A Review and Comparison of Competency Question Engineering Approaches

Reham Alharbi[1,2][0000−0002−8332−3803], Valentina Tamma[1][0000−0002−1320−610X], Floriana Grasso[1][0000−0001−8419−6554], and Terry R. Payne[1][0000−0002−0106−8731]

[1] University of Liverpool, Liverpool, UK
{R.Alharbi, V.Tamma, Floriana, T.R.Payne}@liverpool.ac.uk
[2] Taibah University, Madinah, KSA
rfalharbi@taibahu.edu.sa

Abstract. Competency Questions (CQs) are essential in ontology engineering; they express an ontology's functional requirements through natural language questions, offer crucial insights into an ontology's scope, and are pivotal for various tasks, such as ontology reuse, testing, requirement specification, and pattern definition. CQ engineering approaches are gaining prominence, transitioning from manual to automatic methods, each with its own purpose, techniques, outcomes, and limitations. In this paper, we provide a review aimed at positioning these approaches within a formal categorisation, highlighting the main challenges among them and facilitating the inclusion of upcoming initiatives, and propose a benchmark to support comparative studies.

Keywords: CQs authoring · CQs generating · Ontology development phases.

1 Introduction

Competency Questions (CQs) [30] are natural language questions that characterise the scope of the knowledge represented by an ontology. They model the functional requirements that an ontology-based information system should satisfy to achieve its intended purpose. Within the early stages of ontology development, they are used to suggest possible concepts and relationships the ontology should model [42,43,44,49,51], can be used in subsequent phases to verify and validate the knowledge encapsulated in the ontology [13,31], recommend ontologies for reuse [2,4,11], and facilitate the consumption of ontology content, such as generating APIs [22]. Their significance has been further highlighted in a small number of recent surveys that have targeted ontology developers to identify their practical usage [3,40].

In addressing problems related to CQs formulation in the ontology development cycle, previous studies have focused on artefacts and processes that can enhance the quality of CQs after they have been manually authored. This approach has taken several forms, such as the use of *Controlled Natural Language (CNL)*, which has been used for authoring ontologies [20,41], tests from

CQs [15,21], and the CQs themselves [8,13,32,48,54]. However, ontology developers must first manually develop the CQs and then verify their compliance with respect to the CNL templates/archetypes, therefore they have little effect on the initial authoring effort required. Indeed, the manual process of authoring and verifying CQs is both tedious and time-consuming. It demands a significant level of intellectual effort and attention to detail, which can be a substantial burden on developers. The intricate nature of this task not only extends the development timeline but also increases the potential for errors, which might necessitate further revisions and checks. The complexity and the challenges associated with this process highlight the need for more streamlined approaches or tools that can either automate this process or assist developers in creating and validating these CQs more efficiently.

In this paper, we conduct a scoping review [10,39] over a small, curated set of studies to comprehend how CQs are formulated or constructed in the ontology development process. Scoping reviews are emerging as a useful tool for identifying and analysing knowledge gaps in a body of work [10]; therefore by conducting this review we map the salient issues that have been identified in the curated set of papers, which can then be used to conduct a comprehensive systematic review [34]. Thus, we address the following research questions:

RQ1: What are the main dimension(s) that characterise approaches for CQ formulation?
RQ2: What is the level of automation employed by different methodologies in constructing CQs?
RQ3: What are the methods and materials used for validating approaches for constructing (semi)-automatically CQs?

Therefore, our work primarily focuses on categorising CQ formulation approaches, identifying the resources used for each approach, and the validation criteria used to assess the relevance of the CQs with respect to the ontology whose construction they support. The contribution of this paper is twofold: (i) a categorisation of the current state of CQ formulation / construction approaches, together with their purpose, validation criteria, outcomes, and limitations, (ii) a proposal for a CQ benchmark along with its task specifications and evaluation criteria, that emerges as a key requirement from the scoping review as a gap in the literature. In this paper, we provide an overview of the methods underlying CQ construction approaches in Section 2, including recent ones where CQs are generated exploiting Generative AI, followed by the methodology used to conduct the scoping review (Section 3). We then discuss the research questions and how they fit the literature within the review (Section 4). A proposal for benchmarking is then presented based on the initial findings of the review in Section 5, prior to giving our conclusions in Section 6.

2 CQ generation guideline and LLM based approaches

CQs are integral to the specification of requirements in many ontology development methodologies [42,43,44,49,51] and to the validation of ontological artifacts,

as they facilitate the evaluation of the ontological commitments that have been made and are generally accepted as a standard verification technique for ontologies [14,21,31,33]. However, the guidelines for creating these CQs are only vaguely specified.

For example, Gruninger and Fox [30], state that CQs should specify the requirements for an ontology and thus serve as a mechanism for characterising the ontology scope. These CQs act as constraints on what the ontology can be, rather than determining a particular design with its corresponding ontological commitments. They can also be used to assess whether the ontology meets the specified requirements based on the ontological commitments identified. The Ontology 101 [42] development process proposes a set of questions to aid ontology developers in creating their CQs. These include: *"Does the ontology contain enough information to answer these types of questions? Does the answer require a particular level of detail or representation of a particular area?"*. These CQs are intended as a sketch for further development and do not need to be exhaustive.

Rao et al. [45] propose knowledge elicitation techniques to guide ontology developers in creating CQs. For example, the 20-questions technique involves domain experts generating questions they consider important in their domain, which the ontology should be able to answer. Additionally, through card sorting, ontology engineers can identify criteria important to domain experts for grouping similar cases, thus forming the basis of the CQs. These questions are expected to ensure that the ontology can respond to queries about these concepts.

With the advent of Large Language Models (LLMs) and Generative AI, a new shift emerged in 2023 with the possible automation of knowledge engineering activities, in particular the formulation of CQs [6,9,17,46]. A number of approaches have emerged that exploit LLMs to (partially) automate the formulation process, and that differ with respect to the nature of the knowledge resources used in the prompts. In particular, we classify approaches into:

1. **Reverse engineering of CQs:** this is a reversal of the traditional workflow, where a knowledge source that was previously built from a set of ontologies, themselves engineered from CQs [17] is used to generate new CQs for a different purpose.
2. **Retrofitting CQs:** in this context, an ontology exists, but no associated CQs are published as part of its documentation. In this case the aim is to identify possible CQs that were used in the development of the ontology, thus enabling its reuse for future uses [6].
3. **Generating CQs:** these are studies that generate CQs, either from a set of class and property names [46], or generate CQs from a corpus of text describing a domain [9].

3 Research method

This review's protocol is informed by the methodological framework provided in [10], and is illustrated in Figure 1. Having identified the initial *Research Questions* in Section 1, the next stage is that of *Identifying Relevant Studies*. This

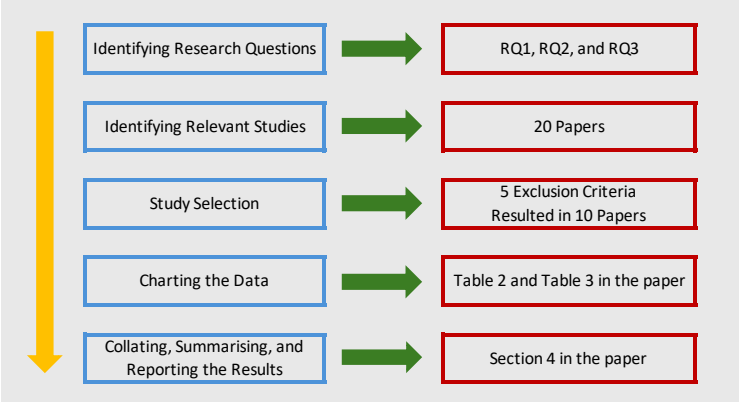

Fig. 1. Methodological framework stages of the *Scoping Study*, summarised from [10].

involved conducting a manual search process using the following archives: IEEE[3], ACM Digital Library,[4] ScienceDirect,[5] Springer,[6] and Elsevier.[7] Additional research papers were included in the search by examining the 'related work' and 'reference list' sections of each of the papers indentified. We also included general and academic search engines such as Google Search and Google Scholar to identify other relevant papers. Furthermore, we considered the citations to certain papers by using the 'cited by' option in Google Scholar to include papers not identified by the previous methods.

The field of CQ engineering is broad and interdisciplinary, encompassing areas such as computer science, educational assessment, and human resources. Therefore, for the purposes of the scoping review, the search was focused on the use of *CQs in ontology engineering*. As a result, studies such as [50] were excluded as they focus on CQs in the educational field. Therefore, different combinations of search terms were formulated using keywords deemed pertinent to the study, resulting in the identification of 20 papers:

CQs (generation OR retrofitting OR revering OR authoring) AND (Knowledge graph OR Ontology), CQs (templates OR archetypes OR patterns).

For the third stage (Figure 1), *Study Selection*, a screening process was employed to identify and eliminate studies that did not address our research questions. This in turn comprised three stages: 1) title and abstract screening, 2) full-text screening, and 3) filtering based on inclusion and exclusion criteria. As a result, 10 papers were included in the final analysis (Table 1). Papers were excluded if they violated one or more of the following criteria:

[3] `https://ieeexplore.ieee.org/Xplore/home.jsp`
[4] `https://dl.acm.org/`
[5] `https://www.sciencedirect.com/`
[6] `https://www.springer.com/`
[7] `https://www.elsevier.com/`

Table 1. Studies included in the Scoping Review.

[24]	**2011**	*Using Goal Modelling to Capture Competency Questions in Ontology-based Systems*
[48]	**2014**	*Towards Competency Question-Driven Ontology Authoring*
[15]	**2014**	*CQChecker: A tool to check ontologies in OWL-DL using competency questions written in controlled natural language*
[54]	**2019**	*Analysis of Ontology Competency Questions and their formalization in SPARQL-OWL*
[32]	**2019**	*CLaRO: A Controlled Language for Authoring Competency Questions*
[8]	**2021**	*Assessing and Enhancing Bottom-up CNL Design for Competency Questions for Ontologies*
[9]	**2023**	*Automating the Generation of Competency Questions for Ontologies with AgOCQs*
[6]	**2024**	*An Experiment in Retrofitting Competency Questions for Existing Ontologies*[8]
[17]	**2024**	*RevOnt: Reverse engineering of competency questions from knowledge graphs via language models*
[46]	**2024**	*Can LLMs Generate Competency Questions*

1. The paper is not written in English.
2. The paper is irrelevant to the Semantic Web domain.
3. The paper primarily focuses on building an ontology or a part of an ontology from CQs, emphasising the evaluation of development methods rather than the CQs themselves.
4. The paper primarily focuses on authoring test-form CQs, where the main interest lies in evaluating the testing method rather than the CQs themselves.
5. The paper primarily focuses on the formalisation of CQs into SPARQL or description logics queries, emphasising the formalisation methods rather than the CQs themselves.

A specific form was designed for the fourth stage of the reviewing framework, *Charting the Data,* for the data extraction process, given the research questions targeted in this scoping review. The form includes the following characteristics: (i) title and year of publication, (ii) automation level (iii) knowledge resource, (iv) evaluation measure, (v) ground truth, (vi) outcomes. This data is presented across two tables, with Table 2 appearing in the discussion on RQ2, and Table 3 appearing in the discussion on RQ3. The final stage, *Collating, Summarizing, and Reporting the Results* is presented in Section 4.

4 Results and Discussion

4.1 RQ1: Categorising CQs Formulation / Construction Approaches

In this section, we address the first research question: *What are the main dimension(s) that characterise approaches for CQ formulation?* This question is necessary to set the scene for the other two research questions that delve in automation support and validation used within the categories identified here.

There are several dimensions or characteristics that on first inspection appear orthogonal (such as automation and validation), but that on closer inspection are inter-dependent (Table 2). One of the most significant characteristics

[8] This paper was originally released as a pre-print in 2023 [5].

is regarding the level of automation involved in the approaches. It is possible to consider approaches as purely *manual* (i.e. with no structured support), beyond that of following best practice when authoring CQs [30,42], or *semi-automated*, involving structured or established resources such as patterns [48,54], templates [8,15,32], and archetypes [15]. These can be also categorised as *filler-based questions*, as they aim to support developers in manually deriving CQs from specific ontologies [15], or support the manual formulation of CQs for ontology development [8,32,48,54]. This contrasts with the automatic construction (or generation) of CQs using a defined resource such as an ontology or Knowledge Graph (KG), and text corpora [9,6,17,46] as input to a generative LLM system. The ultimate goal of such approaches is to produce CQs that cover a specific domain and can be integrated into various ontology development phases, depending on the knowledge source exploited in each method.

It is noteworthy that research in CQ engineering initiated with authoring approaches, which spanned from 2011 to 2021—the most recent study available at the time of this review. Conversely, generating approaches emerged in 2023 and have swiftly evolved, encompassing four distinct methods as of June 2024.

4.2 RQ2: Constructing CQs

In this subsection, we address the question: *What is the level of automation employed by different methodologies in constructing CQs?* The traditional approach whereby CQs are authored manually, following established guidance [30,42] is naturally a barrier to effective participation in ontology development, as it heavily relies on the collaboration of the domain experts, and it gives rise to inconsistencies and *ad hoc* solutions. In response to this challenge, a variety of resources and approaches emerged that were designed to facilitate the semi-automated authoring of CQs, based on an associated *Controlled Natural Language (CNL)* [36], which typically works as a set of *'filler-based questions'*. CNLs can be used to either evaluate the CQs that were manually derived from specific ontologies [15] or support the manual authoring of CQs [8,32,48,54]. These approaches however lack a formal evaluation with respect to their accuracy in the context of CQ authoring and use, evaluation which has been attempted in other contexts [28,29].

A complete automation of the generation of CQs for a prospective ontology, in contrast, is a non-trivial task. Approaches are beginning to emerge that exploit the use of Large Language Models (LLMs) and Generative AI models to propose viable CQs based on the careful crafting of prompts that include resources such as selected triples [6,46] or Knowledge Graph fragments [17]. Each of the studies is discussed below, grouped by the level of automation involved, based on the characterisation in Table 2.

Manual Approaches. In 2011, Fernandes et al. [24] proposed applying the Tropos methodology [16], a development approach for Multi-Agent systems, to enable the definition of CQs through goal modelling. Their methodology begins with an early requirements activity to analyse organisational goals, followed by a

Table 2. Levels of Automation and resources required for CQ construction.

Automation	Study	Approach	Resource
Manual	Fernandes et al. [24]	Methodology to define CQs	N/A
Semi-Automatic	Ren et al. [48]	Pattern of CQs	Pizza ontology CQs
	Bezerra et al. [15]	Template of CQs	Pizza ontology CQs & Software ontology CQs
	Wiśniewski et al. [54]	Pattern of CQs	234 CQs for 5 ontologies
	Keet et al. [32]	Template of CQs	Wiśniewski et al. [54] patterns
	Antia et al. [8]	Template of CQs	CLaRO templates [32] & new dataset of 92 CQs
Automatic	Antia and Keet [9]	Corpus-based method for generating CQs	Corpus of Covid-19 research articles
	Alharbi et al. [6]	LLM + Ontology	CORAL [25] & CQs dataset [54]
	Ciroku et al. [17]	LLM + KG	(WDV dataset)
	Rebboud et al. [46]	LLM + Ontology	5 Ontologies & SBERT

late requirements phase where CQs are captured and linked to these goals. This approach aims to provide a consistent process for ontology engineers to develop ontologies from scratch, addressing the gap between the definition of CQs and the ontology modelling process.

Semi-Automatic Approaches. Ren et al. [48] analysed the structure of CQs and defined a set of 19 CQ *archetypes*, syntactic patterns of CQs that would be instantiated by the ontology developer; e.g., "`Which [CE1] [OPE] [CE2]?`", where `CE1` and `CE2` are class expressions (or individuals, in certain cases), and `OPE` corresponds to an object property expression. However, out of these 19 patterns, 14 were merged with types of ontology elements, specifically OWL classes and object properties, resulting in a 1:1 mapping attribute [32]. This restricts their use to only OWL ontologies with certain limited formalisation patterns. For example, a simple subclass request such as "*What are the types of diagnosis?*" from DemCare_CQ_8 [18] has no applicable pattern [32]. Although these patterns assist ontology engineers in formulating machine-processable CQs for ontology testing, their adequacy or coverage has still to be investigated.

The use of patterns was also used by Wiśniewski et al [54], who proposed a total of 106 patterns, identified by analysing 234 CQs for five ontologies using various natural language processing tasks. This was primarily achieved through a pattern detection process to distinguish between entity and predicate chunks. An *Entity Chunk (EC)* refers to a noun or noun phrase that describes an object (entity) represented in the ontology as either a class or an individual, whereas a *Predicate Chunk (PC)* refers to a verb or phrasal verb that represents the relationship between entities in the ontology. These patterns vary in sentence structure to accommodate different question formulation preferences (e.g., "`Who`"

and "Where" question types, omitted in [13]), and thus provide better coverage for CQs than previous studies [15,48]. Furthermore, the authors deemed that these patterns could be used to specify requirements for an ontology. These patterns were analysed and utilised to design a template-based CNL. Given the limited number of CQ patterns, a template-based approach was adopted for the CNL at this stage, instead of specifying a grammar.

Bezerra et al. [15] proposed 14 patterns to function as a CNL through templates for CQs. For example, "Does <class> + <property> <class>?", which could be filled with vocabulary taken from the ontology. The authors considered these patterns as support for the *Ontology Requirements Specification* phase; i.e., for creating and processing CQs written in natural language. However, the patterns emerging from this study were considered to have limited coverage as they are based specifically on CQ sets taken from the Pizza ontology [19]. Thus, there was the risk that this introduced both *domain bias* and *CQ author bias* (as the Pizza CQs were created *after* the ontology had been developed). It also risked exhibiting *prejudiced* patterns [54]; an example of these is were the CQ "Which pizza has "hot" as spiciness?" is created as it fits with the Pizza ontology's aa hasSpiciness" data property. However, a better CQ would have been to use the more linguistically natural phrase "Which pizzas are hot?" that is fully agnostic with respect to how it is represented in the ontology (whether or not it is with a data property, object property, or a class) [54].

CLaRO [32] is a CNL that is based on templates for use in authoring CQs. Keet et al. transformed the 106 patterns identified by Wiśniewski[54] into 93 main templates (plus 41 variants) using CNL. The additional CNL features specifically addressed issues such as: (i) singular/plural forms; (ii) the use of personal pronouns in patterns; (iii) removal of redundant words in text chunks; and (iv) synonym usage. CLaRO's 134 templates were evaluated, demonstrating coverage of about 90% of the test sets in [54]. It also has the potential to fulfil the objectives outlined in Wiśniewski et al. [54] to streamline the formalisation of ontology content requirements.

CLaRO was subsequently expanded by Antia et al. [8], incorporating an additional 120 main templates (with an additional 12 variants). This new dataset of 92 CQs generated 27 new templates and 7 more variants, significantly increasing the domain coverage and enhancing the effectiveness of the CNL. The resulting CLaRO v2 has since evolved and now includes a total of 147 templates and 59 variants, achieving 94.1% coverage. However, to effectively use CQs, ontology developers must initially create them manually and subsequently verify their compliance using these templates. The consistent dependence on a manual process highlights a prevalent challenge in the solutions developed so far [9].

Automatic Approaches. One of the first automated approaches originated in 2023, with the proposal of AgOCQs, a corpus-based method for generating CQs, that uses a domain text corpus as a knowledge resource to extract text, which is then pre-processed using NLP techniques and fed into a pre-trained language model to generate questions [9]. These questions then undergo filtering to remove

duplicates and meaningless questions through semantic grouping. Although the inclusion of a corpus as a source of knowledge had been prevalent in other domains such as expert systems and data mining [37,53], what made this novel was its application in CQ generation.

The possibility of exploiting generative AI has resulted in new ways to automate the construction of CQs, by formulating different prompts that exploit the source domain content as a knowledge resource. One of the first examples of this originally appeared in 2003 [5] with the proposal of `RETROFIT-CQs`, which *retrofits* CQs from existing ontologies by utilising those ontologies as a knowledge resource to extract triples in the form of `('subject', 'predicate', 'object')` [6]. These triples then provide a contextual boundary, which is fed into a variety of different LLMs via specifically designed prompts to generate CQs for each triple. The generated questions are filtered to remove duplicates and semantically paraphrased questions. An advantage of this approach was that it specifically addressed scenarios were there was a lack of CQs for existing ontologies, that could hamper their use during different ontology development phases such as for *Ontology Reuse* [2,11,4] and *Ontology Exploitation* [22].

The use of LLMs was also explored by Rebboud et al. [46] and Ciroku et al. [17]. Both studies proposed methods for generating CQs but with different objectives. Although the approach proposed by Rebboud et al. shared some similarities with that of Alharbi et al. [6] in that ontologies were used as a knowledge resource for LLM prompts, their processing mechanisms were notably different. Specifically, Rebboud et al. [46] parsed an ontology to extract classes, properties, and schema, represented as triples: `('Classes', 'Properties', 'Classes')`. These elements were then divided into batches of 20 classes, with each iteration processing all of these classes and the related properties that connect them to other classes corresponding to their domain or range. However, it is unclear if the splitting mechanism used is manual or automatic, how the process ensures that all related classes appear in the same iteration, or how repetitions are handled. Although this approach addressed the problem resulting from a lack of CQs for existing ontologies [46] , the ultimate goal was to assess the capabilities of LLMs in several tasks, such as using CQs as inputs to generate parts of an ontology. This is similar to the contributions of studies such as NeOn-GPT [23], and others [27,26], which are outside the scope of this review.

RevOnt [17] was proposed by Ciroku et al. as a method for extracting CQs from knowledge graphs (KGs) using pre-trained language models. The primary aim of this approach was to assist ontology developers in CQs elicitation, therefore, targeting the *Requirement Specification* phase in ontology development methodologies, in a similar way to other approaches such as AgoCQs [9]. The process began with the verbalisation of data from the KG, specifically using the WDV dataset[9] for Wikidata entries, followed by the abstraction of these verbalisations into triples; i.e. `('subject', 'predicate', 'object')`. For each triple, this abstraction is then fed into a pre-trained language model as context, accompanied by three facts: the class of the subject, the property, and the class

[9] `https://github.com/gabrielmaia7/WDV`

of the object. The resulting CQs are grammatically vetted and subjected to a filtration process for similarity and paraphrase detection.

Although various efforts can facilitate the construction of CQs, investigating different methods for validating the outputs remains a significant challenge in comparing methods and choosing which to elaborate in practice.

4.3 RQ3: Validating CQ Construction Approaches

Having previously explored the approaches taken in generating CQs, and in particular, the level of automation involved in such approaches, we now address the question: *What are the methods and materials used for validating approaches for constructing (semi)-automatically CQs.* The way in which validation is conducted very much depends on the automation approach taken; validation is not typically conducted when using most of the manual and semi-automated methods beyond that which is human-based (i.e. primarily verifying that the CQs appear valid with knowledge of the domain, but without any specified resource used to determine the *ground truth*). Automated techniques, on the other hand, may exploit evaluation measures with a dataset used to provide a ground truth against which accuracy or precision/recall etc can be determined. These factors contribute to the difficulty in generating systematic comparisons across different approaches. In particular, the automated approaches employ various knowledge resources and are evaluated against diverse benchmarks. Moreover, the evaluation measures themselves inherently differ among these methods. Furthermore, these methods target different phases of ontology development, where the generated CQs can be further elaborated. These methods are discussed in much more depth below, together with the other characterisation listed in Table 3.

Manual and Semi-Automatic Authoring Methods. Few of different ways of analysing CQ and validating manual and semi-automatic authoring methods have been considered. Approaches that are based on patterns of CQs [48,54] analyse the CQs to propose the patterns but perform no specific validation. Template based approaches [15,32] result in a reduction of patterns (as in some cases, more than one pattern can emerge in one template, and sometimes there may be one pattern that can generate more than one template), but again, no specific validation is typically used. However, CLaRO [32] was validated against subset of the CQs used to derive the patterns, which were then transformed into the templates that were used to build it. A similar approach was adopted when CLaRO was extended to CLaRO2 [8] through the addition of more CQs, which resulted in more patterns and in turn more templates.

The extent to which there has been an uptake of these *filler-based questions* (i.e. patterns, templates, and archetypes, etc) for authoring CQs across the Ontology Engineering community is not clear. An investigation into this could identify the practical barriers that hinder the applicability of these efforts to different contexts; however, a challenge remains, as to how these methods could be compared within a common framework.

Table 3. Evaluation Measures and Resources required for Validation.

	Study	Evaluation Measure	Ground Truth	Outcomes
Manual & Semi-Automated	Fernandes et al. [24]	Case Study	N/A	Method
	Ren et al. [48]	Human-based evaluation	N/A	Patterns
	Bezerra et al. [15]	Human-based evaluation	N/A	CNL-templates
	Wiśniewski et al [54]	N/A	N/A	Patterns
	Keet et al. [32]	Human-based evaluation	N/A	CNL (CLaRO) templates
	Antia et al. [8]	Human-based evaluation	N/A	CNL (CLaRO v2) templates
Automated	Antia and Keet [9]	Human-based evaluation	CLaRO templates	CQs & their templates
	Alharbi et al. [6]	SBERT & Human-based evaluation	Existing CQ	CQs
	Ciroku et al. [17]	BLEU & Human-based evaluation	Annotated benchmark	CQs CQs
	Rebboud et al. [46]	SBERT	Existing CQ	CQs

Automatic Authoring Methods. The automatic CQ generation methods differ from the manual or semi-automatic ones in that they utilise additional resources in the generation method itself, and such resources can also form the basis of *ground truth* and thus be used for validation. For example, Antia and Keet [9] validated their CQs by matching them with the CLaRO templates[32]. Specifically, if the abstract form of a question matched a template, it was classified as a CQ. `AgOCQs` was applied to Covid-19 research articles, and the generated CQs underwent a human-based evaluation through a survey targeting specific user groups and individuals. In this evaluation, 73% of the user group and 69% of the ontology experts judged all the CQs to provide clear domain coverage. According to the authors in [9], `AgOCQs` were designed to assist ontology developers in scoping the ontology and identifying the domain, which suggests that they target the *Requirement Specification* phase of ontology development.

The use of LLMs poses its own challenges, as well as opportunities. The validity of the CQs generated by `RETROFIT-CQs` [7] was evaluated through two approaches: (i) a human-based evaluation, and (ii) using a comparative evaluation with ground truth resources. For the human-based evaluation, an ontology developer assessed the correctness and quality of the CQs for their ontology, with `RETROFIT-CQs` achieving a precision over 0.75 based on developer evaluation. Furthermore, the developer noted that the generated CQs not only reflected the model's ontology representation but also expose unintended modelling outcomes, which could have been included in the initial requirements elicitation phase. The comparative evaluation compared the generated CQs against the

ground truth resources, which in this case constituted CQs for three ontologies selected from existing CQ datasets [54,25]. SBERT [47] was used to measure semantic similarity, with `RETROFIT-CQs` achieving a recall of 0.99. Further investigations explored the affect of changing the LLM parameters (such as varying the creativity parameter to assess the impact on the resulting CQs).

The notion of a comparative assessment was also used in other approaches [46,17], where ontologies were used as part of the ground truth. In one approach [46], semantic similarity between the generated CQs and the ontologies themselves was measured using SBERT [47]. Although in these studies, low precision scores were reported, it was noted that this should not be interpreted as the result of irrelevant questions being generated, but rather that new CQs were discovered that could make a valuable addition to the ground truth dataset. For the second approach [17], the ground truth consisted of manually curated CQs for the WDV dataset[10] (for a detailed discussion on verbalisation evaluation, interested readers can refer to [17]). The results were variable, depending on the type of CQ generated (for example, higher quality scores were attributed to questions where the answer was the object of the triple).

5 `CQsBEN` - A benchmark for CQs Formulation Approaches

Validating manual (or semi-automatic) approaches has always been challenging due to the subjective nature of ontology engineering, and the fact that CQs typically need to address a requirement that is difficult to quantify. However, as more automatic, generative methods emerge, there is a growing desire to perform comparative evaluations between them. To date, few common resources have been used by different studies (one exception being Dem@Care [18], which has been used by several studies [6,46]), and there is little consistency on the use of evaluation measures to assess the CQs. Furthermore, different studies have addressed different phases in the ontology development lifecycle, and thus cannot be directly compared. For example, some approaches target the *Requirement specification* phase [9,17], whereas others address the context where ontologies have missing or non-existent CQs [6,46].

Developing a multi-purpose benchmark is fundamentally different from creating a specific benchmark that could be used for a single purpose (for example, LOVBench [35], which is used ontology term ranking). It should serve all types of formulation / construction approach used (including those that are manual, semi-automated, and fully automated), and thus include criteria that assess them, starting from specifying the tasks to providing evaluation criteria. In the discussion below, we have identified three main tasks for such a benchmark, which we refer to as `CQsBEN`: (i) **Poor/Incorrect Requirements:** This category addresses common errors in question formulation that hinder effective query processing and data retrieval; (ii) **Scoping CQs:** Such questions may help to define the domain, but are not used for querying. These require specialised

[10] `https://zenodo.org/records/10370725`

handling to aid the definition of a domain; (iii) **Verified CQs:** These CQs can be directly queried and can serve as benchmarks for system capabilities. Each of these main tasks includes subtasks, where Table 4 indicates their relevance to either one or both of the authoring or generating CQs approaches. The subtasks for each main task, along with their description, are as follows:

1. **Poor/Incorrect Requirements:**
   (a) Linguistic Perspectives:
      i. *Identify Ambiguous Questions:* Create a repository of CQ examples that exhibit ambiguity in wording or context.
      ii. *Develop Clarity Guidelines:* Formulate standards / templates to help rephrase ambiguous questions for improved clarity and specificity.
   (b) Question Types:
      i. *Classify Question Types:* Systematically categorise CQs into types such as narrative, factual, or descriptive, and assess their suitability in different contexts.
      ii. *Evaluate Contextual Appropriateness:* Develop criteria to measure the effectiveness of question types within their intended contexts.
   (c) Domain Knowledge:
      i. *Align Questions with Domain Relevance:* Establish a review process to ensure questions are pertinent wrt the relevant domain knowledge.
      ii. *Refine Focus Through Filtering:* Implement a mechanism to exclude questions that, while correct, are irrelevant to the task at hand.
   (d) Incorrectness:
      i. *Fact-Check Information:* Set up a robust protocol for verifying the factual accuracy of CQs.
      ii. *Correct Erroneous Inputs:* Introduce a correction mechanism for adjusting factually incorrect CQs.
2. **Scoping CQs:**
   (a) *Catalogue Scoping CQs:* Document all CQs that contribute to defining the scope of the information domain.
   (b) *Analyse for Domain Contribution:* Analyse how these CQs help in shaping the understanding of the domain.
   (c) *Integrate into Information Architecture:* Develop strategies to utilise scoping CQs for enhancing the structure of information repositories.
3. **Verified CQs:**
   (a) *Database of Verified CQs:* Maintain an updated list of CQs that can be directly transformed into SPARQL queries.
   (b) *Transformation into Queries:* Convert verified CQs into effective SPARQL queries.
   (c) *Testing and Validation:* Conduct rigorous testing to ensure the queries retrieve accurate and relevant data.
   (d) *Documentation and Examples:* Create detailed documentation and examples of successful CQ transformations for training and reference.

Table 4. Summary of Tasks, Subtasks, CQs Engineering Approaches, and Evaluation Measures.

Task	**Subtask**	**Automation Approach**	**Evaluation Measure**
Poor/Incorrect Requirements	Linguistic Perspectives	*Manual/Semi-Automatic*	Subjective
	Question Types	*Automatic*	Subjective
	Domain Knowledge	*Automatic*	Subjective
	Incorrectness	*Automatic*	Subjective
Scoping CQs	Catalogue Scoping CQs	*Automatic*	Similarity Matching
	Analyse for Domain Contribution	*Automatic*	Subjective
	Integrate into Information Architecture	*Automatic*	Subjective
Verified CQs	Database of Verified CQs	*Manual/Semi-Automatic*	Subjective
		Automatic	Similarity Matching
	Transformation into Queries	*Manual/Semi-Automatic*	Subjective
		Automatic	Verified SPARQL
	Testing and Validation	*Manual/Semi-Automatic*	Verified SPARQL
		Automatic	Verified SPARQL
	Documentation and Examples	*Manual/Semi-Automatic*	Subjective
		Automatic	Subjective

`CQsBEN` aims to refine the process of handling CQs, ensuring that they adhere to practices for developing high-quality CQs, which are clear, relevant, and effectively transformable into queries. By addressing each category with specific subtasks, we can significantly improve the accuracy and efficiency of CQs engineering approaches.

Each task and subtask has its own evaluation measure to assess an approach's performance related to these tasks. These evaluation measures include: (i) **Subjective Evaluation:** This will be related to tasks that identify poor CQs, evaluating their relevance and accuracy; (ii) **Similarity Matching:** Techniques such as SBERT will be employed for calculating performance metrics—precision, recall, and F1-score—for tasks that involve identifying scoping CQs; (iii) **Testing for Verified CQs:** Involves similarity matching for the CQs and unit/acceptance testing for the corresponding SPARQL queries. Table 4 specifies the evaluation measures for each subtask identified earlier.

It is not trivial to collect a dataset for `CQsBEN` as open-sourced repository data often lack essential components, especially the design documents and testing programs. We identify two main implementation steps to organise the process; (i) **Gathering all Published Requirements:** Collecting and documenting all existing requirements related to tasks such as CORAL [25], the CQs dataset [54], along with individual ontologies that have published their CQs; (ii) **Categorisation According to Tasks:** organising the requirements based on their re-

Table 5. Datasets/ ontologies and their Corresponding Number of CQs.

Datasets/Ontologies	Num. CQs
CORAL [25]	834
CQs dataset [54]	234
WDV-CQ [17]	1786
DOREMUS [1]	218
NORIA-O [52]	55
Odeuropa [38]	13
Polifonia [12]	247

spective tasks to streamline the benchmark design process. Table 5 displays the initial set of proposed datasets, individual ontologies, along with the number of CQs. However, this is not the final set, as we continue to communicate with developers to encourage them to share their CQs and expand this list.

6 Conclusion

In this paper, we have conducted a *Scoping Study* on CQ engineering approaches, exploring the main dimension(s) that characterise approaches for CQ formulation, as well as considering differing levels of automation assumed by the approaches studied. Furthermore, we have examined the resources used by approaches that formulate CQs, and considered the validation criteria used to assess the relevance of the CQs with respect to their corresponding ontology. By doing so, this allows researchers to align their work with established approaches and have assisted ontology developers in selecting and adapting existing approaches to meet their specific needs. Our findings offer a structured overview that not only encapsulates the diversity of existing approaches but also clarifies their application contexts and limitations.

Furthermore, the study reports on the current state of the art in CQ engineering approaches, with a focus on the level of automation assumed, as well as the types of approach for which different types of validation criteria are meaningful. This level of reporting is crucial for positioning new methods relative to the state-of-the-art, thereby facilitating innovation and refinement in the field. Through this analysis, it has become evident that while fully automated approaches have emerged as a recent trend, designed to overcome the limitations of earlier methods. However, the field is still in its infancy, and as yet no single approach has emerged as consistently superior. This underscores the necessity for a standardised framework to assess these emerging methodologies.

Addressing this need, we have proposed a benchmark, `CQsBEN`, for CQs that includes detailed task specifications and evaluation criteria. It is designed to serve as a robust standard for assessing and comparing the effectiveness of different CQ engineering approaches, and should enable not only the comparison across the existing methods, but also provide a foundation for evaluating upcoming initiatives in CQ engineering.

References

1. Achichi, M., Lisena, P., Todorov, K., Troncy, R., Delahousse, J.: Doremus: A graph of linked musical works. In: The Semantic Web – ISWC 2018. pp. 3–19. Springer International Publishing (2018)
2. Alharbi, R.: Assessing candidate ontologies for reuse. In: Proc. of the Doctoral Consortium at ISWC 2021 (ISWC-DC). pp. 65–72 (2021), https://api.semanticscholar.org/CorpusID:244895203
3. Alharbi, R., Tamma, V., Grasso, F.: Characterising the gap between theory and practice of ontology reuse. In: Proceedings of the 11th on Knowledge Capture Conference. p. 217–224. K-CAP '21, Association for Computing Machinery (2021)
4. Alharbi, R., Tamma, V., Grasso, F.: Requirement-based methodological steps to identify ontologies for reuse. In: Intelligent Information Systems. pp. 64–72. Springer Nature Switzerland (2024)
5. Alharbi, R., Tamma, V., Grasso, F., Payne, T.: An experiment in retrofitting competency questions for existing ontologies (2023), https://arxiv.org/abs/2311.05662
6. Alharbi, R., Tamma, V., Grasso, F., Payne, T.: An experiment in retrofitting competency questions for existing ontologies. In: Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing. p. 1650–1658. SAC '24, Association for Computing Machinery (2024)
7. Alharbi, R., Tamma, V., Grasso, F., Payne, T.: The role of Generative AI in competency question retrofitting. In: Extended Semantic Web Conference, ESWC2024. Hersonissos, Greece (2024)
8. Antia, M., Keet, C.M.: Assessing and enhancing bottom-up CNL design for competency questions for ontologies. In: Proc. of the Seventh International Workshop on Controlled Natural Language (CNL 2020/21). pp. 1–11. Association for Computational Linguistics (ACL) (2021)
9. Antia, M., Keet, C.M.: Automating the generation of competency questions for ontologies with agocqs. In: Knowledge Graphs and Semantic Web. pp. 213–227. Springer Nature Switzerland (2023)
10. Arksey, H., O'Malley, L.: Scoping studies: towards a methodological framework. International Journal of Social Research Methodology **8**(1), 19–32 (2005)
11. Azzi, S., Assi, A., Gagnon, S.: Scoring ontologies for reuse: An approach for fitting semantic requirements. In: Proc. of the Reseach Conference on Metadata and Semantic Research, MTSR 2022. pp. 203–208. Springer Nature (2023)
12. de Berardinis, J., Carriero, V.A., Jain, N., Lazzari, N., Meroño-Peñuela, A., Poltronieri, A., Presutti, V.: The polifonia ontology network: Building a semantic backbone for musical heritage. In: The Semantic Web – ISWC 2023. pp. 302–322. Springer Nature Switzerland (2023)
13. Bezerra, C., Freitas, F.: Verifying description logic ontologies based on competency questions and unit testing. In: Proc. of the IX Seminar on Ontology Research and I Doctoral and Masters Consortium on Ontologies. vol. 1908, pp. 159–164 (2017)
14. Bezerra, C., Freitas, F.: Verifying description logic ontologies based on competency questions and unit testing. In: ONTOBRAS. pp. 159–164 (2017)
15. Bezerra, C., Santana, F., Freitas, F.: CQChecker: A tool to check ontologies in OWL-DL using competency questions written in controlled natural language. Learning & Nonlinear Models **12**(2), 115–129 (2014)
16. Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J.: Tropos: An agent-oriented software development methodology. Autonomous Agents and Multi-Agent Systems **8**(3), 203–236 (2004)

17. Ciroku, F., de Berardinis, J., Kim, J., Meroño-Peñuela, A., Presutti, V., Simperl, E.: Revont: Reverse engineering of competency questions from knowledge graphs via language models. Journal of Web Semantics **82**, 100822 (2024)
18. Dasiopoulou, S., Meditskos, G., Efstathiou, V.: Semantic knowledge structures and representation. Technical Report D5.1, FP7-288199 Dem@Care: Dementia Ambient Care: Multi-Sensing Monitoring for Intelligence Remote Management and Decision Support (2012), http://www.demcare.eu/downloads/D5.1SemanticKnowledgeStructures_andRepresentation.pdf
19. Debellis, M.: A practical guide to building owl ontologies using protégé 5.5 and plugins (04 2021), https://www.researchgate.net/publication/351037551_A_Practical_Guide_to_Building_OWL_Ontologies_Using_Protege_55_and_Plugins
20. Denaux, R., Dimitrova, V., Cohn, A.G., Dolbear, C., Hart, G.: Rabbit to OWL: Ontology authoring with a CNL-based tool. In: Controlled Natural Language. pp. 246–264. Springer Berlin Heidelberg (2010)
21. Dennis, M., van Deemter, K., Dell'Aglio, D., Pan, J.Z.: Computing authoring tests from competency questions: Experimental validation. In: The Semantic Web – ISWC 2017. pp. 243–259. Springer International Publishing (2017)
22. Espinoza-Arias, P., Garijo, D., Corcho, O.: Extending ontology engineering practices to facilitate application development. In: Knowledge Engineering and Knowledge Management. pp. 19–35. Springer International Publishing (2022)
23. Fathallah, N., Das, A., De Giorgis, S., Poltronieri, A., Haase, P., Kovriguina, L.: Neon-gpt: A large language model-powered pipeline for ontology learning. In: Extended Semantic Web Conference, ESWC2024. Hersonissos, Greece (2024)
24. Fernandes, P.C.B., Guizzardi, R.S., Guizzardi, G.: Using goal modelling to capture competency questions in ontology-based systems. Journal of Information and Data Management **2**(3), 527 (2011)
25. Fernández-Izquierdo, A., Poveda-Villalón, M., García-Castro, R.: CORAL: A corpus of ontological requirements annotated with lexico-syntactic patterns. In: Proc. of the 16th International Conference on The Semantic Web, ESWC 2019. pp. 443–458. Springer International Publishing (2019)
26. Funk, M., Hosemann, S., Jung, J.C., Lutz, C.: Towards ontology construction with language models. In: Proceedings of the KBC-LM'23: Knowledge Base Construction from Pre-trained Language Models workshop at ISWC. CEUR Workshop Proceedings (2023)
27. Gangemi, A., Lippolis, A.S., Lodi, G., Nuzzolese, A.G.: Automatically drafting ontologies from competency questions with frodo. Studies on the Semantic Web **55**, 107–121 (2022)
28. Gao, T., Fodor, P., Kifer, M.: High accuracy question answering via hybrid controlled natural language. In: 2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI). pp. 17–24 (2018). https://doi.org/10.1109/WI.2018.0-112
29. Gao, T., Fodor, P., Kifer, M.: Knowledge authoring for rule-based reasoning. In: On the Move to Meaningful Internet Systems. OTM 2018 Conferences: Confederated International Conferences: CoopIS, C&TC, and ODBASE 2018, Valletta, Malta, October 22-26, 2018, Proceedings, Part II. pp. 461–480. Springer (2018)
30. Grüninger, M., Fox, M.S.: The Role of Competency Questions in Enterprise Engineering, pp. 22–31. Springer US (1995)
31. Keet, C.M., Ławrynowicz, A.: Test-driven development of ontologies. In: Proc. of the 13th International Conference on The Semantic Web, ESWC 2016. pp. 642–657 (2016)

32. Keet, C.M., Mahlaza, Z., Antia, M.J.: CLaRO: A controlled language for authoring competency questions. In: Metadata and Semantic Research. pp. 3–15. Springer International Publishing (2019)
33. Kim, H.M., Fox, M.S., Sengupta, A.: How to build enterprise data models to achieve compliance to standards or regulatory requirements (and share data). Journal of the Association for Information Systems **8**, 105–128 (2007)
34. Kitchenham, B.A., Charters, S.: Guidelines for performing systematic literature reviews in software engineering. Tech. Rep. EBSE-2007-001, Keele University and Durham University (2007)
35. Kolbe, N., Vandenbussche, P.Y., Kubler, S., Le Traon, Y.: Lovbench: Ontology ranking benchmark. In: Proceedings of The Web Conference 2020. p. 1750–1760. WWW '20, Association for Computing Machinery (2020)
36. Kuhn, T.: A Survey and Classification of Controlled Natural Languages. Computational Linguistics **40**(1), 121–170 (03 2014). https://doi.org/10.1162/COLI_a_00168, https://doi.org/10.1162/COLI_a_00168
37. Li, Q., Li, S., Zhang, S., Hu, J., Hu, J.: A review of text corpus-based tourism big data mining. Applied Sciences **9** (2019). https://doi.org/10.3390/app9163300
38. Lisena, P., Schwabe, D., van Erp, M., Troncy, R., Tullett, W., Leemans, I., Marx, L., Ehrich, S.C.: Capturing the "semantics of smell": The Odeuropa Data Model for Olfactory Heritage Information. In: The Semantic Web: 19th International Conference, ESWC 2022. p. 387–405. Springer-Verlag (2022)
39. Mays, N., Roberts, E., Popay, J.: Synthesising research evidence. In: Fulop, N., Allen, P., Clarke, A., Black, N. (eds.) Studying the organisation and delivery of health services: Research methods. Routledge, London (2001)
40. Monfardini, G.K.Q., Salamon, J.S., Barcellos, M.P.: Use of competency questions in ontology engineering: A survey. In: Conceptual Modeling. pp. 45–64. Springer Nature Switzerland (2023)
41. Namgoong, H., Kim, H.: Ontology-based controlled natural language editor using CFG with lexical dependency. In: The Semantic Web. pp. 353–366. Springer Berlin Heidelberg (2007)
42. Noy, N.F., McGuinness, D.L.: Ontology development 101: A guide to creating your first ontology. Tech. rep., Stanford knowledge systems laboratory technical report KSL-01-05 (2001)
43. Poveda-Villalón, M., Fernández-Izquierdo, A., Fernández-López, M., García-Castro, R.: LOT: An industrial oriented ontology engineering framework. Engineering Applications of Artificial Intelligence **111**, 104755 (2022)
44. Presutti, V., Daga, E., Gangemi, A., Blomqvist, E.: Extreme design with content ontology design patterns. In: Proc. of the 2009 International Conference on Ontology Patterns. vol. 516, p. 83–97 (2009)
45. Rao, L., Reichgelt, H., Osei-Bryson, K.: Knowledge elicitation techniques for deriving competency questions for ontologies. In: Proceedings of the Tenth International Conference on Enterprise Information Systems (ICEIS 2008). vol. ISAS-2, pp. 105–110. Barcelona, Spain (2008)
46. Rebboud, Y., Tailhardat, L., Lisena, P., Troncy, R.: Can LLMs generate competency questions? In: Extended Semantic Web Conference, ESWC2024. Hersonissos, Greece (2024)
47. Reimers, N., Gurevych, I.: Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In: Proc. of the 2019 Conference on Empirical Methods in Natural Language Proc. and the 9th International Joint Conference on Natural Language Proc. (EMNLP-IJCNLP). pp. 3982–3992. Association for Computational Linguistics (2019)

48. Ren, Y., Parvizi, A., Mellish, C., Pan, J.Z., van Deemter, K., Stevens, R.: Towards competency question-driven ontology authoring. In: The Semantic Web: Trends and Challenges. pp. 752–767. Springer International Publishing (2014)
49. Sequeda, J.F., Briggs, W.J., Miranker, D.P., Heideman, W.P.: A pay-as-you-go methodology to design and build enterprise knowledge graphs from relational databases. In: Proc. of the 18th International Semantic Web Conference, ISWC 2019. pp. 526–545 (2019)
50. Sitthisak, O., Gilbert, L., Davis, H.C.: Transforming a competency model to parameterised questions in assessment. In: Web Information Systems and Technologies. pp. 390–403. Springer Berlin Heidelberg (2009)
51. Suárez-Figueroa, M.C., Gómez-Pérez, A., Fernández-López, M.: The neon methodology framework: A scenario-based methodology for ontology development. Applied ontology **10**(2), 107–145 (2015)
52. Tailhardat, L., Chabot, Y., Troncy, R.: NORIA-O: An ontology for anomaly detection and incident management in ict systems. In: The Semantic Web. pp. 21–39. Springer Nature Switzerland, Cham (2024)
53. Tseng, Y.H., Ho, Z.P., Yang, K.S., Chen, C.C.: Mining term networks from text collections for crime investigation. Expert Systems with Applications **39**(11), 10082–10090 (2012)
54. Wiśniewski, D., Potoniec, J., Ławrynowicz, A., Keet, C.M.: Analysis of ontology competency questions and their formalizations in SPARQL-OWL. Journal of Web Semantics **59**, 100534 (2019)