Semantic Knowledge for Autonomous Smart Farming

Ganesh Ramanathan [*] **Danai Vachtsevanou** [*]
Kimberly García [*] **Jérémy Lemée** [*] **Samuele Burattini** [**,*]
Kenan Bektaş [*] **Simon Mayer** [*]

[*] *University of St. Gallen, St.Gallen, Switzerland, (email: firstname.lastname@unisg.ch)*
[**] *University of Bologna, Bologna, Italy, (email: samuele.burattini@studio.unibo.it)*

Abstract: A higher degree of automation – and autonomization – of agricultural processes is expected to lead to productivity gains, especially in light of more environmentally-friendly farming practices, while improving the safety of agricultural processes. To exploit the potential of this development, it should be possible to flexibly integrate devices and services within service mashups, and thereby enable them to provide higher-value services together, However, current farm automation tools instead tend to reinforce vertical functional silos and tight coupling within often proprietary systems that manage the farm environment information. We propose to describe capabilities of individual devices and services and interlink them across components and with the description of the farm environment. We posit that this will better enable autonomous agents – software agents as well as humans – to perform complex agricultural tasks while integrating heterogeneous devices and services across multiple vendors. Concretely, we describe – and demonstrate in a laboratory setting – the usage of a Knowledge Graph to describe the environment and equipment used to perform farming tasks. We show how a multi-agent-based automation system for smart farming uses this graph to reason about the state of the environment and the agents to plan the achievement of user-specified goals. Furthermore, we show how such knowledge-driven autonomous systems may include human agents alongside artificial agents as first-class citizens, towards realizing "Social Machines" in the agriculture domain.

Keywords: smart farming, semantic data, multi-agent system

1. INTRODUCTION

The importance of automating agricultural activities for the purpose of achieving higher productivity while considering safety and environment-friendliness has been highlighted by both, research and farming communities (Edan et al., 2009). Using Knowledge Graphs (KGs) to describe a farm, its crops, implements, and processes, has been used in the farming community – however, the use of such KGs has focused on biology-oriented information (e.g., crop taxonomies), data exchange between farm management systems and external stakeholders (e.g., monitoring, logistics) (Drury et al., 2019), and supporting human understanding of a farm through user interfaces, or for analysis based on semantic understanding of data sources (e.g., sensors), while using KGs for guiding the behavior and automation of smart farming equipment has not been demonstrated.

In this paper, we show that existing ontologies in the agriculture domain can be integrated with machine-readable descriptions of smart farming equipment to enable artificial agents to automate agricultural processes. To accomplish this, our agents use a KG to organize themselves, choose appropriate plans, and interact with devices and the environment. Additionally, our agents are aware of human agents in the farming organization and are able to interact with them to ask for help or to refine their planning. Besides planning, the agents can use the KG to find means for perceiving and acting upon the environment (cf. Ciortea et al. (2018a)).

In the following sections, we describe the integration of well-known ontologies in the agriculture domain to create a KG for smart farming. Then, we explain how this KG was enriched with descriptions of *Things* [1] in the environment. Next, we describe its usage in a multi-agent system with the objective of achieving smart farming that is not bound to pre-programmed behaviour but can dynamically adapt to changes. Finally, we present a laboratory demonstrator of our contribution.

2. ONTOLOGIES AND KNOWLEDGE GRAPHS

Ontologies are formal machine-readable descriptions of a specific domain, they define concepts, relationships among such concepts, and logic for inferencing and reasoning. The objective of an ontology is to provide a common understanding of a specific domain, allowing devices and systems to interoperate. Moreover, in an ontology, the

[1] See `https://www.w3.org/TR/wot-thing-description11/#thing`

descriptions of concepts are enriched by linking them to other relevant concepts, contextualizing these concepts within their domain.

Recently, KGs have been associated with a broad variety of knowledge-based approaches, implementations, and technologies (e.g., DBPedia, Wikidata, and Facebook's graph), with the common goal of stopping producing isolated strings of data and instead linking them to concepts that can better describe the data. Today, it is common to talk about ontologies as the knowledge model that a KG uses. The Internet of Things, and specifically the Web of Things (WoT) (Guinard and Trifa, 2009) – benefit from a semantic approach as well (Pfisterer et al., 2011) since this can further facilitate the interoperation and composition of physical devices (e.g., sensors, actuators) and virtual services based on semantic metadata (Mayer et al., 2016). In the following section, we propose the integration of well-known ontologies in the agriculture domain to create a KG for smart farming.

2.1 A Knowledge Graph for Smart Farming

Our aim is to utilize ontologies and standards that are used and well-known within the relevant communities to create an interoperable system that allows for the addition of new devices and services coming from a variety of contributors and vendors, but which have all been described through widely used and agreed-upon metadata.

In the agricultural domain, many vocabularies and ontologies have been proposed for capturing agricultural knowledge at different levels of specificity. For instance, AGROVOC [2] is the most popular and largest thesaurus in food, agriculture and related science, maintained by the Food and Agriculture Organization of the United Nations. In 2017, the Consultative Group on International Agricultural Research (CGIAR) [3], a global network dedicated to reducing poverty, enhancing food security, and improving the use of natural resources, presented a platform for Big Data in Agriculture (Arnaud et al., 2020). CGIAR provides an up-to-date ontology registry that makes available ontologies from specialized working groups, so as to facilitate the usage of this metadata (e.g., through the GARDIAN [4] platform). In this work, we integrate some of the ontologies provided by CGIAR to semantically describe an environment in which software agents can autonomously discover and operate devices and software. This includes the Agronomy Ontology (AgrO) (Devare et al., 2016), the Environmental Ontology (EnvO) (Buttigieg et al., 2016), and the Plant Ontology (Cooper et al., 2013). Such ontologies are validated by the Open Biological and Biomedical Ontology (OBO) Foundry [5]. Furthermore, we consider other well-known ontologies and standards, such as the W3C WoT Thing Description (TD) ontology [6] to describe devices and software that expose machine-readable description of their programming interface (cf. Section 3). Furthermore, the Semantic Sensor Network Ontology (SSN) [7] describes sensors and actuators; the ontology for Managing Geometry (OMG) [8] allows describing the geometry of a farm; and our proposed ontology serves as an integration point of the aforementioned ontologies through the creation of bridging classes (Hodges et al., 2017). Listing 1 shows the prefixes used in the ontologies we consider.

Listing 1. PREFIXES of the integrated ontologies

```
agro: http://purl.obolibrary.org/obo/AGRO_
envo: http://purl.obolibrary.org/obo/ENVO_
PO: http://purl.obolibrary.org/obo/PO_
td: https://www.w3.org/2019/wot/td#
ssn: http://www.w3.org/ns/ssn/
omg: https://w3id.org/omg#
smtfrm: http://semantics.interactions.ics.
unisg.ch/smartfarming
```

Fig. 1 shows an overview of the most relevant classes that were identified in the ontologies we integrated. Due to space constraints, we only elaborate on a few classes and their relationships. Classes of the same color belong to the the same ontology, identified by a prefix. Fig. 1(a) shows a fundamental class hierarchy for agents that operate and collaborate to achieve farming goals, namely `smtfrm:SoftwareAgent` and `smtfrm:HumanAgent`. A `smtfrm:SoftwareAgent` acts upon a `smtfrm:Artifact` (see Fig. 1(b)); its subclasses specify the type of artifact. Those artifacts that correspond to machines and specifications that are described through a WoT TD are direct instances of `td:Thing` (see Fig. 1(c)) or described further by the `smtfrm:Machine` class (see Fig. 1(d)). `agro:AgriculturalImplement` describes farming equipment, e.g., `smtfrm:SmartTractor`, and `agro:FertilizationImplement`. Other instances of `td:Thing` could be `smtfrm:ProcessSpecification`-type (Fig. 1(e)), i.e., `agro:AgriculturalProcesses` that may be performed around the farm (e.g., `agro:Irrigation` and `agro:Fer-`

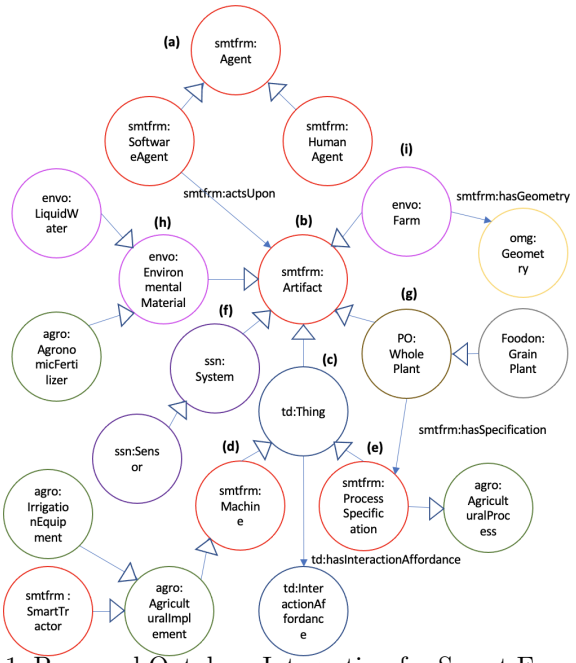

Fig. 1. Proposed Ontology Integration for Smart Farming.

[2] https://www.fao.org/agrovoc
[3] https://www.cgiar.org/
[4] https://gardian.bigdata.cgiar.org
[5] https://obofoundry.org/
[6] https://www.w3.org/TR/wot-thing-description11/
[7] https://www.w3.org/TR/vocab-ssn/
[8] https://www.projekt-scope.de/ontologies/omg/

tilization). A td:Thing can be related to its td:InteractionAffordances and subclasses thereof. Other types of smtfrm:Artifact include ssn:Sensor and ssn:Actuator, which are subclasses of ssn:System (see Fig. 1(f)). For describing plants growing on a farm, we use the PO:WholePlant (Fig. 1(g)) class, which describes a large variety of plants such as Foodon:GrainPlant for maize, rice, wheat, etc. Moreover, executing the process specification of agro:AgriculturalProcesses might require the use of an agro:AgriculturalImplement together with an envo:EnvironmentalMaterial such as envo:LiquidWater or agro:AgronomicFertilizer (see Fig. 1(h)). Finally, the physical farm itself should be described in a machine-readable way, such that agents may guide its exploration and the task performance. Thus, a farm of type envo:Farm (see Fig. 1(i)) is related to an instance(s) of omg:Geometry.

3. WEB OF THINGS

A key challenge faced in the IoT domain is the lack of interoperability amongst devices and applications across vendors and protocol standards. This fragments smart environments into vertical silos. In order to tackle this challenge, the WoT is an architectural recommendation from the W3C to build interoperable applications on the IoT. At its core, WoT provides a standardized way of describing *Things* and their affordances using a model called the Thing Description (TD). The TD contains metadata describing the td:Thing along with the communication protocol and security requirements needed to interact with it, where interactions are classified as *properties*, *actions*, or *events*. In our system and based on the ontologies described above, TDs are used to describe equipment like smtfrm:SmartTractor which is fitted with components like smtfrm:Sprayer so that agents responsible for carrying out agro:AgriculturalProcesses can use the TDs to interact with the equipment. The following listing shows the TD of a smart tractor with the capability for participating in agront:Fertilization process:

```
{"name": "smart tractor 1",
 "@type": "smtfrm:Tractor",
 "actions":[{
    "irrigate": {
    "@type": "agront:Fertilize",
    "type": "number",
    "forms": [{
      "op": "readproperty",
      "href": "http://tractor1/fertilize"}]}}]}
```

4. AGENTS FOR SMART FARMING

Wooldridge defines an *agent* as "a computer system that is situated in some *environment*, and that is capable of some autonomous action in this environment in order to meet its design objectives" (Wooldridge, 1999).

Intelligent agents are agents characterized by their reactiveness, proactiveness, and social ability (actions are taken to interact with other agents, possibly including humans) (Wooldridge, 1999). Such intelligence might be required for agents in a smart farm because of the need to integrate proactive and reactive behaviour to react to changes in the environment while taking actions that are based on the pursuit of goals. For example, an agent given the goal to irrigate a field can use proactive behavior to decide to use a sprinkler to achieve that goal but if the sprinkler becomes unavailable, the agents needs to react to this event and finds another solution, thus integrating proactive and reactive behaviors. A set of agents together with their shared environment, their interactions, and their organizations constitute a Multi-Agent System (MAS; Boissier et al. (2013)).

The Agents & Artifacts meta-model (Ricci et al., 2010) defines the agent environment with artifacts that can be observed and acted upon by agents. Within this model, the agent dimension and the environment dimension are orthogonal, meaning that both can be programmed independently, which allows the designer of a smart farm to choose any suitable agent model. The model further allows addition of artifacts to an existing environment and is therefore suited for dynamic environments – beyond farms, this has been applied to increase the scalability and flexibility of manufacturing in Ciortea et al. (2018b).

In an environment, artifacts correspond to the different tools the agents should be able to act upon, regardless of whether these tools are physical (e.g., smart tractors) or virtual (e.g., services, processes, or human interfaces). The artifacts therefore include all the td:Things present on the farm, and are made accessible through TDs. Considering Things as artifacts was proposed in Ciortea et al. (2018a) and is based on the similarity between the notions of observable properties, observable events, and operations in the Agents & Artifacts meta-model and the the notions of properties, events, and actions in TD.

The KG can be accessed through a KG Artifact that is able to construct and execute queries in the form of SPARQL [9], such as the following query that shows how agents may find suitable equipment for a process:

```
SELECT ?tractor ?sec WHERE {
 ?sec a :FarmLandSection .
    ?sec :hasCrop ?crop.
    ?crop :hasFertilizationSpec ?frspec.
    ?frspec :hasRate ?rate.
    ?sec :hasArea ?area.
    ?tractor a :SmartTractor.
    ?tractor :hasComponent ?sprayer.
    ?sprayer a :Sprayer.
    ?sprayer :currentVolume ?cval.
    FILTER(?cval >= (?rate * ?area))
}
```

Answers for these queries are then sent back to the agent. Even though this artifact is only able to create SPARQL queries according to a template, this does not restrict the generality of the model because the system could easily be extended with further KG artifacts. Humans can also be enabled to interact with the systems through artifacts with natural language processing and generation systems or mixed reality interfaces.

To program agents' procedural knowledge, we use *AgentSpeak* (Rao, 1996), a language to program agents that is based on the Belief, Desire, Intention (BDI) model. In

[9] https://www.w3.org/TR/rdf-sparql-query/

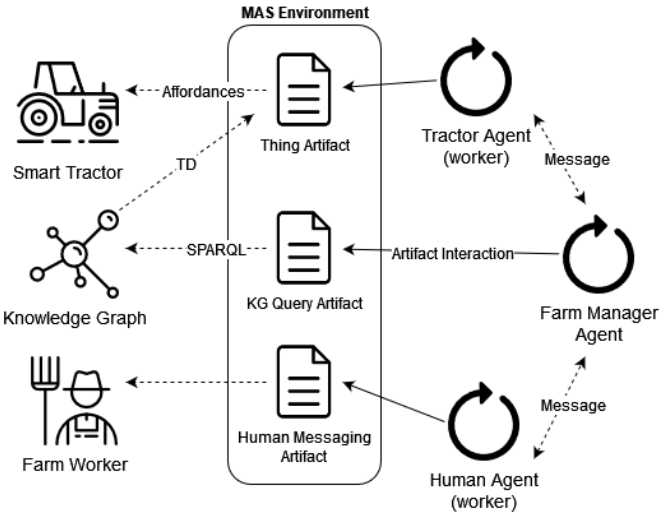

Fig. 2. Relationships among agents in the MAS and with the KG, artifacts, and humans. Dotted lines show on-need interactions.

AgentSpeak, agents have beliefs, goals, and plans. Goals correspond to desires in the BDI model while plans are a sequence of instructions that an agent can use to achieve a goal or react to a change in the belief base – in this way, both reactive (based on belief change) and proactive (based on goal change) behaviors are covered. The *Jason* language (Bordini et al., 2007), an extension of AgentSpeak, is used in the examples provided.

4.1 Agent Tasking and Coordination through KG

A MAS allows different agents to collaborate to achieve shared objectives, similar to a human organization. We propose to integrate agents and the KG within a farming MAS. In this system, our agents interact with the KG in each step of their execution. In our farming scenario, agents know about different pieces of farming equipment that are described in the KG – this equipment is considered as the farming 'workforce' alongside human workers, and is coordinated to complete the necessary processes to sustain the farm's needs and reach its design objectives. The novelty that we introduce here is that the KG is directly mediating the agent coordination since it is through the KG itself that tasks are discovered and that agents are identified as capable of completing them.

We propose a model based on a *Farm Manager Agent* that observes the KG in order to identify farm needs, discover which agents could be involved in a task, and delegates goals to them through agent-to-agent communication (Fig. 2). This is one possible model that emphasizes a hierarchical separation of concerns over anarchic independence of the worker agents.

4.2 Agent Decision Processes and KG

BDI agents choose how to behave based on their current *mental state* that comprises the beliefs and goals of the agent. When programming an agent, the programmer is required to specify *agent plans* [10] and to define in what *context* such plans are valid depending on the agent's knowledge of the world at run time. This can be cumbersome, and created agent programs might be hard to maintain and to evolve over time.

In our proposal, the programmer is relieved of this task since contextual aspects are captured in the KG, and what might be relevant for the decision of whether to carry through a plan or not is checked against the KG itself by the agent at run time. In this way, we reach a better separation of concerns between the domain expert (i.e., a farming engineer) who populates the KG and the agent programmer. The agent (and its programming) may focus on the interaction with the artifacts (including the KG Artifact) to define procedural plans. Our approach furthermore improves the flexibility and maintainability of the system since policies can be adapted in the KG without redeploying the agents.

At run time, if the state matches one of the agent's plans given the policies defined in the KG, the plan describing the agent's sequence of actions for an associated process can be executed. For example, the listing below shows the plan `canIrrigate` of the tractor agent that determines whether the agent is currently capable of irrigating (lines 7-11). This requires that the agent monitors the state of the tractor (e.g., the tractor's battery level; lines 1-5) and queries the KG to know if it currently affords irrigating based on the tractor's state:

```
1  !updateState.
2
3  +!updateState : tractor(T)
4  <-  readProperty("state", S)[artifact_id(T)];
5      -+state(S).
6
7  +?canIrrigate:
8      state(S) & knowledgeGraph(KG)
9  <- readProperty("affords", "irrigate", S, A)
10 [artifact_id(KG)];
11      if(A) { +canIrrigate; }
12      else { !canIrrigate[state(failed)]; }.
```

4.3 Increasing Agent Autonomy through KG

The availability of information in the KG can furthermore be used to decouple the procedural knowledge of an agent (i.e., how to control a machine to irrigate) from the information required to carry out a process in the domain (e.g., how much water is needed for a specific plot of land).

Agents already have mechanisms to retrieve action parameters at run time while executing a plan. This means that an agent which is controlling a smart farming device would only need to be programmed with plans that map a high-level process to the actual controlling operations of the device itself (i.e., its TD affordances) while the agent is able to discover all missing parameters to complete the task from the KG by itself.

The outcome of this is that an agent may then receive a very high-level goal to pursue, which further decouples the agent program from specific deployment contexts: The agent is now autonomous in its management of a process even in different contexts (e.g., for different field types),

[10] That is: To compose the sequence of actions that need to take place in order for an agent to achieve a goals.

without requiring the programming of multiple plans that specialize in diverse situations.

Taking the irrigation example, instead of tasking an agent to *irrigate* a *specific field* with *specific coordinates* by *spraying water* at a *specific flow rate* for a *specific amount of time*, the goal can now be simply to *irrigate a given field*; the agent will then fill in the missing information by using the KG to discover the field's coordinates, type of crop, growth stage, water requirements, etc. at run time, meaning that it can not only adapt to different contexts but also to the most up-to-date state of the environment when parameterizing its actions. [11] . This means that the only computation that the agent needs to carry out by itself is the conversion of the target watering amount to the input parameters of the irrigation device. This could further take into account models of the environment (e.g., for weather, surface runoff, etc.) that are evaluated at run time to match tasks, artifact capabilities, and parameters.

To give an example, Listing 2 shows an AgentSpeak plan that captures the agent's procedural knowledge regarding the irrigation process. By only knowing the field section to be irrigated (e.g., based on instructions from the farm manager agent), the tractor agent is able to find important information regarding the section at run time. This is achieved by exploiting the affordance `readSectionInfo` that is provided by the KG Artifact (lines 5-7). After receiving the updated information of the KG, the tractor agent calculates the spraying time for the section of interest, and proceeds with exploiting the affordance `irrigate` that is provided by the Tractor Artifact (lines 8-11). In case no error occurs during the execution of the plan, the Tractor Agent informs the Farm Manager Agent that the irrigation process is in progress (lines 13-14). Otherwise, the Tractor Agent informs them that the process failed during execution (lines 16-18).

Listing 2. The Tractor Agent queries the KG to resolve a high-level plan based on the current context.

```
1  +!irrigate(FieldSection)[source(Ag)] :
2      knowledgeGraph(KG) & tractor(T)
3  <-  ?canIrrigate
4      readProperty("readSectionInfo", Section,
5      Coordinates, PlantType, PlantCondition)
6      [artifact_id(KG)];
7      !computeSprayingTime(PlantType,
8      PlantCondition, Time);
9      invokeAction("irrigate", Coordinates,
10     Time)[artifact_id(T)];
11     .send(Ag, tell,
12     irrigationStatus(Section,
13     "in  progress")).
14
15 -!irrigate(Section)[source(Ag)]
16 <- .send(Ag, tell,
17     irrigationStatus(Section, "failed")).
```

[11] Note that this abstraction level is very similar to how a human farmer would task their co-workers – also in this case, the separation of concerns between agents is exploited.

4.4 Interaction with Human Agents through KG

In our proposed approach, the KG can be further used to distribute tasks among the artificial agents and human agents in our environment, based on their availability and capabilities. We have recently proposed this way of integrating artificial and non-artificial agents (Spirig et al., 2021), focusing on the split of concerns between *expert* digital companions who know the specifics of an environment (e.g., the available artifacts) and *personal* digital companions who are aware of the preferences of specific users – also in that case, coordination between these agents takes place via a KG, and we proposed to utilize Mixed Reality (MR) interfaces for the communication between artificial agents and humans. Recently, MR interfaces have emerged that support advanced human-machine interactions and provide customized assistance (Bektas, 2020; Orlosky et al., 2021), where we could even use gaze tracking to further enrich the KG of an agent's (predicted) intentions.

In this context, the proposed farming system takes the form of a *Social Machine* (Smart and Shadbolt, 2015) that enables collaborations among human and software agents; in fact, there is a growing interest in a close and continuous interaction between human and software agents in different forms of joint activities (e.g., collocation, cooperation, collaboration) (Bradshaw et al., 2017), while bringing humans closer to the center of the computation-loop (Shneiderman, 2020). Ideally, this would lead to systems where the separation of concerns between artificial and non-artificial agents follows the interests, preferences, and abilities of the individual (types of) agents to optimize the distribution of a domain problem among them [12] .

5. LABORATORY DEMONSTRATOR

We considered the case of a farm with two mobile robots ("Tractors") which are capable of fertilizing the fields. The KG is kept updated by a mocked soil monitoring agent. As soon as our rule-based system (written in SWRL [13]) determines that a section of the field needs nutrients based on the current soil condition, crop type, stage of growth, and weather conditions, the KG Artifact notifies the Farm Management Agent (FMA) that it should achieve the fertilization goal for the field. The FMA then starts an auction for the Tractor Agents (TA) to bid for participation in the achievement of the goal. In this process, each of the TAs uses the KG to determine if it is currently able to satisfy all requirements to reach the goal (based on its battery, presence of sprayer component, etc.), and then bid for participation. If the FMA assigns the goal to multiple TAs, then they coordinate with each other to collaboratively fertilize parts of the field section. In case it is not possible to achieve the goal, the FMA contacts a human agent with the appropriate role and abilities to request intervention. With respect to Section 4, our implementation demonstrates that the KG describes *what needs to be done* (agricultural process and specification), whereas

[12] In recent work, we have made this precise point about communities of artificial and non-artificial agents who together might achieve accurate yet scalable online fact-checking (Wild et al., 2020).

[13] See `https://www.w3.org/Submission/SWRL/`

the design of the software agents are decoupled from this and focus on *how to achieve the goal autonomously.*[14]

6. CONCLUSIONS AND FUTURE WORK

We have shown that integrated semantic descriptions can enable agents to automatically and collaboratively achieve specified goals in an agricultural context while achieving a favorable separation of concerns among stakeholders. This allows for highly flexible farm management systems, in contrast to traditional automation systems where imponderables need to be known and programmed upfront at design time. As a methodological contribution, we proposed the encapsulation of this semantic information as a KG Artifact that, along with all other artifacts in the system, provides a semantic description of its interfaces using standardized W3C WoT TDs. This gives farming application developers a more intuitive interface to the KG and thereby simplifies the creation, maintaining, and reconfiguration of farming applications. We intend to continue our work by creating a prototype which considers further equipment and agricultural processes as well as the regulatory environment in agriculture.

ACKNOWLEDGEMENTS

This research has received funding from the European Union's Horizon 2020 research and innovation program, grant No. 957218 (*IntellIoT*) and from the Swiss National Science Foundation, grant No. 189474 (*Hypermedia Communities of People and Autonomous Agents*). We thank Martijn Rooker and Wolfgang Hollerweger for their valuable comments on the manuscript.

REFERENCES

Arnaud, E. et al. (2020). The Ontologies Community of Practice: A CGIAR Initiative for Big Data in Agrifood Systems. *Patterns*, 1(7), 100–105.

Bektas, K. (2020). Toward A Pervasive Gaze-Contingent Assistance System: Attention and Context-Awareness in Augmented Reality. In *Proc. ETRA 2020 Adjunct*. ACM. doi:10.1145/3379157.3391657.

Boissier, O., Bordini, R., Hübner, J., Ricci, A., and Santi, A. (2013). Multi-Agent Oriented Programming with JaCaMo. In *Science of Computer Programming*. doi:10.1016/j.scico.2011.10.004.

Bordini, R., Hübner, J.F., and Wooldridge, M. (2007). *Programming Multi-Agent Systems in AgentSpeak using Jason*. Wiley.

Bradshaw, J.M., Feltovich, P.J., and Johnson, M. (2017). Human-Agent Interaction. In *The Handbook of Human-Machine Interaction*, 283–300. CRC Press. doi:10.1201/9781315557380.

Buttigieg, P.L., Pafilis, E., Lewis, S.E., Schildhauer, M.P., Walls, R.L., and Mungall, C.J. (2016). The environment ontology in 2016: bridging domains with increased scope, semantic density, and interoperation. *J. of Biomed. Semantics*, 7(1), 1–12.

[14] Our experimental setup with the KG, robots, and the MAS program in the JaCaMo framework (`http://jacamo.sourceforge.net/`) are available at `https://github.com/Interactions-HSG/smart-farming`

Ciortea, A., Boissier, O., and Ricci, A. (2018a). Engineering World-Wide Multi-agent Systems with Hypermedia. In *Proc. EMAS*, 285–301. Springer.

Ciortea, A., Mayer, S., and Michahelles, F. (2018b). Repurposing Manufacturing Lines on the Fly with Multi-Agent Systems for the Web of Things. In *Proc. AAMAS*.

Cooper, L. et al. (2013). The Plant Ontology as a Tool for Comparative Plant Anatomy and Genomic Analyses. *Plant Cell Physiol.*, 54(2). doi:10.1093/pcp/pcs163.

Devare, M., Aubert, C., Laporte, M.A., Valette, L., Arnaud, E., and Buttigieg, P.L. (2016). Data-driven Agricultural Research for Development: A Need for Data Harmonization Via Semantics. In *Proc. ICBO*.

Drury, B., Fernandes, R., Moura, M.F., and de Andrade Lopes, A. (2019). A survey of semantic web technology for agriculture. *Information Processing in Agriculture*, 6(4), 487–501. doi:10.1016/j.inpa.2019.02.001.

Edan, Y., Han, S., and Kondo, N. (2009). Automation in agriculture. *Springer Handbook of Automation*, 1095–1128.

Guinard, D. and Trifa, V. (2009). Towards the Web of Things: Web Mashups for Embedded Devices. In *MEM 2009, Adj. Proc. WWW*.

Hodges, J., García, K., and Ray, S. (2017). Semantic Development and Integration of Standards for Adoption and Interoperability. *IEEE Computer*, 50(11), 26–36.

Mayer, S., Verborgh, R., Kovatsch, M., and Mattern, F. (2016). Smart configuration of smart environments. *IEEE T-ASE*, 13(3), 1247–1255. doi:10.1109/TASE.2016.2533321.

Orlosky, J., Sra, M., Bektaş, K., Peng, H., Kim, J., Kos'myna, N., Höllerer, T., Steed, A., Kiyokawa, K., and Akşit, K. (2021). Telelife: The Future of Remote Living. *Frontiers in Virtual Reality*, 2. doi:10.3389/frvir.2021.763340.

Pfisterer, D., Römer, K., Bimschas, D., Kleine, O., Mietz, R., Truong, C., Hasemann, H., Kröller, A., Pagel, M., Hauswirth, M., Karnstedt, M., Leggieri, M., Passant, A., and Richardson, R. (2011). SPITFIRE: Toward a Semantic Web of Things. *IEEE Comm. Magazine*, 49(11), 40–48. doi:10.1109/MCOM.2011.6069708.

Rao, A.S. (1996). AgentSpeak(L): BDI agents speak out in a logical computable language. In *Proc. MAAMAW*. doi:10.1007/BFb0031845.

Ricci, A., Piunti, M., and Viroli, M. (2010). Environment programming in multi-agent systems: an artifact-based perspective. In *Proc. AAMAS*. doi:10.1007/s10458-010-9140-7.

Shneiderman, B. (2020). Human-centered artificial intelligence: Three fresh ideas. *AIS Transactions on Human-Computer Interaction*, 12(3), 109–124.

Smart, P.R. and Shadbolt, N.R. (2015). *Social machines*, volume 3, 6855–6862. IGI Global.

Spirig, J., Garcia, K., and Mayer, S. (2021). An Expert Digital Companion for Working Environments. In *Proc. IoT2022*. doi:10.1145/3494322.3494326.

Wild, A., Ciortea, A., and Mayer, S. (2020). *Designing Social Machines for Tackling Online Disinformation*, 650–654. ACM.

Wooldridge, M. (1999). Intelligent Agents. In *Multi-Agent Systems*.