Capturing Users' Everyday, Implicit Information Integration Decisions

David W. Archer **Lois M. L. Delcambre**

Department of Computer Science
Portland State University
Portland, OR 97207

{darcher, lmd} @ cs.pdx.edu

Abstract

Integration of large databases by expert teams is only a small part of the data integration activities that take place. Users without data integration expertise very often gather, organize, reconcile, and use diverse information as a normal part of their jobs. Often, they do this by copying data into a text file or spreadsheet. In doing so, they make significant data integration decisions. They often express a mental model, or schema, over their data. They organize data to describe real-world entities. They reconcile redundancy and disagreements in their data. Such integration is both ubiquitous and not generally supported by experts and tools available for large integration efforts. We seek to capture and make explicit the user's mental model, and the attribute and entity correspondences they express, during these activities. This paper contributes the definition of a set of functions that support this type of data integration, a conceptual model to support these functions, and an associated simple tool that supports data integration by end-users in an entity-centric way, with an extensible schema, that makes the user's job easier.

Keywords: information integration, entity resolution, data correspondence, superimposed information.

1 Introduction

We often consider data integration to be the province of the DBA and the integration expert, aided by specialized tools. However, data integration is often performed as part of everyday user tasks. End users gather, organize, reconcile, and use information from diverse sources all the time. They gather information from office documents, the Web, e-mail messages, and other places in order to inform, to influence, and to make decisions. Sometimes users gather information using hardcopy. Increasingly, users take advantage of computers to do this work, often cutting and pasting bits of information into a spreadsheet or text file.

The information gathered, and the semantics given it, are task-dependent. However, our experience shows that users often gather information in much the same way. They collect information about real-world objects or concepts: that is, information gathering tends to be entity-centric. They label information to keep track of its meaning (for the most part users know both the meaning of each piece of information and the entity which it describes). They tend to organize gathered data in simple, tabular form. They frequently make on-the-fly decisions about the data: they combine what initially appear to be disparate real-world objects or concepts into single objects; they combine what initially appear to be different characteristics into single attributes; and they resolve conflicting items of information.

Each of these actions helps to superimpose the user's conceptual model on the gathered information, and adds significant value to the raw data. We seek to capture the information labeling and information integration decisions that are expressed during these activities, in the form of schema definition, attribute correspondences, entity correspondences, and attribute value conflict resolutions. In this research, we seek to:

– make the user's tasks of using and manipulating information drawn from various sources easier,
– provide direct support for tracking the lineage of information extraction, use, and re-packaging, and ultimately,
– exploit the collective user integration information to help solve the general problem of massive information integration.

In this paper, Section 2 describes our understanding of user data integration activities, our refinement of this understanding into a set of specific user actions, and our definition of a conceptual model to support these actions. Section 3 formalizes these user actions into a set of functions on our conceptual model, encompassing entity resolution, attribute resolution, and attribute value conflict resolution. Section 4 describes a simple tool for evaluating both how to support users in performing these tasks, and how to capture the integration metadata they express during these tasks. Section 5 discusses related work from the literature. Section 6 summarizes our contribution and discusses future efforts.

2 Conceptual Model

In order to examine how users gather and organize information, we rely on ten years of one author's direct observation and participation in managing engineering organizations and running a successful software business.

Common Projects	Source Identification	Data Integration	Sense-making	Deployment
Project line strategy and roadmap creation	High	High	Medium	Low
Project Prioritization	High	High	Medium	Low
Resource assignment	Low	Medium	Medium	High
Project scheduling	Medium	High	High	Medium
Project tracking	Low	High	Low	Low
Budgeting	Low	High	Medium	Medium

Table 1: Relative effort required for significant tasks in common engineering management projects.

We reviewed common, repeated projects and the information integration activities common in each. We summarize the most common projects, which taken together account for a significant majority of on-the-job effort, in Table 1. For each of these common projects, we describe the amount of effort required for the following information-related tasks:

– **Identifying sources** from which to gather information
– **Integrating** (gathering and organizing) the information
– **Sense-making** and decision-making
– **Deployment** of the information for communication

This table shows that data integration activities are significant in the overall management effort in the projects shown. Next, we identified common user actions that comprise the tasks from Table 1, and related these to corresponding data integration activities, as shown in Table 2.

User action	Data integration task
Copying items from various sources and pasting into a "worksheet" form	Data collection
Creating and deleting columns for gathered items with similar semantics	Schema creation/modification
Creating and deleting rows of values for the real-world objects of interest	Data instantiation
Merging duplicate columns and resolving items which disagree during this process	Attribute resolution, attribute value disambiguation
Merging rows and resolving items which disagree during this process	Entity resolution, attribute value disambiguation
Tracking integration decisions	Annotation/Not typically supported
Undoing merges when needed	Not typically addressed in data integration

Table 2. Mapping user actions to data integration functions

Users often make use of copy-and-paste and other simple tools to accomplish the tasks and actions outlined above. Our effort seeks to make this job easier, improve access to the context of gathered information, provide access to the history of user decisions about the data, and maintain both expressed schema and decision history for potential use in other ways. In support of this, we have developed a conceptual model called CHIME (Capturing Human Intension Metadata with Entities) that supports creation and manipulation of a single, virtual relation, or table, where rows correspond to real-world entities and columns correspond to attributes of these entities. As shown in Figure 1, our model consists of five entity types. We discuss three of these here: Column proxies, Row proxies, and Cell proxies.

Column proxies represent the column heading, or schema, of the user's view of the conceptual model. Each column has a name, a data type common to all cells in the column, and a flag to indicate whether the column is visible or hidden. Row proxies represent the rows of the user's view of the CHIME model. Each row has a unique identifier and a flag to indicate whether the row is visible or hidden. Cell proxies represent individual attribute values. Each cell has a unique identifier and is related to a *mark* [9], that is, an encapsulated address from which its value may be retrieved in a source document. A cell is related to the row of which it is a member via the "describes" relationship set. A cell is related to the column of which it is a member via the "member_of" relationship set. Any number of cells may be related to a row, and any number of cells may be related to a column. However, a cell may be related to only one row and one column via these relationship sets, giving the resulting structure its tabular form.

If a cell proxy belongs to a row that has been merged from two others, it participates in the "has_row_parents" relationship set with the two other rows from which the merged row was created. Because our model supports only pair-wise entity resolution, cells from a single row may be related only to a given pair of parent rows in this way. Similarly, if a cell proxy belongs to a column that has been merged, it participates in the "has_column_parents" relationship set with the two other columns from which the merged column was created. The user sees only a portion of the entity sets and relationship sets in the diagram. For the visible rows, columns, and cells, the user sees the row proxy, column proxy, and cell proxy entity sets, along with the Value portion of the item entity set. However, the UID attributes of the cell proxies and row proxies are hidden from view. As a result, the

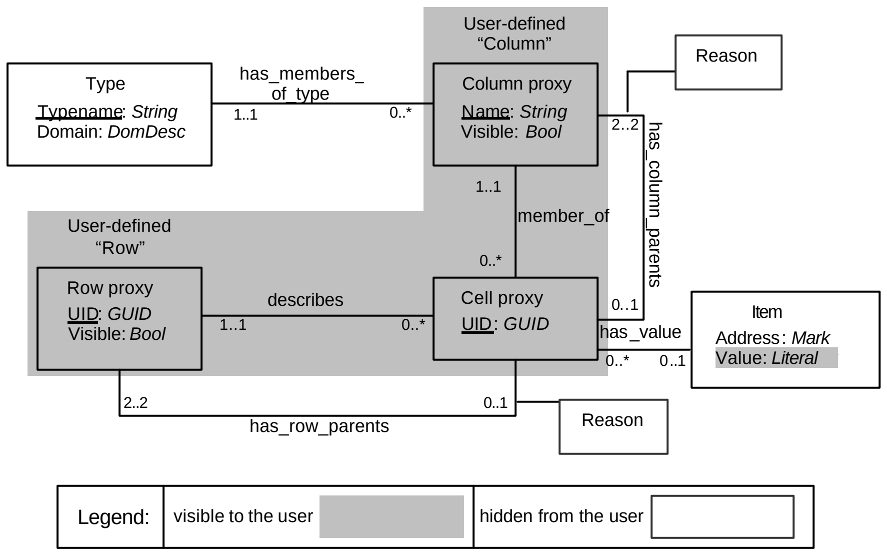


Fig. 1. E-R diagram for managing entity-centric schema and data

user sees only an intuitive tabular representation of the entity sets and their relationships.

3 Functional Model

In order to explore the data integration tasks outlined in Table 2 in the context of our conceptual model presented in Section 2, we wrote an emulator in Haskell. We built an emulator because it enabled a degree of formalization, allowing us to verify our ideas as a first step toward developing a tool for evaluating how to achieve the goals outlined in Section 1. Haskell is a polymorphically typed, pure functional language. Haskell was attractive for our purpose for several reasons. Haskell's polymorphism makes abstractions easy for both data types and functions. This allowed us to focus on what we needed to represent and what our functions needed to achieve, rather than focusing on how to implement them. These abstract definitions are also very concise, making it easier to reason about our code. The recursive approach to iteration required in functional languages, along with strong support for the lists we used to model rows of data, made data manipulation easy, as well.

Though our conceptual model focuses on data integration, our Haskell implementation also includes the data gathering functions discussed in Table 2. In Figure 2, we describe the data structures used in emulating the conceptual model shown in Figure 1.

The primary structure of interest is the Cellproxy. Rows are lists of Cellproxies, providing a natural implementation of the "describes" relationship from Figure 1. A database is simply a list of rows. We model the "member_of" relationship implicitly, by keeping Cellproxies in the same order in each row and including the column name in each. The Cellproxy models the "has_value" relationship by including the cached value of the data item and a string representing the mark to the data item. The AttrVal type used for the data item value also specifies the data item's type, thus modeling the "has_members_of_type" relationship. A union type,

```
data Cellproxy =
Attribute {attributeName:: String,
           attributeValue:: AttrVal,
           mark:: String,
           rowMergeHistory:: Merge,
      colMergeHistory:: Merge,
           cellVisible:: Bool }
        deriving (Eq, Show)

data AttrVal = Str String
        | Int Integer
        | Flag Bool
        deriving (Eq, Show)

data Merge = NeverMerged
|RowMerged String Integer Integer
|ColumnMerged  String String  String
        deriving (Eq, Show)
```

Fig. 2. Haskell definition of the data structures maintained by CHIME.

called Merge, is used in Cellproxies to implement the "has_row_parents" and "has_column_parents" relationships. If a merge has been performed, the Merge structure denotes the reason as well as the source rows (columns) involved. Each cell also contains a flag to control visibility to the user.

4 A Tool for Investigation

We introduce our prototype tool, intended to help us evaluate how users integrate information and how we might capture their decisions, by way of an example project: researching flat-panel monitors. Based on browsing the web, and on advice in e-mail from colleagues, a user constructs a list of models to choose from and a list of criteria she will use to make a selection.

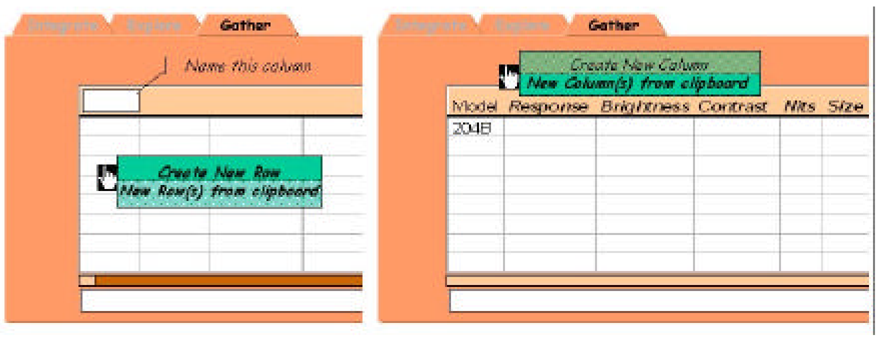


Fig. 3. Pasting data into the CHIME worksheet.

The user gathers data from various on-line reviews, product catalogs, and marketing web sites by using the familiar copy-and-paste operation. Behind the scenes, the system creates a mark to each selected item. In CHIME, the user creates a new row, as shown at left in Figure 3, to represent each new monitor of interest. The system retains both the data item of interest and the mark to its source context.

The user can also copy (and mark) a list of desired features, for example from an e-mail sent by a colleague, and use the pop-up menu function "New columns from clipboard", as shown at right in Figure 3, generating a column for each of the items in the list. The user adds new rows and new columns as needed. At any time, the user can simply mouse-over data items to show them in the context of their source document.

If the user decides that two rows in her worksheet represent the same product, she simply selects the two rows and merges them to create a new row, as shown in Figure 4. The system automatically identifies colums where data items in the two rows mismatch and allows the user to select which value is most appropriate. If the user realizes that two columns represent the same attribute, she can merge them in the same way. By using the Explore tab, the user can review the history of merge decisions that led to the current state of her worksheet and reverse those decisions if needed.

The user can also create new information products using the data in CHIME. The user selects items or arrays of data from her worksheet and pastes them into an e-mail message. Behind the scenes, the CHIME system injects into the message not only the data values selected, but also the marks that provide access to both the worksheet and the original data sources of each data item. This allows the recipient of the message to review the data and draw his or her own conclusions.

5 Related Work

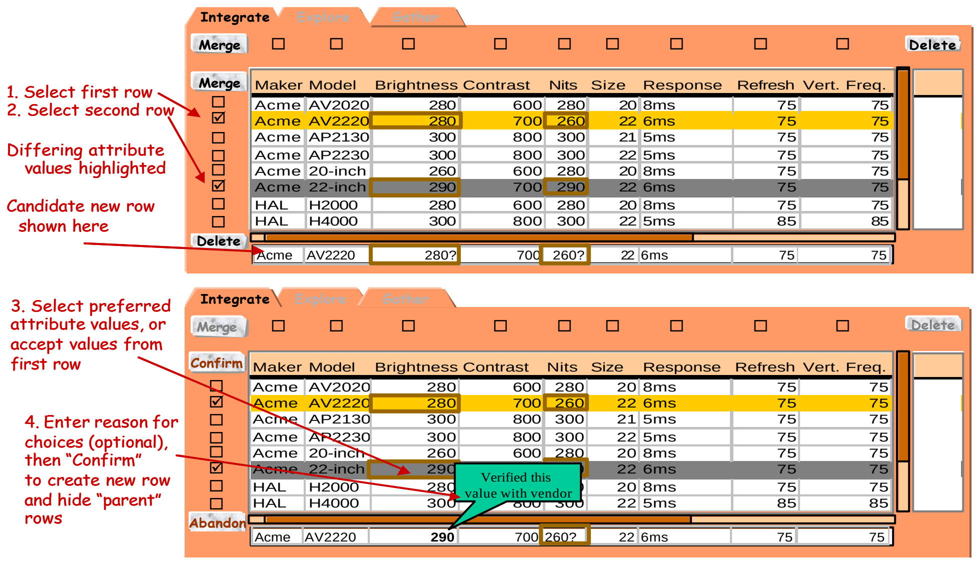
Much of the literature in information integration focuses on integration of data from structured sources, often performed as a "bulk" operation. The goal of Clio [8], for example, is to integrate existing databases by matching schemas and generating inter-schema mappings with human assistance. The schema engine of the Clio suite provides a graphical interface that displays schema and example data to assist the user in understanding schema semantics. Although Clio attempts to make the schema integration task easier, for example by providing data examples to help the user understand correspondences, it is a tool intended for use by database specialists. Our tool is intended for use by end-users. Also, we capture data integration decisions, including entity resolution and attribute value conflict resolution, whereas Clio supports only schema integration.

Windik [13] and Bernstein and Melnik's extensions to the Microsoft BizTalk Mapper [1], are similar to Clio in that they focus only on schema matching.

MOBS [11] seeks to retrieve information relevant to a real-world *entity* from both structured and unstructured sources, given certain initial information about the entity. Seeded with some initial semantic mappings, the MOBS approach relies on deploying these mappings to a community of users who then augment the available metadata. Our long-term vision is similar to MOBS. However, our approach differs in that MOBS emphasizes a system-driven approach to gathering entity-related information, while our approach emphasizes assisting users as they gather entity-related data for their own purposes.

SEMEX [3] uses a pre-defined but extensible ontology to construct an entity-centric, logical view of a user's desktop by constructing a database of objects and associations between them. In this way, SEMEX provides access to data stored in multiple applications without imposing a data organization that is application-centric. The SEMEX approach differs from ours in that SEMEX takes in a user's entire personal information space and automatically builds its logical view, while our approach focuses on user selection of data and user expression of schema and associations.

Superimposed Schematics [2] uses E-R modeling constructs and marks [9] to superimpose an application-specific conceptual structure over unstructured information. With this approach, a user can browse the unstructured base layer information in the context of the superimposed schematic. Our work differs from this approach in that we allow the user to express the conceptual structure to be superimposed on data of interest, rather than pre-defining the conceptual structure.


1. Select first row
2. Select second row
Differing attribute values highlighted
Candidate new row shown here
3. Select preferred attribute values, or accept values from first row
4. Enter reason for choices (optional), then "Confirm" to create new row and hide "parent" rows
Verified this value with vendor
Integrate
Explore
Gather
Merge
Delete
Confirm
Abandon

	Maker	Model	Brightness	Contrast	Nits	Size	Response	Refresh	Vert. Freq.
☐	Acme	AV2020	280	600	280	20	8ms	75	75
☑	Acme	AV2220	280	700	260	22	6ms	75	75
☐	Acme	AP2130	300	800	300	21	5ms	75	75
☐	Acme	AP2230	300	800	300	22	5ms	75	75
☐	Acme	20-inch	260	600	280	20	8ms	75	75
☑	Acme	22-inch	290	700	290	22	6ms	75	75
☐	HAL	H2000	280	600	280	20	8ms	75	75
☐	HAL	H4000	300	800	300	22	5ms	85	85
	Acme	AV2220	280?	700	260?	22	6ms	75	75

	Maker	Model	Brightness	Contrast	Nits	Size	Response	Refresh	Vert. Freq.
☐	Acme	AV2020	280	600	280	20	8ms	75	75
☑	Acme	AV2220	280	700	260	22	6ms	75	75
☐	Acme	AP2130	300	800	300	21	5ms	75	75
☐	Acme	AP2230	300	800	300	22	5ms	75	75
☐	Acme	20-inch	260	600	280	20	8ms	75	75
☑	Acme	22-inch	290	[illegible]	[illegible]	22	6ms	75	75
☐	HAL	H2000	280	[illegible]	[illegible]	20	8ms	75	75
☐	HAL	H4000	300	800	300	22	5ms	85	85
	Acme	AV2220	**290**	700	260?	22	6ms	75	75

Fig. 4. Merging Rows in CHIME

In addition, superimposed schematics do not provide for data integration as CHIME does.

Several personal information management approaches (including CREAM [6], WiCK [4], SemanticWord [12], Semantic neighborhoods [5], and MIT's Haystack[7]) provide varying capabilities for users to superimpose mental models over their data. However, we know of no data integration capabilities in these tools such as those provided by CHIME.

6 Conclusions and Future Work

We have observed that data integration is a common activity for a wide variety of users who are not experts in data management. In the conceptual model we present here, we capture and make explicit the user's mental model as well as any information integration decisions they make along the way. We have implemented an emulator to demonstrate important properties of the conceptual model. We have presented one tool that implements these functions and supports further investigation. We plan to prove the correctness of the key properties of the functions described here. In addition, we intend to deploy the described application to study the benefits of this approach.

If a user community adopts the practice of using tools that embody our conceptual model, then the ultimate goal of our research is to exploit this end-user integration information to integrate information sources – a difficult and very large-scale problem. One benefit of our approach is that rich and detailed integration information is captured essentially for free – as end-users do their day-to-day tasks. Another benefit is that we simultaneously capture schema integration information as well as data resolution information. Finally, we observe that our captured data, since it is associated with its original source data through the use of marks, is associated with rich contextual information extracted from the source files. We believe that we may ultimately be able to use this context information to facilitate schema creation, entity resolution, attribute resolution, and schema matching while integrating databases.

7 References

[1] Bernstein, P., Melnik, S. (2006): Incremental Schema Matching. In *Proceedings of the 32nd International Conference on Very Large Data Bases* Seoul, Korea.

[2] Bowers, S. Delcambre, L., Maier, D. (2002): Superimposed Schematics: Introducing E-R Structure for In-Situ Information Selections. In *Proceedings of the 21st International Conference on Conceptual Modeling*. Tampere, Finland.

[3] Cai, Y., Dong, X., Halevy, A., Liu, J., Madhavan, J. (2005): Personal Information Management with SEMEX. In *Proceedings of the ACM SIGMOD International Conference on Management of Data* Baltimore, Maryland, USA.

[4] Carr, L., Miles-Board, T., Wills, G., Woukeu, A., Hall, W. (2003): Towards a Knowledge-Aware Office Environment. In *Proceedings of 14th International Conference on Knowledge Engineering and Knowledge Management*.

[5] Gersh, J., Lewis, B., Montemayor, J., Piatko, C., Turner, R. (2006) Supporting Insight-Based Information Exploration in Intelligence Analysis. *Communications of the ACM*, Vol. 49, No. 4, pp. 63-68.

[6] Handschuh, S., Staab, S. (2002): Authoring and Annotation of Web Pages in CREAM. In *Proceedings of WWW2002*. Honolulu, Hawaii, USA.

[7] Karger, D., Bakshi, K., Huynh, D., Quan, D., and Sinha, V. (2005): Haystack: A general-purpose information management tool for end users of semistructured data. In *Proceedings of the Second Biennial Conference on Innovative Data Systems Research*. Asilomar, CA.

[8] Miller, R., Hernandez, M., Haas, L., Yan, L., Ho, C., Fagin, R., Popa, L. (2001): The Clio Project: Managing Heterogeneity. SIGMOD Record, 30(1):78--83.

[9] Murthy, S., Maier, D., Delcambre, L., Bowers, S. (2004): Putting Integrated Information in Context: Superimposing Conceptual Models with SPARCE. In *Proceedings of the First Asia-Pacific Conference on Conceptual Modeling.* Dunedin, New Zealand. pp. 71-80.

[11] Sayyadian, M., Shakery, A., Doan, A., Zhai, C. (2004): Toward Entity Retrieval over Structured and Text Data. In *Proceedings of the first Workshop on the Integration of Information Retrieval and Databases*. Sheffield, UK.

[12] Tallis, M. Semantic Word Processing for Content Authors. (2003): In *Proceedings of the Knowledge Markup & Semantic Annotation Workshop*. Florida, USA.

[13] Venugopalan, S., Tamma, K. Applicability of Universal Relation to Data Integration. http://www.cs.wisc.edu/~vshree/cs764/Windik.pdf. Accessed March, 2007.