

User-Oriented Knowledge Organization Systems

James F. Terwilliger, Lois M. L. Delcambre
Portland State University
{jterwill, lmd}@cs.pdx.edu

Judith Logan
Oregon Health & Science University
loganju@ohsu.edu

I. Overview

Data, as it resides in most databases or file systems, is incomprehensible to the average person. This problem arises largely because database schemas are not designed to be user-friendly; rather, they are intended to be efficient for storage and retrieval (Figure 1). Some databases serve as the back-end for sophisticated, domain-specific applications, such as electronic medical records or other clinical software (Figure 2). These software applications are obviously designed to be used by domain experts. We want to reuse the structure and content of the user interface for this kind of software application (used to capture data) to describe the data and allow domain experts who are not computer scientists to query the underlying data using this familiar structure for their data. In addition, we want to support users that would like to transform data in such databases into whatever form is necessary to perform a task, such as a statistical study.

ID	Attr	Value
101	2456	Thomas
101	3395	0
101	9845	AFTE_
101	0236	5
101	1263	Ja~2
102	2456	3
102	3395	1
102	3311	Mod
...

Fig. 1. An example data table from a database. A novice user will not know how to write queries over this data, and even database experts may not know exactly what the data means.

The screenshot shows a 'Finding Description' window with the following elements:

- Title Bar:** Finding Description
- Text Field:** Mucosal Abnormality
- Start Location:** Fundus (dropdown)
- End Location:** Body (dropdown)
- Folds:** (dropdown)
- Mucosa appears:**
 - Erosions
 - Hemorrhage (oozing)
 - Nodularity
 - Erythema
 - Mosaic / Scaly
 - Red Spots
 - Friable
 - Mottled
 - Subepithelial Hemorrh.
 - Granular
 - Ulcers
 - Portal HPT
 - Edema
- Diagnostics:**
 - Biopsy taken
 - Brushing done
 - RUT Results Pending
 - RUT Complete
- Path #:** Two input fields
- RUT Results:** (dropdown)
- ICD9:** (dropdown)
- Comments:** (text area)
- Toolbar (Right):**
 - Image - F2 (with camera icon)
 - Therapy/Dx Test-F9 (with red cross icon)
 - Save - F10 (with green checkmark icon)
 - Delete - Esc (with red X icon)

Fig. 2. A screen from an application that records clinical data – in this case, endoscopy reports. A domain expert, such as a clinician or medical informatics specialist, will be able to understand the terms on the screen and can learn to use the software.

Our research introduces two knowledge organization systems intended to allow knowledgeable yet simple data retrieval and analysis: one (called a *g-tree*) that is generated automatically from a software user interface, and one (called a *user schema*) that is built manually by a domain-expert user. *Classifiers* provide relationships between a *g-tree* and a user schema.

II. G-Trees

A custom-built user interface typically represents the user's understanding of what that data means. We describe the content of a user interface in a GUAVA (GUI As View) tree, or *g-tree* (Figure 3). Each window or control in the application corresponds to a node in the *g-tree*, where the root node of the tree represents whatever screen appears first when the application is launched. Each *g-tree* node also contains context information for its control, such as the surrounding text and the tool-tip help bubble that appears on mouse-over.

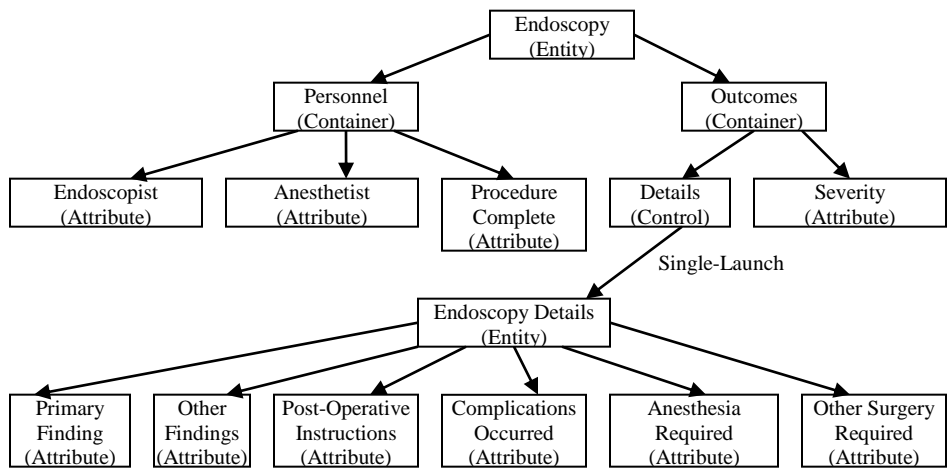
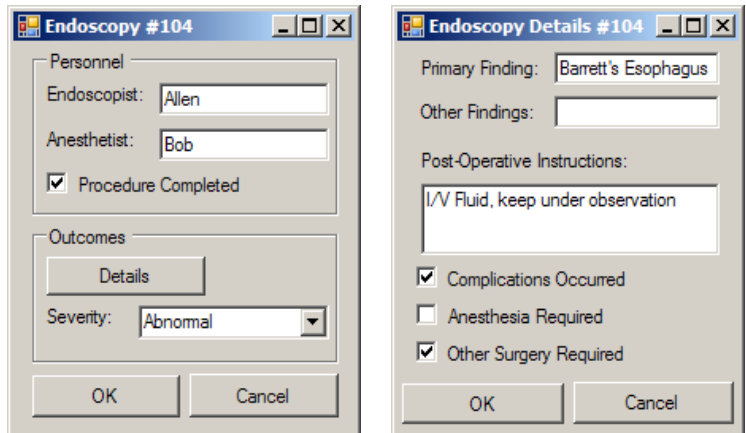


Fig. 3. An example application and its corresponding g-tree. In addition to what is shown in the figure, each node contains context information about how that control appears and operates on the screen.

Behind the g-tree we have defined a *channel*, a sequence of operators that describe how to transform data from the underlying database to how it is displayed in the user interface. The same channel can also be used to answer queries (Figure 4). Because a g-tree contains all of the information about the look, feel, and content of an entire application, a visualization of a g-tree looks exactly like the original application, except that it is a query interface rather than a data entry interface (Figure 5). One uses this query interface to write a query, and the channel generates the SQL to retrieve the correct data.

Finally, because the structure of the g-tree is hierarchical, users can also search the g-tree itself for information. A user can search for ‘medications’ and find all of the places in the application the word ‘medications’ appears, as well as the path of windows and buttons one uses to get there.

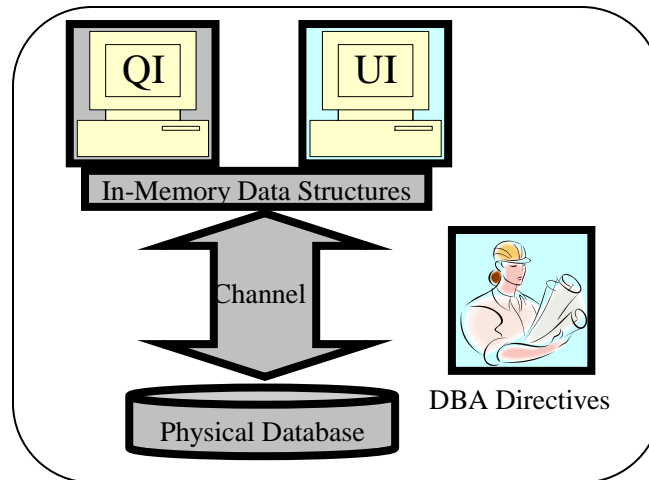


Fig. 4. An overview of the GUAVA framework. The channel translates data in the physical database to be displayed in the user interface (UI). However, it also translates queries from a query interface (QI) into SQL against the physical database based on high-level directives by a database administrator.

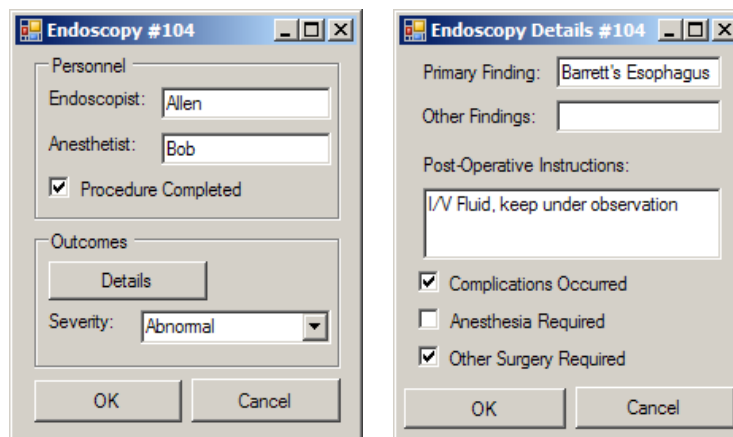


Fig. 5. A mockup of how one can visualize a g-tree. It looks exactly like the original application. However, instead of adding new data, this visualization queries data. The figure above would find all endoscopies whose endoscopist was Allen, whose anesthetist was Bob, etc. The channel for the application translates the above query into SQL.

III. User Schemas

A g-tree can make data visible, searchable, and understandable. It does not necessarily make the data useful. The software may not capture data in exactly the form required by the user (i.e. by the domain expert, the analyst). For example, the user may wish to perform a statistical analysis on smoking habits. Smoking data, however, may not be captured in the correct units or with the correct resolution as required by the study. In addition, the study may require identification of a subset of patients tracked by the software.

A *user schema* represents a user's view of the world, recording the entities, attributes and relationships that the user would like to see. It is similar to an ER-diagram,

except that the user can specify multiple domains for each attribute (Figure 6). This capability allows the user to re-use a user schema in subsequent studies even if, for instance, the subsequent studies require different representations of one attribute but have identical needs for all other data. However, a user schema is not a global schema; several user schemas may exist for the same data if different users have vastly different needs.

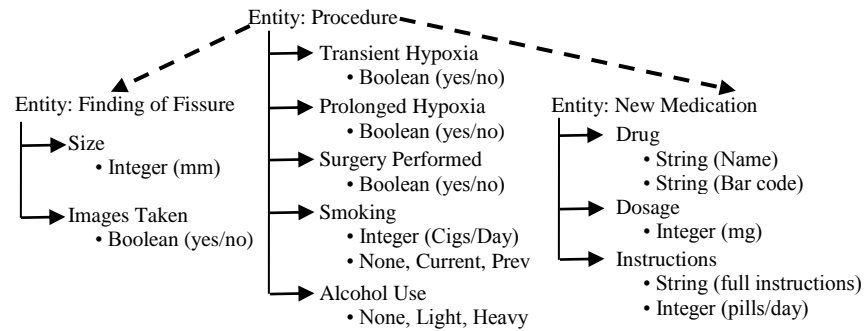


Fig. 6. A user schema. Entities have attributes, which in turn have domain(s) that correspond to different ways to represent them. The dashed lines indicate “has-a” relationships between entities.

A *classifier* relates an element in a user schema with one or more elements in a g-tree (Figure 7). A classifier provides a semantic relationship between the two knowledge organizations, as well as operational instructions to transform data from one organization to the other. Classifiers are specified manually by the user and represent the needs and assumptions of the user at a particular time. In addition, classifiers are declarative and can be annotated by the user; thus they can be recalled by subsequent users that may wish to reproduce results and data transformations.

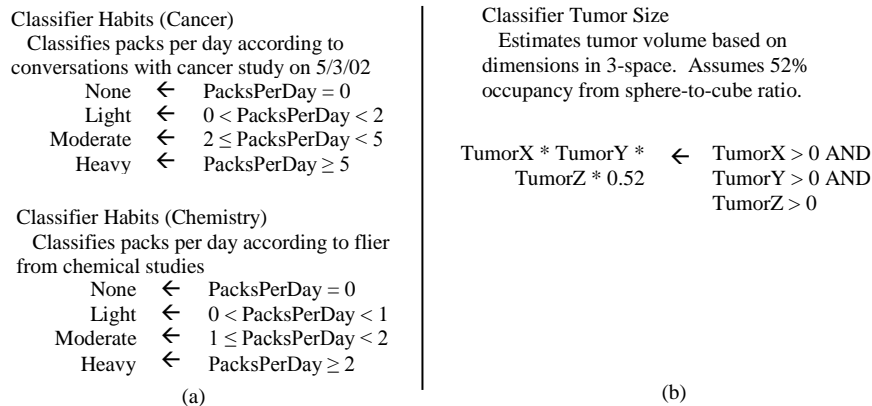


Fig. 7. Example classifiers. Two classifiers (a) can relate data from a g-tree to the same domain for different studies. Each of these classifiers is valid with respect to a study, and analysts can reuse either for future studies. Another classifier (b) shows how to write classifiers that refer to more than one g-tree node.

IV. Methodology

The GUAVA framework is nearing the level of maturity where we can initiate our first user tests. The first set of findings will be available in time for the NKOS workshop. The testing will explore the validity of two hypotheses:

- A g-tree alone is sufficient for a domain expert to understand the meaning of data in an underlying database
- A g-tree alone is sufficient for a domain expert to write meaningful queries against an underlying database

We intend to mine the studies that our research group has run over the past several years for a collection of queries over clinical data. We will then implement a g-tree over a user interface that none of our researchers or software developers have used before. We will have users try to answer the queries using only the g-tree and we will have users work with the developers to write queries using SQL over the underlying database, which is effectively the current process that we use. If our hypotheses are true, the g-tree users will be able to retrieve the same data as the SQL group in the same amount of time or less. We will also capture their qualitative reactions to the use of the g-tree in a post-test survey that asks whether they felt that using a g-tree was easy, whether they found it useful to see the entire set of data items (through the g-tree), and whether they felt more confident about the final result. The implementation of user schemas and classifiers is not as mature, in part because we are waiting for the results of the g-tree testing before we finalize the theory behind our classifier language.