Farm Software Ecosystems that fulfil the requirements presented in Section 3.4.2. As a verification these requirements and our design are compared, see Table 19 that can be found in the Appendix B. For each requirement we describe what part of the Reference Architecture addresses the requirement. Based on this verification we conclude that all requirements are addressed by the reference architecture for Farm Software Ecosystems.

3.6.2 Validation of the reference architecture by mapping existing Farm Software Ecosystems

As a conceptual validation, the Farm Software Ecosystem reference architecture was used to map the existing Farm Software Ecosystems AgroSense and Crop-R. Based on such a mapping similarities and differences between real-world Farm Software Ecosystems should become visible. Additionally it provides a comparison between the Reference Architecture and real-world Farm Software Ecosystems. Too create a mapping the components (Open Software Enterprise, Platform, ICT Components, Actors and Business Services) and sub-components provided in the Reference Architecture are used to describe the real world Farm Software Ecosystem. The results of the mapping can be found in Table 9. A more detailed description of both Farm Software Ecosystems can be found online.[46]

Table 9: Mapping of the Farm Software Ecosystem Reference Architecture components and sub-components on the Farm Software Ecosystems Crop-R and AgroSense.

Components of the Farm Software Ecosystem Reference Architecture	**Sub-Components**	**Part of AgroSense**	**Part of Crop-R**
Open Software Enterprise	Open Software Enterprise structure	Yes	Yes
	A partnership model/Actor model	No	Yes
	IP Strategy Documentation	Yes	No
	Technology Vision	Yes	Yes
	Technology research vision	Yes	Yes
	Technical Architecture Documentation	Yes	Yes
	Farm Information Model	-	-
	- Actor Model	Yes	No
	- Business Control Models	No	Yes
	- Business Process Model	No	No
	- Data Model	Yes	Yes
	Application Programming Interface	Yes	Yes
	Collaborative tools	Yes	Yes
	Configuration Support documentation	No	No
Actors	Orchestrator Role	Yes	Yes
	Niche Player Role	Yes	Yes
	External Actor Role	Yes	Yes
	Vendor/Value Added Reseller Role	Yes	Yes
	End-User/Customer Role	Yes	Yes
	Software Vendor Role	Yes	Yes
	Agriculture Service Provider Role	Yes	Yes
	Infrastructure Provider Role	Yes	Yes
	Customer/End-User	Yes	Yes
Business Services Platform	Business Services Offered	Yes	Yes
	Operating System	Yes	Yes
	Orchestration Module	Partly	No

[46] The description of each Farm Software Ecosystem can be found on: http://tinyurl.com/gwbpdco

Table 9: Continued.

Components of the Farm Software Ecosystem Reference Architecture	Sub-Components	Part of AgroSense	Part of Crop-R
	System and Data integration module	Yes	Yes
	Security Privacy Trust Framework	Yes	Yes
	Development Kit	Yes	No
ICT Component	Atomic Application Components	Yes	Yes
	Composite Application Components	Yes	Yes

The mapping of the real-world farm Software Ecosystems on the reference architecture shows that many components and sub-components of the reference architecture are part of real-world Farm Software Ecosystems. Both Farm Software Ecosystems together provide a more encompassing mapping. This shows that the reference architecture can be used to map the real-world Farm Software Ecosystems.

The interviews with the CTO's showed that this mapping provides insight into real-world Farm Software Ecosystems as similarities and differences can be found that depend on the scope and objectives. Crop-R focusses more on combining data from multiple sources to provide information to farmers. Their focus is less on configuration and system integration and the orchestration of services is coded in the platform. AgroSense has more focus on integration aspects and provides a software development kit and modules can be added by other software developer to the Platform. These modules provide specific functionalities and include the coded orchestration of services. Both Platforms do not yet provide a flexible configuration approach. Consequently, both existing ecosystems AgroSense and Crop-R do not provide documentation for configuration support. This indicates that configuration, and how this should be done, is not yet well developed although it is identified as a major requirement for Farm Software Ecosystems (see Section 3.4.2). An extensive configuration approach is currently missing in both Farm Software Ecosystems. However, it should be noted that these ecosystems are still developing and that new parts are becoming available in the near future.

3.7 Discussion and conclusions

This chapter proposed a reference architecture for Farm Software Ecosystems that contributes to software development to enable smart farming. Seamless data exchange and dynamic interoperability of different Application Components to enable configuration were identified as the most important challenges. A suitable configuration approach is necessary to link Application Components to each other in a meaningful and coherent way. The resulting Composite Application Component needs to be deployed at Farm Enterprises and be supported by a agricultural service provider. An open software enterprise governs and facilitates the ecosystem and the collaboration between Actors. The reference architecture can be used to map current real-world Farm Software Ecosystems and it is expected to be able to design and implement future Farm Software Ecosystems.

It was demonstrated how Farm Software Ecosystems can align software development with multiple Actors in a distributed environment. In such an ecosystem small players can focus on development of small Application Components while relying on larger ICT players that provide general infrastructural components (data storage, servers, etc.) or more complex analyses (e.g. super computers, big data) by generic software components (see Figure 18). In this way innovation can be stimulated giving small start-up companies a fair chance to accelerate their innovative product and gain market share. This approach is currently stimulated in the FIWARE accelerator program,[47] which is partly connected with the FIspace Platform, and in which agriculture is an important focus area. For end-users in the ecosystem it is expected that the alignment of software development will lead to better and more affordable solutions because innovation costs are shared. Additionally, the solution can become more flexible because end-users can change the different sub-components of an integrated solution, which will stimulate competition at that level.

For integration of different sub-components it is important that the configuration process is supported in an appropriate and well-defined

[47] www.fiware.org/fiware-accelerator-programme

manner. It was concluded that both evaluated Farm Software Ecosystems, AgroSense and Crop-R, are lacking such a support. The FIspace Ecosystem and platform provides a business collaboration core for that purpose (Kruize et al., 2014; Verdouw et al., 2014), although this platform and its ecosystem is still being established. Reference information models - especially Business Process models - could play an important role in a configuration process by describing the farm Business Processes at various levels and from different viewpoints (Verdouw et al., 2010a). Such reference information models are also still poorly defined in the current FIspace architecture. They should also refer to common standards as much as possible to ensure interoperability at higher integration levels and acceptation by many users at a global level. The reference models and standards could be offered by an (agricultural) service provider to one or more Farm Software Ecosystems. Future research should focus on developing these information models for Farm Software Ecosystems to provide knowledge about software configuration.

The reference architecture presented in this chapter is supposed to be a common basis for various Instances of Farm Software Ecosystems that could compete with each other. However, the open software enterprises should prevent ecosystem lock-ins, *i.e.* that End-Users are hindered in substituting components from one ecosystem to the other. To that end, a kind of federated structure between the different Farm Software Ecosystems is needed. This can be reached by using common standards mainly focussing on platform and semantic interoperability.

Although the general market principles and business models for Farm Software Ecosystems will not be fundamentally different in comparison to the past, there are definitely new modes of collaboration emerging that make it necessary for companies to reconsider their strategies (Porter and Heppelmann, 2014). Developers of application and ICT Components should realize that their solution will be part of larger integrated solutions at a higher level. As a consequence, they should not focus at the final end-user so much but establish collaboration with agricultural service providers supporting configuration and other Application Component developers. It also becomes more important to comply with common standards as much as possible to increase the potential usage of a component at a global level. Agricultural Service providers could focus on specific end-users groups (e.g. farmers, input suppliers) combining personal advice and communication with the best customized configuration of ICT- and Application Components based on flexible Contracts. These are just a few examples of developments in new business models that can be expected from Farm Software Ecosystem development. Examples from other sectors show that successful business models emerge gradually as the ecosystem develops (Van 't Spijker, 2014). An essential issue in these developments is about data ownership, security, privacy and trust. Although the importance was emphasised in this chapter, further research is needed on how to deal with this topic in the context of open dynamic Farm Software Ecosystems.

From this discussion it can be concluded that there are still technical challenges to be met, especially in the configuration of different components into integrated solutions. It is also clear that there are still many organizational developments needed to successfully develop Farm Software Ecosystems. The reference architecture in this chapter can help to guide these developments, which can enrich the architecture.

4

A reference information model for configuring farm business processes and supporting application services

This chapter is submitted to an international journal:

Kruize, J.W., Verdouw, C.N., Wolfert, J.,
Scholten, H., & Beulens, A. J.M.

Abstract

Integration of ICT components, such as sensing and monitoring devices, software packages, decision support systems and other farm equipment, into an integrated farm information system is challenging. On the one hand, current ICT components often do not support required farm business processes or do not have appropriate application interfaces to enable integration. On the other hand, the business processes of each farm differ, which makes the selection and combination of complementary ICT Components tough. Therefore, a Best-of-Breed approach combined with ICT mass customisation is proposed to allow for customized software systems by configuration of standardized ICT Components that are supplied by multiple vendors. To enable such an approach in agriculture, a reference information model is required that supports (1) the development of ICT Components and a platform to connect and run these components, (2) selection of ICT Components based on the business processes they should support and (3) configuration of these different ICT Components, using a platform, into integrated farm information systems. Hence, this chapter presents a reference information model, named RAAgE 2.0, supporting configuration of farm business processes and the supporting application services as an initial step to enable a Best-of-Breed, combined with a ICT mass customisation approach. This reference model can be used by software developers to gain insight in farm business processes, farm business process configuration and ICT Mass Customisation.

Keywords: Farm information systems, Best-of-Breed Approach, Farm Software Ecosystems, Business Processes, Software Configuration, mass customisation

4.1 Introduction

In the current trend of precision agriculture, farming[48] is becoming more high-tech by the introduction of new sensing and monitoring devices, software and equipment. These technologies enable to control farm processes at a high spatial and temporal resolution. Smart farming extends the concept Precision Agriculture by enhancing farm management with data on context, situation and location (Wolfert et al., 2014). These data enable location awareness and better predictions to improve corrective operations by the farmer. As a consequence, ICT (Information and Communication Technologies) and other technologies are an increasingly important asset in farm enterprises. Additionally, seamless data exchange between different components in and outside organisations becomes of crucial importance. Figure 25 summarizes the concept of smart farming along the management cycle.

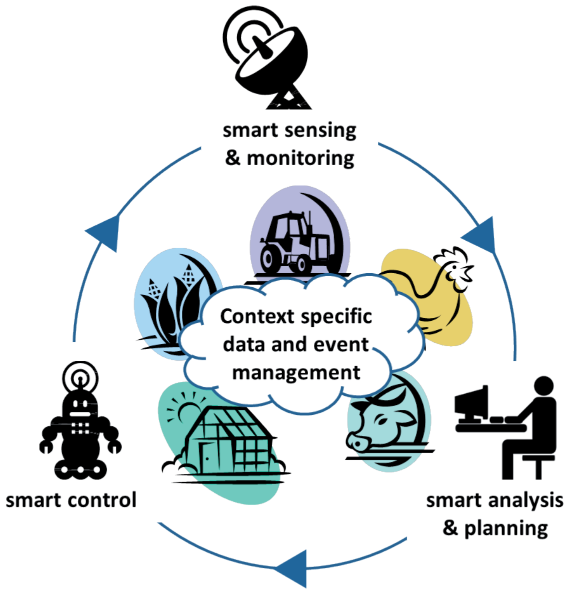

smart sensing & monitoring
Context specific data and event management
smart control
smart analysis & planning

Figure 25: Generic sense-model-act cycle enhanced by cloud-based event management that underpins smart farming (Wolfert et al., 2014).

Practicing a farm management style that utilizes all the benefits technologies can offer is hard to accomplish. Especially changing

[48] A farm enterprise can be an arable farm, livestock farm or horticultural farm. In this chapter we focus on arable farm enterprises.

towards more advanced farm management styles such as smart farming is challenging. A key problem is the integration of all kind of ICT Components into an integrated farm management information system (Pierce et al., 1999). Currently, farm enterprises use several standalone software packages because each package individually does not cover the support of a complete farm business process cycle (Kruize et al., 2013). These software packages overlap in functionality or often do not cover all desired functionalities, which is an undesirable solution. An overarching Farm Management Information System (FMIS), developed by a single vendor, which could be used worldwide by all farmers, is neither a feasible nor a desirable solution from a competitive point of view. An ICT Mass Customisation approach (Verdouw et al., 2010a), in combination with a Best-of-Breed approach (Light et al., 2001), is considered as an appropriate way to enable farmers to change towards more advanced farm management styles. ICT Mass Customisation combines advantages of standard and customised software by enabling on-demand configuration of information systems from components with standardised interfaces (Verdouw et al., 2010a). These components could be supplied by different software vendors to enable Best-of-Breed solutions. In software engineering literature, mass customization of software products is known as software product lines (Clements and Northrop, 2002) or software product families (Pohl et al., 2005). Development of a software product line in agriculture could provide farmers to select application services[49] they require based on their business process and configure the related ICT Components into an integrated, multi-vendor farm management information system.

Research is moving into this direction as there is a trend of improving the integrating capabilities of farm ICT Components by proposing standardized infrastructures that supports integration of components of multiple vendors (Iftikhar and Pedersen, 2011; Kaloxylos et al., 2012; Kaloxylos et al., 2014; Nash et al., 2009; Steinberger et al., 2009; Wolfert et al., 2010) . Kruize et al. (2016) contributed to the development of these standardized infrastructures by introducing the general concept of software ecosystems for farming and designed a reference architecture for Farm Software Ecosystems that meets this

[49] An Application Service exposes automated behaviour and is realized by an ICT component.

approach. Such an ecosystem consists of a common software platform that provides a broad range of ICT Components and services that are offered by various vendors to end-users, *i.e.* farmers and other stakeholders around the farm production process. Farm Software Ecosystems contribute to Best-of-Breed solutions at it encourages collaboration between actors with complementary services and competition between actors with similar services. The critical success factor of such an farm software ecosystem is that multiple ICT Components can be configured into an information system that covers the needs of a farm enterprise or a specific business process within that farm. However, Farm Software Ecosystems are currently unable to adequately facilitate ICT Component configuration (Kruize et al., 2016). To enable ICT Component configuration, development of a software product line for Farm Software Ecosystems should be supported. When organisations decides to set up a software product line it will face the following issues (i) how the particular software product is specified and (ii) how the software product line itself is specified (Benavides et al., 2010). Reference information models can play an important role to set-up a software product line as it can model how the product is specified by describing the relations and provide generic descriptions of an object system (Benavides et al., 2010; Verdouw et al., 2010a; Verdouw et al., 2010b; Verdouw and Wolfert, 2009). Reference Information Models are standardized taxonomies that represent possible configuration options of process instances and interdependencies that exist between components or features, including rules for permitted combinations (Verdouw et al., 2010a). They provide various contexts or viewpoints on business processes at different management levels (e.g. tactical and operational) and can be provided to service configurators – preferably as software-as-a-service – to design, configure and implement integrated farm-specific information systems.

The objective of this chapter is to design a reference information model to enable configuration of ICT Components into integrated farm management information systems as such model is still missing. This reference information model should support software product line development within Farm Software Ecosystems and enable the realisation of farm information systems that are based on a Best-of-Breed and ICT Mass Customisation approach.

The remainder of this chapter describes first the used methodology (Section 4.2). After a thorough requirement analysis in Section 4.3, the design of the reference information model is presented in Section 4.4. In Section 4.5, the model is verified by applying it to the case of late blight protection in potatoes, concluded by a discussion and conclusion in Section 4.6.

4.2 Materials and methods

4.2.1 RAAgE 1.0

RAAgE 1.0 is a reference architecture that supports modelling of specific farm enterprise architectures (Kruize et al., 2013). These farm enterprise architectures depict farm business processes and ICT Components, which can be supplied by multiple vendors, supporting these processes. In RAAgE 1.0 farm business processes are categorized in process groups (Figure 26). Each process group contains business processes part of this specific group. Business processes within a process group are classified in a hierarchical way.

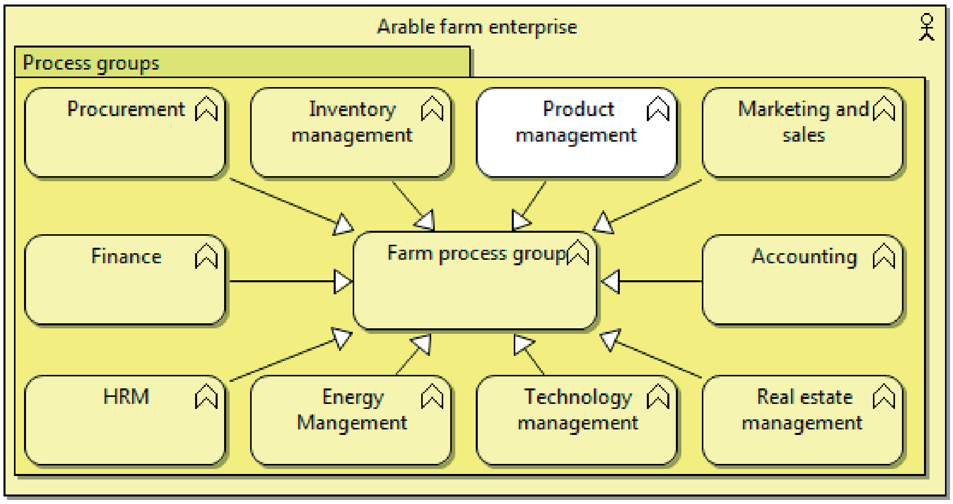


Figure 26: Farm Process Groups; Product Management is the focus of this research.

An abstract description of a farm enterprise, based on RAAgE, is subdivided into a business-, application- and technology layer (Figure

27). The business layer represents the farm enterprise including the elements farm process group and farm business process. The application layer includes one part of the ICT Component, namely the application service, application function, application interface and application component. The technology layer includes the other part, namely the infrastructure's function, services, interface, node, system software, hardware, communication path and network. Furthermore, to enable exchange between ICT Components elements for electronic information exchange are described. The elements include an information object that is realized by data that aggregate a data structure and are associated with a data carrier (e.g. XML, Json).

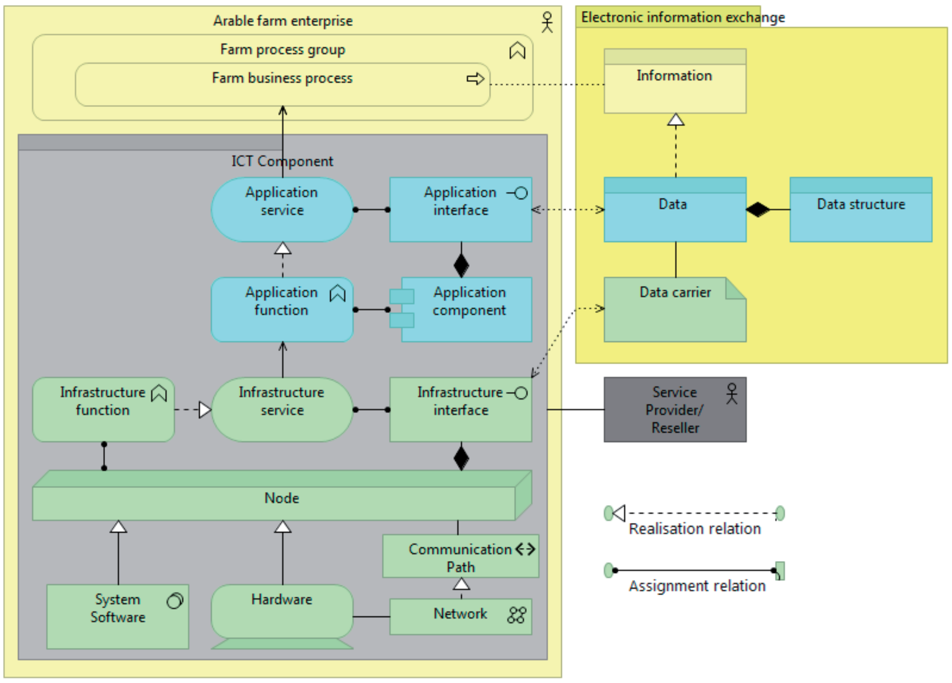


Figure 27: View on the elements from Archimate describing the arable farm business context, ICT Components and elements for electronic information exchange between ICT Components. For more details see Kruize et al. (2013).

RAAgE 1.0 is able to describe management control and operating control of farm business processes related to crop production and supporting ICT Components at a high level. However, it currently lacks describing these farm business processes in detail. Detailed description showing relations between farm business processes and ICT Components can specify the software product, *i.e.* the supporting farm information system. With a model able to depict detailed business processes of farms and the supporting software product a ICT mass customisation approach in agriculture can be enabled. This chapter extends RAAgE 1.0 into RAAgE 2.0 to support configuration of crop production business processes (cultivation processes) and the supporting software product. To support business process configuration the model will be extended with a (crop) product and resources reference model. By extending RAAgE the model will enable more detailed modelling of farm enterprises architectures. These detailed enterprise architectures can depict differences and commonalities between farms. Application components and services (part of an ICT Component) will be linked to the farm business processes they support. RAAgE uses the business modelling language ArchiMate[50] which is especially powerful to represent enterprise architectures and depict the relations between business processes and ICT Components. To develop individual parts of the reference model (the crop Product and Resource model) the ISA95 standard is used.[51] ISA95 is a widely supported standard and is used to develop automated interfaces between enterprise and control systems. The ISA95 standard provides reference models that can be used to describe parts of the object system that needs to be modelled.

4.2.2 Methodology: Design Oriented Research

In this chapter we present a Design Oriented Research (DOR). The purpose of DOR is to create innovative artefacts that extend the boundaries of human and organizational capabilities (Hevner et al., 2004). These artefacts are developed based on a design process. Our design process is based on the Information System Research Framework of Hevner et al. (2004). Therefore we first created requirements based on the environment (farm enterprises) and the knowledge base (literature). Second, we started the design process. Third, the design was evaluated within the environment.

To specify the requirements we follow guidelines from Sommerville (2011) and Nuseibeh and Easterbrook (2000). First, we identify the

[50] http://www.opengroup.org/subjectareas/enterprise/archimate
[51] https://isa-95.com/

system boundaries, the goals of the main stakeholders and elicited functional requirements for a configurable Farm Enterprise Reference Model. The functional requirements are based on literature and re-use of the initial interviews used to develop RAAgE 1.0 (Kruize et al., 2013). These interviews have been held with three farmers from different parts of the Netherlands. In the design RAAgE 1.0 is chosen as a basis and extended into RAAgE 2.0 to enable configuration. In this design process the requirements are used as a guideline. RAAgE 2.0 is evaluated by verifying the requirements and by developing an example how farm business processes and a supporting information system can be configured. This exemplar is based on the FIspace Trial 'Crop Protection and Information Sharing'.[52] In this trial an integrated information system is developed containing components of multiple vendors.

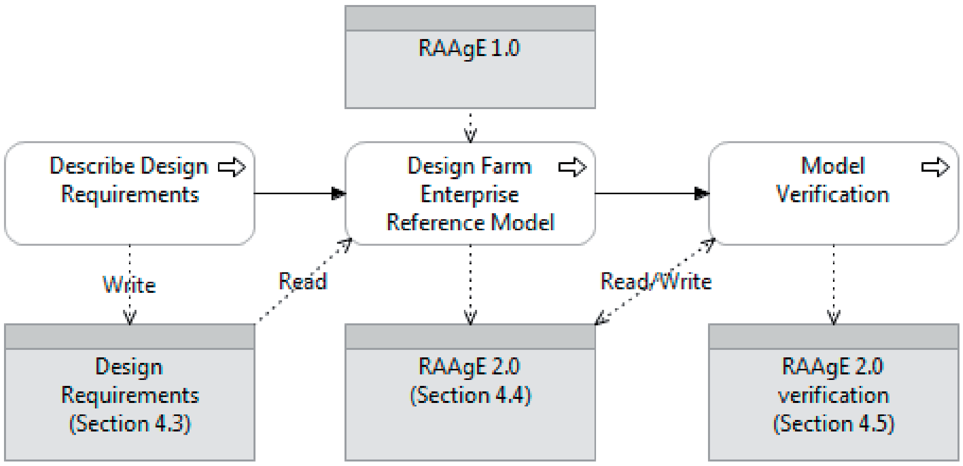

RAAgE 1.0
Describe Design Requirements
Design Farm Enterprise Reference Model
Model Verification
Write
Read
Read/Write
Design Requirements (Section 4.3)
RAAgE 2.0 (Section 4.4)
RAAgE 2.0 verification (Section 4.5)

Figure 28: Description of approach.

4.3 Requirements analysis

The requirements analysis section contains four parts. First, the object system is defined. Second, stakeholders that can use RAAgE 2.0 are described. Third, functional requirements for RAAgE 2.0 are described. Fourth, the design specifications of RAAgE 2.0 are presented which are based on the functional requirements.

[52] http://www.fispace.eu/apps.html#crop-protection

4.3.1 Object system

To define the object system for the reference information model that has to be developed, we chose the farm business process as the minimal, generic model of all possible processes (Beers et al., 1994; Wolfert, 2002). A farm business process, related to crop production, turns inputs (fields, raw materials, human labour) aided by ICT Components and other technology and resources (e.g. sensors, implements, terminals and tractors) into products (*i.e.* crops). The object is system is depicted in Figure 29.

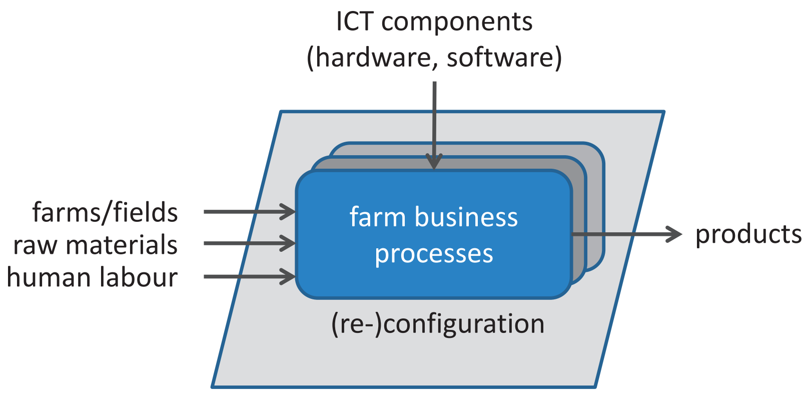


Figure 29 Delineation of the object system of this study.

These business processes of individual farms can differ, while producing the same or a similar product. This can be due to e.g. climatological circumstances or regulations but also due to farmer's individual strategies. The farm-specific business processes can represent a certain farm management style (e.g. precision farming, basic farming, smart farming). These different management styles can be seen as variants of a similar business process. Farmers can shift from one business process variant to another to change their management style or because the farm objects system changes over time (e.g. by unexpected weather changes, crop growth, changing regulations, new ICT Components, technology and resources). When the business processes change, consequently the supporting ICT Components need to change. This means that ICT Components of multiple vendors need to be reconfigured.

Business process configuration and the software configuration of an integrated farm information system are two separated steps. The business processes describe the farm specific activities of farmers to realize a certain product. This process should be supported by a farm specific farm information system that is configured based on the specific business processes. In this chapter we focus on the configuration of business processes and describe the relation with the application services to enable information system configuration. Details how to configure multiple application services and the related ICT Components into a working information system using a platform is in this chapter out of scope. Kaloxylos et al. (2014) have presented a first design of such a platform, named FIspace, for agriculture.

A reference information model can help describing this complex object system in a coherent way and enable the modelling of business processes and the related ICT Components of multiple vendors (*i.e.* the software product). Additionally, such a model can enable reusability of ICT Components to support different business process variants.

4.3.2 Stakeholders

Insight into the farm object system can be used by software developers to (i) develop a software product line for Farm Software Ecosystems and (ii) to gain insight into farm business processes to develop new ICT Components. Development of a software product line can result into ICT Mass Customisation as it enables configuration of ICT Components into integrated farm information system. Currently, this configuration is not provided as a business service, which is one of the bottlenecks hindering farm enterprise integration (Kruize et al., 2016). Therefore, actors should become part of Farm Software Ecosystems that can support ICT Component configuration. A role that can support configuration are business architects. Business architects should be experts in the agricultural domain and in charge of configuring ICT Components of multiple vendors. A farmer could hire a business architect to improve processes at his farm. A business architect should use RAAgE 2.0 to gain insight into the as-is business processes, products and resources of farmers. Based on these insights business architects can configure ICT Components. Furthermore, processes can be improved by suggesting process variants and the required ICT Components. In conclusion, the main stakeholders that can use RAAgE 2.0 are software developers, business architects and farmers.

4.3.3 Functional requirements for RAAgE 2.0

To enable ICT Mass Customisation a farm reference information model should be developed. This reference model should describe the relations between different (software) resources, processes and the purpose the resources are used for (Verdouw et al., 2010b). Based on insight into the business processes (e.g. used raw materials, resources and labour availability), the purpose and its relations ICT Component configurations can be determined. To support configuration based on an ICT Mass Customisation, in combination with a Best-of-Breed approach, the reference model should be able to:

(i) represent the object system (Farm Business Processes, Resources, Products) in a sufficient level of detail (La Rosa et al., 2008; Verdouw et al., 2010a; Verdouw et al., 2010b);
(ii) enable configuration and re-configuration of farm business processes (Kruize et al., 2016; Kruize et al., 2013);
(iii) ensure reuse of ICT Components within different configurations and to contribute to integrated farm information systems as a technically feasible and affordable solution (Aubert et al., 2012; Fountas et al., 2005; Fountas et al., 2006; Sørensen et al., 2010a);
(iv) ensure that multiple vendors collaborate in providing ICT Components that can be configured into Best-of-Breed systems (Kruize et al., 2016).

The next subsections describe these functional requirements in more detail.

4.3.3.1 Representation of the farm object system

To be able to create better representations of the farm object system, including its over time versatility, RAAgE 1.0 needed to be extended. Therefore, RAAgE 2.0 was designed and enables modelling of the farm Business Processes and its relation with resources used in order to deliver certain products (Figure 29). In these descriptions relations between different resources should be known together with the processes and the purpose the resources are used for (Verdouw et al., 2010b). Consequently, RAAgE 2.0 must provide detailed reference models to describe (i) the business processes itself and its variants (ii) the products they deliver and (iii) resources that are used.

4.3.3.2 Configuration and re-configuration of farm business processes

Every farm and its business processes are unique because each farm context is specific. Business processes of farms are dynamic and subject to changes due to the external environment (e.g. weather or market changes) or events (e.g. machine breakdown, pests and diseases). Furthermore, it is possible that a farmer re-configures his business processes to improve his management. For example, if a farmer decides to apply fertilizers at a variable rate within fields this changes the business processes and adds significant requirements to the ICT Components that are needed. RAAgE 2.0 aims to support this dynamic re-configuration of business processes and its resources. Its main complexity lies first in describing the configurations of the baseline (as-is) enterprise architecture that includes farm business processes and the supporting ICT Components. Second, in describing the configuration process to facilitate change (re-configuration) and depict a target (to-be) enterprise architecture. To be able to handle this complexity, RAAgE 2.0 should (i) be able to support business process configuration (ii) describe clearly how configuration is supported. Additionally, as the farm object system changes over time it must be possible to update and extend RAAgE 2.0. Therefore RAAgE 2.0 should distinguish different time phases which are:

(i) Design-Time for updating and extending RAAgE 2.0 to improve the configuration process;

(ii) Configuration-Time to enable configuring as-is and to-be descriptions (output) of the farm business processes and the supporting ICT Components based on the current resources and products (inputs) ;

(iii) Run-Time for executing the configured instances to perform farming processes.

4.3.3.3 Reuse of ICT Components and configuration knowledge

A possible implication of the aforementioned (re-)configuration requirements is that this can lead to labour-intensive and costly solutions, especially if configurations need to be developed from scratch each time. Hence, an ICT Mass Customisation approach is required in which standardized software components can be integrated into context-specific software. To enable such an approach most of the dominant ERP vendors use reference models and supporting software tools to make enterprise specific information systems (van der Aalst et al., 2006). The application of reference models is motivated by the "Design by Reuse" paradigm. Reference models (also referred to as templates) are used to streamline and accelerate the design of specific configurations by providing a generic solution (Fettke and Loos, 2003; Rosemann and van der Aalst, 2007b). RAAgE 2.0 should also support the reuse of configuration knowledge by including templates that consist of often used combinations of products, processes supporting ICT components. In this way, reference models will support the creation of affordable, integrated ICT configurations.

4.3.3.4 Collaboration of multiple vendors in a Best-of-Breed approach

In agriculture the creation of one overarching system developed by one software vendor is neither a feasible nor – from a competitive point of view – a desirable solution. Therefore, a Best-of-Breed approach is required in which ICT Components are provided by multiple vendors and are offered to multiple end-users (Kruize et al., 2016). RAAgE 2.0 should account for this in modelling configurations of various business processes, in which ICT Components play a role. Seamless data exchange between different components is of crucial importance in this. Therefore, RAAgE 2.0 should support ICT Mass Customization, which can be realized within a Farm Software Ecosystem. The purpose of Farm Software Ecosystems is that integrated FMISs are offered based on a configuration of multiple ICT Components of multiple vendors (Kruize et al., 2014; Verdouw et al., 2014).This configuration process can be supported by a business architect role. RAAgE 2.0 should support a business architect in configuring multiple ICT Components of multiple vendors into an integrated solution.

4.3.4 Design specification

Based on the requirements the following list of specifications for designing RAAgE 2.0 can be derived:

- A reference architecture of agricultural enterprises that:
    - is based on RAAgE 1.0
    - contains a *product-, process-* and *resource* reference model;
    - contains a *configuration mechanism* that is able to individualize farm-specific business processes based on the product-, process- and resource reference models;
    - contains *templates* of configurations that are often used.
- Different time phases need to be distinguished:
    - *Design-time* in which a general configuration is developed of the farm business processes;
    - *Configuration-time* in which a farm-specific process model is developed, including a description of specific ICT Components that act as an integrated FMIS;
    - *Run-time* in which a specific instance of the model can be run as an executable process to support farmers.

4.4 Design of a reference information model: RAAgE 2.0

This section contains two main parts: (1) a Meta Model of the reference information model RAAgE 2.0 providing an overview of its components and purpose and (2) details of each individual component that supports the configuration of crop production business processes.

4.4.1 Meta Model of RAAgE 2.0

The meta-model of RAAgE 2.0 is presented in Figure 30.

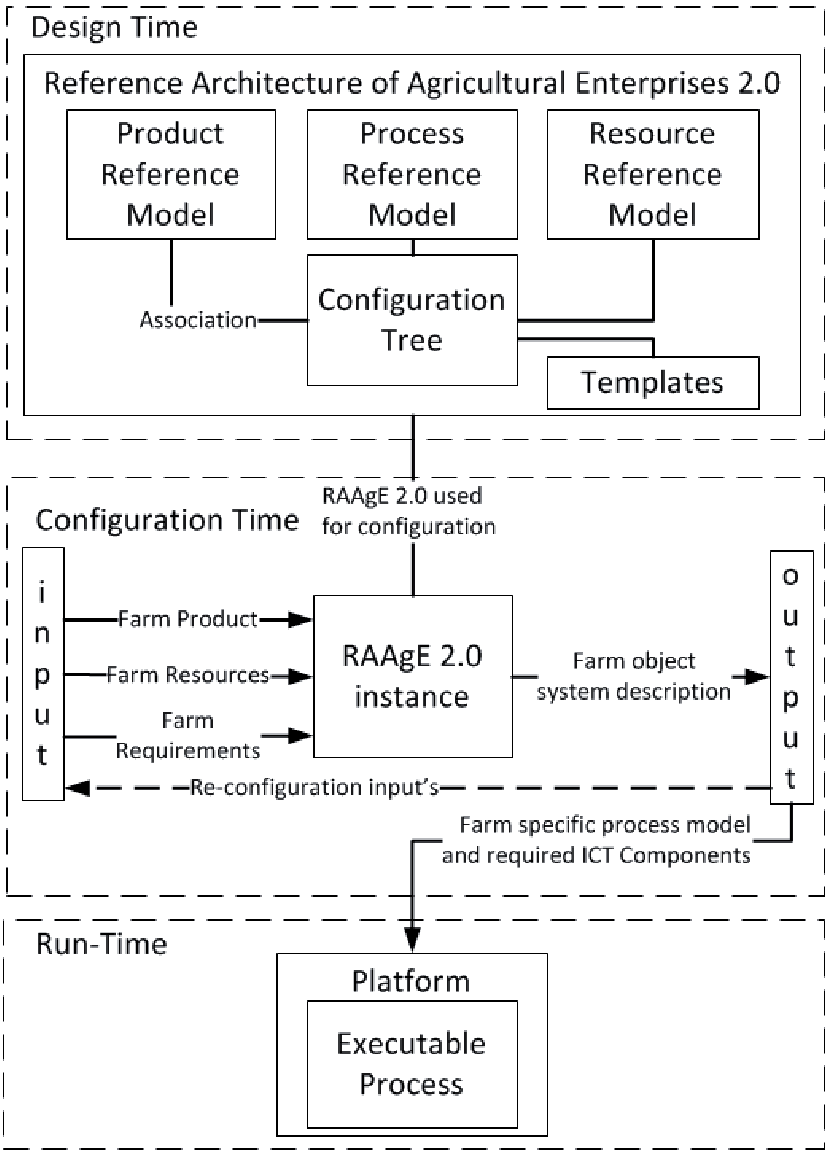


Figure 30: Meta Model of RAAgE 2.0 (explained in the text).

In this meta model three time horizons are distinguished: *design time*, *configuration time* and *run-time.*

In design time, the reference architecture of RAAgE 2.0 is defined, modelled and updated or extended to improve the configuration process. The reference architecture comprises three views, i.e. product, process and resource reference models. The product reference model defines the variety of products that farmers can produce in order to fulfil customers demand. The business process reference model defines the farm processes that are needed to produce the required products as defined in the product reference model. The specific processes include rules that determine the product, process and resource combinations. The resources reference model defines the resources, including farm equipment, input material, personnel and information systems, that are needed to execute the farm processes as defined in the business process model. The dependencies between these views models are defined in rules that define the possible combinations of the products, processes and resources and that constrain the configuration of farm-specific models i.e. instances.

Furthermore, RAAgE 2.0 includes a configuration tree and templates, which support the configuration of farm-specific models i.e. instances. A configuration tree provides a method to instantiate specific farm object system descriptions. It is a wizard-like step-by-step plan that guides users through the configuration process, taking into account the constraints as defined in the rules of the reference model. Templates describe a set of pre-configured product, process and resource models for typical cases.

In configuration time RAAgE 2.0 is used to facilitate the configuration process. Based on the inputs; farm product, farm resources and farm requirements, a farm object system description is created. This description shows the relation between the farm specific business processes, products and resources. These farm business processes are required to produce a Farm Product (e.g. potatoes, sugar beets) taking the available human-, software- and hardware resources into account. To optimize the production of farm products the object system description can be used to reconfigure the business processes and application services (software) by changing the input parameters (e.g. change of the farm requirements, change of the resources by buying/hiring additional resources). The configuration process is supported by the configuration tree to create new descriptions or templates. The resulted object system description can be used to configure a platform and the ICT Components.

In run-time the object system description is used in a platform to configure the application services (software) and support the crop production business. This platform should enable configuration of application services and the related ICT Components in a manner that an integrated information system is formed. The created software instance should support farm business processes in run-time.

The focus of the present chapter is on the design-time Reference Architecture i.e. RAAgE 2.0. The components of this reference architecture will be further elaborated in the next section.

4.4.2 The components of RAAgE 2.0

4.4.2.1 Product reference model

Farm enterprises produce various products for different markets (e.g. the organic or conventional market). To model these products, a Product Reference Model is defined based on ISA95, see Figure 31. ISA-95 is a reference model that focusses on the integration of ERP and production automation in the manufacturing industry. The designed product model defines Farm Product Classes, e.g. with the ID 'Crop'. This Farm Product Class has multiple Farm Product Class Properties: cultivar, specific market, production procedure and product state. Figure 31displays only the farm product class property cultivar. A more extensive example is provided by Table 10.

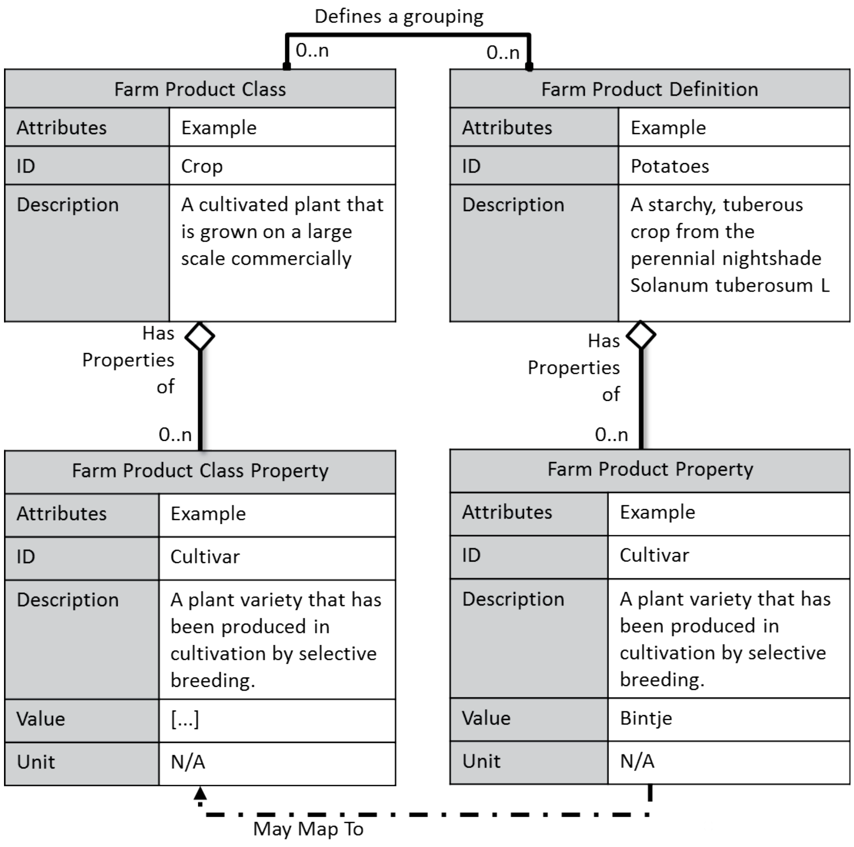


Figure 31: Part of the Product Reference Model of RAAgE 2.0 (based on ISA95).

4.4.2.2 Resource reference model

Resources are required to produce farm products. In the Resource Reference Model four different resources are defined (1) Personnel, (2) Equipment, (3) Software and (4) Raw materials (based on ISA95). These resources can be available internally (owned/bought by the enterprise) or externally (hired by the enterprise). For sake of simplicity, we do not distinguish between internal and external available resources.

These resources are required by certain business processes. For example, an executable canola seeding work-order requires at least one operator (Personnel) and Seeding Equipment (an equipment assembly including a tractor and a seeding machine). Additionally, raw materials such as 200 kg of canola seeds are required to sow some hectares. A reference model of resources is required to categorize resources and be able to link them to the business processes. The focus of this chapter is to link software resources (application services) to the business processes. For all four resource types we created an individual reference model and a description that can be found in the next sections.

Personnel reference model

The Personnel Reference Model is based on ISA-95 containing four classes: Personnel, Personnel Class Property, Person Class and Person Class Property which is sufficient to model personnel on a farm (Figure 32). In this example the Person, Thomas Thomson, is one of the Operators. Relevant for Operators is what kind of skills they have (Operator Type). In this example the Person has General Machine Operating Skills meaning that the person can operate Seeding Equipment, but is not able to spray fields because additional certificates are required to operate such kind of tasks.

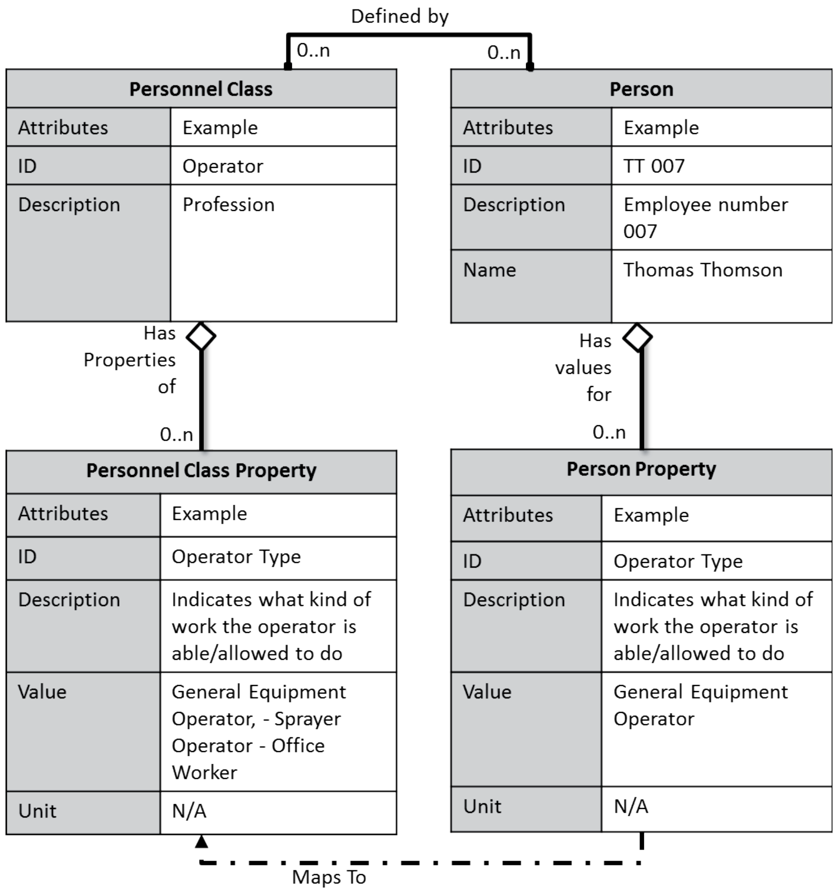


Figure 32: Part of the ISA95 Personnel model, with agricultural examples.

Equipment reference model

At farm enterprises all kind of Equipment is used for cultivating crops (e.g. tractors, seeding devices, implements, terminals). This equipment often contains a software part that is able to exchange data with other application components. For example, terminals, as part of a spraying assembly, are able to exchange task data with farm management information systems that is transferred using a USB stick. The software part of this equipment is modelled in the next section.

Figure 33 presents the Equipment model containing four classes: Equipment Class, Equipment Class Property, Equipment and Equipment Property. A piece of Equipment at a farm enterprise has for example the ID Spraying Assembly 01. This Equipment is defined by the Equipment Class and has the ID Spraying Equipment. The spraying equipment has additional properties describing at what kind of resolution the equipment is able to spray. Some spraying assemblies are not able to execute in-field variable spraying. Other Equipment attributes describe the ability of the assembly to control the different nozzles in different sections.

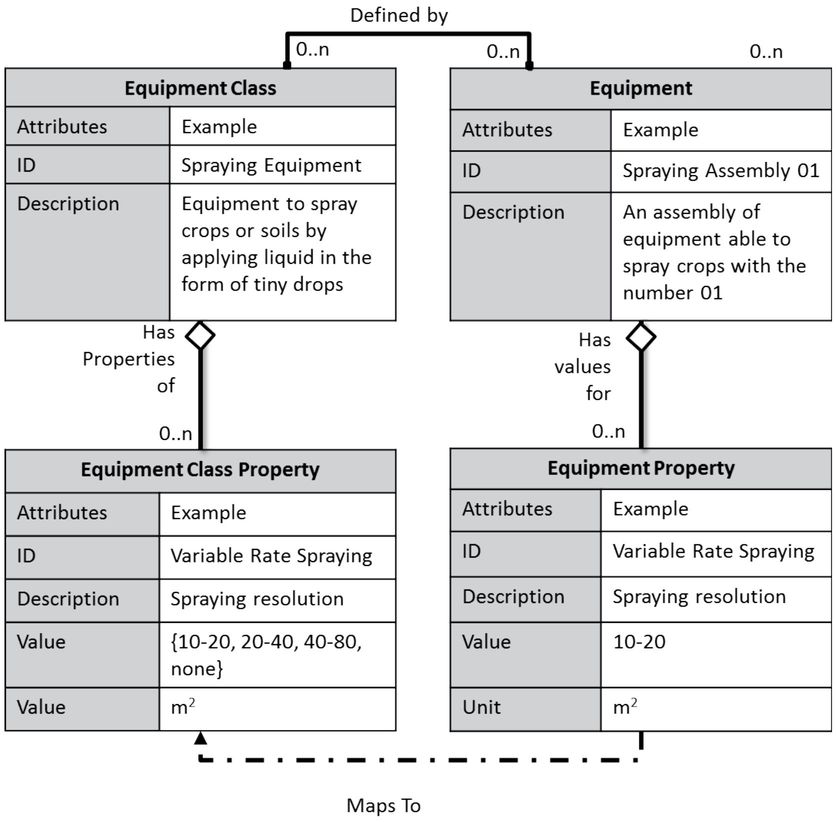

Defined by
0..n
0..n
0..n
Equipment Class
Attributes | Example
ID | Spraying Equipment
Description | Equipment to spray crops or soils by applying liquid in the form of tiny drops
Equipment
Attributes | Example
ID | Spraying Assembly 01
Description | An assembly of equipment able to spray crops with the number 01
Has Properties of
0..n
Has values for
0..n
Equipment Class Property
Attributes | Example
ID | Variable Rate Spraying
Description | Spraying resolution
Value | {10-20, 20-40, 40-80, none}
Value | m2
Equipment Property
Attributes | Example
ID | Variable Rate Spraying
Description | Spraying resolution
Value | 10-20
Unit | m2
Maps To

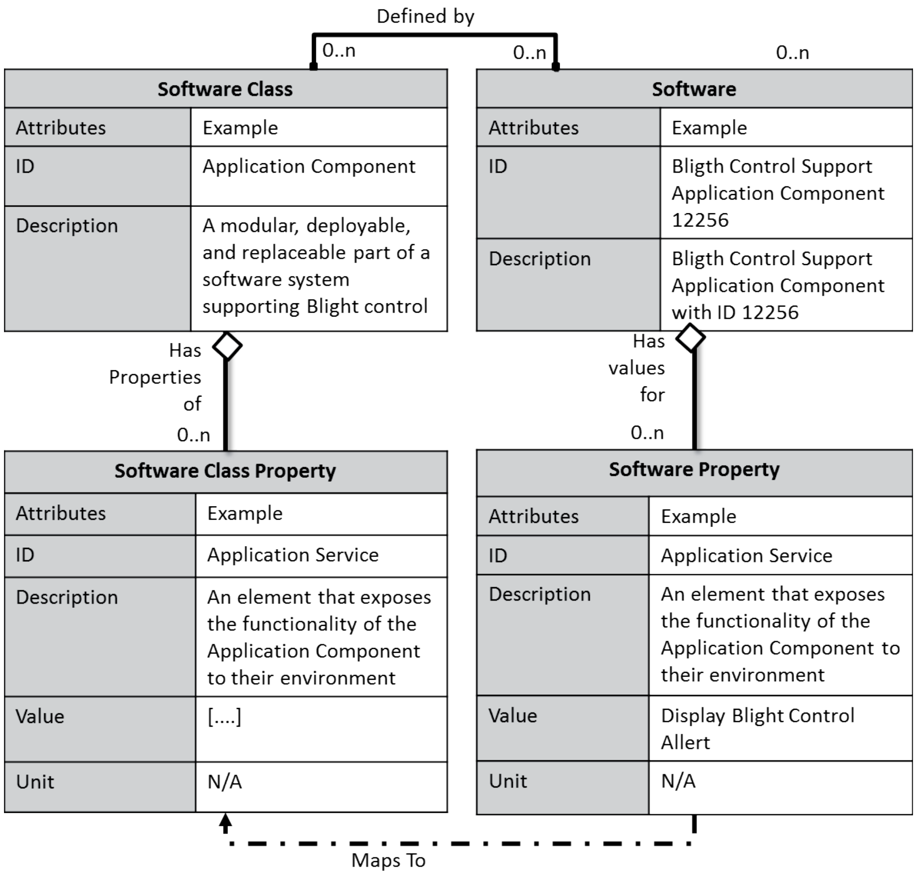
Figure 33: Part of the ISA95 equipment model, with an agricultural example.

Other Equipment Class ID examples are Planting Equipment, Monitoring Equipment, Harvesting Equipment and Irrigation Equipment. Other Equipment Class Property ID examples are capacity (hectare/hour), section control, variable rate planting, variable rate irrigation and yield measurement resolution. In this chapter not all Equipment Class and Equipment Class Property ID's are defined. A more extensive list should be developed to make it complete.

Software reference model

Figure 34 presents the Software Reference Model containing four classes: Software Class, Software Class Property, Software and Software

Property. In the example provided a piece of software (*i.e.* Application Component) at a farm enterprise is identified with the ID Blight Control Support Application Component 12256. This component is defined by the Software Class and has the ID Application Component. Application Components have additional properties describing the kind of Application Services it offers (See Figure 34) and the type of interface is contains. The interface is described by the data structure and the data carrier.



Defined by
0..n
0..n
0..n
Software Class
Attributes | Example
ID | Application Component
Description | A modular, deployable, and replaceable part of a software system supporting Blight control
Software
Attributes | Example
ID | Bligth Control Support Application Component 12256
Description | Bligth Control Support Application Component with ID 12256
Has Properties of
0..n
Has values for
0..n
Software Class Property
Attributes | Example
ID | Application Service
Description | An element that exposes the functionality of the Application Component to their environment
Value | [....]
Unit | N/A
Software Property
Attributes | Example
ID | Application Service
Description | An element that exposes the functionality of the Application Component to their environment
Value | Display Blight Control Allert
Unit | N/A
Maps To

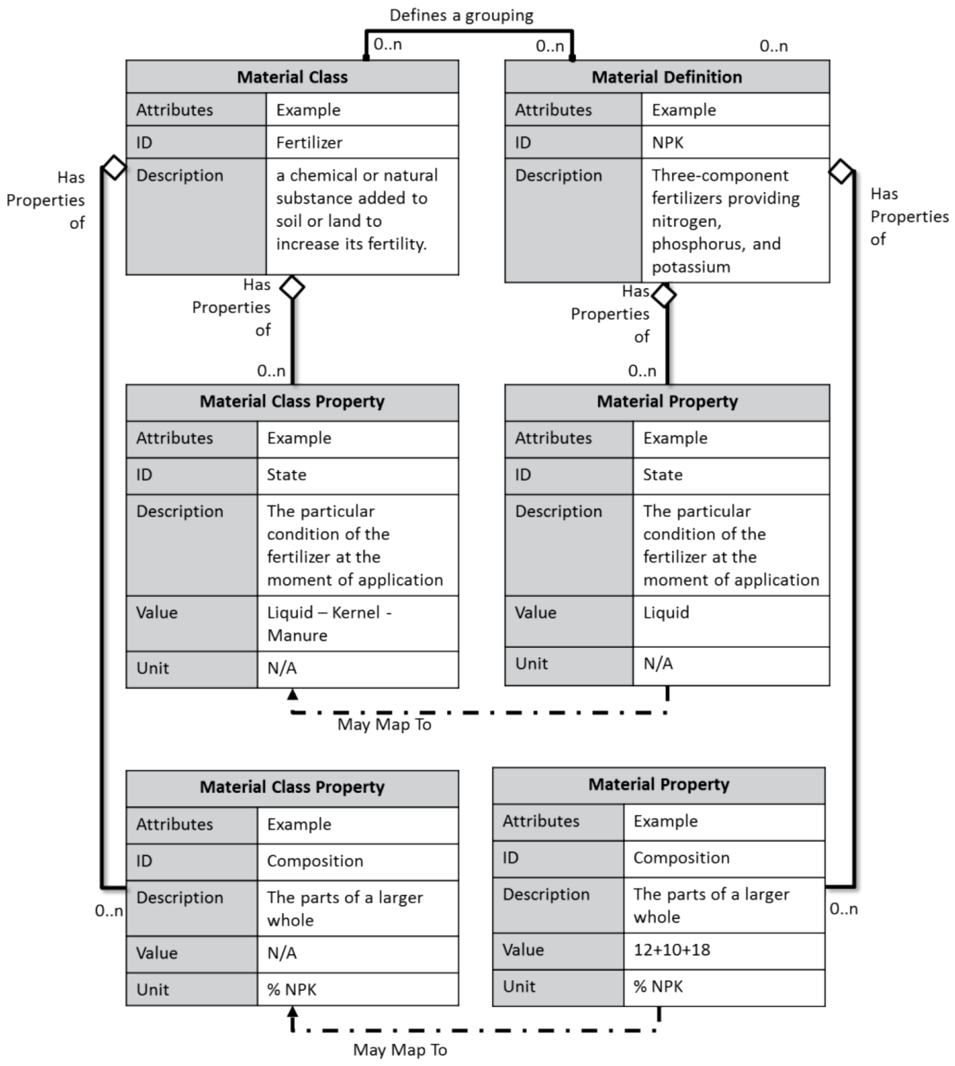
Figure 34: Part of the Software Reference Model of RAAgE 2.0 (based on ISA95).

Examples of values of the Software Property IDs Application Services are: display conditions crop fields, display options and select crop fields, display options and select monitoring data sources, display details of monitoring resources, collect monitoring data. Examples of values for the Software Property ID Data Structure are ISO 11783 Interface or

EDI-Teelt Interface. Examples of the data carriers are EDI, XML and Json. In this chapter not all Software Class and Software Class Property ID’s are defined. A more extensive list should be developed to make it complete.

Materials reference model

Materials are used within certain processes (e.g. pesticides, fertilizers, water and fuel). These resources determine often what kind of Equipment Class ID is needed to be able to execute a business process. Figure 35 presents the Material model based on ISA 95 containing four classes: Material Class, Material Class Property, Material Definition and Material Definition Property. The Material Definition defines the material (e.g. a fertilizer). This definition can be assigned to a Material Lot Class, which defines where the material physically is. The Material is defined by the Material Class with an ID (e.g. fertilizers, fuel, pesticides and water). The Class Property describes additional attributes of the material (e.g. with the ID’s state and composition). Being able to handle differences in the composition of materials used in agriculture is important. Materials can differ in their compositions such as with fertilizers. This can be seen in the example that is presented in Figure 35.



Material Class	
Attributes	Example
ID	Fertilizer
Description	a chemical or natural substance added to soil or land to increase its fertility.

Material Definition	
Attributes	Example
ID	NPK
Description	Three-component fertilizers providing nitrogen, phosphorus, and potassium

Material Class Property	
Attributes	Example
ID	State
Description	The particular condition of the fertilizer at the moment of application
Value	Liquid – Kernel - Manure
Unit	N/A

Material Property	
Attributes	Example
ID	State
Description	The particular condition of the fertilizer at the moment of application
Value	Liquid
Unit	N/A

Material Class Property	
Attributes	Example
ID	Composition
Description	The parts of a larger whole
Value	N/A
Unit	% NPK

Material Property	
Attributes	Example
ID	Composition
Description	The parts of a larger whole
Value	12+10+18
Unit	% NPK

Figure 35: Part of the ISA95 Material Model, with an agricultural example.

4.4.2.3 Process reference model

RAAgE 2.0 focusses on configuring farm specific business process models and the related ICT Components and depicting these with the architectural language Archimate. The farm specific business process models can be categorized in operational control and management control business processes. RAAgE 2.0 focusses on the configuration of operational control business processes. These business processes realize a farm product and require resources (*i.e.* a person to perform a task or software to perform a task). The product that is produced determines the kind of business services that are required (e.g. a planting service, spraying service, harvest service). These services are realized by business processes, which in their turn are realized by the resources. This section describes the process reference model and contains (sets) of business processes able to realize a business service. To provide insight into the Process Reference Model we first describe a hierarchy of operating control business processes and the link with the Product, see Figure 36. This figure presents the hierarchy of the Process Reference Model that links Management Control- and Operating Control processes by the Product Management Plan. This plan defines for one growing season:

(i) Products that will be produced (e.g. grain, sugar beets, potatoes etc.);
(ii) Resources that could be used (e.g. fields, pesticides, fertilizers, employees, equipment etc.);
(iii) Crop services required to produce these products (e.g. till field service, plant crop service, harvest crop service etc.).

The Crop Services are realized by the Operating control business process Cultivate crop fields.

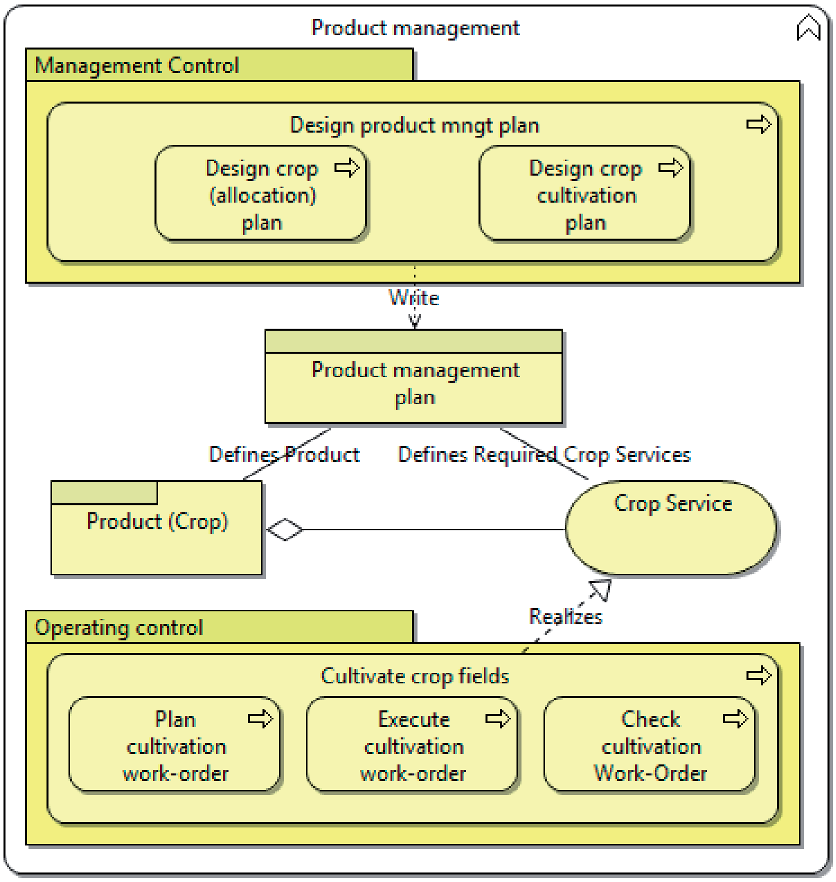


Figure 36: Relation between Processes of the Management Control and Operating Control layer.

This generic cultivate crop field, part of operating control, can be made more specific, see Figure 37. In this figure different Cultivate Crop Field processes are depicted required to produce a product. Examples are Plant Crop Field, Fertilize Crop Field, Protect Crop Field, etc. These Cultivate Crop Field Business Processes all contain a Plan Cultivation Work-Order, Execute Cultivation Work-Order and Check Cultivation Work-Order. These Plan, Execute and Check Cultivation Work-Orders all can contain sub-processes.

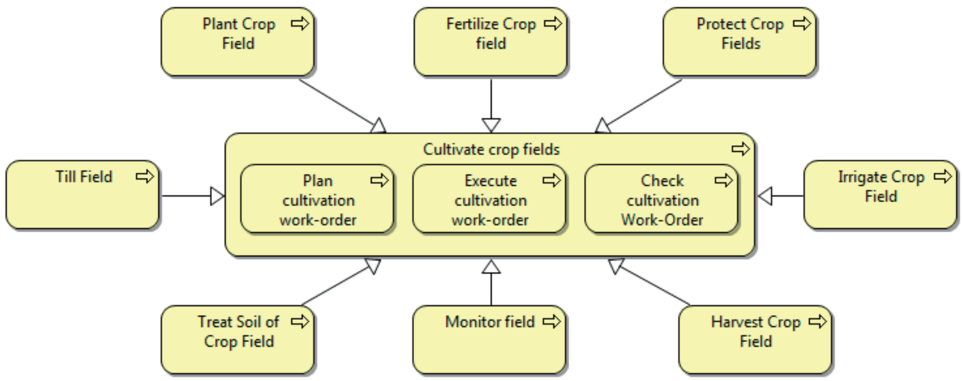


Figure 37: Specialisations of the Cultivate Crop Fields Business Process.

The hierarchy of these sub-processes is shown with the labels Level (L) 1 for the first sub-process and L2 for the sub-sub-process. This hierarchy can be found in Figure 38. In this figure the processes on L1 are depicted and not the L2 processes which are a sub-process of the L1 processes. The farm specific process models can vary between each other especially on theses sub-process levels. These processes can be started by a timing event (according to a planning) or a state event (e.g. errors, crop state, disease alert, etc.), see Figure 38.

Each business process contains a business rule. This business rule determines relations between the Product Definitions, Resource Definitions and Business Processes. These business rules are executed in step 4 and 5 of the configuration tree, which is explained in the following section. These business rules are executed to configure farm-specific business process models, containing a flow of activities that is unique for each individual farm enterprise. A business rule defines the product and resource criteria to assess if the process can become part of the farm specific business process configuration. The structure and a specification of these business rules can be found in the Appendix C. The method to configure process, and the sub-process is described in the configuration tree, see the next section.

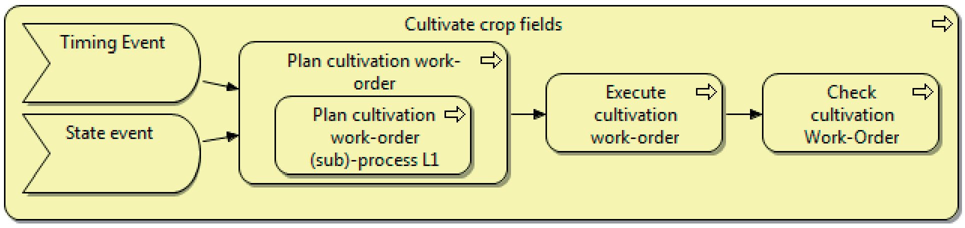

Cultivate crop fields
Timing Event
State event
Plan cultivation work-order
Plan cultivation work-order (sub)-process L1
Execute cultivation work-order
Check cultivation Work-Order

Figure 38: Hierarchy of the Cultivate Crop Field Business Processes.

4.4.2.4 Configuration tree and the business rules

The Configuration Tree provides a method to instantiate specific farm object system descriptions (process configuration) that includes a farm specific process model. Within ISA95 a process configuration is named a Process Segment; which is a collection of Personnel, Equipment Material and Process parameter specifications (Scholten, 2007). The configuration tree to instantiate a farm-specific process configuration comprises the following steps:

(1) Product definition: the user defines the Product he aims to produce using the Product Reference Model.

(2) Business Service selection; the user selects the business services (e.g. planting, fertilization, irrigation etc.) he requires to produce the defined product.

(3) Resource definition: the user defines the available resources to produce selected product using the resource reference model.

(4) Process level 1 configuration: based on business rules taking the product definition and resource definition into account the processes alternatives are presented to the user. Based on the process alternatives the user selects the most appropriate alternative.

(5) Process level 2 configuration: based on the configured level 1 process and the business rules for the sub-processes remaining process alternatives are presented to the user. Based on these process alternatives on a level 2 the user selects the most appropriate alternatives.

(6) The specific farm object system descriptions (process configuration) are presented to the farmer and depicted in the enterprise architectural language ArchiMate. Based on these

object system descriptions the user can adapt its resources to perform a re-configuration (business process re-design) that results in a business process variant.

4.4.2.5 Templates

Templates are specific instances that describe a set of pre-configured business process, a product and the required resources. Currently, RAAgE contains templates to realize the services crop protection, fertilization and planting. For each service, which can be used for multiple farm products, three process variant templates are available which are named Basic, Precision and Smart (e.g. basic spraying, precision spraying and smart spraying). These templates can support users of RAAgE 2.0 (e.g. farmers, business architects) as they can select a template to realize a business service (e.g. crop protection, fertilization, irrigation) with a specific process variant (e.g. basic, precision, smart). To realize such a service, using the processes from the template, the farmer should ensure that the right resources are available. This can mean that a farmer that has the resources available to realize a basic spraying services needs to buy new resources to realize a precision spraying service.

4.5 Evaluation

This section describes the evaluation of the reference information model RAAgE 2.0. First, RAAgE 2.0 is verified by testing and describing the usage of RAAgE 2.0 in a typical case (exemplar). Second, the requirements are verified.

4.5.1 Exemplar for verification: Usage of RAAgE in configuration time

In this exemplar we describe how RAAgE 2.0 can support process and information system configuration. This is a test scenario to show the functionality of RAAgE 2.0. We therefore created a fictional Farm named FarmEx.[53] For FarmEx we describe the business processes of protecting

[53] FarmEx is a fictional farm and its full name is Farm Exemplar.

potatoes against late blight with high precision. In this exemplar we improve the processes of FarmEx by adding additional resources (software) that can make its processes more efficient. This process configuration can be done by a farmer or be supported by a business architect helping farmers to align their business processes and their resources. The resources that require most alignment are the software resources to enable the configuration of an integrated information system. To describe farm business process configuration we use the steps from the configuration tree.

4.5.1.1 The use of the configuration tree

In this exemplar we use the configuration tree to create a farm enterprise architectural description that includes the farm product, business processes, and the related resources. In this exemplar we focus on a process configuration that can protect a potato crop against late blight with high precision. To create this description we followed the steps from the configuration tree:

(1) Section 4.5.1.2: Product Definition of FarmEx using the Product Reference Model.
(2) Section 4.5.1.3: Business service definition of FarmEx to produce the product.
(3) Section 4.5.1.4: Resources Definition of FarmEx resources used for protecting the potatoes against late blight.
(4) Section 4.5.1.5: Configuration of the FarmEx business process on L1 and L2 to protect the potato crop against late blight with high precision.
(5) Section 4.5.1.6: Reconfiguration of the FarmEx business processes based on the object system description by making an additional resource available in the Resource Definition.

4.5.1.2 Product definition

The product FarmEx want to realize is a Potato of the variety Bintje, which will be produced for the conventional market. FarmEx aims to sell the product as seed potatoes. This product is modelled using the product reference model and can be found in Table 10.

Table 10: Farm Product Definition.

Farm Product Reference Attributes	Farm Product Definition
Crop	Potato
Cultivar	Bintje
Specific Market	Conventional
Production Procedure	Seed Potato Procedure
Product State	In field

4.5.1.3 Business service definition

To produce this potato product several business services needs to be realized. From a Template FarmEx can select which business services they need to realize to be able to produce the potato product. The selected business services can be found in Figure 39. In this specific exemplar we will focus on configuring the processes that realize the Protect Crop Service. Specifically, the focus will be on the processes realizing late blight protection with high precision.

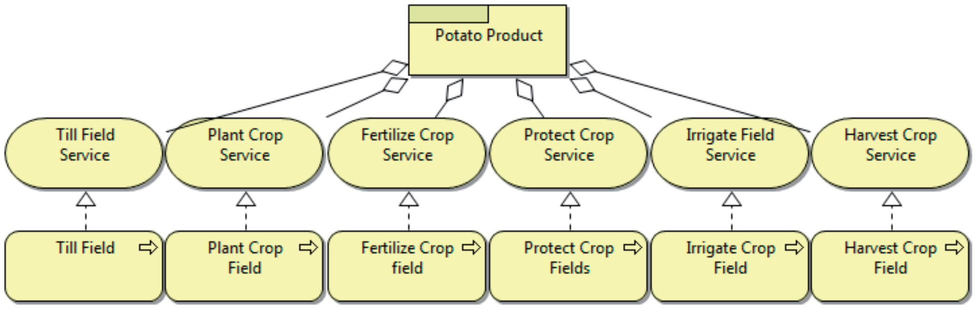


Figure 39: Business Services and Processes to realize a potato product at FarmEx.

4.5.1.4 Resource definition

The resources available for FarmEx that support protecting potatoes against late blight processes are defined in a resource definition based on the resource reference model.

Personnel reference model

Within FarmEx there is one person working (the farm owner named John the Farmer). He can perform tasks at the office, operate equipment and is allowed to operate a sprayer. As an office worker he can prepare and check the spraying work-order. As a general equipment and sprayer operator he can execute the spraying work-order. The values are described in Table 11.

Table 11: Peronal Reference Model atributes selection and values.

Personnel Reference Model attributes	**Person Definition**
Person ID	FarmEx001
Person Name	John the Farmer
Person Operator type	General Equipment Operator
Person Operator type	Office Worker
Person Operator type	Sprayer Operator

Equipment reference model

Within FarmEx a Spraying Assembly is available with a spraying resolution of 10-20 m^2 and section control as the available equipment, see Table 12 This assembly contains multiple interoperable components such as a tractor that includes a terminal that controls the spraying implement. This spraying implement, controlled by the terminal, can spray on a precise level (managing in field variability) and contains section control to reduce spraying overlap.

Table 12: Equipment Reference Model atributes selection and values.

Equipment Reference Model attributes	**Equipment definition**
Equipment ID	Spraying assembly 01
Equipment property ID	Variable rate spraying
Equipment Property value	10-20 m^2

Software reference model

Within the FarmEx example several Software Resources are available, which are a Crop Sensing Component, Job Handling Component, Geographical Information System and the Terminal of the Spraying Assembly. These software resources offer a variety of application services to support the farm business process and have interfaces to share data between them. These application services, interfaces and the application components are presented in Table 13.

Table 13: **Software Reference Model atributes selection and values.**

Software ID	Crop Sensing Component	Job Handling Component	Geographical Information System	Terminal Spraying Assembly 1
Application Service	Receive Crop Conditions Data	Define Work-Order Objective	Display options and Select CropFields	Receive Jobs
Application Service	Display Crop Conditions	Create Job	Display available Implement Assembly instructions relevant for Job	Display execution instructions
Application Service		Edit Job	Receive GIS Data	Execute Job's
Application Service		Receive all Job items and send Job to worker(s)	Create missing Implement Assembly instructions	Send Work-Order data
Application Service		Receive Jobs	Display details of Implement Assembly instruction	
Application Service		Display Jobs in execution	Display and edit Implement Assembly instructions	
Application Service		Receive Work-Order data		
Application Service		Check Work-Order data on completeness		
Application Service		Automatically delete fault data and save		
Application Service		Delete fault data		
Application Service		Save data		
Data Structure	RMcrop	RMcrop	RMcrop	ISO-11783
Data Structure		ISO-11783		
Data carrier	XML	XML	XML	XML

Material reference model

FarmEx has liquid pesticides available as raw materials that can be sprayed as a protection to prevent late blight in the potatoes, see Table 14. These resources enable FarmEx to protect potatoes against late blight with high-precision spraying.

Table 14: Material Reference Model atributes selection and values.

Equipment Reference Model attributes	Equipment definition
Material Class ID	Pesticide
Material ID	Amphore Flex
State	Liquid

4.5.1.5 Farm-specific process model

The FarmEx-specific process model to realize the business service 'late blight protection with high precision' is configured using the configuration tree. The high level processes of FarmEx that realize the service "Protect crop against late blight with high precision" are Plan, Execute and Check late blight control work-order, see Figure 40.

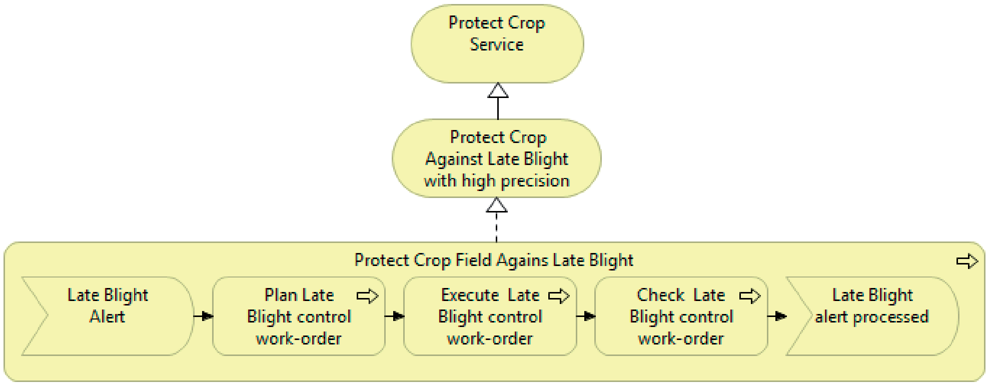

Protect Crop Service
Protect Crop Against Late Blight with high precision
Protect Crop Field Agains Late Blight
Late Blight Alert
Plan Late Blight control work-order
Execute Late Blight control work-order
Check Late Blight control work-order
Late Blight alert processed

Figure 40: High-level business processes to realize the business service Protect crop against late blight with high precision.

Within the Plan Late Blight Control Work-Order Process, see Figure 41 processes on L1 are configured. In this process it is depicted that the farmer receives a late blight alert. This alert is generated based on a regional weather prediction. Based on the alert the farmer starts the workability process. In this process the farmer determines if spraying is needed and possible, see for details Figure 42. Based on the farmers decision the farmer or start designing jobs, or waits for a new alert as spraying was not required, or starts an alternative process. When designing a job, the farmer defines a Work-Order, determines the resources required, and creates a job. This process is supported by the Job Handling Component. After the creation of the job additional instructions for the spraying assembly are created (i.e. variable rate spraying map) within the design or select execution instructions. This process is supported by the job GIS component and Crop sensing component.

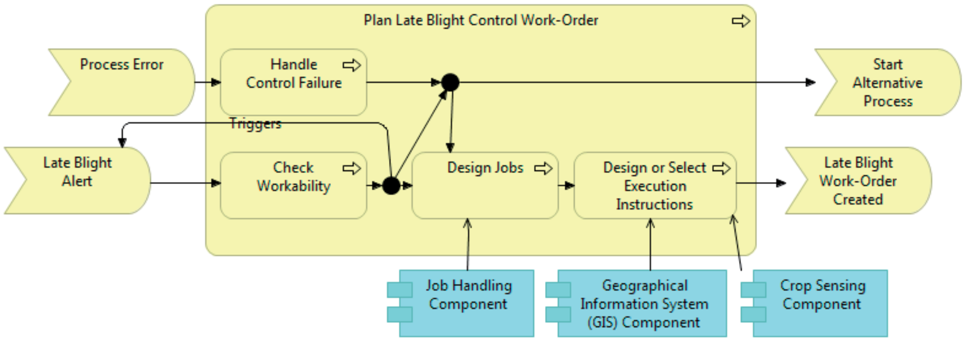

Plan Late Blight Control Work-Order
Process Error
Handle Control Failure
Start Alternative Process
Triggers
Late Blight Alert
Check Workability
Design Jobs
Design or Select Execution Instructions
Late Blight Work-Order Created
Job Handling Component
Geographical Information System (GIS) Component
Crop Sensing Component

Figure 41: Plan Late Blight Control Work-Order using a Crop Sensing, Job Handling and Geographical Information System (GIS) Component. The service each component offers can be found in the resource definition.

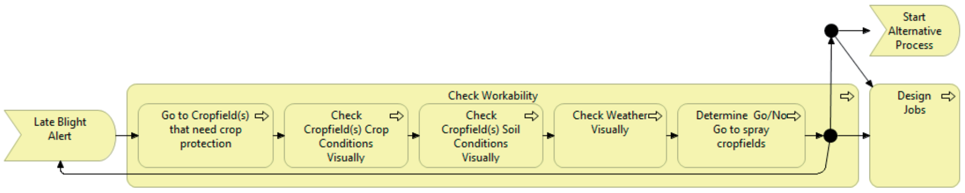

Start Alternative Process
Check Workability
Design Jobs
Late Blight Alert
Go to Cropfield(s) that need crop protection
Check Cropfield(s) Crop Conditions Visually
Check Cropfield(s) Soil Conditions Visually
Check Weather Visually
Determine Go/No Go to spray cropfields

Figure 42: The Check Workability sub-business (L2) process as part of the Plan Late Blight control work-order business process.

The "Execute Late Blight Control Work-Order" executes the task on the field. In this process the farmer receives the work-order that includes the spraying assembly instructions, executes the job by going to the field to spray the potatoes, returns to the farm premises and puts away the sprayer, which is modelled by the perform aftercare process. The receive work-order and execute job with high precision is supported by the terminal of the spraying assembly. The terminal can receive the work-order and instructions and supports the execution of the spraying managing in field variability and using section control to reduce the overlap in spraying. At FarmEx the "Execute Job with high Precision" Process could be monitored using the Job Handling Component. However, these Application Services are not used, as the farmer works alone and no person from the office is checking real-time his activities.

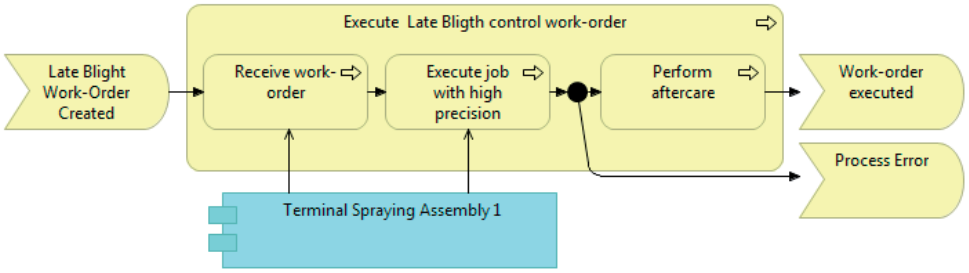

Execute Late Bligth control work-order
Late Blight Work-Order Created
Receive work-order
Execute job with high precision
Perform aftercare
Work-order executed
Process Error
Terminal Spraying Assembly 1

Figure 43: Execute Late Blight Control Work-Order with high Precision.

After the potatoes are sprayed against the late blight disease the tasks, part of the work-order, need to be documented for multiple reasons (e.g. compliance, cost accounting). This process is modelled within the Check Late Blight Control Work-Order (Figure 44). This process is fully automated and the Job Handling Component receives the work-order from the terminal, checks the information on completeness and assess the information and saves it. This Check Late Blight Control Work-Order is the final process and completes this specific "Protect Crop Fields Against Late Blight Basic" process.

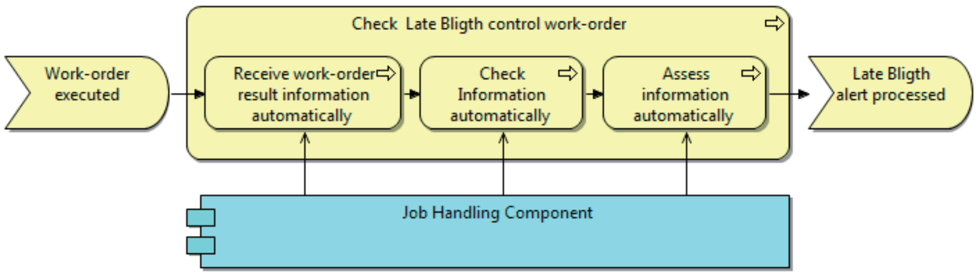

Check Late Bligth control work-order
Work-order executed
Receive work-order result information automatically
Check Information automatically
Assess information automatically
Late Bligth alert processed
Job Handling Component

Figure 44: The Check Late Blight Control Work-Order Processes which are executed automatically making use of the Job Handling Component.

4.5.1.6 Business process re-configuration

The farmer can improve his business processes based on his ambitions. Within FarmEx the aim of the farmer is to reduce his work load and make his farm as less labour-intensive as possible as he is working alone and has to manage several crops. To support this change FarmEx can hire a business architect. The business architect has experience in improving farm business processes and can suggest additional (software) resources that make the business process more efficient. Based on the presented description and other available Templates the Business architect can suggest automating the 'late blight alert' and the 'check-workability' processes. Currently, the farmer needs to check if the potatoes need to be sprayed against late blight and if the soil, crop and weather conditions make spraying feasible by going to the field, which costs time. Therefore, a suggestion is to buy additional software resources that can automate this process. These software resources that are required to automate this process are:

(i) Late Blight Alert component;
(ii) Blight Listener component;
(iii) Weather component;
(iv) Spraying Workability Component.

The Crops Sensing Components can be used to automate the "Check workability" process as well and does not need to be bought additionally as this component can be integrated with the additional components.

These components offer varying services that enable FarmEx to receive a late blight alert when the crop needs spraying (this is done for a region) and to check if spraying for FarmEx specific field crop is possible based on local weather and crop conditions information. With these components the farmer does not need to check his potato fields physically but receives an alert knowing that he should start the creation of a spraying work-order. This process is therefore less labour intensive.

In Figure 45 and Figure 46 the reconfigured processes of FarmEx are shown. These processes show that the farmer receives an alert electronically and that it is automatically detected if a spraying work-order needs to be created. Therefore the farmer is not obliged to check these fields physically. This reconfigured business process is a business process variant and contains a set of automated processes.

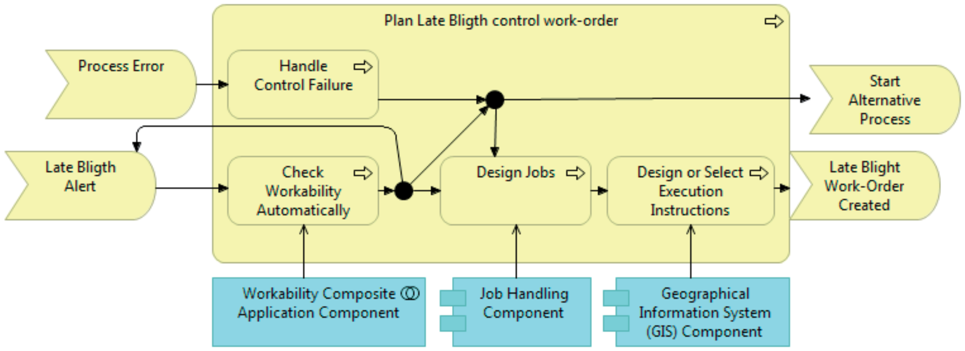

Plan Late Bligth control work-order
Process Error
Handle Control Failure
Start Alternative Process
Late Bligth Alert
Check Workability Automatically
Design Jobs
Design or Select Execution Instructions
Late Blight Work-Order Created
Workability Composite Application Component
Job Handling Component
Geographical Information System (GIS) Component

Figure 45: The FarmEx reconfigured Plan Late Blight Control Work-Order and the supporting Application Components.

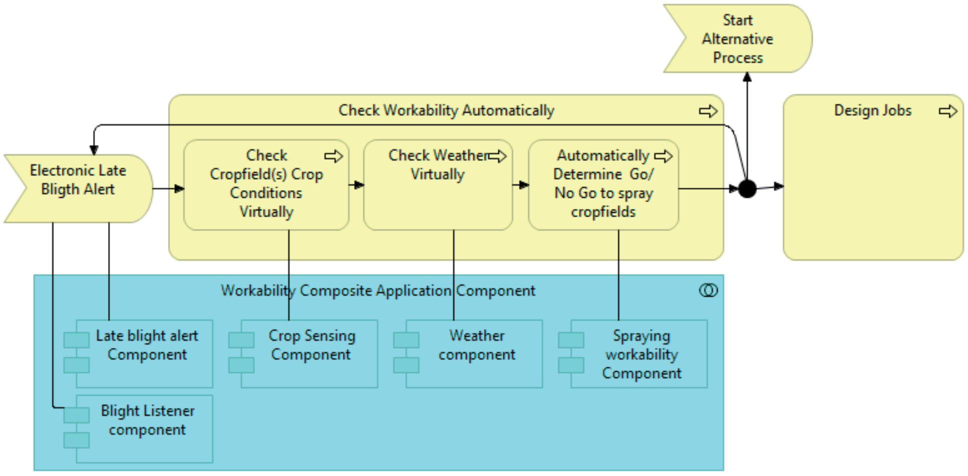

Start Alternative Process
Check Workability Automatically
Design Jobs
Electronic Late Bligth Alert
Check Cropfield(s) Crop Conditions Virtually
Check Weather Virtually
Automatically Determine Go/ No Go to spray cropfields
Workability Composite Application Component
Late blight alert Component
Crop Sensing Component
Weather component
Spraying workability Component
Blight Listener component

Figure 46: The FarmEx automated Check Workability process and the supporting Application Components.

With this exemplar we present how RAAgE 2.0 can be used. This exemplar is based on the FIspace Trial 'Crop Protection and Information Sharing'. The process and ICT Components are realistic and depicted using RAAgE 2.0. With this exemplar we validated that RAAgE 2.0 can depict farm business processes and the required application services and ICT Components. Furthermore, configuration and reconfiguration of farm business processes is demonstrated.

4.5.2 Requirements verification

Verification intends to check that a product, service, or system meets the design specifications. Validation intends to ensure that a product, service, or system meets the operational needs of the user. In this research we have only be able to verify our design as it has not been used in the real world. The verification has been done systematically by the creation of a table. This table describes the requirements and its relation with the design, see Table 15. Based on this requirements verification we conclude that all requirements are addressed with the designed Reference Model RAAgE 2.0. We conclude that RAAgE 2.0 supports the design of a farm object system descriptions by enabling configuration and reconfiguration of figures depicting the farm enterprise architecture. It is expected that these figures can support software developers and business architects to develop software components and configure these into a solution specific for farmers. Validation of the model in a context with real farmers and resources is still required.

Table 15: Requirement verification.

Requirements Category	#	Requirement	Design verification
Representation of the farm object system	1.1	Extend RAAgE 1.0 and improve the ability to represent the farm object system	RAAgE has been extended with a Meta Model (Section 4.4) and additional Reference models to represent the object system (Section 4.4.2).
	1.2	Describe the configuration and the relation of the different resources and business processes	Configuration and the relations are described in the meta model and by describing the individual components (Section 4.4). More insight into the relations between the resources, products and business processes can be found in the business rules (Appendix C Business Rules).
	1.3	Describe the business processes itself	Business process are described (see Section 4.4.2.3)
	1.4	Describe the business processes variants	Business process variants are described the exemplar reconfiguration (Section 4.5.1.6)

Table 15: continued

Requirements Category	#	Requirement	Design verification
	1.5	Describe the products a farm delivers	Products are described with the Farm Product Reference Model (see Section 4.4.2.1)
	1.6	Describe the resources used within a farm	Resources are described with the Farm Resource Model (see Section 4.4.2.2)
Enable configuration and re-configuration	2.1	Support configuration	Configuration is supported and demonstrated with an Exemplar (see Section 4.5.1)
	2.2	Describe how configuration is supported	Configuration is supported with a Configuration Tree (see Section 4.4.2.4). The exemplar provides more information into configuration (see Section 4.5.1).
	2.3	Distinguish different time phases (Design-Time, Configuration-Time, Run-Time)	Different time phases are distinguished within the Meta Model (See Section 4.4).
Ensure reuse of ICT components	3.1	Include templates with combinations of products, processes and supporting ICT components	Templates are presented, see Section 4.4.2.5. Furthermore, an exemplar is presented which can be seen as an template as well (see Section 4.5.1)
	3.2	Provide insight into the differences between specific and generic processes	Specific and generic process can be found in the Process Reference Model. The details to select a specific process are presented in the configuration tree (Appendix C Business Rules).
Ensure collaboration of multiple vendors to enable best-of-breed solutions	4.1	Contribute to Best-of-Breed solutions	RAAgE 2.0 can support software development within Farm Software Ecosystems as the software reference model describes the application services. The Application Services can be related to ICT components of multiple vendors. RAAgE 2.0 can support these vendors to create ICT components as it describes the business processes of different farms.

4.6 Discussion and conclusions

This chapter presents a reference model to enable farm business process configuration. The reference model provides insight into farm business processes and how a particular supporting software product (farm information system) is specified. This specification is provided by depicting the enterprise architecture of a particular farm. To configure enterprise architectural models the reference model comprises three architectural views, i.e. product, process and resource reference models. The dependencies between these views are defined in rules that define the possible combinations of the products, processes and resources and that constrain the configuration of farm-specific models i.e. instances. The reference model also includes a configuration tree and templates, which support the configuration of farm-specific models i.e. enterprise architecture instances. A configuration tree provides a method to instantiate specific farm enterprise architectural descriptions. It is a wizard-like step-by-step plan that guides users through the configuration process, taking into account the constraints as defined in the rules of the reference model. Templates describe a set of pre-configured product, process and resource models for typical cases.

The main value of the reference model is that it helps to dynamically configure customized farm-specific information models in a timely, punctual and coherent way. As such it helps to deal with the high diversity and variability of farm processes while at the same time knowledge is optimally reused. With this it can enable the development of software product lines within farm software ecosystems, resulting in farm information systems that are based on a Best-of-Breed and ICT Mass Customisation approach.

More specifically, the reference model of the present chapter can (1) help farmers in creating business process configurations that reflect their practices and help them to improve their business processes (business process re-configuration), (2) enhance modular based software development within Farm Software Ecosystems by describing the interfaces of application components and the a method to enable (re)configuration of farm information systems and (3) support farm enterprises in the selection and configuration of the resources that are needed to execute their (improved) business processes.

The designed reference model is verified based on the requirements and usage in an exemplar. However, further steps are required. First, RAAgE 2.0 should be validated with other use cases and involvement of software developers and business architect. Second, the model should be extended with process descriptions of other use cases and inclusion of more business rules. Third, future research is needed for the implementation of the designed reference model in a software product line. Currently, the model provides insight how the particular software product is specified. However, the specification of the software product line itself is still to be described and implemented.

In the next chapter we describe a proof of concept showing how RAAgE 2.0 can support modular based software development within Farm Software Ecosystems and enable configuration of different components into a composite application component. This proof of concept will be based on the FIspace Platform.

5

Configuring a farm information system from multi-vendor apps

A case of late blight protection

This chapter will be submitted to an international journal:

Kruize, J.W., Goense, D.,Wolfert, J., Scholten, H., & Beulens, A.J.M.

Abstract

Farm enterprises need to increase yields while using fewer resources due to the growing world population and concerns regarding sustainability of our food production. Simultaneously, farms become bigger and legislation to assure food safety increases resulting in a high administrative burden. All these developments make farm management more complex and require tight monitoring and control of crops over the lifecycle. This challenge is being addressed by all kinds of ICT Components from multiple vendors to support registration, planning, execution, monitoring and control processes. Cost-effective integration and configuration of these ICT Components customized for business processes of individual farmers is required. Currently integration is cumbersome and hampering adoption of more advanced farm management styles. Solving this problem requires advanced farm business processes that are supported by farm-specific and integrated farm information systems that are composed of ICT Components from multiple vendors. To realize such information systems two approaches should be combined, namely ICT Mass Customisation and Best-of-Breed, allowing farmers to rapidly configure customized software systems from the best, standardized ICT Components that supports their specific business processes. Recent research has delivered various models and platforms that unravel the complexity of the combination of these approaches. However, a proof of concept including prototype software that shows how this approach can be realized has not been provided. This chapter presents a prototype application that connects six advanced ICT Components from multiple vendors to support the late blight protection process in potato growing. The configuration of this application was supported by a Reference Architecture for Agricultural Enterprises (RAAgE) that is based on ICT Mass Customization. The development was embedded in an initial Farm Software Ecosystem that can be extended in the future, supporting many other business processes by development of similar applications in the same way.

Keywords: Best of Breed approach, ICT Mass Customisation, farm enterprise integration, software ecosystem

5.1 Introduction

Due to the growing world population and concerns regarding the sustainability of our food production farm enterprises[54] need to increase yields while using fewer resources. Simultaneously, legislation increases to assure food safety. To secure advancement on these themes all kinds of ICT Components, such as software for registrations, geographical information systems, sensors, terminals, decision support systems and other devices or application components, are developed. However, utilization of these ICT Components by farmers is severely hindered due to a lack of interoperability between the ICT Components of multiple vendors (Pierce et al., 1999). In many cases, farmers would like to use the best functionalities of different ICT Components, but interoperability problems hamper this best of breed approach. The systematic analysis of Kruize et al. (2013) showed that current ICT Components that are used within the same farm enterprise (i) have partly overlapping and partly unique services, functions and interfaces, (ii) are missing required application services, functions and interfaces, (iii) have separate data repositories and (iv) have inadequate and incomplete data exchange. This means that farmers would have to spend a lot of money on buying several software packages that cover their needs and on top of that would have to hire a technician to combine all these packages in an effective manner. In conclusion, most of the available ICT Components are lacking both technical and semantic interoperability, resulting in data sharing issues and non-coherent user interfaces (Kruize et al., 2013).

To utilize the merits that technology can offer in farming, interoperable ICT Components of multiple vendors are required. These ICT Components should offer a broad range of functionalities that support farm specific business processes (Kruize et al., 2016). A promising approach that enables integration of ICT Components is referred to as ICT Mass Customisation. ICT mass customisation combines advantages of standardized and customised software by enabling on-demand configuration of information systems from standard components with standardised interfaces (Verdouw et al., 2010a). To enable ICT Mass

[54] A farm enterprise can be an arable farm, livestock farm or horticultural farm. In our research we focus on arable farm enterprises.

Customisation five requirements have to be fulfilled which are (Verdouw et al., 2010): (i) availability of a generic information model, (ii) modular software, (iii) information integration platform, (iv) configuration support and (v) component availability. In the ideal situation, these components could be supplied by different software vendors to enable Best-of-Breed solutions. It is expected that such an approach will solve the aforementioned problem situation for farmers because (i) integration of ICT Components becomes easy and affordable as this is done with a standardized method, (ii) the ICT Components can be configured to address farm-specific needs and (iii) less atomic application components are needed as the components of a single farm can be re-used in other configurations. This requires extensive interaction and alignment between various actors that can be situated at different geographical locations all over the world. To that end the concept of Software Ecosystems was defined, which consists of collaborating actors governed by an open software enterprise, models, a common platform including documentation to enable software development and interoperable ICT Components of multiple vendors (Jansen et al., 2012).

In previous research, the authors have developed a technological and organizational framework that enables farm enterprise integration based on ICT mass customization using a Best-of-Breed approach. In Kruize et al. (2016) a software ecosystem for integrating ICT Components for farming was defined to address organizational challenges. Then a reference architecture for agricultural enterprises (RAAgE 2.0) was developed using reference information models to enable farm-specific configurations of ICT Components (Kruize et al., Forthcoming). With RAAgE it is possible to analyse current business processes at a farm to systematically identify the ICT integration problems, especially in case a farmer wants to apply a more advanced farm management style such as precision agriculture (Kruize et al., Forthcoming). RAAgE also supports the configuration of ICT Components into an integrated farm management information system that enables a farmer to apply a Best-of-Breed approach and that handles interoperability problems.

However, a full implementation of this approach to prove that it is working is still missing. The objective of this chapter is to create a proof of concept that shows how ICT Mass Customisation in combination with a best of breed approach can be realized for arable farming within a farm software ecosystem. It will show how multiple ICT Components, supplied by different vendors in Europe, can be configured to support late blight protection in potatoes in a more advanced manner. The development is embedded in an initial Farm Software Ecosystem that can be extended in the future, supporting many other farm business processes by development of similar applications in the same way.

5.2 Materials and methods

This research was closely connected to a trial on ‘crop protection information sharing’ within the FIspace project.[55] Therefore the implementation in this chapter is largely based on that trial, using the FIspace platform developed in that project. The case of late blight protection in potatoes is used as a representative example for crop protection. As materials, we will first introduce a case study of a Dutch arable farmer embedded in a farm software ecosystem framework followed by a brief description of the RAAgE framework including an ontology that will be used. In the method section we will describe how these materials were used to develop an integrated farm management information system based on the principles of ICT mass customization and a Best-of-Breed approach.

5.2.1 Materials

5.2.1.1 Case study description: advanced late blight protection in potatoes

Late blight is a disease in potatoes caused by the oomycete *Phytophthora infestans*. This disease infects both leaves and tubers and is well-known from the Irish potato famine in 1845. Control of late blight requires regular spraying with fungicides. Farmers can basically follow two strategies. The first strategy is preventive by spraying in short intervals. The second strategy is a more rational one in which spraying is based on actual crop status and (projected) environmental conditions. The second strategy is preferred by most farmers because of lower costs

[55] www.FIspace.eu

and environmental effects. In the second strategy parameters to determine risk for development of late blight are deducted from advisory systems and the spraying scheme is adapted accordingly. In this spraying scheme there are three spraying options; act preventive, act curative or act eradicative. The first option should prevent late blight to occur in the potatoes. Curative spraying should stop the infection. Eradicative spraying is done when late blight occurs in the potato crop and should be reduced. Preventive spraying is preferred due to lower cost of the crop protection product. Eradicative is the least preferred option as these crop protection products are expensive and yields can be affected by the late blight infection. Current advisory systems for late blight in potatoes require information from the potato crop such as variety and growth stage and in particular the fungicide applications already applied before. In the Netherlands, late blight advisory systems are integrated in Farm Management Information Systems (FMISs). A vendor lock-in can be identified because switching between advisory systems would require the conversion of all management data into a new FMIS. Moreover, the weather information service that is needed for the advisory system is usually also determined by the FMIS provider while a farmer sometimes already has his own weather information provider or he would like to use this information also for other purposes. In general, this lock-in is hampering the farmer to buy other ICT Components for other purposes (e.g. scheduling) because his information system lacks an appropriate interface to communicate with.

In this research a farmer in the north of the Netherlands, hereafter called 'Farmer A' acts as a model for our case study. Farmer A grows potatoes and uses a standard FMIS including an advisory system for late blight protection. He has indicated that he would like to move towards a more advanced advisory system using a Best-of-Breed solution so that he is able to buy the right services and components for his purpose. At a later stage he wants to be able to move towards 'precision spraying' in which the crop is sprayed more precise in place and time. This means that (part of) the components should be replaced by more advanced ones. Figure 47 summarizes this use case in terms of an 'as-is' and 'to-be' scenario at a high abstraction level. At the left, three business processes, which e.g. represent the process of late blight protection, are supported by a standard FMIS that comprises three functions (e.g. get field data, get weather information and give spraying advice). The FMIS is provided by Vendor X, including all functions that are determined by this vendor. At the right, the desired to-be scenario is depicted for a more advanced farm business process. In this scenario the same standard FMIS is still used but only one function (e.g. get field data) of it is used. The other functions are delivered by other, more advanced components provided by different Vendors Y and Z. Therefore the sub-processes B en C have a green colour indicating that these processes have become more advanced. The green block around all components and the connecting lines imply that these components need to interact with each other through well-defined interfaces.

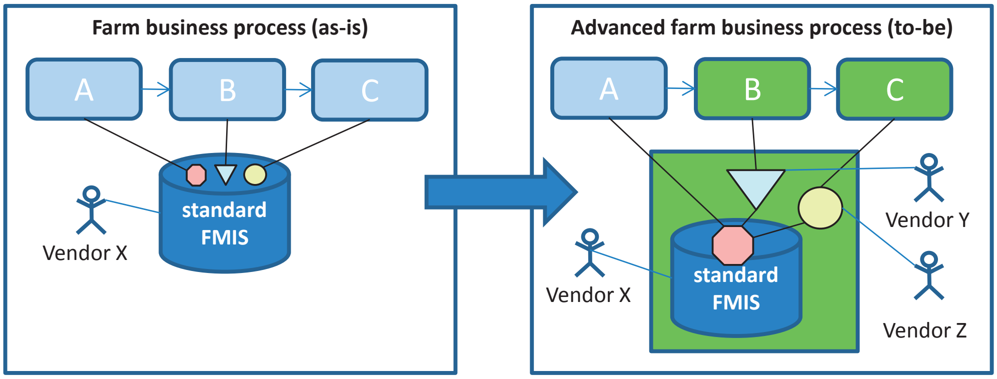

Farm business process (as-is)
A
B
C
standard FMIS
Vendor X
Advanced farm business process (to-be)
A
B
C
standard FMIS
Vendor X
Vendor Y
Vendor Z

Figure 47: The use case as an as-is and to-be scenario of the farm business process. Further explanation in text.

5.2.1.2 Farm Software Ecosystem reference architecture

For Farm Software Ecosystems a reference architecture was developed to improve communication and collaboration between multiple actors that are part of real-world Farm Software Ecosystems (Kruize et al., 2016). It helps to understand Software Ecosystems in support of joining, forming or improving Farm Software Ecosystems that enable the development of integrated farm information systems. Figure 48 provides a schematic view of the Farm Software Ecosystem reference architecture describing the relation between a Platform, Actors, ICT Components, Business Services and an Open Software Enterprise. An ideal farm software ecosystem would consist of many actors, components and services, but within the scope of this research this was not feasible to realize. Instead, we will demonstrate a very small software ecosystem for Farmer A, a few vendors and ICT Components and only one software instantiation for this single farmer that is able to support late blight protection business processes. Nevertheless it is expected that the basic principles for real software ecosystems remain applicable.

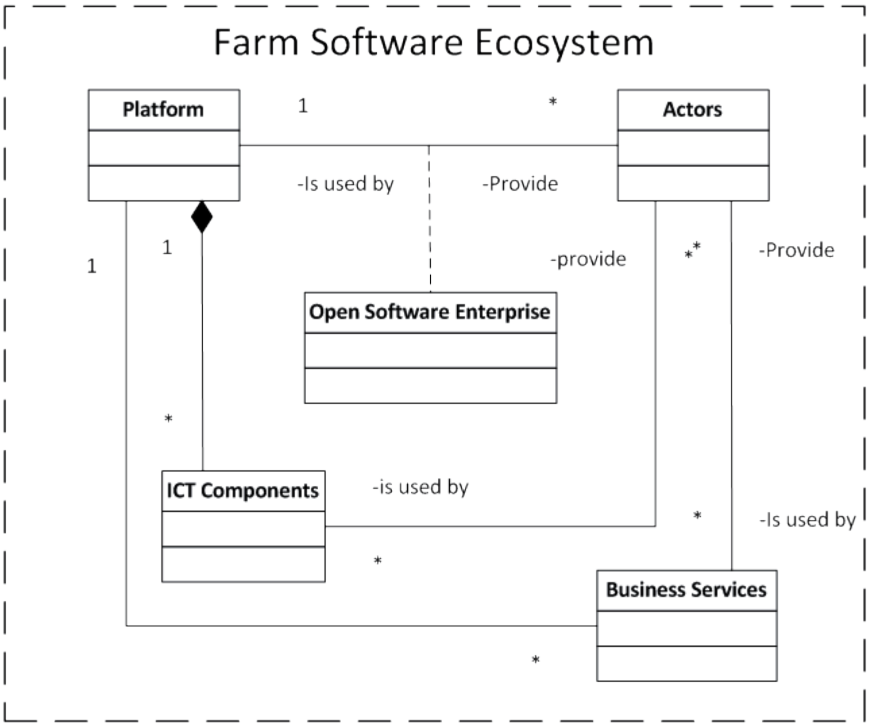


Figure 48: High-level view of the Farm Software Ecosystem Reference Architecture (Kruize et al., 2016).

We will use the FIspace platform to enable configuration of different components from different vendors (cf. the green block in Figure 47). FIspace consists of different integrated modules:

- **User Front-End:** serves as the main point of access for users of the platform services and Apps, and constitutes a configurable graphical user interface.
- **B2B Collaboration Core:** ensures that all information and status updates are provided to each involved stakeholder in real-time. The B2B core allows for the creation, management, execution, and monitoring of collaborative workflows (business processes) in the FIspace platform.
- **FIspace App Store:** provides the tool-supported infrastructure for providing, finding, and purchasing atomic application

components (FIspace Apps), which provide re-usable IT-solutions supporting business collaboration scenarios and which can be used and combined for the individual needs of users.

- **System and Data Integration:** allows for the integration of existing legacy and business systems as well as the integration of external systems and services. It includes facilities for data mediation. Data can be offered to the platform based on the capabilities concept.
- **Security, Privacy and Trust:** provides secure and reliable access and, where needed, exchange of confidential business information and transactions using secure authentication and authorization methods that meet required levels of security assurance. Authentication, authorization and accounting technologies will provide user management and access control features.
- **Operating Environment:** ensures the technical interoperability and communication of (possibly distributed) FIspace components and FIspace Apps and the consistent behaviour of FIspace as a whole. Its main feature is the Cloud Service Bus (CSB) providing event bus and pub/sub capabilities.
- **Software Development Toolkit (SDK):** provides tool-support for the development of atomic application components. The SDK will ease the work of App developers during the implementation of the Apps, providing specific tools and libraries that hide the more complex aspects of the platform.

The FIspace platform distinguishes three particular roles for Actors within a Farm Software Ecosystem (Verdouw et al., 2014):

(1) **App developer:** the actual software and system providers who offer “packaged” / componentized solutions and applications in form of Apps. These atomic application components offer capabilities to the FIspace platform to enable configuration. By configuration atomic application components are clustered into a composite application component. This configuration can be done by a business architect.

(2) **Business Architect:** an expert (internal or external to the end-user organization) that is in charge of configuring

FIspace for their individual business needs. Particularly they define business process based on available resources and individual needs of the end-users. This business process defines requirements regarding the capabilities which are provided by atomic application components (Apps). The business architect selects the appropriate apps and can configure these into a composite application component to support the process.

(3) **End User**: the actual user (aka. supply chain actors such as farmers) of the composite application component provided by FIspace. The end users will be supported in their daily business activities, with special focus on their interaction and collaboration with business partners.

More details regarding the FIspace platform can be found on the website[56] or in various papers (Barmpounakis et al., 2015; Kaloxylos et al., 2012; Kaloxylos et al., 2014; Kruize et al., 2014; Verdouw et al., 2014).

The Open Software Enterprise within a Farm Software Ecosystem should enable the development of interoperable Application Components, the configuration process and the operation of the configured Application Components in run-time. For this run-time environment a technological (cloud) infrastructure should be available to host the platform and which is able to connect to all Application Components. Furthermore, it should provide a revenue and cost sharing model as software developers, infrastructure providers and configuration service providers are using each other's components and services. To facilitate this, basic support for e.g. contracts, payments, etc. is required. Furthermore, when possible disputes arise a governance structure should provide a resolving mechanism. However, because we are not developing a complete ecosystem we do not further take the role of such an Open Software Enterprise into account in this research.

[56] www.FIspace.eu

5.2.1.3 A reference architecture for business process configuration and the related application services (RAAgE 2.0)

To configure a farm-specific application for late blight protection for Farmer A that is composed by different components from different vendors the reference architecture for agricultural enterprises (RAAgE 2.0) will be used (Kruize et al., Forthcoming). Figure 30 provides a schematic overview of the RAAgE 2.0 framework that is used for this case study. It distinguishes three time phases: design-, configuration- and run-time. In design-time, several reference models are used to model the relevant business processes and referring products and resources that are involved. This process is supported by a configuration tree and accelerated by templates of combinations of processes, products and resources that are often used. In configuration time, the model is used to define farm-specific process models including the resources that should support the process. RAAgE 2.0 focusses more specifically on the configuration of application components to enable configuration of ICT Components. In FIspace, configuration is typically the task of the business architect supported by the Cloud Service Bus in the Operating Environment of the platform. Finally in run-time, the farm-specific model is instantiated into an executable process running within the platform in which the interaction between different sub-components is clearly handled. In this research we focus especially on the configuration time phase and develop a prototype application to show that real software is derived from this configuration process.

The configuration is based on a hierarchical configuration methodology which facilitates specification of the reference model and the configuration of (atomic) application components into specific software instances. This hierarchy is divided in two parts (i) business process configuration and (ii) software configuration.

Business process configuration consists of the following steps. In the first step business services (e.g. crop protection, fertilization, planting, harvest) are selected that are required to produce a specific product (e.g. potatoes, grain, sugar beets) at a specific farm. These business services identify the support during the crop cycle and are realized by business processes. Both the business services and business processes are generic at a high level as most farmers require the same business service and similar processes to produce a crop. In the second step, the business sub-processes are configured. These business sub-processes describe the activities of farmer in more detail and can differ between farms. Different configurations of business processes that realize the same business service are named variants (e.g. basic fertilization, precision fertilization). The business process variant depends on the available resources at the farm (e.g. machines, ICT Components, other equipment). Unavailability of certain resources obstructs realization of some business process variants. To change a business process from a current business process variant (e.g. basic fertilization) to a more advanced business process variant (e.g. precision fertilization) can require additional resources. The outcome of this business process configuration is a description of the farm specific business processes, its sub-processes and the required resources (including ICT Components) to produce a specific crop. More details regarding business process configuration can be found in the previous chapter.

The second part of the hierarchical configuration methodology is the software configuration that uses the description of the configured business processes and related resources as an input. Based on this input a composite application component can be configured. This prototype software will contain atomic application components of multiple vendors to support late blight protection. This software will be a proof of concept that has generic applicability as it will follow the hierarchical configuration methodology which specifies configuration of a specific solution but can be used to develop other solutions as well.

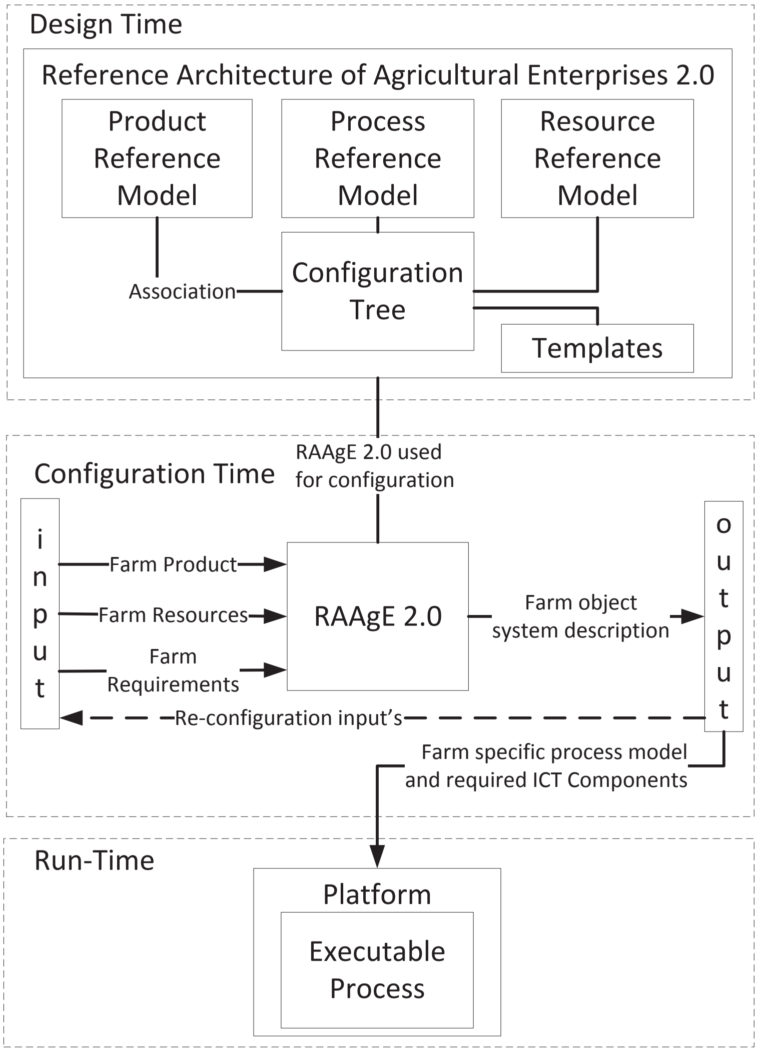


Figure 49: The RAAgE 2.0 framework (Kruize et al., Forthcoming).

5.2.1.4 Ontologies and standards used

To support the interaction that is needed between various actors that are involved in the Farm Software Ecosystem an ontology was used that can be found in the Appendix A (Kruize et al., 2016). An ontology is a concise and precise, formal specification, shared by a group of persons and providing sufficient vocabulary such that a piece of knowledge can be formalized for its purpose, is understandable for its human users and manageable for its machine users (computers) (Scholten, 2008). The ontology in this research focusses on the relation between business processes, the supporting software system and its development and configuration.

Data reference models are used to enable the definition of standardized object description (virtualizations) and messages that can exchange data between multiple components. The data standards used in this research are the drmCrop data model and ISO11783 Part 10. The drmCrop data model is part of the reference model rmCrop for crop production (ftp://pragmaas.com/rmCrop) and is developed during several research projects in the Netherlands. It is based on "Informatie Model Open Teelten" (Anonymous, 1987) and ISO11783 Part 10. The drmCrop data model is a platform-independent UML class model which can be transformed to an XML model, which includes transforming datatypes into XML-specific datatypes. From the XML model, XML schemas can be generated that are used to define standardized messages for data exchange. ISO11783 Part 10[57] is the international standard for data exchange on tractors and farm implements, including data exchange with farm management information systems (FMISs). The drmCrop data model is aligned with the ISO11783 model and they are complementary to each other because both focus on another object system (farm enterprise vs. farm equipment).

5.2.2 Methods

5.2.2.1 Design Oriented Research using a case study approach

This research uses Design Oriented Research (DOR) in a case study approach. The purpose of DOR is to create innovative artefacts that extend the boundaries of human and organizational capabilities (Hevner et al., 2004). These artefacts are developed based on a design process. In Information System (IS) Research these artefacts are constructs/concepts, methods, models, and instantiations (Hevner et al., 2004; March and Smith, 1995). To develop IS artefacts, guidelines of an Information System Research Framework can be followed (Hevner et al., 2004). In this Information System Research Framework three concepts

[57] http://www.iso.org/

are distinguished; the Environment, Information System Research and Knowledge Base (Hevner, 2007; Hevner et al., 2004). The environment provides the context in which the artefact is used. The artefact developed in this research is an instantiation that provides a proof of concept. A proof of concept is defined as a phase in development in which experimental hardware or software is constructed and tested to explore and demonstrate the feasibility of a new concept (United et al., 1969). The developed software is composed of ICT Components provided by multiple vendors and supports late blight protection in potatoes. This case study is used to validate the design. A case study method refers to a research strategy which focuses intensively on individual cases to draw insights about causal relationships in a broader population of cases (Poteete et al., 2010; Yin, 2009). Therefore, "A case study may be understood as the intensive study of a single case where the purpose of that study is, at least in part, to shed light on a larger class of cases (a population)" (Gerring, 2006). This specific case, late blight protection, is representative for other cases in agriculture as late blight protection requires integration of ICT Component of multiple vendors. The developed prototype software is validated using data from Farmer A who is located in the North of the Netherlands.

5.2.2.2 Approach

To arrive at the proof of concept in this research the following steps will be taken:

(1) Configure the business processes using RAAgE 2.0 that are involved in late blight protection to identify which advanced ICT Components are needed to support this process for Farmer A.
(2) Develop the required advanced ICT Components that were identified in the previous step using the FIspace platform.
(3) Configure a composite application component within the FIspace platform using the configuration framework of RAAgE 2.0.
(4) Instantiate and run the application component within the FIspace platform for Farmer A.

The next section will present the results of these steps.

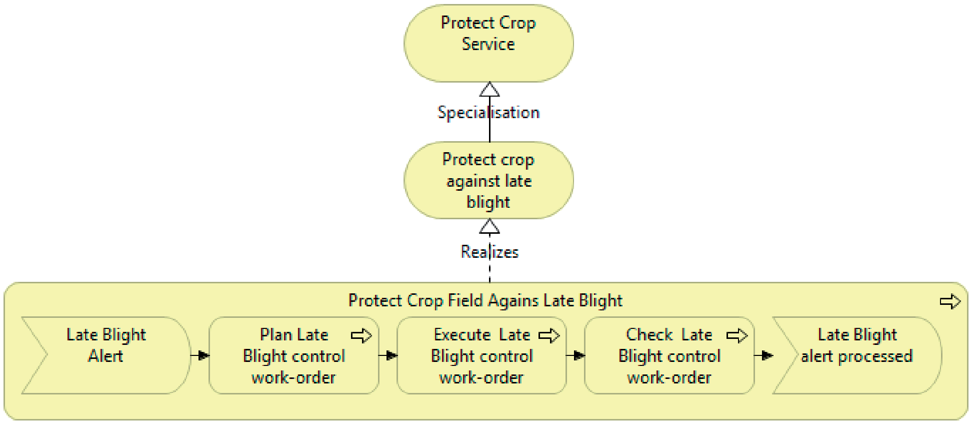
5.3 Results

5.3.1 Requirements definition by analysis of the business processes

To support the whole process of late blight control, business process expertise in different disciplines is required such as in management software, meteorology, phytopathology, operations research, agricultural engineering, real time software development and more. This implies that the following functions are required:

(i) virtualization of the actual field and crop conditions;
(ii) information of already applied protective measures;
(iii) a good weather prediction;
(iv) an advise of when to take measures based on the probability of late blight development,
(v) planning when the measures have to be performed in respect of multiple fields to treat and other fam operations to be performed in the same time period;
(vi) machines and plant protection products to realize crop protection
(vii) correct registration of plant protection products with reporting of actually applied products;
(viii) monitoring and control of the whole process.

To describe late blight control processes in detail, RAAgE 2.0 contains templates that depict late blight control processes. Based on these templates a late blight control business process variant is configured that describes the more advanced processes Farmer A requires. Based on these processes, atomic application components are defined that can support these business processes. To describe the business processes and atomic application components, the high level business processes are presented in Figure 50. These business processes describe how a 'protect crop against late blight' business service is realized. This business service is a specialisation of the 'protect crop' business service. The processes to realize a 'protect crop against late blight' business service are (i) plan late blight control work-order, (ii) execute late blight control work-order and (iii) check late blight control work-order. A work-order specifies what resources are used within a specified time for a specific crop to achieve a certain goal. In this case we focus on the crop potatoes. The start of the business processes is triggered by one or more events. In this case such an event is a late blight alert triggering the sub-processes part the of ‘plan late bight control work-order’ business process.



Protect Crop Service
Specialisation
Protect crop against late blight
Realizes
Protect Crop Field Agains Late Blight
Late Blight Alert
Plan Late Blight control work-order
Execute Late Blight control work-order
Check Late Blight control work-order
Late Blight alert processed

Figure 50: Business Process realising late blight protection in potatoes.

The sub-processes of this ‘plan late blight control work-order’ and the required application components are depicted in Figure 51. The blight alert can be generated by a late blight advice atomic application component. This component can monitor the probability of a late blight outbreak within a certain time window. To determine the change of an outbreak weather data should be provided by another atomic application component named ‘weather scenario provider’. Based on this alert the workability can be determined, meaning a check when and if a work-order can be executed. When there is a timeframe that the work-order can be executed a job can be created using a Farm Management Information System (FMIS). Based on the created job, the job should be scheduled, which can be supported by a scheduler atomic application component. Finally, a detailed prescription map can be created to spray the crops with high precision. These business processes can be supported by atomic application components that are configured into a blight control composite application component. In case of a process error (e.g. a machine breakdown on the field) this should be handled by designing a new job, or by starting an alternative process.

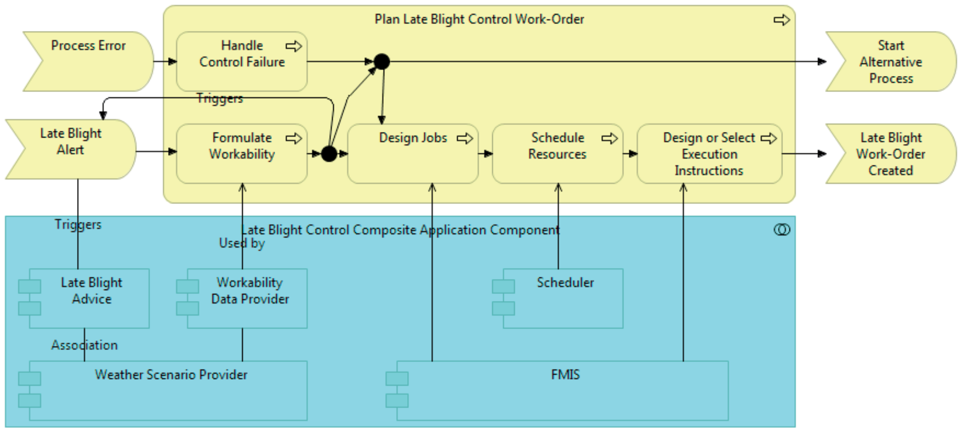

Plan Late Blight Control Work-Order
Process Error
Handle Control Failure
Start Alternative Process
Triggers
Late Blight Alert
Formulate Workability
Design Jobs
Schedule Resources
Design or Select Execution Instructions
Late Blight Work-Order Created
Triggers
Late Blight Control Composite Application Component
Used by
Late Blight Advice
Workability Data Provider
Scheduler
Association
Weather Scenario Provider
FMIS

Figure 51: Plan late blight control work-order sub-processes and Application Components.

In the 'execute late blight control work-order' business process the job is executed. This means that a worker (employee) receives the work-order, goes to the fields to spray the potatoes with a sprayer assembly (which is a specific resource instantiation) and perform aftercare by bringing the sprayer assembly back and if needed cleans it. This sprayer assembly includes a terminal with task controller. This task controller can become part of the configured blight control composite application component.

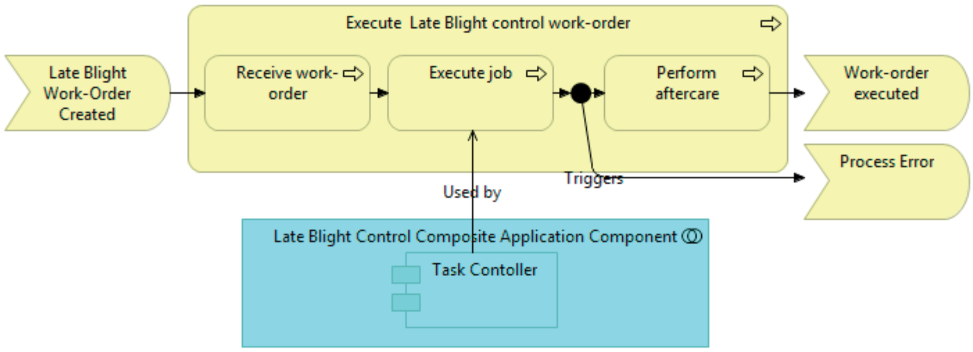

Execute Late Blight control work-order
Late Blight Work-Order Created
Receive work-order
Execute job
Perform aftercare
Work-order executed
Process Error
Triggers
Used by
Late Blight Control Composite Application Component
Task Contoller

Figure 52: Executer late blight control work-order.

After the fields are sprayed the 'check late blight control work-order' is executed, see Figure 53. In this process the data is received, checked and assessed. Based on these processes the data can be stored in a

FMIS and be used for other purposes (e.g. accounting, salary calculation, etc.).

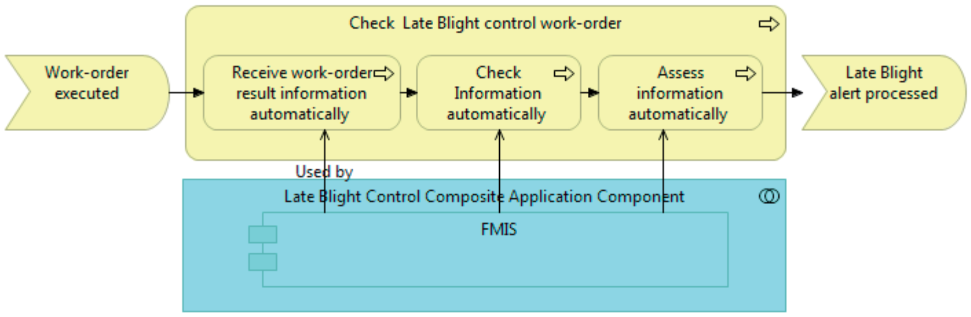

Check Late Blight control work-order
Work-order executed
Receive work-order result information automatically
Check Information automatically
Assess information automatically
Late Blight alert processed
Used by
Late Blight Control Composite Application Component
FMIS

Figure 53: Check late blight control work-order.

Based on this analysis using RAAgE 2.0 the following Atomic Application Components are identified that are required to support the business processes in a more advanced manner:

(i) Farm Management Information System provided by a software developer which has agriculture in its domain.
(ii) A Weather Scenario provided by a Weather Bureau.
(iii) A Late Blight Advice provided by a late blight Advisory Service Bureau.
(iv) Workability Data by a Workability Service Provider.
(v) A Schedule provided by a scheduling service bureau.
(vi) A Task Controller provided by a farm machinery manufacturer.

Details regarding these atomic application components can be found in the following section.

It can be expected that the business process configuration for Farmer A will not change so much in the future so it can be easily re-used when he just decides to use different variants of components (e.g. another weather service provider). At the same, time this specific configuration is added to the knowledge base and can be re-used as templates to configure applications for other farmers in a fast and affordable manner.

5.3.2 The developed or adapted atomic application components

In this section the developed or adapted application components are presented. These components are developed using the SDK that the FIspace platform offers. Furthermore, documentation describing the FIspace architecture, including capability model, is used for development.

5.3.2.1 The atomic application components

This section provides a brief description of the atomic applications that were developed for this use case including the actor that provides each component.

Farm Management Information System: The farm management application holds information about the fields, the crops growing on those fields and the available resources on the farm. The application keeps track of the performed field operations, used materials and resources and the development of the crops. For potato crops it can request an advice for late blight control. For the Late Blight Advice Application it will provide crop field data and for the Workability Application it will provide workability criteria. Based on the received advice (do nothing, act preventive, act curative or act eradicative) a crop protection product or a combination of products is chosen and if necessary, the manner of application. A request for a schedule can be done to a scheduler app. This schedule will provide an advice in the form of a proposed timing of the tasks which realize the operations to be performed (see Scheduler component). Based on the proposed schedule, tasks are formulated and forwarded to the sprayer following the ISO11783 standard. When the spraying is executed, the records of the performed field operations are actualized. This FMIS is provided by a software development company in the Netherlands.

Weather Scenario: This App delivers a weather scenario, which consists of the past weather for a specific location in the Netherlands and a prediction of the weather for the same location. The past weather is based on interpolation of weather data of official meteorological stations and can be enhanced by data from private weather stations near the specific location. The predicted weather is obtained by running the Weather Research and Forecasting (WRF) Model (Skamarock et al., 2008) with Global Forecast System (GFS) data[58] as start input. Predicted data is calculated for a narrow raster of 6 x 6 km and the spatial resolution can be further interpolated for the requested location. This app is provided by a weather bureau from Slovenia and runs as a web service at their premises.

Late Blight Advice: This App provides a late blight advice at the whole field level. It takes into account the variety, growth stages and applied crop protection actions of the potato crop, and uses a weather scenario to give an advice at four levels: do nothing, act preventive, act curative or act eradicative. At a later development stage, the atomic application component will use crop sensor data to estimate the biomass and/or leaf area. Based on these estimates, a site-specific advised dose of a chosen crop protection product can be determined. This app is provided by a research institute from the Netherlands.

Workability: This application calculates the conditions during spraying for a particular location based on criteria specified by the farmer and the actual and predicted weather, which is obtained from the Weather Scenario Application. Based on these data time periods to spray are determined. Workability for a particular technique of spraying is determined by wind speed that is not above a threshold within above mentioned time periods, that it does not actually rain in above mentioned time periods and that it is expected not to rain in a further specified time period after spraying. This app is provided by a company from Spain.

Scheduler: Based on (i) the required field operations, including the advised crop protection measures, (ii) the available workable periods, (iii) the cost of each field operation per proposed time frame and (iv) availability of resources, an optimal proposed schedule for the field operations is determined. The schedule is always feasible, as an ultimate consequence of not being able to perform an operation; the crop is lost with as consequence very high cost per proposed time frame. Operations are the agricultural measures to be taken; in the case of late

[58] https://www.ncdc.noaa.gov/data-access/model-data/model-datasets/global-forcast-system-gfs

blight control this is full field spraying. Tasks define how the operations are realized; in the case of late blight control that can be a tractor with a sprayer and a driver, or a self-propelled sprayer with a driver. This app is provided by a university and a company from the Netherlands.

Task Control: Task control implements the control over field operations performed by farm machinery. The field operation in this trial is spraying, which is to apply a spraying fluid mixed from water and crop protection products according to a specified dose. The dose can be specified for the whole field, but in a later stage also site-specifically i.e. for small patches in the field. In that later stage the dose can be varied e.g. depending on the measured crop reflectance. During the spraying process variables are logged to specify how the operation has been executed. This atomic application component is provided by a machine manufacturer that is located all over the world.

5.3.2.2 Capabilities and messages of the atomic application components

The FIspace platform enables configuration of multiple atomic application components into a working software solution based on capabilities. These capabilities enable communication between atomic application components of multiple vendors. The communication starts when an atomic application component (Capability Consumer) wants to use a capability of another atomic application component (Capability Provider) in the scope of a B2B process. During such a communication, FIspace coordinates the request and response messages of capabilities. In such a communication flow, the capabilities of the two application components need to be registered in FIspace on beforehand, and a business process definition in FIspace is also required in order to connect the registered capabilities of the Provider and Consumer.

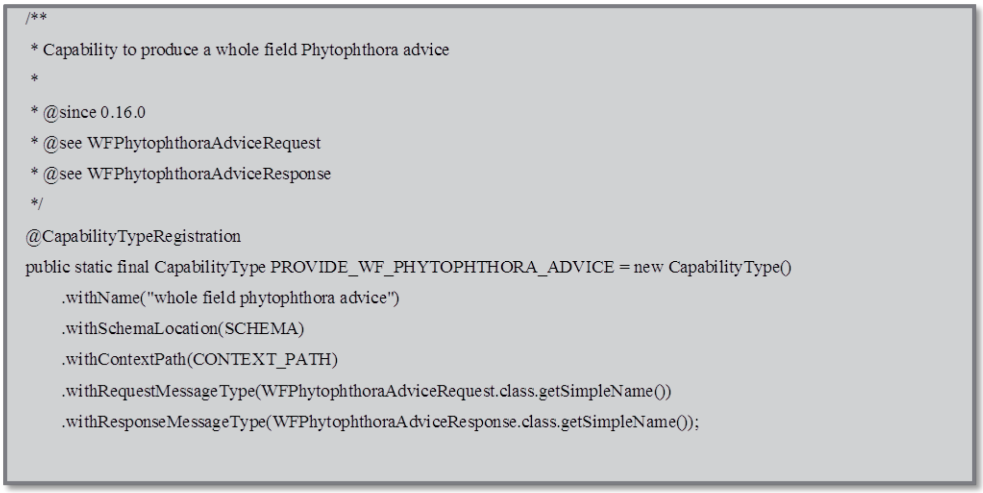
The atomic application components must provide at least one, and in some cases more Capabilities to the FIspace platform. For each Capability one or more Messages and eventually Events are defined, apart from the HTML responses. The capabilities are defined by the messages and events they can process. The messages have an element with the name of the message as root. These root elements contain for the agricultural domain elements from the drmCrop based XSDs and fulfil requirements regarding semantics, quality of the data (e.g.

resolution of the weather prediction). The capabilities must provide other requirements (e.g. performance to ensure that each atomic application component is accessible). An overview of the applications described in 0 with their Capabilities and Messages or Events is given in Table 16.

Table 16: Atomic Application Components with the capabilities and associated messages and events they realize or receive.

Atomic Application Component	Provided Capability	Message/Event
Application Component Composite (All Components)	RECEIVE_RESOURCE_ AVAILABLE_NOTIFICATION	ResourseAvailableNotification
Farm Management System	PROVIDE_ACTIVITYFIELD_ DATA	ActivityFieldDataRequest ActivityFielddataResponse
	PROVIDE_ACTIVITYFIELD_ LOCATION	ActivityFieldLocationRequest ActivityFieldLocationResponse
	PROVIDE_CROPFIELD_DATA_ WFPA	CropFieldDataWFPARequest CropFieldDataWFPAResponse
	PROVIDE_WORKABILITY_ CRITERIA	WorkabilityCriteriaRequest WorkabilityCriteriaResponse
	PROVIDE_TASK_DATA	TaskDataRequest TaskDataResponse
Late Blight Advice System	PROVIDE_WF_ PHYTOPHTHORA_ADVICE	WFPhytophthoraAdviceRequest WFPhytophthoraAdviceResponse
Scheduler	PROVIDE_SCHEDULE	ScheduleRequest ScheduleResponse
Task Controller	PROVIDE_TASK_DATA	TaskDataRequest TaskDataResponse
Weather Scenario Provider	PROVIDE_WEATHER_ SCENARIO	WeatherScenarioRequest WeatherScenarioResponse
Workability Data Provider	PROVIDE_WORKABILITY_ DATA	WorkabilityDataRequest WorkabilityDataResponse

The capability types and associated message types are defined in agricultural domain of the FIspace API. This ensures that the capability types can be found and can be configured. In Figure 54 an example is given of such a specification which is generated in java source code. An application developer can request for new capabilities and associated messages in a form on the FIspace platform. The java source code will be generated by the platform. The objective of the capability must be specified and is included as comment in the generated code as can be seen in the second line of Figure 54. It specifies in which version of the FIspace API the capability type is introduced and refers to messages where additional information can be found. The capability type itself is defined with a common name, the location where the schema of the message types can be found, the context in which it is used (in this case the agricultural domain) and the messages it is able to handle.



```
/**
 * Capability to produce a whole field Phytophthora advice
 *
 * @since 0.16.0
 * @see WFPhytophthoraAdviceRequest
 * @see WFPhytophthoraAdviceResponse
 */
@CapabilityTypeRegistration
public static final CapabilityType PROVIDE_WF_PHYTOPHTHORA_ADVICE = new CapabilityType()
        .withName("whole field phytophthora advice")
        .withSchemaLocation(SCHEMA)
        .withContextPath(CONTEXT_PATH)
        .withRequestMessageType(WFPhytophthoraAdviceRequest.class.getSimpleName())
        .withResponseMessageType(WFPhytophthoraAdviceResponse.class.getSimpleName());
```

Figure 54: Example of source code which specifies a capability with its associated messages.

The messages themselves are also specified in the FIspace API in the schema AGMessages.xsd. An example is shown in Figure 55. New messages are generated by the FIspace platform, based on information given by an app developer on a web form of the FIspace platform. The messages inherit the specification of the request message (ygg:RequestMessage) and response message (ygg:ResponseMessage) respectively, which are specified in the core components of FIspace. Both message types have an id (MessageID). The messages also use elements like the FieldGUID which in case of late blight advice request is the GlobalUniqueIdentifierType of a CropField. These latter elements are derived from the reference model for crop production, rmCrop.

```
<xsd:element name="WFPhytophthoraAdviceRequest">
  <xsd:complexType>
    <xsd:complexContent>
      <xsd:extension base="ygg:RequestMessage">
        <xsd:sequence>
          <xsd:element name="MessageID" type="xsd:ID" minOccurs="1" maxOccurs="1"/>
          <xsd:element name="FieldGUID" type="crpdt:GlobalUniqueIdentifierType" minOccurs="1" maxOccurs="1"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>

<xsd:element name="WFPhytophthoraAdviceResponse">
  <xsd:complexType>
    <xsd:complexContent>
      <xsd:extension base="ygg:ResponseMessage">
        <xsd:sequence>
          <xsd:element name="MessageID" type="xsd:ID" minOccurs="1" maxOccurs="1"/>
          <xsd:element name="CropField" type="crp:CropField" minOccurs="1" maxOccurs="1"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>
```

Figure 55: Example of the specification of two messages in AGMessages.xsd of the FIspace API.

These capability type definitions enable the individual atomic application components to exchange data. Within the FIspace platform these capability types are registered so that the FIspace platform can connect to the implemented capability of each individual component. The generic version of a capability is named a capability type. Different vendors can compete by registering their respective capabilities that are of the same capability type.

5.3.2.3 Evaluation - app Development

In the process of app development we found that the requirements and definition of capabilities is currently insufficient. Capabilities enable configuration of multiple atomic application components and must be unambiguously defined. Therefore, clear naming conventions for capabilities and messages are required, especially in an international community where English is not the mother language of many participants. Furthermore, it is currently not clear what functionality a capability should entail and how capabilities can be defined in a systematic way so that they can be re-used in other configurations. At least a detailed description of the capability type must be available, but in the future a systematic description will be required as well. Additionally, a generic reference for capabilities is required. Such a reference capability should describe the generic fields that are required of a capability (e.g. time, location, etc.). Within a capability a subset of the elements of this reference should be implemented.

5.3.3 App configuration: Late Blight protection

5.3.3.1 App Configuration

The app configuration process can be divided in two separated parts. First, there is the template set-up in which mainly the business architect and app developers are involved. Template set-up is linked with the development of the atomic application components. Second, there is the software configuration in which the business architect and end-user are involved.

Template set-up

The template set-up is an important step to facilitate the configuration process. In this phase the business architect creates a configuration template based on a description of a business process variant. In this configuration template the business architect determines the capability types that are required to support a business process variant (defined in Table 16). For other business process variants or other business processes the business architect can add, change or remove certain capability types. These capability types can be re-used in different processes (e.g. scheduling that can be used in spraying, planting, harvesting, etc.). These capability types can be used by software developers to develop capabilities. Software developers develop atomic application components, of which they can register capabilities in the FIspace platform. In an ideal situation multiple vendors would offer capabilities for each capability type. We created one configuration template. This template describes the interactions between the different capability types. This template is presented in the following section.

Software configuration

To create a composite application component that can support late blight control, configuration of atomic application components is required. In Figure 56 a process description is provided describing the configuration process of the blight control composite application component within the FIspace platform. In this figure the configuration hierarchy is represented by the processes select/configure business process template, select configuration template and configure apps.

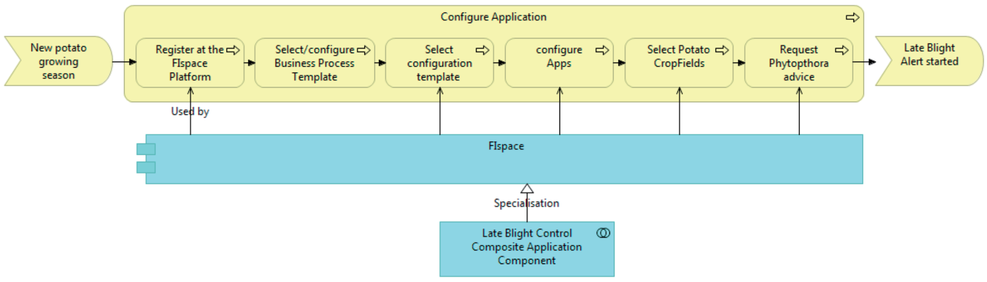
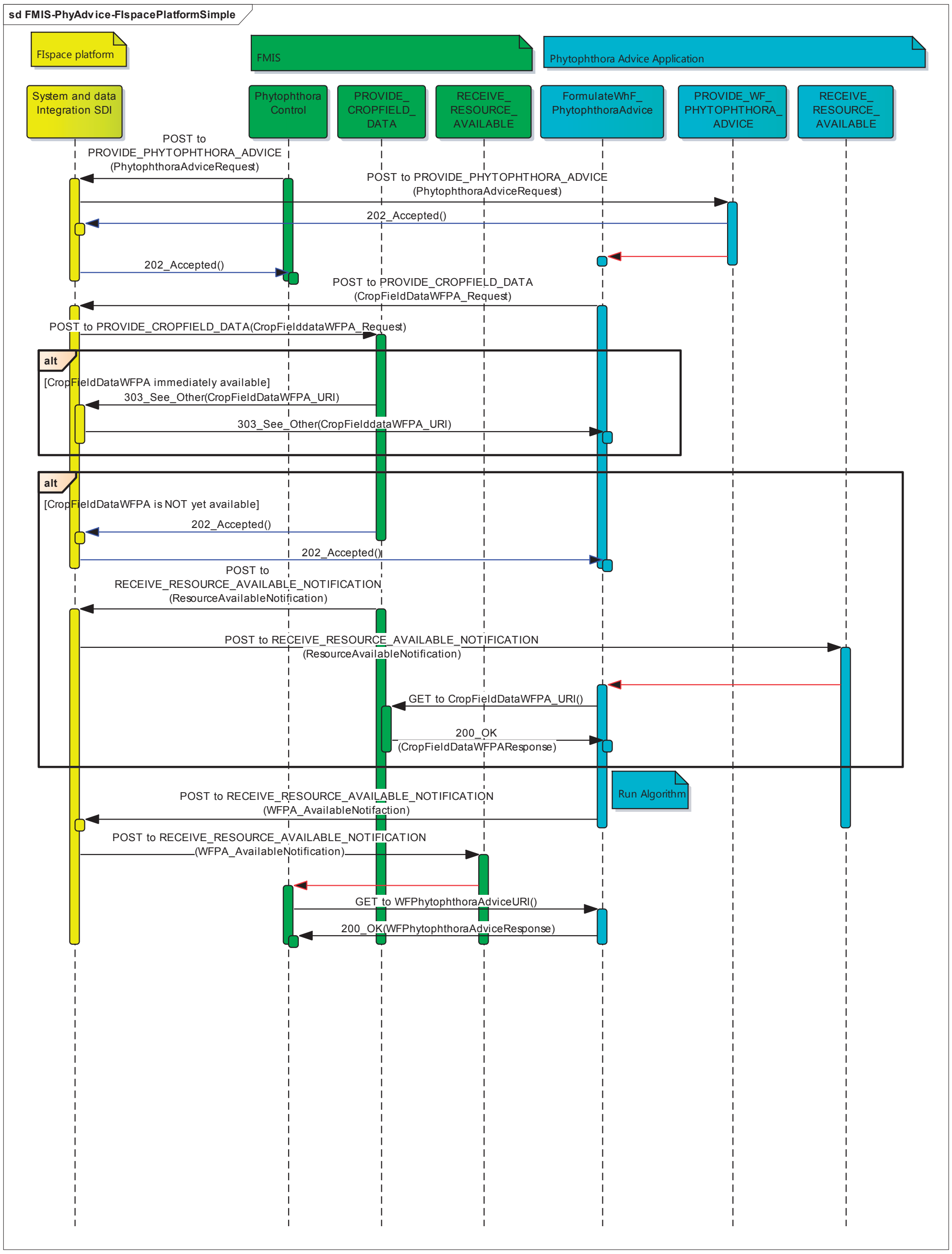

Configure Application
New potato growing season
Register at the FIspace Platform
Select/configure Business Process Template
Select configuration template
configure Apps
Select Potato CropFields
Request Phytopthora advice
Late Blight Alert started
Used by
FIspace
Specialisation
Late Blight Control Composite Application Component

Figure 56: configuration process to create a composite application component.

To start the process Farmer A needs to register as a user of the FIspace platform and then select or configure a Business Process Template which reflects his requirements. The business process template shows the business processes and the atomic application components to realize a business process. Based on the business process template, describing a business process variant, the configuration template is selected. Based on this configuration template the farmer chooses those apps from which he expects that they have the best capabilities for his farm. In this case study the business process will be supported by a selection of the developed atomic application components that are listed in Section 5.3.2.1.

5.3.3.2 App interaction diagram

The business architect uses the FIspace Platform to configure the atomic application components, which offer capability types, into a composite application component. This composite application component is supported by a configuration template describing the interactions of the atomic application components. The current prototype software has interactions between different clients and servers. The interaction between the FMIS as client and Late Blight Advice as server can be seen in Figure 57. In this case study the Late Blight Advice applications needs a weather scenario from a weather scenario provider, which is registered by a separate interaction template in the FIspace platform. Also the request for workable time, the request for a schedule, and exchange of task information are described as independent interaction templates. These additional interaction templates are not presented in this chapter, as they follow the same principle.



sd FMIS-PhyAdvice-FIspacePlatformSimple
FIspace platform
FMIS
Phytophthora Advice Application
System and data Integration SDI
Phytophthora Control
PROVIDE_ CROPFIELD_ DATA
RECEIVE_ RESOURCE_ AVAILABLE
FormulateWhF_ PhytophthoraAdvice
PROVIDE_WF_ PHYTOPHTHORA_ ADVICE
RECEIVE_ RESOURCE_ AVAILABLE
POST to PROVIDE_PHYTOPHTHORA_ADVICE (PhytophthoraAdviceRequest)
POST to PROVIDE_PHYTOPHTHORA_ADVICE (PhytophthoraAdviceRequest)
202_Accepted()
202_Accepted()
POST to PROVIDE_CROPFIELD_DATA (CropFieldDataWFPA_Request)
POST to PROVIDE_CROPFIELD_DATA(CropFielddataWFPA_Request)
alt
[CropFieldDataWFPA immediately available]
303_See_Other(CropFieldDataWFPA_URI)
303_See_Other(CropFielddataWFPA_URI)
alt
[CropFieldDataWFPA is NOT yet available]
202_Accepted()
202_Accepted()
POST to RECEIVE_RESOURCE_AVAILABLE_NOTIFICATION (ResourceAvailableNotification)
POST to RECEIVE_RESOURCE_AVAILABLE_NOTIFICATION (ResourceAvailableNotification)
GET to CropFieldDataWFPA_URI()
200_OK (CropFieldDataWFPAResponse)
Run Algorithm
POST to RECEIVE_RESOURCE_AVAILABLE_NOTIFICATION (WFPA_AvailableNotifaction)
POST to RECEIVE_RESOURCE_AVAILABLE_NOTIFICATION (WFPA_AvailableNotification)
GET to WFPhytophthoraAdviceURI()
200_OK(WFPhytophthoraAdviceResponse)

Figure 57: Diagram showing the interaction between a farm management information system as client and a provider of a late blight (phytophthora) advice as service using the SDI module of the FIspace platform.

The diagram in Figure 57 shows that communication between clients and servers is based on the REST protocol (Fielding, 2000). An important characteristic is that the communication between client and server is stateless, which means that the request message must contain all the information required to understand the request. As can be seen in Figure 57, this is implemented in such a way that in the initial request for an advice contains only the Global Unique Identifier (GUID) of the crop field, which requires a request from the server to the client for additional information from the crop field. As the server requires time to collect additional information on the crop field, which is a weather scenario that requires time for processing, it will not be able to respond within a reasonable time-out period. Therefore it will respond with html code 202, which indicates that the request is accepted for further processing. The server also requires additional information of the CropField from the FMIS by a POST to PROVIDE_CROPFIELD_DATA. There are two alternative responses shown in the diagram. The first alternative is that data is ready available and responds directly with the URI where the data can be collected. The second alternative is that there is time required to collect this data and there is a response with html code 202. When the crop field data is available a WFPA_ResourceAvailableNotification is posted. When the late blight protection advice is available, the server sends a resource available notification which contains the URI where the advice can be found. All messages are send to the FIspace platform, which forwards them to the other party, except the message to collect the advice after a resource available, which is directly sent from the client to the server.

The above described situation shows that the communication itself, following REST, is stateless, but this does not mean that the client and the server themselves should not keep track of the states of relevant objects.

5.3.3.3 Evaluation – configuration

From the app configuration process we found that the configuration of atomic application components requires detailed insight into the apps, capabilities and FIspace Platform. Currently, apps can be configured into a composite application component technically, although the quality of each or some of the capabilities can be insufficient to meet the farmer’s requirements. Therefore, capabilities and messages need detailed descriptions to reflect and clarify their functionality. An example is the whole field late blight advice capability and associated messages. Whether the advice request is intended incidentally or for a whole growing season is not specified. A whole season request requires a much more complex configuration and interaction template.

5.3.4 App usage - run-time example

5.3.4.1 The application usability

To test the usability of the FIspace platform to control late blight we used data from Farmer A of the growing season 2016. His farm has eight fields with potatoes which require from June 1st onwards advice on late blight control.

A complete late blight control process requires all the capabilities mentioned in Table 16. We were able to implement the request for a late blight advice, the request for a weather scenario by the advice application and the advice for workable time. The other capabilities have not been implemented at this moment. Figure 58 shows the user interface of the prototype software. This user interface is shown in a prototype FMIS (PragMis) that functions as the composite application component. The user interface describes the result of the advice of one of the eight fields with an indication for preventive, curative and eradicative control. Furthermore, it describes the available moments to spray based on the workability (in green) and the proposed spraying action. The data is exchanged using the FIspace SDI module. Although not all capabilities are implemented to support the whole late blight control business process, interaction between the different atomic application components is illustrated. Extending these interactions can be done in a similar way.

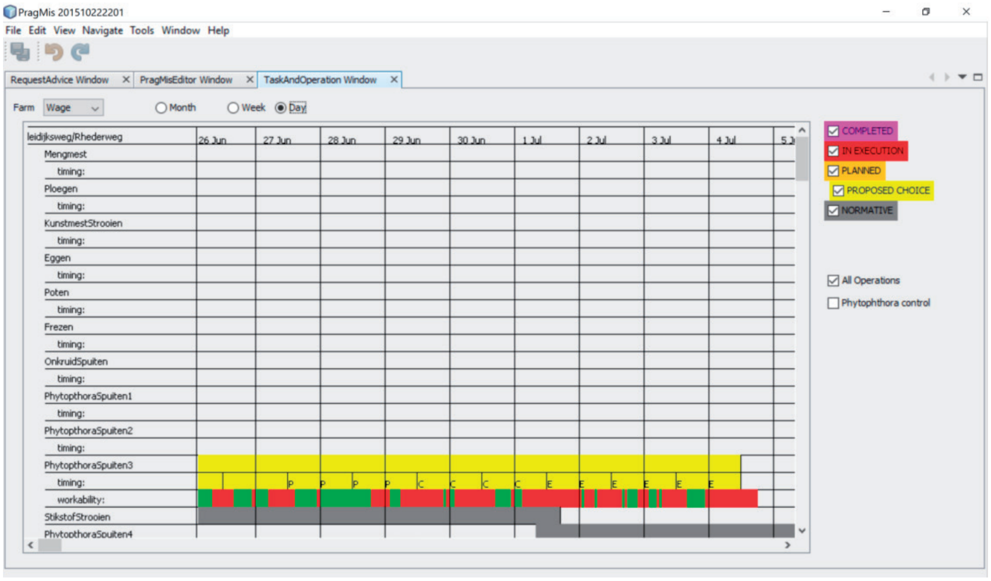


Figure 58: the user interface showing the available moments to spray against late blight (Phytophthora) based on the workability (in green) and the proposed spraying action (P: preventive, C: curative and E: eradicative).

5.3.4.2 Evaluation – app usage

From the usage of the composite application component we learned the following. First, consistent datasets must be used in the execution of a whole business process like late blight control. An example is the weather scenario which exists of historical data and predicted data. A weather scenario is used by the Late Blight Advice provider, but also by the provider of workability data. It is important that the same weather scenario is used while it is in reality possible that both providers would obtain data from different weather bureaus. Usage of different scenarios (data sets) would lead to mistakes in the scheduling and usage of the composite application component. Second, more capabilities are required than the ones that are mentioned in Table 16 to make good scheduling possible. To realize a realistic schedule much more information is required then yet foreseen by the specified messages and capabilities. Availability of resources includes also the agenda of the farmer as private obligations interfere with farming activities. In case of high cost of field operations per proposed time frame some obligations will be skipped, but others can't. Third, scheduling requires correct estimation of the time required to perform tasks. This could be provided as normative data by an application for example based on the Feldarbeitsrechner[59], but specific data recorded by the farm will improve the estimate. Fourth, workability of spraying must be differentiated depending on the product used for spraying, as new spraying products are introduced which require a shorter period without rain. Fifth, cost of tasks must be provided, preferably based on farm-specific information or alternatively based on normative data. Sixth, cost per proposed time frame can currently not be calculated as this requires estimation of yield losses when applications are delayed. This requires an extension of the Late Blight Advice app with for example the model of Shtienberg et al. (1990).

5.4 Discussion and conclusions

In this chapter we present an initial farm software ecosystem around the FIspace platform that is used to create prototype software that can be configured. The objective of this prototype software development within a farm software ecosystem was to provide a proof of concept showing that ICT Mass Customization combined with a Best-of-Breed approach can be realized in agriculture. To create the prototype software we started with describing a case study, late blight control, and the requirements to perform good control. Second, atomic application components have been developed. In this development process a design has been created based on RAAgE 2.0 describing the business processes, and required atomic application components. Then, prototype software has been developed in the form of six atomic application components. A selection of these atomic application components could interact with each other based on capabilities that were registered in the FIspace platform. The FIspace platform was used to develop and configure the atomic application components. Third, to configure the selected atomic application components using the FIspace platform, interaction templates were developed and used to connect the atomic application components and transform these into a composite application component. Finally, the composite application component has been deployed and used in a run-time environment. Data from the

[59] http://daten.ktbl.de/feldarbeit/home.html

case study was used to validate the composite application components. From these results it can be concluded that we have been able to create of proof of concept showing how an integrated FMIS can be configured from various apps provided by different vendors.

From this software development we learned that development and configuration of atomic application components into a composite application component requires extensive collaboration. Each vendor developing their own component should use the specifications that are determined by the platform which was in this case FIspace. Furthermore, the capabilities and documentation to facilitate collaboration needs to be unambiguously defined. Moreover, the capability types and capabilities implemented by the software vendors should be unambiguously described to enable configuration by the Business Architect.

Despite this complexity, operationalization of this configuration hierarchy will bring major benefits because software can become more modular and be re-used in a variety of different software instantiations. This requires that all software developers use and strictly follow the specifications provided by the open software enterprise of the Farm Software Ecosystem. This would increase the re-usability of atomic application components as more generic components can be re-used in other business processes variants or in business processes that realize another business service (e.g. planting, seeding). A first example that is shown in this research is the re-use of the weather component capability which is reused in two other atomic application components (Late Blight Advice and Workability Application Components) within the same business process variant.

Still, future research is required to realize ICT Mass Customisation in a more commercial setting. First, the FIspace platform should be improved by (i) configuration should become more user friendly, (ii) be able to keep track of the process and the orchestration of the atomic application components and (iii) the platform should be offered in a commercial setting. With these improvements the FIspace platform could become in a more commercially viable state.

Second, the RAAgE 2.0 model should be implemented in a software tool to enable automation of the configuration processes and operationalization of the hierarchical configuration methodology.

Third, configuration should be investigated in more detail for this specific case by (1) determining business process variants, using RAAgE 2.0, (2) defining all required capability types for these business process variants, (3) create instantiation templates for each business process variant, (4) create performance test for these instantiation templates and (5) start implementing these capabilities by multiple vendors and configure these in working solutions. Applying such an approach on one specific case would enable to dig into all details of that case and result in well-defined business process and configuration templates. Such detailed descriptions will be re-usable to create templates for all kind of other farm business processes such as fertilization, other types of crop protection, seeding and harvesting.

However, although all this future research will contribute to ICT Mass Customization we should not forget that to create a composite application component, all relations between the atomic application components should be very well understood. Therefore, configuration of atomic components will stay complex and templates of configurations are required to accelerate instantiations. Nevertheless, we expect that the realization of ICT Mass Customisation in combination with Best-of-Breed will bring major benefits as software can become more modular and be re-used in a variety of different software instantiations.

Based on a successful application of our hierarchical configuration methodology, we completed a proof of concept, described in this chapter, which encourages us to claim a plausible genericity of the methodology. We expect that applying this methodology to other arable farm business processes will only encounter small hurdles and be successful at a larger scale.

6

General discussion

6.1 Introduction

At the present time a third green revolution is needed as resources diminish while an increase in agricultural production is required to fulfil the demands of a growing world population. Farm enterprises can contribute to meet this challenge by advancing their management to increase food production while producing in a sustainable, safe and transparent manner. Advancement of farming requires integrated farm information systems as the related management styles are knowledge and information intensive. However, advancement of farm management is currently hindered because of interoperability issues between software systems of multiple vendors. In this thesis the aim was to improve farm enterprise integration by designing artefacts (ontologies, reference models and instantiations) that can enable development, configuration and instantiation of integrated farm information systems. This approach was largely based on the principles of ICT Mass Customisation (Verdouw et al., 2010a) in combination with Best-of-Breed (Light et al., 2001).

The overall approach and main results are depicted in Figure 59. Development of atomic application components – nowadays usually called Apps – takes place by different developers within the platform of a Farm Software Ecosystem. Specific platform features enable seamless configuration and integration of apps into a composite application component, called ICT Component. This configuration is based on and aligned with the specific business processes of an end-user, i.e. the farmer, and can be further customized and instantiated by connecting it to farm-specific processes and data. Configuration is tied to the role of a business architect/software developer who takes responsibility for an aligned configuration of business processes on the one hand and associated application components on the other hand. In practice this role can be performed by a same person or company providing for one or more atomic application components and willing to combine it with offerings form others participating in the platform. However, a crucial feature of this system is that components from multiple vendors can be integrated into one ICT Component which guarantees a Best-of-Breed selection. As indicated at the right side, by using this approach current farm business processes can be turned into advanced farm business processes. These advanced farm business processes are thus supported by an associated integrated ICT Component that is based on multi-vendor apps. The change towards advanced farm business processes does not only require instantiation of configured ICT Components but can require instantiation of additional business processes as well. The actors involved in a Farm Software Ecosystem are supported by an Open Software Enterprise. This Open Software Enterprise is an organization that regulates both technical and organization aspects (*cf.* Linux/Apache foundation). To facilitate collaboration and the development, configuration and instantiation of integrated ICT Components they provide collaboration artefacts such as data standards, a Software Development Kit and others.

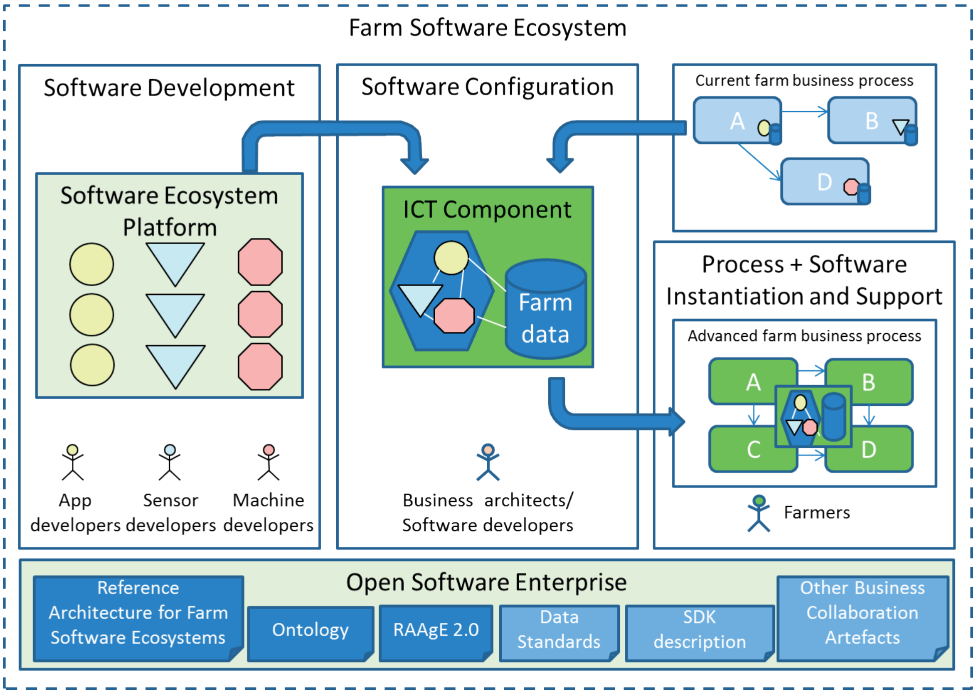

Farm Software Ecosystem
Software Development
Software Ecosystem Platform
App developers
Sensor developers
Machine developers
Software Configuration
ICT Component
Farm data
Business architects/ Software developers
Current farm business process
A
B
D
Process + Software Instantiation and Support
Advanced farm business process
A
B
C
D
Farmers
Open Software Enterprise
Reference Architecture for Farm Software Ecosystems
Ontology
RAAgE 2.0
Data Standards
SDK description
Other Business Collaboration Artefacts

Figure 59: A Farm Software Ecosystems able to configure integrated ICT Components that enable farmers to practice advanced farm management styles. The business collaboration artefacts (in dark blue), as part of the Open Software Enterprises, are developed in this thesis to support realization of such farm software ecosystems.

In this research we have added to these artefacts (i) a Reference Architecture for Farm Software Ecosystems that defines generic relationships between various actors and components, (ii) an ontology that provides a common language between the various actors within the Farm Software Ecosystem that have to collaborate and (iii) a Reference

Architecture for Agricultural Enterprises (RAAgE) 2.0 that enables development and configuration of business processes and ICT Components that are easy to replicate for various end-users. A more extensive description of Farm Software Ecosystems in provided in Chapter 3.

The remainder of this chapter provides more details on the developed artefacts by answering the research questions (Section 6.2), describing major contributions to practice and science (Section 6.3) and finally suggestions for future research (Section 6.4).

6.2 Answering the research questions

In the introduction of this thesis one main research question and three sub-questions are presented. The main question was:

How can reference architectures and models help to develop farm information systems that improve enterprise integration for advanced farm management styles?

This main question will be answered by first answering each sub question. The first sub question was:

What is the cause and nature of integration problems at farm enterprises?

Within this research we created a detailed problem description that elaborates on the cause and nature of current integration problems in farming (Chapter 2). To find problems with farm enterprise integration we focused on the research line called enterprise integration. Enterprise integration aims to improve interaction between enterprise entities to achieve domain objectives. In agriculture these domain objectives are advancement of farm management to increase food production while producing in a sustainable, safe and transparent manner. Enterprise integration is supported by enterprise architectures as these are used to facilitate the process of changing an organization from a baseline (as-is) to a target (to-be) architectural state. Based on this research line we created our first result which is a method to detect the cause and nature of integration problems in agriculture. This method was applied in a case study including three arable farm enterprises producing potatoes. These farm enterprises focused on improving their management and invested in new technologies for innovation. Within multiple steps of the method the architectural descriptions developed, facilitated communication and provided insight into problems of farm enterprises to achieve more advanced farm management. These case specific problems related to farm enterprise integration were analysed and formulated as more generic problems for farm enterprise integration. These generic problems where discussed with national and international experts as a validation.

To apply this method in case study research a reference model was required to design enterprise architectures in a uniform and efficient manner. Therefore, a reference model named the Reference Architecture of Agricultural Enterprises 1.0 (RAAgE) was developed. This reference model is described in a standard modelling language, named ArchiMate (TheOpenGroup, 2012), and shows important interrelations between the business, application and technology layers of farm enterprises. The reference model includes an ontology to provide a concise and precise, formal specification of the object system. This is required to allow shared understanding and effective communication between researcher, farmers, software developers and other stakeholders involved. The architectural descriptions depict in particular relations between farm business processes and ICT Components.

With RAAgE 1.0 and this method detailed problems descriptions have been created by instantiating the model for specific cases to show farm specific problems related to enterprise integration. These specific problems have thereafter been generalized and validated to show integration problems. Based on this research we found the cause and nature of integration problems at farm enterprises which are that ICT Components used within the same farm enterprise:

- (i) have partly overlapping and partly unique application services, functions and interfaces (that are non-standard);
- (ii) are missing required application services, functions and interfaces;
- (iii) have separate data repositories;

(iv) have inadequate and incomplete data exchange as semantics are not unambiguously defined;

(v) are hard to configure while this configuration is not supported by actors and tools.

Figure 60 provides an illustrative example of this problem using the representation formalism of RAAgE 1.0 to the cause and nature of a specific integration problem at farm enterprises.

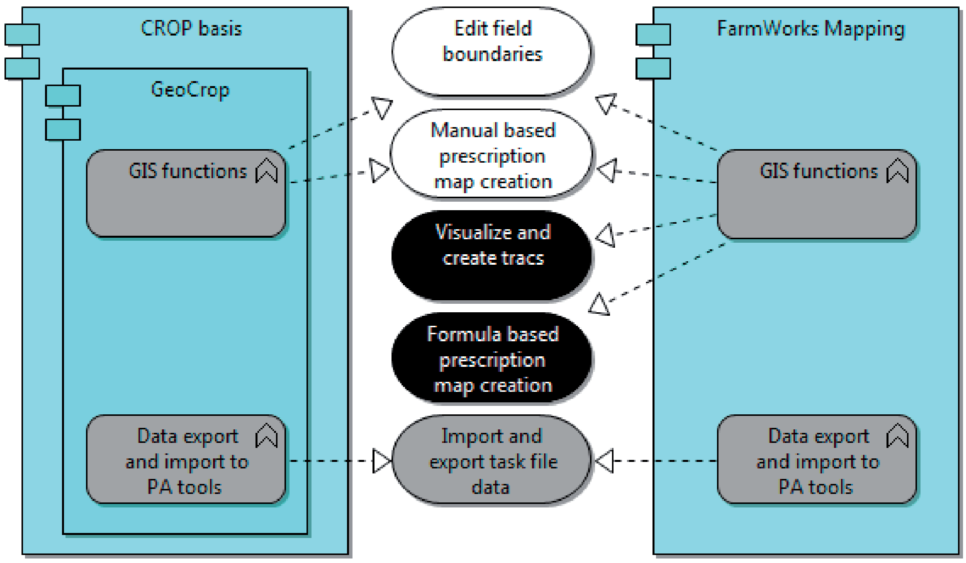

CROP basis
GeoCrop
GIS functions
Data export and import to PA tools
Edit field boundaries
Manual based prescription map creation
Visualize and create tracs
Formula based prescription map creation
Import and export task file data
FarmWorks Mapping
GIS functions
Data export and import to PA tools

Figure 60: Two application components with similar functions realizing overlapping (white), unique (black) and comparable (grey) services. Not all services offered by the applications are depicted.

Based on these results we described requirements that should address these integration problems. The main functional requirement categories are:

- smooth data handling and seamless data exchange between ICT Components (Kaloxylos et al., 2012; Kruize et al., 2013);
- a configuration approach to link ICT Components to each other in a meaningful and coherent way (Kaloxylos et al., 2012; Verdouw et al., 2014);
- interoperability of different ICT Components (Kaloxylos et al., 2012; Kruize et al., 2013);

- an open software enterprise that smoothly facilitates the previous points (Jansen et al., 2012).

It is expected that designing a framework that addresses these problems can solve current integration bottlenecks. First, this design must enable smooth data handling and seamless data exchange between ICT Components to solve inadequate and incomplete data exchange and enable integration of data repositories of multiple vendors. Second, it must include a configuration approach to link ICT Components to each other in a meaningful and coherent way fitting the business processes to be supported. This should be supported by actors that are willing to configure ICT Component of multiple vendors into an integrated solution. Third, the design must enable the formation of an open software enterprise to address the previous points and to organize collaboration between actors involved. This open software enterprise should focus both on improving interoperability to contribute in solving problems with partly overlapping and partly unique application services, functions and interfaces as well as on organizing the development of missing application services, functions and interfaces.

To address these integration challenges a framework was developed that could fulfil these requirements and answer the second research question:

How can integration problems at farm enterprises be solved by a framework with reference architectures and models that include both technical and organizational aspects?

In this research we first focused on organizational challenges to enable collaboration between software developers from multiple vendors. From literature we found that collaboration can take place in Software Ecosystems. Software Ecosystems are defined as the interaction of a set of actors on top of a common technological platform that results in a coherent set of ICT Components or Services (Manikas and Hansen, 2013). They can provide an effective way to construct large software systems on top of a software platform by combining components, developed by actors that are part of different organisations (Bosch, 2009; Manikas and Hansen, 2013; te Molder et al., 2011). To support instantiation of Software Ecosystems for farming, a Reference Architecture was developed. This reference architecture describes how software developers, farmers and other stakeholders need to collaborate to enable development, configuration and instantiation of integrated software solutions. More specifically, it can be used to map, assess, design and implement Farm Software Ecosystems that can decrease current problems with farm enterprise integration.

The reference architecture for Farm Software Ecosystems is described in detail in Chapter 3 and comprises five main components:

- (i) **Actors**, which are basically app developers, business architects/software developers and end-users, *i.e.* farmers that finally use the configured ICT Components and services;
- (ii) **Platform** that enables configuration of Atomic Application Components into integrated information systems for farmers;
- (iii) **Open software enterprise** that manages the relation between the actors and the platform;
- (iv) **Business services** that support software configuration, development and hosting;
- (v) **ICT Components** that include configured atomic application components from multiple vendors allowing seamless data exchange based on standards

The Reference Architecture for Farm Software Ecosystems mainly addresses the organizational part of this research question. The technical part on the configuration of different ICT Components into integrated solutions was not yet sufficiently covered in the framework. Therefore we started a research, presented in Chapter 4, to develop a technical framework that helps to improve integrating capabilities of ICT Components, focussing on configuration and ICT Mass Customisation. In this research RAAgE 1.0 was extended into RAAgE 2.0 supporting technical aspects related to configuration of ICT Components by providing a hierarchical configuration methodology. This methodology divides configuration in two steps (i) business process configuration and (ii) software configuration. To enable business process configuration the model comprises three reference models, i.e. on products, processes and resources. The dependencies between these models are defined in rules that define possible combinations of products, processes and resources and that constrain the configuration of farm-specific models i.e. instances. RAAgE 2.0 also includes a configuration tree and templates. Templates describe a set of pre-configured product, process and resource models for typical cases. Variety in farm business processes can be modelled with business process variants. Such a variant realizes a similar kind of business services (e.g. basic fertilization, precision fertilization). Each variant has partly overlapping business processes and resources and unique ones. RAAgE 2.0 provides insight into these specific and generic parts. The other part of the methodology, software configuration, is divided into two additional sub-steps. The first sub-step is to create configuration templates that describe required (generic) application services (capability types) and their interactions to support specific business process variants. This sub-step is typically performed by a business architect in close collaboration with software developers. The second sub-step is the selection and configuration of the specific capabilities for one configuration template. Capabilities are realized by atomic application components that can be developed by multiple vendors. This second sub-step is performed by a business process architect in close collaboration with a farmer. With this extension RAAgE 2.0 supports (i) development of ICT Components that fit within an ICT Mass Customisation and Best-of-Breed approach, (ii) selection of ICT Components based on business processes that they should support and (iii) getting insight into configuration of different atomic application components into an integrated ICT Component.

With this organisational and technical framework, including an ontology, Reference Architecture for Farm Software Ecosystems and RAAgE 2.0, it is expected that we can solve integration problems of farm enterprises. To proof that this is really possible we answered the third research question:

How can we substantiate that the framework will enable a solution for integration problems at farm enterprises?

To answer this question a proof of concept, including prototype software, is presented in Chapter 5 showing the feasibility of ICT Mass Customisation in combination with Best-of-Breed in arable farming. A proof of concept is defined as a phase in development, in which experimental hardware or software is constructed and tested to explore and demonstrate the feasibility of a new concept (United et al., 1969). Realizing ICT Mass Customisation requires (Verdouw et al., 2010a): (i) software modularity, (ii) an information integration platform, (iii) component availability, (iv) configuration support and (v) reference information models. To fulfil these requirements a design was developed and instantiated for a specific use case on late blight protection in potato growing for a specific farmer in The Netherlands. For that purpose we:

(i) configured the business processes that are involved in late blight protection using RAAgE 2.0 to identify which advanced components are needed to support this process for this farmer;
(ii) developed the required advanced components that were identified in the previous step using the FIspace platform. These components were provided by different app developers from five different European countries;
(iii) configured a composite application component using the configuration methodology of RAAgE 2.0;
(iv) instantiated and executed the ICT Component using the FIspace platform for this specific farmer.

This resulted in prototype software that showed how we can configure business processes and multi-vendor components into an integrated ICT Component to support late blight protection in potatoes for a specific farmer. It was made plausible that this approach is also applicable to create software able to support other business processes in agriculture.

By answering the sub-questions we are able to answer the main research question and conclude that an organisational and technical reference model can help to develop ICT Components that improve farm enterprise integration by facilitating collaboration between the actors involved. We substantiated that reference models can contribute to solving integration problems by developing a proof of concept. In this proof of concept we have used our hierarchical configuration methodology defined in RAAgE 2.0, the ontology and the Reference Architecture for Farm Software Ecosystems. This proof of concept substantiated that software could be developed by combining contributions of multiple vendors and that these could be configured into an integral part of a farm information system (ICT Component). Based on this successful application of our artefacts, we completed a proof of concept, which encourages us to claim a plausible genericity of the methodology. We expect and firmly believe that applying this methodology to other arable farm business processes will only encounter some surmountable hurdles and will be successful at large. Still, the integration problem itself should be solved by software vendors as these organisations need to work together on some aspects and be competitors on other aspects to realize ICT Mass Customisation in combination with Best-of-Breed.

6.3 Major contributions to practice and science

The main results in this thesis are a number of artefacts namely:

(1) RAAgE 1.0 that can describe farm enterprise architectures in a uniform and efficient manner;
(2) A problem description, which is a case specific instantiation of RAAgE 1.0 generalized into a generic problem description;
(3) Ontology that supports communication between collaborating actors;
(4) Reference Architecture for Farm Software Ecosystems that defines generic relationships between actors and components;
(5) RAAgE 2.0 that is a technical reference model to support configuration of business processes and ICT Components;
(6) Prototype software that serves as a proof of concept substantiating that all previous components will provide a solution for integration problems at farm enterprises.

These results are developed within a design oriented research (DOR) approach based on case studies. The purpose of DOR is to create innovative artefacts that extend the boundaries of human and organizational capabilities. Within such an approach the scientific relevance of the artefacts, and the prerequisite to become part of the knowledge base, is that the artefacts must be applicable in the appropriate Environment. Furthermore, an artefact is an addition to the knowledge based when it proofs that it provides in its environment (1) a new solution to an existing problem that has not been solved before or (2) a better solution to a problem that has already been solved. In this research we provide artefacts that that can enable farm enterprise integration. These artefacts provide a new solution to a problem that has not been solved in this way in agriculture before.

All artefacts that were developed were verified or validated in different steps of the research mainly by case studies. In Table 17 we provide an overview of the artefacts and their verification or validation.

Table 17: Description of the verification and validation of the artefacts that are developed in this thesis.

Artefacts	**Verification**	**Validation**	**Description**
RAAgE 1.0	no	Yes, experts validation and usage of the model in a case study	Semi-structured interviews were held with two experts. Additionally, the reference model was used to describe the enterprise architecture of three arable farms from the Netherlands.
Problem Description	No	Yes, expert validation	Semi-structured interviews were held first with the farmers to validate the case specific problems. Second semi-structured interviews were held with national and international experts to validate the generic problem descriptions.
Ontology	No	Yes, usage of the ontology	The ontology is used in multiple researches to describe the object system.

Table 17: Continued.

Artefacts	Verification	Validation	Description
Reference Architecture for Farm Software Ecosystems	Yes	Yes, expert validation and usage of the model in a case study	First a verification of the model was performed based on the requirements. Second, semi-structured interviews have been held with experts to validate the model. Moreover, the assessment and mapping functionally was validated by usage of model in a case study in which two existing farm software ecosystems are assessed and mapped.
RAAgE 2.0	Yes	Yes, by a case study in which prototype software was created	First a verification of the model was performed based on the requirements and the usage of the model in an exemplar. Second, the model was validated by usage of the model to describe and create prototype software.
Prototype Software	Yes	Yes, by usage in a case study in which data of a farmer was used	First, the prototype software was verified by testing if the software worked. Second, the model was validated by testing the software in a case study. In this case study data of a farmer was used to test if the advice was realistic.

The remainder of this section describes the practical relevance and scientific contribution of the results from this thesis.

Practical relevance

In this research artefacts and other results are developed that enable farm enterprise integration to advance their management. To enable farm enterprise integration we focus on primary processes of farm enterprise, which are the processes involved in crop production (the management cycle of a crop). To contribute to improvement of farm enterprise integration we developed artefacts of which their practical relevance is described in this section.

First, the ontology provides definitions of objects regarding farm enterprise integration. It is used in the reference models RAAgE 1.0, RAAgE 2.0, the Reference Architecture of Farm Software Ecosystems and for the development of the proof of concept. The ontology defines objects related to farm enterprises and provides unambiguous semantics and shared understanding between stakeholders. Therefore, the ontology is relevant for all stakeholders in agriculture focusing on farm enterprise integration and can be reused by them.

Second, a problem description has been created describing problems related to farm enterprise integration in detail. This problem description is a case specific instantiation of RAAgE 1.0 generalized to a generic problem description. The case specific instantiations provides explicit insight into functionalities a farmer needs to support his business processes and how these functionalities match or not match with software packages. These descriptions are relevant making decisions about what software to use or buy. The generalized problem descriptions are relevant for software vendors in deciding how to develop software for farmers. The problem description was created using a method and RAAgE 1.0. Both can be reused in the future to find other or new problems with farm enterprise integration.

The problem description was used to develop a framework that is able to contribute to solving current integration problems. The organizational part of this framework, a Reference Architecture for Farm Software Ecosystem enables collaboration between multiple actors as it can:

(i) describe what constitutes a farm software ecosystem to support its development;
(ii) map current Farm Software Ecosystems describing commonalities and differences;
(iii) assess current Farm Software Ecosystems and suggest improvements;

(iv) facilitate stakeholders in decision making to determine in what kind of Farm Ecosystems to participate in;
(v) support development of integrated software solutions by providing an organizational structure.

The functionality provided by this reference architecture can be useful for multiple stakeholders involved in farm enterprise integration.

The other, more technical part of the framework, RAAgE 2.0, supports the development and configuration of ICT Components that can be used for farm enterprise integration. RAAgE 2.0 provides insight into farm business processes and related ICT Components as it enables mapping of enterprise architectures. The reference model can support software developers by providing insight into farm business processes and into more specific configurations (variants) of farm business processes. Moreover, business architects can use the model to depict current farm business processes and reconfigure these, based on new resources to advance farm management.

Finally, to show the usability of the developed artefacts a proof of concept including prototype software was developed. This prototype software shows the usages of the different artefacts developed and provides detailed knowledge to software developers, farmers and other stakeholders on how ICT Mass Customisation in combination with Best-of-Breed can be realized.

With the development of these artefacts multiple organisations including software developers, farmers, business architects and other stakeholders can start collaborations that result into ICT Components and Business Services that solve current integration challenges.

Scientific contribution
Through this thesis and publications in several peer-reviewed journals the developed artefacts have been added to the knowledge base of Design Oriented Research and therefore scientifically relevant. To create these artefacts we have used multiple modelling languages, concepts, reference models and other knowledge from literature. We used knowledge from general science and applied this to another domain, i.e. agriculture, and thus enriched the knowledge base. This enrichment is based on the specific characteristics of the agricultural domain, more

specifically arable farming, which can be interesting for other domains for several reasons. First, farmers aim to control dynamic, biological processes in an environment that can only be controlled to a certain extent. Second, resources of multiple organisations (e.g. fertilizers, tractors, implements) need to be brought together in a business process to enable management of infield variability of crops. Third, software systems of multiple vendors (e.g. advisory services, FMISs, scheduling services) need to collaborate to support these farm business processes.

The problem description based on RAAgE 1.0 has enriched the knowledge base because we have been able to systematically describe integration problems in detail. In literature, integration problems are commonly described as problematic, but mainly either on a high level of abstraction or within the context of a generic framework (Giachetti, 2004). However, the nature and cause of integration problems, as subject in this thesis, are often not made explicit for a specific domain. The integration problems found in agriculture can make current issues in enterprise integration more explicit for other domains.

To operationalize configuration of ICT Components an organisational Reference Architecture for Farm Software Ecosystems was developed that facilitates collaboration between actors. In other domains such collaborations have taken place within Software Ecosystems for a longer time (e.g. Linux, Android) but also there this is a relatively new research line (Messerschmitt and Szyperski, 2003). We contributed to this research line by creating a reference architecture for farming that can be re-used for other domains. Most pertinent in Farm Software Ecosystems is that configuration of multiple atomic application components is required. These insights can be reused in other domains as organizing configuration within Software Ecosystems is a new development.

To solve challenges regarding farm enterprise integration we used the principles of ICT Mass Customisation and Best-of- Breed. Based on these principles RAAgE 2.0 was developed supporting the configuration of atomic application components into a composite application component that is aligned with farm business processes. This is supported by a hierarchical configuration methodology that contains two steps: business process configuration and software configuration. In this methodology first business process configuration starts to understand the requirements of the supporting software systems, taking available resources into account. However, most literature usually focusses on either process configuration (Buijs et al., 2013; Gottschalk et al., 2008; Rosemann and van der Aalst, 2007b) or on product configuration (Forza and Salvador, 2002; Jiao et al., 2007). Both approaches do not directly fit to the requirements for agriculture because constraints by farm-specific resources (e.g. soil, equipment) are restricting the configuration process. The business process configuration for agriculture in this thesis is unique because it departs from the available resources and subsequently reduces the possible variants of business processes.

Finally, these contributions are not only theoretical constructs, as they are implemented in a proof of concept. In this proof of concept we show how these theoretical concepts can be used to realize ICT Mass Customisation in combination with Best-of-Breed in agriculture.

6.4 Directions for further research

In this thesis artefacts and other results are presented contributing to farm enterprise integration by ICT Mass Customisation in combination with Best-of-Breed. Each individual artefact that was developed has proven practical and scientific relevance contributing to the existing knowledge base. Although each artefact was validated and/or verified and a proof-of-concept of the whole approach was provided, there is still room for future improvements and extensions. The remainder of this section will discuss the most important directions for further research.

First, the ontology can be extended to provide more specifications to enable description of the object system of interest in a more concise, precise and formal manner. Furthermore, there are possibilities to better align the ontology and the business processes of RAAgE with the rmCrop data model by merging these into the same tool (e.g. enterprise architect[60]) that would enhance its use. Second, the problem description and method using RAAgE 1.0 can be used to extend the list of specific and generic problem descriptions in farming. Third, the RAAgE 2.0

[60] http://www.sparxsystems.com/products/ea/index.html

model could be implemented in a software tool to enable automation of configuration processes and operationalize the model. During this implementation process business rules of processes should be extended.

Implementation could encourage usage of the model, which will then result in availability of additional templates. Fourth, the Reference Architecture for Farm Software Ecosystems could be enriched by using the model in mapping, assessment, design and implementation processes of other farm software ecosystems. Usage of the Reference Architecture can result in new insights and improvements such as extensions of the list of collaboration artefacts. Finally, the developed proof of concept to support late blight protection in potatoes is only available as prototype software that can be improved. More specifically, currently the definition of capability types, selection of required capability types in a configuration template and the selection of capabilities and instantiation is still cumbersome. In future research the prototype software could be improved to make it available as commercial software.

Furthermore, to realize ICT Mass Customisation in combination with Best-of-Breed in configuration processes we recommend starting a research on business models to gain insight into the motives of software developers to become part of Farm Software Ecosystems. Insight into these motives can enhance the adoption of software ecosystems for agriculture which makes the concept of ICT Mass Customisation more feasible. It is expected that in this line of research attention to governance aspects such as liability, data ethics and ownership of data is needed.

Finally, configuration of atomic application components and supporting tools should be researched in more detail. The size of the proof of concept was not really able to show ICT Mass Customisation in combination with Best-of-Breed to its full extent because that would require many actors and components, so it is not clear if this can be done in a cost-efficient and adequate manner. In this research line configuration should be investigated in more detail for a specific case (e.g. late blight control) by (1) determining business process variants, using RAAgE 2.0, (2) defining all required capability types for these business process variants, (3) create instantiation templates for each business process variant, (4) create performance test for these instantiation templates and (5) start implementing these capabilities by multiple vendors and configure these into working solutions. Applying such an approach on only one specific case would enable to dig into all details of the case. Such a detailed description will be re-usable for all kind of other farm business processes such as fertilization, other types of crop protection, seeding and harvesting.

Future work should lower current hurdles to realize ICT Mass customisation in combination with a Best-of-Breed approach. Therefore we suggest continuing this research line. In research programs, consortia of researchers, companies and governmental organisations need to collaborate. With additional research, ICT Mass Customisation in combination with Best-of-Breed can become a common practice. This would result into future Farm Software Ecosystems that facilitate collaboration between app developers to develop modular software components that can be configured. In this manner, small software development companies could focus on development of small Application Components while relying on larger ICT players that provide general infrastructural components (data storage, servers, etc.) or more complex analyses (e.g. super computers, big data) by generic software components. This will stimulate innovation by giving small start-up companies a fair chance to accelerate their innovative product and gain market share. Business architects can keep an overview of available application components and advice farmers to configure customized farm information systems. This configuration can be supported by RAAgE 2.0. In the end this would lead to farm enterprise integration resulting in advancement of farm management and consequently into higher yields while fewer resources are used.

References

Akkermans, H.A., Bogerd, P., Yücesan, E., van Wassenhove, L.N., 2003. The impact of ERP on supply chain management: Exploratory findings from a European Delphi study. European Journal of Operational Research 146, 284-301.

Anonymous, 1987. Informatiemodel "Open Teelten"-bedrijf, Publicatie no. 37, juni 1987. Proefstation voor akkerbouw en groenteteelt in de volle grond, Lelystad, the Netherlands.

Anthony, R.N., Govindarajan, V., 2006. Management Control Systems. 12th ed. McGraw-Hill/Irwin, New York.

Aubert, B.A., Schroeder, A., Grimaudo, J., 2012. IT as enabler of sustainable farming: An empirical analysis of farmers' adoption decision of precision agriculture technology. Decision support systems 54, 510-520.

Baldwin, C.Y., Woodard, C.J., 2008. The architecture of platforms: A unified view, Platforms, Markets and Innovation. Research Collection School Of Information Systems, Singapore.

Barmpounakis, S., Kaloxylos, A., Groumas, A., Katsikas, L., Sarris, V., Dimtsa, K., Fournier, F., Antoniou, E., Alonistioti, N., Wolfert, S., 2015. Management and control applications in Agriculture domain via a Future Internet Business-to-Business platform. Information Processing in Agriculture 2, 51-63.

Beers, G., Beulens, A.J.M., Trienekens, J.H., 1994. Global reference information models for product chains in agriculture: A case of apples and pears, In: Hagelaar, G. (Ed.), Management studies and the Agri-business: Management of Agri-chains. Proceedings of the First International Congress on Agri-Chain Management, Wageningen, the Netherlands, March 24-25, 1994. Department of Management Studies, Wageningen Agricultural University, Wageningen, pp. 205-217.

Benavides, D., Segura, S., Ruiz-Cortés, A., 2010. Automated analysis of feature models 20 years later: A literature review. Information Systems 35, 615-636.

Benbasat, I., Goldstein, D.K., Mead, M., 1987. The case research strategy in studies of information systems. MIS quarterly, 369-386.

Beulens, A.J.M., Broens, D.F., Folstar, P., Hofstede, G.J., 2005. Food safety and transparency in food chains and networks - relationships and challenges. Food Control 16, 481-486.

Blackstone, J.H., Cox, J.F., 2005. APICS Dictionary Alexandria, VA.

Bongiovanni, R., Lowenberg-Deboer, J., 2004. Precision Agriculture and Sustainability. Precision Agriculture 5, 359-387.

Bonoma, T.V., 1985. Case research in marketing: opportunities, problems, and a process. Journal of marketing research 22, 199-208.

Bosch, J., 2009. From software product lines to software ecosystems, Proceedings of the 13th international software product line conference. Carnegie Mellon University, pp. 111-119.

Bramley, R., 2009. Lessons from nearly 20 years of Precision Agriculture research, development, and adoption as a guide to its appropriate application. Crop and Pasture Science 60, 197-217.

Buijs, J.C., van Dongen, B.F., van der Aalst, W.M., 2013. Mining configurable process models from collections of event logs, In: F. Daniel, J. Wang, Weber, B. (Eds.), Business Process Management. Springer Berlin, Heidelberg, pp. 33-48.

Bussler, C., Haller, A., Thomas, O., 2006. Understanding the Term Reference Model in Information Systems Research: History, Literature Analysis and Explanation, Business Process Management Workshops. Springer, Berlin, Heidelberg, pp. 484-496.

CBS, 2012. Landbouw; gewassen, dieren en grondgebruik naar regio. Centraal Bureau voor de Statistiek, Den Haag.

Chen, D., Doumeingts, G., Vernadat, F., 2008. Architectures for enterprise integration and interoperability: Past, present and future. Computers in Industry 59, 647-659.

Chen, D., Vernadat, F., 2004. Standards on enterprise integration and engineering—state of the art. International Journal of Computer Integrated Manufacturing 17, 235-253.

Chituc, C.-M., Azevedo, A., Toscano, C., 2009. A framework proposal for seamless interoperability in a collaborative networked environment. Computers in Industry 60, 317-338.

Clarke, M.P., 1998. Virtual logistics: an introduction and overview of the concepts. International Journal of Physical Distribution & Logistics Management 28, 486-507.

Clements, P., Northrop, L., 2002. Software product lines. Addison-Wesley, Boston.

Cook, S., Adams, M., Bramley, R., Robert, P., Rust, R., Larson, W., 2000. What is obstructing the wider adoption of precision agriculture technology, Proceedings of the 5th International Conference on Precision Agriculture, Bloomington, Minnesota, USA, 16-19 July, 2000. American Society of Agronomy, pp. 1-7.

Cox, S., 2002. Information technology: the global key to precision agriculture and sustainability. Computers and Electronics in Agriculture 36, 93-111.

Crossette, B., Kollodge, R., 2011. The state of World Population 2011. UNFPA, New york.

Eisenmann, T.R., Parker, G., Van Alstyne, M.W., 2008. Opening platforms: how, when and why? This paper has been published under the same title as Chapter 6 in Platforms, Markets & Innovation (ed. Gawer, 2009) pp 131-162; Harvard Business School Entrepreneurial Management Working Paper No. 09-030., 131-162.

Evenson, R.E., Gollin, D., 2003. Assessing the impact of the Green Revolution, 1960 to 2000. Science 300, 758-762.

FAO, 1996. Lessons from the green revolution: towards a new green revolution, World Food Summit FAO, Rome.

Fettke, P., Loos, P., 2003. Classification of reference models: a methodology and its application. Information Systems and E-Business Management 1, 35-53.

Fielding, R.T., 2000. Architectural styles and the design of network-based software architectures. University of California, Irvine.

Forza, C., Salvador, F., 2002. Managing for variety in the order acquisition and fulfilment process: The contribution of product configuration systems. International journal of production economics 76, 87-98.

Fountas, S., Kyhn, M., Jakobsen, H., Wulfsohn, D., Blackmore, S., Griepentrog, H., 2009. A systems analysis of information system requirements for an experimental farm. Precision Agriculture 10, 247-261.

Fountas, S., Pedersen, S.M., Blackmore, S., 2005. ICT in Precision Agriculture–diffusion of technology. ICT in agriculture: perspective of technological innovation, E. Gelb and A. Offer (eds), E:Book: http://departments. agri. huji. ac. il/economics/gelb-main. html.

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, 192-210.

Gawer, A., Cusumano, M.A., 2002. Platform leadership: How Intel, Microsoft, and Cisco drive industry innovation. Harvard Business School Press, Boston.

Gerring, J., 2006. Case study research: principles and practices. Cambridge University Press.

Giachetti, R.E., 2004. A Framework to Review the Information Integration of the Enterprise. International Journal of Production Research 42, 1147-1166.

Gottschalk, F., Van Der Aalst, W.M., Jansen-Vullers, M.H., La Rosa, M., 2008. Configurable workflow models. International Journal of Cooperative Information Systems 17, 177-221.

Grunert, K.G., 2005. Food quality and safety: consumer perception and demand. European Review of Agricultural Economics 32, 369-391.

Handoyo, E., Jansen, S., Brinkkemper, S., 2013. Software ecosystem roles classification, Software Business. From Physical Products to Software Services and Solutions. Springer, pp. 212-216.

Harris, M., 1977. Cannibals and Kings - The orgins of cultures. Vintage Books, New York.

Hartog, R.J., 2012. On design-oriented research and digital learning materials in higher education. Wageningen University, Wageningen.

Hevner, A.R., 2007. The three cycle view of design science research. Scandinavian Journal of Information Systems 19, 87-92.

Hevner, A.R., March, S.T., Park, J., Ram, S., 2004. Design Science in Information Systems Research. MIS Quarterly 28, 75-105.

Hossain, E., Babar, M.A., Paik, H.-y., 2009. Using scrum in global software development: a systematic literature review, Global Software Engineering, 2009. ICGSE 2009. Fourth IEEE International Conference on. Ieee, pp. 175-184.

Iftikhar, N., Pedersen, T.B., 2011. Flexible exchange of farming device data. Computers and Electronics in Agriculture 75, 52-63.

Jacobs, F.R., 2007. Enterprise resource planning (ERP)—A brief history. Journal of Operations Management 25, 357-363.

Jansen, S., Brinkkemper, S., Souer, J., Luinenburg, L., 2012. Shades of gray: Opening up a software producing organization with the open software enterprise model. Journal of Systems and Software 85, 1495-1510.

Jiao, J.R., Simpson, T.W., Siddique, Z., 2007. Product family design and platform-based product development: a state-of-the-art review. Journal of intelligent Manufacturing 18, 5-29.

Jochinke, D.C., Noonon, B.J., Wachsmann, N.G., Norton, R.M., 2007. The adoption of precision agriculture in an Australian broadacre cropping system--Challenges and opportunities. Field Crops Research 104, 68-76.

Kaloxylos, A., Eigenmann, R., Teye, F., Politopoulou, Z., Wolfert, S., Shrank, C., Dillinger, M., Lampropoulou, I., Antoniou, E., Pesonen, L., 2012. Farm management systems and the Future Internet era. Computers and Electronics in Agriculture 89, 130-144.

Kaloxylos, A., Groumas, A., Sarris, V., Katsikas, L., Magdalinos, P., Antoniou, E., Politopoulou, Z., Wolfert, S., Brewster, C., Eigenmann, R., 2014. A cloud-based Farm Management System: Architecture and implementation. Computers and Electronics in Agriculture 100, 168-179.

Kinsey, J.D., 2001. The new Food Economy: consumers, farms, pharms and science. American Journal of Agricultural Economics 83, 1113-1130.

Kruize, J., Wolfert, J., Scholten, H., Verdouw, C., Kassahun, A., Beulens, A., 2016. A reference architecture for Farm Software Ecosystems. Computers and Electronics in Agriculture 125, 12-28.

Kruize, J.W., Robbemond, R.M., Scholten, H., Wolfert, J., Beulens, A.J.M., 2013. Improving arable farm enterprise integration – Review of existing technologies and practices from a farmer's perspective. Computers and Electronics in Agriculture 96, 75-89.

Kruize, J.W., Verdouw, C.N., Wolfert, J., Scholten, H., Beulens, A.J.M., Forthcoming. A reference information model for configuring farm business processes and supporting application services. Forthcomming.

Kruize, J.W., Wolfert, J., Goense, D., Scholten, H., Beulens, A.J.M., Veenstra, T., 2014. Integrating ICT applications for farm business collaboration processes using FIspace, Global Conference (SRII), 2014 Annual SRII. IEEE, San Jose, pp. 232 - 240.

La Rosa, M., Dumas, M., ter Hofstede, A.H., Mendling, J., Gottschalk, F., 2008. Beyond control-flow: Extending business process configuration to roles and objects, Conceptual Modeling-ER 2008. Springer, Berlin, Heidelberg, pp. 199-215.

Lamb, D.W., Frazier, P., Adams, P., 2008. Improving pathways to adoption: Putting the right P's in precision agriculture. Computers and Electronics in Agriculture 61, 4-9.

Lee, J., Siau, K., Hong, S., 2003. Enterprise Integration with ERP and EAI. Communications of the ACM 46, 54-60.

Light, B., Holland, C.P., Wills, K., 2001. ERP and best of breed: a comparative analysis. Business Process Management Journal 7, 216-224.

Lowenberg-DeBoer, J., 2003. Precision farming or convenience agriculture, Proceedings of the 11th Australian Agronomy Conference. Australian Society of Agronomy, Geelong.

Manikas, K., Hansen, K.M., 2013. Software ecosystems–a systematic literature review. Journal of Systems and Software 86, 1294-1306.

March, S.T., Smith, G.F., 1995. Design and natural science research on information technology. Decision Support Systems 15, 251-266.

McBratney, A., Whelan, B., Ancev, T., Bouma, J., 2005. Future Directions of Precision Agriculture. Precision agriculture 6, 7-23.

Messerschmitt, D.G., Szyperski, C., 2003. Software ecosystem: understanding an indispensable technology and industry. MIT Press, Cambridge.

Nash, E., Dreger, F., Schwarz, J., Bill, R., Werner, A., 2009. Development of a model of data-flows for precision agriculture based on a collaborative research project. Computers and Electronics in Agriculture 66, 25-37.

Nikkilä, R., Seilonen, I., Koskinen, K., 2010. Software architecture for farm management information systems in precision agriculture. Computers and Electronics in Agriculture 70, 328-336.

Nuseibeh, B., Easterbrook, S., 2000. Requirements engineering: a roadmap, Proceedings of the Conference on the Future of Software Engineering. ACM, pp. 35-46.

Pedersen, S.M., Fountas, S., Blackmore, B.S., Gylling, M., Pedersen, J.L., 2004. Adoption and perspectives of precision farming in Denmark. Acta Agriculturae Scandinavica, Section B - Plant Soil Science 54, 2 - 8.

Pierce, F.J., Nowak, P., 1999. Aspects of precision agriculture. Advances Agronomy 67, 1-85.

Pierce, F.J., Nowak, P., Donald, L.S., 1999. Aspects of Precision Agriculture. Advances in Agronomy Volume 67, 1-85.

Pohl, K., Böckle, G., van Der Linden, F.J., 2005. Software product line engineering: foundations, principles and techniques. Springer Berlin, Heidelberg.

Porter, M.E., Heppelmann, J.E., 2014. How Smart, Connected Products are transforming competition. Harvard Business Review November 2014, 65-88.

Poteete, A.R., Janssen, M.A., Ostrom, E., 2010. Working together: collective action, the commons, and multiple methods in practice. Princeton University Press, Princeton.

Reichardt, M., Jürgens, C., 2009. Adoption and future perspective of precision farming in Germany: results of several surveys among different agricultural target groups. Precision Agriculture 10, 73-94.

Rettig, C., 2007. The trouble with enterprise software. MIT Sloan Management Review 49, 21-27.

Robbemond, R., Kruize, J.W., 2011. Data standards used for data-exchange of FMIS (44). A research carried out within the Dutch program on precision agriculture. LEI Wageningen, 1-37.

Rosemann, M., van der Aalst, W.M., 2007a. A configurable reference modelling language. Information Systems 32, 1-23.

Rosemann, M., van der Aalst, W.M.P., 2007b. A configurable reference modelling language. Information Systems 32, 1-23.

Schiefer, G., 2004. New technologies and their impact on the agri-food sector: an economists view. Computers and Electronics in Agriculture 43, 163-172.

Scholten, B., 2007. The road to integration: A guide to applying the ISA-95 standard in manufacturing. ISA.

Scholten, H., 2008. Better Modelling Practice: an ontological perspective on multidisciplinary, model-based problem solving. Wageningen University, Wageningen.

Scholten, H., Kassahun, A., Refsgaard, J.C., Kargas, T., Gavardinas, C., Beulens, A.J., 2007. A methodology to support multidisciplinary model-based water management. Environmental Modelling & Software 22, 743-759.

Seichter, D., Dhungana, D., Pleuss, A., Hauptmann, B., 2010. Knowledge management in software ecosystems: software artefacts as first-class citizens, Proceedings of the Fourth European Conference on Software Architecture: Companion Volume. ACM, pp. 119-126.

Seuring, S., Müller, M., 2008. From a literature review to a conceptual framework for sustainable supply chain management. Journal of Cleaner Production 16, 1699-1710.

Shtienberg, D., Bergeron, S., Nicholson, A., Fry, W., Ewing, E., 1990. Development and evaluation of a general model for yield loss assessment in potatoes. Phytopathology 80, 466-472.

Simon, H.A., 1977. The new science of management decision. revised ed. Prentice Hall, Englewood Cliffs, New Jersey.

Skamarock, W.C., Klemp, J.B., Dudhia, J., Gill, D.O., Barker, D.M., Duda, M.G., Huang, X.-Y., Wang, W., Powers, J.G., 2008. A Description of the Advanced Research WRF Version 3. National Center for Atmospheric Research.

Slack, N., Chambers, S., Johnston, R., 2010. Operations Management. 6th ed. FT Prentice Hall, Harlow.

Sommerville, I., 2011. Software Engineering. Pearson, London.

Sørensen, C.G., Fountas, S., Nash, E., Pesonen, L., Bochtis, D., Pedersen, S.M., Basso, B., Blackmore, S.B., 2010a. Conceptual model of a future farm management information system. Computers and Electronics in Agriculture 72, 37-47.

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, 266-276.

Sørensen, C.G., Pesonen, L., Fountas, S., Suomi, P., Bochtis, D., Bildsøe, P., Pedersen, S.M., 2010b. A user-centric approach for information modelling in arable farming. Computers and Electronics in Agriculture 73, 44-55.

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, 238-246.

te Molder, J., van Lier, B., Jansen, S., 2011. Clopenness of systems: The interwoven nature of ecosystems, Third International Workshop on Software Ecosystems (IWSECO-2011), pp. 52-64.

TheOpenGroup, 2011. TOGAF Version 9.1. First edition ed. Van Haren Publishing, Zaltbommel.

TheOpenGroup, 2012. ArchiMate 2.0 Specification. The Open Group, Reading.

Tilman, D., Cassman, K.G., Matson, P.A., Naylor, R., Polasky, S., 2002. Agricultural sustainability and intensive production practices. Nature 418, 671-677.

Trienekens, J., Wognum, P., Beulens, A.J., van der Vorst, J.G., 2012. Transparency in complex dynamic food supply chains. Advanced Engineering Informatics 26, 55-65.

Turner, M., Budgen, D., Brereton, P., 2003. Turning software into a service. Computer 36, 38-44.

United, S., Congress, House, Committee on, S., Astronautics, Subcommittee on Advanced, R., Technology, 1969. Aeronautical Research : hearings before the United States House Committee on Science and Astronautics, Subcommittee on Advanced Research and Technology, Ninety-First Congress, first session, on Dec. 1, 2, 4, 8-11, 1969. U.S. G.P.O., Washington.

Van 't Spijker, A., 2014. The new oil - using innovative business models to turn data into profit. Technics Publications, Basking Ridge.

van der Aalst, W.M.P., Dreiling, A., Gottschalk, F., Rosemann, M., Jansen-Vullers, M.H., 2006. Configurable Process Models as a Basis for Reference Modeling, Business Process Management Workshops, pp. 512-518.

Van Wezel, W., Van Donk, D.P., Gaalman, G., 2006. The planning flexibility bottleneck in food processing industries. Journal of Operations Management 24, 287-300.

Verdouw, C., Beulens, A., Van Der Vorst, J., 2013. Virtualisation of floricultural supply chains: A review from an Internet of Things perspective. Computers and Electronics in Agriculture 99, 160-175.

Verdouw, C., Beulens, A., Wolfert, J., 2014. Towards software mass customization for business collaboration, Global Conference (SRII), 2014 Annual SRII. IEEE, San Jose, pp. 106-115.

Verdouw, C., Robbemond, R., Wolfert, J., 2015. ERP in agriculture: Lessons learned from the Dutch horticulture. Computers and Electronics in Agriculture 114, 125-133.

Verdouw, C.N., Beulens, A.J.M., Trienekens, J.H., Verwaart, T., 2010a. Towards dynamic reference information models: Readiness for ICT mass customisation. Computers in Industry 61, 833-844.

Verdouw, C.N., Beulens, A.J.M., Trienekens, J.H., Wolfert, J., 2010b. Process modelling in demand-driven supply chains: A reference model for the fruit industry. Computers and Electronics in Agriculture 73, 174-187.

Verdouw, C.N., Wolfert, J., 2009. Reference process modelling in demand-driven agri-food supply chains: A configuration-based approach, In: Simons, A., Trienekens, J.H., Van der Vorst, J., Top, J.L., Beulens, A.J.M. (Eds.), Agri-Food Supply Chains and Logistics. Wageningen Academic Publishers, Wageningen.

Vernadat, F.B., 2007. Interoperable enterprise systems: Principles, concepts, and methods. Annual Reviews in Control 31, 137-145.

WHO, 1990. Public health impact of pesticides used in agriculture. (WHO in collaboration with the United Nations Environment Programme, Geneva, 1990).

Wiederhold, G., 1992. Mediators in the architecture of future information systems. Computer 25, 38-49.

Wolfert, J., 2002. Sustainable agriculture: how to make it work? : a modeling approach to support management of a mixed ecological farm. Wageningen UR, Wageningen, p. 278.

Wolfert, J., Sørensen, C.G., Goense, D., 2014. A future internet collaboration platform for safe and healthy food from farm to fork, Global Conference (SRII), 2014 Annual SRII. IEEE, San Jose, pp. 266 - 273.

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, 389-405.

Yin, R.K., 2009. Case study research: Design and methods. Sage, Thousand Oaks

Zanden, J.L.v., 1991. The First Green Revolution: The Growth of Production and Productivity in European Agriculture, 1870-1914. The Economic History Review 44, 215-239.

Appendices

Appendix A – Ontology

Table 18: First version of the ontology (References of the definitions can be found on: http://tinyurl.com/gwbpdco).

Concept	Definition
Active Structure element	An entity that is capable of performing behaviour.
Actor	An organizational entity that is capable of performing behaviour.
Application Interface	An *Interface* where an *Application Service,* as part of an *Application Component,* is made available to an *End-User* or another *Application Component*.
Application *Service*	A *Service* that exposes automated behaviour.
Architecture	The fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution
Artefact	An object made by a human being to support a software engineering process.
Behavioural element	A unit of activity performed by one or more *active structure elements.*
Business Process	A B*ehavioural Element* that groups behaviour based on an ordering of activities. It is intended to produce a defined set of *Products* or *Business Services*.
Business System	*Information System*, *Business Processes*, Policy Statements, Activities, *Standards*, and People which together implement a *Function.*
Component	*Component* is a part or element of a larger whole.

Table 18: Continued.

Concept	**Definition**
Contract	A formal or informal specification of agreement between actors that are part of different legal entities that specifies the rights and obligations associated with a *Product.*
End-User	Actors who ultimately uses *ICT Products* or *Services* to support their *Business Processes.*
Farm Enterprise	A *Farm Enterprise* is a *Business Actor* which has agricultural production as its main activity.
Farm *Information System*	An *Information System* that is part of a *Business System* that is part of a *Farm Enterprise.*
Farm Management *Information System*	A *Farm Management Information System* (FMIS) is an *ICT Component,* consisting out of one or more *Application Components*, for collecting, processing, storing and disseminating of data in the form of information needed to carry out the operational functions of the farm.
Farm Software Ecosystem	A *Software Ecosystem* that has *Farm Enterprises* and its collaborating *Actors* as *End-Users.*
Function	An intentional activity of a *System.*
Implementation	The process to create an *Instance*.
Information *System*	A computer based system to support organizational processes of an Actor.
Instance	A realization of an *Artefact* in its environment.
Product	A coherent collection of *Services* or *Artefacts*, accompanied by a C*ontract* and usage documentation which is offered as a whole to customers.
Service	A unit of functionality that a system exposes to its environment, while hiding internal operations, which provides a certain value (monetary or otherwise).

Table 18: Continued.

Concept	**Definition**
Software Ecosystem	The interaction of a set of Actors on top of a common technological Platform that result in a coherent set of *ICT Products* or *Services.*
Software Producing Organization	A *Business Actor* developing *ICT Components.*
System	A collection of *Components* organized to accomplish a specific Function or set of functions.
Artefact	**Definition**
Application *Component*	A *modular* deployable, and replaceable part of an *Information System* that encapsulates its behaviour and data and exposes these through a set of *Interfaces*. Its *instance* that is deployed on a *Device* is named an *ICT Component.*
Atomic Application *Component*	An Atomic *Application Component,* deployed on a *Device,* is part of an *Information System* however not able to share data automatically with other *Application Components.*
Composite Application *Component*	A composition of *Atomic Application Components*, which is created by ICT Customization using a configuration process (Verdouw et al., 2010), that performs collective behaviour by exchanging data automatically between the Atomic Application Components and which can be deployed on a *Device*.
Communication Path	A communication path is defined as a link between two or more *Nodes*, through which these *Nodes* can exchange data.
Data	A digital representation of an object
Data Structure	The data structure describes the *data* and is the so called metadata
Device	A *Hardware* resource upon which *Application Components* may be stored or deployed for execution.

Table 18: Continued.

Artefact	**Definition**
ICT *Component*	An *ICT Component* is an *Application Component (Composite Application Component* or *Atomic Application Component)* that is deployed on a *Node* and which supports one or more *Business Processes* of a Business *Actor*. A *ICT Component* can act as an *Farm Information System* or be part of an *Farm Information System.*
ICT *Product*	A Product including a *Contract*, *ICT Component,* Configuration support and usage support that is offered by an *Actor.*
Network	A network is defined as a communication medium between two or more *Devices*.
Node	A computational resource upon which *Application Components* may be stored or deployed for execution.
Platform	A set of stable *Components* that supports variety and evolution in a system by constraining the linkages among the other components. These components include hardware, software and service modules, along with an architecture that specifies how they fit together.
Process group	A process group is defined as a behavioral element that groups behavior. In ArchiMate a process group in named a Business Function. Examples are inventory management, product management and accounting
Software Ecosystem Instance	A real-world and operational *Software Ecosystem.*
Standard	A *Standard* is something used as a measure, norm, or *Model* in comparative evaluations. There are four standards classifications in ICT Development: Business Standards, Data Standards, Application Standards and Technology Standards.
Farm Specific Concept	**Definition**
Batch	A *Batch* is a volume of a *Product* or *Produce*, which is kept separate from other batches.

Table 18: Continued.

Farm Specific Concept	**Definition**
CulturalPractice	A *CulturalPractise* describes one or series of activities to realize an objective in crop production. Examples are primary soil tillage, planting, harvesting, etc.
Device Allocation	*DeviceAllocation* describes the period during which a Device is (planned to be) used for a (planned) *Task*
Field	A *Field* is a surface of land which is bordered by land of an other user, Ditches, Canals, roads or a strip of land what is not allowed to be cultivated. It is the maximum piece of land that can be used to cultivate a crop. The *Field* can stay the same over more years
Governmental manure allocation plan	The allocation plan of the manure required by the government
Guidance Reference Line	This is a line in or outside a *PartField* that is used as a reference for Track 's
Job	A Job describes what *Task* and the order (*Route*) of the *Tasks* to be executed.
Operation	An *Operation* performs a particular *OperationTechnique* on a particular *PartField*, *CropProductionUnit* or *Batch* to realise a certain *CulturalPractise*.
Operation Technique	An *OperationTechnique* is the means by which a *CulturalPractise* can be realized.
Operator	A *Operator* is a person that performs activities (*Task's*) on a Farm
Operator Allocation	OpeartorAllocation defines a time period that a work role is (planned to be) active in a Task.
Person Operator Type	A Person Operatator Type defines what kind of task or operation a Operator is able to execute
Product Management	A *Process group* including *Business processes* that are related to the production and preparation of vegetable products for internal or external customers
Route	The order in which the *fields tasks* need to be executed

Table 18: Continued.

Farm Specific Concept	**Definition**
Task	A Task is the execution, within a 'part *field*', of one or more *Operations*
Work-Order	A *Work-Order* specifies what resources are used within a specified time for a specific crop to achieve a certain goal. A Work-Order consists of one or more *Jobs* that are allocated to a *Operator*.
Work Order DateTime	The time moment or the time period over which the Work-Order is valid.
Information objects in RAAgE	**Description**
Crop cultivation plan	The crop cultivation plan describes what crop/crop variety requires what resources at what place at which moment to meet the requirements of sales plan.
Crop processing plan	The plan that describes how the crop will be processed.
Crop distribution plan	A plan describing what crops will be grown on what fields in a certain time period (Example of a sub-plan is the potato distribution plan: Plan that describes the fields on which potatoes are grown for a specific year)
Crop plan	A plan describing what crops will be grown on what fields in a certain time period
Crop protection plan	The plan that describes how the crop will be protected against different threats.
Fertilization plan	The fertilization plan describes what minerals will be allocated to what field
Field category	The field category describes the physical or historical conditions of the field
Field inspection list	A list describing what *Field* inspections should be executed to acquire information about the *Field*
Field Monitoring plan	The monitoring plan describes what method is used to inspect the state of the crop

Table 18: Continued.

Information objects in RAAgE	**Description**
Historic field data	Historic data about the F*ield* acquired by different *Devices*
Inspections list	A list that describes what information is required from the field or crop during the cultivate crop process
Irrigation plan	The irrigation plan describes how the crop will be irrigated
Job execution information	Information about the execution of the job
Job instruction	The instruction for a specific operator.
Management control plan	The management control plan describes strategies enterprises will use to achieve its goals
Operation control plan	The operation control plan describes the operations to efficient and effective perform individual tasks
Process crop	Processing of a crop as preparation for an internal or external customer
Processed task information	Cultivation information contains checked information about the cultivation operation. This can contain information about the tate of the crop or fields or the operation
Product Management plan	The product management plan describes the operations to produce the arable farm products (Produce).This plan describes both the cultivation as the processing plan.
Seeding plan	A plan describing what seed in sown at what place with what planting distance
Task and operations list	A list of the task and operations that are needed to cultivate the specific crop of the field.
Task benchmark data	Benchmark data about a certain cultural practice
Tillage plan	The tillage plan describes what tillage operation will be executed on the field.

Appendix B - Requirements Table Farm Software Ecosystem

Table 19: Requirements and design verifications.

Category	#	Requirement	Design verification
Data handling and seamless data exchange	1.1	Enable message (data) exchange between distributed systems	Message exchange is address in the architectures part: • Open Software Enterprise, specifically R&D ○ Stimulates to use standards and the development of an Information model ○ Defining API's • Platform ○ Stimulates linking Application Components (configuration) ○ Proposing System and Data integration module • ICT Components ○ Describe configuration
	1.2	Provide technical and semantic standards to facilitate data exchange	Providing technical and semantic standards is done in the architecture part: • Open Software Enterprise, R&D; ○ Stimulating the use of existing standards ○ Defining API's

Table 19: Continued.

Category	#	Requirement	Design verification
	1.3	Enable the re-use of data	The re-use of data is addressed in the architecture part: • Open Software Enterprise, specifically R&D and Governance ○ Stimulates to use standards and the development of an Information model ○ Defining API's ○ Provide IP Strategy Documentation • Platform ○ Stimulates linking Application Components (configuration) ○ Proposing System and Data integration module • ICT Components ○ Describe configuration
	1.4	Stimulate the use of existing standards	Stimulating the use of standards is addressed in the architecture parts: • Open Software Enterprise, specifically R&D: ○ Stimulates to use standards and the development of an Information model
Configuration of ICT Components	2.1	Deliver customized ICT Component configurations	Customisation of ICT Components is addressed in all the architecture parts: • Open Software Enterprise • ICT Components • Platform • Actor descriptions • Business Services
	2.2	Provide artefacts supporting a multi-vendor software development	This is addressed in the part about the Open Software Enterprise; the artefacts to stimulate collaboration are listed.

Table 19: Continued.

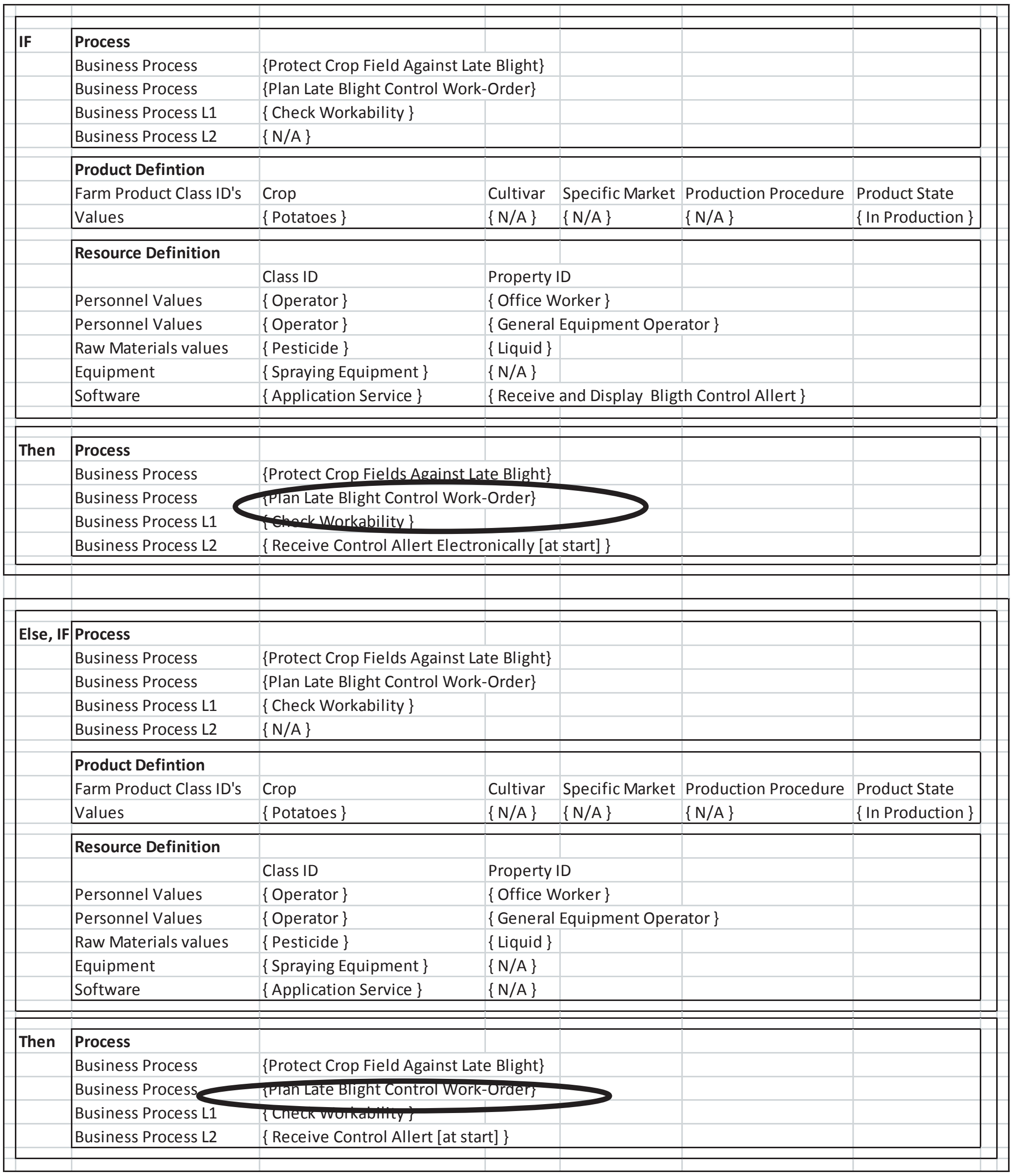
Category	#	Requirement	Design verification
	2.3	Develop integrated systems with a coherent User Interface regarding the look and feel	The development of integrated systems in addressed in the parts about: • the Open Software Enterprise • Platform • ICT Components
	2.4	Enable software modularity	Software modularity is addressed in the parts about: • the Open Software Enterprise • Platform • ICT Components
	2.5	Provide an information integration platform	This is addressed in the part about the Platform
	2.6	Component availability	This is addressed in the parts: • Open Software Enterprise, Marketing and sales where is described how this can stimulated actors to join a Software Ecosystem
	2.7	Configuration support	This is addressed in the part • Business Services • Actors • Open Software Enterprise, Consulting and Support Services
	2.8	Reference Information model	This is addressed in the part: • Open Software Enterprise, R&D
Interoperability of ICT Components	3.1	Provide meta data about Application Components (e.g. performance etc.)	This is addressed in the part: • ICT Components • Open Software Enterprise, Product Management
Open Software Enterprise's	4.1	Provide an Open Software Enterprise	This is addressed in the part: • Open Software Enterprise

Table 19: Continued.

Category	#	Requirement	Design verification
	4.2	Provide hosting for the platform	This is addressed in the part: • Actors, by describing the different roles actors can take in a Farm Software Ecosystem
	4.3	Enable collaboration between multiple vendors	This is addressed in all parts of the reference architecture.
	4.4	Enable competition between actors within Farm Software Ecosystems	This is addressed in the part: • Open Software Enterprise, ○ Governance, proving a Partnership Model
	4.5	Stimulate innovation	This is addressed in the part: • Open Software Enterprise, ○ Governance, by proving a Partnership Model and IP strategy Information ○ Research and development • Software Product Management

Appendix C - Business Rules RAAgE 2.0

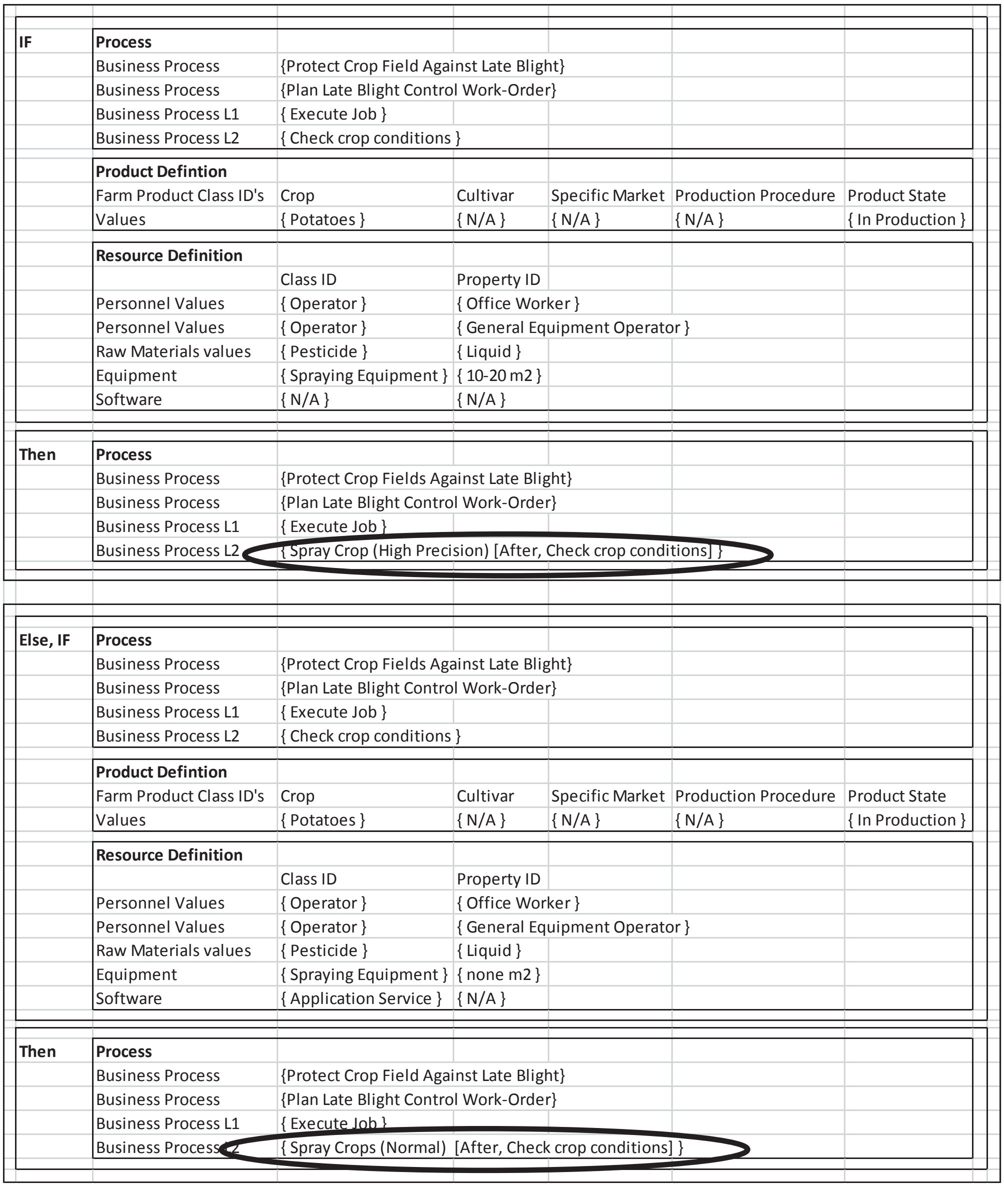
To enable configuration of business processes at a L1 and L2 (Step 4 and 5 of the configuration tree) business rules should be executed. For each business process a business rule determines if it can be part of a set of business process that can together realize a business service (e.g. crop protection). The business rules describe the relation between the business process, product definition and resource definition. If certain criteria are fulfilled a process can become part of the set of business processes. This is described with the criteria in the IF section (see Figure 61). When these criteria are fulfilled the THEN section describes what process is added to the set of business processes and after what previous process it will be placed. If a criteria is not fulfilled the ELSE, IF section is executed.



IF	**Process**					
	Business Process	{Protect Crop Field Against Late Blight}				
	Business Process	{Plan Late Blight Control Work-Order}				
	Business Process L1	{ Check Workability }				
	Business Process L2	{ N/A }				
	Product Defintion					
	Farm Product Class ID's	Crop	Cultivar	Specific Market	Production Procedure	Product State
	Values	{ Potatoes }	{ N/A }	{ N/A }	{ N/A }	{ In Production }
	Resource Definition					
		Class ID	Property ID			
	Personnel Values	{ Operator }	{ Office Worker }			
	Personnel Values	{ Operator }	{ General Equipment Operator }			
	Raw Materials values	{ Pesticide }	{ Liquid }			
	Equipment	{ Spraying Equipment }	{ N/A }			
	Software	{ Application Service }	{ Receive and Display Bligth Control Allert }			
Then	**Process**					
	Business Process	{Protect Crop Fields Against Late Blight}				
	Business Process	{Plan Late Blight Control Work-Order}				
	Business Process L1	{ Check Workability }				
	Business Process L2	{ Receive Control Allert Electronically [at start] }				
Else, IF	**Process**					
	Business Process	{Protect Crop Fields Against Late Blight}				
	Business Process	{Plan Late Blight Control Work-Order}				
	Business Process L1	{ Check Workability }				
	Business Process L2	{ N/A }				
	Product Defintion					
	Farm Product Class ID's	Crop	Cultivar	Specific Market	Production Procedure	Product State
	Values	{ Potatoes }	{ N/A }	{ N/A }	{ N/A }	{ In Production }
	Resource Definition					
		Class ID	Property ID			
	Personnel Values	{ Operator }	{ Office Worker }			
	Personnel Values	{ Operator }	{ General Equipment Operator }			
	Raw Materials values	{ Pesticide }	{ Liquid }			
	Equipment	{ Spraying Equipment }	{ N/A }			
	Software	{ Application Service }	{ N/A }			
Then	**Process**					
	Business Process	{Protect Crop Field Against Late Blight}				
	Business Process	{Plan Late Blight Control Work-Order}				
	Business Process L1	{ Check Workability }				
	Business Process L2	{ Receive Control Allert [at start] }				

Figure 61: Business Rule for the Activity Receive Control Alert Electronically or Receive Control Alert.

The business rule presented in Figure 61 determines if the business process L2 'Receive Control Alert Electronically' or the 'Receive Control Alert' should be part of the Business Process Protect Crop Field (see the circles in the figure). The business rule presented in Figure 62 determines if the L2 business process “Spray Crop (High Precision)” or “Spray Crop (Normal)” becomes part of the set of business processes.



IF	**Process**					
	Business Process	{Protect Crop Field Against Late Blight}				
	Business Process	{Plan Late Blight Control Work-Order}				
	Business Process L1	{ Execute Job }				
	Business Process L2	{ Check crop conditions }				
	Product Defintion					
	Farm Product Class ID's	Crop	Cultivar	Specific Market	Production Procedure	Product State
	Values	{ Potatoes }	{ N/A }	{ N/A }	{ N/A }	{ In Production }
	Resource Definition					
		Class ID	Property ID			
	Personnel Values	{ Operator }	{ Office Worker }			
	Personnel Values	{ Operator }	{ General Equipment Operator }			
	Raw Materials values	{ Pesticide }	{ Liquid }			
	Equipment	{ Spraying Equipment }	{ 10-20 m2 }			
	Software	{ N/A }	{ N/A }			
Then	**Process**					
	Business Process	{Protect Crop Fields Against Late Blight}				
	Business Process	{Plan Late Blight Control Work-Order}				
	Business Process L1	{ Execute Job }				
	Business Process L2	{ Spray Crop (High Precision) [After, Check crop conditions] }				
Else, IF	**Process**					
	Business Process	{Protect Crop Fields Against Late Blight}				
	Business Process	{Plan Late Blight Control Work-Order}				
	Business Process L1	{ Execute Job }				
	Business Process L2	{ Check crop conditions }				
	Product Defintion					
	Farm Product Class ID's	Crop	Cultivar	Specific Market	Production Procedure	Product State
	Values	{ Potatoes }	{ N/A }	{ N/A }	{ N/A }	{ In Production }
	Resource Definition					
		Class ID	Property ID			
	Personnel Values	{ Operator }	{ Office Worker }			
	Personnel Values	{ Operator }	{ General Equipment Operator }			
	Raw Materials values	{ Pesticide }	{ Liquid }			
	Equipment	{ Spraying Equipment }	{ none m2 }			
	Software	{ Application Service }	{ N/A }			
Then	**Process**					
	Business Process	{Protect Crop Field Against Late Blight}				
	Business Process	{Plan Late Blight Control Work-Order}				
	Business Process L1	{ Execute Job }				
	Business Process L2	{ Spray Crops (Normal) [After, Check crop conditions] }				

Figure 62: Business Rule for the Activity Spray Crop (High Precision) or Spray Crop (Normal).

Summary

Since time began, mankind has been threatened by the combination of growing populations and diminishing resources. Present-day, this threat is very pertinent as mankind is challenged by a growing world population that is expected to exceed 10 billion in 2050, while resources diminish. Simultaneously, increase of food production should be accomplished in a sustainable manner as consumers require food to be produced environmentally-friendly. Moreover, consumers require safe food produced in transparent agri-food supply chain networks. Farm enterprises can contribute by advancing their management to increase food production in a sustainable, safe and transparent manner. A well-known advanced farm management style, which is knowledge and information intensive, is precision agriculture. Precision agriculture increases the profitability of crop production, while simultaneously reducing the negative environmental impact by tight monitoring and control, in which applications rates of agricultural inputs are adjusted to local needs. Such advanced farm management requires integrated farm information systems as it is knowledge and information intensive. However, advancement is hindered because of interoperability issues between software systems of multiple vendors. An integrated farm information system, containing components of multiple vendors, is required as single organisations cannot develop all technical solutions and ICT Components (e.g. tractors, implements, FMIS, decision support tools) that farmers require. A global overarching system, developed by a single vendor, that can support all business functions of farmers is therefore neither a feasible nor, from a competitive point of view, a desirable solution in agriculture. To realize farm enterprise integration we combine the approaches ICT Mass Customisation with Best-of-Breed. ICT mass customisation combines advantages of standard and customised software by enabling on-demand configuration of information systems from standard components with standardised interfaces. These ICT components can be supplied by different software vendors, which allow Best-of-Breed solutions. By realization of these approaches farm enterprise integration can improve. A farm enterprise can be an arable farm, livestock farm or horticultural farm. In this thesis we focus on arable farm enterprises.

To enable farm enterprise integration we have developed six artefacts that are presented in this thesis which are:

(1) The Reference Architecture of Agricultural Enterprises (RAAgE) 1.0 that can describe farm enterprise architectures in a uniform and efficient manner;
(2) A problem description, which is a case specific instantiation of RAAgE 1.0 generalized to a generic problem description;
(3) An ontology that supports communication between collaborating actors and components;
(4) Reference Architecture for Farm Software Ecosystems that defines generic relationships between actors and components;
(5) RAAgE 2.0 that is a technical reference model to support configuration of business processes and ICT components, which is based on RAAgE 1.0;
(6) Prototype software that serves as a proof of concept substantiating that all previous components will provide a solution for integration problems at farm enterprises.

RAAgE 1.0 supports designing enterprise architectures in a uniform and efficient manner. The reference model is described in a standard modelling language, named ArchiMate, and shows the interrelations between the business, application and technology layers of farm enterprises. The reference model includes an ontology to provide a concise and precise, formal specification of the object system. This is required to have a shared understanding and effective communication between researchers, farmers, software developers and other stakeholders involved. This ontology is used and extended in other parts of our research. The architectural descriptions can depict the relations between farm business processes and the ICT Components used. The model is validated by two experts that have experience in developing reference architectures and models.

A detailed problem description is created using RAAgE 1.0 to gain insight in the cause and nature of integration problems at farm enterprises. To find these problems a method was developed and applied in a case study research including three arable farm enterprises producing potatoes. These farm enterprises focused on improving their management and invested in new technologies for innovation. Within multiple steps of the method the architectural descriptions developed with RAAgE 1.0 facilitated communication and provided insight into problems of farm enterprises to achieve more advanced farm management. The case specific problems, described by instantiating RAAgE 1.0, have been analysed and formulated as more generic problems for farm enterprise integration. These generic problem descriptions have been validated with national and international experts. Based on this research we found that the cause and nature of current integration problems in farming are that ICT components used within the same farm enterprise:

(i) have partly overlapping and partly unique application services, functions and interfaces (that are non-standard);
(ii) are missing required application services, functions and interfaces,
(iii) have disjoint data repositories;
(iv) have inadequate and incomplete data exchange as semantics are not unambiguously defined;
(v) are hard to configure while this configuration is not supported by an actors and tools.

A design, addressing these problems is expected to solve current integration bottlenecks. First, this design must enable smooth data handling and seamless data exchange between ICT Components to solve inadequate and incomplete data exchange and enable integration of data repositories of multiple vendors. Second, it must include a configuration approach to link ICT Components to each other in a meaningful and coherent way. This should be supported by actors that are willing to configure ICT Component of multiple vendors into an integrated solution. Third, the design must enable the formation of a software enterprise to address the previous points and to organize collaboration between actors involved. This software enterprise should focus both on improving interoperability to contribute in solving problems with partly overlapping and partly unique application services, functions and interfaces as well as on organizing the development of missing application services, functions and interfaces.

To address these integration challenges a Reference Architecture for Farm Software Ecosystems and RAAgE 2.0 were developed, focusing on both technical and organizational aspects.

From literature we found that collaboration can take place within Software Ecosystems. Software Ecosystems are defined as the interaction of a set of actors on top of a common technological platform that results in a coherent set of ICT components or Services. They can provide an effective way to construct large software systems on top of a software platform by combining components, developed by actors that are part of different organisations. To support instantiation of Software Ecosystems for farming, a Reference Architecture was developed. This Reference Architecture describes how software developers, farmers and other stakeholders can collaborate to enable development, configuration and instantiation of integrated software solutions. More specifically, it can be used to map, assess, design and implement Farm Software Ecosystems to help to decrease current integration problems. The reference architecture comprises five main components:

(i) **Actors**, which are basically app developers, business architects/software developers and end-users, *i.e.* farmers that finally use the configured ICT components and services;
(ii) **Platform** that enables configuration of Atomic Application Components into integrated information systems for farmers;
(iii) **Open software enterprise** that manages the relation between the actors and the platform;
(iv) **Business services** that support software configuration, development and hosting;
(v) **ICT Components** that are configured application components from multiple vendors allowing seamless data exchange based on standards

After the design the reference architecture was first verified based on the requirements. Second, semi-structured interviews were held with experts to validate the model. Moreover, the assessment and mapping functionally was validated by using the reference architecture in a case study, in which two existing farm software ecosystems were assessed and mapped.

The Reference Architecture for Farm Software Ecosystems mainly addresses the organizational part of this research question. The technical part on the configuration of different ICT components into integrated solutions was not yet sufficiently covered in the Reference Architecture for Farm Software Ecosystems. Therefore we designed RAAgE 2.0 to improve the integrating capabilities of ICT Components, focussing on configuration and ICT Mass Customisation. In this research RAAgE 1.0 was extended into RAAgE 2.0 supporting technical aspects related to configuration of ICT Components by providing a hierarchical configuration methodology. This methodology divides configuration in two steps (i) business process configuration and (ii) software configuration. To enable business process configuration the model comprises three reference models, i.e. on products, processes and resources. The dependencies between these models are defined in rules that define possible combinations of products, processes and resources and that constrain the configuration of farm-specific models i.e. instances. The reference model also includes a configuration tree and templates. Templates describe a set of pre-configured product, process and resource models for typical cases. Variety in farm business processes can be modelled with business process variants. Such a variant realizes a similar kind of business services (e.g. basic fertilization, precision fertilization). Each variant has partly overlapping business processes and resources and unique ones. RAAgE 2.0 provides insight into these specific and generic parts. The other part of the methodology, software configuration, is divided in two additional sub-steps. The first sub-step is to create configuration templates that describe the required (generic) application services (capability types) to support specific business process variants. These configuration templates describe the interactions between the capability types. This sub step is typically performed by a business architect in close collaboration with software developers. The second sub-step is the selection and configuration of the specific capability of a capability type. Capabilities can be offered by atomic application components of multiple vendors that need to be selected. This second sub-step is performed by a business architect, in close collaboration with a farmer. With this extension RAAgE 2.0 supports (i) development of ICT components that fit within an ICT Mass Customisation and Best-of-Breed approach, (ii) selection of ICT components based on business processes that they should support and (iii) getting insight into configuration of different ICT components into an integrated farm information system.

To substantiate that our artefacts contribute to realizing ICT Mass Customisation in combination with Best-of-Breed in arable agriculture a proof of concept was developed. A proof of concept is defined as a phase in development, in which experimental hardware or software is constructed and tested to explore and demonstrate the feasibility of a new concept. Realizing ICT Mass Customisation requires: (i) software modularity, (ii) an information integration platform, (iii) component availability, (iv) configuration support and (v) reference information models. To fulfil these requirements a design was developed and instantiated for a specific use case on late blight protection in potato growing for a specific farmer in The Netherlands. For that purpose we:

(i) configured the business processes that are involved in late blight protection using RAAgE 2.0 to identify which advanced ICT components are needed to support this process for this farmer;
(ii) developed the required advanced ICT components that were identified in the previous step using the FIspace platform. These components were provided by different app developers from 5 different European countries;
(iii) configured a composite application component within the FIspace platform using the configuration framework of RAAgE 2.0. This included involvement of 5 different European organizations;
(iv) instantiated and executed the application component within the FIspace platform for this specific farmer.

This resulted in prototype software that showed how we can configure business processes and multi-vendor atomic application components into a composite component to support late blight protection in potatoes for a specific farmer. It was made plausible that this approach is also applicable to other cases to create software able to support other business processes in agriculture.

Within this research we developed artefacts and substantiated that they facilitate collaboration between the actors involved and can help to develop ICT Components that improve farm enterprise integration. Still, to make ICT Mass Customisation and Best-of-Breed a more common practice, future research is required. In this research we recommend to focus on:

(1) Development of business models to gain insight into the motives of software developers to become part of Farm Software Ecosystems. Insight into these motives can enhance the adoption of Software Ecosystems for agriculture, which makes the concept of ICT Mass Customisation more feasible.

(2) Improving configuration of atomic application components and supporting tools as this is currently still cumbersome. We recommend focussing on one specific case to dig into all details of the case. Such a detailed description will be re-usable for many other farm business processes such as fertilization, other types of crop protection, seeding and harvesting.

Although, there are still hurdles to take we recommend continuing this research line as it can result in improved farm enterprise integration and adoption of advanced farm management styles by famers. This can enable farm enterprises to increase food production, while producing in a sustainable, safe and transparent manner.

Samenvatting

De groeiende wereldbevolking en de afnemende natuurlijke bronnen vormen een bedreiging voor de mensheid. Op dit moment is het de verwachting dat de wereldbevolking in 2050 ongeveer uit 10 miljard mensen zal bestaan. Om deze groeiende bevolking te kunnen blijven voeden moet er de komende jaren meer voedsel worden geproduceerd, maar moet dit ook op een duurzame, veilige en transparante manier gebeuren. De landbouw moet hieraan bijdragen door productieprocessen aan te passen en te verbeteren. Een management methode die een hogere productie met minder middelen realiseert wordt precisielandbouw genoemd. Precisielandbouw kan opbrengsten verhogen, terwijl tegelijkertijd de negatieve milieueffecten verminderen, door de input aan te passen aan de plaatselijke behoeften binnen het veld. Om deze management methode te realiseren moet er kennis zijn over het veld, gewas en de plaatsspecifieke eigenschappen. Om deze kennis te genereren en plaatsspecifiek te kunnen sturen moeten (software-) componenten en databronnen van verschillende leveranciers met elkaar geïntegreerd en gecombineerd kunnen worden. Voorbeelden van softwarecomponenten zijn bedrijfsmanagementsystemen, beslissingsondersteunende systemen, apps, en softwaresystemen op trekkers en werktuigen. Het integreren en combineren van softwaresystemen en databronnen is echter complex en hindert adoptie van precisielandbouw. Om de adoptie te verbeteren is een bedrijfsmanagementsysteem nodig dat bestaat uit softwarecomponenten van meerdere leveranciers. Een dergelijk systeem is nodig omdat (i) het niet te verwachten is dat een enkele organisatie alle benodigde softwarecomponenten kan ontwikkelen die nodig zijn voor een wereldwijde optimale boerenbedrijfsvoering en (ii) een systeem van een enkele leverancier een niet wenselijke oplossing is vanuit concurrentieoverwegingen en marktwerking. Om deze redenen wordt er in dit onderzoek gewerkt aan de realisatie van een bedrijfsmanagementsysteem dat bestaat uit softwarecomponenten van verschillende leveranciers. Om dit te realiseren baseren wij ons op de aanpak 'ICT Mass Customization' en 'best-of-breed'.

ICT Mass Customization combineert de voordelen van standaard en klant specifieke softwaresystemen door configuratie van modulaire softwarecomponenten mogelijk te maken. Hierdoor ontstaan er klant specifieke systemen die toch betaalbaar blijven. Deze modulaire softwarecomponenten moeten geleverd kunnen worden door verschillende softwareleveranciers waardoor er best-of-breed oplossingen ontstaan. Met een bedrijfsmanagementsysteem dat bestaat uit softwarecomponenten van verschillende leveranciers kunnen bedrijfsprocessen in de landbouw beter worden ondersteund. Dit proefschrift richt zich specifiek op dergelijke systemen voor akkerbouwbedrijven.

Om bij te dragen aan de realisatie van softwaresystemen gebaseerd op ICT Mass Customisation in combinatie met best-of-breed zijn er in dit proefschrift zes artefacten ontwikkeld:

(1) Een referentie architectuur voor het modeleren van akkerbouwbedrijven genaamd (RAAgE) 1.0;
(2) Een probleembeschrijving, met behulp van RAAgE 1.0, die de huidige integratie problematiek beschrijft;
(3) Een ontologie die de communicatie tussen samenwerkende partijen kan ondersteunen;
(4) Een referentie architectuur voor Farm Software Ecosystems die relaties tussen verschillende betrokken actoren beschrijft;
(5) RAAgE 2.0 waarmee configuratie van bedrijfsprocessen en ICT-componenten beschreven wordt;
(6) Prototypesoftware waarmee wordt aangetoond hoe verschillende modulaire softwarecomponenten geïntegreerd kunnen worden.

RAAgE 1.0 is een referentiemodel en ondersteunt het modeleren van bedrijfsprocessen en ondersteunende software en hardware op een uniforme en efficiënte manier. Het referentiemodel wordt beschreven in de standaard modeleringstaal genaamd ArchiMate 2.0. Het referentiemodel bevat een ontologie om objecten binnen een akkerbouwbedrijf met behulp van een specificatie te modeleren. Modellen van akkerbouwbedrijven zijn nodig voor een gemeenschappelijk begrip en voor effectieve communicatie tussen onderzoekers, landbouwers, softwareontwikkelaars en andere betrokkenen. RAAgE 1.0 is gevalideerd door twee experts met ervaring in het ontwikkelen van referentie architecturen en modellen.

RAAgE 1.0 is gebruikt voor het opstellen van een gedetailleerde probleembeschrijving. Deze probleembeschrijving geeft inzicht in de oorzaak en aard van integratieproblemen, met betrekking tot ICT, in de akkerbouw. Om deze problemen in kaart te brengen is er een case studie onderzoek uitgevoerd. In deze case studie zijn drie akkerbouwbedrijven betrokken die aardappelen produceren. Al deze akkerbouwbedrijven zijn er op gericht om hun bedrijfsprocessen te verbeteren door te investeren in nieuwe technologieën. RAAgE 1.0 is gebruikt om modellen te ontwikkelen die inzicht geven in de bedrijfsprocessen en de ondersteunende software en hardware. Deze modellen geven inzicht in de integratieproblemen van de individuele akkerbouwbedrijven. Deze specifieke problemen zijn geanalyseerd en als meer generieke problemen beschreven. Deze generieke probleembeschrijvingen zijn gevalideerd met behulp van nationale en internationale experts. Op basis van het onderzoek is naar voren gekomen dat de akkerbouwers de volgende integratie problemen hebben:

(i) Softwaresystemen hebben deels overlappende en deels unieke softwarefuncties en interfaces;
(ii) er ontbreken softwarefuncties en interfaces,
(iii) verschillende softwaresystemen hebben verschillende databases;
(iv) informatie-uitwisseling tussen de softwaresystemen is ontoereikend doordat de beschrijving van de data niet helder is;
(v) Softwaresystemen van verschillende leveranciers zijn moeilijk te configureren waardoor het lastig is om een geïntegreerd softwaresysteem te realiseren.

Om deze integratie problemen te verhelpen heeft dit onderzoek zich gericht op zowel organisatorische als technische aspecten om integratie te verbeteren. Daarom is er als eerste een organisatorisch model ontwikkeld. Dit model wordt een referentie architectuur voor Farm Software Ecosystems genoemd. Ten tweede is er een technisch model ontwikkeld dat kan bijdragen aan technische aspecten van integratie, genaamd RAAgE 2.0.

Het organisatorische model is gebaseerd op literatuur waarin beschreven wordt dat gezamenlijke softwareontwikkeling kan plaatsvinden binnen software ecosystemen. Een software ecosysteem biedt een effectieve manier om softwarecomponenten te bouwen bovenop een softwareplatform. Deze softwarecomponenten kunnen worden ontwikkeld door verschillende partijen. Om bij te dragen aan de ontwikkeling van software ecosystemen voor de landbouw is een referentie architectuur ontwikkeld. Deze referentie architectuur beschrijft hoe softwareontwikkelaars, boeren en andere belanghebbenden kunnen samenwerken om ontwikkeling, configuratie en implementatie van geïntegreerde softwareoplossingen te realiseren. Zo kan de referentie architectuur gebruik worden voor het in kaart brengen, beoordelen, ontwerpen en implementeren van Farm Software Ecosystems zodat deze de huidige integratieproblemen kunnen oplossen of verkleinen. De referentie architectuur omvat vijf hoofdonderdelen:

(i) **Actoren**, dit kunnen app ontwikkelaars, business architecten / softwareontwikkelaars en eindgebruikers, die gebruik maken van de geconfigureerde softwaresystemen en diensten;
(ii) **Platform** waarmee configuratie van softwarecomponenten mogelijk is;
(iii) **Open software enterprise** die de relatie tussen actoren en het platform organiseert;
(iv) **Diensten** zoals ondersteuning van softwareconfiguratie en hosting;
(v) **Software componenten** van meerdere leveranciers die geconfigureerd worden in een geïntegreerd systeem.

De referentie architectuur voor Farm Software Ecosystems richt zich voornamelijk op organisatorische aspecten om samenwerking tussen actoren te bevorderen. Het technische gedeelte, om configuratie van verschillende ICT-componenten te realiseren, is hierdoor niet opgelost. Daarom is RAAgE 2.0 ontworpen. Dit model beschrijft hoe softwarecomponenten geconfigureerd kunnen worden. Dit model is gebaseerd op RAAgE 1.0 en is uitgebreid met een hiërarchische configuratiemethodologie en bijbehorende componenten. Deze methode bestaat uit twee stappen (i) business proces configuratie en (ii) de softwareconfiguratie. Om de eerste stap, business proces configuratie, te ondersteunen bestaat het model uit drie referentiemodellen die producten, processen en middelen kunnen beschrijven. De afhankelijkheden tussen deze drie modellen zijn gedefinieerd in regels die bepalen wat de mogelijke combinaties van producten, processen en middelen zijn. Het referentiemodel bevat ook een configuratieboom en sjablonen. Sjablonen beschrijven een set geconfigureerde product, proces- en resource-modellen voor specifieke akkerbouwbedrijven. Variatie tussen de bedrijven kan worden gemodelleerd door middel van business proces model varianten. Nadat de bedrijfsprocessen van een specifiek bedrijf gemodelleerd zijn kan de software geconfigureerd worden. Hierbij is de eerste stap het aanmaken van een sjabloon voor de softwareconfiguratie. Het sjabloon beschrijft de vereiste (generieke) softwareservices (capability types) die een specifiek bedrijfsproces kunnen ondersteunen. Het ontwikkelen van dit configuratiesjabloon kan worden uitgevoerd door een software architect in nauwe samenwerking met softwareontwikkelaars. De volgende stap is de selectie en configuratie van de specifieke softwarecomponenten van een specifieke leveranciers. Deze stap wordt uitgevoerd door een softwarearchitect, in nauwe samenwerking met de akkerbouwer die het systeem gaat gebruiken.

Om aannemelijk te maken dat de in dit onderzoek ontwikkelde artefacten bijdragen aan geïntegreerde softwaresystemen, die gebaseerd zijn op ICT Mass Customisation en best of breed, is er een case studie uitgevoerd waar binnen prototype software is ontwikkeld. In deze case studie is onderzocht hoe een specifieke akkerbouwer zijn aardappelen beschermd tegen de gewasziekte phytophthora en hoe hij dit proces met geïntegreerde software kan ondersteunen. In deze case studie zijn de volgende stappen uitgevoerd:

(i) Configureren van de business proces modellen ter voorkoming van phytophthora. In deze stap is RAAgE 2.0 gebruikt voor het identificeren van de processen en de benodigde softwarecomponenten;
(ii) Het ontwikkelen van de benodigde softwarecomponenten. Deze softwarecomponenten zijn ontwikkeld door 5

verschillende software ontwikkelaars uit verschillende Europese landen;

(iii) Configuratie van een geïntegreerd softwaresysteem met behulp van het FIspace Platform;

(iv) Instantiatie en gebruik van de softwarecomponenten.

Binnen deze case studie is aangetoond dat verschillende modulaire softwarecomponenten geïntegreerd kunnen worden en dat de prototypesoftware bedrijfsprocessen kan ondersteunen ter voorkoming van phytophthora. Door ontwikkeling van dit prototype hebben we aannemelijk gemaakt dat deze aanpak en de beschikbare artefacten integratie van software en de aansluiting met de bedrijfsprocessen verbeteren.

In dit onderzoek hebben we aangetoond dat de ontwikkelde artefacten bijdragen aan software integratie door samenwerking tussen betrokken actoren te ondersteunen. Om software integratie in de praktijk te verbeteren is vervolg onderzoek noodzakelijk. In vervolg onderzoek adviseren wij om te concentreren op:

(1) Ontwikkeling van business modellen. Op dit moment is het nodig dat er meer helderheid komt in de mogelijke motieven van softwareontwikkelaars om deel te nemen aan Farm Software Ecosystems om modulaire softwarecomponenten te ontwikkelen die geconfigureerd kunnen worden.

(2) Verbetering van softwareconfiguratie. Op dit moment is de configuratie van modulaire softwarecomponenten nog lastig. Om deze reden raden we aan om een verdiepende case studie uit te voeren. In deze case studie moet er met meer detail ingegaan worden op variatie van configuraties. Een gedetailleerde configuratiebeschrijving van een specifieke case kan worden hergebruikt voor andere cases zoals bemesting, gewasbescherming, zaaien en oogsten.

Hoewel er nog steeds hindernissen te nemen zijn, raden wij aan om onderzoek naar softwareconfiguratie en verbetering van software integratie voort te zetten. De verwachting is dat voortzetting van dit onderzoek kan leiden tot verbeterde integratie van softwaresystemen waardoor geavanceerde management stijlen, zoals precisielandbouw, mogelijk wordt. Deze geavanceerde managementstijlen kunnen leiden tot hogere voedselproductie, terwijl het voedsel meer duurzaam, veilig en transparant geproduceerd wordt.

About the author

Jan Willem Kruize was born on the 19th of January 1985 in Uithuizermeeden. He grew up in the northernmost tip of Groningen, the Netherlands. A passion for agriculture runs in the family. Both his father's as his mother's side of the family worked in agriculture, one grandfather was a farmer in the Noord Oost Polder and the other was working as a potato inspector. These roots transferred to Jan Willem and his family. His father owns a company that cultivates, forces and exports flower bulbs in which Jan Willem started working at an early age.

In 2003 Jan Willem started his studies at Wageningen University and graduated his masters in 2009. In 2010 he started his PhD research in which he studied how farm enterprise integration and sharing of data in agri-food supply chains can be improved using information and communication technologies. During his PhD he published several articles in international journals and conference proceedings. The graduation ceremony of his PhD was in January 2017.

In 2014 he started as a scientific researcher at Wageningen Economic Research, part of Wageningen University & Research. He works both in national and international projects as a scientific expert and project leader. His research focuses on information management in agri-food chains. His purpose is to improve agri-food supply chains and individual enterprises with information and communication technologies to increase yields and to enable sustainable, transparent and safe food production.

Completed training and supervision plan

Jan Willem Kruize
Wageningen School of Social Sciences (WASS)

Name of the learning activity	Department/Institute	Year	ECTS*
A) Project related competences			
Business Process Management (PhD level)	Department of Industrial Engineering & Innovation Sciences, TU Eindhoven	2012	7
Information Systems for Operations and Supply Chains	Faculty of economics and business, University of Groningen	2012	5
Participation and Organization Workshop Process modelling in Agriculture	ARVALIS - Institut du vegetal	2011	2.4
Cordys Process Factory Course	Cordys Academy	2010	1
Writing project proposal	WASS	2010	6
B) General research related competences			
"Supporting precision agriculture by technology integration"	EFITA conference	2011	2
Poster presentation Midterm review entitled: "Design of a framework to integrate information system components in arable farm business processes"	WASS midterm review	2012	0.3
"Towards a design of a generic integration framework for agri-food supply chain networks"	EFITA conference	2013	2

Name of the learning activity	Department/Institute	Year	ECTS*
"An application integrating framework for agri-food supply chain networks"	WASS PhD day	2013	1
Involvement, coordination & presentations in the INF31306 course Information Management	Information Technology group, Wageningen University	2011/ 2012/ 2013	4
Writing for academic publication	Paleotours, Consultancy	2012	3
C) Career related competences/personal development			
Leergang persoonlijke effectiviteit en management	Boertien Vergouwen Overduin	2016	2.5
People Leadership Expedition	Direction	2016	2.6
Project Management (PPL Project 124 and 130)	Information Technology group, Wageningen University	2013	2.0
Scientific Writing	Wageningen Graduate Schools	2011	1.7
Scientific Publishing	Wageningen Graduate Schools	2011	0.3
Competence Assessment	Wageningen Graduate Schools	2010	0.3
Total			43.1

*One credit according to ECTS is on average equivalent to 28 hours of study load

The studies presented in this thesis were performed with funding by the Dutch Program on Precision Agriculture (PPL), Wageningen Economic Research and by the European Community’s Seventh Framework Programme (FP7/ 2007-2013) within the project FIspace under grant agreement n_ 604 123.

Print:
ProefschriftMaken || Proefschriftmaken.nl

Cover design:
Petra van Berkum

seeds 80 %
humidity 20%
no dev
needs attention
12 cm
ETA 34 days - needs fert. in 10 days
28 % GRW
32 % DRY
10% ACQ
65% FRT