Computers in Industry 151 (2023) 103979

Contents lists available at ScienceDirect

Computers in Industry

journal homepage: www.sciencedirect.com/journal/computers-in-industry

A novel agile ontology engineering methodology for supporting organizations in collaborative ontology development

Daniele Spoladore [a,b,*], Elena Pessot [a,c], Alberto Trombetta [b]

[a] National Research Council of Italy, Institute of Intelligent Industrial Technologies and Systems for Advanced Manufacturing, Via Alfonso Corti 12, Milan 20133, Italy
[b] Department of Pure and Applied Sciences, Insubria University, Via Ottorino Rossi 9, Varese 21100, Italy
[c] University of Siena, Department of Information Engineering and Mathematics, via Roma, 56, Siena 53100, Italy

ARTICLE INFO

Keywords:
Ontology Engineering
Ontology Authoring
Agile Ontology Development Methodology
Knowledge Engineering
Industry 4.0

ABSTRACT

Ontologies can represent technological enablers for knowledge elicitation and management in different kinds of organizations, especially with the exponential growth of sources and types of data fostered by digital transformation. However, their adoption in business applications is still limited, with existing Ontology Engineering Methodologies (OEMs) lacking adequate support during knowledge elicitation, authoring and reuse phases. This paper introduces a novel agile ontology engineering methodology (AgiSCOnt) to support ontologists (especially novice ones) in ontology development workflow, fostering collaboration with domain experts in an iterative, flexible and customizable approach. AgiSCOnt combines macro-level instructions with micro-level guidance, leveraging existing techniques and a management framework to help novice ontologists throughout the whole ontology engineering process. The methodology is compared to existing OEMs and assessed with three other agile methodologies (UPONLite, SAMOD, and RapidOWL). The evaluation is conducted with a sample of novice ontologists in a learning environment on Industry 4.0 technologies. Both the development process with a methodology from a user perspective and the quality of the developed ontologies were considered in the evaluation. Preliminary results show that AgiSCOnt effectively supports authoring and reuse, with developed ontologies of good quality. It is perceived as clear and simple, while being flexible and adaptable enough, thus supporting knowledge management and sharing in industrial organizations through the documentation of the ontologies.

1. Introduction

Over the last decade, several organizations' knowledge management and decision-making processes have been challenged by the spread of digital technologies. The adoption of different solutions pertaining to the Industry 4.0 (I4.0) paradigm, such as the Internet of Things (IoT), cloud computing, and big data analytics, resulted in an exponential growth of sources and types of data in terms of volume, variety, velocity, scope, and accessibility (O'Leary, 2017). These large amounts of data generated should be thus captured and transformed into useful information to be adequately integrated and exploited by organizations – in particular, by companies (Jaskó et al., 2020). In this sense, the possibility to structure, manage and reuse this data became a critical success factor for different kinds of organizations (SMEs, large enterprises, academia, etc.), combining human factors and technology-based methods for effective digital transformations (Evans and Price, 2020; Meski et al., 2021). To this aim, there is the need to rely on interoperable data models that ensure the sharing of heterogenous knowledge among different internal resources (Kumar et al., 2019) and enhance integration and collaboration with an unambiguous and normative representation of contextual knowledge (Fatfouta and Le-Cardinal, 2021). The ability of industrial organizations to manage and share their organizational knowledge is also argued to enhance the organization's overall performance (Osman et al., 2022).

Among I4.0 enabling technologies, the Semantic Web and, in particular, ontologies are promising solutions for data representation and complex knowledge modeling (Kumar et al., 2019; Jaskó et al., 2020). An ontology is a formal and explicit specification of a shared conceptualization of a knowledge domain (Gruber, 1993; Guarino et al., 2009) that can effectively support a common understanding of concepts

* Correspondence to: Institute of Intelligent Industrial Technologies and Systems for Advanced Manufacturing (STIIMA) – National Research Council of Italy (CNR) Via Alfonso Corti 12, Milan 20133, Italy.
E-mail address: daniele.spoladore@stiima.cnr.it (D. Spoladore).

https://doi.org/10.1016/j.compind.2023.103979
Received 13 January 2023; Received in revised form 9 June 2023; Accepted 12 June 2023
Available online 17 June 2023

and relationships composing that domain. As computable artifacts, ontologies represent knowledge relying on expressive ontological languages. Through automated reasoning processes, they can infer entailed knowledge (Baader et al., 2008) and can fulfill the inter-machine and inter-human communication (Feilmayr and Wöß, 2016).

Domain ontologies can also serve as basis for knowledge management systems, which, however, are still very limited in real-world adoption (Mora et al., 2022), hindering their widespread adoption in large-scale business applications (Feilmayr and Wöß, 2016). Many industries are not properly exploiting the capabilities provided by advanced knowledge management system technology (O'Leary, 2017). The lack of standardized methods to capture domain knowledge, conceptualize it, and modeling with formal ontological languages are the "traditional" limitations characterizing Ontology Engineering (OE) (Simperl and Tempich, 2006; Keet, 2018): these results in high costs, limited reliability of the engineering process, long development periods.

To tackle some of these limitations, since the early 2000s, researchers have proposed many Ontology Engineering Methodologies (OEMs) to support the development of ontologies and overcome these limitations while considering the specific application requirements. Specifically, OE is the discipline guiding ontology development with cross-discipline expertize and practical guidance to deliver usable semantic models (Mizoguchi and Ikeda, 1998; Rebstock et al., 2008). The methodologies provide a set of guidelines to help ontologists to move from informal knowledge (e.g., documents, data, practices, know-how) to its formal representation (i.e., based on Description Logic). As noted by researchers adopting ontologies in industrial contexts (Sanya and Shehab, 2015), the successful development of an ontology and its subsequent adoption often depend on the methodology selected. Thus, OEMs are considered not interchangeable – i.e., OEMs focus on different aspects of the OE process, thus the adoption of different OEMs to model the same domain may result in different ontologies. Among the features characterizing OEMs, the workflow and the cooperative aspects can play a pivotal role (Kotis et al., 2020). In the last decade, agile methodologies in systems and software engineering domains have also permeated the research on ontology-based knowledge management systems (Mora et al., 2022). Agile OEMs underline the central role of collaborative development, involving domain experts and stakeholders within the overall OE process.

They have the potential to overcome the problems of traditional OE methods, including the duration of the development periods, high costs, low reliability and weak integration ability (Jaskó et al., 2020), thus seeming valuable in overcoming the issues of cost-effectiveness and profitability for OE (Simperl and Tempich, 2009). However, they could considerably differ in the quality of the ontologies developed, because of the focus on the fast adaptation of ontology prototype, minimizing the development efforts (Kotis et al., 2020; Peroni, 2016)) and in the ontologists experience about the entailed OE process (Spoladore and Pessot, 2022). In addition, agile OEMs can foster innovation through collaboration and support digital transformation, but their adoption requires users to have some pre-existing skills in OE. The process of retrieving and structuring knowledge in ontology-based knowledge management tools is usually complex for novice users (Osman et al., 2022), thus making agile OEMs less suitable for supporting them in the early phases of adoption in industrial organizations. Moreover, none of the agile methodologies investigated in literature proposes techniques to overcome the "knowledge elicitation bottleneck" cited by Gavrilova (2007) – a well-known issue of OE and knowledge engineering in general (Mizoguchi, 2019) –, nor they provide instructions for authoring, which includes tasks such as exploring ontologies, comparing versions, debugging, and testing, thus limiting their adoption by novice ontology engineers (Vigo et al., 2014a). In addition, there is an increasing expectation in the literature and from practitioners of different kind of organizations, including the educational context, for agile versions of comprehensive and systematic development methodologies that properly integrate project management and technical systems engineering processes (Burchardt and Maisch, 2019; Mora et al., 2022).

This paper aims to contribute to the literature on OE and ontology-based knowledge management systems by presenting a novel agile OEM named Agile, Simplified and Collaborative Ontology engineering methodology (AgiSCOnt). This methodology tackles some of the traditional issues of OE, trying to support ontologists in acquiring relevant domain knowledge and supporting them both at a macro and micro level. AgiSCOnt is then compared to three well-known agile OEMs through a qualitative experiment involving a sample of novice ontologists from Italian companies.

The remainder of this paper is organized as follows: Section 2 introduces the relevant theoretical background, focusing on agile methodologies. Section 3 presents AgiSCOnt and its features. Section 4 describes the empirical research conducted to evaluate AgiSCOnt and compare it with other agile methodologies, while Section 5 discusses the results, contributions and remarks on some limitations of the study. Finally, the Conclusions summarize the main outcomes of this work.

2. Ontology engineering methodologies: Advantages and limitations of the agile approach

2.1. The status of ontology engineering and its methodologies

Following the growing attention that Semantic Web and ontologies gathered during the Nineties and in the first decade of the 2000s, several OEMs were introduced to support ontology engineers in OE activities.

OEMs can generally be classified into two types: macro-level OEMs specify the activities to transform an informal domain representation into an ontology, while micro-level OEMs (or ontology authoring methodologies) deal with supporting ontology engineers in selecting the best constructs to model domain knowledge. Traditionally, these two types of support are scarcely compatible (Vigo et al., 2014b), and researchers focused most of their attention on the development of macro-level OEMs. Almost all macro-level OEMs are more focused on providing a set of instructions to be followed to get to an ontological representation of a domain, and structure their activities in three stages (Simperl and Tempich, 2006; Vigo et al., 2014b): 1) the ontology management, which is dedicated to conducting feasibility studies or cost-benefits analysis; 2) the development, which identifies the domain (s) involved and acquires the knowledge (domain analysis) to be conceptualized (conceptualization) and to develop into an ontology using an ontological language (implementation); 3) the use of the developed ontology, which also includes maintenance activities. The guidance provided by micro-level OEMs is essential – in particular for novice ontologists – to understand *how* to formalize the domain at hand with ontological languages. Authoring methods can support ontology developers in modeling common problems, guiding them to move from an informal representation to a logic-based one (Keet, 2018). Nonetheless, micro-level OEMs are a part of macro-level OEMs, as ontologists are required to face authoring issues at some point in the OE process, since they have to perform ontology design choices and write axioms to formalize knowledge (Keet, 2018; Davies et al., 2019).

More recently, Keet (2018) proposed a three-sided classification of macro-level development OEMs, adopting the general approach as a taxonomical criterion. Firstly, *Waterfall OEMs* foresee an ordered sequence of steps that must be followed to achieve the development of the ontology. A set of preliminary activities has often to be conducted before development stage (a feasibility study and/or the definition of a management framework). They illustrate why the ontology is necessary to solve the problem at hand and identify the main actors (ontologists and domain experts) involved in the development process. Waterfall OEMs were the first methodologies to be discussed, and the collaborative dimension is not particularly stressed, as domain experts may have a passive role in the engineering process (Garcia et al., 2010), with ontology engineers leading the whole process and controlling the interaction with domain experts. Secondly, the *Lifecycle approach*

Table 1
Comparison of AgiSCOnt with other methodologies.

OEM	Activity workflow	Collaborative engineering	Ontology reuse	Ontology evaluation/ validation	Ontology maintenance through community	Domain experts involvement	Iterative process	Detailed versioning/ documentation	Authoring support	Project management processes	**Real-world cases testing**
Methontology	Detailed	No	No specific activities foreseen	Evaluation of the technical aspects of the ontology (correctness)	Entailed by the "evolving prototype lifecycle" workflow	During the knowledge acquisition phase	Entailed by the "evolving prototype lifecycle" workflow	CQs, glossary; final documentation at the delivery of the ontology	No specific activities foreseen	No specific activities foreseen	Chemistry (Fernández-López al, 1999); Law (Corcho et al., 2005); Disability (Mohemad et al., 2019)
Diligent	Detailed	Yes; collaboration on in decentralized settings	No specific activities foreseen	No specific activities foreseen	Users provide feedback for the "revision" phase	During the "ontology building" phase	No	No specific activities foreseen	No specific activities foreseen	No specific activities foreseen	Tourism; FAQ system (Pinto et al., 2009)
OTK	Detailed	Yes; not detailed	Top-level ontologies reused in "Refinement" step	"Evaluation" step to validate technical and formal aspects of the ontology and user satisfaction	No specific activities foreseen	In "Kickoff" step	Limited to "Refinement", "Evaluation", and "Application & evolution"	ORSD; graph (similar to conceptual map)	No specific activities foreseen	No specific activities foreseen	Business strategy (Staab et al., 2001)
HCOME	Detailed	Continuous throughout all phases and supported by HCONE	Collaboratively discussed in the "conceptualization" phase	Supported during the "Exploitation" phase by HCONE	Supported during the "Exploitation" phase by HCONE	During the "conceptualization" phase	Entailed by the collaborative environment	Mentioned but not specified	No specific activities foreseen	No specific activities foreseen	-
DOGMA	Detailed	Yes; distributed and collaborative development environment	No specific activities foreseen	No specific activities foreseen	Domain axiomatization foreseen in DOGMA can support maintenance	Involved in axiomatization phases	No	CQs	No specific activities foreseen	No specific activities foreseen	-
RapidOWL	Not foreseen	Yes; most of its practices are collaborative	No specific activities foreseen	No specific activities foreseen	Partially foreseen in "Information integration" dedicated to maintenance	In many practices	Entailed by the agile workflow	No specific activities foreseen	No specific activities foreseen	No specific activities foreseen	-
NeOn	Detailed	Yes; not detailed	Scenarios for the reuse of ontological and non-ontological resources	Technical	No specific activities foreseen	In Scenario 1 (knowledge acquisition)	Yes (if lifecycle)	CQs; ORSD	Scenario 7 (reuse of ODPs)	No specific activities foreseen	Ambient Intelligence (Spoladore et al., 2022); Health (Mariño et al., 2018)
LOT	Detailed	Yes; all phases are collaborative	In "Ontology requirement specification", "Ontology implementation", "Ontology maintenance" phases	Validation and verification	Foreseen in "Ontology maintenance"	In "Ontology requirements specification" and "Ontology implementation" phases	Entailed by the agile workflow	CQs; ORSD; HTML documentation, graphical representations, and examples ("Ontology publication" phase)	No specific activities foreseen	No specific activities foreseen	SAREF, social networks, building information modeling (Poveda-Villalón et al., 2022)

(*continued on next page*)

Table 1 (*continued*)

OEM	Activity workflow	Collaborative engineering	Ontology reuse	Ontology evaluation/ validation	Ontology maintenance through community	Domain experts involvement	Iterative process	Detailed versioning/ documentation	Authoring support	Project management processes	**Real-world cases testing**
UponLite	Detailed	In steps from 1 to 5	Reuse of lexical databases	Technical, logical and "social" evaluation	No specific activities foreseen	In steps from 1 to 5	Entailed by the agile workflow	glossary; taxonomy	No specific activities foreseen	No specific activities foreseen	Public organization (De Lille and Roelens, 2021)
SAMOD	Detailed	Possible (two teams of ontologists)	In the "Refactoring" step	Testing the ontology with test cases and CQs	No specific activities foreseen	Limited to knowledge acquisition phase	Yes	CQs; glossary; set of test cases	No specific activities foreseen	No specific activities foreseen	-
AgiSCOnt	Detailed	Yes; all Steps are collaborative	Foreseen in Steps 1 and 2; envisaged in Step 3	Logical validation in Step 2; evaluation by stakeholders in Step 3	Foreseen in Step 3	Actively involved in all Steps	Yes	CQs, glossary, ORSD, Conceptual map (graphical representation), use cases	Mapping conceptual map to ODPs	Yes (ISO/ IEC/IEEE 16326:2019)	Health (Spoladore et al., 2023)

conceives an ontology as an evolving product passing through a life-cycle, usually coupled with an iterative engineering process where the ontology is expected to go through each phase more than once during its life, but not necessarily in an orderly fashion (Neuhaus et al., 2014). Some of the phases may occur in sequence (with some phases dependent from others) or in parallel. The ontology is seen as part of a complex system in which human operators, different processes and technologies take part and are guided by a list of questions (concerning the expected outputs of each phase). Thirdly, the *Agile approach* follows the trend in software engineering of rapid prototyping and development of models in a collaborative dimension. The list of steps envisaged by these methodologies is usually limited – if not completely absent – and steps are connected in an iterative cycle that evolves a prototypical ontology into the final model. Rather than presenting a sequence of activities, these OEMs present some fundamental intervention areas that can be reiterated until the final version of the ontology is reached. In this context, the dependencies among the various activities are partially explicit (a characteristic of waterfall and lifecycle OEMs), thus limiting the complexity of the methodology and fostering innovative and creative approaches. Agile OEMs focus on the role of domain experts and stakeholders as co-participants in the engineering process, reducing the role of ontology engineers to the final area (the one dedicated to formalization with ontological languages). These OEMs are often suggested in contexts where there is the need to involve a community of stakeholders in development activities, and to support people with limited expertize in participating to the ontology engineering process (Peroni, 2016; Spoladore and Pessot, 2021).

As pointed out by Kotis et al. (2020) and Tudorache (2020), OEMs development is shifting towards agile and customized methodologies to better answers specific ontologists' needs – in particular, cooperative development and flexibility.

2.2. Agile OEMs

Despite the exponential growth of the agile development approach in the software engineering domain, agile OEMs are still very limited, with few methodologies are described in the literature. Nevertheless, agile OEMs answer organizations and companies' need for "stepping up a gear" in digital transformation (Burchardt and Maisch, 2019), providing the means to facilitate the digitalization of knowledge management processes leveraging a large amount of data sources (Sandkuhl et al., 2019). The current agile OEMs focus on some specific aspects and adopt approaches that are very different in terms of steps and involved activities.

UPONLite (De Nicola and Missikoff, 2016) is an OEM organized in six steps. These are: 1. identifying the domain terminology, 2. defining a glossary, 3. generating a taxonomy of the concepts composing the glossary, 4. connecting the entities through relationships, 5. defining the parthood, and 6. developing the ontology. Only the first two steps are dependent, while the others can co-occur and be reiterated. In this methodology, stakeholders and domain experts are encouraged to work with steps 1–5 using familiar tools (like spreadsheets and conceptual maps) to produce a conceptualization and specification of the domain, leaving the ontology engineers to formalize them in step 6.

SAMOD (Peroni, 2016) adopts an evolving prototype approach: starting from a motivating scenario, ontologists collect the requirements from domain experts (then their role is concluded) and develop a modelet (a "micro-ontology" of the scenario). The modelet is tested to understand if it answers to all the requirements foreseen by the target ontology. If it does not, the scenario is modified by adding more complexity or another scenario to merge the modelet with the modelet produced for the second scenario. This process ends when the latest iteration of the modelet can answer all the competency questions, thus making this version of the modelet the ontology.

RapidOWL (Auer and Herre, 2007) is based on a set of values and practices (roughly corresponding to ten activities in OE) dedicated to

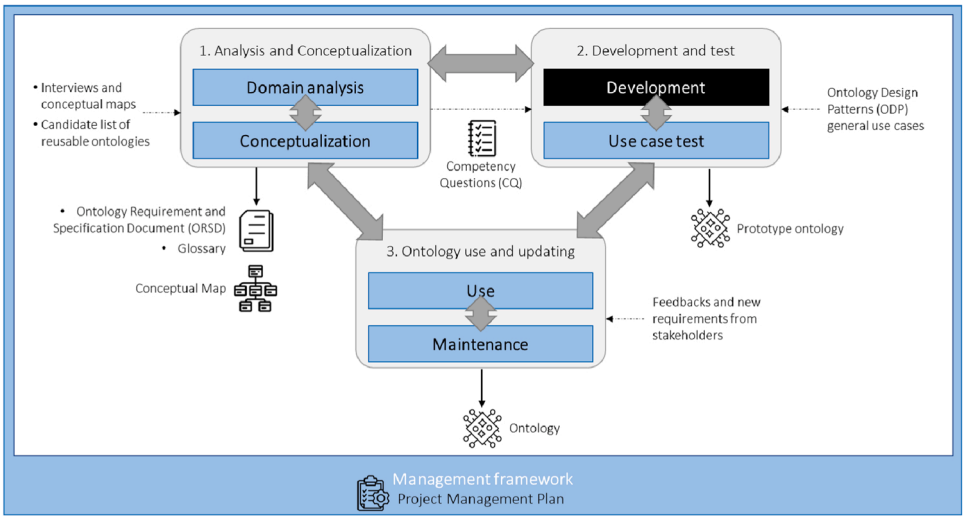

Fig. 1. Steps and activities composing the AgiSCOnt agile methodology (blue boxes indicate collaborative activities, while black backgrounds indicate activities that may be conducted only by ontologists, as described in Section 3.4).

structuring knowledge, focusing on cooperation among ontology engineers and domain experts. RapidOWL does not prescribe a sequence of modeling activities that should be followed precisely: it is up to the participants to decide which practices to adopt and when, and none of the activities is explicitly dependent on the others.

Despite the importance of self-organization and iterative process that characterizes the agile approach, available agile OEMs convey some of the issues of the macro-level methodologies. Firstly, the three methodologies described mention the possibility of reusing existing ontologies – a cornerstone of Semantic Web and OE. However, they do not provide any operative support in identifying the models to be reused, which is fundamental for developing and maintaining them especially when they concern the same knowledge domain (Fernández-López et al., 2019). Secondly, proper support in authoring, which includes tasks such as exploring ontologies, comparing versions, debugging, and testing, is missing, thus limiting their adoption by novice ontology engineers (Vigo et al., 2014a). Similarly to waterfall and most of the lifecycle methodologies, agile OEMs do not delve into how to model concepts and relationships, leaving ontologists the burden of identifying and adopting modeling solutions. Thirdly, there is a scarce debate in agile OEM on specific techniques to tackle knowledge acquisition, traditionally considered one of the bottlenecks of OE (Gavrilova, 2007), and mainly devoted to the developer after the further development and deployment of the shared knowledge within the organization (Gavrilova and Andreeva, 2012). Finally, the activities pertaining to ontology management (in particular, the management of the OE process) are often disregarded, while their implementation in a real-world setting should take into account both project management and engineering processes (Mora et al., 2022).

These features of existing agile OEMs, together with elements such as collaborative engineering, ontology maintenance through community, and iterative process, are compared in Table 1. Other well-known methodologies in literature are examined – i.e., Methontology (Fernández-López et al., 1997), DILIGENT (Pinto et al., 2004), On-To-Knoweldge (Sure et al., 2004), HCOME (Kotis et al., 2005; Kotis and Vouros, 2006), DOGMA (Jarrar and Meersman, 2008; Spyns et al., 2008), RapidOWL (Auer, 2006; Auer and Herre, 2007; Auer, 2010), NeOn (Gómez-Pérez and Suárez-Figueroa, 2009; Suárez-Figueroa et al., 2015), LOT (Poveda-Villalón, 2012; Poveda-Villalón et al., 2022), UponLite (De Nicola and Missikoff, 2016), and SAMOD (Peroni, 2016) – and the methodology proposed in this paper, i.e. AgiSCOnt. The features and elements considered in the comparison are from Kotis et al. (2020) and Poveda-Villalón et al. (2022).

Some of the issues presented above and emerging in the comparison include the long development periods, the efforts for knoweldge elicitation, the limited involvement of domain experts in the domain conceptualization phases, which are tackled in the AgiSCOnt methodology, presented in Section 3.

2.3. Evaluation of OEMs

Beyond the comparison of the main features, proper evaluation frameworks could detect the shortcomings and strengths of agile OEMs. Albeit the interest in OEMs gained momentum until the first decade of the 2000s, researchers introduced tests and models to evaluate the methodologies when they began to question whether having several OEMs could impact OE and correspond to the concrete community's necessities related to different ways of developing ontologies. Consequently, some researchers started to investigate the efficacy of different OEMs in various ways – although research in this field is still very limited.

For instance, Tempich et al. (2005) evaluated their DILIGENT methodology, with particular attention to its engineering process. The methodology is tested with a group of legal experts to evaluate the collaborative engineering of an ontology in the legal domain. The evaluation of the OEM was based mostly on qualitative consideration, such as the possibility to cooperate even in geographically distributed environments via a Wiki, and participation to discussions, and on testing the developed ontology against a set of CQs. A proposal for a weighted evaluation of OEMs is described in (Hakkarainen et al., 2005), which allows quantifying seven aspects of a methodology – Weltanschauung (i.e., the philosophy of the OEM), coverage in process, coverage in product, reuse of product and process, participation of stakeholders, representation of product and process, and maturity. The seven aspects roughly correspond to the main activities provided by macro-level methodologies. The paper does not put the evaluation framework to the test. Therefore, its contribution is somehow limited to a theory for OEM evaluation.

Chimienti et al. (2009) adapted Balanced Score Cards to provide a quantitative evaluation for Unified Process for ONtology (UPON) development (De Nicola et al., 2009). Four perspectives are evaluated: ontology engineers' satisfaction, the efficiency, and simplicity of the process proposed by the OEM being evaluated, the analysis of the competencies and skills necessary for the people involved to perform the activities implied by the OEM, the evaluation of the ontology developed from a user's perspective.

Table 2
List of main elements and clauses from the ISO/IEC/IEEE 16326:2019 standard that compose AgiSCOnt's management framework.

Clause number	Clause Name	Description	Commonalities with AgiSCOnt
1	**Project Overview**		
1.1	Project summary		
1.1.1	Purpose, scope, and objectives	This clause enables OE process' participants to clearly state the ontology's purpose(s), scope, and objectives.	Step 1 (Ontology Ontology Requirement Specification Document)
1.1.2	Assumptions and constraints	Assumptions on which the ontology is developed should be stated; if some technological requirements or constraints emerge, they should also be detailed in this clause.	Step 1 (Ontology Ontology Requirement Specification Document)
1.1.3	Project deliverables	Work products (conceptual maps, Ontology Requirement Specification Document, Competency Questions, etc.) are listed in this clause, and provisional delivery dates are specified.	Step 1 (Ontology Ontology Requirement Specification Document and Documentation for the target ontology)
1.1.4	Schedule and budget summary	The amount of budget foreseen for the OE process should be stated and scheduled throughout the whole duration of the process. See also Clause 5.	
1.2	Evolution of the plan	Considering the recursive nature of AgiSCOnt's activities, the management of the OE process should be flexible enough to foresee the possibility of updating the managerial plan.	
5	**Definitions**	This clause defines (and references documents, if necessary) all terms necessary for the understanding of the project.	This step is not the Glossary foreseen in AgiSCOnt's Step 1.
6	**Project context**		
6.1	Process model	The management framework should specifically refer to AgiSCOnt and its engineering methodology to identify major project work activities (Steps), their relationships, and the outputs foreseen.	
6.4	Methods, tools, and techniques	AgiSCOnt provides methods and techniques for all the Steps composing it. OE process' participants should state here whether or not they require different methods or tools to achieve their goals.	
6.6	Project Organization		
6.6.1	External interfaces	This clause identifies the boundaries between the project's external entities. In here, the identification of external participants as domain experts (and stakeholders such as observers) should be stated.	Essential clause to identify domain experts involved in Step 1.
6.6.2	Internal interfaces	This clause describes the internal resources that participate in the OE process with AgiSCOnt, including internal domain experts, stakeholders, and ontologists.	Essential clause to identify domain experts, stakeholders, and ontologists belonging to the organization and involved in the OE process.
7	**Project planning**		
7.2	Project initiation		
7.2.1	Estimation plan	In this clause, an estimation of the OE process cost, the amount of time required, and the resource requirements should be conducted.	
7.2.2	Staffing plan	The number of staff (and their skills) required for the OE process should be specified. This clause should also account for external interfaces (4.6.1) domain experts that are actively involved in the OE process.	
7.2.4	Project staff training plan	Internal and external staff may require time (thus costs) to acquire domain-specific knowledge. The methods and costs of staff training are detailed in this clause.	
7.3	Project work plans		
7.3.1	Work activities	This clause specifies the work activities to be performed in the OE process. These should be extracted by the activities foreseen in each of AgiSCOnt's Steps.	AgiSCOnt's Steps provides each a set of activities that can be placed in this clause.
7.3.2	Schedule allocation	In this clause, scheduling relationships between the works activities can be specified to depict the time-sequencing constraints and illustrate the opportunities for concurrent activities.	AgiSCOnt's Steps are thought to be recursive, therefore project managers can a) provide a preliminary time schedule for each step or b) consider the whole activities from Step 1 to the delivery of the prototype (Step 3) as a one-time slot to be allocated according to OE process needs.
7.3.3	Resource allocation	The resources identified in clause	

(*continued on next page*)

Regarding agile OEMs, three works from the same author (Gobin, 2014a; b; c) are dedicated to evaluating agile methodologies. These works are interested in assessing the agility of the OEMs investigated, but focus less on users' perspectives and output quality.

On the contrary, Spoladore and Pessot (2022) propose a framework for the evaluation of the quality of the ontologies produced with an agile OEM (the outcome) and the perception of the ontologists participating in the OE process (the process). This framework is then used to evaluate the main agile OEMs (UPONLite, SAMOD, and RapidOWL) in the context of digital transformation of organizations.

This framework is adopted to evaluate the novel agile ontology engineering methodology (AgiSCOnt) introduced in this paper. Indeed, the methodology considers the features of the OEMs presented in Sections 2.1 and 2.2, combining macro-level instructions with micro-level guidance, leveraging existing techniques and a management framework to help novice ontologists throughout the whole ontology engineering process.

3. AgiSCOnt – Agile, simplified and collaborative ontology engineering methodology

In this Section, a novel agile OEM – Agile, Simplified and Collaborative Ontology engineering methodology (AgiSCOnt) – is introduced. AgiSCOnt was developed to propose a lightweight structure of its activities, which can be reiterated, and to tackle the abovementioned issues of knowledge elicitation and authoring by relying on existing techniques and solutions. Contrary to some methodologies (such as SAMOD, which foresees a partial involvement of domain experts), AgiSCOnt adopts a collaboration framework that involves domain experts in almost all its activities.

As shown in Fig. 1, AgiSCOnt divides its activities into three main steps, with an underpinning Management framework. The three steps are inspired by Simperl's general structure for macro-level OEMs (Simperl and Tempich, 2006), and in each step, a set of activities is identified and described. AgiSCOnt foresees a close collaboration (to be conducted either in the presence or in decentralized settings) among ontology engineers and domain experts: the latter are involved in all the steps of the OE process, with the sole exclusion of the Development activities in Step 2. Domain experts with experiences or knowledge of OE and its languages may take part to this activity, according to the specific aims.

3.1. Key features of AgiSCOnt

The role of domain experts is explicitly requested in each step of this OEM, as the assumption underlying AgiSCOnt's structure is that a solid definition of the domains involved and of the stakeholders' requirements is essential to the success of the whole OE output (Yang et al., 2019). Also, AgiSCOnt recognizes that knowledge elicitation from the knowledgeable individuals and the consequent domain analysis activities – with the ontologist acting as intermediary between the domain experts and the transfer of their knowledge to overall model – are a bottleneck for the OE process (Gavrilova and Andreeva, 2012). For such reasons, this methodology commits to techniques that have the twofold goal of (1) extracting knowledge from domain experts and their knowledge sources and (2) providing (with the help of domain experts) an informal conceptual map of the domain to guide the following development activities. The development activities foreseen in Step 2 are concentrated on helping ontology engineers move from the informal conceptual map to a formal language. In this effort, the OEM underlines the role of Ontology Design Patterns ODPs (Gangemi and Presutti, 2009) – reusable "building blocks" for larger ontologies – as a form of reuse. The "target ontology" (i.e., the ontology being developed with this OEM) is tested against a set of use cases provided by domain experts, which aim to understand whether the model contains all the relevant pieces of information or it requires further modifications. A similar approach is at the foundation of Step 3 activities, which revolve around the use of ontology. Feedback from users and stakeholders is fundamental to identifying possible requirements' modifications and updates to the target ontology.

The order of the steps is not linear, as outputs from use case testing may highlight the necessity of adding or removing some concepts from the model, thus modifying the outputs of the previous Step 1. Similarly, users and stakeholders adopting the target ontology may provide technical feedback on the model, so asking to modify Step 2's outputs, or to update the knowledge underlying the target ontology, therefore intervening on Step 1's output.

The recursive characteristic of this agile OEM can thus enable both an evolving-prototype approach or a model enrichment by reiteration paradigm. Also, AgiSCOnt suggests adopting a familiar tool to facilitate the domain analysis and conceptualization activities, referring to

Table 2 (*continued*)

Clause number	Clause Name	Description	Commonalities with AgiSCOnt
		5.1.1 should be allocated to each of AgiSCOnt's work activities (identified in clause 5.2.1), including staff (clauses 4.6.1 and 4.6.2)	
7.3.4	Budget allocation	The budget for the OE process (clause 1.1.4) should be detailed for each of AgiSCOnt's work activities (identified in clause 5.2.1).	The budget allocation should be proportional to the efforts and resources identified in clause 7.3.3 and to the Scope in 1.1.1
8	**Project assessment and control**		
8.2	Requirements management	This clause details control mechanisms for measuring, reporting, and controlling changes to the OE process management plan.	This clause includes reporting the changes that affect the outputs of Step 1 and Step 2.
8.4	Schedule control	This clause specifies control mechanisms to measure the progress of work.	AgiSCOnt's expected outputs at the end of each Step can serve as milestones for this clause.
8.5	Budget control	This clause specifies control mechanisms to compare the cost of work completed with the costs foreseen in 5.1.1 and 5.2.3 and to identify possible corrective actions.	
9	**Product delivery**	This clause foresees plans to deliver the developed ontology.	This clause should take into account the requirements from AgiSCOnt's Step 1 and Step 3.
10	**Supporting processes**		
10.3	Decision management	Considering the collaborative approach underlying AgiSCOnt, a decision mechanism should be determined by all OE process' participants.	
10.4	Risk management	This clause specifies the risk management plan for identifying, analyzing, and prioritizing project risk factors and describes the contingency plans.	

graphical conceptual maps to involve all domain experts and ontologists in these activities.

The following subsections detail the steps composing AgiSCOnt's approach.

3.2. Management framework

Managerial activities of the OE process are essential for organizations and could require different employees' expertize. Therefore, a structured management framework is pivotal to organizing OE activities properly. The management framework adopted by AgiSCOnt underpins all development stages in an end-to-end perspective and relies on a subset of International Standard ISO/IEC/IEEE 16326:2019 Systems and software engineering – Life cycle processes – Project management (ISO, 2019). The ISO/IEC/IEEE 16326:2019 standard is thought to support project managers in successfully achieving the results of projects related to software-intensive systems and products. The Project Management Plan (PMP) is divided into clauses (specifications of one of its elements). It aims to help project managers specify project objectives, tools, and expected results while scheduling working activities and planning budget allocations.

The management framework provided by AgiSCOnt reuses some of the ISO/IEC/IEEE 16326:2019 standard's elements and clauses that can support ontologists in managing the whole OE process with this methodology: in fact, AgiSCOnt and the ISO/IEC/IEEE 16326:2019 standard share some similarities concerning fundamental aspects such as providing definitions, identifying the scope and objectives of the project, scheduling working activities, time-sequencing constraints for working activities, assessing and controlling of the outputs. The essential elements and clauses of the ISO/IEC/IEEE 16326:2019 standard that compose AgiSCOnt's management framework are summarized in Table 2.

Considering the pivotal role of domain experts and stakeholders in the methodology, Clause 6 (Project context) and its sub-clauses can actively support the project managers in identifying the (external and internal) expertize that can contribute to the Domain analysis and Conceptualization activities (Step 1 of AgiSCOnt), as well as taking part to the development activities and test phases (by proposing Competency Questions, test cases, and participating in the discussion of the results).

Clause 7 and its sub-clauses delve into project management, identifying a time schedule (7.3.2), budget allocation (7.3.4), and resources (human and technological) (7.3.2), and detailing the work activities (7.3.1). On this clause, AgiSCOnt's three-Steps structure (and its activities) can support the project manager in identifying and planning time and budget schedules. In particular, the schedule (7.3.2) and budget (7.3.4) allocation subclauses could benefit from AgiSCOnt's recursive structure of activities. A total amount of time and cost should be carefully planned between the developer and the customer(s), but, the project manager can either provide a preliminary time schedule and budget for each Step or consider the whole Steps as a single time slot and budget item to be allocated according to contingent needs emerging during the OE process (e.g., more time is needed for Domain analysis, less time is required for Conceptualization, an issue emerged during the Step 2 forces the activities to be allocated more time, etc.).

Assessment and control of the project are ensured by clause 8 and its sub-clauses. Also, in this case, AgiSCOnt's expected outputs at the end of each Step can serve as milestones to assess the achievements of the OE process (8.4) and to evaluate the costs of such activities (8.5). Control activities are also extended to changes on the PMP (8.2) that are caused by Step 1 and Step 2 recursive activities. In each iteration, the change in the scope should also consider whether this affects the budget and schedule allocation, modifying also clauses 7.3.2 – 7.3.4 only if during the control specific criticalities emerge. The prioritization of requirements in the Step 1 described below, and the collaborative behaviors between stakholders supported in each stage, should avoid uncontrolled changes that could potentially make the project delayed and overspent. A careful definition of milestones should also consider that more iterative steps are needed if not all milestones are encountered, thus the budget and schedule allocation should be mainly based on milestones, leaving the space for the iterations needed.

Clause 9, dedicated to the delivery of the developed ontology, can support the activities of AgiSCOnt's Step 3 – as long as the flexibility of requirements foreseen for Step 1 is ensured in the previous clauses of the PMP. Finally, clause 10 and its sub-clauses delve into supporting the whole OE process. In this case, relying on the collaborative nature of AgiSCOnt, decisions should be determined by the project manager together with involved stakeholders (10.3), while the identification of potential risks could involve domain experts or other stakeholders only in specific cases (10.4).

3.3. Step 1 – Analysis and conceptualization

The first set of activities to be conducted regards the analysis of the domain and its conceptualization, leveraging existing techniques and tools. The main stakeholders of this phase are the domain experts, including especially the customer(s) when these are knowledgeable of the scope of the domain. These must agree with ontology engineers on the purposes of the target ontology, what it will do, who will use it, and what it is expected to deliver. Once this information is defined, another delicate and fundamental activity takes place: gathering the knowledge relevant to and related to the domain. This activity is indicated as knowledge elicitation, i.e., the adoption of a set of methods and techniques to extract knowledge from domain experts (Shadbolt et al., 2015). It is very time-consuming and expensive, to the point where it has been defined as "knowledge acquisition bottleneck" since 1983 (Hayes-Roth et al., 1983), as the timing for gathering knowledge from experts and documentation is higher than the one required for structuring and formalizing it into software writing (Gavrilova and Andreeva, 2012). Knowledge elicitation is both the core of OE and a pivotal activity, considering that the quality of the output (the ontology) depends on the quality of the acquired knowledge. Among the various knowledge elicitation techniques available, AgiSCOnt suggests adopting existing techniques and tools, well-known for knowledge elicitation (Shadbolt and Burton, 1995), such as unstructured group interviews and Conceptual maps in OE (for example, Castro et al. (2006) proposed a methodology that makes use of conceptual maps in OE; also de Almeida Falbo, 2014 stressed the importance of graphical models as tools to enhance negotiation and communication in the conceptualization activities). Both techniques can be used in decentralized settings, thus allowing also remote and asynchronous cooperation. While unstructured interviews enable domain experts to discuss their expertize in a non-constrained way, Conceptual maps force them to agree on a shared conceptualization. AgiSCOnt does not prescribe any specific approach for discussing the domain at hand or developing the Conceptual map: domain experts can tackle the problems from a bottom-up perspective (i.e., by delving into specific cases and then abstracting them into more general concepts and relationships), or top-down (i.e., by starting with the most general concepts and then detailing specific use cases).

During unstructured interviews, ontology engineers suggest identifying Competency Questions (CQs) (Ren et al., 2014) to investigate the ontology's functional requirements. CQs (and their answers) are formulated in natural language and then used to identify the main concepts and relationships that inform the development of the domain's conceptual map. Therefore, in this phase, ontology engineers can collect information to compile the Ontology Requirement and Specification Document (ORSD) (Suárez-Figueroa and Gómez-Pérez, 2012), a document that keeps track of the knowledge elicitation process. The ORSD – the first output of AgiSCOnt's Step 1 – states the purpose of the ontology, its intended end-users (person or application expecting to be using the ontology), the intended use of the target ontology, and its functional requirements (i.e., a list of content-specific ontology requirements expressed through CQs). The ORSD is a document that must be updated

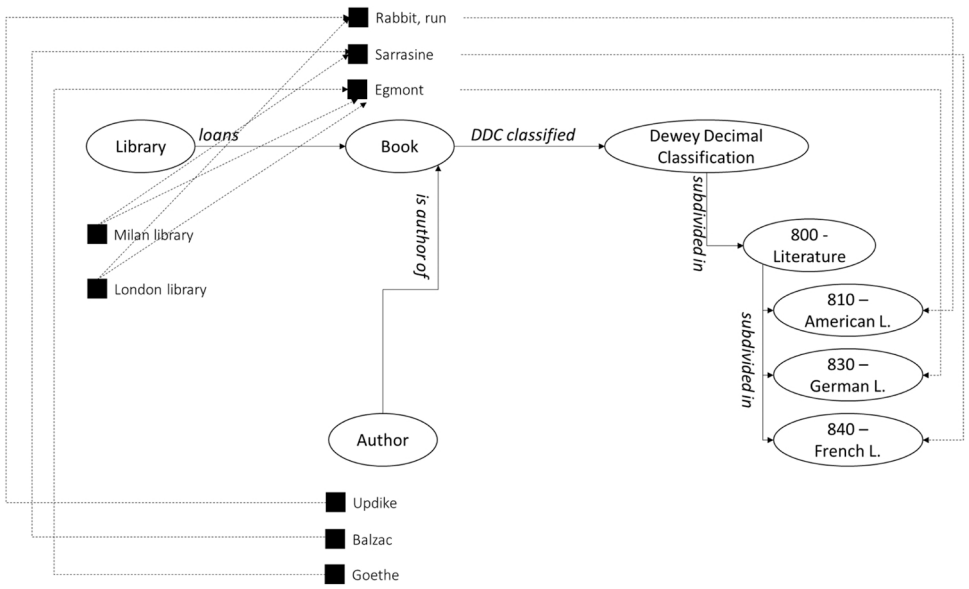

Fig. 2. A Conceptual map representing libraries, the books they loan to readers, their Dewey decimal classification, and their authors. The circles represent concepts, the labeled arrows represent the relationships held among them, and the black squares represent the examples – for which the same relationships modeled for concepts hold.

as domain experts discuss, introduce new terms (and formulate new CQs), and sketch new concepts and relationships in the Conceptual map. Also, the ontology engineers should keep track of the most relevant terms that appear in the CQs and their answers, as well as of those terms that domain experts indicate: these terms constitute the Glossary, a document listing all the terms and their definitions – which are provided (and agreed on) by domain experts. The Glossary – the second document coming out of this phase – provides the definitions for concepts and relationships that are going to be modeled in the ontology and, therefore, must be kept updated. During this step, ontologists and domain experts should also agree on the type of ontology they want to achieve. In particular, players should keep in mind temporal constraints and resources limitations before setting their goals – i.e., it is difficult to develop a reference ontology in a limited amount of time and scarce resources. The type of the ontology they want to develop should be stated in the PM (clause 1.1.1 and 1.1.3). If any changes to the final output's type arise, following PM's clause 1.1.2, the clauses 1.1.1 and 1.1.3 should be modified accordingly.

Drafting a Conceptual map does not rely on any particular modeling technique. Domain experts may use UML or simpler maps, such as those adopted in learning environments. As Conceptual maps are a common and intuitive tool – already appreciated in the context of knowledge management processes (Barão et al., 2017) – it is assumed that domain experts could easily grasp maps and their functioning. In particular, Novak (2013) and Novak and Gowin (1984) Concept maps provide very intuitive schematic devices for the representation of concepts and the relationships holding among them: also, they can provide the means to sketch a preliminary hierarchy of concepts. AgiSCOnt does not force players into adopting any specific type of maps – ontologists and domain experts should autonomously decided which type of conceptual map to use. Nevertheless, AgiSCOnt stresses the importance of some features a Conceptual map should have in order to be used in Step 2. In particular, the conceptual map's granularity is very relevant: the domain's graphical representation should contain all the elements necessary to identify the knowledge correctly, with their different hierarchies, and possibly the type of entities being represented. In other words, the map should indicate what is a concept – i.e., an abstraction, an idea, or a class –, the position a concept occupies with respect to other concepts (its "location" in the hierarchy of the concepts), which relationships a concept holds with the other concepts represented (how a concept is "linked" with the others and through what link). In this way, AgiSCOnt's Conceptual map supports ontologists and domain experts in identifying "is-a" and "part-whole" relationships (as in Castro et al., 2006). The map is completed with examples: domain experts, encouraged by ontology engineers, should provide factual examples of instances representative of a concept and its relationships with other concepts. The examples can be provided by use cases, which are "enacted" in the Conceptual map. Fig. 2 provides an example of a Conceptual map with some instances.

The Conceptual map populated with examples can help the domain experts and ontologists in the early identification of misrepresentation of concepts or relationships, and provides some suggestions on how to model some facts in the subsequent step. Moreover, combined with data collected from unstructured interviews to different stakeholders, the Conceptual map can help elicit a preliminary list of knowledge sources to be reused or modeled from scratch into the target ontology.

3.4. Step 2 – Development and test

This Step foresees the selection of an ontological language and the development of the target ontology and the use cases. The main players are the ontology engineer(s), as highlighted by the gray color in Fig. 1. Although development activities are a prerogative of ontology engineers, domain experts with experience or knowledge of OE and its languages may also participate. The test activities should involve the customer(s) as will be the main user, but especially domain experts to understand if the prototype OEM is fitting requirements while avoiding bias. A key point of the methodology in this step is the reuse of patterns to support authoring.

Development activities in AgiSCOnt are strictly connected to the outputs of Step 1: the development should take advantage of the Conceptual map drafted in the previous step and leverages ODPs to identify modeling solutions that can be reused. In fact, ODPs can accelerate and optimize the development of ontologies, because of their characteristics (modularity, reduced size, possible solutions to recurring modeling problems) (Tudorache 2020). AgiSCOnt suggests ontologists and domain experts to search for existing ODPs that can describe (a portion of) the Conceptual map: considering that each ODP is described with a General Use Case (which details how the ODP works and in which scenarios) (Gangemi and Presutti, 2009), ontology engineers can observe the target ontology's Conceptual map and see whether parts of the map

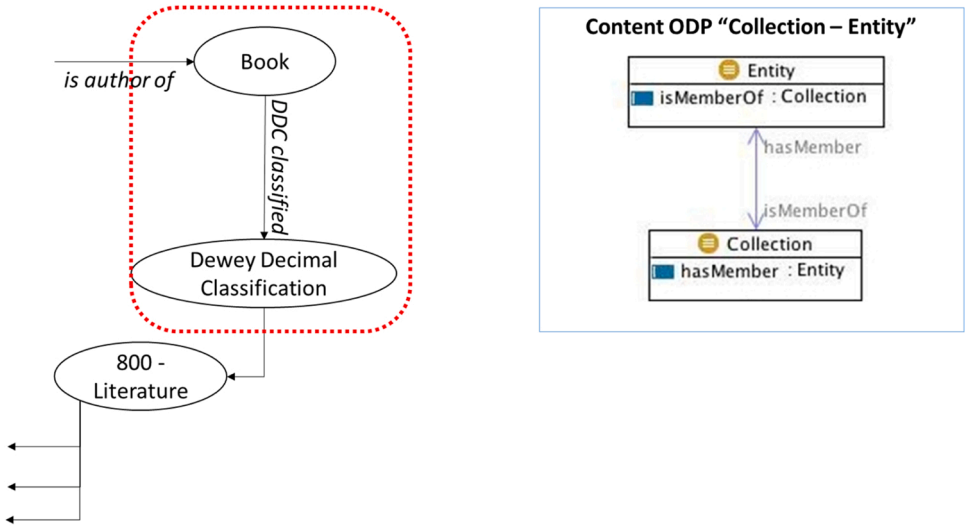

Fig. 3. An example of reuse of the Content ODP Collection-Entity (on the right) to model part of the Conceptual map generated in Step 1 (on the left).

share structural, logical or content similarities with one or more ODP. As in the example shown in Fig. 3, in which players looked for an ODP able to represent the relationship holding between a member of the class ex: Book and a member of the ex:Dewey_Decimal_Classification (DDC), deciding to adopt the Collection-Entity ODP for which a book (an Entity) is a member of a DDC class (a collection of books on the same DDC subject.

Since ODPs are self-contained (Gangemi and Presutti, 2009; Blomqvist et al., 2016), their use as "off-the-shelf" solution can support ontologists in authoring practice. However, it is clear that by reusing an ODP, ontologists and domain experts commit to the philosophical perspective the pattern proposes: for this reason, AgiSCOnt endorses a collaborative approach to identify ODPs and understand whether they match the representation entailed by the Conceptual map. OE process' participants should thus take advantage of ODPs' General Use Cases and documentation to guide their decisions.

To these ends, the ODP catalogs can provide ontologists with support in authoring the entities and relationships that populate the target ontology. In particular, Content and Structural ODPs can effectively support building the target ontology while fostering the reuse of (micro-) ontologies (Blomqvist et al., 2016). Patterns can help novice ontologists resolve authoring issues by identifying part-whole relationships, subclass or superclass relationships among classes, or equivalence between two concepts modeled in the Conceptual map (Poveda Villalón et al., 2010). ODPs can either be a) imported or b) replicated and referred to via their URIs ("soft" reuse (Fernández-López et al., 2019)). Many ODPs are available as "building blocks" for target ontologies: since ODPs are largely independent of any ontological language (or subset), they can be used as ready-to-(re)use solutions to domain modeling problems. In particular, content and logical ODPs can support ontology engineers in understanding different possibilities regarding how to model their target ontology (or parts of their ontology). The ODPs are collected in some ODPs catalogs, although many ODPs are not yet indexed in any of them. In fact, it is possible to elicit and extract patterns from existing ontologies, although it is far from a trivial task – for example, Ruy et al. (2017) provide a method for domain ODP from core ontologies that could be followed by (expert) ontologists.

The reuse of ODPs does not prevent ontologists from interviewing the domain experts to understand whether some existing ontologies can be adopted (because they describe some portions of the Conceptual map). The list of preliminary candidate reusable ontologies identified in the previous step should be discussed, and the actual reuse of one or more models should be agreed upon by all the participants in the OE process. In particular, domain experts may be aware of existing ontologies pertaining the domain at hand, or ontologists can search the repositories or scientific literature to identify reusable ontologies. The candidate ontologies should, therefore, be discussed among all players to reach a decision on whether to reuse (parts of) existing models. Also in this case, ontologists should made domain experts aware that reusable ontologies may be developed with a different expressivity (or with a different ontological language) – therefore, they might require a (partial) re-engineering phase before being reused. In case of partial reuse of ontological resources, AgiSCOnt suggests ontologists to rely on existing and validated approaches to extract modules from the ontologies (e.g., the one described by Doran et al., 2007; MIREOT by Courtot et al., 2011; OntoFox by Xiang and He, 2009). The adoption of core ontologies (Scherp et al., 2011) is also encouraged – in this case, ontologists and domain experts should discuss how to extend the entities of the core ontologies to represent the conceptualization of the domain at hand.

In this Step, ontologists are expected to select the ontological language and its expressivity. In particular, the expressivity of the language should be sufficient to answer to all CQs produced during Step 1. However, it is worth observing that the expressivity (and the selection of the language) may also depend on the ODPs selected – ontology engineers should therefore check that the language profile they selected is expressive enough to represent CQs and ODPs selected in this Step. It may also happen that some CQs might be reformulated to meet the language expressivity, therefore domain experts should be made aware of the limitations of the differences permeating the profiles of the ontological languages.

Once the ontology engineers believe to have reached a stable prototype of the target ontology, they need to test it against a) the list of CQs gathered in Step 1 and b) the set of use cases provided by domain experts. To verify that the target ontology satisfies the requirements expressed with CQs, the prototype must be queried with SPARQL (Pérez et al., 2009). Converting natural-language CQs in SPARQL enables the retrieval of answers for the requirements underlying the CQs. Also, the target ontology should be tested to understand whether it can represent those examples and use cases satisfactorily (b). Therefore, ontologists must collaborate closely with domain experts to populate the target ontology with use cases and verify that the prototype provides all the necessary entities to model them. Suppose the ontology does not provide all the necessary entities or domain experts recognize that some ODPs may be enhanced and/or modified to provide better expressiveness. In that case, ontology engineers can intervene with ad hoc modeling solutions. However, if the testing with use cases underlines a lack of entities, the Conceptual map and CQs may be modified. In this way, Step 2 is able to directly influence the outputs of Step 1 by re-opening the

discussion among domain experts and ontologists to finalize the prototype (a discussion limited to the parts that require interventions and to those entities that are linked to those parts). To ensure that use cases provided by domain experts are not tampered with the conceptualization developed in the prototype, AgiSCOnt suggests involving the domain experts in the use cases gathering process before showing them the finalized prototype. Also, if possible, use cases should be provided by (or gathered from) other stakeholders to which domain experts may have access. The use cases gathering process may lead to the elicitation of new requirements, but it may also happen that some stakeholders would provide some use cases that are not in the scope of the ontology. Consequently, domain experts and ontology engineers must scrutinize use cases and discern those use cases (or parts of use cases) that can provide relevant information to the target ontology.

Finally, the test of the target ontology has also the purpose of evaluating the overall performance of the ontology. For example, it may result that some CQs significantly impact the reasoning performance of the ontology, thus requiring ontologists and domain experts to reformulate them.

3.5. Step 3 – Ontology use and updating

The main output of Step 2, the prototype target ontology, is then adopted in the applications that foresee its use – and that are mentioned in the PMP (clause 9). Step 3 foresees that ontologists and domain experts have reached a version of the target ontology that is ready to be adopted for the purposes identified in Step 1 (and specified in PMP clause 1.1.1). The ontologist here assume a minor role, with the customer(s), the users and other experts, also beyond the ones involved in Step 1 and 2, becoming the major players. According to the specific availability of the target ontology (i.e., if it is publicly accessible or not), the possibility of gathering feedback from external stakeholders can play a pivotal role in updating the ontology. A target ontology complete with documentation enables other ontology engineers and stakeholders to fully understand the modeling choices, the purpose, and the scope of the model. In this way, external stakeholders are encouraged to investigate the possibility of integrating new knowledge into the prototype to match their objectives. In this case, AgiSCOnt refers the ontologists to its first Step so that the OE process can start to modify the original ontology.

Similarly, suppose the ontologists and the domain experts who took part in the OE process of the original target ontology acknowledge the possibility of updating the ontology with new pieces of knowledge. In that case, AgiSCOnt's recursive structure allows modifying the outputs of Step 1 (and, consequently, the outputs of Step 2).

Generally, feedback generation for the target ontology is essential to update and refine it, particularly if it is expected to be reused by third parties (stakeholders not involved in the original OE process) or integrated into third-party applications. According to the PMP, once external feedback is gathered, the project manager(s) or the decision authority (identified in clause 10.3 of the PMP) chooses the best course of action, taking into account also the risks (PMP clause 10.4) related to heavy re-engineering of the target ontology.

One of the outcomes of the interactions occurring among users, other external experts, and ontologists may be the possibility of aligning (or mapping) the developed target ontology with other existing models – or parts of them. The alignment activities generate a set of correspondences between entities of different ontologies (Euzenat et al., 2007), favouring opportunities for ontology reuse or update.

In case no feedback from the practices presented above are collected (by the end of the timeframe decided by players and specified in the PMP), the latest target ontology (output of the previous Step) is adopted as the "final output" (and stated in the PMP). Nonetheless, ontologists and domain experts may keep on modifying the latest output: in this case, players should decide whether to extend the project (and its PMP) or act independently from it.

4. Evaluation of AgiSCOnt

To understand whether AgiSCOnt's structure and suggested techniques can effectively help ontology engineers develop ontologies, the agile OEM needs to be tested and compared to other existing methodologies. AgiSCOnt is evaluated with the three main agile OEMs presented in Section 2.2, i.e., UPONLite (De Nicola and Missikoff, 2016), SAMOD (Peroni, 2016) and RapidOWL (Auer and Herre, 2007), following the evaluation framework proposed in Spoladore and Pessot (2022). The evaluation involved a sample of novice ontology developers who applied the four agile OEMs in use cases of Industry 4.0 transformation.

The experiment for OEM evaluation was conducted in a learning environment. It involved a sample of 16 company employees attending the Advanced professional Master course on Sustainable Industry 4.0 – which included disciplines like Semantic Web and OE, Artificial Intelligence, Sustainability and energy efficiency, Smart buildings, and Building Information Modeling. The participants were all male, with average age 26.1 years, and all employed in Italian medium-sized enterprises in middle-management roles. Part of the Master didactics is a 45-hours long course on Semantic Web, with an introduction on OE (including Ontology Design Patterns).

Participants were divided into three groups (Group A composed of 5 members, Group B composed of 6 members, and Group C composed of 5 members) and each group developed four ontologies, each one using one of the four different agile OEMs – respectively UPONLite, SAMOD, RapidOWL and AgiSCOnt. The participants had at their disposal the complete description and workflow of each OEM, together with the test cases to be considered for the development of the ontology (Table A2). According to Bittner and Spence (2003), test cases were modeled by the researchers to provide informal requirements and features of the systems, together with their outlines (Table A2). All test cases share the same level of complexity, since they were designed in a didactic context to present the same learning challenges and objectives. Their aim was to "force" participants in adopting the constructs they learned throughout the course using agile OEMs. Each test case foresees participants to adopt a class hierarchy, instances, properties, and rules to model the complexity of the case tests. In addition, participants could rely on and collaborate with one or more domain experts. The use cases for the ontologies pertained to different examples of I4.0 implementation areas, so that participants already had adequate know-how in the domain to be modeled. The first case-test was on management of product portfolio in a manufacturing company; the second case-test on developing an e-commerce platform; the third case-test on advanced production planning; and the fourth on developing a recipes recommender system. The domain experts involved were managers from manufacturing and service companies, and a biomedical engineer in the last case-test. In the development, participants adopted Protégé ontology editor 5.5 (Musen, 2015), equipped with Pellet reasoner for logical consistency checking (Sirin et al., 2007). They were asked to complete the OE process in one week for each case-test.

After each ontology was developed, participants were asked to answer a questionnaire to evaluate their perceptions while modeling with the OEM. Finally, developers were asked to express their preferences regarding which of the agile OEMs they experienced addresses at best the features investigated in the questionnaire. In addition, the outputs, i.e., the ontologies developed by participants, were evaluated by two senior developers (researchers) and trainers. Researchers' presence during the experiment was limited to registering secondary data sources and answering questions related to the OEMs to be adopted.

The evaluation framework proposed in Spoladore and Pessot (2022) was chosen to compare AgiSCOnt with other agile OEM. This framework leverages previous works investigating both the features of OEMs and agile programming paradigms. It proposes to assess agile OEMs under two perspectives: the process (i.e., the "journey" ontology engineers have to take to move from a set of informal requirements to a formal model,

Table 3
Results on agile OEMs outcome feature.

OEM outcome features		Case-test 1 with UPONLite			Case-test 2 with SAMOD			Case-test 3 with RapidOWL			Case-test 4 with AgiSCOnt		
		Group A	Group B	Group C	Group A	Group B	Group C	Group A	Group B	Group C	Group A	Group B	Group C
Reused models	*Non-ontological*	Website	Website	Website	Website	Website	Website	no	no	no	ICF	ICF	ICF
	Ontologies	no	no	no	no	no	no	no	no	no	ODPs (1 imported, 1 soft reuse)	ODPs (2 soft reuse)	ODPs (2 imported)
Documentation delivery	*List of Competency Questions*	partial	yes	yes	no	partial	yes	no	partial	partial	yes	yes	yes
	Glossary / Lexicon	partial	yes	yes	yes	partial	no	no	no	no	partial	partial	partial
	Conceptual map	no	no	no	no	no	partial	no	yes	no	yes	yes	yes
Iterations		2	1	3	10	7	6	2	n.d.	n.d.	3	4	3
Relevance of the model	*Domain and range defined*	partial	no	partial	partial	no	partial	no	no	no	partial	yes	partial
	Disjunctions defined	no	no	no	no	no	no	no	no	no	no	yes	yes
	Restrictions defined	no	no	yes	no	no	no	no	no	no	no	no	yes
	Unsatisfiable concepts	no	no	no	no	no	no	no	no	no	no	no	no
Structural measures		17 classes, 5 object properties, 1 datatype properties, 21 individuals, 79 axioms, 2 SWRL rules [incomplete model]	14 classes 9 object properties 4 datatype properties 33 individuals, 143 axioms 1 SWRL rules	26 classes 3 object properties 5 datatype properties 28 individuals, 147 axioms, 6 SWRL rules, 3 annotation properties	20 classes, 12 object properties, 11 datatype properties, 29 individuals, 166 axioms, 6 SWRL rules [incomplete model]	12 classes, 5 object properties, 11 datatype properties, 12 individuals, 96 axioms, 4 SWRL rules [incomplete model]	21 classes, 8 object properties, 12 datatype properties, 41 individuals, 256 axioms, 7 SWRL rules	12 classes, 3 object properties, 15 datatype properties, 29 individuals, 149 axioms, 9 SWRL rules [incomplete model]	17 classes, 5 object properties, 9 datatype properties, 19 individuals, 121 axioms, 5 SWRL rules [incomplete model]	19 classes, 7 object properties, 6 datatype properties, 33 individuals, 153 axioms, 6 SWRL rules [incomplete model]	12 classes, 8 object properties, 5 datatype properties, 42 individuals, 165 axioms, 10 SWRL rules	14 classes, 8 object properties, 4 datatype properties, 43 individuals, 153 axioms, 8 SWRL rules	5 classes, 9 object properties, 4 datatype properties, 44 individuals, 149 axioms, 8 SWRL rules
Time		9.5 h	9.5 h	10 h	12.5 h	12.5 h	11.5 h	14 h	14 h	14 h	13 h	13.5 h	13.5 h
Logical consistency		yes (some incorrect inferences)	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes

Table 4
Results on agile OEMs process features investigation.

OEM process features	Questionnaire item		Assessment of each OEM Mean (Standard Deviation)				Preference among OEMs			
			UPONLite	SAMOD	RapidOWL	AgiSCOnt	UPONLite	SAMOD	RapidOWL	AgiSCOnt
Clarity and simplicity	1	The instructions provided by this methodology are clear	3.13 (0,72)	2.63 (0.50)	2.13 (0.62)	**3.38** (0.62)	**7**	0	0	**9**
	2	This methodology is simple to learn and use	3.19 (0.66)	2.44 (0.73)	2.44 (0.81)	**3.25** (0.68)	5	1	1	**9**
	3	This methodology identifies and details all the steps and activities needed to develop the ontology	3.06 (0.57)	2.88 (0.72)	1.63 (0.50)	**3.25** (0.45)	6	1	1	**8**
	4	Every step of this methodology is clearly presented and detailed	3.13 (0.50)	2.81 (0.66)	1.81 (0.66)	**3.44** (0.63)	**7**	1	0	**8**
Adaptability and flexibility	5	This methodology allows modifying the ontology at any moment of the development process	2.63 (0.50)	**3.25** (0.68)	3.19 (0.75)	3.06 (0.68)	1	**5**	**5**	**5**
	6	This methodology can be applied to different domains of knowledge	**3.31** (0.48)	3.25 (0.45)	2.81 (0.54)	**3.31** (0.60)	**6**	4	0	**6**
	7	This methodology can be used to develop ontologies in domains characterized by different scales and complexity	3.19 (0.54)	3.13 (0.50)	2.56 (0.51)	**3.38** (0.50)	5	5	0	**6**
	8	This methodology enables the personalization of steps/activities, including taking into account domain experts' feedback	3.00 (0.37)	**3.06** (0.44)	**3.06** (0.57)	**3.06** (0.77)	3	3	**5**	**5**
Knowledge management support	9	This methodology provides support in identifying resources to be reused or re-engineered (e.g., other ontologies, taxonomies, conceptualizations)	2.94 (0.57)	3.00 (0.52)	1.81 (0.75)	**3.44** (0.51)	4	4	0	**8**
	10	This methodology provides support in the creation of documentation	3.06 (0.57)	2.75 (0.45)	1.63 (0.62)	**3.56** (0.51)	**7**	1	1	**7**
Teamwork and cooperation support	11	This methodology simplifies the cooperation between developers and domain experts	3.19 (0.40)	3.06 (0.44)	2.56 (0.63)	**3.50** (0.52)	7	0	1	**8**
	12	This methodology requires the domain expert to have an active role in the development team	**3.13** (0.72)	2.81 (0.66)	2.81 (0.75)	**3.13** (0.62)	**6**	1	3	**6**
	13	This methodology eases the teamwork within the development team	3.31 (0.48)	3.06 (0.57)	**3.38** (0.50)	3.31 (0.60)	4	3	**5**	4
Developer perceived effort	14	This methodology enables the development of an ontology in an adequate amount of time	**3.06** (0.44)	2.19 (0.54)	2.13 (0.81)	2.56 (0.51)	**7**	1	3	5
	15	This methodology provides a substantial amount of steps to be followed	2.75 (0.58)	**3.00** (0.63)	2.13 (0.81)	2.06 (0.85)	2	**8**	3	3
	16	The use of iterations foreseen by this methodology simplifies the development of the ontology	2.94 (0.57)	**3.13** (0.62)	2.38 (0.72)	2.81 (0.66)	2	**10**	2	2
Developer perceived role	17	I felt engaged in the team and during the development process	3.25 (0.45)	3.06 (0.57)	2.69 (0.60)	**3.56** (0.51)	6	2	1	**7**
	18	In my opinion, I was able to contribute to the development process using this methodology	3.31 (0.48)	3.25 (0.58)	3.00 (0.63)	**3.44** (0.51)	**4**	**4**	**4**	**4**
	19	My role in the development team was always clear and explicit	3.31 (0.48)	3.25 (0.58)	2.69 (0.48)	**3.38** (0.62)	5	2	2	**7**
Innovation support	20	Using this methodology, the development team is encouraged to adopt new ideas to achieve the scopes of the ontology.	3.13 (0.62)	3.25 (0.58)	**3.31** (0.48)	2.94 (0.68)	3	3	**7**	3
	21	This methodology encourages developers to be creative.	3.06 (0.44)	3.19 (0.40)	**3.25** (0.58)	3.00 (0.37)	2	6	**5**	3
	22	This methodology allows for making and fixing mistakes easily.	3.13 (0.50)	**3.25** (0.45)	3.19 (0.66)	**3.25** (0.59)	2	**6**	4	4
	23	Team members took the initiative to perform the tasks foreseen by this methodology.	**3.19** (0.40)	**3.19** (0.54)	3.06 (0.57)	3.00 (0.52)	**6**	5	2	3
	24	Using this methodology, the development team can freely take decisions at any moment of the modeling and development phases.	2.94 (0.57)	3.13 (0.62)	**3.19** (0.54)	3.13 (0.81)	3	4	**6**	3
Operational support	25	The methodology provides me with operational support for the activities pertaining the analysis of the domain.	2.69 (0.60)	2.31 (0.48)	1.75 (0.45)	**3.31** (0.48)	4	3	0	**9**
	26	The methodology provides me with operational support for the activities pertaining the development of the ontology.	2.00 (0.52)	1.94 (0.68)	1.56 (0.63)	**3.50** (0.52)	4	**2**	0	**10**

Table A1
Comments provided by participants and their mapping to process features.

Group	Quotations	Case-test/ OEM	Process feature
A	"It is interesting that we can use any tool we want to make a conceptualization […] This can make some of the work easier and faster."	1/ UPONLite	Teamwork and cooperation support
C	"It is not clear when *Parthood* and *Predication* start. We ended up doing them at the same time."	1/ UPONLite	Clarity and simplicity
C	"Most of [*UPONLite*] steps require everyone to participate and share their opinion, otherwise the ontology may not be shared by everyone"	1/ UPONLite	Teamwork and cooperation support
B	"This methodology makes you prepare more than half of the documentation you need."	1/ UPONLite	Knowledge management and support
A	"Most of the time effort is dedicated to steps from 1 to 5."	1/ UPONLite	Developer perceived effort
A	"In practice there are no suggestions on how to identify test cases and model them"	2/SAMOD	Operational support
B	"There are not enough instructions in identifying the first modelet"	2/SAMOD	Clarity and simplicity
C	"This methodology requires a lot of time […] test case identification for the modelets is not easy"	2/SAMOD	Developer perceived effort
C	"Each time you merge [*a modelet*] to the model you have to modify the whole documentation."	2/SAMOD	Knowledge management and support
B	"We would have preferred the domain expert could take part to the development activities […] he could have checked what we were doing while we were doing it"	2/SAMOD	Teamwork and cooperation support
A	"I am not sure [*SAMOD*] could be adopted in my company. It takes too much time."	2/SAMOD	Developer perceived effort
C	"It is not clear where we should start to model this case-test."	3/ RapidOWL	Clarity and simplicity
A	"[*RapidOWL*] does not limit you, the group must find solutions because the methodology does not give you any hint on how to find them."	3/ RapidOWL	Innovation support
B	"There is no structure, so we can do everything at any time."	3/ RapidOWL	Innovation support
C	"Without guidance you cannot understand when an iteration is over […] it is too fluid"	3/ RapidOWL	Developer perceived effort
C	"When you have to develop with Protégé […] there are not guidelines at all."	3/ RapidOWL	Operational support
B	"It is not hard to follow [*AgiSCOnt*]'s steps because they are only three."	4/ AgiSCOnt	Clarity and simplicity
C	"If we had a deeper knowledge of ontology design patterns we could be faster in developing the ontology"	4/ AgiSCOnt	Developer perceived effort
B	"[*AgiSCOnt*] tells you to reuse patterns […] in this way there is less room for creativity, because we are adopting an existing solution".	4/ AgiSCOnt	Innovation support
A	"When you sketch the conceptual map with the domain experts you are basically developing the ontology […] they are the key to everything with this methodology."	4/ AgiSCOnt	Teamwork and cooperation support
C	"Using ontology design patterns helps in detailing the ontology […] they [*ODPs*] are "ready for use" bits of our ontology."	4/ AgiSCOnt	Operational support
B	"With the patterns you always reuse something."	4/ AgiSCOnt	Knowledge management and support
A	"Documentation is done by asking questions to domain experts."	4/ AgiSCOnt	Knowledge management and support

and ontologists' perspectives in following the methodology's instructions, using supportive tools, conducting activities as specified, etc.) and the outcome (i.e., whether the results of the process – the ontology and its characteristics – are in-line with the purposes declared for the ontology during the process, and whether the developed model is logically consistent). The outcome is assessed through basic metrics from ontology evaluation (Burton-Jones et al., 2005; Lourdusamy and John, 2018), including the reuse of formal (or informal) existing models, the documentation of the target ontology, the number of times the agile OEM was browsed to achieve the delivery of the target ontology (iterations), the degree to which the ontology provides the information that is expected to be modeled, the features in the selected ontology language that are used to engineer the ontology, the logical consistency of the ontology and the amount of time spent to develop the ontology. The process assessment includes features such as clarity and simplicity of the OEM, flexibility, documentation and cooperation support.

These data were collected in a 26-item questionnaire, with participants evaluating each questionnaire item on a 4-point Likert scale ranging from "1-strongly disagree" to "4-strongly agree" at the end of each ontology development (and thus OEM complete application). A total of 64 questionnaires were collected as primary data, and these were then triangulated with secondary data sources (i.e., participant observations, field notes, and comments raised during the development tasks). In addition, a total of 16 questionnaires were collected with preferences expressed by the participants at the end of the overall experiment.

5. Results and discussion

A total of 12 ontologies were analyzed by trainers according to the outcome features. The results of the analysis are summarized in Table 3.

Concerning the process features, Table 4 summarizes the answers provided by participants – registered as mean values with standard deviation. Secondary data (comments and notes from participants) are reported in Appendix in Table A1. Examples of the materials (e.g., CQs, lexicons, glossaries, conceptual maps, etc.) produced by participants are provided within the Supplementary materials.

5.1. Results of agile OEMs outcome evaluation

Results for the outcome clearly indicate that the quality of the work performed is independent of the temporal order of developed OEMs (i.e., there are no better ontologies in case-test 3 or 4 only because the participants already carried out development tasks and achieved learning outcomes). The completeness of the ontologies can be mainly ascribed to the limited expertize of developers (being novice ones, and in a learning environment). All the ontologies could be better in terms of Relevance of the model: only some of the available constructs were adopted to represent the complexity of the domains in a coherent way. This aspect also impacts the Structural measures, which suggest that half of the ontologies produced failed to represent all the characteristics of the domains underlying the four case-tests. No Unsatisfiable concepts were registered, as well as participants provided no Logical inconsistent ontologies. However, in one case (case-test 1, Group A), the need for more quality in Structural measures led participants to develop SWRL rules that draw incorrect inferences in the domain. Nevertheless, there are

Table A2
The four domains, case tests, and OEMs adopted in the study.

Case-test	Domain	Case test	OEM
1	Product portfolio	An ontology for representing and managing a company's product portfolio producing custom packaging boxes. The model must be able to represent different types of products, including characteristics such as: • Raw materials; • Available size(s) for each product; • Available colors for each product; • Available shapes for each product; • Availability of finished products; • Availability of raw materials in the warehouse; • Processing time for each type of product. The framework is designed to support B2B customers in understanding the company's product portfolio and estimated time between an order and shipment (orders for available products, custom orders requiring production). **Domain expert**: orders department senior executive – local box factory, SME	UPONLite
2	E-commerce platform	An online clothing store scenario. From a B2C perspective, the framework must be able to: • Represent different products and their information (type of product, color, available sizes, price); • Represent customers; • Model previous orders performed by customers; • Suggest new products to customers according to their chromatic preferences, sizes of previously purchased products, and products availability. From a backend perspective, the framework must be able to: • Model the cost for each product; • Calculate the margin for each product; • Calculate the overall margin on each order. The framework is designed to support e-commerce managers in automating recommended products marketing activities, taking into account bot customer preferences and margin-based benefits. **Domain expert**: e-commerce marketing & product manager – regional clothing franchise	SAMOD
3	Production planning	A framework to plan capacity utilization of shop floor resources able to: • Represent workers and their general information (name, surname, age, role on the shop floor); • Each worker's hourly cost, including additional costs for hours between 8:00 PM – 6:00 AM (night shift); • Represent shop floor's machinery, including information on their functioning (time to set up, amount of time for completing a task, required breaks); • Model working shifts (on three shifts: 6.00 AM - 2:00 PM; 2:00 PM - 10:00 PM; 10:00 PM - 6:00 AM); The framework is designed to provide information such as: • Identify machinery's periods of inactivity; • Calculate total personnel cost for each shift and for every working day; • Support organization manager in planning working shifts (considering that a worker may work the same shift for an entire week). **Domain expert**: organizational manager – plastic molding SME	RapidOWL
4	Recipes recommender system	An ontology for recommending patients healthy recipes. The model: • From a patient perspective, must be able to: ▪ Represent patients and their personal data • Illustrate patients' health condition, which can be characterized also by more than one disease • From a recipe perspective, must be able to: ▪ Represent recipes and their names ▪ Represent all the steps composing a recipe, taking advantage of the "Recipe book" provided by the domain expert • From a recommendation perspective, the system must be able to: ▪ Propose, for each patient, one or more recipes deemed suitable for his/her health condition The recommender system is designed with the aim of helping patients identify the most healthy recipes according to their health conditions **Domain expert**: biomedical engineer (involved in a nutrition research project); the domain expert provides a simplified version of a recipe book, containing for each recipe indications on the suitability of the dish for people characterized by specific health issues	AgiSCOnt

clear differences that can be observed for AgiSCOnt OEM compared to the other three agile OEMs.

Specifically, development with AgiSCOnt was able to provide three complete models. From a Reuse perspective, it was interesting to notice that other OEMs resulted in the reuse of only non-ontological models (especially websites) as references for TBox modeling. At the same time, with AgiSCOnt all Groups adopted the ontology International Classification of Functioning, Disability and Health (ICF), following the domain expert's suggestion. However, none of the Groups reused the existing ontology on ICF (BioPortal, 2012), but they decided to model the bits of the ICF they were interested in directly in the model (soft reuse).

Participants were able to import and use some ODPs – namely: Group B and C imported the Sequence ODP (Gangemi, 2008), a submission to the Ontology Design Patterns Portal, to model the sequence of steps composing each recipe; and the ODP for health condition modeling with ICF, which was reused based on an open access paper that provides clear pictures and General Use Case of the ODP (Spoladore and Sacco, 2018). Although Groups reused patterns and existing sources to model their ontologies, it must be noted that the Relevance of the model features domain and range were partially compiled by ODPs adopted – so the effort participants had to make was limited. The Documentation delivery for case-test 4 can be comparable to the one in UPONLite, with the list of CQs and the Glossary produced by all participants for case-test 1, although not always complete. In particular, it is worth noticing that the only incomplete model (Group A) is the one that has the least adequate documentation: this fact also impacted the Relevance of the model and Structural measure outcome features for the same Group (see also: Supplementary materials).

In general, the developed ontologies are poor (i.e., they do not adopt the full extent of the ontological constructs to model the domains presented in each case-test), even though Structural measures in case-test 4 indicate a slight enhancement that allowed Groups to provide all the entities necessary to represent all the concepts and provide the recommendations. It is interesting to observe that all Groups dedicated a considerable amount of Time for the development with AgiSCOnt, the second highest if compared with other OEMs: this may partly be due to the fact that Groups had to browse the ODPs to understand the ones to reuse (Table A1). Conversely, SAMOD resulted in less development time, but was perceived as one of the most time-consuming agile OEMs (according to Developer perceived effort, as explained in the Section 5.2). In this case, the effort of defining domain, ranges, and class expressions was left to the last step of the methodology. Developing with RapidOWL required the highest amount of Time (Table 3) and raised concerns about the starting point of the OEM: two Groups were unable to identify when one Iteration of the ontology started – thus experiencing mainly obstacles with the high level of self-organization required in identifying the practices to be tackled at first (among the set of RapidOWL's ten practices). The higher development time in AgiSCOnt could also be explained by developers' higher commitment to collect feedback from the domain expert. Conversely, participants complained that domain experts could not take a direct role in SAMOD (as prescribed by the OEM) and faced issues in identifying the test cases to develop the modelets (Table A1). This could partly explain the paucity of Groups A and B's ontologies, which lack some structural measures to properly represent some concepts. Finally, the ontologies developed with AgiSCOnt are all Logical consistent, and no Unsatisfiable concepts are present.

5.2. Results of agile OEMs process evaluation

By taking into account each process feature, Clarity and simplicity shows that participants particularly appreciated AgiSCOnt, and UPONLite immediately follows: both OEMs provide a structured and limited set of steps, with clear dependencies among them. This feature seems to make a difference in understanding the methodology (Table 4). One Group needed help understanding how to divide the Parthood and Predication steps in UPONLite, declaring that they conducted steps from 3 to 5 simultaneously (Table A1). Dependencies holding among steps seem to cover a relevant role in the perception of the Clarity of an OEM: this might explain the very close scores gained by UPONLite and AgiSCOnt (Table 4).

Knowledge management and support, which includes the reuse of existing resources (both ontological and non-ontological), confirms AgiSCOnt as the most suitable OEM for providing support in identifying other resources (item 9) and generating documentation (item 10). Conversely, the results of the process feature Adaptability and flexibility recognize SAMOD (and its iterative structure) as the OEM that better enables to modify the ontology at any moment. Item 6 (the possibility to reuse an OEM for different domains) sees UPONLite and AgiSCOnt equally appreciated by participants. The personalization of activities according to developers' needs (item 8) is a feature recognized in all OEMs that share an iterative structure.

In Teamwork and cooperation features, the cooperation between developers and domain experts was perceived as facilitated by AgiSCOnt, with UPONLite following very closely (item 11). Nevertheless, the teamwork in the development team (item 13) is better evaluated in RapidOWL as it is judged as the agile OEM that "forces" participants to come together with solutions (Table A1).

AgiSCOnt is less considered in the feature Developer perceived effort than other OEMs: SAMOD is deemed the OEM that consists of more steps to be followed (item 15) but whose iterative structure can simplify the development of the ontology (item 16). UPONLite is judged to support the development of ontologies in an adequate amount of time (item 14), even if in AgiSCOnt the use of ODPs could have made the development process faster (Table A1).

In Developer perceived role, AgiSCOnt and UPONLite present very similar scores, but with a slight preference for AgiSCOnt, thanks to the use of the management framework. It is worth noticing that item 18 indicates that participants do not have a clear preference among which of the agile OEM fosters a better perception of personal contribution to the development activities. This could be interpreted as a feature characterizing in general agile OEMs, with more space left to the self-organization and self-activation.

With regard to the Innovation support, which portrays the creativity and innovation of the agile OEMs, AgiSCOnt is recognized more valuable than the other OEMs only to support fixing mistakes easily, together with SAMOD (item 22). Differently from RapidOWL – where the absence of a clear dependency among this OEM's practices puts the creativity of participants to the test to achieve solutions (item 20), in a creative and collaborative effort to identify the best modeling solutions (item 21), and without being constrained by the methodology's structure (item 24) – AgiSCOnt does not emerge as a particularly creative OEM. This statement is also endorsed by some comments participants provided (Table A1), suggesting that relying on existing solutions (to be reused, like ODPs) reduces the creative efforts performed by the Groups of participants.

However, the Operational support feature indicates that AgiSCOnt is the most suitable OEM to provide practical support and suggestions to developers. This OEM is explicitly developed to foster ontology authoring, with suggestions to reuse ODPs and a Conceptual map that asks participants to sketch a TBox on paper. However, it is interesting to note that RapidOWL scored below average in this feature, in particular for the support to be provided during the development of the model (item 26), with participants commenting on the fact that for practical development, this methodology's practices are not particularly useful (Table A1).

5.3. Discussion and implications

AgiSCOnt's performance enabled participants to provide complete models, although not particularly rich in constructs. However, the results – in particular, the differences in output – indicate that participants adopted different approaches to model the same test cases. From an authoring perspective, the selection, reuse, and adoption of ODPs is perceived as useful (Tables 4 and A1), but not effortless (i.e., it takes time to find them – Table 3). These results are also in line with those retrived in an experiment aimed at evaluating the role of ODPs in OE (Blomqvist et al., 2009).

Beyond assessing the validity of AgiSCOnt against different criteria, the results of this study highlight some implications in OE and the successful deployment of agile OEMs for organizations' knowledge management. It is essential to involve domain experts throughout the OE process: the OEMs that managed to leverage the contributions of domain

experts were evaluated as more complete from an outcome perspective. However, this study confirms that the high "level of agility" of an OEM does not necessarily translate into high-quality ontologies (as it happens for RapidOWL as shown in Spoladore and Pessot, 2022). There is a trade-off between the possibility of being creative, adapting, and innovating the OE process and the support the methodology provides. Moreover, the outcome is characterized by better quality when clear guidelines are provided. Regarding supporting novice ontologists with authoring, AgiSCOnt proved effective in developing ontologies that reused existing models and patterns. The authoring recommendations should be perceived more as levers for a creative ontology development that does not neglect previous knowledge to be reused and shared, directly impacting the resources required and the outcome's success (Feilmayr and Wöß, 2016).

This study provides some remarks for organizations facing the possibility of adopting an OEM to support the capture and transformation of large amounts of data into useful information to be properly exploited (Jaskó et al., 2020). Some agile methodologies (RapidOWL, SAMOD) require the users to have some previous knowledge of OE: as the results illustrate, lacking knowledge of OE results in poor models and higher amounts of time dedicated to development. On the contrary, more structured OEMs (UPONLite and AgiSCOnt) seem to support users through the OE process, with more valuable outcome results and contained times for development. Clarity and simplicity of OEMs is fundamental for adopting agile approaches, which leverages a deep understanding of their contents (Kiv et al., 2022). The difficulty related to OE concerns the elicitation of organizations' tacit knowledge, expertize, and know-how and their formalization (Meski et al., 2021). In this regard, relying on a methodology supporting knowledge acquisition and involving all experts should help considerably reduce the knowledge acquisition bottleneck while enhancing the quality of the model. This is important especially in those organizations relying on information and communication technologies and thus producing large amounts of data, such as manufacturing companies (Centobelli et al., 2019) and R&D-intensive businesses (Barão et al., 2017). In addition, the promising results from AgiSCOnt stress the importance of having a comprehensive methodology that considers not only the engineering but also the management perspective, which is pivotal for implementing the ontologies and knowledge management systems based on them on a large scale (Mora et al., 2022).

Finally, AgiSCOnt's steps are not limited to modeling ontologies in the I4.0 area: the methodologies instructions can be adopted as a guide to engineer models in any field, leveraging the collaboration among ontologists and experts.

5.4. Limitations and future work

This Section highlights some of the inherent and experimental limitations of the proposed OEM.

AgiSCOnt leverages ODPs to foster authoring and reuse, thus it suffers from some of the "traditional" problems permeating pattern-based OE. For example, ontologists' access to an updated catalog of ODPs is pivotal to ensure an "off-the-shelf" reuse. There is a lack of relevant patterns both at a generic and domain-specific level. In particular, identifying patterns that have been validated against specific requirements (or real use cases) is still a non-trivial task (Blomqvist et al., 2016). Also, many patterns can be identified from existing ontologies. However, this activity is far from being trivial and researchers devoted many works to this purpose (e.g., Ruy et al., 2015; Ruy et al., 2017). However, further reaserch on the topics of engineering with ODPs, extracting patterns, and make them available is needed (Blomqvist et al., 2016; Tudorache 2020).

The empirical study also presents some limitations. The sample of participants is limited in number and origin (all participants are employed in Italian SMEs). Future research could investigate whether samples from other countries and different organizations confirm the results retrieved in the Italian scenario.

The experiment took place in a learning setting, preventing the possibility of testing the effective implementation of the developed ontologies while granting a controlled environment that simulates a team belonging to the same organization. The setting did not allow mixing the three groups of participants, which would have increased the complexity of tracking changes to groups and would have required a larger set of case-tests. In addition, a "learning curve" effect should be considered as potentially influencing the assessment by the developers once more case-tests and OEMs were developed. Each case-test was developed with one agile OEM: future research could investigate whether different ontologists applying different OEMs to the same domain may develop significantly different ontologies. Also, the evaluation of the OEMs performed in this work is limited to agile methodologies. A future research direction can take into account modifying the evaluation framework (in particular, the process features) to enable assessing differences and similarities between agile approaches (including AgiSCOnt) and waterfall and lifecycle workflows OEMs. Finally, AgiSCOnt could be evaluated by developers with more expertize in OE and in the agile approach, in order to identify eventual changes in perceptions of OEM features.

In conclusion, the results on AgiSCOnt's are promising, but preliminary: more tests are needed to tune the OEM and to confirm the findings of this study. Also, the methodology's domains of application are not limited to I4.0. For example, AgiSCOnt is currently being used to develop an ontology-based decision support system for diabetic patients within a hospital, leveraging the collaboration of clinical personnel and their expert knowledge (Spoladore et al., 2023). The adoption of the methodology to support collaborative ontology development in different domains and kinds of organizations could bring out some interesting insights, contributing to the further tuning of AgiSCOnt and its steps, and a wider adoption of the methodology by practitioners. The digital transformation and the consequent exploitation of a vast amount of sources and types of data in knowledge management and decision-making processes with agile approaches is becoming pivotal not only for companies, but also for a wide range of working contexts that range from governments to educational institutions (Barão et al., 2017; Burchardt and Maisch, 2019).

6. Conclusions

This paper introduces a novel agile OEM, i.e. the Agile, Simplified and Collaborative Ontology engineering methodology (AgiSCOnt), developed to tackle some of the traditional issues of OE (management, knowledge acquisition, and authoring in particular). To this aim, the methodology combines macro-level instructions with micro-level techniques, thus capitalizing on existing literature on OEM (and especially agile ones) while introducing novel elements that address existing gaps in OEMs. AgiSCOnt is tested in the context of companies' digital transformation towards I4.0, where agility and rapid prototyping are essential together with cost-effectiveness and profitability. The OEM is then compared to three agile methodologies, leveraging an assessment framework that considers both the ontologists' experience and perspective (process) and the quality of the developed ontologies (outcome). The main features differentiating AgiSCOnt from other OEMs are the support to knowledge elicitation, authoring and reuse, integrated with a simple management framework. From the preliminary results, AgiSCOnt is perceived as clear and simple while being flexible and adaptable enough. It balances an agile approach with the documentation of the ontologies for knowledge management and sharing, and a close collaboration among ontologists and domain experts. These features are pivotal considering the application of OEM in an industrial context, and by novel ontologists within different kinds of organizations undertaking a digital transformation of their knowledge management process.

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.

Appendix

See Table A1 and A2.

Data availability

No data was used for the research described in the article.

Appendix A. Supporting information

Supplementary data associated with this article can be found in the online version at doi:10.1016/j.compind.2023.103979.

References

, 2019International Organization for Standardization (ISO), 2019. ISO/IEC/IEEE 16326: 2019 Systems and software engineering — Life cycleprocesses — Project management; Second edition.

de Almeida Falbo R. , 2014. SABiO: Systematic Approach for Building Ontologies. In *Onto. Com/odise@ Fois*.

Auer S. , 2006. The RapidOWL methodology–towards agile knowledge engineering. In: Proceedings of the 15th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'06) (pp. 352–357). IEEE.

Auer S. , 2010. RapidOWL: A Methodology for Enabling Social Semantic Collaboration. In Social Computing: Concepts, Methodologies, Tools, and Applications (pp. 669–692). IGI Global.

Auer S., Herre H. , 2007. RapidOWL—An agile knowledge engineering methodology. In: Proceedings of the International Andrei Ershov Memorial Conference on Perspectives of System Informatics. Springer, pp 424–430.

Baader, F., Horrocks, I., Sattler, U., 2008. Description logics. Found. Artif. Intell. 3, 135–179.

Barão, A., Vasconcelos, J.B., de, Rocha, Á., Pereira, R., 2017. A knowledge management approach to capture organizational learning networks. Int. J. Inf. Manag. 37, 735–740.

BioPortal , 2012. International Classification of Functioning, Disability and Health (ICF) Ontology - available onlines: ⟨https://bioportal.bioontology.org/ontologies/ICF⟩.

Bittner, K. , & Spence, I. , 2003. Use case modeling. Addison-Wesley Professional.

Blomqvist E. , Gangemi A. , Presutti V. , 2009. Experiments on pattern-based ontology design. In: Proceedings of the fifth international conference on Knowledge capture (pp. 41–48).

Blomqvist, E., Hitzler, P., Janowicz, K., Krisnadhi, A., Narock, T., Solanki, M., 2016. Considerations regarding ontology design patterns. Semant. Web 7 (1), 1–7.

Burchardt, C., Maisch, B., 2019. Digitalization needs a cultural change–examples of applying agility and open innovation to drive the digital transformation. Procedia Cirp 84, 112–117.

Burton-Jones, A., Storey, V.C., Sugumaran, V., Ahluwalia, P., 2005. A semiotic metrics suite for assessing the quality of ontologies. Data Knowl. Eng. 55 (1), 84–102.

Castro, A.G., Rocca-Serra, P., Stevens, R., Taylor, C., Nashar, K., Ragan, M.A., Sansone, S. A., 2006. The use of concept maps during knowledge elicitation in ontology development processes–the nutrigenomics use case. BMC Bioinform. 7 (1), 1–14.

Centobelli, P., Cerchione, R., Esposito, E., 2019. Efficiency and effectiveness of knowledge management systems in SMEs. Prod. Plan. Control 30, 779–791.

Chimienti M. , Dassisti M. , De Nicola A. , Missikoff M. , 2009. Evaluation of ontology building methodologies – a method based on balanced scorecards. In: Proceedings of the International Conference on Knowledge Engineering and Ontology Development. SCITEPRESS, pp 141–146.

Corcho, O., Fernández-López, M., Gómez-Pérez, A., & López-Cima, A., 2005. Building legal ontologies with METHONTOLOGY and WebODE. Law and the semantic web: legal ontologies, methodologies, legal information retrieval, and applications, 142–157.

Courtot, M., Gibson, F., Lister, A., Malone, J., Schober, D., Brinkman, R.R., Ruttenberg, A., 2011. MIREOT: The minimum information to reference an external ontology term. Appl. Ontol. 6 (1), 23–33.

Davies, K., Keet, C.M., Lawrynowicz, A., 2019. More effective ontology authoring with test-driven development and the TDDonto2 tool. Int. J. Artif. Intell. Tools 28 (07), 1950023.

De Lille, N., & Roelens, B. , 2021, March. A practical application of upon lite for the development of a semi-informal application ontology. In: Proceedings of the 15th International Workshop on Value Modelling and Business Ontologies (pp. 63–70). CEUR-WS.

De Nicola, A., Missikoff, M., 2016. A lightweight methodology for rapid ontology engineering. Commun. ACM 59, 79–86.

De Nicola, A., Missikoff, M., Navigli, R., 2009. A software engineering approach to ontology building. Inf. Syst. 34, 258–275.

Doran P., Tamma V., Iannone L. , 2007. Ontology module extraction for ontology reuse: an ontology engineering perspective. In: Proceedings of the Sixteenth ACM Conference on Conference on information and knowledge management (pp. 61–70).

Euzenat, J., Shvaiko, P., et al., 2007. Ontology Matching. Springer.

Evans, N., Price, J., 2020. Development of a holistic model for the management of an enterprise's information assets. Int. J. Inf. Manag. 54, 102193.

Fatfouta, N., Le-Cardinal, J.S., 2021. An ontology-based knowledge management approach supporting simulation-aided design for car crash simulation in the development phase. Comput. Ind. 125, 103344.

Feilmayr, C., Wöß, W., 2016. An analysis of ontologies and their success factors for application to business. Data Knowl. Eng. 101, 1–23.

Fernández-López M., Gómez-Pérez A., Juristo N. , 1997. Methontology: from ontological art towards ontological engineering.

Fernández-López, M., Gómez-Pérez, A., Sierra, J.P., Sierra, A.P., 1999. Building a chemical ontology using methontology and the ontology design environment. IEEE Intell. Syst. Appl. 14 (1), 37–46.

Fernández-López, M., Poveda-Villalón, M., Suárez-Figueroa, M.C., Gómez-Pérez, A., 2019. Why are ontologies not reused across the same domain? J. Web Semant. 57, 100492.

Gangemi A. , 2008. Content ontology design patterns: sequence - available online: ⟨http://ontologydesignpatterns.org/wiki/Submissions:Sequence⟩.

Gangemi, A., Presutti, V., 2009. Ontology design patterns. Handbook on Ontologies. Springer, pp. 221–243.

Garcia, A., O'Neill, K., Garcia, L.J., Lord, P., Stevens, R., Corcho, O., Gibson, F., 2010. Developing ontologies within decentralised settings. Semant. e-Sci. 99–139.

Gavrilova, T., Andreeva, T., 2012. Knowledge elicitation techniques in a knowledge management context. J. Knowl. Manag.

Gavrilova, T., 2007, September. Ontological engineering for practical knowledge work. In: Proceedings of the International Conference on Knowledge-Based and Intelligent Information and Engineering Systems (pp. 1154–1161). Springer, Berlin, Heidelberg.

Gobin B. , 2014b. Assessing the suitability of existing Agile Ontology Engineering Methodologies for ontology module development.

Gobin B. , 2014a. A Quantitative Framework for assessing Agile Ontology Engineering Methodologies. In: Proc. International Conference on Web and Information Systems. Citeseer.

Gobin B.A. , 2014c. Using the 4-DAT Tool to evaluate Agile Ontology Engineering Methodologies.

Gómez-Pérez A., Suárez-Figueroa M.C. , 2009. NeOn methodology for building ontology networks: a scenario-based methodology.

Gruber, T.R., 1993. A translation approach to portable ontology specifications. Knowl. Acquis. 5, 199–220.

Guarino, N., Oberle, D., Staab, S., 2009. What is an ontology? Handbook on Ontologies. Springer,, Berlin, Heidelberg, pp. 1–17.

Hakkarainen S. , Strasunskas D. , Hella L. , Tuxen S. , 2005. Weighted Evaluation of Ontology Building Methods. In: CAiSE Short Paper Proceedings.

Hayes-Roth, F., Waterman, D.A., Lenat, D.B., 1983. Building Expert Systems. Addison-Wesley Longman Publishing Co., Inc.,.

Jarrar, M., Meersman, R., 2008. Ontology engineering–the DOGMA approach. Advances in Web Semantics I. Springer, Berlin, Heidelberg, pp. 7–34.

Jaskó, S., Skrop, A., Holczinger, T., et al., 2020. Development of manufacturing execution systems in accordance with Industry 4.0 requirements: a review of standard-and ontology-based methodologies and tools. Comput. Ind. 123, 103300.

Keet, M., 2018. An Introduction to Ontology Engineering. Maria Keet, Cape Town, South Africa.

Kiv, S., Heng, S., Wautelet, Y., et al., 2022. Using an ontology for systematic practice adoption in agile methods: expert system and practitioners-based validation. Expert Syst. Appl. 195, 116520.

Kotis, K., Vouros, G.A., 2006. Human-centered ontology engineering: the HCOME methodology. Knowl. Inf. Syst. 10 (1), 109–131.

Kotis, K., Vouros, G.A., Alonso, J.P., 2005. HCOME: A tool-supported methodology for engineering living ontologies. Proceedings of the International Workshop on Semantic Web and Databases. Springer, Berlin, Heidelberg, pp. 155–166.

Kotis, K.I., Vouros, G.A., Spiliotopoulos, D., 2020. Ontology engineering methodologies for the evolution of living and reused ontologies: status, trends, findings and recommendations. Knowl. Eng. Rev. 35.

Kumar, V.R.S., Khamis, A., Fiorini, S., et al., 2019. Ontologies for industry 4.0. Knowl. Eng. Rev. 34.

Lourdusamy R., John A., 2018. A review on metrics for ontology evaluation. In: Proceedings of the 2018 2nd International Conference on Inventive Systems and Control (ICISC). IEEE, pp. 1415–1421.

Mariño, B.D.R., RodríGuez-FóRtiz, M.J., Torres, M.V.H., Haddad, H.M., 2018. Accessibility and activity-centered design for ICT users: ACCESIBILITIC ontology. IEEE Access 6, 60655–60665.

Meski, O., Belkadi, F., Laroche, F., et al., 2021. A generic knowledge management approach towards the development of a decision support system. Int. J. Prod. Res. 59, 6659–6676.

Mizoguchi, R., 2019. Knowledge engineering. Ontol. Makes Sense 69–81.

Mizoguchi, R., Ikeda, M., 1998. Towards ontology engineering. J. Jpn. Soc. Artif. Intell. 13, 9–10.

Mohemad, R., Mamat, N.F.A., Noor, N.M.M., Alhadi, A.C., 2019. The development of an ontology model for early identification of children with specific learning disabilities. Int. J. Electr. Comput. Eng. (IJECE) 9 (6), 5486–5494.

Mora, M., Wang, F., Gómez, J.M., Phillips-Wren, G., 2022. Development methodologies for ontology-based knowledge management systems: a review. Expert Syst. 39, e12851.

Musen, M.A., 2015. The protégé project: a look back and a look forward. AI Matters 1, 4–12.
Neuhaus, F., Ray, S., Sriram, R.D., 2014. Toward ontology evaluation across the life cycle. US Department of Commerce, National Institute of Standards and Technology.
Novak, J.D., 2013. Concept mapping. International Guide to Student Achievement. Routledge,, pp. 362–365.
Novak, J.D., Gowin, D.B., 1984. Learning How to Learn. Cambridge University press,.
O'Leary, D.E., 2017. Big Data and knowledge management with applications in accounting and auditing: The case of Watson. The Routledge Companion to Accounting Information Systems. Routledge, pp. 145–160.
Osman, M.A., Noah, S.A., Saad, S., 2022. Ontology-based knowledge management tools for knowledge sharing in organization – a review. IEEE Access 10, 43267–43283.
Pérez, J., Arenas, M., Gutierrez, C., 2009. Semantics and complexity of SPARQL. ACM Trans. Database Syst. (TODS) 34, 1–45.
Peroni, S., 2016. A simplified agile methodology for ontology development. OWL: Experiences and Directions–Reasoner Evaluation. Springer,, pp. 55–69.
Pinto, H.S., Staab, S., Tempich, C., 2004. DILIGENT: Towards a fine-grained methodology for DIstributed, Loosely-controlled and evolving engineering of ontologies. ECAI 16, 393.
Pinto, H.Sofia, Christoph, Tempich, and Steffen Staab. "Ontology engineering and evolution in a distributed world using DILIGENT." Handbook on ontologies2009: pp. 153–176.
Poveda Villalón, M., Suárez-Figueroa, M.C., & Gómez-Pérez, A, 2010. Reusing ontology design patterns in a context ontology network. In: WOP'10: Proceedings of the 2nd International Conference on Ontology Patterns. ACM, pp 35–52.
Poveda-Villalón, M., 2012. A reuse-based lightweight method for developing linked data ontologies and vocabularies. Extended Semantic Web Conference. Springer,, Berlin, Heidelberg, pp. 833–837.
Poveda-Villalón, M., Fernández-Izquierdo, A., Fernández-López, M., García-Castro, R., 2022. LOT: an industrial oriented ontology engineering framework. Eng. Appl. Artif. Intell. 111, 104755.
Rebstock, M., Fengel, J., Paulheim, H., 2008. Ontology engineering. Ontology-Based Business integration. Springer Berlin Heidelberg,, pp. 97–123.
Ren Y., Parvizi A., Mellish C., et al., 2014. Towards competency question-driven ontology authoring. In: Proceedings of the European Semantic Web Conference. Springer, pp 752–767.
Ruy, F.B., Guizzardi, G., Falbo, R.A., Reginato, C.C., Santos, V.A., 2017. From reference ontologies to ontology patterns and back. Data Knowl. Eng. 109, 41–69.
Ruy F.B., Reginato C.C., Santos V.A., Falbo R.A., Guizzardi G., 2015. Ontology engineering by combining ontology patterns. In: Proceedings of the Conceptual Modeling: 34th International Conference, ER 2015, Stockholm, Sweden, October 19–22, 2015, Proceedings 34 (pp. 173–186). Springer International Publishing.
Sandkuhl, K., Shilov, N., Smirnov, A., 2019. Facilitating digital transformation by multi-aspect ontologies: approach and application steps. IFAC-Pap. 52, 1609–1614.
Sanya, I., Shehab, E., 2015. A framework for developing engineering design ontologies within the aerospace industry. Int. J. Prod. Res. 53, 2383–2409.
Scherp, A., Saathoff, C., Franz, T., Staab, S., 2011. Designing core ontologies. Appl. Ontol. 6 (3), 177–221.
Shadbolt, N.R., Smart, P.R., Wilson, J., Sharples, S., 2015. Knowledge elicitation. Eval. Hum. Work 163–200.
Shadbolt N.R., Burton M., 1995. Knowledge elicitation: a systematic approach. In valuation of Human Work: A Practical Ergonmoics Methodology. pp. 406–440.
Simperl, E., Tempich, C., 2009. Exploring the economical aspects of ontology engineering. Handbook on Ontologies. Springer, Berlin, Heidelberg, pp. 337–358.
Simperl, E.P.B., Tempich, C., 2006. Ontology engineering: a reality check. OTM Confederated International Conferences" On the Move to Meaningful Internet Systems. Springer,, pp. 836–854.
Sirin, E., Parsia, B., Grau, B.C., et al., 2007. Pellet: a practical owl-dl reasoner. J. Web Semant. 5, 51–53.
Spoladore, D., Pessot, E., 2021. Collaborative ontology engineering methodologies for the development of decision support systems: case studies in the healthcare domain. Electronics 10 (9), 1060.
Spoladore, D., Pessot, E., 2022. An evaluation of agile ontology engineering methodologies for the digital transformation of companies. Comput. Ind. 140, 103690.
Spoladore, D., Sacco, M., 2018. Semantic and dweller-based decision support system for the reconfiguration of domestic environments: RecAAL. Electronics 7, 179.
Spoladore D., Stella F., Tosi M., Lorenzini E.C., 2023. Towards a knowledge-based Decision Support System for the management of Type 2 diabetic patients. In: Proceedings of the Towards a Smart, Resilient and Sustainable Industry. ISIEA 2023 Conference Proceedings. [IN PRESS].
Spyns, P., Tang, Y., Meersman, R., 2008. An ontology engineering methodology for DOGMA. Appl. Ontol. 3 (1–2), 13–39.
Staab, S., Studer, R., Schnurr, H.P., Sure, Y., 2001. Knowledge processes and ontologies. IEEE Intell. Syst. 16 (1), 26–34.
Suárez-Figueroa, M.C., Gómez-Pérez, A., 2012. Ontology requirements specification. Ontology Engineering in a Networked World. Springer, pp. 93–106.
Suárez-Figueroa, M.C., Gómez-Pérez, A., Fernandez-Lopez, M., 2015. The neon methodology framework: a scenario-based methodology for ontology development. Appl. Ontol. 10 (2), 107–145.
Sure, Y., Staab, S., Studer, R., 2004. On-to-knowledge methodology (OTKM). Handbook on Ontologies. Springer,, Berlin, Heidelberg, pp. 117–132.
Tempich, C., Pinto, H.S., Sure, Y., et al., 2005. Evaluating DILIGENT ontology engineering in a legal case study. Proceedings of the IVR 22nd World Congress–Law and Justice in a Global Society. International Association for Philosophy of Law and Social Philosophy,.
Tudorache, T., 2020. Ontology engineering: Current state, challenges, and future directions. Semantic Web 11 (1), 125–138.
Vigo, M., Bail, S., Jay, C., Stevens, R., 2014a. Overcoming the pitfalls of ontology authoring: Strategies and implications for tool design. In: International Journal of Human-Computer Studies, 72, pp. 835–845.
Vigo M., Jay C., Stevens R., 2014b. Design insights for the next wave ontology authoring tools. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp 1555–1558.
Xiang Z., He Y., 2009. OntoFox. Nature Precedings, 1–1.
Yang, L., Cormican, K., Yu, M., 2019. Ontology-based systems engineering: a state-of-the-art review. Comput. Ind. 111, 148–171.