PROJECT DESCRIPTION

1 Introduction

The web contains a wealth of knowledge. Unfortunately, most of the knowledge is not encoded in a way that enables direct user query. We cannot, for example, directly google for a used car that is a 2003 or newer selling for under 15 grand; or for the names of the parents of great-grandpa Schnitker; or for countries whose population will likely decrease by more than 10% in 50 years.

Our particular focus in this proposal is on users who wish to gather facts, perhaps like the facts just mentioned, organize them, and use them for decision making. Applications of this sort include, for example, government employees wishing to gather comparative information about degrees awarded by educational institutions, individuals interested in product comparison or their own family history, military leaders gathering facts surrounding front-line casualties, and bio-researchers assembling facts for a bio-medical study.

All people who use the web for research, whatever their motivation, need to be able to find, retrieve, filter, extract, integrate, organize, and share information in a user-friendly, timely, and high-precision manner. In all cases, facts contained in web pages are central; they must be located, interpreted, and annotated for further study. Our approach is to annotate stated facts with respect to ontologies.[1] We populate these ontologies, turning them into a database over which structured queries can be executed. Annotation links also provide a form of provenance and authentication, allowing users to verify query results by checking original sources. Furthermore, facts and ontological concepts may appear in more than one populated ontology. Linking facts and ontological concepts across ontologies can provide navigation paths to explore additional, related knowledge. The web with a superimposed layer of interlinked ontologies each annotating a myriad of facts on the underlying web becomes a *Web of Knowledge*, a *WoK*.[2]

Our solution to the challenge of developing a WoK is to create "knowledge bundles" (KBs), which are conceptual-model representations of organized information superimposed over source documents. A "knowledge-bundle builder" (KBB) helps researchers develop KBs in a synergistic and incremental manner and is a manifestation of learning in terms of its semi-automatic construction of KBs.

Although this vision of a WoK and associated KBs is appealing, there are significant barriers preventing both its creation and its use. Ontology languages exist, with OWL being the *de facto* standard. RDF files can provide data for these ontologies and can also store annotation information linking data to facts in web pages and linking equivalent information in RDF files to one another. The SPARQL query language is a standard for querying RDF data. Thus, all constituent components for a WoK are industry standards in common use, and they even all work together allowing for immediate WoK development and usage. Nevertheless, the barriers of creation and usage remain high and effectively prevent WoK deployment. The creation barrier is high because of the cost involved in developing OWL ontologies and annotating web pages by linking RDF-encoded facts in web pages to these OWL ontologies. The usage barrier is high because untrained users cannot write SPARQL queries.

Extraction ontologies provide a way to solve both creation and usage problems [DEL06]. To describe extraction ontologies and show how they resolve creation and usage problems, we formalize extraction ontologies (Section 2), formalize the notion of a WoK (Section 3), formalize

[1] We view an ontology as a formal theory captured in a model-theoretic view of data within a formalized conceptual model based on predicate calculus.

[2] To many, this vision of a WoK constitutes the semantic web [W3C].

WoK construction procedures (Section 4), and formalize user-friendly, WoK query-processing procedures (Section 5). The success of the WoK vision depends on a solid theoretical foundation. Thus, the contributions we hope to make with this project are: (1) the formalization of extraction ontologies, knowledge bundles, and interconnected knowledge bundles as a WoK and (2) based on the formalization, the development of WoK construction tools and WoK usage tools that largely overcome WoK creation and usage barriers. Realizing these contributions is no small task, but nearly a decade of prior research (Section 6) positions us to push forward (Sections 6 and 7) and achieve our objectives (Section 8).

2 Extraction Ontologies

We represent web information via the conceptual modeling language OSM (Object-oriented Systems Model) [EKW92], which provides a graphical representation of a first-order-logic language. We restrict OSM to be decidable, yet powerful enough to represent desired ontological concepts and constraints. We call our restriction *OSM-O*, short for *OSM-Ontology*. We thus base our foundational conceptualization directly on an appropriate restriction of first-order logic. This WoK foundation should be no surprise since it is the basis for modern information systems and has been the basis for formalizing information since the days of Aristotle [AriBC].

In this proposal we sketch the incremental sequence of formal models that characterize our approach to representing the various kinds of facts, data, information, and knowledge that are encapsulated in the web. We present these models as n-tuples comprised of specialized but closely correlated components.

Initially, we view an OSM ontology (OSM-O) as a triple (O, R, C) where:

- O is a set of intensional object sets; each is a one-place predicate; and each predicate has a designation as *lexical* (e.g., a city name like "Chicago") or *non-lexical* (e.g., an object identifier for a particular city). These are sometimes called concepts or classes, though they may also serve as properties or attributes. (Examples: $Person(x)$, $CityName(x)$.)

- R is a set of n-ary relationship sets ($n \geq 2$); each is an n-place predicate. (Examples: $Person(x)$ *is citizen of* $Country(y)$, $Car(x)$ *has* $Feature(y)$.)

- C is a set of constraints that assure (i) referential integrity; (ii) appropriate participation of objects in relationship sets in terms of min:max cardinality; (iii) generalization/specialization properties; (iv) aggregation (holonymy/meronymy) properties. (Examples: $\forall x(Student(x) \Rightarrow Person(x))$, $\forall x(Car(x) \Rightarrow \exists^1 y(Car(x)\ has\ Year(y)))$

Figure 1 shows an OSM-O model instance. Rectangular boxes are object sets—dashed if lexical and solid if non-lexical. Lines between object sets denote relationship sets. Participation constraints are next to relationship-set/object-set connections in a *min:max* format. (We use *1* as shorthand for *1:1*.) A white triangle denotes generalization/specialization with the generalization connected to the apex of the triangle and the specializations connected to the base. A black triangle denotes aggregation with the super-part connected to the apex of the triangle and the sub-parts connected to the base. Although graphical in appearance, an OSM-O diagram is merely a two-dimensional rendition of predicates and closed, well-formed formulas.

The quadruple (O, R, C, I) characterizes *information* and is an information system or database. Given M, an OSM-O model instance, we can relate it to the domain under consideration by declaring truth values for each of the predicates that are instantiated (whether lexical or non-lexical). All instantiated predicates represent **True** facts. When all constraints C hold, we

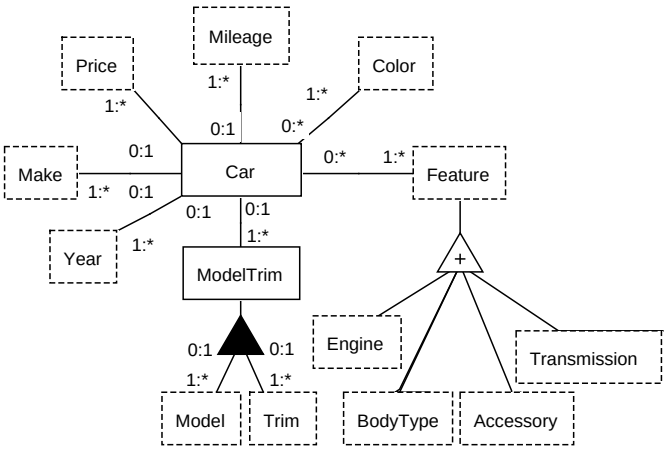

Figure 1: OSM-O Model Instance.

have a model for M which we refer to as a *valid interpretation* to avoid making "model" ambiguous in the context of creating conceptual models for domain ontologies. A valid interpretation of the OSM-O model instance in Figure 1 contains facts about cars. A possible valid interpretation might include the facts $Car(Car_3)$, $Year(2003)$, $Car\text{-}Year(Car_3, 2003)$, $Model(\text{"Accord"})$, $Trim(\text{"LX"})$, $ModelTrim(ModelTrim_{17})$, $Trim\text{-}isPartOf\text{-}ModelTrim$ ("LX", $ModelTrim_{17}$), and $Car\text{-}ModelTrim(Car_3, ModelTrim_{17})$. Note that the object sets *Car* and *ModelTrim*, being non-lexical, have identifiers for their domain-value substitutions. Constraints, such as $\forall x(Car(x) \Rightarrow \exists^{\leq 1} y(Car\text{-}Year(x, y)))$, all hold.

In our implementation, we use OWL to represent ontologies and RDF for storing instances with respect to OWL ontologies. A detailed technical discussion of the OSM-O formalism, its mapping to standard description logics, and its decidability appears elsewhere [EZ10].

Similar in approach to LexInfo [BCHS09] and OpenDMAP [HLF+08], we are able to linguistically ground OSM-O. The L component of the quintuple (O, R, C, I, L) adds linguistic grounding to an OSM-O information system and turns OSM-O model instances into *OSM-Extraction-Ontology* (*OSM-EO*) model instances. To do this we define an abstract data type for each object set and then add linguistic recognizers for instance values, operators, operator parameters, and relationships. We call these linguistically augmented abstract data types *data frames* [Emb80]. Like a frame in AI [Min75], which describes a "chunk of knowledge" in some domain, a data frame describes a "chunk of data"—its value set including lexical instance recognizers and its operation set including lexical recognizers for operators and their parameters.

Figure 2 shows two partial data frames for object sets in the OSM-O model instance in Figure 1, one for *Price* and one for *Make*. The *Price* data frame uses regular expressions for its recognizers, whereas the *Make* data frame uses a lexicon. In our implementation, we can use either or both together. The *Price* data frame also shows a recognizer for an operator. The *p2* within curly braces indicates the expected appearance of a *Price* parameter *p2*. Thus a phrase like "under 15 grand" is recognized as indicating that the price of a car—parameter *p1*—should be less than $15,000. A data frame for a non-lexical object set is typically degenerate: its value set is a set of object identifiers. Its operation set consists only of operators that add and remove object identifiers. Its name and synonyms for its name that identify the presence of one of its non-lexical object instances, however, can be quite rich. For relationship sets, the definition of a data frame does not change, but a typical view of the definition shifts as we allow value sets to be n-tuples

Price
internal representation: Integer
external representation: \$[1-9]\d{0,2},?\d{3} | \d?\d [Gg]rand | ...
context keywords: price|asking|obo|neg(\.|otiable)| ...
...
LessThan(p1: Price, p2: Price) **returns** (Boolean)
context keywords: (less than | < | under | ...)\s*{p2} | ...
...
Make
...
external representation: CarMake.lexicon
...

Figure 2: Data Frames.

of values rather than scalar values. Further, like recognizers for operators, they rely on instance recognizers from the data frames of their connected object sets. For example, suppose the *Car* object set in Figure 1 has a relationship set to a *Person* object set representing a person selling a car. Then, the relationship-set data frame would have a recognizer such as {*Person*} (*is selling* | *places ad for* | ...) {*Car*}, where the object sets in curly braces specify the object-set instance recognizers on which the relationship-set instance recognizer depends.

When we have an OSM-O model instance M with a data frame associated with each object set and relationship set, M is an OSM-EO model instance. An OSM-EO model instance is linguistically grounded in the sense that it can both "read" and "write" in some natural language. To "read" means to be able to recognize facts in natural language text and to extract fact instances with respect to the ontology in the OSM-EO model instance. To "write" means to display fact instances so that they are human-readable.

How well a particular OSM-EO model instance can "read" and "write" makes a difference in how well it performs. Our experience is that OSM-EO model instances can "read" some documents well (over 95% precision and recall [ECJ+99]), but it is clear that opportunities abound for further research and development. Writing human-understandable descriptions is less difficult to achieve—the system just selects any one of the phrases for each object set and relationship set (e.g., $Person(Person_{17})$ *is selling* $Car(Car_{734})$, $Car(Car_{734})$ *has* $Make(Honda)$). Making the written description pleasing is more difficult, of course.

3 Web of Knowledge

Ontology is the study of "the nature of existence." Epistemology is the study of "the origin, nature, methods, and limits of human knowledge." In the previous section we have sketched our computational view of ontology—a view that lets us work with ontologies in information systems. We similarly give a computational view of epistemology. The computational view of epistemology we give here constitutes a formal foundation for a web of knowledge, a WoK.

The collection of facts in an OSM-O model instance constitutes the *extensional knowledge* of the OSM-O model instance. The collection of implied facts derived from the extensional knowledge by inference rules constitutes the *intentional knowledge*. The extensional and intentional knowledge together constitute the *knowledge* of the OSM-O model instance.

Although this view of knowledge is common in computing, Plato and those who follow his line of thought also demand of knowledge that it be a "justified true belief" [PlaBC]. "Knowledge" without some sort of truth authentication can be confusing and misleading. But how can we

attain truth authentication? We see three possibilities: (1) truth as community agreement—e.g., Wikipedia style; (2) probabilistic truth; and (3) truth derived from proper reasoning chains grounded in original sources. All three, unfortunately, are problematic: community agreement depends on the willingness of individuals to participate and to agree; probabilistic truth depends on establishing probabilities and on being able to derive probabilities for answers to queries—hard problems that do not scale well [DRS09]; and reasoning with rules and fact sources depends on acceptance of the rules and fact sources as genuine and authoritative.

For our vision of a WoK, we attempt to establish truth via provenance and authentication. We provide for reasoning with rules and for ground facts in sources. We cannot, however, guarantee that rules and facts in sources are genuine. We thus compensate by simply exposing them. When an extraction ontology extracts a fact from a source document, it retains a link to the fact; and when a query answer requires reasoning over rules, the system records the reasoning chain. Users can ask to see fact sources and reasoning chains, and in this way they can authenticate facts and reasoning the way we usually do—by checking sources and fact-derivation rules.

The sextuple (O, R, C, I, L D) characterizes a *computational view of knowledge*. This includes D, a deductive rule set where rules are safe, positive, horn clauses formulated as datalog rules, which are decidable [Ros05]. Adding the D component allows us to reason deductively over the base facts in the information system. In our implementation, we use SWRL rules and the Pellet reasoner.

The septuple (O, R, C, I, L, D, A) characterizes a *Platonic view of knowledge*. Adding the A component provides a form of authentication since users can trace knowledge back to its source, which Plato insists is part of the definition of knowledge [PlaBC]. It provides for extensional links from domain concepts to source information.

We define a *knowledge bundle* (KB) as the septuple (O, R, C, I, L, D, A). Finally, we define a *Web of Knowledge* (WoK) as a collection of knowledge bundles interconnected with binary links, $<x, y>$, of two types: (1) *object identity:* non-lexical object identifier x in knowledge bundle B_1 refers to the same real-world object as non-lexical object identifier y in knowledge bundle B_2. (2) *Object-set identity:* object set x in knowledge bundle B_1 designates the same set of real-world objects as object set y in knowledge bundle B_2.

4 WoK Construction

To construct a WoK, we must be able to construct a knowledge bundle, and we must be able to establish links among knowledge bundles. We can construct knowledge bundles and establish links among them by hand (and this should always be an option). However, scaling WoK construction demands semi-automatic procedures, with as much of the construction burden placed on the system as possible—all of it when possible. For knowledge bundles, our automated construction tools identify applicable source information and transform it into knowledge-bundle components. For links among knowledge bundles, we apply record-linkage and schema-mapping tools.

We define the *transformation* we seek as a 5-tuple (R, S, T, Σ, Π), where R is a set of resources, S is the source conceptualization, T is the target conceptualization for an S-to-T transformation, Σ is a set of declarative source-to-target transformation statements, and Π is a set of procedural source-to-target transformation statements. We have conducted preliminary work on these transformations [TEL+05, LE09]. For the R component in this preliminary work, we used some of the basic links in WordNet [Fel98] and a minimal data-frame library. For the S component, we focused on canonicalized tables [JN08, PJK+09]. For the T component, we aimed at OSM-O knowledge bundles. In our implementation, we coded procedural (Π) source-to-target transformation statements, some of which were based on declarative (Σ) statements.

In previous work we have also shown how information contained in tables (including nested tables) can be harvested from web sites like WormBase and converted into OSM-O ontologies [TE07], and then to knowledge bundles for a WoK. Furthermore, when several tables on a web site have similar tabular content and structure (i.e., sibling pages), we can automatically construct an ontology for the entire site. We can also reverse the process and let users specify ontologies via nested forms [TEL09].

To the extent possible, we want our transformations to preserve information and constraints. Let S be a predicate calculus theory with a valid interpretation, and let T be a populated OSM-O model instance constructed from S by a transformation t. Transformation t *preserves information* if there exists a procedure to compute S from T. Let C_S be the closed, well-formed formulas of S, and let C_T be the closed, well-formed formulas of T. Transformation t *preserves constraints* if $C_T \Rightarrow C_S$.

For a WoK, preservation and transformation requirements entail extraction of each base fact and its representation in an ontology. Hence, to make a WoK highly meaningful, we should recover as much as is possible of the underlying semantics—the facts, the constraints, and the linguistic connections. Therein lies the difficulty: some of the underlying semantics in source conceptualizations exist only implicitly and are thus difficult to capture, and some of the underlying semantics do not exist at all, having been discarded in the abstraction process of producing the conceptualization. But therein also lie the research opportunities. Many researchers are endeavoring to create automatic and semi-automatic procedures to capture richer semantics—making extensive use of the Σ and Π components of transformation. And some are making use of the R component to recover semantics lost in the abstraction process. In general, there is an effort to recover as much of the semantics as possible from many different source genres. For example, we and other researchers have investigated semantic recovery from relational databases [EX97, Ast04, ABA08], XML [AK07, WNB06], human-readable tables [PSC+07, TEL+05, LE09], forms [GMJ04, SWL09], and free-running text [Cim06].

For the last stage of WoK creation—creating links among knowledge bundles—we rely on *record linkage* and *schema mapping*, also called *ontology alignment* or *ontology matching*. *Record linkage* is the task of finding entries that refer to the same entity in two or more data sources, and *ontology matching* is the task of finding the semantic correspondences between elements of two ontology schemas. An extensive body of work exists for both record linkage [EIV07] and ontology matching [ES07]. Unfortunately, both problems are extremely hard. General solutions do not exist and may never exist. However, "best-effort" methods do exist and perform reasonably well. For the WoK vision, we can use these best-effort methods to initialize and update *same-as* links for object identity and *equivalence-class* links for concept identity. Using "best-effort" methods in a "pay-as-you-go" fashion appears to be a reasonable way to enable a WoK.

5 WoK Usage

The construction of extraction ontologies leads to "understanding" within a WoK. This "understanding" leads to the ability to answer a free-form query because, as we explain in this section, a WoK system can identify an extraction ontology that applies to a query and match the query to the ontology. This identification leads to a reformulation of the free-form query as a formal query, so that it can be executed over a knowledge bundle.

Let S be a source conceptualization and let T be a target conceptualization formalized as an OSM-EO model instances. We say that T *understands* S if there exists an S-to-T transformation that maps each one-place predicate of S to an object set of T, each n-place predicate of S to an n-place relationship set of T ($n \geq 2$), each fact of S to a fact of T with respect to the predicate

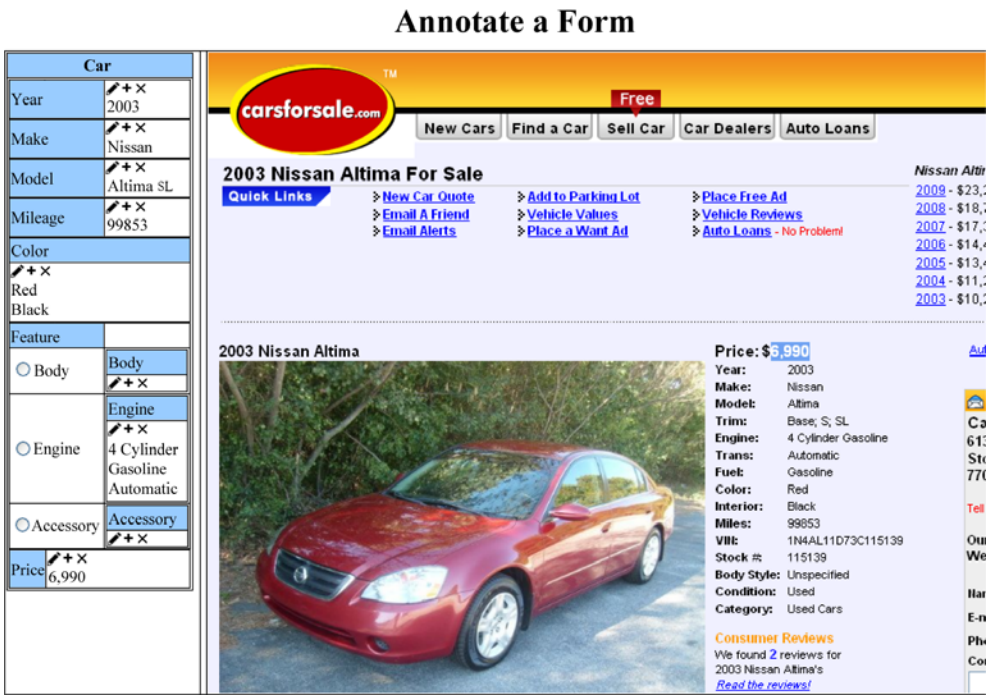

Figure 3: A Filled in Form with a Source Data Page

mappings, and each operator of S to an operator in a data frame of T, such that the constraints of T all hold over the transformed predicates and facts.

Observe that although this definition of "understanding" states how T is formalized, it does not state how S is formalized. Thus, the predicates and operators of S may or may not be directly specified. This is the hard part of "understanding"—to recognize the applicable predicates and operators. But this is exactly what extraction ontologies are meant to do. If an OSM-EO model instances is linguistically well grounded, then it can "understand" so long as what is stated in S is within the context of T—that is if there is an object set or relationship set in T for every predicate in S and if there is an operator in a data frame of T for every operator in S.

Applications and tools for "understanding" include automated ontology construction, free-form query processing, and advanced form-query processing.

5.1 Form-based Ontology Builder

Developing ontologies by hand is difficult and time-consuming. On the other hand, using forms is a natural way to elicit information from users. As one way to semi-automatically construct OSM-EO ontologies, we have developed a tool called *FOCIH* (*Form-based Ontology Creation and Information Harvesting*) that allows users to create their own forms to harvest information they are seeking [Tao08]. Once users define forms, they can copy and paste information from web pages into form fields. From this information, FOCIH generates an ontology that can annotate web pages with respect to the ontology and automatically harvest information from comparable pages elsewhere.

For example, suppose we wish to buy a used car. We can let FOCIH harvest the information we wish to consider and then query the harvested information to find cars we may want to buy. As Figure 3 indicates in its left panel, we specify features of interest: *Year*, *Make*, *Model*, *Mileage*, and *Price* are examples of single-value features. We may also want to specify colors; since a car

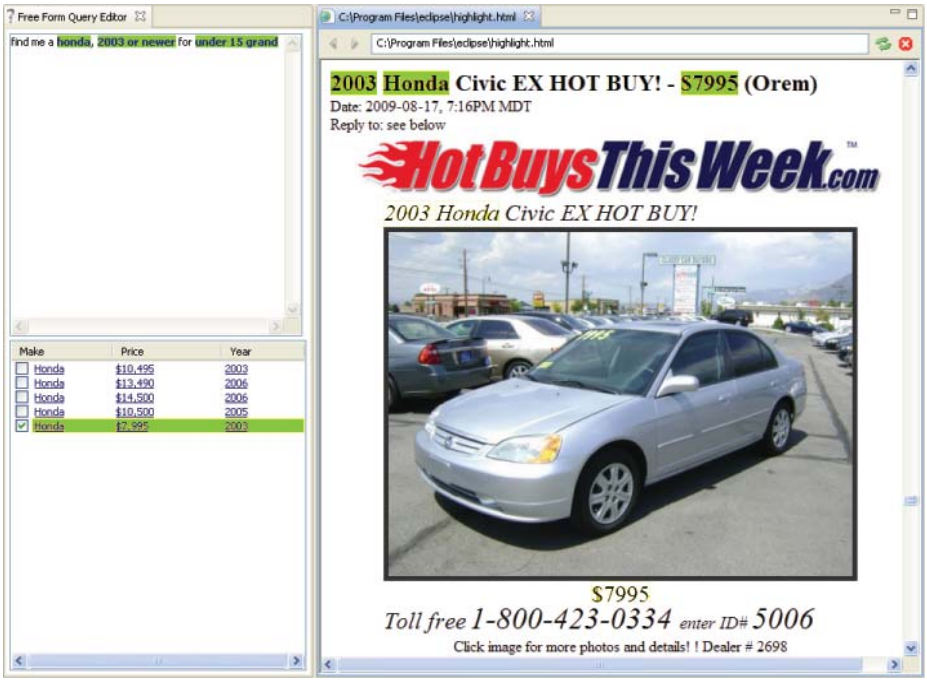

Figure 4: Screenshot of WoK Prototype Showing Free-Form Query Processing.

may have more than one color, this would be an example of a multiple-value element. A car may have several other features of interest: *Body* features, *Engine* features, and *Accessory* features. Once a form has been filled in with these fields, users can browse to a web page they wish to annotate and copy and paste values into form fields. A user highlights values in the web page and then clicks on the pencil icon in the form field to fill in a value. Figure 3 shows the price $6,990 highlighted and entered in the *Price* form field. To add several values to a multiple-value field, a user adds each to the field one at a time. The values "4 Cylinder", "Gasoline", and "Automatic" for example are all *Engine* features. To concatenate values that may be separate such as "Altima SL" in Figure 3, a user adds subsequent components by clicking on the plus icon instead of the pencil icon. (The $\times$ icon deletes filled-in values.)

From the filled-in form, FOCIH can generate both a conceptual model, eventually to be represented as an OWL ontology, and an annotation document, eventually to be represented as RDF triples. Further, FOCIH also records the annotation information: (1) paths to leaf nodes in the DOM tree of an HTML page containing each value and, for concatenated values, each value component; (2) for each value the most specific instance recognizer from the data-frame library (e.g., string, number, year, make, model, color); and (3) enough left, right, and delimiter context within each leaf node to identify the value or values within the DOM-tree node. This enables FOCIH to harvest the same information from all machine-generated sibling pages from the same web site.

Moreover, from the user-declared form and the harvested information, FOCIH is able to assemble an OSM-EO ontology (O, R, C, I, L). FOCIH obtains its O, R, and C components from analyzing the user-declared form, obtains its I component from the harvested information, and constructs its L component for each object set in O and relationships in R from the I information harvested. Lexicons for L are lists of harvested values. For example, for *Make*, FOCIH would likely have harvested "Dodge", "Ford", "Honda", ... from the pages of the web site. Value recognizers for L come from matching harvested values with data frames in a data-frame library. For example, a *Price* data frame would match the highlighted value $6,990 in Figure 3.

5.2 Free-Form Query Processing

Figure 4 illustrates free-form query processing within our WoK prototype. To "understand" a user query, our WoK prototype first determines which OSM-EO ontology applies to the query by seeing which one recognizes the most instances, predicates, and operators in the query request. Having chosen the *Car* extraction ontology illustrated in Figures 1 and 2, the WoK applies the S-to-T transformation highlighting what it "understands" ("Find me a *honda*, *2003 or newer* for *under 15 grand*"). Figure 5 shows the result of this transformation—each predicate and each operation is mapped correctly and the constraints of the OSM-EO model instance all hold. Given this "understanding," it is straightforward to generate a SPARQL query. Before executing the query, our WoK prototype augments it so that it also obtains the stored annotation links. Then, when our WoK prototype displays the results of the query in the lower-left box in Figure 4, it makes returned values clickable. Clicking on a value, causes our WoK prototype to find the page from which the value was extracted, highlight it, and display the page appropriately scrolled to the location that includes the value. The right panel of Figure 4 shows several highlighted values, which happens when the user checks one or more check-boxes before clicking.

5.3 Advanced Form-Query Processing

The form in Figure 5 is for an alerter system that we have implemented for craigslist.org. We use it as feedback to a user that a query has been understood (as just explained in Section 5.2). As such, it illustrates not only the ability of an OSM-EO ontology to "read" and "understand," but also its ability to "write." Note, for example, the conversion of "15 grand" to "$15,000" and the mnemonic names for predicates and operations. Besides providing feedback, this writing ability also lets the user know what else the OSM-EO ontology knows about. A user then has the opportunity to adjust the query or add additional constraints. For example, the user may also wish to know if Toyotas are listed so long as they are not Camrys. Clicking on *OR* for *Make* and adding *Toyota* and then clicking on *NOT* for *Model* and adding *Camry* makes this possible. The plus icons show that more operators are available; clicking on the plus displays them. For example, the user might wish to limit prices with $Between(Car.Price, \$11K, \$16K)$. Since the OSM-EO ontology has general recognizers for prices, a user can enter them in any recognizable format.

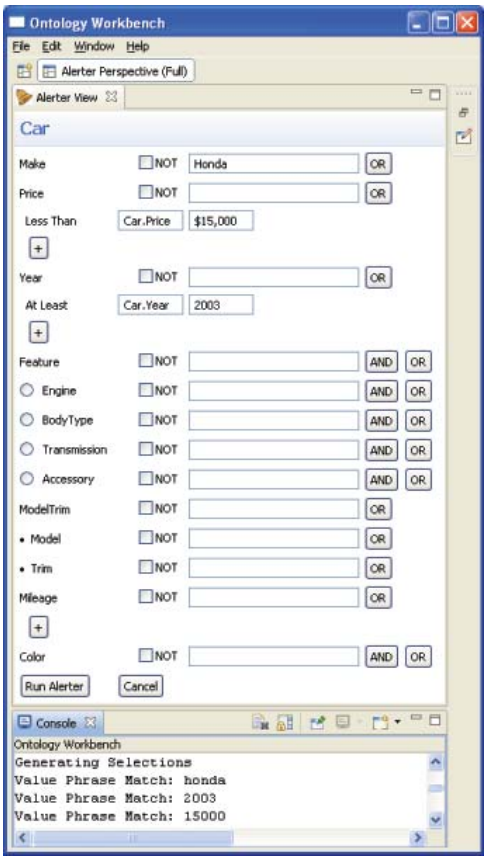

Figure 5: Generated Form.

6 Prior and Proposed Research

We have discussed our WoK vision from a theoretical perspective and sketched how to build a WoK using sound techniques. We have also presented examples of how a user could use tools to specify and find information that would be of interest, so that any information located and extracted can be encoded, annotated, and queried. Our vision for a WoK and its associated KB's is informed by years of work in this area. Although there is still much work to accomplish toward this goal, we have made incremental and innovative progress in developing and prototyping the ideas, methods, and tools we have presented here. We now explain what we have accomplished and what we intend to accomplish.

6.1 Prior NSF-sponsored Research

Our previous NSF-sponsored collaborative project "TANGO: Table ANalysis for Generating Ontologies" leveraged the strengths of our research teams at Brigham Young University (IIS-0414644, PI: David W. Embley) and Rensselaer Polytechnic Institute (IIS-0414854, PI: George Nagy). It began on 8/01/2005 and ended 7/31/2008 with a one-year no-cost extension that ended on 7/31/2009. TANGO is a framework for organizing domain-specific factual data appearing in lists and tables in independently generated web pages. Algorithms and software were developed for extracting and interpreting individual lists and tables and integrating them with the contents of other tables that have partially overlapping information.

The tools developed in TANGO along with the tools developed in an even earlier NSF-funded project on extraction ontologies (IIS-0083127, PI: David W. Embley) are the starting point for the proposed research. We list these tools below in Section 7 where we also show how they provide the basis for the work we plan to do for this proposal.

In addition to developing tools, these two cross-disciplinary and cross-university endeavors introduced 25 graduate students (including 6 women) and 9 undergraduate students to cutting-edge research. The effort produced 17 MS theses ([Yau01, Jac02, Cha03, Din03, Tao03, Che04, AM04, AK04, Wal04, Wes05, Par05, Zho05, Vic06, Jha08, Lyn08, Lia08, Pad09]). and 4 PhD dissertations ([Xu03, AK07, AM07, Tao08]). and over 45 student-co-authored, peer-reviewed publications, many of which are referenced throughout this proposal.

6.2 Proposed Work

In general, we intend to enhance and integrate the tools we have built so far, so that they can enable the envisioned WoK. More specifically, however, we have four research objectives, which we explain here.

Grammar-Based Data Frames for Relationship Sets

We have defined and implemented data frames to locate facts that correspond to simple extensional objects and to link them to concepts (IIS-0083127). For the most part this work involves only one-place predicates which typically represent the semantics of nouns and adjectives. Any relationships between concepts identified and represented in this manner are currently only implicit.

To accommodate relationships between concepts explicitly, rather than implicitly (as we do now), we plan to extend our data-frame representations. This will involve developing data frames for verbs and adapting our framework to accommodate this substantial enhancement. We acknowledge that word-sense ambiguity will be more of an issue for verbally grounded data frames

than it has so far; luckily excellent resources have been developed to help specify subcategorization frames, valency, and co-occurrence restrictions.[3] In fact, we expect that our linguistically grounded ontology formalism is ideally suited to help provide insight into the lexical semantics of verb constructions.

By boosting the semantic content of our ontologies to involve clausal and predicative linguistic content, we will be creating more versatile knowledge sources. In addition to these "atomic" structures, we plan to define "molecular"-size ontology snippets that include a small number of interconnected and hence interrelated data frames. Where text is coherent or structure is consistently arranged, we will be able to better analyze and extract the associated content.

The nexus between syntactic structure and semantic content is well understood and has been implemented in various frameworks. Some view the connection as interpretive: syntactic content must first be recognized, and then semantic content is derived from it. Indeed, in prior research we have extracted semantic predicates from syntactically parsed clinical-trial documents [LTPE08]. Others view the association as unfolding in tandem: syntactic and semantic structures are incrementally derived synchronously. With relationship sets that are both grounded in syntactic structure and indicative of semantic correspondences, we should be in a position to explore the full spectrum of approaches to the syntax/semantics interface.

Related research would also include recognition of meaningful aggregates of atomic concepts. Detecting event structure in text, for example, would involve matching "molecular" snippets of related ontological content. Traditionally, natural language processing approaches to event detection involve extensive domain-specific and world knowledge, and our ontologies are capable of providing exactly this kind of information.

Further, although we have worked some with high-precision filtering [XE08], we have not yet applied our work to free-running text repositories, choosing instead to focus on data-rich documents. Extending extraction ontologies with data frames for relationships and "molecular"-size ontology snippets should help extend our range of coverage to more traditional text types, but we also believe that we will likely need to adapt techniques from natural language processing such as probabilistic approaches to parsing.

Synergistic Layout-Based/Ontology-Based Annotator

Based squarely on our previous NSF research on ontology-based annotation (IIS-0083127) and on FOCIH's layout-based annotation capabilities (IIS-0414644), we propose to build a synergistic annotator that capitalizes on the strengths of one annotation method to largely overcome the weaknesses of the other. We envision our proposed synergistic annotator working as follows:

- If we encounter a new page P we wish to annotate in a domain for which we have no extraction ontology, we use FOCIH to first let a user define, via a form, the information of interest and then fill in the form, by copy and paste, to annotate and gather the information from P. FOCIH's normal mode of operation is to generate a layout-based annotator for P and all the sibling pages of P from the same site.

- Behind the scenes, we create the structure of an extraction ontology from the given form, and we begin to linguistically ground the extraction ontology from the given information from P and all its sibling pages from the same site. We initialize lexicons with the data extracted from these pages. For example, from the country web pages we have experimented with [TEL09], we can obtain fairly complete lexicons for countries, country capital cities, religions, agricultural products, and so forth. Further, we can match data in P and its sibling pages with data-frame recognizers from a library of data frames for common items

[3] See, for example, http://verbs.colorado.edu/verb-index/.

of interest. If, for example, one of the data items extracted from P is a phone number, the system would associate the form field (equivalently, the OSM-EO object set) with the phone-number data frame. The result is a reasonably good extraction ontology, built from scratch with little user effort.

- If we now encounter a different page P' from a different site in the domain for which we have generated an initial extraction ontology, we first apply the extraction ontology to annotate the page. If P' is semi-structured, having regular patterns for data-element placement, the system can take as input the extraction-ontology's annotation and produce, in FOCIH-like fashion, a layout-based annotator for P'. (An interesting open research question here is how accurate the extraction ontology has to be to enable the layout-based annotator to be automatically generated. A human can be in the loop when the accuracy is too low to correct, with the FOCIH interface, any mistakes made by the extraction ontology, so that the generation of the layout-based annotator can always succeed.)

- The immediate use of the generated layout-based annotator is to annotate and extract information from P' more accurately and to also accurately harvest the information of interest from any and all sibling pages of P'. In addition, however, the system can also update the extraction ontology by augmenting lexicons and, perhaps, refining the selection of data-frames from the library. Thus, we can semi-automatically "grow" extraction ontologies, and as we use them in this synergistic fashion, they improve with use.

- Having learned by experience [TEL09] that sibling pages of an initial page are not as regular as expected, the synergistic annotator can play yet another role in solving this problem. First, it can help detect when the pattern breaks: i.e., when the layout-based annotator returns nothing for a field or returns garbage but the extraction ontology finds data for the field on the page. Second, it can rerun the pattern generator to discover an alternate pattern for the field. In general, the synergistic annotator can use this mode of operation to also detect and fix broken layout-based annotators (page wrappers) and thus help resolve the so-called wrapper maintenance problem [LMK03, MHL03].

Structured Data Annotation

Extracting from and annotating structured documents is akin to work in reverse engineering, which is a field of research unto itself. In previous work, most of it from NSF-sponsored research (IIS-0414644), we have developed reverse-engineering algorithms to OSM-like conceptual models that preserve information and constraints: i.e., for normalized, constraint-explicit relational databases [EX97], for XML-Schema documents [AKEL08, ME06], and for well structured sibling tables of the kind often found on generated pages of hidden web sites [TE09].

For our proposal here, we intend to convert these reverse-engineering algorithms into data-annotation algorithms for OSM-EO knowledge bundles (KBs). Thus, for example, we should be able to directly annotate XML documents, allowing us to to superimpose knowledge bundles over XML documents as well as ordinary web pages. Further, it should be clear from our discussion above about our proposed synergistic annotator that reverse-engineering structured data to OSM-EO allows us to constitute extraction ontologies for the application domain of the data. Then, as explained in [LED+09], we should be able to cut out meaningful subcomponents from any large application domain to constitute reusable extraction ontologies.

KB Information Organization

Much of what we have accomplished has been done as part of the work of master's theses and doctoral dissertations. We have begun to piece together these student projects into a unified

prototype, including a comprehensive integrated workbench that knowledge workers can exploit to specify, manipulate, and use ontologies for their own information needs. Given the diversity of project tools and their independent nature, integrating them together into a unified whole is and will continue to be challenging.

Tool integration involves not only WoK creation and usage tools but also KB organization and enhancement tools, which includes schema integration, record linkage, data cleaning, and semantic enrichment—each, by themselves, entire fields of research. Data integration within KBs can reduce uncertainty and mitigate the need for users to assimilate data from different sources manually. Data linkage among KBs can lead to opportunities for cross KB data mining and for serendipity by presenting researchers with (perhaps unexpected) connections among research studies. Data cleaning and semantic enrichment can lead to community accepted KBs useful for instruction and learning as well as question answering and fact finding. All of this is particularly true in the current era of global language resource interoperability, and we expect to be part of this emphasis in the technological landscape.

Although we have considerable experience with schema integration (e.g., [BE03, EJX01, EJX02, EXD04, XE06]) and some experience with semantic enrichment [LE09], we need not resolve all, or even any, of the issues in these areas of research to be successful. However, opportunities abound and we can provide some basic data integration within KBs, some basic data linkage among KBs, and some basic data cleaning and semantic enrichment in our organization and use of KBs.

7 Research Plan

Several related fields of research are at the heart of our work: information extraction [Sar08], information integration [ES07], ontology learning [Cim06], and data reverse engineering [Aik98]. The WoK- and KB-construction approach discussed here is a unique, synergistic blend of techniques resulting in tools to efficiently locate, extract, and organize information for research studies. (1) It supports directed, custom harvesting of high-precision technical information. (2) Its semi-automatic mode of operation largely shifts the burden for information harvesting to the machine. (3) Its interactive mode of operation allows research users to do their work without intrusive overhead. The KB tools for building a WoK are helpful assistants that "learn as they go" and "improve with experience."

We are not starting from scratch. We have already implemented several tools (as student project prototypes): (a) An OSM-EO editor lets users graphically create (O, R, C) conceptual models as Figure 1 shows, add (O, R, C ,I) instance data, and specify (O, R, C, I, L) linguistic grounding for object sets as Figure 2 shows; the result is OntoES (our Ontology-based Extraction System) [ECJ$^+$99]. (b) An OntoES Alerter for craigslist.org lets users specify a search that runs periodically and sends email when items satisfy the search criteria; Figure 5 is from this project. (c) TISP [TE09] interprets and harvests information from sibling tables from hidden web sites. (d) FOCIH [TEL09] lets users build ontologies via form specification and semi-automatically annotate a collection of pages from the same site; Figure 3 shows one of the interfaces for FOCIH. (e) AskOntos [Vic06] and SerFR [AME07] are free-form query processors; the free-form query in Figure 4 is an adaptation of these tools. (f) MapMerge [Lia08] allows users to merge populated OSM-O model instances; it runs semi-automatically or automatically based on schema-mapping algorithms [BE03, XE06]. Further, we have implemented several algorithms (also as student projects): algorithms to convert an XML-Schema specification to an OSM-O conceptual schema [AKEL08, ME06], algorithms to establish mappings from source-to-target schemas for schema integration [EJX01, EJX02, EXD04, XE06], and algorithms to semantically enrich semantic models for canonicalized tables [LE09]. In connection with our OWL/RDF-generated ontologies and tuple

stores, we have successfully used SWRL within Protegé [NSD+01] to write and execute deductive rules for (O, R, C, I, L, D) model instances. And, we have built a basic framework to execute queries over these rules and the extracted base data, which supports the A component of (O, R, C, I, L, D, A) model instances as Figure 4 shows.

We plan to carry out the proposed research according to the following timeline:

<table>
<tr><td></td><td>Graduate student 1
(Computational Linguistics)</td><td>Graduate student 2
(Computer Science)</td><td>Graduate student 3
(Computer Science)</td></tr>
<tr><td>1st academic year</td><td>Design grammar-based data frames for relationship sets and begin implementation.</td><td rowspan="2">Assemble, enhance, and integrate prior work on FOCIH (layout-based annotator) and OntoES (ontology-based annotator) and implement the synergistic annotator.</td><td rowspan="2">Assemble, enhance, design and build structured data annotators for XML, well-formed relational databases, and Wang tables [Wan96, EHLN06].</td></tr>
<tr><td>1st summer</td><td>Complete implementation, do initial testing, and make adjustments.</td></tr>
<tr><td>2nd academic year</td><td>Design and carry out experimental evaluation of data frames for relationship sets</td><td>Design and carry out experimental evaluation of the synergistic annotator.</td><td>Develop proofs for information/constraint-preserving transformations.</td></tr>
<tr><td>2nd summer</td><td colspan="3">Write up and publish results</td></tr>
<tr><td>3rd academic year</td><td>Extend “atomic” ontology snippets to “molecular” snippets and integrate results with other KB and WoK tools.</td><td colspan="2">Enhance and integrate developed KB and WoK tools.</td></tr>
<tr><td>3rd summer</td><td colspan="3">Test and evaluate KB construction tools and WoK performance; write up and publish results.</td></tr>
</table>

The three co-PI’s have worked together on topics related to this project for more than a decade. Going forward, Lonsdale will be specifically responsible for supervising the research of Graduate Student 1. Embley will have project director responsibilities and will specifically supervise the research of Graduate Students 2 and 3. Liddle has always been the chief implementation architect of our research efforts and will continue in this role.

For experimental evaluation, we plan to conduct experiments much as we have in the past. We will establish test sets for training, development, and blind tests and compute precision, recall, and F-measures to test implementations of our proposed grammar-based data frames for relationship sets and our proposed synergistic annotator. For the synergistic annotator, we can in addition check learning accuracy by checking for an increase in F-measure after automatically updating an extraction ontology. Test web pages will come from a broad range of applications: past applications have included government-published demographic statistics, items for sale (e.g., cars), apartment rentals, genealogy and family history, bio-research applications, clinical trials, and many more. For structured data annotation, as we have in the past, we can prove that our reverse-engineering algorithms preserve information and constraints. Finally, as we continue to build and add tools to our WoK workbench, we can field-test them. We can deploy prototype tools on the web and allow open alpha-testing. This testing should not only serve the purpose of allowing us to obtain and adjust according to received feedback, but should also provide for dissemination of our work to all who have interest in WoK-like tools.

The work we propose presents a grand vision—one that others share [BL07]. What is different and innovative, however, is that the proposed work shows a practical way to move toward this vision.

8 Significance

The intellectual merit and broader impact of the proposed work have the potential to make a significant and positive difference in how people interact with information on the web, itself the "great equalizer," enabling diverse populations across the country and throughout the world to connect and collaborate.

8.1 Intellectual Merit

Our research addresses the question of how to advance knowledge by direct use of the vast store of heterogeneous information available on the internet. The methods and tools we have developed and plan to develop involve the information extraction, integration, and access of several types of data across various application domains. Specifically, we intend to provide an answer to the question about how to turn lexical and syntactic symbols into semantic knowledge and ultimately into a web of knowledge. We also intend to provide a way for untrained users to directly query for facts in fact-filled knowledge bundles and to enable provenance trace-back to fact sources.

Our team of PIs/coPIs has complementary expertise in the underlying theory and technologies supporting this effort, and we have collaborated in various combinations for many years. Our previous NSF-funded efforts have helped develop the core of an infrastructure that has proven reliable and flexible. We follow up-to-date software engineering principles in the development of our code base. Our tools and knowledge representations are built following best-practice guidelines for data encoding, annotation, and exchange.

The research we have proposed here directly addresses difficult issues that will require experience, imagination, and vision. We are confident that what we propose to accomplish, while based on our prior work and hence somewhat evolutionary, is also taking our research into revolutionary new directions that we are eager to explore.

8.2 Broader Impact

Benefit to Education. In our research group we collaborate with a variety of undergraduate and graduate students—men, women, and both domestic and foreign students. The bulk of our request is to support this student team. To broaden our impact on students we teach several graduate courses (in three colleges) related to our work on web-based data extraction and integration (Embley), information architecture and web development (Liddle), and computational linguistics (Lonsdale). We plan to continue recruiting a diverse group of students and to continue mentoring students individually, in collaborative lab and implementation work, and in weekly group research meetings, as well as in the classroom.

Dissemination of Results. Our papers, presentations, and other artifacts appear on our project web site [DEG]. We continually publish our work in scientific conferences, workshops, and journals.

Benefit to Society. The internet and the web have become an integral and essential part of modern life. Looking forward, the vision for the semantic web is compelling and highly ambitious and could benefit humankind in a significant way—if it can be developed. Significant challenges remain, and we are poised to help in overcoming them. Ultimately, the semantic web will be populated directly with annotated information created purposefully by developers and publishers. But we cannot rely solely on hand-annotated sources because the ordinary web is too large to engineer into a completely annotated whole. Our proposal provides a practical way forward. With our method, ordinary web pages can become a part of the semantic web without the need for highly engineered, largely manual annotation processes.