Web Semantics: Science, Services and Agents on the World Wide Web 71 (2021) 100659

Contents lists available at ScienceDirect

Web Semantics: Science, Services and Agents on the World Wide Web

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

Review article

Intelligent software web agents: A gap analysis

Sabrina Kirrane

Institute for Information Systems and New Media, Vienna University of Economics and Business, Austria

ARTICLE INFO

Article history:
Received 8 March 2021
Received in revised form 9 July 2021
Accepted 31 August 2021
Available online 23 September 2021

Keywords:
Intelligent software agents
Agent architectures
Intelligent software web agents
Web standards

ABSTRACT

Semantic web technologies have shown their effectiveness, especially when it comes to knowledge representation, reasoning, and data integration. However, the original semantic web vision, whereby machine readable web data could be automatically actioned upon by intelligent software web agents, has yet to be realised. In order to better understand the existing technological opportunities and challenges, in this paper we examine the status quo in terms of intelligent software web agents, guided by research with respect to requirements and architectural components, coming from the agents community. We use the identified requirements to both further elaborate on the semantic web agent motivating use case scenario, and to summarise different perspectives on the requirements from the semantic web agent literature. We subsequently propose a hybrid semantic web agent architecture, and use the various components and subcomponents in order to provide a focused discussion in relation to existing semantic web standards and community activities. Finally, we highlight open research opportunities and challenges and take a broader perspective of the research by discussing the potential for intelligent software web agents as an enabling technology for emerging domains, such as digital assistants, cloud computing, and the internet of things.

Contents

E-mail address: sabrina.kirrane@wu.ac.at.

https://doi.org/10.1016/j.websem.2021.100659

1. Introduction

At the turn of the millennium, Berners-Lee et al. [1] coined the term semantic web and set a research agenda for this new research field. The authors used a fictitious scenario to describe their vision for a web of machine-readable data, which would be exploited by intelligent software agents who would carry out data centric tasks on behalf of humans. Hendler [2] further elaborated on the intelligent software agent vision with a particular focus on the key role played by ontologies in terms of service capability advertisements that are necessary in order to facilitate interaction between autonomous intelligent software web agents.

Several years later, in 2007, Hendler [3] highlighted that although the interoperability and intercommunication infrastructure necessary to support intelligent agents was available the intelligent agent vision had not yet been realised. Almost a decade later, in 2016, Bernstein et al. [4] discussed the evolution of the semantic web community and identified several open research questions, which they categorised under the following headings: (i) representation and lightweight semantics; (ii) heterogeneity, quality, and provenance; (iii) latent semantics; and (iv) high volume and velocity data. Interestingly, the authors identified the need to better understand the concrete requirements of intelligent software web agents both from a semantics and a deployment perspective. Hinting that the semantic web agent vision had not progressed from a practical perspective. Further evidence that the intelligent software agents vision has received limited attention in recent years was provided by Kirrane et al. [5], who used three data driven approaches in order to extract topics and trends from a corpus of semantic web venue publications from 2006 to 2015 inclusive. The authors highlighted that the intelligent agents topic did not feature in any of the top 40 topic lists produced by the three topic and trend detection tools used for their analysis.

Nevertheless, according to Luck et al. [6], a semantically rich data model, vocabularies, and ontologies, which can be used to describe media and services in a manner that facilitates discovery and composition, are key components of the proposed strategic agent technology roadmap. Indeed, over the years, the semantic web community has produced various standards and best practices that support data integration and reasoning using web technologies [4]. Many of which are discussed in the recent knowledge graphs tutorial article [7], which examines the role of semantic technologies when it comes to publishing and consuming knowledge graphs. Although several application areas (e.g., web search, commerce, social networks, finance) are discussed, there is no mention of agency. Interestingly, agents are briefly mentioned in several places, especially in the context of Findable, Accessible, Interoperable and Reusable (FAIR) principles, however agency from a conceptual perspective is not discussed. More broadly, agent technology and semantic web agents in particular could potentially serve as an enabling technology for various emerging domains (e.g., digital assistants, cloud computing, and the internet of things), especially when it comes to integration and governance. However, an important stepping stone to positioning intelligent software web agents as an enabling technology for more complex domains, is to determine what standards, tools, and technologies have been proposed, and to identify open research opportunities and challenges.

Thus, motivated by the desire to better understand the status quo, we perform a focused literature review shepherded by agent requirements and architectural components commonly discussed in the literature. Our work is guided by three primary research questions:

1. Which core requirements and architectural components are routinely used to guide software agent research?
2. What is the status quo in terms of intelligent software web agent research in terms of standards, tools, and technologies?
3. What are the primary opportunities and challenges for intelligent software web agents from both a requirements and an architectural perspective?

In order to answer the aforementioned research questions we adopt an integrative literature review methodology [8,9]. The goal being to integrate literature, in order to better understand the various proposals and how they relate to one another. The objective of our analysis is not to survey all literature that could be used to realise intelligent software web agents, but rather to use agent requirements and standard components used in agent architectures in order to perform a targeted analysis of the original intelligent software web agents motivating use case scenario and the potential solutions proposed to date.

Towards this end, we start by examining well known literature from the agents community that relates specifically to intelligent agent requirements and architectural components. Next, we use the intelligent agent requirements in order to better understand the functional and non functional aspects of the envisaged intelligent software web agents, and the various perspectives on said requirements coming from the semantic web literature. Following on from this, we use the intelligent agent architectural components in order to examine existing standards, tools, and technologies that could be used to realise the proposed hybrid agent architecture. We subsequently provide pointers, in the form of opportunities and challenges, that could be used to realise the semantic web agent vision. Finally, we discuss the potential for intelligent software web agents as an enabling technology for digital assistants, cloud computing, and the internet of things.

Our primary contributions can be summarised as follows: (i) we provide the necessary background information concerning intelligent agent requirements and architectures; (ii) we introduce an agent task environment requirements assessment framework that can be used to perform a detailed analysis of various agent based use case scenarios; (iii) we propose a web based hybrid agent architecture and use it to perform a gap analysis in terms of existing standards, tools, and technologies; and (iv) we identify existing research opportunities and challenges, and reinforce the need for intelligent software web agents as an enabling technology for several emerging domains.

The remainder of the paper is structured as follows: Section 2 presents the necessary background information in relation to intelligent agent requirements and architectures. Section 3 outlines our motivating use case scenario and presents the results of our requirements analysis. Section 4 examines related work on intelligent software web agents from the perspective of the various agent requirements. Section 5 proposes a web based intelligent agent architecture and discusses the standards, tools, and technologies that could be leveraged by the individual components. Section 6 performs a gap analysis in terms of existing standards and various research activities, summarises open research challenges and opportunities, and discusses the intelligent software agent potential beyond the original motivating use case scenario. Finally, we present our conclusions in Section 7.

2. Intelligent software agents

Originally robotics was the primary driver for agent based research, however the concept evolved to include software mimicking or acting on behalf of humans (i.e., software agents) and internet robots (i.e., bots) [10]. In this paper we focus specifically on intelligent software agents that use web resources in order to perform data centric tasks on behalf of humans. In order to provide a theoretical grounding for our assessment of the maturity of intelligent software web agents, we provide the necessary background information on intelligent software agent requirements and provide a high level overview of the most prominent agent architectures.

2.1. Agent requirements

Wooldridge and Jennings [11] distinguish between weak and strong intelligent software agents. In the case of the former, the agent is capable of acting autonomously, has the ability to interact both with humans and other agents, is capable of reacting to environmental changes, and exhibits proactive goal directed behaviour. In the case of the latter, the agent exhibits each of the aforementioned traits, however stronger agents are conceptualised based on human like attributes, such as knowledge, belief, intention, or obligation. In the following, we summarise different desiderata for intelligent agent behaviour and group related requirements based on the overarching function.

Basic functions.

Autonomy: Agents should manage both their state and their actions, and should be able to adapt to changes in their environment without direct intervention by humans [11–15].

Reactivity: Agents should be able to autonomously respond to environmental changes in a timely manner [11,13–15].

Pro-activeness: Agents should be able to pursue proactive goal directed behaviour [11,13–15].

Social ability: Agents should be able to interact with humans and other agents [11,14–16].

Behavioural functions.

Benevolence: Assumes that agents do not have goals that conflict with one another and thus are well meaning [11, 17].

Rationality: Assumes that agents do not act in a manner that would be counter productive when it comes to achieving their goals [11].

Responsibility: Involves acting according to the authority level that is entrusted to the agent either by the person or organisation that the agent represents or another agent [18].

Mobility: Refers to the ability to move around an electronic network, for instance using remote programming in order to execute tasks on other machines [11,14,19].

Collaborate functions.

Interoperability, communication, and brokering services: Agents need to be able to discover services and to interact with other agents [18].

Inter-agent co-ordination: Agents need to be able to work together with other agents in order to facilitate collective problem solving [18].

Code of conduct functions.

Identification: The ability to verify the identity of an agent and the person or organisation that the agent represents [18].

Security: Involves taking measures to secure resources against accidental or intentional misuse [18].

Privacy: Relates to being mindful of the privacy of the person or organisation that an agent represents [18], however more broadly an agent should respect the privacy of anyone with whom it interacts.

Trust: Involves ensuring that the system does not knowingly relay false information [11].

Ethics: Involves leaving the world as it was found, limiting the consumption of scarce resources, and ensuring predictable results [18], however in essence this could be interpreted as ensuring that agents do no harm.

Robustness functions.

Stability, performance, and scalability: This is a broad category that relates to ensuring that agents and multi-agent systems can handle increasing workload effectively and are highly available [13,14,18].

Verification: Relates to governance mechanisms that can be used to verify that everything works as expected [18].

2.2. Agent architectures

Existing architectures, encapsulating the software components and interfaces that ultimately denote an agents capabilities, can be classified as reactive, deliberative, or hybrid [10,11,20,21]. Reactive architectures are ideally suited for real time decision making (where time is of the essence), whereas deliberative architectures are designed to facilitate complex reasoning. Learning architectures are designed to enable the agent to improve its performance over time. While, Hybrid architectures strive to leverage the benefits of both deliberative and reactive architectures. In the following, we provide a high level overview of the external inputs and interfaces that are common to all architectures.

Environment: Agents act in their environment, possibly together with other agents. When it comes to web agents, the internet is a complex space that consists of a variety of different networking technologies, devices, information sources, applications, and both human and artificial agents.

Performance Measure: According to Russel and Norvig [22], when it comes to evaluating the effectiveness of an agent it is necessary to define success in terms of the state of the environment. Here there is a need for desirable qualities, taking into consideration that there may be conflicting goals making it necessary to assess and manage trade-offs.

Sensors: Software agents perceive the world through a variety of sensors, for instance the keyboard, cameras, microphone, network ports, etc., that can be used by agents to sense their environment.

Actuators: Software agents are capable of performing actions via a variety of actuators, for instance the screen, printer, headphones, network ports, etc., that can be used by agents to act on their environment.

2.2.1. Reactive architectures

Reactive agents are modelled on human based instinctive or reflexive behaviour [21]. When it comes to reactive agents there is a tight coupling between what the agent perceives and how the agent acts in the form of condition action rules [10]. In the following, we briefly introduce the components predominantly found in reactive architectures.

Condition-action rules: The agent simply retrieves the action associated with a particular condition perceived by its sensor(s) and uses the action to give instructions to the actuator(s).

State: More advanced reactive agents maintain state in the form of information about the world, and previous interactions with the environment. Given a new perception, the agent chooses an action based on both the current perception and its history of previous perceptions.

2.2.2. Deliberative architectures

Deliberative agents are rooted in the physical symbolic system hypothesis proposed by Newell and Simon [23], whereby a symbolic language is used to model the environment and decisions are taken based on logical reasoning [20]. In the following, we highlight the components predominantly found in deliberative architectures.

Knowledge base: The knowledge base is a symbolic encoding of both the agents knowledge of the world and the knowledge that governs its own behaviour.

Reasoning mechanism: Logical reasoning (e.g., deduction, induction, abduction and analogy) relating to conditions perceived by the agents sensor(s), the possible alternative actions, and their impact on the environment are used to enable the agent to give instructions to its actuator(s).

Goal encoding: Goals can be used to guide the agents decision making by describing behaviours that are desirable. The reasoning mechanism is used to select the action or set of actions that will lead to the satisfaction of a given goal.

Utility function: Agents that need to choose between different possible actions or sets of actions, can be guided via a utility function that allows the agent to perform a comparative assessment based on preferences such that it maximises its utility.

2.2.3. Learning architectures

Learning agents strive to become more effective over time, and are deemed especially useful when the agent environment is not known a priori [22]. Although the deliberative component could potentially be enhanced with learning abilities, Bryson [24] argues that modularity from an agent architecture perspective simplifies both design and control, thus in this paper we treat them as separate components. Nonetheless there is a tight coupling between both the deliberative and the learning components. Generally speaking learning agent architectures are composed of four additional components, representing the performance, problem generator, critic, and learning functions.

Performance: The performance component is an all encompassing term used to refer to the core inner functions of the agent.

Problem generator: The goal of the problem generator is to suggest actions that will lead to learning in the form of new knowledge and experiences.

Critic: The critic provides feedback to the agent (in the form of a reward or a penalty) with respect to its performance, which is measured against a fixed performance standard.

Learning element: The learning element performs actions assigned by the problem generator and uses the feedback mechanism provided by the critic to determine how the core inner functions of the agent should be amended.

2.2.4. Hybrid architectures

The individual reactive and deliberative architectural components described thus far can be organised into horizontal and/or vertical layers [11,20]. The proposed layering can be used to combat the shortcomings in terms of both reactive and deliberative architectures, however it increases the complexity of the system from a control perspective [10]. In the following, we briefly introduce the additional components predominantly found in hybrid architectures.

Layering: Horizontal and/or vertical layers are used to combine different functions, such as reactivity, deliberation, cooperation and learning.

Controllers: Controller components are necessary for planning the work, executing and monitoring activities, and managing interactions between activities.

3. Intelligent software web agents

The goal of this section is to revisit the original vision for the web, whereby machine-readable data would be exploited by intelligent software agents that carry out data centric tasks on behalf of humans. We start by summarising the use case scenario originally proposed by Berners-Lee et al. [1] in their seminal semantic web paper. Following on from this, we use the task environment framework proposed by Russel and Norvig [22] together with the requirements from the agents literature, presented in Section 2, in the form of a task environment requirements assessment, in order to provide additional insights into the functions necessary to realise the proposed intelligent software web agents.

3.1. Motivating use case scenario

The original semantic web vision [1] was presented with the help of a motivating use case scenario, revolving around Pete and Lucy and their mother who has just found out that she needs to attend regular physiotherapy sessions. Pete and Lucy ask their personal agents to prepare a physiotherapy appointment schedule such that they are able to share the chauffeuring duties.

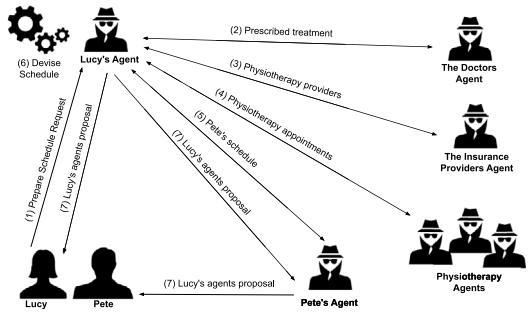

Fig. 1. Physiotherapy appointment planning workflow.

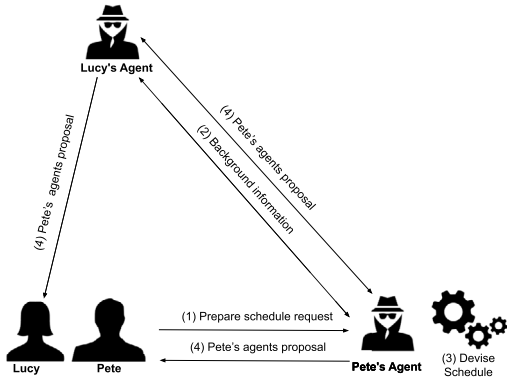

Fig. 2. Physiotherapy appointment planning workflow with additional constraints.

Physiotherapy appointment planning workflow. Lucy's agent is tasked with finding a physiotherapist who: (i) is covered by their mothers insurance; (ii) has a trusted service rating of very good or excellent; (iii) is located within a 20 mile radius of their mother's home; and (iv) has appointments that work with Lucy and Pete's busy schedules. The workflow depicted in Fig. 1 can be summarised as follows:

(1) Lucy requests that her agent devises a plan considering the given constraints.
(2) Lucy's agent consults with the doctor's agent in order to retrieve information relating to the prescribed treatment.
(3) Lucy's agent subsequently consults with the insurance provider agent in order to find physiotherapists considering the given constraints.
(4) Lucy's agent consults the various physiotherapy agents in order to retrieve available appointment times.
(5) Lucy's agent asks Pete's agent to give her access to his schedule.
(6) Lucy's agent uses the available appointment times provided by the various appointment agents, together with Pete's schedule provided by his agent, and Lucy's own schedule (that her agent already had access to) in order to prepare a physiotherapy appointment schedule considering available appointments and Pete's and Lucy's busy schedules.
(7) Lucy's agent shares the proposed plan with Lucy, and Pete's agent who in turn shares it with Pete.

Physiotherapy appointment planning workflow with additional constraints. Given the physiotherapist is quite far from Pete's work and the appointment times coincide with the lunch time rush hour, Pete instructs his agent to redo the task with stricter constraints on location and time. The workflow depicted in Fig. 2 is described as follows:

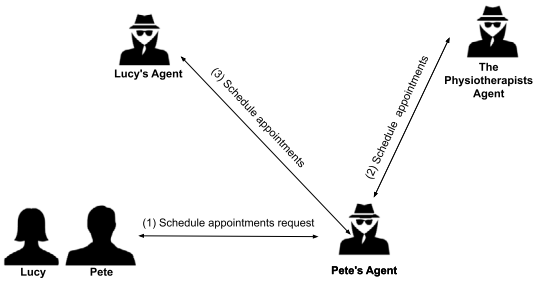

Fig. 3. Physiotherapy appointment conformation workflow.

(1) Pete requests that his agent propose a new plan that takes into consideration the additional location and time constraints.
(2) Pete's agent obtains all relevant background information relating to the initial proposal from Lucy's agent.
(3) Pete's agent uses the new constraints, together with the available appointment times and information related to Pete's and Lucy's schedules, in order to prepare a new plan, with two compromises: (i) Pete needs to reschedule some conflicting appointments, and (ii) the provider was not on the insurance companies list, thus his agent verified that the service provider was eligible for reimbursement using an alternative mechanism.
(4) Pete's agent shares the proposed plan and the compromises with Pete, and Lucy's agent, who in turn shares it with Lucy.

Physiotherapy appointment conformation workflow. Both Pete and Lucy agree to the plan, and Pete instructs his agent to make the appointments with the physiotherapist and update their schedules accordingly. The workflow depicted in Fig. 3 is outlined below:

(1) Pete instructs his agent to confirm the appointments and update his schedule accordingly.
(2) Pete's agent makes the booking with the physiotherapists agent.
(3) Pete's agent instructs Lucy's agent to update her schedule.

3.2. A task environment assessment

Next we use the Performance Measure, Environment, Actuators, and Sensors (PEAS) assessment criteria proposed by Russel and Norvig [22] to get a better understanding of high level goals of the various agents and to examine the external interfaces and inputs.

3.2.1. Information agents

In the given scenario, the doctor's agent and the insurance company service provider's agent can be classified as *information agents*. Given a request the agent uses the search parameters submitted with the request together with its own knowledge base in order to return an appropriate response. The details of our PEAS assessment can be found below:

Agent: The doctor's agent.

Performance Measure: The treatment information is provided.

Environment: The patient's agent.

Sensors: The doctor's agent `RetrievePrescribedTreatment` interface.

Actuators: The doctor's agent `RetrievePrescribedTreatment` interface.

Agent: The insurance company service provider's agent.

Performance Measure: The physiotherapy service provider information is provided.

Environment: The client's agent.

Sensors: The insurance company service provider's agent `RetrieveServiceProviderInfo` interface.

Actuators: The insurance company service provider's agent `RetrieveServiceProviderInfo` interface.

3.2.2. Booking agents

The physiotherapy appointment agent has two functions: (i) to provide information about available appointments; and (ii) to accept and confirm appointment requests. Thus this agent can be classified as a *booking agent* that allows for appointments to be scheduled based on availability. The details of our PEAS assessment is as follows:

Agent: The physiotherapy provider agents.

Performance Measure: The available appointments are provided, and/or the requested appointments are confirmed.

Environment: The client's agent.

Sensors: The physiotherapy appointment agent `RetrieveAvailableAppointments` and `BookAvailableAppointments` interfaces.

Actuators: The physiotherapy appointment agent `RetrieveAvailableAppointments` and `BookAvailableAppointments` interfaces.

3.2.3. Personal planning agents

Taking the given scenario into consideration, both Pete and Lucy's personal agents can be classified as `planning agents` that are tasked with providing an optimal plan based on the given constraints in terms of available treatments, location, time, etc. In addition, the agents need to work together in order to find a schedule that works for both Pete and Lucy.

Agent: Lucy's personal agent.

Performance Measure: The physiotherapist is covered by insurance, the physiotherapist is located near their mothers house, and the appointments fit with Pete and Lucy's schedules.

Environment: The doctor's agent, the insurance company agent, the physiotherapy provider agents, Pete's agent, and Lucy.

Sensors: Personal agent `PrepareSchedule`, `RequestInfo`, `MakeBooking` interfaces, and a web user interface.

Actuators: Personal agent `PrepareSchedule`, `RequestInfo`, `MakeBooking` interfaces, and a web user interface.

Agent: Pete's personal agent.

Performance Measure: The physiotherapist is covered by insurance, the physiotherapist is located near their mothers house, appointments fit with Pete and Lucy's schedules, the physiotherapist is near Pete's work, and appointments are not during busy traffic periods.

Environment: (Potential) The doctor's agent, the insurance company agent, the physiotherapy provider agents, Lucy's agent, and Pete.

Sensors: Personal agent `PrepareSchedule`, `RequestInfo`, `MakeBooking` interfaces, and a web user interface.

Actuators: Personal agent `PrepareSchedule`, `RequestInfo`, `MakeBooking` interfaces, and a web user interface.

3.3. A task environment requirements assessment

Unfortunately, the PEAS assessment does not provide any guidance with respect to the inner workings of the various agents, even though such information is necessary in order to determine the various architectural components and how they interact. Thus, in this section we propose a task environment requirements assessment and use it to perform a more detailed analysis of our motivating use case scenario.

We start by examining the *basic functions*, summarised in Table 1, that are needed to determine the type of architecture (i.e., reactive, deliberative, learning, and hybrid) required:

Autonomy: All three agent types are able to perform tasks without human interaction.

Reactivity: The information and booking agents are simple request response agents, however scheduling agents need to both interact with other agents and to examine possible solutions to the task that they have been given.

Pro-activeness: Although all three agents exhibit goal directed behaviour, scheduling agents would be classified as more pro-active as they need to explore various alternatives.

Social ability: When it comes to the information and booking agents, although the scenario focuses primarily on agent to agent interaction, all three agent types need to be able to interact with humans and agents.

Next, we examine the various *behavioural functions*, summarised in Table 2, that govern how our agents are expected to act:

Benevolence: We assume that information, booking, and scheduling agents are well meaning, however it is conceivable that different personal agents in the broader sense may have conflicting goals.

Rationality: Agents should be designed in order to ensure that agents do not act in a manner that would be counter productive when it comes to achieving their goals.

Responsibility: In the case of all three agent types, there is a tight coupling between responsibility and the overarching goals of the various agents, namely providing access to the requested information, completing the booking, finding an optimal schedule given a set of constraints.

Mobility: In the given use case scenario, the agents either request information from other agents or request that they perform a specific action, thus we assume that the agents are immobile.

Table 1
Basic functional requirements assessment.

	Information agent	Booking agent	Scheduling agent
Autonomy	handles information requests	handles information and booking requests	consult relevant sources, devise an optimal schedule
Reactivity	immediate response	immediate response	immediate response where possible
Pro-activeness	information goal	booking goal	scheduling goal, explore alternatives
Social ability	humans and agents	humans and agents	humans and agents

Table 2
Behavioural functional requirements assessment.

	Information agent	Booking agent	Scheduling agent
Benevolence	well meaning by design	well meaning by design	well meaning by design, manages conflicting goals
Rationality	rational by design	rational by design	rational by design
Responsibility	provides access to information	provides access to information, complete the booking	manages access to information, finds an optimal schedule given a set of constraints.
Mobility	–	–	interacts with several other agents

Table 3
Collaborative functional requirements assessment.

	Information agent	Booking agent	Scheduling agent
Interoperability	agreed/common schema	agreed/common schema	agreed/common schema
Communication	pull requests	pull requests	push and pull requests
Brokering services	collects information from multiple service providers	handles bookings for multiple service providers and clients	collects information from a variety of sources
Inter-agent co-ordination	–	–	agents support each other via information sharing

Next we examine the *collaborate functions*, summarised in Table 3, that can be used to determine how agents interact with humans and other agents, and how internal communication between agent components should work:

Interoperability: The requester needs to know which services it can call and how it can process the responses, thus there is a need for a schema that is understood by both the requester and the requestee.

Communication: In the given scenario, we assume that agents cater for pull requests via services, and in the case of scheduling agents push notifications on completion of a task.

Brokering services: Information agents are responsible for gathering information from multiple providers, the booking agents need to handle bookings for multiple service providers, and the personal agents are tasked with collecting the information needed in order to complete its task.

Inter-agent co-ordination: In the given scenario, the personal agents engage in a form of collaborative problem solving. The personal agents do not work collectively but rather support each other via information sharing (e.g., Pete's agent obtained all relevant background information relating to its task from Lucy's agent).

The *code of conduct functions*, summarised in Table 4, refer to the security, privacy, and ethical requirements that need to be built into the system:

Identification: In the case of the information agent it may or may not be necessary to authenticate the requester, for instance the personal heath records are available only to patients or their agents, however service provider information is usually public. Similarly, given that some service providers work on an honours system, authentication may or may not be needed in order to make a booking, however in both cases the client would need to be identifiable. When it comes to the scheduling agents, considering the amount of personal data needed by the agents in our motivating scenario, they would need to be able to authenticate their owners, and also the other personal agents with whom they interact.

Security: The system should be designed to protect against unauthorised access to, and inappropriate use of, data, as well as protecting against denial of service attacks.

Privacy: In the case of the information agents, it is necessary to differentiate between public and private information providers. However, in all other cases, agents will need to manage personal data and thus they need to adhere to the respective data protection legislation.

Trust: All three agent types need to be able to assess if they can trust the providers that they interact with, and ideally should be able to assess if the information they obtain from others is indeed correct. In this context, trust is a broad concept linked to reliability, fairness, transparency, explainability, verification, and validation.

Ethics: Although there are many things that could be discussed in detail under the ethics umbrella, here we envisage systems that do no harm, thus the agents should avoid behaving in a way that would bring about negative consequence either for the agent itself, or the agents and humans it interacts with.

Finally, the *robustness* requirements assessment, summarised in Table 5, defines criteria that should be used to determine the effectiveness of the architecture in terms of both functional and non functional requirements:

Stability: Stability is an all encompassing term used for availability, reliability, and security. Such metrics have an important role to play when it comes to evaluating the effectiveness of any system.

Performance: While information and booking agents need to be able to respond in real time, the scheduling agents will require time in order to source and analyse the data needed to derive an optimal schedule that satisfies a given set of constraints.

Table 4
Code of conduct functional requirements assessment.

	Information agent	Booking agent	Scheduling agent
Identification	may need to differentiate between public and private information providers	may need to differentiate between public and private information providers, some service providers may work on an honours system	needs to differentiate between public and private information, may need to prove who they represent
Security	protect against unauthorised access, inappropriate use, and denial of service	protect against unauthorised access, inappropriate use, and denial of service	protect against unauthorised access, inappropriate use, and denial of service
Privacy	may need to handle personal information	needs to handle personal information	handles personal information appropriately
Trust	manages information accuracy	manages information accuracy	manages information and scheduling accuracy, reliable and fair, provides transparency and explainability, robust in terms of verification and validation
Ethics	do no harm by design	do no harm by design	do no harm by design

Table 5
Robustness functional requirements assessment.

	Information agent	Booking agent	Scheduling agent
Stability	available, reliable & secure	available, reliable & secure	available, reliable & secure
Performance	provides real time access to information	provides real time access to information	provides real time access to information, timely goal completion
Scalability	handles increasing requests & data	handles increasing requests & data	handles increasing requests, data, & task complexity
Verification	checks information is correct	checks information is correct	checks information is correct, the reasoning is explainable

Scalability: All three agent types need to be able to scale with increasing data and increasing requests.

Verification: All three agent types need mechanisms that can be used to verify that everything works as expected. In addition, scheduling agents need to be able to explain what information sources they used and the logic behind their proposal.

4. Intelligent software web agents: Requirements

In this section, we take a closer look at intelligent software agents and how the various requirements are perceived from a semantic web perspective. Considering the broad nature of the topic, the goal is not to summarise all relevant literature, but rather to better understand the different perspectives on the various requirements introduced in Section 2 and discussed in the context of the original semantic web agent use case scenario in Section 3.

4.1. Basic functions

The works categorised in Table 6 and discussed below describe how intelligent agents could leverage semantic technologies, usually from a theoretical perspective.

Autonomy. Paolucci and Sycara [53] focus on service provision and usage, using the term autonomous semantic web services to refer to services that are capable of reconfiguring their interaction patterns, such that it is possible for them to react to changes with minimal human involvement. Several authors [30,33,36,39,40,47,63] focus specifically on autonomous agents and how they can leverage web services. While, Bryson et al. [28,29] take a more conservative view referring to agent behaviours as semi-autonomous intelligent modules. Artz and Gil [25] discuss the relationship between autonomy and trust. Van Riemsdijk et al. [65] in turn focus on adaptability, discussing how agents can adapt their behaviour in order to comply with norms. Tamma et al. [62] identify autonomous components as a desiderata for searching the semantic web. Sycara et al. [60] highlight key role played by brokers when it comes to discovery and synchronisation between autonomous agents. While, Tamma and Payne [61] argue that the sheer scale and heterogeneity of knowledge and services available on the web calls for autonomy not only on the part of the data and service providers but also the intelligent agents that are best placed to adapt to such dynamic, uncertain, and large scare environments. Payne [56], Leite et al. [50], Leite and Girardi [51], Buoncompagni et al. [31], Kootbally et al. [48], Merkle and Philipp [52] and Ghanadbashi and Golpayegani [41] discuss autonomy from a learning perspective, highlighting the need for agents to be self-aware by building up a knowledge base that allows them to learn alternative strategies and solutions that can be used to fulfil future goals. Huhns [44] focuses specifically on the tension between autonomy and co-ordination when it comes to inter-agent co-operation. The author highlights the need for extending web service standards to cater for federated servers and co-operating clients. Whereas, the autonomous agent architectures proposed by Fornara et al. [37], Fornara and Colombetti [38], Tonti et al. [64] and Van Riemsdijk et al. [65] are designed to cater for constraints in the form of policies or norms.

Reactivity. Bryson et al. [28,29] discuss how an agent-oriented approach to software engineering, entitled behaviour-oriented design, can be used to define reactive intelligent software web agents that are capable of managing interconnected (possibly conflicting) reactive plans. Boley et al. [26], Papamarkos et al. [54,54,55], Poulovassilis et al. [58], Gomes and Alferes [42], Ksystra and Stefaneas [49] and Jochum et al. [45] propose solutions that can be used to encode reactive functionality in the form of event-condition-action rules. Whereas, the architecture proposed by Käfer and Harth [46] makes use of simple condition-action rules. The discussion on web services from an agents perspective by Payne [56] and the framework proposed by Khalili et al. [47] consider reactivity in terms of an agents response to environmental changes. Bonatti et al. [27] in turn propose a formal framework that can be used to express and enforce reactive policies, while at the same time catering for trust negotiation between agents. While, Tamma et al. [62] discuss the role played by both reactive and pro-active components in the proposed searching for semantic web content system, whereby a reactive approach is used to keep indexes up to date.

Table 6
Intelligent software web agents basic function perspectives.

	Autonomy	Reactivity	Pro-activeness	Social ability
Artz and Gil [25]	autonomy & trust	–	–	social networks & reputation
Boley et al. [26]	–	event condition action rules	–	–
Bonatti et al. [27]	–	trust negotiation between agents	–	social network use case
Bryson et al. [28]	semi-autonomous	reactive plans	–	–
Bryson et al. [29]	modules & autonomous agents			
Buhler and Vidal [30]	autonomous agents & workflows	–	semantic web services & behavioural descriptions	social structures & workflows
Buoncompagni et al. [31]	learning agents	–	learning ability	–
Challenger et al. [32]	–	–	belief–desire intention	–
Chiu and Leung [33]	autonomous agents	–	believe-desire-intention framework & ontologies	–
Demarchi et al. [34]	–	–	belief–desire intention	–
Dong et al. [35]	–	–	belief–desire intention	–
Ermolayev et al. [36]	autonomous agents	–	collaborative goals	social commitments & conventions
Fornara et al. [37]	policy agents	–	–	–
Fornara and Colombetti [38]				
García-Sánchez et al. [39]	autonomous agents	–	semantic web services & behavioural descriptions	–
García-Sánchez et al. [40]				
Ghanadbashi and Golpayegani [41]	learning agents	–	learning ability	–
Gomes and Alferes [42]	–	event condition transaction language	–	–
Harth and Käfer [43]	–	condition action rule language	–	–
Huhns [44]	autonomy & co-operation	–	–	–
Jochum et al. [45]	–	event condition action rules	–	–
Käfer and Harth [46]	–	condition action rule language	–	–
Khalili et al. [47]	autonomous agents	environmental changes	goals	communication language
Kootbally et al. [48]	learning agents	–	learning ability	–
Ksystra and Stefaneas [49]	–	event condition action rules	–	–
Leite et al. [50]	learning agents	–	learning ability	–
Leite and Girardi [51]				
Merkle and Philipp [52]	learning agents	–	learning ability	–
Paolucci and Sycara [53]	autonomous semantic web services	–	–	–
Papamarkos et al. [54]	–	event-condition-action rule language	–	–
Papamarkos et al. [55]				
Payne [56]	learning agents	environmental changes	goals	social awareness
Pham and Stacey [57]	–	–	goals	–
Poulovassilis et al. [58]	–	event condition action rules	–	–
Rajpathak and Motta [59]	–	–	goals	–
Sycara et al. [60]	discover & synchronisation	–	–	–
Tamma and Payne [61]	service providers & autonomous agents	–	–	–
Tamma et al. [62]	autonomous system components	update indexes	update indexes	collaborative query answering
Terziyan [63]	autonomous resources	–	modelling context, dynamics, & co-ordination	–
Tonti et al. [64]	policy agents	–	–	social awareness
Van Riemsdijk et al. [65]	normative agents	–	–	socially adaptive agents

Pro-activeness. Buhler and Vidal [30] argue that semantic web services together with semantic behavioural description can be used by agents in order to achieve pro-active behaviour. The proposed approach also serves as a foundation for the ontology based intelligent agent framework proposed by García-Sánchez et al. [39,40]. Rajpathak and Motta [59], Khalili et al. [47] Payne [56] and Pham and Stacey [57] consider pro-activeness in terms of goal directed agent behaviours, with Ermolayev et al. [36] also considering pro-activeness in terms of collaborative goals in a multi-agent system. The agents proposed by Chiu and Leung [33], Dong et al. [35], Demarchi et al. [34] and Challenger et al. [32] all employ the belief–desire–intention software model. While, Payne [56], Leite et al. [50], Leite and Girardi [51], Buoncompagni et al. [31], Kootbally et al. [48], Merkle and Philipp [52] and Ghanadbashi and Golpayegani [41] examine how agents can be enhanced with pro-active learning ability. In turn, the multi-agent information system infrastructure proposed by Chiu

and Leung [33] is rooted in the believe-desire-intention framework whereby ontologies are used to encode knowledge that the agent acts upon. While, Terziyan [63] argues that semantic web standards need to be extended in order to cater for context, dynamics, and co-ordination, necessary to facilitate proactivity between agents. In the context of their searching for semantic web content system, Tamma et al. [62] argue that a proactive should be used by agents to inform other agents of any local changes.

Social ability. Tamma et al. [62] are guided by requirements relating to searching the semantic web whereby agents collaborate in order to answer queries. Artz and Gil [25] highlight the importance of social networks when it comes to trust in and among agents. Buhler and Vidal [30] discuss the role of agent cooperation and co-ordination from a workflow enactment perspective. While, Ermolayev et al. [36] identify the need for social commitments and conventions to regulate group activities. Both Khalili et al. [47] and Payne [56] highlight the fact that agents are socially aware, however Khalili et al. [47] furthers the notion by highlighting the importance of a common agent communication language. Bonatti et al. [27] demonstrate the effectiveness of their reactive policies and negotiation framework using a social network communication tool. The agents proposed by Van Riemsdijk et al. [65] are socially adaptive agents in the sense that they strive towards norm compliance. Tonti et al. [64] also adopt a social perspective, highlighting the need for policies that can constrain agent behaviour.

4.2. Behavioural functions

The works presented in Table 7 and discussed in more detail below provide different perspectives on the behavioural functions that could potentially be built into intelligent software web agents.

Benevolence. Both Artz and Gil [25] and Jutla et al. [68] briefly mention benevolence in the context of making decisions, with Artz and Gil [25] qualifying its use as a willingness to expend the effort needed to establish trust. Khalili et al. [47] focus on the assumption that benevolent agents do not have conflicting goals. Ermolayev et al. [36] discuss benevolence from a multi-agent group utility perspective, highlighting the need to balance self-interest and benevolence. While, Gandon [67] focus on the societal benefit of the web, arguing that artificial intelligence based applications need to be benevolent by design.

Rationality. According to Payne [56] agents need to act rationally when it comes to decision making, for instance by considering the utility gain in terms of a reward or a perceived advantage. In addition to defining rational behaviour, Khalili et al. [47] also specifically state that it is assumed that agents do not act in a counter productive manner. While, Ermolayev et al. [36] discuss rationality from a multi-agent perspective, focusing on the need to balance individual-rationality from a self-interest perspective and benevolence when it comes to group dynamics. Whereas, Tamma and Payne [61] identify the need for bounded rational deliberation when it comes to partial knowledge and updates to existing knowledge.

Responsibility. Paolucci and Sycara [53] discuss responsibility purely from a web service architecture perspective. While, Bryson et al. [29] focus on responsibility from a data retention perspective. The context broker architecture proposed by Chen et al. [66] focuses specifically on the responsibilities of the context broker agent which is at the core of the proposed meeting system. Demarchi et al. [34] in turn examine responsibility from an architectural perspective, identifying the need for responsible components. While, García-Sánchez et al. [39,40] take an intelligent agent perspective, identifying several different types of agents that are differentiated from one another via roles and responsibilities.

Mobility. Khalili et al. [47] list mobility in terms of ability to move around a network as one of the requirements of an agent based system. Ermolayev et al. [36] highlight the need for mobile agents in order to ensure the robustness of the system from an availability and a performance perspective. Several authors [66, 69,70] highlight the key role played by semantic web services when it comes to service discovery in mobile and ubiquitous environments. While, Outtagarts [71] performs a broad survey of mobile agent applications, with semantic web services being one of them.

4.3. Collaborate functions

In the following, we further elaborate on various works that fall under the collaborative functions heading. Table 8 presents existing proposals for intelligent software web agents that are particularly relevant for both agent to human and agent to agent interactions, as well as internal interactions between agent components.

Interoperability. Several authors [2,36,39,40,53,60,74] focus on interoperability from a web service perspective, putting a particular emphasis on automatic discovery, execution, selection, and composition. However, only García-Sánchez et al. [39,40] distinguish between data, process, and functionality interoperability. Both Gladun et al. [73] and Tamma and Payne [61] highlight the need for standardisation when it comes to the interoperability in multi-agent systems. In particular, Tamma and Payne [61] differentiate between syntactic, semantic, and semiotic interoperability. While, Shafiq et al. [77] focus specifically on communication between software agents and semantic web services by proposing an architecture that allows for interoperability via middleware that performs the necessary transformations. More recently, Harth and Käfer [43], Käfer and Harth [46] and Schraudner and Charpenay [76] have proposed agent architectures that are heavily reliant on linked data standards, which are interoperable by design.

Communication. Berners-Lee et al. [1] discusses the difficulties encountered when it comes to co-ordination and communication internationally. Huhns [44], Bryson et al. [28], Paolucci and Sycara [53], Shafiq et al. [77], and Motta et al. [74] highlight the role played by various protocols (e.g., Web Services Description Language (WSDL), Universal Description, Discovery and Integration (UDDI), and Simple Object Access Protocol (SOAP)), when it comes to web service publishing, finding, and binding. Berners-Lee et al. [1], Hendler [2], Sycara et al. [60], and García-Sánchez et al. [39,40] propose the use of shared vocabularies in the form of ontologies for communication between service providers and consumers. García-Sánchez et al. [39,40] extend their use in order to cater for communication between architectural components. Both of which raise issues from an interoperability perspective, especially in relation to ontological equivalence and reconciliation, as argued by Tamma and Payne [61]. When it comes to communication between agents, Ermolayev et al. [36], Gibbins et al. [72] and Huhns [44] highlight the need to standardise communication languages and vocabularies in order to facilitate communication between agents. While, Bonatti et al. [27] discuss the role played by policies and trust with a particular focus on negotiation. From a communication management perspective, the communication architecture proposed by Schraudner and Charpenay [76] ensures that agents can only communicate with each other indirectly via the environment, whereas Tonti et al. [64] use policies to control communication between agents.

Table 7
Intelligent software web agents behavioural function perspectives.

	Benevolence	Rationality	Responsibility	Mobility
Artz and Gil [25]	benevolence & trust	–	–	–
Bryson et al. [29]	–	–	data retention	–
Chen et al. [66]	–	–	context broker agent	service discovery
Demarchi et al. [34]	–	–	agent platform components	–
Ermolayev et al. [36]	self-interest & benevolence.	rationality & group dynamics	–	service mobility & availability
Gandon [67]	benevolence & societal benefit	–	–	–
García-Sánchez et al. [39] García-Sánchez et al. [40]	–	–	agent types, roles, & responsibilities	–
Jutla et al. [68]	benevolence, trust & integrity	–	–	–
Khalili et al. [47]	conflicting goals	goal oriented decision making	–	move around a network
Paolucci and Sycara [53]	–	–	web service architecture	–
Payne [56]	–	goal oriented decision making	–	–
Scioscia et al. [69]	–	–	–	service discovery
Sheshagiri et al. [70]	–	–	–	service discovery
Tamma and Payne [61]	–	partial & updated knowledge	–	–

Table 8
Intelligent software web agents collaborative function perspectives.

	Interoperability	Communication	Brokering services	Inter-agent co-ordination
Bonatti et al. [27]	–	policies & trust	–	–
Berners-Lee et al. [1]	–	ontologies, co-ordination & collaboration	–	–
Bryson et al. [28]	–	protocols	–	internal co-ordination
Ermolayev et al. [36]	web services	standard languages & vocabularies	–	ontologies
García-Sánchez et al. [39] García-Sánchez et al. [40]	web services	ontologies	data, process & function mediation	ontologies
Gibbins et al. [72]	–	standard languages & vocabularies	system architecture	–
Gladun et al. [73]	standards for interoperability	–	–	–
Harth and Käfer [43]	linked data standards	–	–	–
Hendler [2]	web services	ontologies	logical descriptions	–
Huhns [44]	–	protocols & standard languages & vocabularies	enhanced directory services	autonomy vs co-ordination
Käfer and Harth [46]	linked data standards	–	–	–
Motta et al. [74]	web services	protocols	interaction framework	–
Mcllraith et al. [75]	–	–	interaction framework	–
Paolucci and Sycara [53]	web services	protocols	–	coordinating role
Schraudner and Charpenay [76]	linked data standards	indirect communication	–	–
Shafiq et al. [77]	web services & agents	protocols	–	–
Sycara et al. [60]	web services & agents	ontologies	interaction framework	matchmaking & brokering
Tamma and Payne [61]	standards for interoperability	ontological equivalence & reconciliation	–	–
Tonti et al. [64]	–	communication policy	–	–

Brokering services. Hendler [2] highlight that adding logical descriptions to web services will facilitate automated match making and brokering. Mcllraith et al. [75], Motta et al. [74], and Sycara et al. [60] propose frameworks whereby agent brokers are used to manage the interaction between service providers and consumers. While, Gibbins et al. [72] propose a system architecture and discuss its effectiveness via a proof of concept simulator based application. Huhns [44] in turn discusses how directory services could be enhanced via brokerage services that help to refine the number of potential sources that need to be consulted. García-Sánchez et al. [39,40] highlight the importance of interoperability when it comes to the brokering process, which is further subdivided into data, process, and functional mediation.

Inter-agent co-ordination. Huhns [44] highlights the tensions between autonomy and co-ordination, as it is necessary to relinquish some autonomy in order to honour commitments. The architecture proposed by Paolucci and Sycara [53] distinguishes between peers and super peers, the latter being responsible for coordinating several peers. While, Bryson et al. [28] argue that there is also the need to have co-ordination internally, for instance between software modules, such that it is possible to develop composite services. Both Ermolayev et al. [36] and García-Sánchez et al. [39,40] propose the use of common vocabularies in the form of ontologies for both inter-agent communication and co-ordination. While, Sycara et al. [60] highlight the key

roles played by matchmaking and brokering when it comes to multi-agent co-ordination.

4.4. Code of conduct functions

The functions summarised in Table 9 and further elaborated on below are particularly relevant for the controller component, however they may also impact the design of several other components, thus they need to be considered when it comes to the architectural design of the system.

Identification. Several authors [1,2,36,39,40,43,46,61,85] highlight the role played by Uniform Resource Identifiers (URIs) when it comes to the identification of resources (e.g., web services, ontologies, agents). While, Artz and Gil [25], Gandon and Sadeh [80] and Kirrane and Decker [83] focus on the authentication of actors using credentials in the form of digital signatures together with policies.

Security. Both Gandon and Sadeh [80] and Kirrane and Decker [83] identify the need for access control policy specification and enforcement. Although, Chen et al. [66] argue that together ontologies and declarative policies can be used for both privacy and security, they do not go into specific details on their use from a security perspective. Artz and Gil [25] discuss security in terms of using credentials and policy languages in order to determine trust in an entity. While, Kagal et al. [82] propose ontologies that can be used to sign and encrypt messages exchanged between service providers and consumers. Sycara et al. [60] in turn argue that matchmakers can be used to address security concerns by offering a choice of providers.

Privacy. Chen et al. [66] discuss how their context broker architecture can be used to control the sharing and use of personal data. Jutla et al. [68] propose an agent based architecture that can be used to allow the specification and enforcement of privacy preferences. Gandon and Sadeh [80] in turn propose an agent based architecture that protects and mediates access to personal resources. Both Jutla and Xu [81] and Palmirani et al. [86] propose high level ontologies that can be used to specify privacy protection mechanisms in the form of laws, standards, societal norms, and guidelines. In addition, the authors describe how the proposed ontologies could be used by privacy agents to identify privacy issues. Whereas, Bao et al. [78] propose a framework that can be used for privacy preserving reasoning when only partial access to data is permitted. Artz and Gil [25] discuss privacy from a trust negotiation perspective and point to several policy languages that can be used to protect privacy. Kravari et al. [84] also focus on enhancing privacy via trust, proposing a policy-based e-Contract workflow management methodology. Whereas, Sycara et al. [60] identify matchmakers as a means to offer better privacy by offering a choice of providers.

Trust. Berners-Lee et al. [1] discuss the role played by digital signatures when it comes to verifying that information has been provided by trusted sources. While, Hendler [2] in turn focuses more broadly on using proof exchange to facilitate trust. Additionally, the detailed survey on trust models and mechanisms at the intersection of trust and the semantic web, conducted by Artz and Gil [25], is motivated by the need for agents to make trust judgements based on available data that may vary in terms of quality and truth. The web service composition framework proposed by Ermolayev et al. [36] considers both the credibility and trustworthiness of service providers as a key requirement that needs to be considered. While, Chen et al. [66], Jutla and Xu [81], Jutla et al. [68] examine trust from a personal data processing perspective, proposing a system that can be used to determine if the users privacy preferences are adhered to. Kirrane and Decker [83] highlight the link between trust, transparency, and provenance and point to several potential starting points. Kravari et al. [84] propose a policy-based e-Contract workflow management methodology that can be used to establish trust between agents and service providers. While, Tonti et al. [64] highlight the role between trust and policies from a trust management perspective.

Ethics. When it comes to intelligent software agents, Casanovas [79] argues that there is a need for both normative and institutional regulatory models in order to not only reason over legal norms, but also judicial and political decision making, best practices, ethical principles and values. Oren et al. [85] focus specifically on good behaviour when it comes to crawling data, stating it is important to respect the robot.txt access restrictions and to be mindful of the resource limitations of the data provider. While, Kirrane and Decker [83] identify usage restrictions in the form of access policies, usage constraints, regulatory constrains, and social norms as key requirements needed to realise the intelligent web agent vision.

4.5. Robustness functions

Finally, the existing work at the intersection of intelligent software web agents and robustness, summarised in Table 10, is useful for both assessing the maturity of the exiting proposals, and comparing and contrasting different technological choices.

Stability, performance & scalability. Shafiq et al. [77] examine the performance of both their service lookup via UDDI and service invocation via WSDL, and conclude that lookups scale linearly with increasing parameters, while service invocation depends on the complexity of the input and output parameters. Although Jutla et al. [68] do not conduct a performance assessment they identify the need for borrowing/extending metrics from other domains, such as response time, throughput, effectiveness, ease of use, and usefulness. Likewise, García-Sánchez et al. [39,40] discuss the importance of performance assessment and provide detailed plans that they aim to execute in future work. While, Scioscia et al. [69] use a reference dataset to compare their reasoning engine performance, in terms of both classification and satisfiability, to other well known reasoners, and the memory usage on a mobile device to that of a personal computer. Sycara et al. [60] take a broad view on performance identifying the need to assess different performance characteristics, such as privacy, robustness, adaptability, and load balancing.

Verification. Scioscia et al. [69] use a reference dataset to assess the effectiveness of their reasoner on a classification task, in terms of correctness, parsing errors, memory exceptions, and timeouts. While, Gandon and Sadeh [80] perform an empirical evaluation of their architecture via a campus community agent application, where participants were asked to perform various tasks with a view to obtaining feedback on the effectiveness of the system. Leite et al. [50] and Leite and Girardi [51] demonstrate the effectiveness of their proposals by performing a comparative analysis to other hybrid agent architectures. Additionally, a number of authors evaluated their proposals using smart home [46], production environment [76], and real time traffic [41] simulations. Other evaluations included ruleset correctness [45], safety [49], and norm compliance checking [65].

5. Intelligent software web agents: Architectural components

In the following, we propose a hybrid semantic web agent architecture, which discusses how the architecture components introduced in Section 2, together with semantic web standards and community activities, could potentially be used to realise our

Table 9
Intelligent software web agents code of conduct function perspectives.

	Identity	Security	Privacy	Trust	Ethics
Artz and Gil [25]	credentials & policies	credentials & policies	policies	trust judgements	–
Bao et al. [78]	–	–	privacy preserving reasoning	–	–
Berners-Lee et al. [1]	URIs	–	–	digital signatures	–
Casanovas [79]	–	–	–	–	regulatory models
Chen et al. [66]	–	ontologies & policies	ontologies & policies	trust & privacy	–
Ermolayev et al. [36]	URIs	–	–	credibility & trust	–
García-Sánchez et al. [39] García-Sánchez et al. [40]	URIs	–	–	–	–
Gandon and Sadeh [80]	credentials & policies	access policies	privacy architecture	–	–
Harth and Käfer [43]	URIs	–	–	–	–
Hendler [2]	URIs	–	–	proofs	–
Jutla and Xu [81]	–	–	privacy ontology	trust & privacy	–
Jutla et al. [68]	–	–	privacy architecture	trust & privacy	–
Käfer and Harth [46]	URIs	–	–	–	–
Kagal et al. [82]	–	digital signatures & encryption	–	–	–
Kirrane and Decker [83]	credentials & policies	access policies	–	trust & provenance	usage policies
Kravari et al. [84]	–	–	privacy contracts	trust via contract	–
Oren et al. [85]	URIs	–	–	–	social conventions e.g., robots.txt
Palmirani et al. [86]	–	–	privacy ontology	–	–
Sycara et al. [60]	–	matchmakers & security	matchmakers & privacy	–	–
Tamma and Payne [61]	URIs	–	–	–	–
Tonti et al. [64]	–	–	–	trust & policies	–

Table 10
Intelligent software web agents robustness function perspectives.

	Stability, Performance & Scalability	Verification
Gandon and Sadeh [80]	–	empirical evaluation
García-Sánchez et al. [39] García-Sánchez et al. [40]	evaluation plans	–
Ghanadbashi and Golpayegani [41]	–	simulation
Jochum et al. [45]	–	ruleset correctness
Jutla et al. [68]	borrowing/extending metrics	–
Käfer and Harth [46]	–	simulation
Ksystra and Stefaneas [49]	–	safety properties
Leite et al. [50],Leite and Girardi [51]	–	comparative analysis
Merkle and Philipp [52]	–	learning tasks
Schraudner and Charpenay [76]	–	simulation
Scioscia et al. [69]	reasoning engine performance & memory usage	reasoning correctness
Shafiq et al. [77]	service lookup & invocation performance	–
Sycara et al. [60]	several different performance characteristics	–
Van Riemsdijk et al. [65]	–	norm compliance checking

information, booking, and planning agents. Rather than proposing three different architectures we propose a single architecture with optional components. The proposed hybrid agent architecture, which is depicted in Fig. 4, provides support for realtime interaction via its reactive component and sophisticated reasoning via its deliberative component, both of which are necessary in order to realise our scheduling agent.

5.1. Interface component

In our use case scenario, `Sensors` and `Actuators` take the form of either web interfaces rendered via networked devices used for agent to human interaction, or web services residing on networked devices used for agent to agent interactions. Table 11 summarises the relevant W3C standardisation efforts and community activities discussed in detail below.

Sensors & actuators. The Hypertext Transfer Protocol (HTTP)[1] is an application level protocol that forms the basis for data communication via the web. Communication involves a simple request/response protocol that can be used for data exchange. The Linked Data Notifications (LDN)[2] specification in turn describes how HTTP together with RDF can be used by senders to push messages to recipients. When it comes to serving web content there are numerous web servers to choose from (cf., NGINX,[3] Apache Tomcat[4]). From a web interface perspective, the Hypertext Markup Language (HTML)[5] and Cascading Style Sheets (CSS)[6] can be used to develop responsive web applications that enable humans to interact with intelligent software agents. From a web

1 https://tools.ietf.org/html/rfc7231.
2 https://www.w3.org/TR/ldn/.
3 https://www.nginx.com/.
4 http://tomcat.apache.org/.
5 https://www.w3.org/TR/html52/.
6 https://www.w3.org/TR/CSS2/.

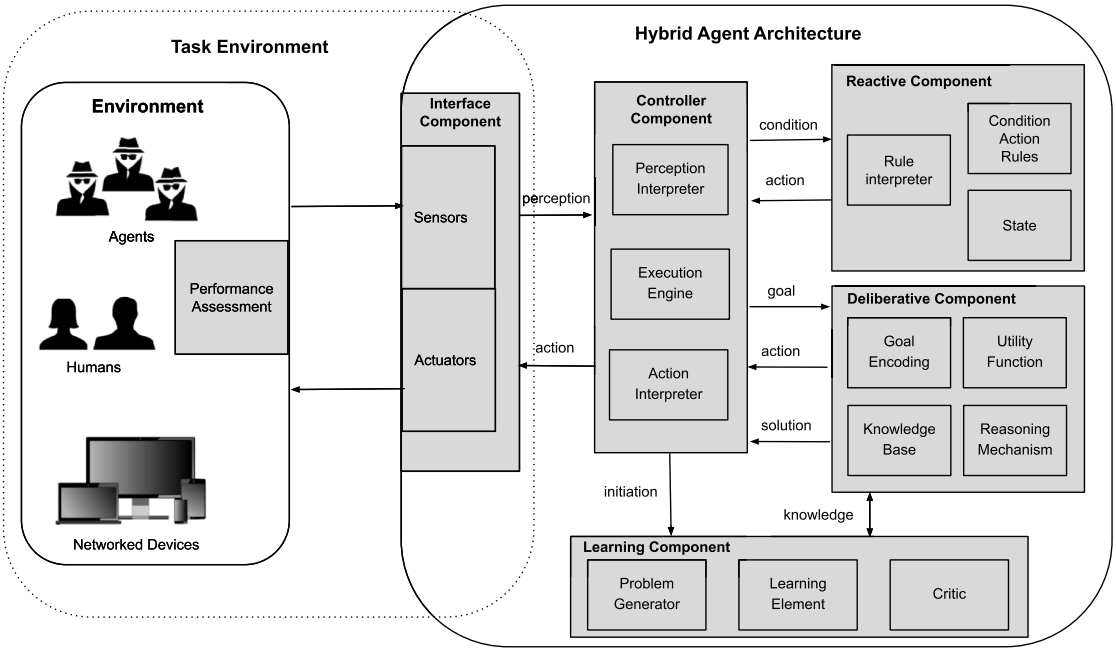

Fig. 4. A hybrid agent architecture.

Table 11
Intelligent software web agents interface component.

	Standards	Community activities
Languages, Ontologies & Vocabularies	HTML, CSS, WSML, WSDL, FIPA ACL, FIPA RDF CL, OWL, OWL-S, SAWSDL, WSMO	FIPA RDF CL extension [87], SEA_ML [32], compare OWL-S, WSMO and SAWSDL [88], FIPA process ontology [72], OWL-S & policies
Models & Frameworks	RDF, UDDI, REST	SWS discovery & composition [89–92], SWS & JADE [93], SWS design methodology [94], hypermedia controls [95]
Protocols	HTTP, SOAP, LDN	WS publication protocol [91], LDN agent communication protocol [96], ACL/SOAP converter [77]

services perspective, the Simple Object Access Protocol (SOAP)[7] and Representational State Transfer (REST)[8] are the predominant Application Programming Interface (API) styles used in practice. Web service discovery is supported via registries and indexes, whereby protocols such as the Universal Description, Discovery and Integration (UDDI)[9] can be used to publish and discover web services [97]. There are also several standardisation initiatives relating to semantic web services that use formal ontology-based annotations to describe the service in a manner that can be automatically interpreted by machines. For instance, the Web Ontology Language for Web Services (OWL-S),[10] the Web Service Modelling Language (WSML),[11] the W3C standard Semantic Annotations for Web Services Description Language (WSDL) and XML Schema (SAWSDL).[12] When it comes to agent specific standardisation efforts, the Foundation for Intelligent Physical Agents (FIPA) propose several standards that support agent to agent communication [98], such as the FIPA Agent Communication Language (ACL)[13] and the FIPA RDF Content Language Specification[14] which describes how RDF can be used to encode the message content.

Additionally, there have been numerous works that focus on using enhancing, and supplementing existing standards, from an intelligent software web agent perspective. In terms of agent specific languages, Wang et al. [88] compare OWL-S, Web Service Modelling Ontology (WSMO)[15] and SAWSDL from the providers, requesters, and brokers perspectives. On the other hand Pai et al. [87] propose a lightweight ontology-based content language based on FIPA RDF Content Language (CL). While, Challenger et al. [32] introduce a semantic web enabled agent modelling Language (SEA_ML), which they apply in the context of an E-barter system. Gibbins et al. [72] propose a process ontology, inspired by the FIPA agent communication language, that describes various message types that can be used to describe web services. Whereas, Kagal et al. [82] demonstrate how policies can be embedded into OWL-S descriptions. When it comes to models and frameworks, Venkatachalam et al. [90] provide a comprehensive survey of existing work on semantic web service (SWS) composition and discovery. More recent works primarily focus on using ontologies to semantically described RESTful web services [99], new approaches for service discovery that leverage user profiles and metadata catalogs [89,91,92], and proposing methodologies that support the modelling and design of SWSs [94]. From an implementation perspective, Zapater et al. [93] demonstrate how the JADE Multi-agent System (MAS) development platform can be enhanced with service discovery capabilities. Verborgh et al. [95] in turn argue that there is a need to construct Web APIs out of reusable building blocks and using hypermedia controls to describe both the functional and

7 https://www.w3.org/TR/soap12-part1/.
8 https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm.
9 http://uddi.xml.org/specification.
10 https://www.w3.org/Submission/OWL-S/.
11 https://www.w3.org/Submission/WSML/.
12 https://www.w3.org/TR/sawsdl/.
13 http://www.fipa.org/specs/fipa00061/index.html.
14 http://www.fipa.org/specs/fipa00011/XC00011B.html.
15 https://www.w3.org/Submission/WSMO/.

non-functional aspects of the service. Whereas from a protocol perspective, Seghir et al. [91] propose web service publication and discovery protocols that are represented in the form of sequence diagrams. While, Shafiq et al. [77] propose an abstract architecture, combining web service and FIPA standardisation efforts, which is capable of translating FIPA ACL to SOAP and visa versa. Others have demonstrated how LDNs can be extended to cater for agent communication [96].

5.2. Reactive component

The `Reactive Component` takes as input a condition and returns an action based on a set of `Condition Action Rules`. More sophisticated reactive components use `State` to further refine the conditions used to determine the action that is required. Table 12 summarises existing work, both in terms of standardisation and community activities.

Condition action rules. When it comes to the specification of condition action rules there are several applicable standardisation efforts. The Production Rule Representation (PRR)[16] specification, developed by the Object Management Group, provides a standard mechanism for encoding rules of the form IF *condition* THEN *action* statements. The Rule Markup Language (RuleML)[17] is a family of languages that provide support for the specification and interchange of both derivation and reaction rules [100]. The W3C Rule Interchange Format (RIF)[18] in turn is an interchange format that can be used to exchange rules between different rule systems. The RIF Production Rule Dialect[19] caters specifically for production rules. While, the W3C Semantic Web Rule Language (SWRL)[20] is a language that combines rules and logic, for a subset of RuleML and a subset of the Web Ontology Language (OWL).[21]

Over the years, researchers have proposed a variety of reactive rules languages that allow for the specification of condition action rules [43,46], event condition action rules [26,45,49,54,55,58], and event condition transaction rules that combines condition action rules with transaction logic [42]. When it comes to models and frameworks, Harth and Käfer [43] and Schraudner and Charpenay [76] propose W3C standard based architectures. While, Ksystra and Stefaneas [49] propose a formal framework for analysing reactive rules in order to safeguard against unpredictable behaviour.

State. A reactive agent with state maintains knowledge about the world and the current state of the environment. The Resource Description Framework (RDF)[22] is a general purpose language that could be used to represent information in a machine interpretable format. The PRR specification describes a metamodel for encoding production rules using the Extensible Markup Language (XML) Metadata Interchange,[23] which is abstract in nature. XML is the native encoding for RuleML, however a JavaScript Object Notation (JSON) serialisation is also provided. Although RIF supports several different encodings, XML is the primary medium of exchange between different rule systems. The SWRL specification uses an abstract Extended Backus–Naur Form (EBNF) syntax, which can easily be encoded in XML and/or RDF.

The agents envisaged by Harth and Käfer [43] model state using RDF, RDFS, and OWL LD. While, Papamarkos et al. [54,55] and Poulovassilis et al. [58] use state to refer to RDF graphs. Boley et al. [26] enumerate several categories that encompass the mental state of an agent, from a memory, commitments, claims, goals, and intentions perspective. Considering their transactional focus, Gomes and Alferes [42] work with a sequence of knowledge base states, which they refer to as paths. The system proposed by Jochum et al. [45] proposes three workflow states: active, inactive and done. In their formal verification framework Ksystra and Stefaneas [49] use model checking to find violated states. Whereas, the basic agent architecture proposed by Schraudner and Charpenay [76] does not maintain state.

Rule interpreter. Although some rule engines provide support for RuleML[24] and SWRL rules (cf., RDFox[25]), when it comes to condition action rules, a simple interpreter that is able to match conditions would suffice. The rule interpreter is responsible for finding rules whose conditions are satisfied, and for triggering the corresponding actions. In the case of conflicting rules, a conflict resolution mechanism is required.

Boley et al. [26] elaborate on the design rationale underpinning RuleML, which provides support for reactive rules, derivative rules, and integrity constraints. Poulovassilis et al. [58] propose an event condition action rule language that can be applied to RDF data, entitled RDF Triggering Language (RDFTL) in the form of an RDF repository wrapper that leverages the repositories querying capabilities. The interpreter used by Papamarkos et al. [54,55] and Poulovassilis et al. [58] includes a parser that performs syntactic validation and a translator that translates queries such that they can be executed by the underlying RDF store. In the system proposed by Harth and Käfer [43] conditions are checked against state, by a SPARQL enabled interpreter, with optional support for some simple RDFS and OWL LD based reasoning. Bonatti et al. [27] in turn propose a reactive policy framework, called Protune, that can be used to guide system behaviour.

5.3. Deliberative component

The `Deliberative Component` takes as input a goal and either returns a solution or an action (that needs to be carried out before a solution can be determined). The `Knowledge Base` is used to store the knowledge the agent has about the world. The `Goal Encoding` is responsible for intercepting the request and updating the knowledge base accordingly. Together the `Reasoning Engine` and the `Utility Function` are responsible for deriving a solution or further actions that need to be fed back to the execution engine. Table 13 summarises relevant standardisation efforts that could be used to realise this component and various community activities that make use of them.

Knowledge base. A deliberative agent maintains knowledge about the world and the current state of the environment in its knowledge base. In the case of our intelligent software web agents, RDF is used to represent information about resources accessible via the web. The RDF Schema[26] specification defines a set of classes and properties used to describe RDF data. However, using RDFS, it is not possible to represent complex statements that include cardinality constraints, or to model complex relations between classes, such as disjointness or equivalence. The OWL Web Ontology Language[27] standard caters for the encoding of

16 https://www.omg.org/spec/PRR/About-PRR/.
17 http://wiki.ruleml.org/index.php/Specification_of_RuleML.
18 https://www.w3.org/TR/rif-overview/.
19 https://www.w3.org/TR/rif-prd/.
20 https://www.w3.org/Submission/SWRL/.
21 https://www.w3.org/TR/2012/REC-owl2-overview-20121211/.
22 https://www.w3.org/TR/rdf11-primer/.
23 https://www.omg.org/spec/XMI/2.5.1/PDF.
24 https://wiki.ruleml.org/index.php/Engines:Master.
25 https://www.oxfordsemantic.tech/.
26 https://www.w3.org/TR/rdf-schema/.
27 https://www.w3.org/TR/owl2-overview/.

Table 12
Intelligent software web agents reactive component.

	Standards	Community activities
Condition Action Rules		
Languages, Ontologies & Vocabularies	PRR, RuleML, RIF, SWRL, RDF, XML	condition action rules [43,46], event condition action rules [26,45,49,54,54,55,58], event condition transaction language [42]
Models & Frameworks	RDF	standards based system architectures [43,76], formal verification framework [49]
State		
Languages, Ontologies & Vocabularies	RDFS, OWL LD	RDF, RDFS, and OWL LD [43], RDF graphs [54,55,58]
Models & Frameworks	RDF	memory, commitments, claims, goals, and intentions [26], sequence of states [42], active, inactive & done workflow state [45], check violated state using model checking [49]
Rule Interpreter		
Languages, Ontologies & Vocabularies	RuleML engine, SPARQL	RuleML design rationale [26], SPARQL enabled interpreter [43], parser and translator [54,55,58]
Models & Frameworks	RDF	workflow meta model [45], Protune policy engine [27]

Table 13
Intelligent software web agents deliberative component.

	Standards	Community activities
Knowledge Base		
Languages, Ontologies & Vocabularies	RDFS, OWL, ODRL	belief-augmented OWL [35], normative language ontology [37,38]
Models & Frameworks	RDF, ODRL	ODRL policy activation & temporal validity [37,38], belief desire intention principles [32], Jason MAS Platform & ontological knowledge [34], ontology integration & automatic reconciliation [101]
Reasoning Engine		
Languages, Ontologies & Vocabularies	RDFS, OWL, ODRL, SPARQL 1.1 Entailment Regimes	OWL reasoning [102], rule based reasoning [103], reasoning over obligations & permissions [37,38]
Models & Frameworks	RDF, ODRL	belief desire intention reasoning [32,34], reasoning over incomplete, subjective & inconsistent data [35]
Goal Encoding		
Languages, Ontologies & Vocabularies	RDFS, OWL	roles, behaviours, plans, beliefs, and goal concepts [32], task, planing & scheduling ontologies [57,59,104],
Models & Frameworks	RDF	constraint logic programming goals [35]
Utility Function		
Languages, Ontologies & Vocabularies	RDFS, OWL	degree of inclination [35], utility values assigned to classes using the uDecide protégé plugin [102]
Models & Frameworks	RDF	utility theory based modelling [103], e-bartering economics [32]

relations between classes, roles and individuals, while at the same time providing support for logical operations and cardinality constraints. Additionally, the Open Digital Rights Language (ODRL)[28] could potentially be used to represent other constraints in the form of policies and norms.

Dong et al. [35] combine OWL with belief augmented frames based logic, which can be used to model evidence for or against a statement. While, Fornara et al. [37] and Fornara and Colombetti [38] describe an OWL based normative language ontology and demonstrate how their ODRL extension caters for policy activation and temporal relevancy can be used by agents to reason about obligations and permissions. The agents envisaged by Challenger et al. [32] are modelled based on the belief–desire intention (BDI) principles [105]. Whereas, Demarchi et al. [34] demonstrates how the Jason multi-agent system development platform [106] can be amended to make use of ontological knowledge available on the web in order to update the agents knowledge base. From an interoperability perspective, Lister et al. [101] propose several alternative strategies that could be used for automatic ontology reconciliation.

Reasoning engine. OWL2[29] comes in two flavours: OWL2 Full and OWL2 DL. In turn, OWL2 DL is composed of three profiles (OWL2EL, OWL2QL and OWL2RL) that are based on well used DL constructs. The syntactic restrictions imposed on each profile are used to significantly simplify ontological reasoning. OWL 2 EL is designed for applications that require very large ontologies, whereby polynomial time reasoning is achieved at the cost of expressiveness. OWL 2 QL is particularly suitable for applications with lightweight ontologies and a large number of individuals, which need to be accessed via relational queries. Finally, OWL 2 RL provides support for applications with lightweight ontologies, and a large number of individuals that make use of rule based inference and constraint mechanisms. Over the years the community has developed several reasoners that are capable of reasoning over OWL ontologies, albeit often with some restrictions (cf., Pellet,[30] HermiT,[31] FACT++,[32] Racer,[33] and RDFox[5.2]). Additionally, SPARQL 1.1 Entailment Regimes[34] can be used to consider implicit (i.e. inferred) data during query execution based on RDF entailment regimes.

Beyond simple OWL based [35,37,38,102] and rule based [57,103] reasoning, researchers have demonstrated the potential for reasoning over beliefs, desires and intentions [32,34], policies and norms [37,38], and incomplete, subjective, and inconsistent data [35].

28 https://www.w3.org/TR/odrl-model/.
29 https://www.w3.org/TR/owl2-overview/.
30 https://www.w3.org/2001/sw/wiki/Pellet.
31 http://www.hermit-reasoner.com/
32 http://owl.cs.manchester.ac.uk/tools/fact/.
33 http://www.ifis.uni-luebeck.de/~moeller/racer/.
34 https://www.w3.org/TR/sparql11-entailment/.

Goal encoding. Given a goal, or set of goals, the agent uses the current state of the world together with the desired state of the world, deduced from its goal(s), in order to infer a solution or further actions that need to be performed. Here RDF, RDFS, and OWL can be used to encode the agents goal(s).

Although there are no standard mechanisms for goal encoding, the domain-specific modelling language proposed by Challenger et al. [32] covers goals and other predominant agent concepts (i.e., roles, behaviours, plans, and beliefs). Additionally, several researchers have proposed task, planing, and scheduling ontologies [57,59,104]. While, the belief framework proposed by Dong et al. [35] uses constraint logic programming goals.

Utility function. The utility function is responsible for assessing possible solutions based on the desired state of the world, and the preferences defined by the person or agent that specifies the goal(s) and associated constraints. According to utility theory [107] informed decisions should be made by examining the goal(s), the actions needed to achieve the goal(s), and the various preferences from a greatest expected satisfaction perspective. Here, a utility theory based modelling, such as that adopted by Brown et al. [108] and Ming et al. [103], could be used to guide the development of the utility function.

Dong et al. [35] define a utility function based on the difference between belief and disbelief values (i.e. the degree of inclination). While, Acar et al. [102] propose a protégé plugin called uDecide that can be used to assign utility values to classes that are subsequently used by the utility function in order to determine the optimal course of action. The template-based ontological method proposed by Ming et al. [103] is rooted in utility theory based modelling [108]. From a domain specific perspective, the semantic web enabled BDI multi-agent system proposed by Challenger et al. [32] builds upon the utility function research specifically focused on the proposed e-bartering system.

5.4. Learning component

The learning component, which is composed of the `Problem Generator`, `Learning Element`, and `Critic`, could be used to develop more advanced intelligent software web agents that are capable of learning from past experiences and thus become more effective over time. This component interacts with both the `Controller Component` and the `Deliberative Component`. The former is responsible for initiating the learning process, while the latter is used to ascertain existing knowledge, perform learning based reasoning tasks, and store the outputs of the learning process. Considering that simple agents (such as the information and booking agents presented in Section 3) do not necessarily need learning capabilities, in the proposed architecture, following a typical separation of duties engineering practice, we separate the learning component from the deliberative component. That being said, it is worth noting that there is a high level of interaction between these components. Although the tools and techniques that could be used to support agent learning have not yet been considered from a standardisation perspective, in Table 14 we summarise preliminary research that could form a starting point for potential standardisation discussions.

Learning element. Although the W3C does not have any specific groups exploring standardisation potential with respect to learning agents, these agents could benefit from many of the standards discussed under the deliberative component. Additionally there has been several related initiatives that could be considered for the realisation of the learning element. For instance, the W3C Ontology-Lexicon Community Group[35] has developed a lexicon model for ontologies[36] that can be used to enrich ontologies with linguistic information. While, the W3C Web Machine Learning Working Group[37] aims to develop Web APIs that enable machine learning in the browser.

From an ontological perspective, both Wong [109] and Puerto et al. [110] demonstrate how various ontology learning techniques can be used to enhance manually crafted ontologies. While, Merkle and Philipp [52] show how reinforcement learning can be used to enhance the policies or strategies used by agents to complete their tasks. That being said, it is worth noting that according to Albrecht and Stone [111] many learning techniques are computationally complex making them unsuitable for many real world use case scenarios. When it comes to the agent learning semantic web models and frameworks, Leite et al. [50] and Leite and Girardi [51] propose high level ontology-driven hybrid agent architectures that include separate problem generator, critic and learning components that are used by the deliberative component in order to improve both the deliberative knowledge base and the reactive rules. While, Young et al. [112] demonstrate how spatial information about unknown objects together with their semantic web meaning can be used by robots to classify the unknown object. Ghanadbashi and Golpayegani [41] in turn introduce their automatic goal generation model and a corresponding workflow that enables agents to evolve existing goals or create new goals based on emerging requirements. More broadly, Asim et al. [113] highlight that ontology learning has benefited from a variety of domains, namely natural language processing, machine learning, information retrieval, data mining and knowledge representation. The authors perform a comprehensive survey of existing work, categorising them as linguistic, statistic, and logic based.

Problem generator & critic. According to Russel and Norvig [22] the problem generator suggests actions that will lead to learning in the form of new knowledge and experiences. While the critic provides feedback to the agent in the form of a reward or a penalty. Although the problem generator and the critic could vary greatly from an internal implementation perspective, there is a need for standardised vocabularies and APIs that can be used to manages synchronisation and communication between the various internal and external components.

From a vocabularies perspective, there has been some relevant work in terms of robotics, whereby Buoncompagni et al. [31] and Kootbally et al. [48] demonstrate how the Planning Domain Definition Language (PDDL)[38] problem generator can make use of OWL reasoners to check for and solve issues with respect to norm compliance. As for models and frameworks, Leite et al. [50] and Leite and Girardi [51] present architectures whereby the agents perceive the effects that their actions have on the environment, pass this information to the critic, which in turn informs the learning component about poor performance. The learning component also recommends improvements and the problem generator is responsible for proposing new actions based on these recommendations. Van Riemsdijk et al. [65] focus on the weaker notion of norm compliance and propose a semantic framework that demonstrates how agents identify problems and adapt their behaviour in order to avoid violating norms.

5.5. Controller component

The `Controller Component` is responsible for interpreting perceptions from sensors via the `Perceptions Interpreter`,

35 https://www.w3.org/community/ontolex/.

36 https://www.w3.org/2016/05/ontolex/.

37 https://www.w3.org/2021/04/web-machine-learning-charter.html.

38 https://helios.hud.ac.uk/scommv/IPC-14/repository/kovacs-pddl-3.1-2011.pdf.

Table 14
Intelligent software web agents learning component.

	Standards	Community activities
Problem Generator & Critic		
Languages, Ontologies & Vocabularies	RDFS, OWL	OWL & PDDL [31,48]
Models & Frameworks	RDF	critic, learning element & problem generator [50], adaptive behaviour & norms [31,65]
Learning Element		
Languages, Ontologies & Vocabularies	OWL	ontology learning [109,110], reinforcement learning & policies/strategies [52], computationally complex [111]
Models & Frameworks Models	RDF	critic, learning element & problem generator [50,51], spatial information & semantic web mining [112], automatic goal generation model [41], linguistic, statistic and logic based [113]

Table 15
Intelligent software web agents controller component.

	Standards	Community activities
Perception & Action Interpreters		
Languages, Ontologies & Vocabularies	FIPA_Contract_Net	messages & message sequences [32], generic planning ontology [57],
Models & Frameworks		ontological representation [50,51], semantic descriptions for unknown objects [112]
Execution Engine		
Languages, Ontologies & Vocabularies	OWL, OWL-S, ODRL	control via policies [64], Protune policy engine [27], normative language ontology [37,38], agent modelling language [32], legal ontology [86]
Models & Frameworks	LDP	KAoS, Rei, Ponder comparison [64], context broker architecture [66], agent platform comparison [114], JACK & OWL-S [32], abstract state machines, Linked Data-Fu & LDP [46], RDF/RDFS & RuleML [26], Jason interpreter [34]

devising execution plans that leverage the reactive and deliberative components, and executing the plans via the `Execution Plan Management`. In addition, this component is responsible for advising the actuators what action(s) need to be taken via the `Actions and Solutions Interpreter`. Although the tools and techniques that are needed to support agent control have not yet matured in terms of W3C standardisation efforts, in Table 15 we summarise preliminary research that could form the basis of initial standardisation discussions.

Perception & action interpreters. Irrespective of whether we are dealing with a web service or a web application, there is a need to define interfaces that can be used to interact with the agent. The perception interpreter is responsible for forwarding perceptions to the `Execution Engine`, while the action interpreter in turn is responsible for initiating actions forwarded by the `Execution Engine`. When it comes to simple read and write operations, the Linked Data Platform (LDP)[39] specification provides a set of best practices for an architecture that supports accessing, updating, creating and deleting Linked Data resources. However, said architecture would need to be amended to provide support for additional functions required in order to cater for interaction between intelligent agents.

Challenger et al. [32] discuss the role played by messages and message sequences when it comes to agent interaction and highlight that they could be based on some standard such as FIPA_Contract_Net. More broadly, there are a range of FIPA standards[40] that could potentially be leveraged by intelligent software web agents. From a goal encoding perspective, Pham and Stacey [57] propose an ontology than can be used for modelling planning problems that could be worked on by goal driven agents. In the frameworks proposed by Leite et al. [50] and Leite and Girardi [51] perceptions and actions are represented as ontologies, however the authors focus on the general framework as opposed to the languages, vocabularies and ontologies that could be used for modelling perceptions and actions. Young et al. [112] in turn focus on leveraging external knowledge bases in order to obtain semantic descriptions for unknown objects.

Execution engine. The `Execution Engine` is responsible for routing conditions to the `Reactive Component` and goals to the `Deliberative Component`, thus the component needs to be able to handle both real-time and delayed responses. Although there is a lack of specific W3C standardisation activity concerning the execution engine, the semantic web community have proposed several tools and technologies that make use of web standards.

Boley et al. [26] discuss how RDF/RDFS and RuleML can together be used to develop simple reactive software agents. Tonti et al. [64] investigate how policies can be used to control agent behaviour by separating a systems functional and governance aspects. The authors compare and contrast the policy management approaches of the KAoS [115,116], Rei [82,117] and Ponder [118] policy languages and frameworks when it comes to controlling communication. Chen et al. [66] also focus on policy enforcement, proposing a context broker architecture that can be used to control the sharing and use of personal data. When it comes to general policy languages, the Protune policy engined proposed by Bonatti et al. [27] has also been used to control reactive behaviour. Fornara et al. [37] and Fornara and Colombetti [38] in turn propose a normative language ontology, derived from the ODRL standard, that could be used to control agent behaviour. While, Palmirani et al. [86] introduce their legal ontology that could be used to design privacy preserving intelligent agents. Poulovassilis et al. [58] propose an abstract architecture that could guide the development of an event-condition-action reactive agent. Whereas, Käfer and Harth [46] use abstract state machines in order to model the internals of their reflexive agents and demonstrate the effectiveness of their proposal using Linked

39 https://www.w3.org/TR/ldp/.

40 http://www.fipa.org/specs/fipa00025/XC00025E.html, http://www.fipa.org/repository/ips.php3

Data-Fu[41] together with their Linked Data Platform implementation.[42] Challenger et al. [32] propose a platform independent multi-agent system development methodology and demonstrate its effectiveness using the JACK multi-agent system development framework together with OWL-S models. Whereas, Demarchi et al. [34] demonstrate how the Jason interpreter can be adapted to benefit from ontological knowledge. More broadly, Pal et al. [114] provide a comprehensive review of platforms that can be used to develop agent based systems, however their suitability for developing intelligent web agents is still and open area of research.

6. Intelligent software web agents: The future

The goal of this section is to use insights gained from the analysis of the intelligent software web agent requirements and the hybrid agent architecture components, in order to highlight existing research opportunities and challenges. In addition, we take a broader perspective of the research by discussing the potential for intelligent software web agent as an enabling technology for emerging domains, such as digital assistants, cloud computing, and the internet of things.

6.1. Opportunities and challenges

A condensed overview of the intelligent software web agents requirements analysis (focusing on the scheduling agent, which is the most complicated out of the three agents we examined), their impact from an architectural perspective, and the corresponding opportunities and challenges discussed below is presented in Table 16.

Core aspects of the hybrid agent architecture. The reactivity, proactiveness, interoperability, and communication requirements are classified as core modules that are inherent to the hybrid agent architecture presented in the previous section. When it comes to instantiating the architecture there are several standards and technologies that could be leveraged in order to realise the `Interface`, `Reactive`, and `Deliberative Components`. Additionally, over the years the research community have proposed various approaches for semantic web service discovery and composition methods; event condition action rule languages and frameworks; and approaches for representing and reasoning over roles, behaviours, norms, beliefs, goals and plans, that could serve as a basis for developing a simple scheduling agent prototype. However, the suitability of the various proposals from both a practical perspective and a performance and a scalability perspective has yet to be determined.

Other challenges relate to the development of the `Controller Component`, which is responsible for internal co-ordination. Existing proposals have focused on defining messages and message protocols; proposing ontologies for specifying norms and legal requirements; and controlling agent behaviour via policies. Unfortunately, much of the work has focused on basic technology research, and many of the proposals have not been validated via prototyping. Here a reference architecture [119] could serve to bridge the gap between theory and practice and to identify potential open challenges that still need to be addressed from an architecture perspective.

Additionally, the `Learning Component`, which is often discussed in the context of learning agents or normative/policy agents, has received little attention to date. Broadly speaking, existing proposals focus on using ontology or reinforcement learning techniques to enhance the agents knowledge base, or demonstrating how agents can adapt their behaviour based on changes in the environment. Here again, there is the need to determine the effectiveness of existing proposal in the form of a prototype. When it comes to multi-agent learning in general, there are several survey articles (cf., [120–122]) that could serve as the basis for the development of this component. While, from a practical implementation perspective, further research is needed to better understand its role in the overall architecture and what are the concrete standardisation needs.

Task specific considerations. Both the responsibility and verification requirements has been classified as task specific. In our motivating use case scenario, we identified three different types of agents, namely information agents, booking agents, and scheduling agents. Clearly this is not an exhaustive list of agent types, however considering the original semantic web vision has not yet been realised it is beneficial to revisit this simple use case, before moving on to more complex scenarios that can leverage intelligent web agents. The key take home from an architectural perspective, is that we should strive to develop optional task modules that could serve a variety of use case scenarios. Here again there is an need to consider how such task specific modules could be integrated into a reference architecture and described in detail from a system design perspective. The agent task environment requirements assessment framework proposed herein can be used not only to perform a detailed analysis of various agent based use case scenarios, but also to better understand the potential solutions and the technological and standardisations gaps that still exist. Interesting directions for future work include adopting software engineering requirements elicitation techniques [123–125] and lessons learned [126] in order to better understand the various use case requirements.

Cross cutting generic considerations. The behavioural functions (i.e., benevolence, rationality, and mobility), code of conduct functions (i.e., identification, security, privacy, trust, and ethics) and two of the basic functions (i.e., autonomy and social ability) have been classified as cross cutting, as they need to be considered when it comes to the architecture as a whole and also the individual components. Ideally they should be integrated into a reference architecture in the form of optional generic modules that can be used by the agent depending on the task that needs to be carried out. Following common engineering practices, these modules would need to be described from a system design perspective.

In the early days of semantic web research, the behavioural functions (i.e., benevolence, rationality, and mobility) received some interest from intelligent software web agent researchers. In particular, researchers highlighted the need for agents to be benevolent and rational by design, to be capable of balancing self interest and group interests, to take on various roles and responsibilities, and to the need to cater for mobility from a robustness perspective. However, when it comes to the proposed tools, technologies, and standards benevolence, responsibility, and mobility requirements were not even mentioned in the corresponding papers. The lack of recent research in terms of intelligent software web agents behavioural functions is indicative of the communities diversification of interests and the need to better understand the needs of intelligent software web agents both from semantics and a deployment perspective, as argued by Bernstein et al. [4]. The analysis of the intelligent software web agent requirements, and the standards, tools and technologies presented here-in is a first step towards better understanding the status quo and the

41 https://linked-data-fu.github.io/.
42 https://github.com/kaefer3000/ldbbc/.

Table 16
Intelligent software web agents requirements assessment.

Functions	Use case requirements	Architectural impact	Opportunities	Challenges
Basic Functions				
Autonomy	consult relevant sources, devise an optimal schedule	cross cutting	ontology learning and reinforcement learning techniques	adopt a learning theory perspective
Reactivity	immediate response where possible	core	event condition action rule languages and frameworks	reference architecture
Pro-activeness	scheduling goal, explore alternatives	core	well established standards, techniques for representing & reasoning over roles, behaviours, norms, beliefs, goals & plans	reference architecture
Social ability	humans and agents	cross cutting	policy & norm languages	virtual organisations management techniques, policy & norm standards
Behavioural Functions				
Benevolence	well meaning by design, manage conflicting goals	cross cutting	–	benevolent by design
Rationality	rational by design	cross cutting	–	rational by design
Responsibility	mange access to information, finds optimal schedule given a set of constraints	task specific	task environment requirements assessment	requirements elicitation techniques
Mobility	interacts with several other agents	cross cutting	–	discovery of services in mobile and ubiquitous environments, technological advances supporting mobility
Collaborative Functions				
Interoperability	agreed/common schema	core	established standards	reference architecture
Communication	push and pull requests	core	established standards, norms & policies	reference architecture
Brokering services	collects information from a variety of sources	cross cutting	established standards, semantic web service discovery & composition techniques	reference architecture
Inter-agent co-ordination	agents support each other via information sharing motivating	cross cutting	policies & norms	virtual organisations management techniques, policy & norm standards
Code of Conduct Functions				
Identification	handle public and private information, may need to prove who they represent	cross cutting	unique identifiers & authentication mechanisms	agent architectures adaptation
Security	protect against unauthorised access, inappropriate use, and denial of service	cross cutting	access control, encryption	agent architectures adaptation
Privacy	handle personal information appropriately	cross cutting	privacy policies, anonymisation	agent architectures adaptation
Trust	manages information and scheduling accuracy, consults reliable sources	cross cutting	trust frameworks and architectures	agent architectures adaptation
Ethics	do no harm by design	cross cutting	legal policies, norms, general guidelines	agent architectures adaptation
Robustness Functions				
Stability	available, reliable & secure	robustness	–	attacker models
Performance	real time access to information, timely goal completion	robustness	–	agent benchmarking tools
Scalability	handles increasing requests, data, & task complexity	robustness	–	agent benchmarking tools
Verification	checks information is correct, the reasoning is explainable	task specific & robustness	task environment requirements assessment, formal method, simulation	requirements elicitation techniques, test driven development

requirements that should guide agents that leverage semantic web technologies.

When it comes to the code of conduct functions, there is a body of work from the semantic web community that has not been applied directly to the semantic web agent use case, that could potentially be leveraged in order to realise the proposed architecture. In the following, we identify several interesting works that could potentially inform the design of our intelligent software web agents. Broadly speaking, existing work in terms of identification focuses on access control for RDF [127–133] or demonstrating how policy languages can be used for the specification and enforcement of access restrictions [115,134,135]. Besides access control, security based research has primarily focused on applying encryption algorithms [136–139] and digital signatures [140] to RDF data. Work on privacy primarily focuses on applying and extending existing anonymisation techniques such that they work with graph data [141–144] or catering for the specification and enforcement of privacy preferences [145,146]. When it comes to trust, Artz and Gil [25] conducted a survey of existing trust mechanisms in computer science in general, and the Semantic Web in particular. In addition, several authors have proposed trust frameworks and architectures [147–150]. When it comes to ethics, Gordon et al. [151] focus on requirements that are necessary for modelling and reasoning over legal rules and regulations, whereas Palmirani et al. [152] extend RuleML in the

form of LegalRuleML such that it can be used to model and reason over both legal norms and business rules. More generally, existing work at the intersection of intelligent agents and ethics [153, 154] or behavioural aspects of intelligent agents [11,155] could provide insights into the detailed design of these cross cutting generic modules. Interestingly, the *Ethics Guidelines for Trustworthy AI* [156], recently released by the European Commission only briefly mentions agent technologies, instead focusing on artificial intelligence in general. Thus, while this document serves as a useful starting point with respect to codes of conduct for agents, further work is needed to make these guidelines actionable from an intelligent agents perspective.

Basic agent functional requirements relating to autonomy and social ability serve as motivation for research concerning the role of policies and norms when it comes to controlling intelligent software web agent behaviour. However, there has been limited research by the semantic web community in terms of developing truly autonomous agents that are capable of interacting with other agents and the tools, technologies, and standards needed to enable agents to form virtual organisations in order to collaboratively solve problems. Beyond the semantic web community Van Der Vecht et al. [157] propose gradual levels of autonomy that can be catered for via commitments and contracts. More generally, the technical opportunities and challenges relating to the field of agent based computing, identified by Luck et al. [6], are also relevant from an intelligent software web agents perspective. Primary considerations include: viewing autonomy from a learning theory perspective and examining social ability in terms of virtual organisations. The authors also highlight the need for the advancement of tools and technologies to support scalable service discovery and composition, and semantic integration and additional research in terms of transparency, trust, reputation, and negotiation.

Robustness considerations. All four robustness functions (i.e., stability, performance, scalability, and verification) have simply been classified as robustness from an architectural perspective. These requirements need to be considered both when it comes to the detailed design of the system and the choice of technologies. In the proposed architecture the `Performance Assessment` entity which is part of the `Task Environment` is responsible for evaluating the effectiveness of the system from both a functional and a non-functional perspective.

Several of the requirements papers identified the need to evaluate existing proposals in terms of stability, performance and scalability. However, when it comes to the development of tools, technologies, and standards that could be used to evaluate the effectiveness of existing proposals, researchers have primarily focused on developing proof of concepts in the form of basic simulations or assessing formal aspects such as correctness, safety, and compliance. Here again, we see evidence that intelligent software web agent research is more foundational than applied. Considering the crucial role played by both functional and non functional testing from an engineering perspective, there is a need to develop testing strategies and benchmarks in order to advance the research further. More broadly, when it comes to measuring robustness, besides an array of individual performance evaluations for various query and reasoning engines, there is a body of work in relation to benchmarking that could be used/extended in order to benchmark the proposed architecture. For instance, there are a number of well established benchmarks, such as the Lehigh University Benchmark (LUBM) [158] or the Berlin SPARQL Benchmark (BSBM) [159] and promising newcomers, such as the Linked Data Benchmark Council (LDBC) Social Network Benchmark [160]. In addition, it may be possible to leverage existing benchmarking frameworks, such as the general entity annotator benchmarking framework (GERBIL) [161] or the holistic benchmarking for big linked data framework (HOBBIT) [162].

6.2. Semantic web agents as an enabling technology

Moving beyond the original intelligent software web agent motivating scenario, the tools, technologies, and standards discussed herein could potentially have a much broader impact. For instance, according to Luck et al. [6], the semantic web community provides a semantically rich data model, vocabularies, and ontologies that can be used to describe media and services in a manner that facilitates discovery and composition; and allows for agent to agent information exchange. In the following, we move beyond the original motivating scenario by highlighting the potential impact of intelligent software web agents on emerging domains, such as digital assistants, cloud computing, and the internet of things.

Digital assistants. Although well known voice assistants, such as Siri, Alexa, and Cortana, are not as sophisticated as the Knowledge Navigator concept proposed by Sculley [163] (when he was the chief executive officer at Apple) the technology has been embedded in various smart home and smart phone products. Common features include sending and receiving text messages and emails, making calls, setting timers and reminders, and control of hardware (e.g., thermostats, lights, audio, video) [164]. However, in order to realise the Sculley's Knowledge Navigator these voice assistants need to be enhanced with data discovery and reasoning capabilities, which are at the core of envisaged intelligent software web agents. Considering the sensitive personal nature of the data often captured by such agents, intelligent software web agents could also be employed in order to provide users with more control and transparency with respect to personal data processing.

Cloud computing. Cloud computing has been around for quite some time, however as technology rapidly evolves so too does the service offering, for instance edge computing is a paradigm whereby computation is performed closer to where the data is consumed [165]. New data infrastructure initiatives, such as GAIA-X [166], envisage virtual data spaces developed on top of federated infrastructure (including high performance computing and edge systems), where data sovereignty and secure exchange are built-in by design. Recently, the term private 5G networks is used to refer to industrial networks that require increased reliability, low latency, and strong security [164]. In this context, intelligent software web agents could potentially play a major role both from a resource allocation and a governance perspective. In the case of the former, agents could take on a coordinating role when it comes to virtual organisation/private network formation and monitoring. In the case of the latter, both data and service providers could encode usage constraints and provenance trails using policy languages and ontologies in a manner that supports agent based negotiation and automated compliance checking.

The internet of things. The W3C Web of Things initiative[43] focuses on building on existing Web standards in order to facilitate data integration across various IoT platforms. Here, semantic technologies have already been used in order to describe things[44] and facilitate thing discovery.[45] When it comes to the intersection of intelligent software web agents and the internet of things, semantic web agents could also play a crucial role in terms of coordinating the usage, management, and governance of things. Additionally, the standards, tools, and technologies discussed herein could provide support for analytics needed in order to optimise supply and value chains that make use of IoT technologies.

[43] https://www.w3.org/WoT/.
[44] https://www.w3.org/TR/wot-thing-description/.
[45] https://www.w3.org/TR/wot-discovery/.

7. Conclusions

Motivated by the desire to further advance existing research into intelligent software web agents, in this paper we revisited the original use case scenario proposed in the seminal semantic web paper from a gap analysis perspective. We started by collating and summarising requirements and core architectural components relating to intelligent software agents in general. Following on from this, we used the intelligent software agent requirements to both further elaborate on the semantic web agent motivating use case scenario, and to summarise and classify existing semantic web agent literature. We subsequently used the insights gained in order to propose a hybrid semantic web agent architecture that guided our discussion with respect to relevant standards, tools, and technologies. Following on from this, we used the functional and non-functional agent requirements together with the scheduling agent use case requirements to better understand the opportunities and challenges concerning the realisation of intelligent software web agents. Finally, we broadened the discussion and highlighted the potential of intelligent software web agent as an enabling technology for digital assistants, cloud computing, and the internet of things.

Key outputs include: (i) a task environment requirements assessment framework, based on agent requirements gleaned from the literature, that could be used to perform an in-depth assessment of various agent use case scenarios; and (ii) a hybrid architecture and the corresponding assessment of existing standards, tools, and technologies, which serves as the basis for developing a reference architecture that can be used to realise the original intelligent software web agent vision and to build the foundations needed in order to support more complex use case scenarios.

Based on our analysis, there are a number of gaps that still need to be addressed in order to move the intelligent software web agent vision forward. Firstly, from an architectural perspective, there is a need to develop a reference architecture that could serve to bridge the gap between theory and practice, and to identify potential open research challenges that still need to be addressed. Secondly, from an implementation perspective there is a need to better understand the specific requirements relating to the cross cutting behavioural functions (i.e., benevolence, rationality, and mobility), code of conduct functions (i.e., identification, security, privacy, trust, and ethics), and basic functions (i.e., autonomy, and social ability), the adaptations/extensions needed to existing tools and technologies, and insights into how these tools and technologies fit together with core intelligent software agent technologies and with each other. Finally, from a robustness perspective, there is a need to develop/extend existing benchmarks such that they can be used to both validate and assess the performance and scalability of various instantiations of our hybrid agent architecture.

CRediT authorship contribution statement

Sabrina Kirrane: Conceptualization, Methodology, Investigation, Writing – original draft, Writing – review & editing, Visualization, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This research is funded by the FWF Austrian Science Fund and the Internet Foundation Austria under the FWF Elise Richter and netidee SCIENCE programmes as project number V 759-N.

References

[1] T. Berners-Lee, J. Hendler, O. Lassila, The semantic web, Sci. Am. 284 (5) (2001) 34–43.
[2] J. Hendler, Agents and the semantic web, IEEE Intell. Syst. 16 (2) (2001) 30–37.
[3] J. Hendler, Where are all the intelligent agents? IEEE Ann. Hist. Comput. 22 (03) (2007) 2–3.
[4] A. Bernstein, J. Hendler, N. Noy, A new look at the semantic web, Commun. ACM (2016).
[5] S. Kirrane, M. Sabou, J.D. Fernández, F. Osborne, C. Robin, P. Buitelaar, E. Motta, A. Polleres, A decade of semantic web research through the lenses of a mixed methods approach, Semant. Web 11 (6) (2020) 979–1005.
[6] M. Luck, P. McBurney, O. Shehory, S. Willmott, Agent Technology: Computing as Interaction (A Roadmap for Agent Based Computing), Technical Report, University of Southampton, 2005.
[7] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. de Melo, C. Gutierrez, J.E.L. Gayo, S. Kirrane, S. Neumaier, A. Polleres, R. Navigli, A.-C.N. Ngomo, S.M. Rashid, A. Rula, L. Schmelzeisen, J. Sequeda, S. Staab, A. Zimmermann, Knowledge graphs, 2021, arXiv:2003.02320.
[8] R. Whittemore, K. Knafl, The integrative review: updated methodology, J. Adv. Nurs. 52 (5) (2005) 546–553.
[9] R.J. Torraco, Writing integrative literature reviews: Guidelines and examples, Hum. Resour. Dev. Rev. 4 (3) (2005) 356–367.
[10] J.P. Müller, Architectures and applications of intelligent agents: A survey, Knowl. Eng. Rev. 13 (4) (1998) 353–380.
[11] M.J. Wooldridge, N.R. Jennings, Intelligent agents: Theory and practice, Knowl. Eng. Rev. 10 (2) (1995) 115–152.
[12] C. Castelfranchi, Guarantees for autonomy in cognitive agent architecture, in: International Workshop on Agent Theories, Architectures, and Languages, Springer, 1994, pp. 56–70.
[13] P. Maes, Artificial life meets entertainment: lifelike autonomous agents, Commun. ACM 38 (11) (1995) 108–114.
[14] S. Franklin, A. Graesser, Is it an agent, or just a program?: A taxonomy for autonomous agents, in: International Workshop on Agent Theories, Architectures, and Languages, Springer, 1996, pp. 21–35.
[15] N. Jennings, M. Wooldridge, Software agents, IEE Rev. 42 (1) (1996) 17–20.
[16] M.R. Genesereth, N.J. Nilsson, Logical Foundations of Artificial Intelligence, Morgan Kaufmann, 2012.
[17] J.S. Rosenschein, M.R. Genesereth, Deals among rational agents, in: Readings in Distributed Artificial Intelligence, Elsevier, 1988, pp. 227–234.
[18] B. Hermans, Intelligent software agents on the internet: Chapters 6-7, First Monday (1997) URL https://journals.uic.edu/ojs/index.php/fm/article/download/516/437.
[19] J. White, Telescript Technology: The Foundation for the Electronic Marketplace, Technical Report, General Magic, 1994, URL http://www.datarover.com/Telescript/Whitepapers/wp1/whitepaper-1.html.
[20] M. Wooldridge, N.R. Jennings, Agent theories, architectures, and languages: a survey, in: International Workshop on Agent Theories, Architectures, and Languages, Springer, 1994, pp. 1–39.
[21] R. Girardi, A. Leite, A survey on software agent architectures, IEEE Intell. Inf. Bull. 14 (1) (2013) 8–20.
[22] S. Russel, P. Norvig, Artificial Intelligence: A Modern Approach, Pearson Education Limited, 2013.
[23] A. Newell, H.A. Simon, Computer science as empirical inquiry: Symbols and search, in: ACM Turing Award Lectures, 2007, p. 1975.
[24] J. Bryson, Cross-paradigm analysis of autonomous agent architecture, J. Exp. Theoret. Artif. Intell. 12 (2) (2000) 165–189.
[25] D. Artz, Y. Gil, A survey of trust in computer science and the semantic web, J. Web Semant. 5 (2) (2007) 58–71.
[26] H. Boley, S. Tabet, G. Wagner, Design rationale for RuleML: A markup language for semantic web rules., in: SWWS, vol. 1, 2001, pp. 381–401.
[27] P.A. Bonatti, P. Kärger, D. Olmedilla, Reactive policies for the semantic web, in: Extended Semantic Web Conference, Springer, 2010, pp. 76–90.
[28] J.J. Bryson, D.L. Martin, S.A. McIlraith, L.A. Stein, Toward behavioral intelligence in the semantic web, Computer 35 (11) (2002) 48–54.
[29] J.J. Bryson, D. Martin, S.A. McIlraith, L.A. Stein, Agent-based composite services in DAML-s: The behavior-oriented design of an intelligent semantic web, in: Web Intelligence, Springer, 2003, pp. 37–58.
[30] P.A. Buhler, J.M. Vidal, Towards adaptive workflow enactment using multiagent systems, Inf. Technol. Manag. 6 (1) (2005) 61–87.

[31] L. Buoncompagni, A. Capitanelli, F. Mastrogiovanni, A ROS multi-ontology references services: OWL reasoners and application prototyping issues, 2017, arXiv preprint arXiv:1706.10151.
[32] M. Challenger, B.T. Tezel, O.F. Alaca, B. Tekinerdogan, G. Kardas, Development of semantic web-enabled BDI multi-agent systems using SEA_ML: An electronic bartering case study, Appl. Sci. 8 (5) (2018) 688.
[33] D.K. Chiu, H.-f. Leung, Towards ubiquitous tourist service coordination and integration: a multi-agent and semantic web approach, in: Proceedings of the 7th International Conference on Electronic Commerce, pp. 574–581, 2005.
[34] F. Demarchi, E.R. Santos, R.A. Silveira, Integration between agents and remote ontologies for the use of content on the semantic web., in: ICAART, vol. 1, 2018, pp. 125–132.
[35] J.S. Dong, Y. Feng, Y.-F. Li, C.K.-Y. Tan, B. Wadhwa, H.H. Wang, BOWL: augmenting the semantic web with beliefs, Innov. Syst. Softw. Eng. 11 (3) (2015).
[36] V. Ermolayev, N. Keberle, S. Plaksin, O. Kononenko, V. Terziyan, Towards a framework for agent-enabled semantic web service composition, Int. J. Web Serv. Res. 1 (3) (2004) 63–87.
[37] N. Fornara, A. Chiappa, M. Colombetti, Using semantic web technologies and production rules for reasoning on obligations and permissions, in: International Conference on Agreement Technologies, Springer, 2018, pp. 49–63.
[38] N. Fornara, M. Colombetti, Using semantic web technologies and production rules for reasoning on obligations, permissions, and prohibitions, AI Commun. 32 (4) (2019) 319–334.
[39] F. García-Sánchez, J.T. Fernández-Breis, R. Valencia-García, J.M. Gómez, R. Martínez-Béjar, Combining semantic web technologies with multi-agent systems for integrated access to biological resources, J. Biomed. Inform. 41 (5) (2008) 848–859.
[40] F. García-Sánchez, R. Valencia-García, R. Martínez-Béjar, J.T. Fernández-Breis, An ontology, intelligent agent-based framework for the provision of semantic web services, Expert Syst. Appl. 36 (2) (2009) 3167–3187.
[41] S. Ghanadbashi, F. Golpayegani, Using ontology to guide reinforcement learning agents in unseen situations, Appl. Intell. (2021) 1–17.
[42] A.S. Gomes, J.J. Alferes, A procedure for an event-condition-transaction language, in: International Conference on Web Reasoning and Rule Systems, Springer, 2015, pp. 113–129.
[43] A. Harth, T. Käfer, Specifying and executing user agent behaviour with condition-action rules, in: Proceedings of the 1st Workshop on Decentralizing the Semantic Web Co-Located with the 16th International Semantic Web Conference, 2017.
[44] M.N. Huhns, Agents as web services, IEEE Internet Comput. 6 (4) (2002) 93–95.
[45] B. Jochum, L. Nürnberg, N. Aß falg, T. Käfer, Data-driven workflows for specifying and executing agents in an environment of reasoning and restful systems, in: International Conference on Business Process Management, Springer, 2019, pp. 93–105.
[46] T. Käfer, A. Harth, Rule-based programming of user agents for linked data, in: LDOW@ WWW, 2018.
[47] A. Khalili, A.H. Badrabadi, F. Khoshalhan, A framework for distributed market place based on intelligent software agents and semantic web services, in: 2008 IEEE Congress on Services Part II (Services-2 2008), IEEE, 2008, pp. 141–148.
[48] Z. Kootbally, T.R. Kramer, C. Schlenoff, S.K. Gupta, Overview of an ontology-based approach for kit building applications, in: 2017 IEEE 11th International Conference on Semantic Computing (Icsc), IEEE, 2017, pp. 520–525.
[49] K. Ksystra, P. Stefaneas, Formal analysis and verification support for reactive rule-based web agents, Int. J. Web Inf. Syst. (2016).
[50] A. Leite, R. Girardi, P. Novais, Using ontologies in hybrid software agent architectures, in: 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), vol. 3, IEEE, 2013, pp. 155–158.
[51] A. Leite, R. Girardi, A case-based reasoning architecture of a hybrid software agent, in: 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), vol. 3, IEEE, 2014, pp. 79–86.
[52] N. Merkle, P. Philipp, Cooperative web agents by combining semantic technologies with reinforcement learning, in: Proceedings of the 10th International Conference on Knowledge Capture, 2019, pp. 205–212.
[53] M. Paolucci, K. Sycara, Autonomous semantic web services, IEEE Internet Comput. 7 (5) (2003) 34–41.
[54] G. Papamarkos, A. Poulovassilis, P.T. Wood, Event-condition-action rule languages for the semantic web, in: Proceedings of the First International Conference on Semantic Web and Databases, Citeseer, 2003, pp. 294–312.
[55] G. Papamarkos, A. Poulovassilis, P.T. Wood, RDFTL: An event-condition-action language for RDF, in: Proc. of the 3rd International Workshop on Web Dynamics, 2004.
[56] T.R. Payne, Web services from an agent perspective, IEEE Intell. Syst. 23 (2) (2008) 12–14.
[57] H. Pham, D. Stacey, Practical goal-based reasoning in ontology-driven applications, in: KEOD, 2011, pp. 99–109.
[58] A. Poulovassilis, G. Papamarkos, P.T. Wood, Event-condition-action rule languages for the semantic web, in: International Conference on Extending Database Technology, Springer, 2006, pp. 855–864.
[59] D. Rajpathak, E. Motta, An ontological formalization of the planning task, in: International Conference on Formal Ontology in Information Systems (FOIS 2004), 2004, pp. 305–316.
[60] K. Sycara, M. Paolucci, J. Soudry, N. Srinivasan, Dynamic discovery and coordination of agent-based semantic web services, IEEE Internet Comput. 8 (3) (2004) 66–73.
[61] V. Tamma, T.R. Payne, Is a semantic web agent a knowledge-savvy agent? IEEE Intell. Syst. 23 (4) (2008) 82–85.
[62] V. Tamma, I. Blacoe, B.L. Smith, M. Wooldridge, Serse: searching for semantic web content, in: ECAI, vol. 16, 2004, p. 63.
[63] V. Terziyan, SmartResource–PRoactive self-maintained resources in semantic web: Lessons learned, Int. J. Smart Home 2 (2) (2008) 33–57.
[64] G. Tonti, J.M. Bradshaw, R. Jeffers, R. Montanari, N. Suri, A. Uszok, Semantic Web languages for policy representation and reasoning: A comparison of KAoS, Rei, and Ponder, in: International Semantic Web Conference, Springer, 2003, pp. 419–437.
[65] M.B. Van Riemsdijk, L. Dennis, M. Fisher, K.V. Hindriks, A semantic framework for socially adaptive agents: Towards strong norm compliance, in: Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, Citeseer, 2015, pp. 423–432.
[66] H. Chen, T. Finin, A. Joshi, L. Kagal, F. Perich, D. Chakraborty, Intelligent agents meet the semantic web in smart spaces, IEEE Internet Comput. 8 (6) (2004) 69–79.
[67] F. Gandon, The web we mix: Benevolent AIs for a resilient web, in: Proceedings of the 10th ACM Conference on Web Science, 2019, pp. 115–116.
[68] D.N. Jutla, P. Bodorik, Y. Zhang, PeCAN: An architecture for users' privacy-aware electronic commerce contexts on the semantic web, Inf. Syst. 31 (4–5) (2006) 295–320.
[69] F. Scioscia, M. Ruta, G. Loseto, F. Gramegna, S. Ieva, A. Pinto, E. Di Sciascio, A mobile matchmaker for the Ubiquitous Semantic Web, Int. J. Semant. Web Inf. Syst. (IJSWIS) 10 (4) (2014) 77–100.
[70] M. Sheshagiri, N. Sadeh, F. Gandon, Using semantic web services for context-aware mobile applications, in: MobiSys 2004 Workshop on Context Awareness, Citeseer, 2004.
[71] A. Outtagarts, Mobile agent-based applications: A survey, Int. J. Comput. Sci. Netw. Secur. 9 (11) (2009) 331–339.
[72] N. Gibbins, S. Harris, N. Shadbolt, Agent-based semantic web services, in: Proceedings of the 12th International Conference on World Wide Web, 2003, pp. 710–717.
[73] A. Gladun, J. Rogushina, F. Garcı, R. Martínez-Béjar, J.T. Fernández-Breis, et al., An application of intelligent techniques and semantic web technologies in e-learning environments, Expert Syst. Appl. 36 (2) (2009) 1922–1931.
[74] E. Motta, J. Domingue, L. Cabral, M. Gaspari, Irs–II: A framework and infrastructure for semantic web services, in: International Semantic Web Conference, Springer, 2003, pp. 306–318.
[75] S.A. McIlraith, T.C. Son, H. Zeng, Semantic web services, IEEE Intell. Syst. 16 (2) (2001) 46–53.
[76] D. Schraudner, V. Charpenay, An HTTP/RDF-based agent infrastructure for manufacturing using stigmergy, in: European Semantic Web Conference, Springer, 2020, pp. 197–202.
[77] M.O. Shafiq, Y. Ding, D. Fensel, Bridging multi agent systems and web services: towards interoperability between software agents and semantic web services, in: 2006 10th IEEE International Enterprise Distributed Object Computing Conference (EDOC'06), IEEE, 2006, pp. 85–96.
[78] J. Bao, G. Slutzki, V. Honavar, Privacy-preserving reasoning on the semanticweb, in: IEEE/WIC/ACM International Conference on Web Intelligence (WI'07), IEEE, 2007, pp. 791–797.
[79] P. Casanovas, Semantic web regulatory models: Why ethics matter, Phil. Technol. 28 (1) (2015) 33–55.
[80] F.L. Gandon, N.M. Sadeh, Semantic web technologies to reconcile privacy and context awareness, J. Web Semant. 1 (3) (2004) 241–260.
[81] D. Jutla, L. Xu, Privacy agents and ontology for the semantic web, in: AMCIS 2004 Proceedings, 2004, p. 210.
[82] L. Kagal, T. Finin, M. Paolucci, N. Srinivasan, K. Sycara, G. Denker, Authorization and privacy for semantic web services, IEEE Intell. Syst. 19 (4) (2004) 50–56.
[83] S. Kirrane, S. Decker, Intelligent agents: The vision revisited, in: Proceedings of the 2nd Workshop on Decentralizing the Semantic Web Co-Located with the 17th International Semantic Web Conference, 2018.
[84] K. Kravari, N. Bassiliades, G. Governatori, A policy-based B2C e-contract management workflow methodology using semantic web agents, Artif. Intell. Law 24 (2) (2016) 93–131.

[85] E. Oren, R. Delbru, M. Catasta, R. Cyganiak, H. Stenzhorn, G. Tummarello, Sindice. com: a document-oriented lookup index for open linked data, Int. J. Metadata, Semant. Ontol. 3 (1) (2008) 37–52.
[86] M. Palmirani, M. Martoni, A. Rossi, C. Bartolini, L. Robaldo, PrOnto: Privacy ontology for legal reasoning, in: International Conference on Electronic Government and the Information Systems Perspective, Springer, 2018.
[87] F.-P. Pai, I.-C. Hsu, Y.-C. Chung, Semantic web technology for agent interoperability: a proposed infrastructure, Appl. Intell. 44 (1) (2016) 1–16.
[88] H.H. Wang, N. Gibbins, T. Payne, A. Patelli, Y. Wang, A survey of semantic web services formalisms, Concurr. Comput.: Pract. Exper. 27 (15) (2015) 4053–4072.
[89] N.B. Seghir, O. Kazar, K. Rezeg, A decentralized framework for semantic web services discovery using mobile agent, Int. J. Inf. Technol. Web Eng. (IJITWE) 10 (4) (2015) 20–43.
[90] K. Venkatachalam, N. Karthikeyan, S. Kannimuthu, Comprehensive survey on semantic web service discovery and composition, Adv. Nat. Appl. Sci. AENSI Publ. 10 (5) (2016) 32–40.
[91] N.B. Seghir, O. Kazar, K. Rezeg, S. Bourekkache, A semantic web services discovery approach based on a mobile agent using metadata, Int. J. Intell. Comput. Cybern. (2017).
[92] N.B. Seghir, O. Kazar, A new framework for web service discovery in distributed environments, in: 2017 First International Conference on Embedded & Distributed Systems (EDiS), IEEE, 2017, pp. 1–6.
[93] J.J.S. Zapater, D.M.L. Escrivá, F.R.S. García, J.J.M. Durá, Semantic web service discovery system for road traffic information services, Expert Syst. Appl. 42 (8) (2015) 3833–3842.
[94] M.S. Hajji, A. Al Maghrabi, Semantic web services methodology and tool extensions, Int. J. Appl. Eng. Res. 12 (2) (2017) 256–262.
[95] R. Verborgh, E. Mannnens, R. Van de Walle, Bottom-up web APIs with self-descriptive responses, in: Proceedings of the First Karlsruhe Service Summit Workshop-Advances in Service Research, 2015, p. 143.
[96] J.-P. Calbimonte, D. Calvaresi, M. Schumacher, Multi-agent interactions on the web through linked data notifications, in: Multi-Agent Systems and Agreement Technologies, Springer, 2017, pp. 44–53.
[97] D. Booth, H. Haas, F. McCabe, E. Newcomer, M. Champion, C. Ferris, D. Orchard, Web Services Architecture, Technical Report, W3C Working Group, 2004, URL https://www.w3.org/TR/ws-arch/#wsdisc.
[98] S. Poslad, Specifying protocols for multi-agent systems interaction, ACM Trans. Auton. Adapt. Syst. (TAAS) 2 (4) (2007) 15–es.
[99] J.R.V. Dantas, H.A. Lira, B. de Azevedo Muniz, T.M. Nunes, P.P.M. Farias, Semantic web services discovery adopting SERIN, in: 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing, IEEE, 2015, pp. 387–394.
[100] H. Boley, A. Paschke, O. Shafiq, RuleML 1.0: the overarching specification of web rules, in: International Workshop on Rules and Rule Markup Languages for the Semantic Web, Springer, 2010, pp. 162–178.
[101] K. Lister, L. Sterling, et al., Reconciling ontological differences for intelligent agents, Meaning Negotiation, AAAI Press (Menlo Park, America, 2002) (2002).
[102] E. Acar, M. Fink, C. Meilicke, H. Stuckenschmidt, uDecide: A protégé plugin for multiattribute decision making, in: Proceedings of the 8th International Conference on Knowledge Capture, 2015, pp. 1–4.
[103] Z. Ming, G. Wang, Y. Yan, J. Dal Santo, J.K. Allen, F. Mistree, An ontology for reusable and executable decision templates, J. Comput. Inf. Sci. Eng. 17 (3) (2017).
[104] R. Mizoguchi, M. Ikeda, K. Seta, J. Vanwelkenhuysen, Ontology for modeling the world from problem solving perspectives, in: Proc. of IJCAI-95 Workshop on Basic Ontological Issues in Knowledge Sharing, 1995, pp. 1–12.
[105] A.S. Rao, M.P. Georgeff, et al., Bdi agents: From theory to practice, in: Icmas, vol. 95, 1995, pp. 312–319.
[106] R.H. Bordini, J.F. Hübner, BDI agent programming in AgentSpeak using Jason, in: International Workshop on Computational Logic in Multi-Agent Systems, Springer, 2005.
[107] O. Morgenstern, J. Von Neumann, Theory of Games and Economic Behavior, Princeton University Press, 1953.
[108] S.M. Brown, E. Santos, S.B. Banks, Utility theory-based user models for intelligent interface agents, in: Conference of the Canadian Society for Computational Studies of Intelligence, Springer, 1998, pp. 378–392.
[109] W.Y. Wong, Learning Lightweight Ontologies from Text Across Different Domains using the Web As Background Knowledge, University of Western Australia, 2009.
[110] E. Puerto, J. Aguilar, et al., Automatic learning of ontologies for the Semantic Web: Experiment lexical learning, Respuestas 17 (2) (2012) 5–12.
[111] S.V. Albrecht, P. Stone, Autonomous agents modelling other agents: A comprehensive survey and open problems, Artificial Intelligence 258 (2018) 66–95.
[112] J. Young, V. Basile, L. Kunze, E. Cabrio, N. Hawes, Towards lifelong object learning by integrating situated robot perception and semantic web mining, in: 22nd European Conference on Artificial Intelligence, ECAI 2016, 285, IOS Press, 2016, pp. 1458–1466.
[113] M.N. Asim, M. Wasim, M.U.G. Khan, W. Mahmood, H.M. Abbasi, A survey of ontology learning techniques and applications, Database 2018 (2018).
[114] C.-V. Pal, F. Leon, M. Paprzycki, M. Ganzha, A review of platforms for the development of agent systems, 2020, arXiv preprint arXiv:2007.08961.
[115] A. Uszok, J. Bradshaw, R. Jeffers, N. Suri, P. Hayes, M. Breedy, L. Bunch, M. Johnson, S. Kulkarni, J. Lott, KAoS policy and domain services: Toward a description-logic approach to policy representation, deconfliction, and enforcement, in: Proceedings of the 4th IEEE International Workshop on Policies for Distributed Systems and Networks, IEEE Computer Society, 2003, p. 93.
[116] A. Uszok, J.M. Bradshaw, M. Johnson, R. Jeffers, A. Tate, J. Dalton, S. Aitken, KAoS policy management for semantic web services, IEEE Intell. Syst. 19 (4) (2004) 32–41.
[117] L. Kagal, et al., Rei: A Policy Language for the Me-Centric Project, Technical Report, HP Laboratories Palo Alto, 2002.
[118] N. Damianou, N. Dulay, E.C. Lupu, M. Sloman, Ponder: A Language for Specifying Security and Management Policies for Distributed Systems, Technical Report, Imperial College, Department of Computing, 2000.
[119] G. Muller, A Reference Architecture Primer, Eindhoven Univ. of Techn., Eindhoven, White Paper, 2008.
[120] L. Panait, S. Luke, Cooperative multi-agent learning: The state of the art, Auton. Agents Multi-Agent Syst. 11 (3) (2005) 387–434.
[121] Y. Shoham, R. Powers, T. Grenager, Multi-Agent Reinforcement Learning: A Critical Survey, Technical Report, Stanford University, 2003.
[122] D. Bloembergen, K. Tuyls, D. Hennes, M. Kaisers, Evolutionary dynamics of multi-agent learning: A survey, J. Artificial Intelligence Res. 53 (2015) 659–697.
[123] D. Zowghi, C. Coulin, Requirements elicitation: A survey of techniques, approaches, and tools, in: Engineering and Managing Software Requirements, Springer, 2005, pp. 19–46.
[124] I. Sommerville, P. Sawyer, S. Viller, Viewpoints for requirements elicitation: a practical approach, in: Proceedings of IEEE International Symposium on Requirements Engineering: RE'98, IEEE, 1998, pp. 74–81.
[125] J.A. Goguen, C. Linde, Techniques for requirements elicitation, in: [1993] Proceedings of the IEEE International Symposium on Requirements Engineering, IEEE, 1993, pp. 152–164.
[126] A. Davis, O. Dieste, A. Hickey, N. Juristo, A.M. Moreno, Effectiveness of requirements elicitation techniques: Empirical results derived from a systematic review, in: 14th IEEE International Requirements Engineering Conference (RE'06), IEEE, 2006, pp. 179–188.
[127] P. Reddivari, T. Finin, A. Joshi, Policy-based access control for an RDF store, in: Proceedings of the Policy Management for the Web Workshop, 2005, pp. 78–83.
[128] A. Jain, C. Farkas, Secure resource description framework: An access control model, in: Proceedings of the Eleventh ACM Symposium on Access Control Models and Technologies, ACM, 2006, pp. 121–129.
[129] F. Abel, J. De Coi, N. Henze, A. Koesling, D. Krause, D. Olmedilla, Enabling advanced and context-dependent access control in RDF stores, in: The Semantic Web, vol. 4825, Springer Berlin Heidelberg, 2007, pp. 1–14.
[130] G. Flouris, I. Fundulaki, M. Michou, G. Antoniou, Controlling access to RDF graphs, in: Proceedings of the Third Future Internet Conference on Future Internet, Springer-Verlag, 2010, pp. 107–117.
[131] S. Dietzold, S. Auer, Access control on RDF triple stores from a semantic wiki perspective, in: Proceedings of the ESWC'06 Workshop on Scripting for the Semantic Web, 2006.
[132] A. Gabillon, L. Letouzey, A View Based Access Control Model for SPARQL, in: Network and System Security (NSS), 2010 4th International Conference on, pp. 105–112, 2010.
[133] S. Kirrane, A. Abdelrahman, A. Mileo, S. Decker, Secure manipulation of linked data, in: The Semantic Web - ISWC 2013, vol. 8218, Springer Berlin Heidelberg, 2013.
[134] L. Kagal, T. Finin, A policy language for a pervasive computing environment, in: Proceedings POLICY 2003. IEEE 4th International Workshop on Policies for Distributed Systems and Networks, IEEE Comput. Soc, 2003, pp. 63–74.
[135] P.A. Bonatti, D. Olmedilla, Rule-based policy representation and reasoning for the semantic web, in: Proceedings of the Third International Summer School Conference on Reasoning Web, Springer-Verlag, 2007, pp. 240–268.
[136] M. Giereth, On partial encryption of RDF-graphs, in: Proc. of International Semantic Web Conference, vol. 3729, pp. 308–322.
[137] S. Gerbracht, Possibilities to encrypt an RDF-graph, in: Proc. of Information and Communication Technologies: From Theory to Applications, 2008, pp. 1–6.
[138] A. Kasten, A. Scherp, F. Armknecht, M. Krause, Towards search on encrypted graph data, in: Proc. of the International Conference on Society, Privacy and the Semantic Web-Policy and Technology, 2013, pp. 46–57.

[139] J.D. Fernández, S. Kirrane, A. Polleres, S. Steyskal, Self-enforcing access control for encrypted RDF, in: European Semantic Web Conference, Springer, 2017, pp. 607–622.
[140] A. Kasten, A. Scherp, P. Schauß, A framework for iterative signing of graph data on the web, in: The Semantic Web: Trends and Challenges: 11th International Conference, ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014. Proceedings, Springer International Publishing, 2014, pp. 146–160, http://dx.doi.org/10.1007/978-3-319-07443-6_11.
[141] F. Radulovic, R. García Castro, A. Gómez-Pérez, Towards the anonymisation of RDF data, in: SEKE, KSI Research, 2015, http://dx.doi.org/10.18293/SEKE2015-167.
[142] B. Heitmann, F. Hermsen, S. Decker, k-RDF-neighbourhood anonymity: Combining structural and attribute-based anonymisation for linked data, in: Proceedings of the 5th Workshop on Society, Privacy and the Semantic Web - Policy and Technology (PrivOn2017) (PrivOn), 2017.
[143] Z. Lin, From Isomorphism-Based Security for Graphs to Semantics-Preserving Security for the Resource Description Framework (rdf) (Master's thesis), University of Waterloo, 2016.
[144] R.R.C. Silva, B.C. Leal, F.T. Brito, V.M. Vidal, J.C. Machado, A differentially private approach for querying RDF data of social networks, in: Proceedings of the 21st International Database Engineering & Applications Symposium, ACM, ISBN: 978-1-4503-5220-8, 2017, pp. 74–81.
[145] P.A. Bonatti, S. Kirrane, Big data and analytics in the age of the GDPR, in: 2019 IEEE International Congress on Big Data (BigDataCongress), IEEE, 2019, pp. 7–16.
[146] O. Sacco, A. Passant, A privacy preference ontology (PPO) for linked data, in: Linked Data on the Web, CEUR-WS, 2011.
[147] L. Ding, L. Zhou, T.W. Finin, Trust based knowledge outsourcing for semantic web agents, in: Web Intelligence, 2003, pp. 379–387.
[148] L. Ding, P. Kolari, T. Finin, A. Joshi, Y. Peng, Y. Yesha, On homeland security and the semantic web: A provenance and trust aware inference framework, in: AAAI Spring Symposium: AI Technologies for Homeland Security, 2005.
[149] C. Laufer, D. Schwabe, On modeling political systems to support the trust process, in: Proceedings of the 5th Workshop on Society, Privacy and the Semantic Web - Policy and Technology (PrivOn2017) (PrivOn), 2017.
[150] C. Bizer, R. Oldakowski, Using context-and content-based trust policies on the semantic web, in: Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers & Posters, ACM, 2004, pp. 228–229.
[151] T.F. Gordon, G. Governatori, A. Rotolo, Rules and norms: Requirements for rule interchange languages in the legal domain, in: International Workshop on Rules and Rule Markup Languages for the Semantic Web, Springer, 2009, pp. 282–296.
[152] M. Palmirani, G. Governatori, A. Rotolo, S. Tabet, H. Boley, A. Paschke, LegalRuleML: XML-based rules and norms, in: International Workshop on Rules and Rule Markup Languages for the Semantic Web, Springer, 2011.
[153] C. Dowling, Intelligent agents: some ethical issues and dilemmas, in: Proc. AIC 2000, 2000, pp. 28–32.
[154] V. Dignum, Ethics in Artificial Intelligence: Introduction to the Special Issue, Springer, 2018.
[155] M. Wooldridge, Intelligent agents: The key concepts, in: ECCAI Advanced Course on Artificial Intelligence, Springer, 2001, pp. 3–43.
[156] High-Level Expert Group on Artificial Intelligence (AI HLEG), https://digital-strategy.ec.europa.eu/en/library/ethics-guidelines-trustworthy-ai,
[157] B. Van Der Vecht, A.P. Meyer, M. Neef, F. Dignum, J.-J.C. Meyer, Influence-based autonomy levels in agent decision-making, in: International Workshop on Coordination, Organizations, Institutions, and Norms in Agent Systems, Springer, 2006, pp. 322–337.
[158] Y. Guo, Z. Pan, J. Heflin, LUBM: A benchmark for OWL knowledge base systems, J. Web Semant. 3 (2–3) (2005) 158–182.
[159] C. Bizer, A. Schultz, The berlin sparql benchmark, Int. J. Semant. Web Inf. Syst. (IJSWIS) 5 (2) (2009) 1–24.
[160] R. Angles, J.B. Antal, A. Averbuch, P. Boncz, O. Erling, A. Gubichev, V. Haprian, M. Kaufmann, J.L.L. Pey, N. Martínez, et al., The LDBC social network benchmark, 2020, arXiv preprint arXiv:2001.02299.
[161] R. Usbeck, M. Röder, A.-C. Ngonga Ngomo, C. Baron, A. Both, M. Brümmer, D. Ceccarelli, M. Cornolti, D. Cherix, B. Eickmann, et al., GERBIL: general entity annotator benchmarking framework, in: Proceedings of the 24th International Conference on World Wide Web, 2015, pp. 1133–1143.
[162] A.-C.N. Ngomo, M. Röder, HOBBIT: holistic benchmarking for big linked data, ERCIM News 2016 (105) (2016).
[163] J. Sculley, J.A. Byrne, Odyssey, Harper & Row Publishers, Inc., 1987.
[164] M.B. Hoy, Alexa, siri, cortana, and more: an introduction to voice assistants, Med. Ref. Serv. Q. 37 (1) (2018) 81–88.
[165] W. Shi, J. Cao, Q. Zhang, Y. Li, L. Xu, Edge computing: Vision and challenges, IEEE Internet Things J. 3 (5) (2016) 637–646.
[166] A. Braud, G. Fromentoux, B. Radier, O. Le Grand, The road to European digital sovereignty with Gaia-X and IDSA, IEEE Netw. 35 (2) (2021) 4–5.