Functional Modeling of Engineering Designs for the Semantic Web

Joseph B. Kopena and William C. Regli*
Geometric and Intelligent Computing Laboratory
Department of Computer Science, College of Engineering
Drexel University
http://gicl.cs.drexel.edu/

Abstract

A goal of this paper is to briefly describe work on a novel application of the Semantic Web to design repositories—collections of diverse engineering knowledge with sophisticated reasoning services. However, a more general goal of this paper is to demonstrate the extra utility enabled in database-styled applications by the incorporation of even simple semantics. It also discusses the development of practical, useful representations and applications for the Semantic Web.

1 Introduction

Engineers spend large portions of their time searching through corporate legacy data and catalogs searching for existing solutions which can be modified to solve new problems or to be assembled into a new device [14]. Corporations maintain huge collections of current and past product designs, each associated with a wide variety of media and information, including detailed geometry, tolerances, mechanical assembly models, finite element meshes, simulation output, and natural language documentation. In current practice such information is often underutilized due to the absence of adequate methods and tools to organize, manage, and work with such bodies of design knowledge. Browsing and navigating such collections are based on manually-constructed categorizations which are error prone, difficult to maintain, and often based on an insufficiently dense hierarchy. Search functionality is limited to inadequate keyword scanning or matching on overly simplistic attributes.

Design repositories [11, 13] are an evolution of traditional design databases. They aim to overcome current limitations through the application of knowledge representation techniques. Function, behavior, rationale, and other aspects of the designs are captured and reasoned on to enable search, categorization, and other tasks in support of the engineer, similar to case-based reasoning [5]. Figure 1 depicts this process.

Although many standards and assembly representations exist for capturing engineering artifact data, few are adequate for use in a design repository. Most, such as the prominent STEP AP203, capture only detailed geometric information, ignoring the wide range of information available and in particular not capturing abstract concepts such as function. In addition, most define only a shared syntax with human-interpretable semantics, unsuitable for automated reasoning beyond a syntactic level.

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

*Also of the Department of Mechanical Engineering; Email: regli@drexel.edu

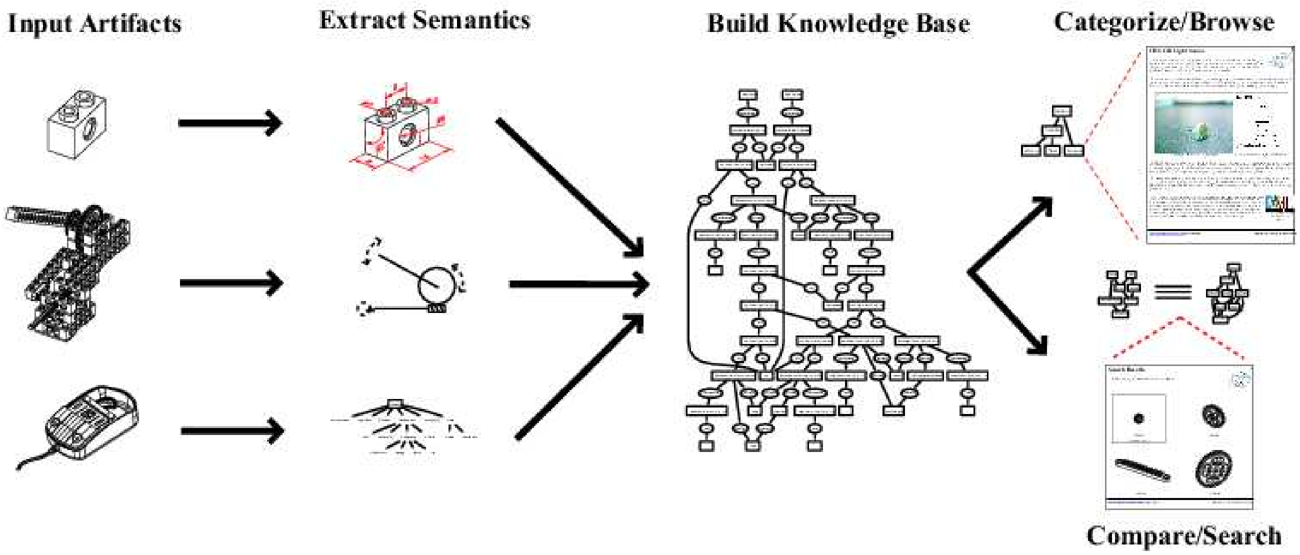


Figure 1: The design repository process.

In this work, description logics and the Semantic Web are used to develop a representation and set of reasoning mechanisms which can support a design repository. An ontology of electromechanical devices based on function and flow is used in representing, classifying, and comparing such devices. Description logic reasoning mechanisms are used to provide design repository inferences not possible with traditional database implementations. New methods of repository interaction, architecture, and applications are enabled by Semantic Web technologies, including uses outside engineering design.

This paper is organized as follows: Section 2 describes the representation developed in this work. Reasoning tasks required of a repository are demonstrated using this representation in Section 3. The role of the Semantic Web in this work is discussed in Section 4. Finally, Section 5 presents some closing remarks.

2 A Representation Based on Function

In order to tackle the many abstract concepts and underlying structures involved, effective reasoning on assemblies mandates knowledge-based mechanisms. Assemblies are primarily defined by their intended *function*—goals achieved and tasks performed—and the decomposition of that function into subfunctions. These in turn define form, structure, and behavior. It is therefore the representation and reasoning of function to which this work has been initially scoped. Reasoning at this abstract, design rationale-oriented level is best accomplished through the application of knowledge representation and reasoning techniques. Doing so under a framework with clear semantics, desirable for automated reasoning, entails formal logic.

Applying knowledge representation to repository reasoning requires an *ontology* for assemblies, a formal description of a conceptualization of this domain. It has been assumed in this work, in the absence of evidence to the contrary, that a great deal of repository-oriented reasoning can be accomplished within first-order logic. However, an ontology in this domain with strong semantics defined in first order logic faces several issues:

- A quick study of the problem and previous efforts makes it clear that it would be difficult to construct. This observation is discussed further within the context of the Semantic Web in Section 5.
- It might be unnecessary, given the problems at hand. The focus on this work is not on solving difficult integration issues, or analysis tasks such as simulation and verification. Rather, repository operations

such as search and compare only require that mechanisms be represented in a manner strong enough to differentiate designs and capture basic properties of a device.

- Although more favorable than in higher order logics, reasoning in first order logic is still not advantageous in that it is NP-Complete and only semi-decidable. In addition, it seems plausible that the complexity of a strong ontology in this domain might overcome practical capabilities of current first order logic reasoners. These points are important in light of the potentially large volumes of designs and knowledge that a repository might work with, especially when tasked with organizing and maintaining legacy data.

Each of these issues can be addressed by utilizing a subset of first order logic. This work has studied the use of description logics; in particular the logic $\mathcal{ALEN}$[1]. Their lower expressivity limits the ability to define detailed semantics for a domain. However, in return they make several positive tradeoffs, including desirable computability and tractability results [1]. Within the repository application, this loss of expressiveness is acceptable so long as enough design knowledge can be captured to enable necessary operations, such as matching devices against adequately sophisticated searches and comparing mechanisms.

This work employs a representation based on function and *flow*—the materials, energies, and signals on which functions operate. Heavily based on current functional modeling research within engineering design theory [12, 7, 15], this effort differs in that the representation is placed within a formal framework, description logic, enabling automated reasoning. The representation is broken up into two parts: a core ontology, defining the basic structure of a function-based mechanism description language, and vocabulary extensions, providing terminologies with which rich descriptions can be written in the language.

The core ontology defines the universe of objects as consisting of assemblies, components, functions, and flows. Several relations are also given, for example to associate assemblies with functions and functions with their input and output flows. The vocabulary extensions provide taxonomies of functions and flows derived from those presented in [12] and [7], which were developed from large surveys of engineers. Figure 2 illustrates this representation with a description of a CDS Cell, a common sensor shown in Figure 2(a). A graphical form of the description is given in Figure 2(b) as a *function and flow diagram*, similar to those presented in [12] but not identical. This diagram is interpretable as statements in a description logic model as in Figure 2(c).

This representation does not rigorously and unambiguously capture the semantics of mechanisms. Instead, it provides a language expressive enough to describe and distinguish devices while maintaining efficiency and computability. It is neither so formal as to prevent practical computing, nor so informal as to prohibit automated reasoning. The following section outlines the use of such reasoning in a design repository.

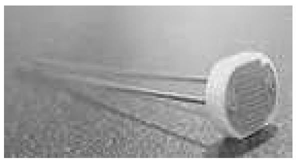
3 Reasoning for Design Repositories

Standard description logic inferences can be used in a number of repository reasoning tasks. Figure 3 demonstrates the most obvious of these, *classification*. In addition to being interpretable as statements in a description logic model, the function and flow diagrams presented in this paper may also be interpreted as concept descriptions. In this way, they may be used to manually create a categorization against which devices in the repository are sorted. Figure 3(a) shows such a hierarchy for some simple sensors and effectors. In this example, the CDS-Cell from Figure 2 has been classified as a light sensor because it matches the definition of the class, given in Figure 3(b)[2]. Such reasoning can greatly improve current, slow, error prone, and limited manual classification.

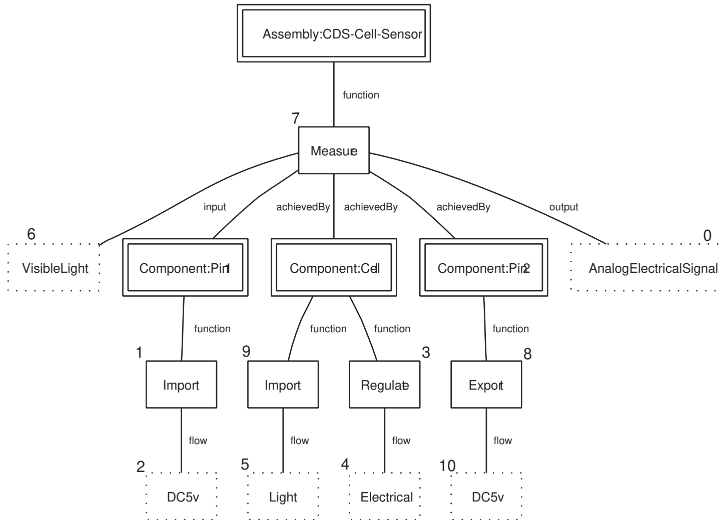
Search functionality may be similarly implemented, treating the query as a concept description and classifying device instances against that definition. Description logic *subsumption* may be used to manually construct a categorization in a bottom-up, generally more intuitive fashion. Less standard inferences may also be applied, for example induction of the *least common subsumer* of sets of classes [9] may be used to automatically create

[1]A good reference for description logic expressivity notations is [1].

[2]In the DL language $\mathcal{EL}$.



(a) Typical CDS Cell.



(b) Simplified function and flow diagram (numbering on anonymous nodes corresponds to skolemization scheme used in Figure 2(c)).

⟨ ⟨ *CDS-Cell-Sensor type Assembly* ⟩	⟨ *Cell function node3* ⟩	⟨ *node7 type Measure* ⟩	⟨ *Pin2 type Component* ⟩	⟨ *node8 flow node10* ⟩
⟨ *CDS-Cell-Sensor function node7* ⟩	⟨ *node2 type DC5v* ⟩	⟨ *node1 type Import* ⟩	⟨ *node9 type Import* ⟩	⟨ *node8 type Export* ⟩
⟨ *Pin2 function node8* ⟩	⟨ *node5 type Light* ⟩	⟨ *node7 input node6* ⟩	⟨ *node4 type Electrical* ⟩	⟨ *node7 output node0* ⟩
⟨ *node0 type AnalogElectricalSignal* ⟩	⟨ *node6 type VisibleLight* ⟩	⟨ *Pin1 type Component* ⟩	⟨ *Cell function node9* ⟩	⟨ *Pin1 function node1* ⟩
⟨ *node1 flow node2* ⟩	⟨ *node7 achievedBy Cell* ⟩	⟨ *Cell type Component* ⟩	⟨ *node7 achievedBy Pin2* ⟩	⟨ *node10 type DC5v* ⟩
⟨ *node3 flow node4* ⟩	⟨ *node3 type Regulate* ⟩	⟨ *node7 achievedBy Pin1* ⟩	⟨ *node9 flow node5* ⟩	⟩

(c) Figure 2(b) in equivalent description-logic model form.

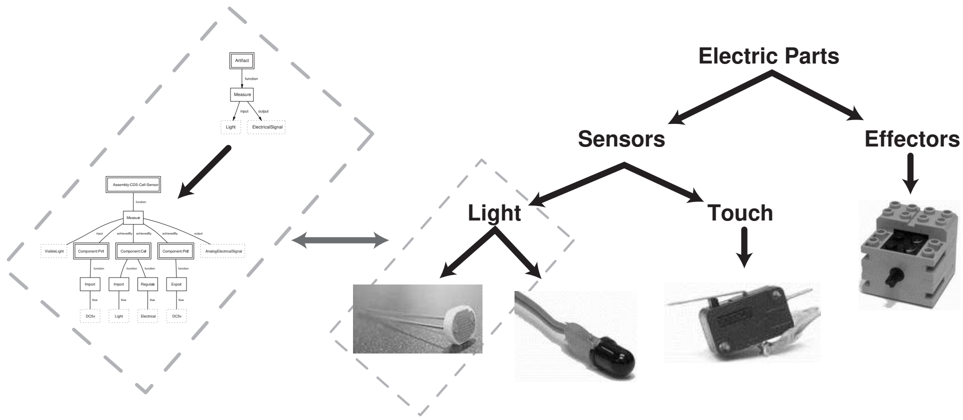
Figure 2: Function and flow modeling of a Cadmium Sulfide (CDS) Cell, a common photoresistor.

a categorization from a collection of devices. This has the potential to be a powerful information management tool once techniques have been developed to address several issues, e.g. the density of the generated hierarchy.

The capacity of the underlying reasoning mechanisms to operate effectively in a system the size of a full-sized design repository is a significant concern. As noted above, this is a primary motivation in using description logics. A formal logic is necessary for defining and reasoning with the semantics of a devices at an abstract, knowledge-based level. Employing a formal logic as the representation framework enables:

- The ability to infer implicit knowledge and data.
- Proofs or characterizations of a reasoning algorithm's soundness and completeness properties.
- Well-defined mechanisms for manipulating knowledge, e.g. deriving the least common subsumer.
- Improved integration and interoperation between heterogenous systems.

Description logics strike a favorable balance between providing a level of expressiveness suitable for these tasks and maintaining efficient, scalable reasoning [2]. Algorithms and implementations do exist for these and other logics which support effective, large-scale reasoning, e.g. [6, 8, 4, 10, 3]. Scalability and practical usability is a legitimate major concern in any effort operating on data at the scale of a design repository. However, this does not necessarily prohibit the use of knowledge-based techniques.



(a) Example part hierarchy with function and flow diagram for the class of light sensors.

$$LightSensor \equiv Artifact \sqcap \exists function.[Measure \sqcap \exists input.Light \sqcap \exists output.ElectricalSignal].$$

(b) Definition of class of light sensors from diagram in Figure 3(a).

Figure 3: Reasoning example: classifying against a hierarchy.

4 Semantic Web

The focus in this work is mostly on the repository reasoning tasks enabled by using a representation grounded in description logic. However, Semantic Web technology has been used in trial implementations of this system, in particular the W3C Ontology Web Language (OWL)[3]. OWL is a description logic language which may be embedded in Web content. Figure 4 provides examples of the two primary uses of OWL in this work: defining ontologies as in Figure 4(a), and encoding the representation for a particular device as in Figure 4(b). The underlying description logic of OWL contains more expressivity than appears to be necessary for this work.

The Semantic Web introduces several new capabilities to the repository setting. One is that unlike existing centralized database architectures, a Semantic Web-based repository may keep its data distributed across the Internet or an intranet. Only device markup and links to the original sources and associated data must be stored by the repository service. More interestingly, the ability to markup device-related Web content with the representation outlined in this paper makes it straightforward to post a wide variety of data—text, CAD, images, simulations—and publish it to any number of repositories and other reasoning engines. Within engineering design, this makes it easier for loosely coupled design teams to collaborate, for example by avoiding redundant design efforts. Another application is for part catalogs and similar materials to be marked up on corporate websites. That information may then be aggregated by a portal site providing an interface to many vendors. With a repository running at the core of the site, large amounts of data may be collected, organized, and utilized in more robust and sophisticated ways than under current practice.

Such a portal site also has applications outside of formal engineering design. With appropriate interfaces, the same techniques could be used to provide portals for e-commerce users shopping for household appliances, electronics, and other purchases. Repositories could support hobbyists in pooling a community's knowledge. They could also be applied in educational settings, for example as a medium for design students to exchange ideas, as well as to search for information online.

[3] `http://www.w3.org/2001/sw/WebOnt/`

```
<owl:Class rdf:about="&eng;#Artifact" />
<owl:Class rdf:about="&eng;#Function" />
<owl:Class rdf:about="&eng;#Flow" />

<owl:Class rdf:about="&eng;#Assembly">
  <rdfs:subClassOf rdf:resource="&eng;#Artifact" />
</owl:Class>

<rdf:Property rdf:about="#function">
  <rdfs:domain rdf:resource="#Artifact" />
  <rdfs:range rdf:resource="#Function" />
</rdf:Property>

        ...

<owl:Class rdf:about="&eng;#Function">
  <owl:disjointWith rdf:resource="&eng;#Flow" />
  <owl:disjointWith rdf:resource="&eng;#Artifact" />
</owl:Class>

        ...

<owl:Class rdf:about="&flow;#AnalogElectricalSignal">
  <rdfs:subClassOf rdf:resource="&flow;#Electrical" />
  <rdfs:subClassOf rdf:resource="&flow;#Signal" />
</owl:Class>
```

(a) Snippets from core ontology along with function and flow vocabulary extensions.

```
<eng:Assembly rdf:about="#CDSCellSensor">
  <eng:function>
    <func:Measure>
      <eng:input><flow:VisibleLight /></eng:input>

      <eng:achievedBy>
        <eng:Component rdf:about="#Pin1">
          <eng:function>
            <func:Import>
            <eng:flow>
              <flow:DC5v />
            </eng:flow>
            </func:Import>
          </eng:function>
        </eng:Component>
      </eng:achievedBy>

      ...

      <eng:output>
        <flow:AnalogElectricalSignal />
      </eng:output>
    </func:Measure>
  </eng:function>
</eng:Assembly>
```

(b) Portion of CDS Cell function and flow representation as given in Figure 2.

Figure 4: Ontologies and data in OWL form.

5 Conclusions

This paper has briefly introduced work on applying description logics to constructing design repositories. An ontology of devices based on function and flow has been presented along with an outline of the use of several associated reasoning mechanisms in supporting such an application. The authors feel this style of ontology is one that could serve the Semantic Web well—a light, easy to author form which provides enough reasoning abilites to be useful without incurring the costs of more sophisticated ontology. For at least the near future, more complex ontologies will be difficult to develop, stalling successful deployment and adoption, will overwhelm potential users and limit the utility of the Semantic Web, and may potentially burden automated reasoners. This paper has also briefly discussed applications of this work to the Semantic Web. It is the authors' hope that this technology will develop into a mature, usable application in both the engineering design and home-use domains and provide an example of new capabilities enabled by the Semantic Web.

Acknowledgements: This work supported in part by National Science Foundation (NSF) Knowledge & Distributed Intelligence (KDI) Grant CISE/IIS-9873005 and CAREER Award CISE/IIS-9733545 as well as Office of Naval Research (ONR) Grant N00014-01-1-0618. All opinions, findings, and conclusions expressed in this material are those of the author(s) and not necessarily those of the supporting organizations.

References

[1] F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi, and P.F. Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge University Press, 2002.

[2] Alex Borgida. On the relative expressiveness of description logics and predicate logics. *Artificial Intelligence*, 82(1-2):353–367, 1996.

[3] Paul R. Cohen, Robert Schrag, Eric K. Jones, Adam Pease, Albert Lin, Barbara Starr, David Gunning, and Murray Burke. The DARPA high-performance knowledge bases project. *AI Magazine*, 19(4):25–49, 1998.

[4] M.P. Evett, J.A. Hendler, and L. Spector. Parallel knowledge representation on the connection machine. *Journal of Parallel and Distributed Computing*, 22:168–184, 1994.

[5] James E. Fowler. Variant design for mechanical artifacts: A state-of-the-art survey. *Engineering with Computers*, 12:1–15, 1996.

[6] Volker Haarslev and Ralf Möller. High performance reasoning with very large knowledge bases: A practical case study. In B. Nedel, editor, *Proceedings of Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01)*, August 4–10, Seattle, Washington 2001. pgs. 161–166.

[7] J. Hirtz, R. Stone, D. McAdams, Szykman S, and K. Wood. Evolving a functional basis for engineering design. In *ASME Design Engineering Technical Conferences, 13th conference on Design Theory and Methodology*, New York, NY, USA, September 9-11, Pittsburgh, PA 2001. ASME, ASME Press. DETC01/DTM-21688.

[8] Ian Horrocks, Ulrike Sattler, and Stephan Tobies. Practical reasoning for expressive description logics. In Harald Ganzinger, David McAllester, and Andrei Voronkov, editors, *Proceedings of the 6th International Conference on Logic for Programming and Automated Reasoning (LPAR'99)*, number 1705, pages 161–180. Springer-Verlag, 1999.

[9] Ralf Küsters and Ralf Molitor. Computing least common subsumers in $\mathcal{ALEN}$. In Bernhard Nebel, editor, *Seventeenth International Joint Conference on Artificial Intelligence*, pages 219–224. Morgan Kaufmann, 2001.

[10] D. B. Lenat. Cyc: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):168–184, 1995.

[11] S. Szykman, C. Bochenek, J. W. Racz, J. Senfaute, and R. D. Sriram. Design repositories: Engineering design's new knowledge base. *IEEE Intelligent Systems*, 15(3):48–55, May/June 2000.

[12] S. Szykman, J. W. Racz, and R. D. Sriram. The representation of function in computer-based design. In *ASME Design Engineering Technical Conferences, 11th International Conference on Design Theory and Methodology*, New York, NY, USA, September 12-16, Las Vegas, NV 1999. ASME, ASME Press. DETC99/DTM-8742.

[13] Simon Szykman, Ram D. Sriram, and William C. Regli. The role of knowledge in next-generation product development systems. *ASME Transactions, the Journal of Computer and Information Science in Engineering*, 1(1):3–11, March 2001.

[14] David G. Ullman. *The Mechanical Design Process*. McGraw-Hill, Inc., 1997. ISBN 0-07-065756-4.

[15] William H. Wood and Manish Verma. Functional modeling: Toward a common language for design and reverse engineering. In *ASME Design Engineering Technical Conferences, 15th International Conference on Design Theory and Methodology*, New York, NY, USA, September 2-6, Chicago, Illinois 2003. ASME, ASME Press. DETC03/DTM-48660.