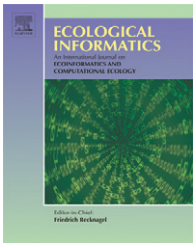
available at www.sciencedirect.com

ScienceDirect

www.elsevier.com/locate/ecolinf



Database design for ecologists: Composing core entities with observations

Anne C.S. McIntosh, Judith B. Cushing, Nalini M. Nadkarni, Lee Zeman*

The Evergreen State College, 2700 Evergreen Pkwy NW, Olympia, WA, 98505, USA

ARTICLE INFO

Article history:
Received 1 January 2007
Received in revised form
11 June 2007
Accepted 30 July 2007

Keywords:
Data visualization
Domain-specific conceptual structures
Ecoinformatics
End-user programming
Forest canopy
Semantic data integration
Database design

ABSTRACT

The ecoinformatics community recognizes that ecological synthesis across studies, space, and time will require new informatics tools and infrastructure. Recent advances have been encouraging, but many problems still face ecologists who manage their own datasets, prepare data for archiving, and search data stores for synthetic research. In this paper, we describe how work by the Canopy Database Project (CDP) might enable use of database technology by field ecologists: increasing the quality of database design, improving data validation, and providing structural and semantic metadata — all of which might improve the quality of data archives and thereby help drive ecological synthesis.

The CDP has experimented with conceptual components for database design, *templates*, to address information technology issues facing ecologists. Templates represent forest structures and observational measurements on these structures. Using our software, researchers select templates to represent their study's data and can generate normalized relational databases. Information hidden in those databases is used by ancillary tools, including data intake forms and simple data validation, data visualization, and metadata export. The primary question we address in this paper is, which templates are the right templates.

We argue for defining simple templates (with relatively few attributes) that describe the domain's major entities, and for coupling those with focused and flexible observation templates. We present a conceptual model for the observation data type, and show how we have implemented the model as an observation entity in the DataBank database designer and generator. We show how our visualization tool CanopyView exploits metadata made explicit by DataBank to help scientists with analysis and synthesis. We conclude by presenting future plans for tools to conduct statistical calculations common to forest ecology and to enhance data mining with DataBank databases.

DataBank could be extended to another domain by replacing our forest–ecology-specific templates with those for the new domain. This work extends the basic computer science idea of abstract data types and user-defined types to ecology-specific database design tools for individual users, and applies to ecoinformatics the software engineering innovations of domain-specific languages, software patterns, components, refactoring, and end-user programming.

* *Corresponding author.*
E-mail address: fialaa@evergreen.edu (A.C.S. McIntosh).

1574-9541/$ – see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.ecoinf.2007.07.003

1. Introduction

Critical advances in science often depend on the synthesis of many individual studies, a process that includes gathering and exchanging data, refining and understanding what the data mean, and conceptualizing how the constituent studies interrelate. These activities allow scientists to identify patterns in nature and to extract emerging properties and processes that explain these patterns. However, these activities also require the development and improvement of data infrastructure, informatics, and analysis tools. Ecology is no exception: as with many fields of science that require multiple investigators and sharing and comparing research results from different laboratories and sources, ecologists have recognized the pressing need for synthesis, as evidenced by the Long Term Ecological Research (LTER) Program declaring 2000–2010 as *The Decade of Synthesis* (LTER, 2002). Ecological synthesis will likely involve large-scale and long-term data from multiple sites and require extensive data sharing and data mining (http://www.evergreen.edu/bdei; e.g., Knapp and Smith, 2001; Peterson and Vieglais, 2001; Worm et al., 2006).

Validated and well-documented data archives increase the likelihood that data integration and data mining will enable ecological synthesis. First, archived data must be stored or displayed in formats that can be retrieved and understood by researchers who did not themselves gather the data, and preferably be machine processable. Research data sets with flat-file data formats such as text files and spreadsheets have an arbitrary organization, and are therefore difficult to archive in useful ways; they lack verification and built-in structural metadata[1]. In contrast, properly designed relational databases provide valid structural metadata, facilitate some verification, and avoid data repetition, all of which ease data integration. In addition, relational databases can help in organizing semantic metadata; which will help determine if datasets share common assumptions (e.g., comparable units, vocabulary, experimental design). Ecoinformatics tools that produce field databases using relational databases could help overcome both structural and semantic obstacles to data and metadata archiving and subsequent synthesis efforts.

Many ecology-specific informatics tools and protocols developed thus far are aimed at how fully-documented data archives can be effectively used for data browsing and integration. They rely on valid semantic metadata associated with datasets that are already machine-processable (e.g., parsable). Many ecologists, however, even if they are willing to share their data, lack the expertise or inclination to render their field datasets "well-behaved" in the formal sense. They face problems in documenting and validating datasets, and readying their data for long-term archives (Michener et al., 1998; Michener and Brunt, 2001). We therefore need two kinds of improvements in archiving and synthesis tools: those that can work with incomplete and not-validated data, and those that improve the quality of our archives. Our approach is to provide productivity tools, using database technology, for scientists during their research process; tools that gather and refine metadata at design and data collection/validation and tools that use those metadata during analysis and to archive data sets. Our work thus focuses on making databases easier for individual ecologists to use during study design, data collection and validation, data analysis, and archiving. We apply recent software engineering research in high-level domain-specific languages (Kieburtz, 2000; Sheard, 2001), component-oriented object-oriented design (Szyperski et al., 2002), abstract data structures, and end-user programming (Burnett et al., 2006) to create these tools. Our work has three foci: 1) help end users design better databases; 2) provide productivity enhancements for the end-user; and 3) help users create databases that are more easily archived. This paper focuses on database design aspects of our work (focus 1). We address foci (2) and (3) by suggesting how our approach to design will facilitate them, and present some preliminary evidence to that effect. We emphasize the criticality of domain analysis for our work. As new informatics tools solve syntax problems, tools for capturing and exploiting semantics will become more important for data integration research (Poulin, 1995).

One ecology subdiscipline that could benefit from better database design by its researchers, as well as from other informatics advances that increase researcher productivity, is the study of forest canopies. The forest canopy has been termed "the last biotic frontier", and is one of the richest but most poorly understood habitats in the biosphere (Lowman and Nadkarni, 1995). In the past two decades, the field of canopy studies has become a data-rich discipline that bears on many fields of science, including ecology, microclimatology, geography, and conservation biology. Originally, canopy-related datasets were small, and drew upon basic forestry measurements collected with simple instruments (e.g., branch diameter, throughfall volume). However, new access techniques and technology have allowed canopy researchers to use instruments and approaches of increasing complexity. The growing volume of data now requires increasingly sophisticated conceptual frameworks, data management approaches, and software tools. Further, the extreme longevity of trees (over 1000 years in some cases) incurs the need to make repeated measurements of the same object through periods that can exceed the lifetime of a single grant or even of a single human researcher, thus requiring that data be "passed down" from study to study, and from researcher to researcher. This characteristic of the discipline means that data must be archived and well documented. The three-dimensional (3-D) and irregular structure of trees is difficult to visualize with 2-D software tools readily available to ecologists. Relational database technology is particularly helpful for forest ecology data because the large number of interlocking parts and systems that make up forest structure data: branches relate to trees by growing out of them and sometimes to each other by splitting or budding, and trees relate to each other by physical proximity. To understand how forest structure drives important tree and forest functions, researchers need to clearly define relationships among those parts.

Because of the need for good field databases, as described above, we aim to help ecologists design better databases. This

[1] In a relational DBMS, structural metadata contains the syntax that defines the database structure, including table and attribute names, attributes data types, primary keys for each table, and relationships among data elements, i.e., which table is related to which table and which attribute (foreign key) implements that relationship. We extend this definition to cover analogous metadata for data in other forms, and contrast it to semantic metadata.

is no simple task, as it is well known in the professional and teaching database communities that design is "more art than science" (Storey and Goldstein, 1993), and that it takes a long time to become a good database designer. Some people believe that, as with software design, this is because the designer must experience many different data patterns and operations on those patterns before he or she acquires design ability (Gamma et al., 1995). We have packaged patterns of good database design and operations on those that novices can use and re-use in our domain-specific abstract data structures (ADTs) that we call "templates". Using our templates also helps overcome errors in database design most commonly made by novices, i.e., in defining relationships, because our templates contain rules for composing them into related tables in a formally correct manner (e.g., 3rd Normal Form). Some empirical studies (Antony and Batra, 2002) suggest that novice designers perform better when using conceptual models than when using the more formal logical models (such as the relational model); our system presents the first cut designs in a format quite analogous to conceptual models. Finally, we chose to use relational technology (RDBMS) because easy to use database management systems, like MS Access or MySQL, are readily available to our user community. Many RDBMS now incorporate object-oriented features that forest ecologists will find useful, such as user-defined data types and binary large objects (BLOBS).

In this paper we describe how our Canopy Database Project (CDP) is using the emerging field of forest canopy studies as a model discipline to advance fundamental informatics and eco-informatics research. Since 2002, we have experimented with real-world forest ecology databases, refined our software and applied-software engineering refactoring concepts (Ambler and Sadalage, 2006) to redesigning our templates. The question for us is no longer "can tools using templates do useful eco-informatics work?" but rather "which are the right templates to use?" In this paper, we argue for defining simple templates (with relatively few attributes) that describe the major entities under study in the domain, and for coupling those with focused and flexible observation templates. In Section 2, we describe the current status of our ecoinformatics tool prototypes, how our database templating system works, and our experience using the tools with real users. Then, in Section 3, we share lessons learned from the use of large granularity templates in the first iteration of system development, and discuss resulting advances in our informatics research. We describe the observation template in some detail, including a conceptual model of the observation. In Section 4, we use several real-world examples to show how databases created with our templates, when coupled with our tools and their embedded semantics, have helped scientists analyze their data with visualizations and conduct preliminary ecological synthesis. Finally, in Sections 5 and 6, we enumerate areas for our own future informatics work and conclude the paper.

2. The Canopy Database Project (CDP)

The CDP is an interdisciplinary project between computer scientists and forest ecologists who collaborate on informatics tools for forest ecologists (Nadkarni and Cushing, 2001; Cushing et al., 2003a), and has developed three prototypes: DataBank, CanopyView, and StudyCenter. DataBank and CanopyView use entity and observation templates.

2.1. *Templates*

Our tools are based upon a simple concept: software components that we call templates are provided with the system. Templates are used by our tools to generate database designs, databases, data entry forms, and visualizations. Particular templates, which behave much like design patterns or abstract data structures are maintained in libraries, and users can re-use designs and extend existing templates or create new ones. Templates potentially allow greater database flexibility than centralized data models, which are too restrictive in most scientific domains and which require users to transform data to fit the centralized data model (Halevy et al., 2003; Franklin et al., 2005). The use of standardized database components and a standard observation data structure, however, still enable us to build some tools around those structures, and reap software engineering benefits from re-use (Krueger, 1992; Szyperski et al., 2002). Further, if the datasets of separate studies have one or more common database templates, the data integration necessary for synthesizing those studies would be significantly easier than if each were developed idiosyncratically. Templates contain rules on how one template can be composed with others into a database design or what elements might be needed for a particular visualization; they are currently represented as XML documents (http://www.w3.org/XML; Nottrott et al., 1999). Fig. 1, for example, depicts a branch template; as an entity template, it will become a table in the generated database and will be explicitly related to a stem template in the design (and a stem table in the generated database). The template XML dialect is compatible with the Ecological Metadata Language (EML; http://knb.ecoinformatics.org/software/eml; Michener et al., 1997).

Each template includes information that is used as metadata in the generated database. DataBank embeds this metadata by creating a special system table in each new database. This table is hidden from the end-user, but provides descriptions of the entities and observations. Template information takes three forms: 1) The "canonical name" of the template identifies the template; every template has a unique canonical name. Even if a user renames a template or a database table, e.g., from "stem" to "trunk", the canonical name remains the same in the metadata (hidden in the system table). These canonical names, along with user-defined names, provide a kind of user-extensible, controlled vocabulary or lexicon. 2) Attributes of a template (its associated data fields) associated with a template are described, including name, description, relationships, units, and data type (e.g., boolean, integer, decimal, text). Attribute names can also be changed by the user, but the canonical name is retained in a hidden system table, as described above. 3) The template's relationship to other templates, such as the fact that the 'branch' template (Fig. 1) corresponds to the 'stem' template. These characteristics of templates make it possible to infer (intuitively if not formally) an ontology from a collection of DataBank databases.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
 <Template xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="http://scidb.evergreen.edu/databank
      /databaseCenter/schemas/teof.xsd" description="This is a simple
      branch entity associated with a tree. You can add observations to
      customize it for your study." name="Branch"
      path="canopy/branch/branch" type="Entity">
   <Entity name="Branch">
      <Observation name="" rq="oneToOne" type="structure">
         <AttributedataType="String" description="Branch tag or
              number" index="0" name="BranchName" />
        <Attribute dataType="Reference" description="Foreign Key"
              index="2" name="Stem_id" referenceTarget="Stem">
          <Dependency name="Stem" type="specific">
            <Targetpathname="canopy/stem/stem" />
          </Dependency>
        </Attribute>
      </Observation>
   </Entity>
   <InfodisplayDescription="This is a simple branch entity associated with
        a tree. You can add observations to customize it for your study."
        displayName="Branch" name="Branch" parameterized="false"
        path="canopy/branch/branch" type="Entity" />
 </Template>
```

Fig. 1 – DataBank Branch template (an XML file).

2.2. *DataBank*

DataBank helps ecologists design normalized relational databases with built-in data-entry forms, along with metadata files, including the equivalent of a data dictionary. Users select entity and observation templates in a drag and drop fashion to compose a database design that can be saved for re-use. The database design (represented as Java Objects) is mapped to SQL tables and relationships. From this an MS Access database, with data entry forms and some data validation procedures, is generated. The usability and power of DataBank depends heavily on which templates are provided to users; the refactoring of our initial templates is discussed in Section 3. Databases designed with templates can be translated into EML and hence documented, archived, and validated using LTER and SEEK tools (e.g., Metacat and Morpho — Higgins et al., 2002). For more information about DataBank internals and architecture, see Cushing et al. (2007a). The current system implementation of DataBank can be seen at http://canopy.evergreen.edu/DataBank.

While we have been using DataBank for forest canopy studies, and most of the templates are specific to that subdiscipline, other domains could make use of the system by adding their own domain-specific templates to the system.

2.3. *CanopyView*

Our domain-specific data visualization tool CanopyView evaluates a DataBank database using the database's metadata, and offers to the user pre-designed static or animated visualizations that fit the dataset (Cushing et al., 2003b; Finch, 2003; Zeman et al., 2006). The visualizations display forest canopy datasets in ways that illustrate in three dimensions the relationships among individual data elements, such as where a branch is connected to a tree stem. CanopyView provides the benefits of data visualization: identifying patterns, finding errors, and superimposing multiple observations (e.g., overlay epiphyte cover on tree structure). Several datasets can be viewed on the same canvas, and visualizations from two distinct databases superimposed; this provides some synthesis capability and will be discussed further in Section 4. CanopyView is implemented in Java, using the Visualization Toolkit (VTK; Schroeder et al., 1998). The current system implementation of CanopyView can be seen at http://canopy.evergreen.edu/CanopyView/.

2.4. *StudyCenter*

The StudyCenter (http://canopy.evergreen.edu/DataBank/studycenter/studies/) serves as a data, metadata, and artifact repository for forest ecology studies. The StudyCenter is not restricted to DataBank-generated databases. Datasets in any form, such as flat files or MS Excel spreadsheets, can be archived there. Visitors to the site can search for studies, and download data and associated metadata and artifacts (e.g., images). They can also create their own study pages and upload study data and artifacts, restricting access so that their datasets are available only to their project team, or expand access to the general public. The StudyCenter is currently powered by Plone (http://plone.org/). Because DataBank databases comply with LTER metadata requirements, StudyCenter databases could be easily archived at LTER sites, or other EML-compliant archival sites, or integrated with other EML datasets. Like many current data repositories, the StudyCenter provides

only browsing and download; to integrate two or more databases, users must download them and perform the integration themselves. We discuss our plans for building tools that query and integrate DataBank databases in Section 5.

2.5. *Test cases and real-world use of our database design, visualization, and archiving tools*

To date we have used StudyCenter to archive DataBank databases and metadata for ten test-case forest canopy research studies. The databases range in complexity from three related tables in the Luquillo LTER Forest Dynamics Plot Canopy and Elevation Database, to a database consisting of eight separate databases, each with 33 related tables for the Thousand-year Chronosequence Study (1kcs; Van Pelt and Nadkarni, 2004). Some databases have been proofs of concept (e.g., whether we can generate databases that cover sufficiently different ecological questions and a broad enough range of data types). Others have demonstrated the usability of our system (e.g., a scientist used DataBank to generate a database and then converted his data from spreadsheets into the database so he could visualize his data in CanopyView, and then uploaded his study to the StudyCenter). Some have raised questions about post facto data integration (e.g., how easy it is to integrate data sets that have been converted to our format). Finally, we have used the datasets to determine which visualizations are useful to our researchers, and subsequently refined the CanopyView visualizations accordingly. Workshops conducted at forest canopy research conferences (http://scidb.evergreen.edu/leipzig-workshop/ and http://scidb.evergreen.edu/esa_workshop06/) and for six LTER Information Managers (http://canopy.evergreen.edu/workshops_lter1103.asp) also helped us refine the usability of our systems. In written workshop evaluations, a majority of participants indicated an interest in using the system for their research studies, now that they had more familiarity with database capabilities. For most, this was their first introduction to database technology; one commented:

> At the moment I manage my epiphyte distribution data in Excel, where I use different sheets for branches, trees and localities. I link the sheets by using formulas, thus creating a more or less 'relational' database. This seemed a logical structure at the time when I entered my data in the field, but now I find it difficult to extract the information I need. The queries in Access should give me a lot more freedom in getting the right information.

Our test cases and workshops on how to use our tools helped us understand that our users needed more flexibility in design, and the ability to rename tables and attributes. Even when users agree on the semantics of a particular database table or attribute, they rarely agree on the name they want to use for that concept. Allowing users to change names in templates (e.g., rename a 'stem' to be 'trunk' or 'bole'), however, required a major rewrite of the system. Paradoxically, the more challenging problem, which required virtually no changes in the DataBank software itself, was the question of "which templates are the right templates". This problem was challenging because it required rewriting almost every template we had originally written for the system, and changing our approach to template design.

3. Which templates are the right templates?

We originally designed templates to represent the basic components of forest structure. These were reverse-engineered from forest canopy studies that encompassed a wide range of forest structure data (e.g., studies conceptualizing a tree stem as a point, a line, a cylinder, or a cone; conceptualizing a tree branch as a point, a line, a cylinder, or a cone). We envisioned these structure templates could be combined into a database design, with each user choosing one template for a particular structural entity in the forest (e.g., stem, branch) from the suite of templates for that entity. Built into each of those templates were all observations necessary to model one conceptual structure of the particular entity (e.g., an aggregate stem template called "Stem with crown radii and decay" included crown radii and decay class information within it). While such large granularity "top-down" templates were good for modeling a user's conceptual understanding of the forest, and our original templates were received with enthusiasm in forest ecology focus groups, using large templates did not work in practice. Although these templates captured stable domain concepts and were highly expressive, informal usability testing showed them to be too restrictive; they did not adequately cover all forest canopy studies. Because many ecologists lack standardized protocols or measurements, our templates needed to capture a wide variety of observations based on users' idiosyncratic conceptual models of the forest. Thus, we shifted from a 'top-down' to 'bottom-up' approach for template design, de-coupling our pre-designed large granularity templates into two types that make templates modular and customizable: 1) entities and 2) observations. We also created tools for users to create custom observations and to create their own templates either from scratch or using existing templates. The primary ecoinformatics research result of this work has been our conceptual model of the observation and its implementation in the observation template, along with a scheme for our tools to infer the semantics of observations in a generated database.

3.1. *Entity templates*

The core entities of canopy research (e.g., tree stems and branches) are quite stable and generally well understood. This is because ecological data are inherently spatial and most research involves making observations about structural elements (Roberston, 1987; Martens et al., 1991), which we have observed are less likely to change over time or differ among studies than functional (i.e., observational) measurements. Further, having a few core domain-specific entities has been considered crucial to integrating datasets (Yin et al., 2005). Entity templates are used to describe the domain-specific objects found in forest canopy studies, for which researchers typically record observations. Entity templates typically represent the structural components of the forest canopy, such as tree stems and branches. Entity templates can be protected or user supplied. Protected templates are those that the system will not allow users to change. A user can, however, copy the protected template to the custom template library area, and modify it using the *Custom Template Editor* (described in Section 3.4 below). Our new simplified entity templates include no or very few observations they are used as building blocks on which to

overlay observations. Having a few simple core entity templates and a large collection of system-supplied observation templates (see Section 3.3 below) that can all be re-used, allows our users to place observations on those entities in a flexible, yet somewhat controlled, manner.

3.2. A conceptual model of the observation data type

To design the observation templates and to reason more formally about how a user might compose them into a database, we developed a conceptual model for the *Observation Data Type*. This conceptual model draws heavily on Ch. 3 of Fowler (1997). Table 1 shows an instantiation of our model, which was used as the foundation on which we built Observation Templates. To explain the model, we walk the reader through one row of the table. An *Observation* is of some *Domain Type*, for example *height*. *Domain Types* can be thought of as corresponding to database attributes, modifying or describing an entity of interest. *Observation Type* is analogous to a user-defined type, a facility allowed in most programming languages that extends the type system of the programming language beyond system-defined types such as integer, character, etc. All *Observations* of the same *Domain Type* are also of the same *Observation Type*. The *Observation Type* associated with the *height Domain Type* is a linear measure. An *Observation* also has one or more *Units* associated with it, e.g., cm, m, kg, m^3, etc. The unit for a *Domain Type* can be null as for a percent observation, fixed, or specified at compile time, i.e., when the *Observation Type* is placed into a program or a database, or even added at run-time, i.e., when data are added to the database (though this is more complicated). Some *Domain Types* have modifiers. For example, a height *Observation* might be associated with the core entity *Tree*, but *Tree* might have two height observations: *crown* height and *foliage* height. *Observation Types* can be used for data validation at data entry, similarly to how type checking is used in compilers to check the correctness of programs, either at compile time or as in our case at "run" time. The *English Language Equivalent* of a *Domain Type* can be used to "seed" metadata descriptions of instantiations of that *Domain Type* in a generated database. *Domain Types* are grouped (as they are in Table 1) so that they can share common *Observation Types* and *English Language Descriptions*.

The observation data type model has power to represent a wide variety of ecological measurements. For example, some observations might be enumerated types, e.g., the core entity *Tree* might have a species observation associated with it, which is an enumerated type whose values might be provided in a database source table. In fact, some of DataBank's core entities come preset with certain observations, and some of those are enumerated types that are implemented as database source tables, e.g., the tree entity has a species Observation, and hence an associated source table for Tree Species. Consider the observation on a tree of percent epiphyte coverage, by species;

Table 1 – Instantiation of the observation Data Type appropriate for Canopy Science

Observation Type	Has Units	Domain Type	Has Modifier (examples)	English language equivalent
Percent	Percent	Percent [relative-] humidity	<species-code>, <form-code> above canopy, below canopy,…	A percentage
Temperature	degrees-C, degrees-F	Temperature	air-, water-, ground-,…	A temperature
Volume (wet)	l, ml, qt, pt, gal, …	Volume capacity	Rain, throughfall	A liquid volume
Volume (dry)	cm^3, bushel, …	Quantity		A 3D linear measure of space or dry volume
Area	cm^2,m^2,ft^2, …	Area		A 2D linear measure of space
Weight	lb, kg, g, oz, …	Weight	Net productivity	A weight
Compass-point	degrees	n, e, s, w, ne, sw (etc.) azimuth		A point on the compass
Angle	degrees	Angle aspect		An angle
Cartesian-coordinate	m, cm, ft, yd, …	x, y, z		A point relative to a base point, in a direction (plane)
Linear-measure	m, cm, ft, yd,	Length, height, width diameter, dbh, radius extent top, base in (enter) start (begin) out (exit) end (leave)	crown-, basal-, foliage-, at-bole-,	A measure of distance
Date-time	ccyymmddhh, ccyymmdd, etc.	Date, time, age		A point in linear time, at an identified time-precision
Time (elapsed)	h, min, s,			A quantity of time
Boolean (yes/no)	–	isDead isAlive isSampled isMain is_____		A truth value (true/false, yes/no)
Enumerated-type	{coded value}	Species {user-defined- e.g., status, decay-class, …}	pnw- conifer-, …	A set of code-value pairs
Count	–	Count tally total	<species-code>, evergreen-, herbaceous-, …	A summary value of the number of some set of things
Label		Name tag address		Short alpha-numeric text, used to label some thing
Description		Description comment notes		Freeform, narrative text, used to describe some thing

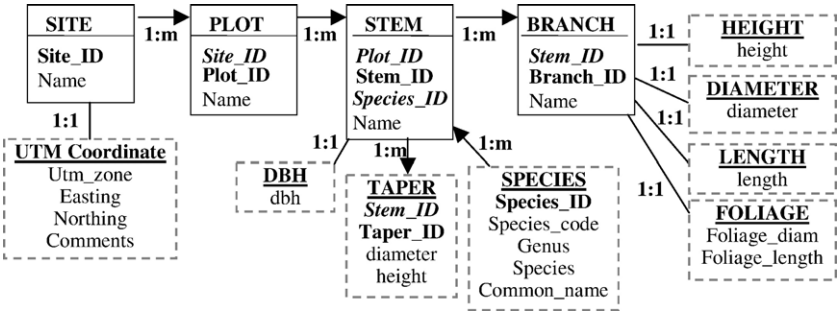


Fig. 2 – Entity and observation templates using the refactored 'bottom-up' approach. Each rectangle represents a template. Template names are in capitals and underlined at the top of each template box. Entity templates are enclosed in solid rectangle boxes. Observation templates are enclosed in dashed boxes. Directional arrows between templates indicate the direction for the one-to-many (1:m) relationship between the templates. The primary key for each template is highlighted in bold, while foreign keys are highlighted in bold and italicized. The attributes for each template are provided below the primary and foreign keys. One-to-one observation templates do not have a primary key.

this is modeled by the Percent *Observation Type*, with two modifiers: epiphyte coverage and species. In a different database design, an ecologist might want to use an epiphyte plant association group (form), rather than species, as a modifier. If the species template has as an optional observation 'form' attached, and there are data showing which species belong to which form or plant association group, then the epiphyte coverage observations from the two databases are likely comparable. A further example of the power of this model is that it allows for modeling complex *Observation Types* (i.e., those with more than one associated value, possibly built from other *Observation Types*); an example is the *Domain Type* Thiessen polygon whose *Observation Type* would be a collection of points, possibly ordered.

We anticipate performing queries on our databases to export data sets in certain target formats, e.g., the LTER PASTA architecture, where observation data are stored as date-value pairs and all remaining data are stored as metadata (Servilla et al., 2006). The *observation* conceptual model provides us with direction on how to extract database content into metadata and data, depending on the target format.

3.3. *Observation templates*

Observation templates in DataBank implement the conceptual *Observation Data Type* model. They describe information (e.g., measurements) recorded for a given entity (e.g., the diameter at breast height of a stem). We based our implementation of observation templates on both the conceptual model (Section 3.2 above) and an early preliminary survey of forest ecology data. Observation templates, like entity templates, can be protected or user supplied. DataBank has the following protected templates: i) eight location-based observation templates (e.g., location on an *x*, *y* grid, utm coordinates), ii) 26 measurement-based templates (e.g., diameter, crown taper), and iii) a generic *Customizable Observation Template* (described in Section 3.4 below).

DataBank presents observation templates to the user in a relatively simple manner: an ecologist selects relevant observation templates by browsing through the small-granularity observation templates in the Observation template libraries. In contrast to entity templates, observation templates cannot be added independently to a DataBank database design; they must be attached to one of the entity templates already in the database design window. The user specifies whether an observation is one-to-one (for each entity, the observation is measured only once, e.g., recording the diameter at breast height) or one-to-many (for each entity, the observation is measured multiple times, e.g., measuring stem taper at 5-m height intervals along a tree trunk). The information about the correspondence between the observation and the entity is encoded in the structural design of the database. A researcher might attach observation templates to entity templates to create such a database for a sample study with four entities of interest (study area, plot, stem, and branch), with observations recorded, as was done by Van Pelt and Nadkarni (2004) (e.g., Fig. 2). The database designer and template preview windows in DataBank are congruent with the sample study conceptualized by the researcher (Fig. 3), which is then used to generate a MS Access database (Fig. 4).

3.4. *Custom templates — the customizable observation template and custom template editor*

Since we cannot foresee all observations, or even major domain entities, that an ecologist might want to model, we provide the capability of customizing an observation template or creating a new entity template. For example, a researcher might be interested in studying the empty air space among trees in a forest, for which there are currently no built-in templates (e.g., Dial et al., 2004). We thus added two new features to the DataBank software: i) a *Customizable Observation*, and ii) *Custom Template Editor*. These allow users to represent structural and functional data not currently captured by our DataBank template library, such as for the empty air space and observations recorded for it. Another researcher who studies the bark of a tree could either add a "bark" attribute to the "stem" template, or create a "bark" observation template. These represent different ways of thinking about the relationship between "bark" and "tree".

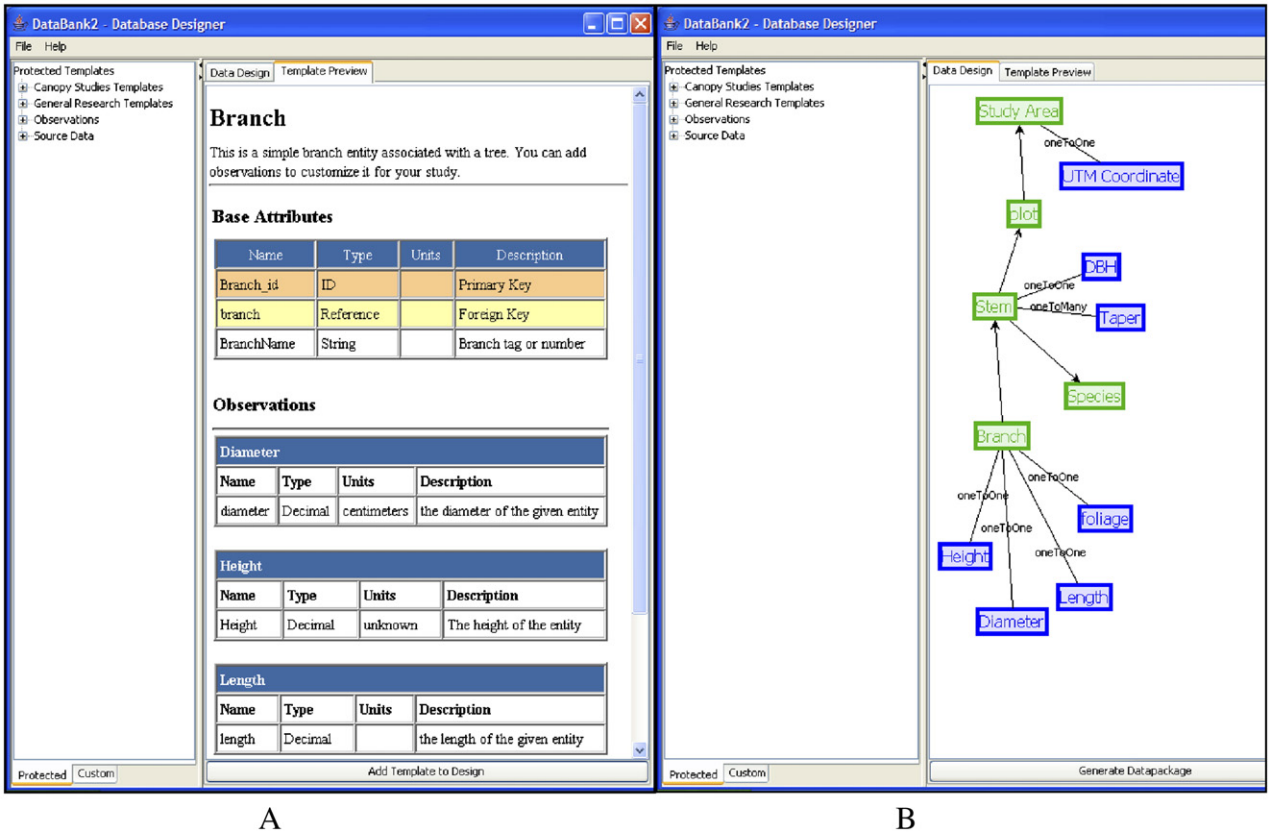


Fig. 3 – Screenshots showing creation of the DataBank database for the example in Fig. 1. A) Sample DataBank template preview window showing the attribute descriptions for the 'stem' entity template and some of the associated observation template descriptions. The left window is where the protected or custom templates are selected or edited from. Attribute descriptions populate the attribute description fields in the created MS Access database. B) The DataBank design window displays the entity and observation templates that are used to create a MS Access database. Entity templates are connected to other entity templates by lines with arrows (green boxes), and associated observation templates (blue boxes) are connected to entity templates with a line that describes the relationship between observation and entity (e.g., one to many). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

If a researcher considers bark an essential part of the tree, i.e., something he or she will want to measure every time he or she measures a tree, he or she will want to add it to the stem entity template so that whenever the stem template is called up, bark will be included. But if bark type is an observation that is not measured every time, he or she can make a new bark observation template that can be re-used. Neither of these ways is more correct — each reflects a different way of measuring the forest; the resulting relational database is the same. Through use of the customizable observation and custom template editor, the researcher acts as a programmer, using DataBank to build, test, and publish his or her own templates and then, as a user, use those templates to build databases.

Sometimes, a user might wish to make a certain observation only once, or simply load a dataset into DataBank to use its visualization capability or compare it with a DataBank database. In this case, rather than creating a new custom template, the user could use the *Customizable Observation Template*; this template is a generic observation that allows the user to specialize it and add a specialized single-attribute observation to an entity template already in his or her database design. The created observation is not available for re-use. A benefit to the user is that he or she is walked through all steps for adding the observation to the template, i.e., when users click on "Customizable Observation" in the protected templates Observation section, they are asked to "Select an entity on which to add observation" and can select from the entity templates already in their data design. Then the user is asked to select whether they "plan to take this measurement once or more than once" (which determines whether the observation will be within the same table as the entity onto which it is being applied, or if the measurement is recorded more than once, will determine that the observation will be in a separate table, e.g., taper which is recorded many times for a single stem will have its own table). Finally the user defines a name for the observation, the name for the attribute, a description of the attribute, and defines the type. The user then hits okay and their customized observation is added to the selected template in the data design window.

Using the *Custom Template Editor*, on the other hand, a user can create re-usable templates specific to the unique

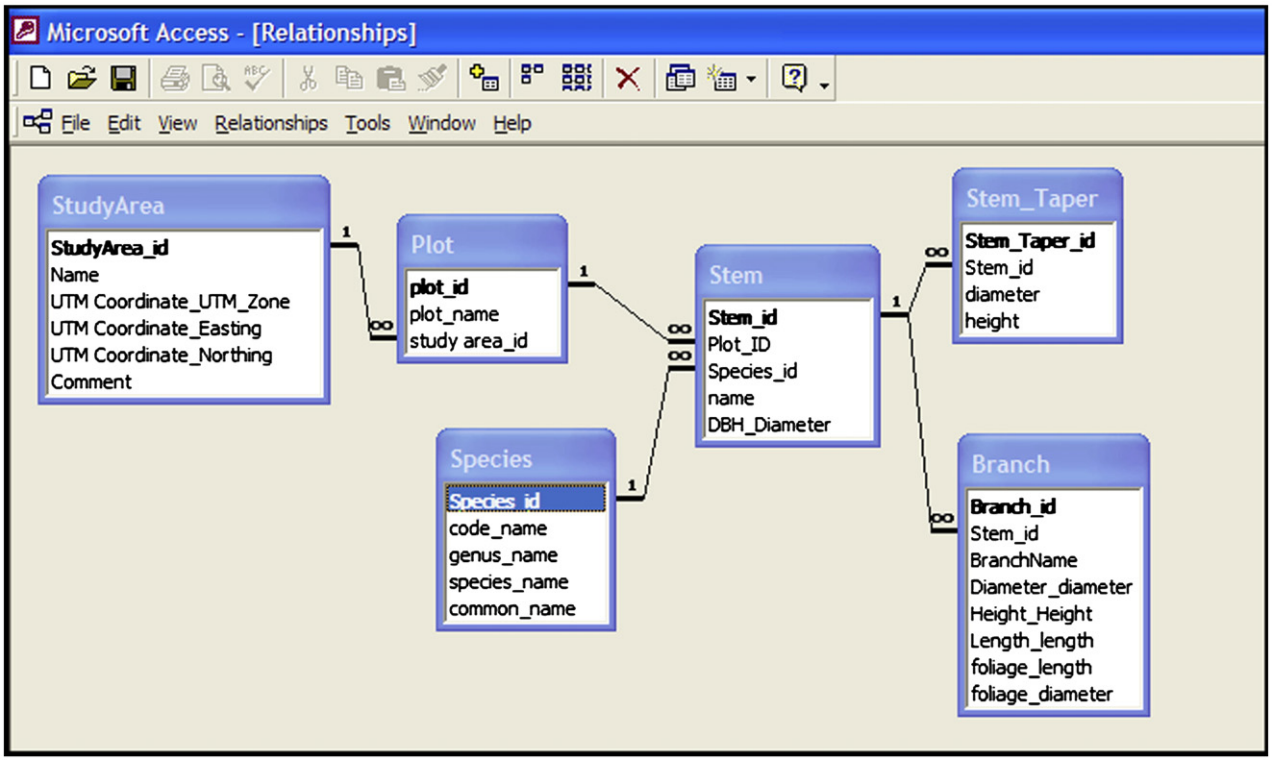


Fig. 4 – The entity-relationship diagram in MS Access for the DataBank database design example shown in Figs. 1–3.

entities or observations of his or her study. These templates can be based on existing templates where the user can paste and modify an existing template, delete variables, add new variables, delete, add, or edit relationships, and rename templates and variables. Alternatively, a customized template can be created from scratch. Observations can be created in two ways, either created by using an existing observation template or from scratch in a customized entity template. For each new entity or observation template the user is required to provide the same information as was described for the customizable observation template. The only difference between the two is that the primary key for a one-to-many type observation can be assigned its own data type (e.g. text, integer); the primary key is automatically autonumber in the customizable observation.

3.5. *Section summary*

The template refinements described in this section required few if any changes to the DataBank software, but greatly enhanced the extensibility of DataBank to other domains beyond forest canopies. Domains with complex related objects could more readily fit their measurements into the template definition scheme and then benefit from use and re-use of data entities and observations. The template refinements required a major rewrite, however, of our visualization tool CanopyView; in particular we needed to figure out how to infer the semantics of observations in a generated database. We could no longer simply assume the conceptual forest structures represented in a database from a single template used to create that design.

4. How entity and observation templates facilitate visualization

Data-driven visualization is a helpful tool for forest canopy ecologists, one that incentivizes their use of DataBank. We found need for flexibility to customize visualization for each researcher's dataset. Also important is having a visualization tool that communicates directly with the data, eliminating the need to transform data from one software package to another. Visualizations that perform calculations (e.g., volume of a tree) are also worthwhile. CanopyView meets these needs, but can do so only because DataBank "hides" metadata[2] where it can later be exploited. This section explains how CanopyView exploits syntax and semantics from templates to create powerful visualizations, which have been used by our ecologists in preliminary synthesis work. We believe it shows some benefits of refactoring templates into entity and observation types.

To determine if two or more datasets are comparable, one must determine at the least if two variables, whether of

[2] Every Microsoft Access Database contains several "system" tables used to encode metadata about the database, such as primary key/foreign key relationships (MsysRelationships) and query creation and update times (MsysObjects). By default, these tables are not accessible to the user, but an advanced user can run high-level queries against them.

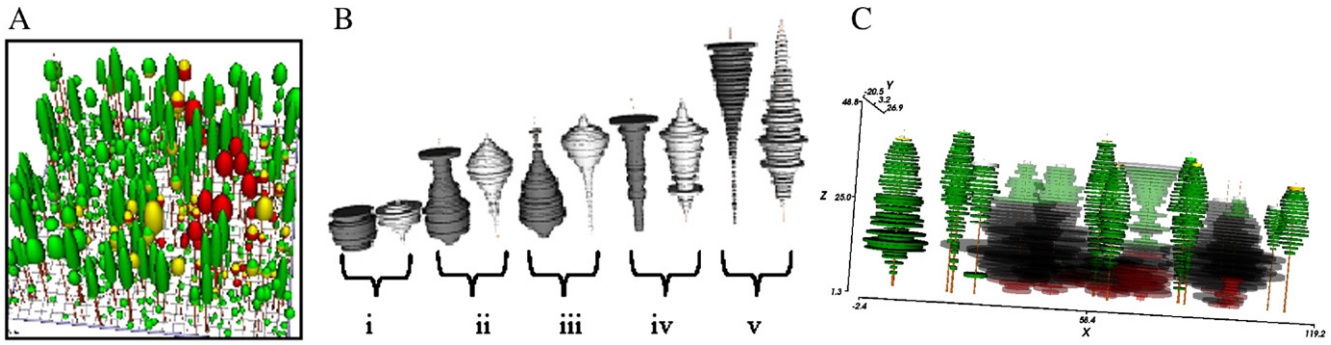


Fig. 5 – Examples of CanopyView visualizations that demonstrate synthesis. A) Dwarf mistletoe infection ratings (different shades represent different levels of infection) in an old-growth Douglas-fir forest, WA, USA (Shaw et al., 2005). B) Aggregated branch length data and air space data for all measured trees at 1-m vertical intervals at each of five of the thousand year chronosequence (1kcs) sites: i) Plantation — age 50, ii) Martha Creek — age 100, iii) Panther Creek — age 165, iv) Trout Creek — age 500, and v) Cedar Flats — age 650. The dark grey figures on the left in each pair of objects represent aggregated air space data, and the light grey figures on the right represent the area occupied by the sum of tree branches. Each concentric ring represents a single 1-m vertical height interval. C) Geo-referenced canopy airspace and branch volume data overlain along a transect of measured trees and airspace at Martha Creek.

the same name or different name, refer to the same concept. Databank uses a lexicon approach to accomplish this, by adding two custom system tables to each database it creates: DatabankObject and DatabankRelationship. DatabankObject contains a list of the names and types of each entity and observation from which the database was created, even if a user has supplied a custom name. The DatabankRelationship table recreates the entity graph, showing which observations are applied to which entities and relationships between entities. CanopyView reads these two system tables to determine what templates and observations make up the database, and thus what visualizations are possible. CanopyView must also establish, as best as it can, that no tables or columns have been changed or deleted by the user and the data are intact, even if the user has changed the name of a particular table or observation. In addition, if the necessary variables are provided in the database, CanopyView can perform calculations, including stem volume. Most importantly, it can visualize two or more databases in one 3-D scene, which helps users determine the value of integrating those databases and performing further statistical analysis.

Each possible CanopyView visualization is characterized as a rubric that specifies what type of data is needed to generate the three-dimensional scene, and how the needed data must be inter-related. This rubric corresponds to the relationships between entity and observation templates. CanopyView reads the system tables created by DataBank to determine whether particular combinations of visualizable templates are present. First, CanopyView looks for all the entities required for a given visualization by checking whether each entity is present in the DatabankObject table. If CanopyView finds the presence of a particular entity, such as stem, it must also determine the way stem is present in the database, i.e., its semantics. There are three ways "stem" can be present in the DatabankObject table: 1) the user has selected a pre-built protected "stem" entity template to construct his or her database; 2) the user has selected a pre-built "stem" entity template to construct his or her database, but edited or modified the name; or 3) the user has created a custom entity template called "stem" for his or her database. Since a user can edit a template or delete columns from a table, it is not sufficient to make sure a particular template was used to generate the database. CanopyView must also make sure that all the needed associated data are still present. To do this, CanopyView queries the system tables to find out the user-defined names of each attribute, and then queries the primary database to make sure all the needed data are present. Finally, CanopyView checks for the presence and completeness of observation data in the same fashion. If all the entities, attributes, and observations specified in a visualization's requirements are present[3], CanopyView offers the user a chance to use that technique to visualize his or her data. In many cases, several possible visualizations are available, and the user may overlay them.

Fig. 5 contains sample CanopyView visualizations that show how using a few major domain entity templates, along with many observation templates constitutes a controlled lexicon, and can enable preliminary ecological synthesis study. Fig. 5A shows a 3-D visualization of one forest stand, with superimposed observations about mistletoe infection. These datasets were collected at different points in time, but through the use of georeferenced templated databases they can be visualized on the same canvas using our standard pre-programmed visualizations. Fig. 5B allows the user to compare open space among trees with tree branch data for each of five sites; one

[3] CanopyView also checks to see if all required data for an entity are present with canonical names in the absence of entries in the DatabankObject Table. This functionality is present primarily for backwards compatibility with databases generated by older versions of DataBank, but it is sometimes useful in constructing synthesis databases — it allows the user to transfer one table from one database to another using Access' export functions to create new visualizations.

would expect (and both the visualization and preliminary analysis of the two independent datasets support this) that the shape of open space (to the left of each pair) would be the complement of tree branches. Fig. 5C shows individual trees with detailed branch-level observations, georeferenced at one location, and superimposed on the air space observations for the same location.

5. Future directions — a statistical tool, usability studies, and concept extraction

With implementation of the refactored templates complete, we now will move on to explore statistical and data integration tools that also harness the power of templates. We also plan some preliminary usability studies to determine if database design and data management, and data synthesis, are more effective and efficient using DataBank than using spreadsheets or flat files or traditional database software. Finally, we would like to determine explicit links between DataBank databases and a conceptual model of forest structure and function that we have described elsewhere (Cushing et al., 2007b). Below, we describe the additional software tools we envision.

Just as CanopyView capitalizes on the fact that it "knows" through metadata stored in MS Access system tables which entities and observations compose a database, other tools could similarly infer semantics that would enable programs that increase researcher productivity[4]. One example is to add a statistical capability that includes a semantic analysis similar to that done in CanopyView with parameterized statistical scripts and the capability to load data into a statistical package. We are in the very early stages of designing a statistical analysis package CanopyStats, which would allow researchers to re-use simple statistical scripts in R (http://www.r-project.org/), and store their own scripts for future re-use.

Extensions to current capabilities for searching and downloading StudyCenter datasets to include a better data mining capability would also be worthwhile. Working with large databases can be unnecessarily challenging, especially if a researcher does not know MS Access and is interested in only some parts of particular large datasets. Our first step towards this goal would be a tool that allows users to download a subset of the tables or variables from a DataBank StudyCenter database, e.g., for a synthesis of tree-level data, to download only tree-level data from existing studies that also recorded detailed branch, understory, and functional measurements. We are building a web-based warehouse for one of our databases to better understand the kinds of queries (and extractions) a user might wish to do online. Concept extraction from DataBank databases would be a second, more ambitious step towards the data mining goal. This would involve inferring, more formally than we do now with CanopyView, the higher-level structural or conceptual views that a user could impose on a particular DataBank database. We believe it is plausible to infer from the database whether a researcher, for example, conceptualizes crowns or stems as cones, or the forest as a continuous medium. We have developed a conceptual framework for the forest canopy, which could be used in this effort (Cushing et al., 2007b), but discussion of it is outside the scope of this paper.

Another way to facilitate data mining of DataBank databases, but one that we have not currently added to our development schedule, would be to export an ontology for a collection of DataBank databases. Our approach to database design with templates does not automatically produce an ontology, so it must be classified as a database-driven, not an ontology-driven approach. However, many database researchers, including Meersman (2001) and R. J. Miller in her informal remarks at a VLDB 2002 Keynote Panel (http://www.cs.ust.hk/vldb2002/program-info/panels.html) have argued that database structural metadata contain much of the information available with ontologies; systems like Miller's CLIO capture semantic mappings among different tables as well. Similarly, DataBank templates include adequate information for generating metadata with domain entities and attributes, the relationships between them, and short definitions of each. When users introduce synonyms of the terms included in the database, those are captured (in the system tables). We believe it is feasible, were DataBank databases to be integrated with other databases that use ontologies, to export the ontology implicit within one or more DataBank databases to a formal ontology language such as the OWL (http://www.w3.org/TR/owl-ref/) or into a tool specialized for ecology such as GrOWL (http://ecoinformatics.uvm.edu/technologies/growl-knowledge-modeler.html).

6. Conclusions

The major positive results of our experiences prototyping tools for ecologists were that it is possible to automatically generate databases using DataBank, write tools (e.g., CanopyView) that exploit structural and semantic metadata from DataBank databases for ecologically useful ancillary services (e.g., visualization), and derive other information from the databases, such as structural metadata. The primary negative finding of our subsequent three-year experience using those tools with users on field databases was that the building blocks we designed for database generation (our initial templates) were too specific for wide applicability. Additionally, some ecologists resisted using standard terms to refer to their measurements. Therefore, we redesigned templates to be smaller and more flexible, and modified our tools so that users could change names of tables and variables. The major innovations described in this paper are our definition and use of an "observation template" for forest canopy researchers and of our ability to infer the observation semantics of a database. Because our observation templates are based on a general conceptual model of observations, database schemas could later be exploited programmatically to extract general structural characteristics. We have also enhanced our tools so that end-users themselves can define new templates, creating a template editor that allows easier browsing, modification, and creation of templates.

Our work has shown the technical promise of using templates to construct individualized field databases. Such databases would be more correct and more comparable than idiosyncratically-designed databases, and more practical and

[4] This capability is known as "end-user programming".

flexible than global schemas. If end-users can design effective and comparable databases using templates, then productivity gains in their research, as well as easier data archiving, data mining, and data synthesis, will be more likely to follow. DataBank itself can be applied to other domains by replacing the forest canopy specific templates with those applicable to the new domain, and would be particularly useful in fields where many individual contributors design and run their own research. Granted, much of the work we have done to date involves deciding which templates are the right templates; however the particular entity and observation templates we have developed for forest ecology can be used as models in the many ecological and scientific domains that also concern objects with complex and dynamic structures.

Acknowledgments

We acknowledge seminal ideas from our collaborators David Maier and Lois Delcambre, now of Portland State University, on building databases from templates on forest structure templates, and from Keri Anderson Healy, a principal in The Automated Reasoning Corporation (ARCorp Inc.) and consultant for Model Systems Consultants, Inc., for the modeling of ecological observations. We also acknowledge implementation efforts by former staff members Michael Finch, Erik Ordway, Steven Rentmeester, Abraham Svoboda, and Robert Van Pelt, and current and former students Chris Pierce, Aaron Crosland, Jen Rawson, Youngmi Kim, Emerson Murphy-Hill, James Tucker, Brook Hatch, Neil Honomichl, and Peter Boonekamp. We thank the Computer Applications Lab staff Rip Heminway, Gregory Stewart, Erik Ordway, and others at The Evergreen State College for technical assistance. We also acknowledge input from past and current research collaborators including Roman Dial, Barbara Bond, Betsy Lyons, Hiroaki Ishii, Akihiro Sumida, Ann Auman, Kristin Vanderbilt, Nicole Kaplan, Ken Ramsey, Eda Melendez-Colom, and Jonathan Walsh. We thank the anonymous reviewers and special issue editor Matthew B. Jones for valuable feedback on the paper. This work has been supported by the National Science Foundation grants: DBI 04-17311, CISE 01-31952, BIR 03-19309, 99-75510, 96-3O316, 93-07771.

REFERENCES

Ambler, S.W., Sadalage, R.J., 2006. Refactoring Databases: Evolutionary Database Design. Addison-Wesley Professional.

Antony, S.R., Batra, D., 2002. CODASYS: a consulting tool for novice database designers. ACM SIGMIS Database 33, 54–68.

Burnett, M., Rothermel, G., Cook, C., 2006. An Integrated Software Engineering Approach for End-User Programmers," Margaret Burnett, Gregg Rothermel, and Curtis Cook. In: Lieberman, H., Paterno, F., Wulf, V. (Eds.), End User Development. Springer, pp. 87–113.

Cushing, J.B., Nadkarni, N., Bond, B., Dial, R., 2003a. How trees and forests inform biodiversity and ecosystem informatics. Computing in Science & Engineering 5, 32–43.

Cushing, J.B., Nadkarni, N.M., Finch, M., Kim, Y., 2003b. The Canopy Database Project: component-driven database design and visualization for ecologists. IEEE VISUALIZATION 2003. Poster.

Cushing, J.B., Nadkarni, N.M., Finch, M., Fiala, E.A.C.S., Murphy-Hill, E., Delcambre, L., Maier, D., 2007a. Component-based end-user database design for ecologists. Journal of Intelligent Information Systems 27, 7–24.

Cushing, J.B., Fiala, A.C.S., Nadkarni, N.M., Zeman, L.C., 2007b. Semantics for integrating forest structure data. 15th Annual International Conference on Intelligent Systems for Molecular Biology (ISMB)& 6th European Conference on Computational Biology (ECCB), The PLOS (Public Library of Science) Track, Vienna, Austria.

Dial, R., Bloodworth, B., Lee, A., Boyne, P., Heys, J., 2004. The distribution of free space and its relation to canopy composition at six forest sites. Forest Science 50, 312–325.

Finch, M.D., 2003. The Canopy Database Project: component-driven database design and visualization for ecologists. IEEE VISUALIZATION 2003, Seattle, WA. Demonstration.

Fowler, M., 1997. Chapter 3: Observations and Measurements. Analysis Patterns: Reusable Object Models. Addison-Wesley.

Franklin, D., Halevy, A., Maier, D., 2005. From databases to dataspaces: a new abstraction for information management. ACM SIGMOD Record 34, 27–33.

Gamma, E., Helm, R., Johnson, R., Vlissides, J., 1995. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, Reading, Mass.

Halevy, A.Y., Ives, Z.G., Suciu, D., Tatarinov, I., 2003. Schema mediation in peer data management systems. Proceedings of the 19th International Conference on Data Engineering 2003 (ICDE 2003), pp. 505–516.

Higgins, D., Berkley, C., Jones, M.B., 2002. Managing heterogeneous ecological data using morpho. 14th International Conference on Scientific and Statistical Database Management (SSDBM'02), p. 69.

Kieburtz, R., 2000. Defining and implementing closed domain-specific languages. OGI Technical Report. http://www.cse.ogi.edu/PacSoft/publications/bibliograph.html.

Knapp, A.K., Smith, M.D., 2001. Variation among biomes in temporal dynamics of aboveground primary production. Science 291, 481–484.

Krueger, C.W., 1992. Software reuse. ACM Computing Surveys (CSUR) 24, 131–183.

Lowman, M.D., Nadkarni, N.M., 1995. Forest Canopies. Academic Press, San Diego, CA, p. 624.

LTER, 2002. LTER 2000–2010: a decade of synthesis (white paper). 20 pp.

Martens, S.N., Ustin, S.L., Norman, J.M., 1991. Measurement of tree canopy architecture. International Journal of Remote Sensing 12, 1525–1545.

Meersman, R., 2001. Ontologies and databases: more than a fleeting resemblance. OES/SEO Workshop, Rome, Italy.

Michener, W., Brunt, J.E., 2001. Ecological data — design, management and processing. Blackwell Science Methods in Ecology Series, p. 180.

Michener, W., Brunt, J., Helly, J., Kirchner, T., Stafford, S., 1997. Non-spatial metadata for the ecological sciences. Ecological Applications 7, 330–342.

Michener, W., Porter, J.H., Stafford, S.E., 1998. Data and Information Management in the Ecological Sciences: A Resource Guide. LTER Network Office. University of New Mexico, Albuquerque, NM.

Nadkarni, N.M., Cushing, J.B., 2001. Lasers in the jungle: the forest canopy database project. Bulletin of the Ecological Society of America 82, 200–201.

Nottrott, R., Jones, M.B., Schildhauer, M., 1999. Using XML-structured metadata to automate quality assurance processing for ecological data. Proceedings of the Third IEEE Computer Society Metadata Conference.

Peterson, A.T., Vieglais, D.A., 2001. Predicting species invasions using ecological niche modeling: new approaches from bioinformatics attack a pressing problem. Bioscience 51, 363–371.

Poulin, J.S., 1995. Domain analysis and engineering: how domain-specific frameworks Increase Software Reuse Proceedings of CASE JAPAN'95, Tokyo, Japan, 12–15 July 1995, p. B-3/1-3. http://home.stny.rr.com/jeffreypoulin/Papers/JAPAN95/japan95.html.

Roberston, G., 1987. Geostatistics in ecology: interpolating with known variance. Ecology 68, 744–748.

Schroeder, W., Martin, K., Lorensen, B., 1998. The Visualization Toolkit (VTK) — an object-oriented approach to 3D graphics. Prentice Hall, p. 520.

Servilla, M., Brant, J., San Gil, I., Costa, D., 2006. Pasta: A network-level architecture for automating the creation of synthetic products in the LTER network. Technical report of the LTER Network Office, Department of Biology, University of New Mexico, Albuquerque, N.M.

Shaw, D.C., Chen, J., Freeman, E.A., Braun, D.M., 2005. Spatial and population characteristics of dwarf mistletoe infected trees in an old-growth Douglas-fir — western hemlock forest Canadian Journal of Forest Research 35, 990–1001.

Sheard, T., 2001. Accomplishments and research challenges in meta-programming. Invited Talk. Semantics, Applications, and Implementation of Program Generation 2001. LNCS, vol. 2196. Springer, Florence, Italy, pp. 2–44.

Storey, V.C., Goldstein, R.C., 1993. Knowledge-based approaches to database design. MIS Quarterly 17, 25–46.

Szyperski, C., Gruntz, D., Murer, S., 2002. Component Software — Beyond Object-Oriented Programming, 2nd edition. Addison-Wesley/ACM Press.

Van Pelt, R., Nadkarni, N.M., 2004. Development of canopy structure in *Pseudotsuga menziesii* forests in the southern Washington Cascades. Forest Science 50, 326–341.

Worm, B., Barbier, E.B., Beaumont, N., Duffy, J.E., Folke, C., Halpern, B.S., Jackson, J.B.C., Lotze, H.K., Micheli, F., Palumbi, S.R., Sala, E., Selkoe, K.A., Stachowicz, J.J., Watson, R., 2006. Impacts of biodiversity loss on ocean ecosystem services. Science 314, 787–790.

Yin, X., Han, J., Yang, J., 2005. Searching for related objects in relational databases. Proceedings of the 17th International Scientific and Statistical Database Management Conference 2005, pp. 227–236.

Zeman, L., Cushing, J.B., Nadkarni, N.M., Fiala, A.C.S., Pierce, C., 2006. The CanopyView Visualization Project. IEEE VISUALIZATION 2006, Baltimore, MD, USA. Poster and Demonstration.