Context-based Naming in Semantic Networks
Manos Theodonakis

Department of Computer Science, University of Crete and
Institute of Computer Science, FORTH
email: theodor@ics.forth.gr

Abstract

Semantic data models and semantic networks use logical names for the external identification of objects. Yet the naming schemes they employ are not “natural” enough and several problems often arise: logical names can be ambiguous excessively long and difficult to handle, unrelated to the environment of the named object, or unable to follow changes of that environment. In natural language similar problems are resolved by the context within which words (i.e., “names”) are used. This paper focuses on developing a context mechanism for a semantic network - structured information base and, in particular, exploiting this mechanism for naming purposes. This context-based naming mechanism enhances name readability, resolves ambiguities, saves a lot of redundant name substrings, localizes and thus facilitates consistency checking, query processing and update operations.

1 Introduction

Semantic data models and semantic networks provide a “natural” mechanism (that is, similar to the way a user or a system designer views an application) for specifying the design of a database [13]. In natural language names are used to identify concepts (specific or abstract) and context mechanisms to make the language frugal and concise, thus actually enhancing the expressiveness of a finite vocabulary. Although semantic data models also use logical names in order to identify externally the objects, they do not support context mechanisms. Offering contexts in semantic networks (and in data models in general) is essential, in order for their contents to be more understandable and more expressive and their management to be simpler, more flexible, and more effective by both the designers and the users.

A general approach introducing a notion of context in an information base is to provide structuring mechanisms for decomposing it into possibly overlapping parts. This paper focuses on developing a context mechanism for a semantic network - structured information base and, in particular, exploiting this mechanism for naming purposes. Our work is motivated by the need to provide a naming mechanism closer to the real world perception of the user or the designer of the information base.

Semantic data models allow the database designer to represent the objects of interest and their relationships in a way that resembles quite closely the view the user has of these objects and relationships. The objects are identified to the user by logical names. These names are usually composed from the names of the real world concepts that these objects represent. This produces a variety of problems: logical names can be ambiguous (synonyms, homonyms), excessively long and difficult to handle, unrelated to the environment of the named object, or unable to follow changes of that environment.

In order to overcome these problems we introduce a notion of context and an object naming mechanism which is based on context. The basic idea of this mechanism is that the object names are given with respect to a specific context. The global name\(^1\) of an object, which identifies this object over the entire information base, is generated automatically taking its context into account. Thus, different objects may have the same name in different contexts (homonyms). In order to generate these names the usual conceptual modeling mechanisms supported by many semantic data models and semantic networks, including classification, generalization and aggregation [10, 12], are taken into account.

This paper proposes a generic naming mechanism for semantic networks based on the concept of context. Because of the three basic structural mechanisms (classification, generalization, attribution), supported by the majority of semantic data models and semantic networks, are caring different semantics, this mechanism is specialized for each of them. The features of proposed mechanism include: definition of the object context, automatic object name generation taking into account its context query operations.

In sections 2, we refer to some crucial areas relating to context and naming. In section 3, we introduce a context-based naming mechanism for semantic networks and in the next section we present its benefits.

\(^1\)For us there are no global names. All the names are relative to a specific context. Whenever we use the terms global or absolute for a name we mean that this name is global in the context of the particular database.
2 Review of naming and context

There are several manifestations of context-based naming: In programming languages, the parts of the program which are visible to a particular program segment are determined by the scopes and scope rules using scope resolution operators. In traditional databases, views present a consistent partition of the database [9]. Such mechanisms have been adopted in object-oriented databases [13] and semantic data models [8]. In addition, database integration has to deal with naming conflicts of two types, homonyms and synonyms, because the global schema of the integrated database is usually generated by merging one or more user-oriented schemas [1, 2].

Moreover, contexts or perspectives have been proposed as a partitioning scheme of hypertext databases [7]. More recently, mechanisms for partitioning information bases with contexts have been proposed dealing with naming conventions, authorization, transaction execution and overlapping contexts [11]. Other approaches employ context as a way to face the complexity of information base update [4, 17].

Naming and addressing issues appear in networks, in the sense that the address space is divided into domains, subdomains, and so on, and domains never overlap and subdomains are always strictly nested within domains [14, 16], yielding a tree of domains and subdomains. Similar organizational hierarchies of contexts appear in UNIX file systems and WWW addresses.

3 Context and naming in semantic networks

3.1 Framework

In a semantic network, real-world knowledge is represented as a graph formed of data items connected by edges [10]. For our purposes, a semantic network consists of a finite set of nodes and a set of directed edges. The nodes and the edges are objects. An edge is a binary relation from one object, which is called domain object, to another object, which is called range object of this relation. In the following we call the binary relation simply relation. According to the most common conceptual modeling mechanisms there are three kinds of relations: instance-of, isa, attribute.

An object is a concept which can stand for anything. It usually consists of system-generated, globally unique identifier, completely independent of any physical location (called surrogate in the literature [5, 8]), which distinguishes internally it from all the other objects, and an atomic name (sequence of characters) which identify externally the object.

Context, in semantic networks, is a decomposition of the semantic network which includes two components: the contents of the context, which are objects (nodes and relations) and its lexicon, the relative names of the contents of the context with respect to it. In the following, we define a context-based naming mechanism for nodes, which is called name scope mechanism.

3.2 Name scope mechanism

The basic characteristics of this mechanism are listed below:

- **Context.** Contexts are objects which are identical with the nodes of the semantic network. The contents of a context are nodes.
- **Scope relation.** The contents of a context are determined by a new defined binary relation, called scope relation. This relation is declared from one node to another and denotes that the former is in the contexts (we say in the scope) of the context which is identical with the latter node.
- **Unique scope context assumption.** Every node may belong to several contexts. Yet, we assume that one of these contexts can always be selected as more characteristic in a specific network. Thus a node may belong to one context. We then say the node lies in the scope of that context.
- **Recursive context definition.** A context may belong to the contents of another context, which is called parent context and so on. This recursive context definition should contain no cycles.
- **Individuals.** There are nodes that they do not belong to any context. These are called individuals.
- **Node/context name.** A name of a node is generated by combining the atomic name of that node and recursively one or more of the atomic names of its parent contexts (if any).
- **Global names.** The global name of a node is the name of this node which is generated by all of its parent contexts. The global name is unique. Individuals belongs to no context and their atomic names are also global names.
- **Relative names with respect to a context.** Relative name with respect to a specific context is the name of the node, which is generated by all of its parent contexts, until this specific context.
- **Unique name context assumption.** The atomic name of a node belonging to a context is unique among the atomic names of the contents of this context.
There are three different kinds of scope mechanism (according to the usual conceptual modeling mechanisms): classification, generalization, attribution.

**Classification scope mechanism:** According to this mechanism a node may be in the scope of one of its classes (the scope relation is also an instance-of relation). Figure 1 illustrates an example of this mechanism. The links represent instance-of relations and the bold links scope relation. Thus, the node of the left side with atomic name 'Yannis' is classified under the classes 'Dentist' and 'Person', and it is in the scope of 'Dentist' context, because it is the most characteristic for this node. On the other hand, the node on the right side with atomic name 'Yannis' is classified under the classes 'Person', 'Programmer' and 'Karateca', and it is in the scope of 'Programmer' context. The name of a node according to this mechanism is generated by putting in order the atomic name of the node and a name of its context. These names are separated by the symbol ‘. [cls] .’ which denotes that the name has generated using the classification scope mechanism. Thus, the node 'Yannis' on the left side has name 'Yannis . [cls] . Dentist'. This is also the global name of the object which represents the dentist called Yannis. The relative name of the same object with respect to the 'Dentist' context is 'Yannis'.

![Figure 1](image1.png)

**Generalization scope mechanism:** According to this mechanism a node may be in the scope of one of its superclasses (the scope relation is also an isa relation). The name of the nodes are generated simultaneously with the classification scope mechanism, except that the separated symbol is the ‘. [isa] .’ The figure 2 shows an isa hierarchy representing weapons. The filled arrows represent scope relations. For example, the node 'attacking' is in the scope of 'weapon' and represents the attacking weapons (its name is 'attacking . [isa] . weapon').

![Figure 2](image2.png)

**Attribution scope mechanism:** In attribution mechanism the attributes are usually classified under classes of attributes (called categories). For example, in figure 3 the attribute between 'size' and 'string' is classified under the 'is_variable_of' category. According to this mechanism a node may be in the scope of another one (these nodes are connected with an attribute relation), via one category. For example, the node 'get' is in the scope of 'string' context via the category 'is_method_of' and represent the method of the string class called get. This is allowed only if the category is a function or a kind of function (surjective, injective, bijective). The name of a node according to this mechanism is generated by putting in order the atomic name of the node the name of the category involving the scope and the name of its context. For example, the node 'get' has global name 'get . [att] . method . string'. The capability of declaring relations as functions to define a context addresses to nodes without any atomic name. For example, in figure 4, the relation 'creation' is an inverse bijection function because every painting creation is dedicated to this painting. Thus, every specific creation can be unnamed.

![Figure 3](image3.png)

**Figure 4**

4 **Discussion**

Our approach more closely resembles the naming in natural language: contexts are supported, objects can be identified externally by using names with respect to a context (relative names). This results in the use of polysemous and homonyms names which is common in natural language. The names are generated dynamically by taking into account their context. Object identifying techniques like extending a name in a specific context or restricting the context of a specific name are also supported. All these contribute some quality factors in modeling like simplicity and flexibility. The capability of using relative names addresses the atomic names to be limited (also empty - unnamed objects) and saves a lot of redundant name substring. The global names, on the other hand, are composite and they can be as long as we need. These names identify uniquely the objects of an information base.

The objects are identified externally by names. These names are declared dynamically. This means that names can be as long as we need to identify uniquely the desired object. Moreover, names are consistent with the environment. This enhances the visibility and readability of the names and make
them more understandable to the users. The proposed mechanism localizes and thus facilitates consistency checking (because of the information base decomposition), query processing (query by using relative or global names) and update operations (efficient data entry by defining recursively the contents of a context, names are following the updates of the information base).

The unnamed objects, which are supported by the attribution scope mechanism, can be exploited by higher level update operations which facilitate the interactive creation of composite objects [4, 17]. In addition, the context-based naming mechanism can be exploited for developing hypermedia applications over information bases in order to generate sentences more closely to the natural language [3].

The disadvantage of this mechanism is that we need extra time to compute the name of an object or to access an object identified by a global name.

5 Conclusions and future work

The objective of this paper was to develop a context mechanism for semantic networks and exploiting this mechanism for naming purposes. As a solution of this problem, we proposed an context-based naming mechanism which is based on a tree context hierarchy and enhances name readability, resolves ambiguities, saves a lot of redundant name substrings, localizes and thus facilitates consistency checking, query processing and update operations. Future research direction include dealing with overlapping contexts, synonyms (different atomic names with respect to different context for the same object) and intercontext communication. Moreover, the unique scope context assumption can be relaxed to allow a node being in the scope of several contexts, all of which together identify the node in a unique way.

References