Multilingual Extraction Ontologies

David W. Embley, Stephen W. Liddle, Deryle W. Lonsdale, Byung-Joo Shin, and Yuri Tijerino

Abstract The growth of multilingual web content and increasing internationalization portends the need for cross-language query processing. We offer ML-OntoES (a **M**ulti**L**ingual **Onto**logy-based **E**xtraction **S**ystem) as a solution for narrow-domain/data-rich applications. Based on language-independent extraction ontologies (Embley et al., Conceptual modeling foundations for a web of knowledge. In: Embley D, Thalheim B (eds) Handbook of conceptual modeling: theory, practice, and research challenges. Springer, Heidelberg, Germany, pp 477–516, 2011a), ML-OntoES enables semantic search over domain-specific, semistructured information. Key ideas of ML-OntoES include: (1) monolingual semantic indexing and query interpretation with extraction ontologies and (2) conceptual-level cross-language translation. A prototype implementation, along with experimental work showing good extraction accuracy in multiple languages, demonstrates the viability of the ML-OntoES approach of using multilingual extraction ontologies for cross-language query processing.

Key Words Conceptual-level cross-language information transfer • Cross-language query processing • Extraction ontologies • Monolingual query interpretation • Monolingual semantic indexing

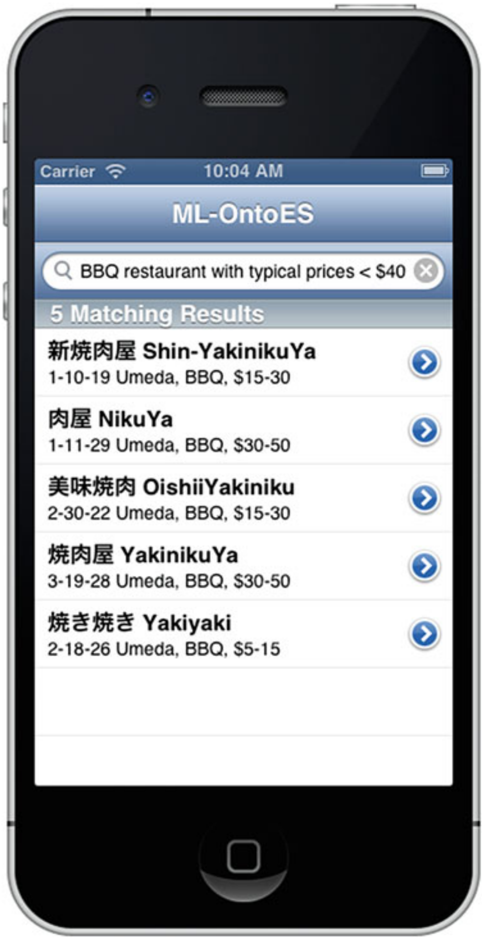
D.W. Embley (✉) • S.W. Liddle • D.W. Lonsdale
Brigham Young University, Provo, UT, USA
e-mail: embley@cs.byu.edu; liddle@byu.edu; lonz@byu.edu

B.-J. Shin
Kyungnam University, Kyungnam, Korea
e-mail: shinbyungjoo@gmail.com

Y. Tijerino
Kwansei Gakuin University, Kobe-Sanda, Japan
e-mail: ontologist@gmail.com

1 Introduction

An ideal cross-language query system would allow users to pose queries and receive answers in their own language when executing queries against foreign-language source documents. A user U, for example, who speaks only English, may wish to enquire about nearby restaurants while visiting Japan. Using an iPhone, U may wish to pose a query to find a "BBQ restaurant with typical prices $<$ \$40." Figure 1 shows an interface with the query in a type-in text field, the English version of the answers retrieved, and a "see further information button" to tap on to obtain more details such as hours of operation, payment method, and rating. Figure 2 gives actual answers retrieved from the web for this sample query (all in Japanese, of course), and this is the challenge—to query the Japanese in Fig. 2 with the English in Fig. 1.



BBQ restaurant with typical prices < \$40

5 Matching Results

新焼肉屋 Shin-YakinikuYa
1-10-19 Umeda, BBQ, \$15-30

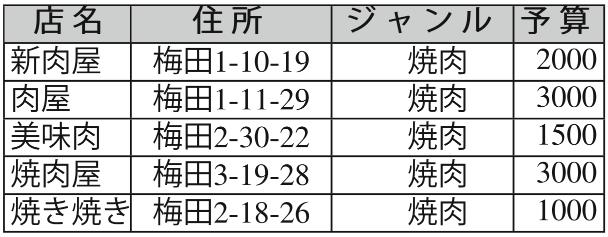
肉屋 NikuYa
1-11-29 Umeda, BBQ, \$30-50

美味焼肉 OishiiYakiniku
2-30-22 Umeda, BBQ, \$15-30

焼肉屋 YakinikuYa
3-19-28 Umeda, BBQ, \$30-50

焼き焼き Yakiyaki
2-18-26 Umeda, BBQ, \$5-15

Fig. 1 English query over Japanese data with results in English



店名	住所	ジャンル	予算
新肉屋	梅田1-10-19	焼肉	2000
肉屋	梅田1-11-29	焼肉	3000
美味肉	梅田2-30-22	焼肉	1500
焼肉屋	梅田3-19-28	焼肉	3000
焼き焼き	梅田2-18-26	焼肉	1000

Fig. 2 Results extracted from Japanese web pages

Queries like the English-Japanese BBQ restaurant query call for CLIR (Cross-Language Information Retrieval) (Olive et al. 2011; Peters et al. 2012). Interest in CLIR and related technologies is growing, and international initiatives are helping mature the field.[1] A typical approach to CLIR consists of query translation followed by monolingual retrieval and retranslation of results. Our approach to CLIR, which we describe in detail in Sect. 2, differs substantially: rather than translate a query at the language level, we first interpret it with respect to a conceptualization with both query and conceptualization in the same language; we then translate the query to an identical conceptualization in the target language, and having previously semantically annotated target documents with respect to the target-language conceptualization, we then retrieve results and reverse the conceptual translation to return final results in the language of the query.

The approach we take is not entirely unprecedented; several other types of systems use an "interlingua" to mediate processing of content between two or more languages. Since the days of symbolic pivot-based machine translation (Mahesh 1996), ontologies of various sorts have served in crosslinguistic applications including information extraction (Declerck et al. 2010; Aggarwal et al. 2013). Recently, ontology localization (Tijerino 2010) has become viable in boosting lexical content for translation. Some support translation via mappings between language-specific ontologies (Fu et al. 2012). Others, with the advent of statistical methods in natural language processing, use hybrid approaches in translating extraction-ontology content (Montiel-Ponsoda et al. 2011).

Because our approach is symbolic and ontology based and implements first-order (but not higher-order) logic for inference, the concerns raised by Hirst (this volume) could be relevant. We note, however, that the technologies for our system at present originate from the conceptual-modeling and data-extraction communities rather than from natural language processing and computational linguistics, though we foresee being able to orient our work more toward the nexus of all of these areas. In particular, our ontologies do not model the lexicon; they model conceptual relations, with relevant grounding in lexical entries, and the assertions they represent are more "data"-like than "information"-like and thus do not suffer as severely from the issues Hirst raises (this volume). In addition, since creating a domain ontology

[1] See, for example, http://www.clef-initiative.eu.

is within the purview of end users, they can either develop a writer-centered view of the data (i.e., more directly modeling the document type) or a reader-centered view (i.e., more oriented to which concepts are of most use to them). To avoid the grand pitfalls in Hirst's warning (this volume), we concentrate on data-rich, narrow-domain applications known a priori and consider our knowledge sources useful, if imperfect, artifacts. Furthermore, we adopt a multifaceted engineering approach for cross-language mappings, and while recognizing the equivalency problem, we allow for various types of correspondence beyond one-to-one mappings (Embley et al. 2011c).

What distinguishes our approach is the narrow, domain-specific, user-definable nature of our ontologies and their construction, as well as the role of these ontologies at the center of a larger infrastructure (Embley et al. 2011c). Our ontologies tend to be less elaborate than others and hence less rich in the types of context required for successful treatment by statistical translation methods. Our work is situated in the space of linguistically grounded, end-user-developed ontologies that incorporate various lexical resources and mappings at various levels of conceptualization.

These semantic conceptualization requirements limit our approach to applications that are easily conceptualizable—those that are data-rich and narrow in scope. Although limited, the applications are significant and practically important covering areas such as service finding like the restaurant example illustrated in Figs. 1 and 2, retail purchasing while shopping abroad, information seeking while traveling and sightseeing, and multicultural topical research such as family history where ancestors have immigrated to a country with a different language.

We call our cross-language query engine ML-OntoES (**M**ulti**L**ingual **Onto**logy **E**xtraction **S**ystem) and describe its architecture in Sect. 2. Like search engines, ML-OntoES assumes the existence of an indexed document collection. Indexes for ML-OntoES, however, are not just for keywords but are also for recognized semantic concepts. Extraction ontologies (Embley et al. 2011a), which we describe in Sect. 2.1, allow ML-OntoES to semantically index a document collection with respect to an ontological conceptualization. Extraction ontologies also allow ML-OntoES to interpret queries with respect to an ontological conceptualization, as we describe in Sect. 2.2. ML-OntoES matches conceptualized queries with the conceptualized semantic index to retrieve results. When the query language differs from the document-collection language, ML-OntoES invokes a conceptual-level translation as we explain in Sect. 2.3. In order for ML-OntoES to work well, semantic recognition accuracy must be high and extraction-ontology construction costs must be low; we address these issues in Sect. 3. In Sect. 4, we conclude by summarizing the principles and practicalities required to make ML-OntoES work successfully.

2 ML-OntoES Architecture

Figure 3 sketches the architecture of ML-OntoES by giving a retail-sales example in which ML-OntoES processes a French query against a collection of Korean car advertisements. Before query processing begins, ML-OntoES applies its Korean

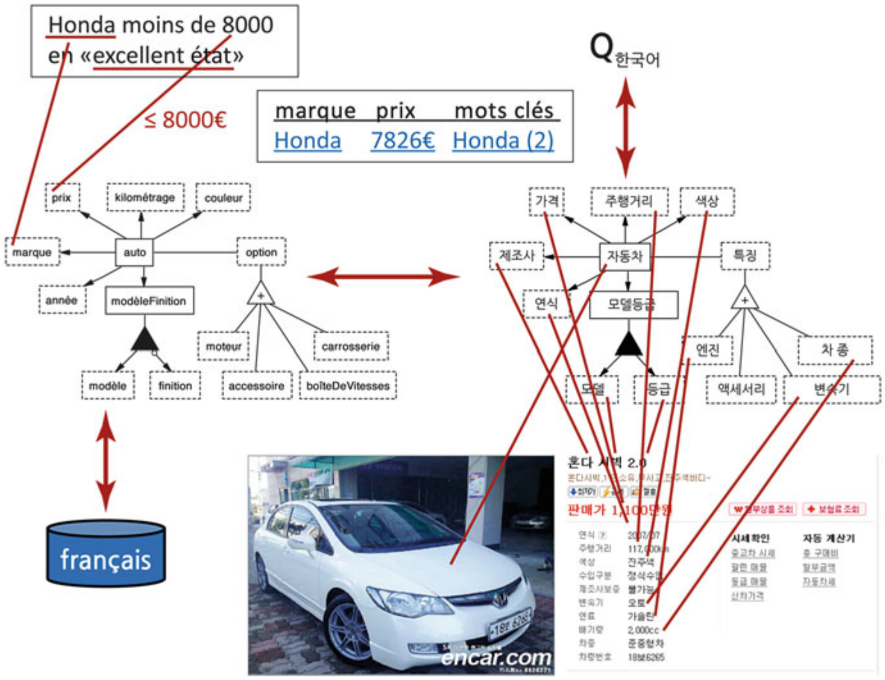


Fig. 3 Cross-language query processing

extraction ontology to Korean source pages to create a semantic index. Once semantic indexes have been built, query processing can begin: as Fig. 3 illustrates, ML-OntoES (1) applies a French car-ad extraction ontology to the query to recognize and conceptualize the query's semantic constraints and to remove semantic-constraint words from query, leaving and thus identifying the keywords; (2) maps the French conceptualization and keywords to the Korean conceptualization and keywords (note that the conceptualizations are structurally one to one, allowing for identical select-project-join processing); (3) matches the Korean conceptualization and keywords with the previously constructed semantic and keyword indexes; (4) maps the resulting Korean conceptualizations and keywords back into French; and (5) displays the results. As Fig. 3 shows, query processing of a Korean query $Q_{제조사}$ over the French repository, *français*, is symmetrical.

2.1 ML-OntoES Extraction Ontologies

An *extraction ontology* (see Figs. 4 and 5) is a 5-tuple (O, R, C, I, L):

O : Object sets—one-place predicates whose instance values are either all *lexical*, denoted by named dashed-border rectangles in Fig. 4, or all *nonlexical*, denoted

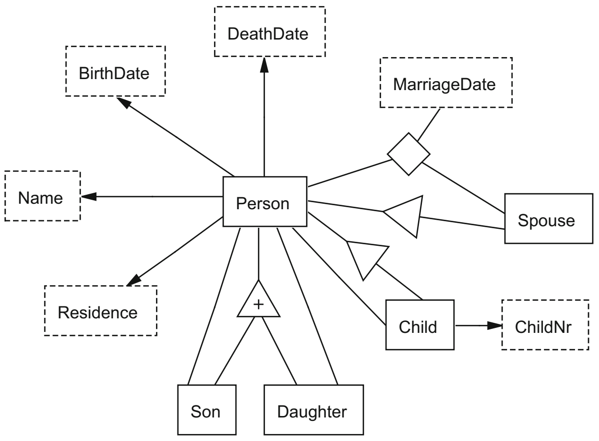


Fig. 4 Ontological conceptualization for assertion extraction

by solid-border rectangles (e.g., *BirthDate* is lexical with values such as "June 7, 1949" and *Person* is nonlexical with object-identifier values)

R : Relationship sets—n-place predicates, $n \geq 2$, represented by lines connecting object-set rectangles (e.g., *Person–Name* in Fig. 4) and also by black-triangle aggregation symbols connecting holonyms (e.g., *modèleFinition* in Fig. 3) to meronyms (e.g., *modèle* and *finition*)

C : Constraints—closed formulas, as implied by the notation (e.g., $\forall x(Person(x) \Rightarrow \exists! y(Person\text{-}BirthDate(x, y)))$—one of the many functional constraints denoted by the arrowhead on the range side of the *Person-BirthDate* relationship set; $\forall x(Child(x) \Rightarrow Person(x))$—a hypernym/hyponym constraint denoted by the triangle, which may optionally also specify mutual exclusion among its hyponym sets by a "+" symbol (e.g., mutual exclusion of *Son* and *Daughter* in Fig. 4) or specify that the hypernym set is a union of its hyponym sets ("$\cup$") or both ("$\uplus$") to form a partition among its hyponyms)

I : Inference rules—logic rules specified over predicates (e.g., *Person–Gender*(x, 'Female') :- *Daughter*(x))

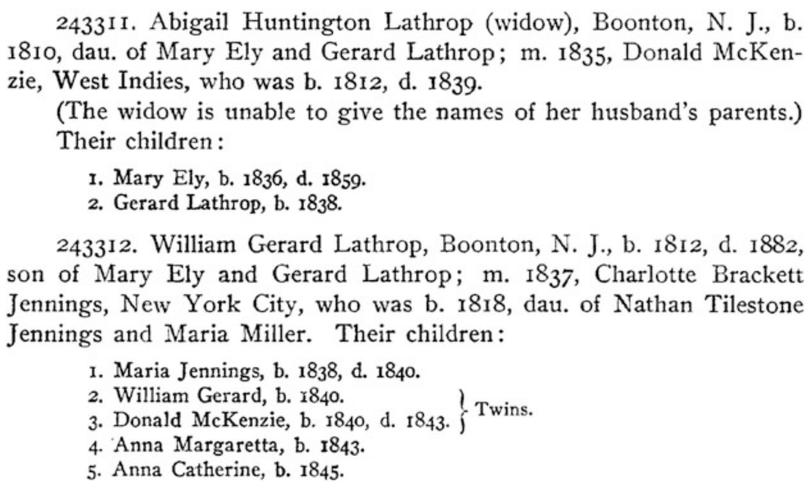
L : Linguistic groundings—text recognizers for populating object and relationship sets and collections of interrelated object and relationship sets (e.g., recognizers for *Name* and *BirthDate* in Fig. 5)

The conceptual foundation for an extraction ontology is a restricted fragment of first-order logic, but its most distinguishing feature is its linguistic grounding,[2] which turns an ontological specification into an extraction ontology. Each object set has a *data frame* (Embley 1980), which is an abstract data type augmented with linguistic recognizers that specify textual patterns for recognizing instance

[2]Similar to the linguistic grounding discussed in Buitelaar et al. (2009), but different in its details.

```
Name
   external representation: \b{FirstName}\s{LastName}\b
   external representation: \b{FirstName}\s[A-Z]\w+\b
   ...
BirthDate
   external representation: \b1[6-9]\d\d\b
      left context: b\.\s
      right context: [.,]
      context keywords: \bborn\b(\sin\b)?|...
   ...
   input method: DateStringToJulianDate
   output method: JulianDateToDateString
   operator methods:
      LessThan(p1: BirthDate, p2:BirthDate) returns (Boolean)
      external representation: (before|earlier than|<)\s{p2} ...
   ...
```

Fig. 5 Sample recognizers for linguistically grounding the ontology in Fig. 4



243311. Abigail Huntington Lathrop (widow), Boonton, N. J., b. 1810, dau. of Mary Ely and Gerard Lathrop; m. 1835, Donald McKenzie, West Indies, who was b. 1812, d. 1839.

(The widow is unable to give the names of her husband's parents.) Their children:

1. Mary Ely, b. 1836, d. 1859.
2. Gerard Lathrop, b. 1838.

243312. William Gerard Lathrop, Boonton, N. J., b. 1812, d. 1882, son of Mary Ely and Gerard Lathrop; m. 1837, Charlotte Brackett Jennings, New York City, who was b. 1818, dau. of Nathan Tilestone Jennings and Maria Miller. Their children:

1. Maria Jennings, b. 1838, d. 1840.
2. William Gerard, b. 1840. } Twins.
3. Donald McKenzie, b. 1840, d. 1843. }
4. Anna Margaretta, b. 1843.
5. Anna Catherine, b. 1845.

Fig. 6 An excerpt from p. 419 of *The Ely Ancestry*

values, context keywords, applicable operators, and operator parameters. The data frame for *BirthDate* in Fig. 5 illustrates recognizers for both instance values and operator applicability. Although any kind of textual pattern recognizer is possible, our current implementation supports only regular expressions or combinations of regular expressions and dictionaries. Relationship sets may also have data-frame recognizers. Recognizers for larger ontological components are also possible—*Ontology Snippets*, as we call them.

We explain how the linguistic recognizers work by showing how they apply to an OCRed excerpt from the *The Ely Ancestry* (Beach et al. 1902) in Fig. 6.

- *Lexical object-set recognizers* identify lexical instances in terms of external representations, context, exclusions, and dictionaries. One of the possibly many **external representation**s for *BirthDate* in Fig. 5 is "\b1[6–9]\d\d\b", representing years between 1600 and 1999, with an immediate **left context** of "b\.\s", an immediate **right context** of "[.,]", and **context keywords** that include "\bborn\b(\sin)?", which may appear close to but not necessarily immediately adjacent to the birth year. Note that these regular-expression patterns match all the birth years in Fig. 6. The **external representation**s for *Name* in Fig. 5 illustrate the use of dictionaries and mixed dictionaries and regular expressions. A name in curly braces within a regular expression references a named regular expression (e.g., "{FirstName}" references a dictionary of given names: "Aaron|Abdul|Abbey|..."). An **input method** converts a recognized string into an appropriate internal representation—for example, a Julian-date representation in Fig. 5, and an **output method** converts an internal representation to a standard format for display to a user. Applicable **operator methods** are particularly useful for constraints in queries like "List Mary Ely's children born before 1840" where parameter $p1$ comes from an extracted value and $p2$ follows "before".
- *Nonlexical object-set recognizers* identify nonlexical objects through object existence rules, which identify text such as proper nouns, that designate the existence of objects. The object existence rule "{Name}" for the nonlexical object set *Person*, for example, references the regular expressions in the *Name* object set, and when a name is recognized, ML-OntoES generates a *Person* object and associates it with the recognized name.
- *Relationship-set recognizers* identify phrases that relate objects. For example, the regular expression "^\d{1,2}\.\s{Person},\sb\.\s{BirthDate}[.,]" for the *Person–BirthDate* relationship set relates Maria Jennings to 1838 and William Gerard to 1840—two of the *Person–BirthDate* relationships that appear in Fig. 6.
- *Ontology-snippet recognizers* identify text patterns that provide instances for groups of object and relationship sets. Recognizers for ontology snippets consist of regular expressions with capture groups and predicate mappings.

To effectively recognize semantic object and relationship instances in text, we must often tune extraction ontologies to the view of the text provided by its author (e.g., tune Figs. 4 and 5 to the author's view in Fig. 6). An author's view, however, may differ in its organization and content from the view we wish to have as we query the extracted information. We can obtain the view we want (e.g., Fig. 7) by using the inference-rule component of ML-OntoES.

In our prototype implementation, we use the Jena reasoner (http://jena.apache.org) over RDF triples to specify inference rules. Since ML-OntoES is fundamentally specified as a set of n-ary predicates ($n \geq 1$), the Jena reasoner immediately applies. Moreover, its results are also n-ary predicates, which lets us conveniently augment an ML-OntoES ontology. We can, for example, have the rules

target:Person(x) :- source:Person(x)
target:Person–Gender(x,'Male') :- source:Son(x)
target:Father(x) :- target:Person–Child(x,y),target:Person–Gender(x,'Male')

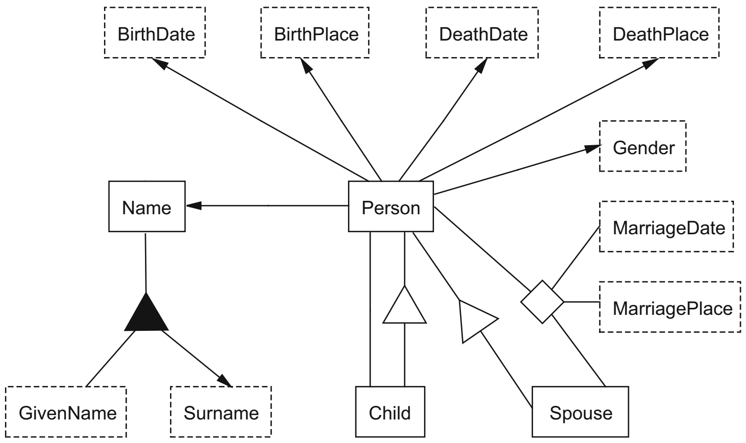


Fig. 7 Target ontology of desired biographical assertions

which, respectively, specify that persons in a source ontology (e.g., Fig. 4) become persons in the target ontology (e.g., Fig. 7), that sons are male, and that persons who have a child and are male are fathers. Furthermore, the Jena reasoner defines a set of built-in predicates that is extensible, and we can create extensions to specify predicates that, for example, can split a name such as "William Gerard Lathrop" into two given names and a surname and that can infer the surname of the children for the culture in which *The Ely Ancestry* was written as the surname of the father. Inferred object and relationship sets may have data-frame recognizers, thus making inferred assertions directly queryable.

In addition to inferring assertions, ML-OntoES also has the ability to reason over the stated and implied assertions to do entity resolution. In our prototype implementation, we use the Duke entity resolver (http://code.google.com/p/duke) and generate OWL *same-as* relationships when, for example, Duke discovers that of the three "Mary Ely"s in Fig. 6, only the first and third are the same.

2.2 ML-OntoES Monolingual Query Processing

Before query processing begins, ML-OntoES preprocesses a document collection and creates a keyword index and a semantic index. In our prototype implementation, ML-OntoES creates its keyword index with Lucene (http://lucene.apache.org) and its semantic index with extraction ontologies. ML-OntoES applies extraction ontologies to text documents to find instance values in the documents with respect to the object and relationship sets in the ontology as explained in Sect. 2.1 and illustrated for Korean in Fig. 3. ML-OntoES returns its semantic index as RDF triples.

Assuming a known context—an identified extraction ontology—ML-OntoES first distinguishes between semantic and keyword text in the query and processes semantics through the semantic index and keywords through the keyword index. ML-OntoES then combines the results and subsequently ranks and displays retrieved documents, for example, as suggested by Fig. 1, allowing users to click on results to view original documents from which information was extracted and, in the case of inferred results, to also see the reasoning chains.

For the French query in Fig. 3, the data-frame recognizers in the French car-ad extraction ontology recognize "Honda" and "moins de 8000" and convert them to the constraints $marque$ = "Honda" and $prix < 8000$€. For monolingual query processing, ML-OntoES generates a SPARQL query from these constraints that not only finds cars that satisfy the constraints in its semantic index but also retrieves information about references to its cached copies of the web pages from which ML-OntoES extracted the information—thus making the semantic index an actual index into its known web pages.

Assuming that users wish to have as many of the semantic constraints satisfied as possible and knowing that users may query for constraints not specified in source documents, ML-OntoES generates conjunctive queries and allows SPARQL constraint satisfaction to be optional. Then, for acyclic conceptualizations (e.g., the application ontologies in Fig. 3), ML-OntoES generates queries in a straightforward way: join over edges in the ontologies that connect identified nodes, and filter conjunctively on identified conditions. For the query in Fig. 3, for example, ML-OntoES produces the SPARQL equivalent of $\pi_{marque,prix}\sigma_{marque='Honda' \wedge prix<8000}(auto\text{–}marque \bowtie auto\text{–}prix)$.[3] For cycles, ML-OntoES identifies all possible paths in the conceptual-model graph that cover identified object and relationship sets and then either acknowledges the ambiguity and returns answers for all paths or discovers that the query explicitly identifies one or more of the paths and returns answers only for these paths.

ML-OntoES processes free-form queries conjunctively. However, like standard search engines, it also provides for advanced-search capabilities for queries that involve disjunctions and negations. When a user requests the advanced-search option for an application, ML-OntoES dynamically generates a form from the application's extraction ontology. The form provides for negations with a checkbox, disjunctions with click-extended OR buttons, and comparators for all declared comparison operations in the application's data frames.

For keyword query processing to work well, it is necessary to remove stopwords plus words and phrases intended to convey semantic constraints or result types. Thus, ML-OntoES removes stopwords such as "de" and "en" and a phrase like "moins de 8000", which it recognizes as generating a semantic constraint. Semantic-phrase removal prevents terms such as "moins" from matching irrelevant tokens in documents. ML-OntoES also removes semantic phrases expressing equality constraints such as "Marque égale Honda", but for recognized equality constraints,

[3] By π and σ, we mean projection and selection, respectively.

it leaves the value word or phrase as a keyword. Thus, in our example, "Honda" becomes a keyword. ML-OntoES also passes quoted phrases, such as «excellent état», to Lucene to process as single-phrase keywords.

2.3 ML-OntoES Cross-Language Query Processing

Given a query Q in language L_1 and an interpretation of Q with respect to a conceptualization also in language L_1, ML-OntoES maps Q from the conceptualization in language L_1 to a corresponding conceptualization in language L_2. Cross-language conceptualizations are structurally identical, and therefore since the semantic concepts and constraints have a one-to-one correspondence, the implied select-project-join operations for query Q will be the same in both conceptualizations. Thus, for example, the SPARQL equivalent of the French query $\pi_{marque,prix}\sigma_{marque='Honda' \wedge prix<8000}$ $(auto\text{–}marque \bowtie auto\text{–}prix)$ becomes a SPARQL equivalent of the Korean query $\pi_{\text{제조사},\text{가격}}\sigma_{\text{제조사}='\text{혼다}' \wedge \text{가격}<11700800}$ $(\text{자동차–제조사} \bowtie \text{자동차–가격})$.

For narrow-domain, data-rich applications, we expect native-language extraction ontologies for different languages/locales to be similar, but not necessarily identical. Thus, when adding a new extraction ontology to ML-OntoES for a new language or new localization of an existing language, we check structural consistency and make adjustments as necessary to retain the structural one-to-one correspondence across all ontologies. In Korean car ads, for example, mention of accidents is common. Assuming the accident concept is not yet part of the existing conceptualizations, we can either drop the concept from the Korean ontology (deeming it not essential) or add it to all other ontologies for the application.

For keywords and instance values in semantic constraints, ML-OntoES uses existing services for currency conversions, keyword translation, unit conversions, and transliterations and uses existing language resources and pay-as-you-go construction for lexicon and commentary translations[4]:

- *Lexicons*. Lexicon mappings substitute one word by another or one word by a small number of others. For common concepts such as colors, corresponding translations are available in cross-language dictionaries. Interestingly, these mappings are not always one to one (e.g., "blue" in Korean is 파랑색 and 파란색 and 청색).
- *Units and Measures*. ISO standard conversion formulas for units and measures are commonly available, and coding them is straightforward. In our implemen-

[4]Our mapping typology here resonates with that of León-Araúz and Faber (this volume), though our lexical type inventory is not as finely articulated.

tation, we use, for example, kilometers for mileage, integers for car years, Julian calendar specifications for dates, and a 24-hour clock for time.

- *Currency*. Because services exist that directly convert amounts in one currency to amounts in any other currency, mappings for currency conversions are direct from one language/localization to another.
- *Transliteration*. Like direct conversion among currencies, transliteration mappings are direct from one language to another.
- *Keywords*. Since keywords can be any word or quoted phrase, we use a general translation service.
- *Commentary*. Ontologies may contain free-form commentary to explain unfamiliar concepts, such as localized tipping protocols.

For answer values returned, we use the mappings to transform values and keywords back into the original language. In Fig. 3, for example, ML-OntoES maps the Korean car make 혼다 first into its language-agnostic equivalent and then into the French "Honda", and the currency converter converts the Korean Won price 1,100만원 into 7,826€ and the twice-appearing keyword 혼다 via a general translation service into "Honda (2)".

Development and maintenance of ML-OntoES cross-language mappings agree in spirit with the principles of Bosca et al. (this volume). Our methods and tools, however, obviously vary somewhat.

3 Practicalities

How well ML-OntoES works in practice primarily depends on the accuracy of its linguistic grounding, which, in turn, depends on the quality of its knowledge engineering. For ML-OntoES to be successful, we must sufficiently increase semantic recognition accuracy and sufficiently decrease engineering construction costs.

3.1 Recognition Accuracy

Cross-language query-processing accuracy depends on (1) extraction accuracy in all languages when indexing the semantics in a document collection and (2) cross-language query transformation so that nothing is lost or spuriously added.

To check extraction accuracy, we built French and Korean extraction ontologies for car-ad and obituary applications. The combinations represent typological variety across languages and document diversity in degree of semistructuredness. From 500 French car ads, 1,500 French obituaries, 430 Korean car ads, and 502 obituaries, gathered from several different online sites, we randomly selected about 100 of each of the four combinations to constitute validation and blind test sets (respectively, 20 and 80 of the 100) and used the rest for training (in the sense that we looked at many of them as we built our ontologies).

Table 1 Car ad within-language extraction results

		Make (%)	Model (%)	Year (%)	Price (%)	Color (%)	Mileage (%)
French	Recall	87	76	96	89	82	98
	Precision	65	67	90	95	47	92
Korean	Recall	99	99	100	100	100	95
	Precision	99	99	100	100	100	95

Table 2 Obituary within-language extraction results

				Death	Funeral		
		Title	Name (%)	Date (%)	Date (%)	Time (%)	Place (%)
French	Recall	76 %	42	80	69	43	38
	Precision	99 %	63	88	70	30	83
Korean	Recall	N/A	97	97	50	50	100
	Precision		97	97	100	100	67

Table 3 Cross-language query transformation results

	Recall			Precision		
Car-ad queries	σ (%)	π (%)	κ (%)	σ (%)	π (%)	κ (%)
French-to-English	77	86	100	81	90	74
Korean-to-English	98	100	100	93	99	52

Tables 1 and 2 show the results. The car-ad domain is ontologically narrow, and accordingly, our extraction ontologies perform quite well on this domain (as we have come to expect (Embley et al. 2011a)). Precision and recall for Korean car ads are high because these ads mostly have a regular structure, allowing our Korean expert to quickly tune the extraction ontology. The French car ads are more free-form, and so the results are lower. The obituary domain is much broader, and extraction is more challenging—particularly for names and places. Even so, our Korean expert was able to quickly tune the extraction ontology, and performance for most concepts was remarkably high. French extraction was hampered by greater variability and complex sentence structures. For example, there are only 187 names in our Korean surname dictionary, compared with 228,429 in our French surname dictionary, which partially explains the relatively high performance for Korean name extraction.

To check cross-language query transformation accuracy, we asked students in two senior-level database classes to generate car-ad queries which they felt an earlier demo version of a free-form query processor should interpret correctly. The students generated 137 syntactically unique queries, of which 113 were suitable for testing ML-OntoES. To obtain Korean and French queries, we faithfully translated 50 of these 113 into each language.

Table 3 shows the results of interpreting the queries in their respective languages and transforming the internal representation of each query, as understood, into the internal representation of the query in English. In the table, σ and π, respectively, represent query selection (i.e., conditionals such as "Price < $12,000") and query projection (i.e., choice of results to include, e.g., the make and model of a car), and κ represents keywords. Since σ and π translations are always correct, the less-than-perfect σ and π results come from inaccurate within-language query interpretation. The lower recall and precision for French conditionals (σ) points to a need for better recognizers. More complete synonym sets for French ontological concepts (π) would increase recall but may decrease precision. Expanded stopword lists in French would remove spurious keywords (κ) like "list" and "want". Stopwords in Korean make little sense because most of the standard English-like stopwords are prefixes and suffixes and become part of glyphs. An attempt to remove them after translation often fails because translations themselves are often poor; for example, 인, which in our query should translate as "which is"—both English stopwords—instead was translated as "inn" (or "hotel").

3.2 Construction Cost

The ML-OntoES architecture requires a substantial amount of information that must be encoded, either by hand or through some automated means. The difficulty of eliciting or otherwise acquiring such data from domain experts—Feigenbaum's "knowledge engineering bottleneck" (Feigenbaum 1984)—is a decades-old issue.

Our approach substantially mitigates, without completely solving, this problem: our system uses narrow, domain-dependent ontologies that a typical user should be able to specify. We have developed interactive tools for designing and populating ontologies with the requisite types of knowledge, and we are investigating the use of machine learning and linguistic analysis to reduce the cost of developing recognizers for linguistically grounding ontologies. Furthermore, we advocate and practice re-using to the degree possible already extant knowledge sources, and we resonate with similar work being done by other researchers to leverage a wide variety of resources in the boosting of ontology content for crosslinguistic extraction while minimizing the cost (Fu et al. 2012), also convincingly advocated by Bond et al. (this volume).

We assume that end users knowledgeable in a particular domain can create focused, narrow-in-scope ontologies that involve extraction of relevant content from data-rich knowledge sources. In the context of crosslinguistic extraction, ontology creators need to know the languages for which they are designing ontologies. Creation of the ontologies involves specifying concepts, relationships, constraints, and lexical items useful for extraction. Three methods are available for ontology creation and population: (1) programmers can hand-populate them by entering data directly into the data structure; (2) experienced users can interact with the data structure via our custom-designed ontology editor, a tool for specifying ontology content; or (3) domain experts with limited experience can interact with a form-driven interface that guides the user through design decisions necessary to provide content. The time and effort involved for developing an ontology typically involve one person's efforts over several days, perhaps at the most a week or two, less time if the user has expertise in language, lexicons, and text processing techniques. As with any knowledge engineering task, there is a point of diminishing returns in specifying expert knowledge: more time and effort can be spent developing content to increase performance but at the risk of experiencing the knowledge-engineering bottleneck. A short but representative list of resource types we have used or are considering using for ontology creation and population follows:

- *Lexical databases*: Several publicly available lexical resources—monolingual and multilingual—provide comprehensive information on lexical semantic relations: synonymy, hypernymy, hyponymy, meronymy, word senses, and crosslinguistic mappings. Example resources include the WordNet (http://wordnet.princeton.edu), the GlobalWordNet (http://www.globalwordnet.org), and the BabelNet (http://lcl.uniroma1.it/babelnet).
- *Lexicons*: Specialized lists of narrow-domain words of interest are readily found on the Web: gazetteers for place names, census indexes for person names, and product name databases are some examples. For our evaluation work in Sect. 3.1, we mined pull-down menus from http://paruvendu.fr which contains all French automobile make/model combinations and mined tabs from http://www.encar.com which lists Korean makes and models.
- *Term banks*: The computerization and subsequent web deployment of vast terminology banks, such as TermiumPlus (http://www.termiumplus.gc.ca) and EuroTerm (http://www.euroterm.org/test1/glossary), has put literally millions of concepts and their single-word and multiword terms within easy reach of the general public. In prior work, we have shown how to integrate terminological resource content into our ontologies (Lonsdale et al. 2002).
- *Transliteration services*: When crosslinguistic mappings involve different character sets, services can perform character conversion. In our current implementation, we use a Hangul/Roman transliterator (http://sori.org/hangul/conv2kr.cgi) for Korean to/from English. Unfortunately, no general transliteration resource appears to be currently available.
- *Translation services*: LabelTranslator (http://www.neon-toolkit.org), for example, provides translation (called by others "localization services") for ontology labels between three European languages. For general-purpose translation, services based on statistical machine translation systems can be used; we currently use Bing (http://api.microsofttranslator.com/V2/Http.svc/Translate) when more direct methods are not readily available.

The crosslinguistic aspect of our system involves a star-based architecture similar to notion in Dorr et al. (2006) that maps between languages at the conceptual-model level (Embley et al. 2011c). At the center of the star is a language-agnostic pivot that mediates between language-specific extraction ontologies. Since conceptual associations are routinely direct, this removes the necessity to translate between languages and allows for recovering the mappings from the isomorphic ontological content. Furthermore, the effort required to add another language to the system only involves developing the relevant knowledge sources for the new language. The complexity of adding a new language to the system is thus reduced from $O(n^2)$ to $O(n)$.

As ML-OntoES becomes more reliant on external resources, it also becomes subject to what Hoekstra calls the "knowledge reengineering bottleneck" in the context of the Semantic Web, with its four new challenges (Hoekstra 2010): (1) Our system is *data dependent* since its effectiveness, robustness, and scalability depend on the appropriateness and quantity of data we incorporate from elsewhere. (2) We have *limited control* over the dirtiness of the data we process and over the coverage of the resources we adopt. (3) ML-OntoES becomes subject to *increased complexity* as disparate resources are integrated into the system. (4) As our system transitions from small-scale systems to large-scale web applications, it assumes *increased importance*. With the star-based architecture of the system and through careful selection of relevant knowledge resources, we hope to be able to strike a pragmatic balance among these issues, at least for data-rich, narrow-domain applications.

4 Conclusion

ML-OntoES processes cross-language, hybrid query and keyword-search requests for narrow-domain, data-rich applications in accord with three principles: (1) monolingual semantic indexing based on extraction ontologies, (2) monolingual extraction-ontology-based semantic analysis of user queries, and (3) structurally identical application ontologies to facilitate conceptual-level cross-language mappings:

1. For query processing to work in reasonable time, semantic indexes must exist. ML-OntoES creates semantic indexes by crawling web pages and documents on the web with application-dependent, monolingual extraction ontologies. Then, for each assertion found (as explained in Sect. 2.1), we can record the assertion's objects in their identified ontological object sets and its relationships among the objects in its identified ontological relationship sets and associate the object and the relationship pointers into a cached copy of the page or document.
2. When a user submits a query, it is best if the system already knows the context in which the query is asked—that is, already knows which ontology or set of ontologies, prepopulated with assertions, should be used to return an answer. Otherwise, the system must search for an application ontology (or a set of application ontologies) by applying candidate extraction ontologies to the query and checking the coverage. Indexes over words and common conceptualizations such as dates and currencies can speed up the process of locating appropriate

ontologies for the query. Then, as explained in Sect. 2.2, ML-OntoES can monolingually construct a query with respect to the structure of the ontology.

3. As noted in Sect. 2.3, since all language-and-locale versions of extraction ontologies for a particular application are structurally identical, generated query expressions have the same form in all versions, and only the instance values, if any, need translation. ML-OntoES uses cross-language dictionaries for word substitutions, standard conversion formulas for units and measures, online currency converters for currency exchange, and transliteration services for name conversions. Keyword and commentary translation are more difficult to translate accurately. But rough approximations, as provided by online translators, are often sufficient. For critical vertical applications where specialized keywords and jargon words matter in hybrid queries, special application-dependent keyword and keyword-phrase cross-language dictionaries can be developed as a supplement for online translators. Likewise, when commentary is critical, such as for business transactions and detailed instructions, careful translations would need to be written, if they do not already exist.

Our prototype implementation demonstrates feasibility, but as a practical matter, for ML-OntoES to be successful, extraction-ontology recognition accuracy must be high (Sect. 3.1), and extraction-ontology construction costs must be low (Sect. 3.2). Summarizing our discussion of these issues in Sect. 3, we point out that the knowledge engineering required for car ads and obituaries returned reasonably good precision and recall results for French and particularly good for Korean, and that the time and effort required to develop the extraction ontologies, given the lexical resources available to us, are within reason. This "knowledge-engineering bottleneck" is, however, a drawback of ML-OntoES.

Because of this drawback, our current and expected future efforts for ML-OntoES are focused on mitigating extraction-ontology construction costs. Focusing on the vertical domain of historical documents and particularly family-history documents (Embley et al. 2011b), we are exploring ways to automate the construction of extraction ontologies. For lists, which are commonly found in family-history documents, we have been able to generate both regular-expression and HMM recognizers that accurately extract genealogical assertions of interest and insert them into ontological structures (Packer and Embley 2013). We are currently working on automating the extraction of more general text patterns found in semistructured documents and on combining a dependency parser with a semantic reasoner to generate assertions that can be inserted into a target ontology. The domain of family history is particularly in need of cross-language query processing, especially for untrained users because many people have ancestors who have come from countries with a language foreign to their own.

Acknowledgments We are grateful to Tae Woo Kim and Rebecca Brinck for annotating our Korean and French document sets. We are also grateful to the reviewers for their insightful feedback and challenging questions.

References

Aggarwal, N., Polajnar, T., & Buitelaar, P. (2013). Cross-lingual natural language querying over the web of data. In E. Métais, F. Meziane, M. Saraee, V. Sugumaran, & S. Vadera (Eds.), *Natural Language Processing and Information Systems: 18th International Conference on Applications of Natural Language to Information Systems (NLDB 2013). Lecture Notes in Computer Science* (Vol. 7934, pp. 152–163). New York: Springer.

Beach, M., Ely, W., & Vanderpoel, G. (1902). *The ely ancestry*. New York: The Columer Press. On-line book: http://www.archive.org/details-/elyancestrylinea00beac.

Buitelaar, P., Cimiano, P., Haase, P., & Sintek, M. (2009). Towards linguistically grounded ontologies. In *Proceedings of the 6th European Semantic Web Conference (ESWC'09)* (pp. 111–125). Heraklion: Greece.

Declerck, T., Krieger, H.-U., Thomas, S., Buitelaar, P., O'Riain, S., Wunner, T., et al. (2010). *Ontology-based multilingual access to financial reports for sharing business knowledge across Europe* (pp. 67–76). Budapest: Memolux Kft.

Dorr, B., Hovy, E., & Levin, L. (2006). Machine translation: Interlingual methods. In *Natural language processing and machine translation. Encyclopedia of Language and Linguistics* (2nd ed., pp. 383–394). Oxford, UK: Elsevier Ltd.

Embley, D. (1980). Programming with data frames for everyday data items. In *Proceedings of the 1980 National Computer Conference*, Anaheim, CA (pp. 301–305).

Embley, D., Liddle, S., & Lonsdale, D. (2011a). Conceptual modeling foundations for a web of knowledge. In D. Embley & B. Thalheim (Eds.), *Handbook of conceptual modeling: Theory, practice, and research challenges* (Chap. 15, pp. 477–516). Heidelberg, Germany: Springer.

Embley, D., Liddle, S., Lonsdale, D., Machado, S., Packer, T., Park, J., et al. (2011b). Enabling search for facts and implied facts in historical documents. In *Proceedings of the International Workshop on Historical Document Imaging and Processing (HIP 2011)*, Beijing, China (pp. 59–66).

Embley, D., Liddle, S., Lonsdale, D., & Tijerino, Y. (2011c). Multilingual ontologies for cross-language information extraction and semantic search. In *Proceedings of the 30th International Conference on Conceptual Modeling (ER 2011)*, Brussels, Belgium (pp. 147–160).

Feigenbaum, E. (1984). Knowledge engineering: The applied side of artificial intelligence. *Annals of the New York Academy of Sciences, 426*, 91–107.

Fu, B., Brennan, R., & O'Sullivan, D. (2012). A configurable translation-based cross-lingual ontology mapping system to adjust mapping outcomes. *Web Semantics: Science, Services and Agents on the World-Wide Web, 15*, 15–36.

Hoekstra, R. (2010). The knowledge reengineering bottleneck. *Semantic Web—Interoperability, Usability, Applicability, 1*, 1–5.

Lonsdale, D., Ding, Y., Embley, D., & Melby, A. (2002). Peppering knowledge sources with SALT: Boosting conceptual content for ontology generation. In *Proceedings of the AAAI Workshop: Semantic Web Meets Language Resources: The 18th National Conference on Artificial Intelligence*, Edmonton, AB, Canada (pp. 30–36).

Mahesh, K. (1996). Ontology development for machine translation: Ideology and methodology. Technical Report MCCS-96-292. Albuquerque, NM: Computing Research Laboratory, University of New Mexico.

Montiel-Ponsoda, E., de Cea, G. A., Gómez-Pérez, A., & Peters, W. (2011). Enriching ontologies with multilingual information. *Natural Language Engineering, 17*(3), 283–309.

Olive, J., Christianson, C., & McCary, J. (Eds.). (2011). *Handbook of natural language processing and machine translation: DARPA global autonomous language exploitation*. New York: Springer.

Packer, T., & Embley, D. (2013). Cost effective ontology population with data from lists in OCRed historical documents. In *Proceedings of the 2nd International Workshop on Historical Document Imaging and Processing (HIP 2013)*, Washington, DC, USA (pp. 1–8).

Peters, C., Braschler, M., & Clough, P. (2012). *Multilingual information retrieval: From research to practice*. New York: Springer.

Tijerino, Y. (2010). Cross-cultural and cross-lingual ontology engineering. In *Proceedings of the 2010 Workshop on Cross-Cultural and Cross-Lingual Aspects of the Semantic Web*, Shanghai, China (pp. 44–53).