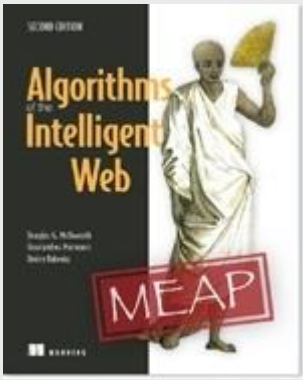

The need for classification

By Douglas G. McIlwraith, Haralambos Marmanis, and Dmitry Babenko

In this article, excerpted from the book Algorithms of the Intelligent Web, we discuss classifications systems and their value for your applications.

Whether we realize it or not, we encounter classification on a daily basis. In our everyday experiences, we can list the food items on a restaurant's menu, which are classified according to menu categories—salads, appetizers, specialties, pastas, seafood, and so on. Similarly, the articles in a newspaper are classified based on their subject—politics, sports, business, world, entertainment, and so on. So we see, classification is abound in our daily lives.

The books in a library carry a *call number*, which consists of two numbers: the *Dewey classification number* and the *Cutter number*. The top categories of that system are things such as generalities, religion, natural science and mathematics, and so forth. The Library of Congress in the United States has its own classification system that was first developed in the late nineteenth and early twentieth centuries to organize and arrange its book collections.

Over the course of the twentieth century, the Library of Congress system was adopted for use by other libraries as well, especially large academic libraries in the United States. We mention two systems of classifying books because the Library of Congress classification system isn't strictly hierarchical as the Dewey classification system is, where the hierarchical relationships between the topics are reflected in the numbers of the classification.

In medicine, a plethora of classification systems are used to diagnose injuries or diseases. For example, radiologists and orthopedic surgeons use the Schatzker classification system to classify tibial plateau fractures (a complex knee injury). Similarly, there are classification systems for spinal cord injuries; for coma, concussion, and traumatic brain injuries; and so on. You must have heard of the term *Homo sapiens*, of which *Homo* is our genus and *sapiens* is our species. This classification can, and typically is, extended to include other attributes such as *family*, *order*, *class*, *phylum*, and so forth.

Generally speaking, the more attributes you use, the finer the degree of classification is going to be. Having a "large" number of attributes is usually a good thing, but there are

caveats to this general principle. One notorious symptom of dealing with many attributes is the *curse of dimensionality*.

The curse of dimensionality refers to the fact that our space becomes more and more homogenous as the number of attributes increases. In other words, the distance between any two points will be roughly the same no matter which points you select and what metric you apply to measure the distances. If that's the case, it becomes increasingly difficult to distinguish which category is "closer" to a given data point, since no matter where you "stand" in your space, everything seems to be the same distance apart!

In general, *flat reference structures* aren't as "rich" as *hierarchical reference structures*. In turn, the hierarchical reference structures are less rich than those that are hierarchical *and* semantically enriched. A reference structure that is semantically enriched falls under the study of ontologies: taxonomies of meaning (usually) hand created in order to encode domain specific knowledge. For the purposes of this article, an ontology consists of three things: *concepts*, *instances*, and *attributes*.

In figure 1, we depict a minute segment of a (rudimentary) general ontology by focusing on the classification of a "vehicle." Concepts are depicted as ellipses, instances are depicted as rectangles, and attributes are depicted as rounded rectangles. Note the hereditary property of attribute assignment. If attribute 1 is assigned to the root of the concept tree then it cascades to the concept leaf nodes. Thus, values for attribute 1 can be assigned to instances of a boat and an automobile. Only an automobile instance can have values for attribute 2. Attribute 1 could be the attribute Name, which for practical reasons you always want, whereas attribute 2 could be the attribute number of wheels. Attributes are defined at the level of the concepts, but only instances have concrete and unique values because only instances represent real "things."

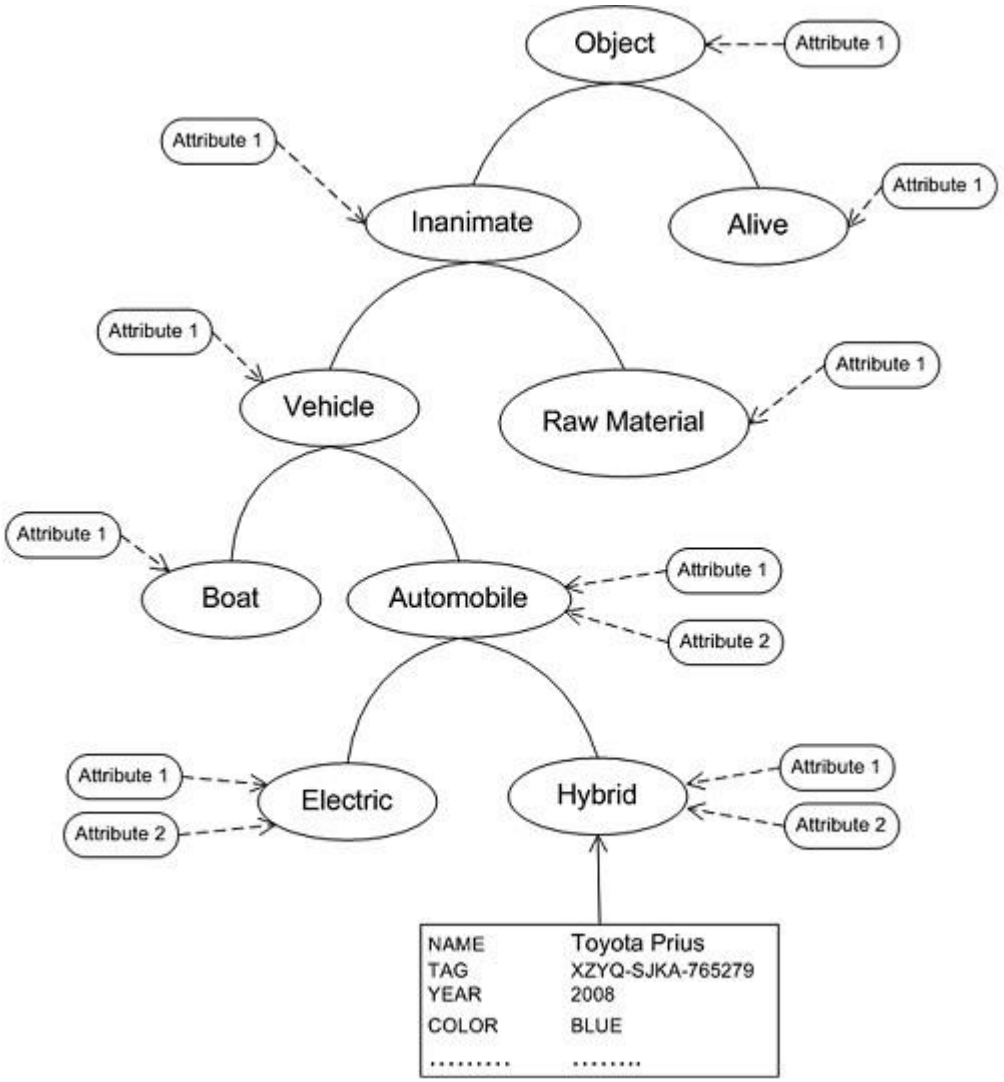

Figure 1 An example that depicts the basic elements of a reference structure (a rudimentary ontology). Ellipses denote concepts while rounded circles denote attributes. Rectangles denote instances. Concepts inherit attributes from above.

Think of concepts as analogous to classes in your favorite object-oriented language. Instances are analogous to concrete instances of a class and attributes are the variables within these class instances. Clearly, a source code base that uses packages to group together

classes by functionality or component, that uses inheritance to abstract common structure and behavior, and that properly uses encapsulation, is superior to a source code base that doesn't have these qualities. In object oriented programming when you define a class, you define the data type of the variables but you don't assign a value to a variable (unless it's a constant). This parallel holds true here; concepts inherit attributes from the concepts above themselves in the tree, and do not become instances until their attributes have been assigned. This is a good working definition that'll serve you well 80% to 90% of the time. If you're ever in doubt, you can consult this analogy to obtain some insight into your structure.

We could obviously go on with more classification systems; they're everywhere. The point is that classifying data is equivalent to organizing it. Classification systems improve communication by reducing errors due to ambiguities. They also help us organize our thoughts and plan our actions. The reference structure, which is used for organizing our data, can be as simple as a set of labels or as advanced as a *semantic ontology*. Have you heard of the *semantic web*? At the heart of the semantic web (see Antoniou and van Harmelen) lie a number of technologies and formal specifications related to creating, using, and maintaining semantic ontologies. Ontologies are also useful in model-driven architectures (see (Gasevic n.d.)), which is a software design initiative of the Object Management Group (OMG) (Group n.d.).

Look at your database. Your application could be an online store, an intranet document management system, an Internet mashup, or any other kind of web application. When you think about your data and the ways it could be organized, you'll realize the value of a classification system for your application.