A cognitive support framework for ontology mapping

Sean M. Falconer and Margaret-Anne Storey

University of Victoria, Victoria BC V8W 2Y2, Canada
{seanf, mstorey}@uvic.ca

Abstract. Ontology mapping is the key to data interoperability in the semantic web. This problem has received a lot of research attention, however, the research emphasis has been mostly devoted to automating the mapping process, even though the creation of mappings often involve the user. As industry interest in semantic web technologies grows and the number of widely adopted semantic web applications increases, we must begin to support the user. In this paper, we combine data gathered from background literature, theories of cognitive support and decision making, and an observational case study to propose a theoretical framework for cognitive support in ontology mapping tools. We also describe a tool called COGZ that is based on this framework.

1 Introduction

Ontologies have seen increasing use in academia and industry, especially as work on the semantic web grows and evolves. A growing reliance on ontologies brings with it many challenges. One challenge is resolving heterogeneity among domain-related ontologies or *ontology mapping*. This is a critical operation for information exchange on the semantic web. Ontology mapping research is receiving increased attention. Mapping contests exist to compare the quality of ontology matchers [9], a mapping API has been proposed [10], and workshops have been organized to discuss this issue. However, the research emphasis has primarily been on the automation of this process, even though most ontology mapping processes require user involvement.

The heterogeneous data mapping problem is not restricted to ontologies and the semantic web. According to Bernstein *et al.* [2], every database research self-assessment has listed interoperability of heterogeneous data as one of the main research problems. Coping with data heterogeneity is still one of the most time-consuming data management problems. Given that this problem is well-known and extensively studied, why is it so difficult to generate mappings between ontologies or any other data source? In addition to different world views and disparate user needs, there are also issues of language and constraints on available data formats. Languages are known to be *locally ambiguous*, meaning that a sentence may contain ambiguous portions unless considered in the context of the whole sentence. Humans use detailed knowledge about the world, connect sentences and fill in missing parts, and infer what someone means (even if he/she did not actually say it) in order to disambiguate [1]. Also, the data format used (e.g., OWL, RDF, XSD) constrains the expressiveness of the data representation. These issues make it unlikely that we will develop fully automated mapping procedures, consequently we concur that "[a] human must be in the loop." [2]

Research has largely ignored the issue of user intervention and instead has focused on algorithms to compute candidate mappings. Many research tools provide only file dumps of potential mappings [7] or interfaces that quickly become unmanageable [23]. The responsibility of working through the mass of data computed by these algorithms is left to the user. This can be extremely difficult, requiring tremendous patience and an expert understanding of the ontology domain, terminology, and semantics. Contrary to this research trend, we feel that since the human is critical to the success of the mapping procedure, we must address and emphasize user needs. We believe that we must first try to understand the decision making processes used in the mapping task. By understanding these processes, *cognitive support* can be introduced to the tools to reduce the *cognitive load* experienced by users. We believe that user interfaces that offer more effective cognitive support will provide greater productivity gains than improvements to precision and recall in matching algorithms.

The paper is organized as follows. We begin by discussing ontology mapping tools and research trends in ontology mapping. This is followed by a discussion and analysis of cognitive support and decision making theories and how these relate to ontology mapping. Next, we discuss an observational case study where we observed users performing ontology mappings. The data and analysis from this study is combined with our literature review of mapping tools and cognitive support theories to create a theoretical framework for ontology mapping. The framework describes the process and concepts central to human-guided ontology mapping. Following this, we describe how this framework was used to guide the design of the COGZ ontology mapping tool.

2 Mapping tools

Ontology mapping is a prerequisite for many semantic web applications including instance mediation across web sites, agent communication over the Internet, web service integration, and query and answer rewriting. The quality of these applications depends largely on the underlying mapping. Many tools exist to help compute mappings. FOAM (Framework for Ontology Alignment and Mapping) [6] performs fully or semi-automatic alignment of two or more OWL ontologies. The alignment algorithm uses heuristics to compute similarity between terms and individual entities. The user supplies a parameter file that specifies alignment location, an optional file of pre-known mappings, and algorithm specifications. The FOAM tool saves computed mappings along with a score representing the confidence in the mapping. FOAM asks the user to verify certain mappings and the user can specify in the parameter file the maximum number of questions that should be posed.

Chimaera [19] is a tool that supports ontology merging and diagnosis. The system has a web-based interface where the user interacts with web forms to upload ontologies, select algorithm parameters, and merge similar ontology entities. The merge algorithm produces a candidate list of mappings as matching terms, based on term name similarity, term definitions, possible acronyms and expanded forms, and suffix matching.

COMA++ [5], PROMPT [21], AlViz [17], and OLA [11] provide graphical user interfaces. COMA++ automatically generates mappings between source and target schemas (XML or OWL), and draws lines between matching terms. Users can also define their own term matches by interacting with the schema trees. Hovering over a potential mapping displays a confidence level about the match as a value between zero and one.

PROMPT, developed by the Stanford Medical Informatics group, was designed as a plugin for the popular ontology editor Protégé. The plugin supports managing multiple ontologies including ontology differencing, extraction, merging, and mapping. The user begins the mapping procedure by specifying a source and target ontology. PROMPT then computes an initial set of candidate mappings based largely on lexical similarity between the ontologies. The user works with this list to verify the recommendations or create custom mappings missed by the algorithm. Once a user verifies a mapping, PROMPT's algorithm uses this to perform analysis based on the graph structure of the ontologies. This usually results in further mapping suggestions and the process is repeated until the user deems the mapping complete. Similarly to PROMPT, AlViz is a plugin for Protégé to do ontology mapping. However, the tool is in an early research phase.

OLA (OWL Lite Alignment) provides automated alignment and an environment for manipulating alignments [11]. OLA supports parsing and visualization of ontologies, automated computing of similarities between entities, manual construction of alignments, visualization of alignments, and comparison of alignments. The mapping algorithm finds matches by analyzing the structural similarity between the ontologies using graph-based similarity techniques. This information is combined with label similarity measures to produce mapping correspondences.

Evaluations of these tools have mostly focused on comparing mappings produced with known mappings. PROMPT is an exception in that the authors performed user evaluation experiments [20]. The experiment evaluated tool-generated mapping suggestions by having several users merge two ontologies. The number of steps required, suggestions followed and not followed, and resulting ontologies were all recorded. Precision and recall was used to evaluate the quality of the suggestions. Similarly, Lambrix and Edberg [16] performed a user evaluation of PROMPT and Chimaera [19] for the specific use case of merging ontologies in bioinformatics. The participants were given a number of tasks to perform, a user manual on paper, and the software's help system for support. They were also instructed to "think aloud" during the experiment while an evaluator took notes. Afterwards, the users completed a questionnaire about their experience. The tools were evaluated with the same precision and recall measurements used in the previously described PROMPT experiment, while the user interfaces were evaluated using the REAL (Relevance, Efficiency, Attitude, and Learnability) [18] approach. Under both criteria, PROMPT outperformed Chimaera, but the participants found learning how to merge ontologies in either tool was equally difficult. The participants found it particularly difficult to perform non-automated procedures in PROMPT, such as creating user-defined merges.

Other than these examples, little research has looked at the user side of mapping. We propose that more comprehensive experiments that focus on how people perform mappings will lead to productivity gains in schema matching [2].

3 Cognitive support and decision making

Cognitive support refers to the assistance that tools provide to humans in their thinking and problem solving [30]. We often rely on external artifacts (tools) to support cognition, e.g. a sticky note can be used as an external memory source - a reminder about a task we need to complete. In software tools, software artifacts (e.g. menus, search, term completion) can be introduced to support the human user's cognition.

The relationship between thinking and artifacts is not new. Humans tend to adapt their environment to the activities they wish to complete [30]. For example, Kirlik observed that short-order cooks develop strategies for using their environment to ease their mental work. They "may organize the placement of meats in order of doneness, [and] may lay out dishes or plates to serve as a temporary external memory of orders to be prepared" [15, pp. 84]. The goal of cognitive support within a software system is to offload some of the user's cognitive processes involved in performing a task to the software. This can reduce the number of items that a user must internally track and process, allowing them to concentrate their expertise on other parts of the task.

There is a tendency to support users by automating tasks. Full automation is total cognitive support, relieving the user of all cognitive responsibility. However, some tasks are too difficult to fully automate, and the user is left to deal with the complexity of the task. Automation sometimes introduces complexity or frustration, e.g., the endless menu options in automated phone systems. Brainbridge observed that automation provides the least assistance when we need it most, as generally, we can only automate rudimentary tasks [3]. This is supported by the previously discussed user evaluation of PROMPT and Chimaera. The participants noted that performing non-automated procedures with PROMPT was difficult. This is also true of other mapping tools, which can only automatically discover the simple mappings. It is left to the user to manually create the rest of the mappings with little or no tool support.

In semi-automatic ontology mapping, the automated procedure helps the user heuristically search for mappings by providing suggested or candidate matches. However, for the tool to be effective, it must also support the user by reducing the complexity of analyzing suggested mappings. There has been a growing realization in the sciences that coping with complexity is central to human decision-making [26]. There are several theories of decision making and thinking process that are relevant to solving ontology mapping problems. For example, research has demonstrated that people often solve problems by selective, heuristic search through large problem spaces and large databases [26]. Experts, such as chess masters, use these techniques to solve complex problems. They cannot analyze all possibilities from one chessboard state so they must prune the search space using heuristics. This type of decision-making process is known as the *heuristic-systematic persuasion model* [28].

Related to heuristic search and contextual cues is *filter theory*, which suggests that we make decisions through a series of selection filters [14]. For example, a doctor may begin by asking a patient about their general symptoms and then narrow the focus of the questions based on which diagnoses match the symptoms. In ontology mapping, this decision model can be supported by an overview of the generated mappings with support for user-driven filtering and searching.

Perceptual contrast effect describes the effect that humans often make decisions by comparing and contrasting a decision item with a reference item [24]. In mapping, users can compare an unknown mapping to existing mappings, which act as reference items to help reinforce the decision the user is making. Decision makers actively build a *confirmation bias*, seeking confirmation that they have made a good decision [27]. Tools must provide interfaces to help with the confirmation process (e.g., identifying the local semantic structure of a term in a mapping, the properties of that term, and other possible mappings for the term).

Multi-attribute choice describes the decisions we make when comparing situations/objects with multiple attributes [8]. We tend to compare shared attributes or focus on differences in order to come to a decision. For example, when deciding which computer to purchase, we compare the shared features or attributes of the two machines as well as the differences. In ontology mapping, users may compare shared and unique properties of a class to determine if two class labels represent the same concept.

Each of these theories of decision making contributes to our understanding of how users make mapping decisions and how tool support can assist in this process. In the next section we describe an observational case study that further investigates the user decision making process during ontology mapping.

4 Observational case study

4.1 Study setup

We observed users performing mappings with two different tools, COMA++ and PROMPT, which were selected for several reasons. First, they both support user-interaction and a graphical user-interface. However, the tools support this interaction differently. COMA++ computes a full mapping between the ontologies and the user then interacts with the ontology trees to remove invalid mappings and create missing mappings. PROMPT produces a list of candidate mappings that the user verifies by completing the suggested mapping or removing the operation. This feedback is used by PROMPT to make further suggestions. Moreover, the user-interfaces for both tools are quite different, allowing us to investigate which type of interface better supports a user's mental model.

Four participants, $P1$, $P2$, $P3$, and $P4$, were involved in the study. $P1$ and $P2$ are graduate students in computer science, while $P3$ is an ocean sciences graduate student with a physics background, and $P4$ has a computer science background and works as a programmer. None of the users had used the tools or performed mappings prior to the study. $P1$ and $P2$ were placed in the first team, $T1$, while $P3$ and $P4$ were on the second team, $T2$. The sessions were video recorded, teams were told to "think aloud", and the generated mappings were saved for later analysis. A team approach was used to encourage discussion. Two university-related ontologies were selected for the experiment, one from the University of Maryland (UMD) and the other from Carnegie Mellon University (CMU). The ontologies cover a domain that should be familiar to all participants and are small enough (UMD has approximately 135 concepts, CMU has approximately 54 concepts) to be explored during the short duration of the experiment.

4.2 Analysis

There was a large difference in the users' satisfaction with the tools. $T1$ felt that by far, PROMPT was the more useful tool. They had a lot of difficulty making sense of the mapping lines drawn in COMA++ and $T1$ ignored the mapping suggestions after using the tool for seven minutes. Productivity greatly improved once they ignored the suggestions. $T1$ started to rely on remembering what they had mapped before in PROMPT and also their knowledge of the ontology's terms. The participants also highlighted context switching issues with COMA++. They found it difficult to tell what had been mapped and what was left to be verified or mapped. $P2$ even stated, "How do we know when we're done?" $T1$ stated during an interview that they felt two people were necessary to use COMA++ effectively because it forced them to remember so much information: where they were in the ontologies, what had been mapped, etc. For example, they mapped one term twice, first correctly, and then later incorrectly. The teams also tended to revisit mapped terms, having forgotten that they had already inspected them.

Conversely, $T2$ primarily felt that COMA++ was the more effective tool. $P3$ stated, "COMA++ was easy, was straight-forward, was obvious. The Protégé [PROMPT] tool was irritatingly complex." $P4$ agreed that PROMPT had a complex interface, but he did not feel that either tool was necessarily better. He stated that COMA++ was simpler, but difficult to use when there were a lot of candidate mapping lines. He did however feel more confident about the mappings he produced using COMA++. PROMPT gave more information for validating a mapping, but that also complicated the process. There could be several contributing factors for this difference in opinion. The order in which the tools were used may have influenced expectations.

From our analysis, we observed that all participants followed a similar decision making process when judging potential mappings. They relied on concept name similarity from either the suggested candidate mappings or the ontology trees as an indicator of a possible alignment. Next, they used both the internal and external structure of the concepts for validation. If the concepts had similar structure (i.e. context), they felt confident that the mapping was valid. $T2$ also highlighted that they relied on their domain knowledge of how a university functions to make decisions. These observations directly correspond to some of the decision making theories previously discussed. Exact matches allow the users to quickly filter the mapping suggestions, as in filter theory. Also, users rely on the internal structure of the mapping terms to compare shared and unshared attributes to infer intended meaning. Domain expertise is used (as in the heuristic-systematic persuasion model) to search for appropriate mappings and also contributes to confirmation bias when inspecting a mapping.

The ability to search and filter mappings and ontology data surfaced during the mapping session and interviews. PROMPT supports searching, but this did not work as the participants expected. $T1$ mentioned searching repeatedly, especially while using COMA++, which does not have any search facilities. Advanced searching and filtering (e.g., fuzzy searches) may also be needed, because ontology elements may use abbreviations, prefixes, suffixes, and different word orders. Searching is a way for users to explicitly explore a user-driven mapping or to reduce the ontology's complexity.

In PROMPT, both teams relied on the list of candidate mappings for navigation, while in COMA++, the teams relied on the tree structure of the ontologies. With COMA++, the only navigational device is the ontology trees. When participants mapped two concepts they were often able to quickly perform several additional mappings. We believe this is because once they found a mapping that they were sure about, they inferred other mappings of parent and child concepts from the ontology trees. However, in PROMPT, they primarily focused on the candidate list, and mostly ignored the ontology trees. Due to this difference in navigation strategy, we believe that COMA++ may better facilitate learning of the ontologies because the user must browse the trees to perform mappings.

Most of the performed mappings were perceived by the participants as simple or "easy" mappings. However, during the study sessions, both teams were forced to ignore some potential mappings when they could not determine if the mapping was correct or could not agree on a decision. This is an interesting result, because both tools do not support a mechanism for returning to a decision point. It is left to the user to remember to come back and inspect a mapping that they initially ignored.

Both teams emphasized the need to determine what has been mapped and what is left to map. $T1$ stated that PROMPT supported this better, as it places an "m" icon beside mapped concepts. However, they found it difficult to get a sense for how much they had accomplished and to understand how much was left to complete. Similarly, $T2$ felt that COMA++ needed to visualize the difference between unverified and verified mappings.

The teams both liked that PROMPT supplied a reason for suggesting a mapping, although sometimes this reason led to confusion and indecision (e.g. "Meeting" potentially mapping to "Thing", as a result of the "ing" suffix). $T2$ did not feel the confidence value provided by COMA++ (a number between zero and one) was particularly useful. How a tool communicates its candidate mappings relates to how much the user trusts the suggestions.

In summary, the main user concerns seem to stem from the usability of the tool and the cognitive support it offers for manual tasks, rather than the automated mapping generation mechanisms. The main concerns raised were:

- Where should my starting point be for mapping ontologies?
- How do I know when the mapping procedure is complete?
- How can I verify the quality of my mapping?
- How can I identify the most similar areas of the ontologies?
- How can I limit the scope of the mapping?
- How do I flag or indicate a questionable or subjective mapping?
- How can I make temporary decisions and reverse decisions about mappings?

In the following section, the cognitive support framework we propose addresses these user issues within the context of the mapping algorithm support.

5 Cognitive support framework

In [12], we proposed preliminary cognitive support requirements for ontology mapping tools. Since then, we used data from the observational study and further research into cognitive psychology to develop a theoretical framework describing mapping concepts relating to cognitive support. The framework is shown in Fig. 1 and discussed below.

The framework has four conceptual dimensions: *User Analysis and Decision Making*, *Interaction*, *Analysis and Generation*, and *Representation*, which are based in part on work from [4, pp.7] and [29]. Each dimension represents a concept in the human-guided ontology mapping process. Users internally perform analysis and decision making to understand and validate mappings. Externally, they interact with the tool to acquire information or create mappings. The tool internally performs analysis and generates mappings and externally presents these to the user. Distributed cognition between

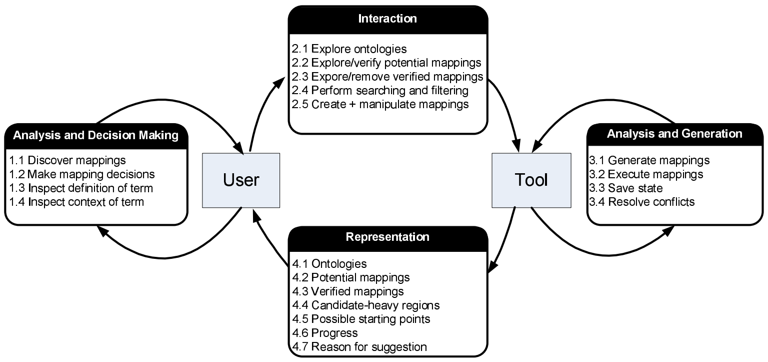

Interaction
2.1 Explore ontologies
2.2 Explore/verify potential mappings
2.3 Expore/remove verified mappings
2.4 Perform searching and filtering
2.5 Create + manipulate mappings
Analysis and Decision Making
1.1 Discover mappings
1.2 Make mapping decisions
1.3 Inspect definition of term
1.4 Inspect context of term
User
Tool
Analysis and Generation
3.1 Generate mappings
3.2 Execute mappings
3.3 Save state
3.4 Resolve conflicts
Representation
4.1 Ontologies
4.2 Potential mappings
4.3 Verified mappings
4.4 Candidate-heavy regions
4.5 Possible starting points
4.6 Progress
4.7 Reason for suggestion

Fig. 1: A theoretical framework for cognitive support in ontology mapping.

user and artifact (tool) makes the task manageable. The framework dimensions are described below and corresponding software tool requirements (**REQ**) are described for each framework principle (**FP**).

Analysis and Decision Making

(#1.1) Discover mappings:
FP: Users discover mappings based on their domain knowledge or by exploring the ontologies. This information is often internalized until the user is convinced of the mapping.
REQ: Support ontology exploration and manual creation of mappings. Provide tooling for the creation of temporary mappings that the user can address at a later time.
(#1.2) Make mapping decisions:
FP: Users internally make mapping decisions. The tool aids this by suggesting potential mappings that the user validates.
REQ: Provide a method for the user to accept/reject a suggested mapping.
(#1.3) Inspect definition of term:
FP: The definition of a term comes from the properties that describe the internal structure of the term. The internal structure helps explain the meaning of the term, which facilitates the user's understanding of the ontology.
REQ: Provide access to full definitions of ontology terms.
(#1.4) Inspect context of term:
FP: Context is how a term is used in an ontology. This is derived from the external structure (the `is_a` hierarchy) and the internal structure (definition of the term). Context of terms in a mapping help the user verify that the intended meaning of terms are the same.
REQ: Show the context of a term when a user is inspecting a suggested mapping.

Interaction Dimension

(#2.1) Explore ontologies:
FP: User-driven navigation of terms, properties, and relationships in the ontologies enforces understanding of the ontology and discovery of mappings.
REQ: Provide interactive access to source and target ontologies.
(#2.2) Explore/verify potential mappings:
FP: Exploring potential mappings aids the user in the verification process.
REQ: Support interactive navigation and allow the user to accept/reject potential mappings.
(#2.3) Explore/remove verified mappings:
FP: Navigation of the verified mappings allows the user to explore what they have completed and what is left to complete.
REQ: Support interactive navigation and removal of verified mappings.

(#2.4) Perform search and filter:
FP: Search and filter facilitates the reduction of information overload for mappings. It also facilitates planning as they allow the user to focus on smaller chunks of the mapping process.
REQ: Provide support for searching and filtering the ontologies and mappings (e.g. filters to display terms in the ontologies with/without mappings, or display only the mappings with exact name matches.)

(#2.5) Direct creation and manipulation of the mappings:
FP: Many mappings are missed by automated procedures, requiring the user to manually create them. Manipulation refers to adding metadata to a verified mapping, such as a reason for the mapping.
REQ: Support for adding details on verified mappings and manually create mappings.

Analysis and Generation

(#3.1) Generate mappings:
FP: Automatic generation of mappings helps users identify simple mappings.
REQ: Support the automatic discovery of some mappings.

(#3.2) Execute mappings:
FP: Executing mappings is the process of transforming instances from one ontology to another based on the available mappings. This can be treated as a *debugging* step in creating a complete mapping: the user can verify if the instances created in the target from the source instances are the ones that (s)he expected.
REQ: Allow the user to test mappings by automatically transforming instances from the source to the target ontology.

(#3.3) Save verification state:
FP: Automatically saving the mapping state and returning to that state with each session relieves the user's working memory from determining where they were, what they were doing, and what their next step is, after an interruption.
REQ: The verification process must support potential interruptions by automatically saving and returning users to a given state.

(#3.4) Conflict resolution and inconsistency detection:
FP: Conflict resolution helps users determine inconsistencies in the created mappings. They can arise from a variety of situations, such as when two concepts are mapped, but some structural elements that are critical for their definition have not been mapped yet.
REQ: Support identification and guidance for resolving conflicts.

Representation Dimension

(#4.1) Source and target ontologies:
FP: Representation of the ontologies facilitates understanding and discovery.
REQ: Provide a visual representation of the source and target ontology.

(#4.2) Potential mappings:
FP: Representation of a potential mapping aids the discovery and decision making process.
REQ: Provide a representation of a potential mapping describing why it was suggested, where the terms are in the ontologies, and their context.

(#4.3) Verified mappings:
FP: Representation of verified mappings frees a user's working memory from remembering what they have already verified.
REQ: Provide a representation of the verified mappings that describe why the mapping was accepted, where the terms are in the ontologies, and their context.

(#4.4) Identify "candidate-heavy" regions:

FP: Identification of candidate-heavy regions aids the planning procedure for performing mappings. It also facilitates understanding of results from the automated procedure.
REQ: Identify visually candidate-heavy regions based on the automated mapping procedure.

(#4.5) Identify possible starting points:
FP: A starting point represents an area of the ontologies or potential mappings where the user may wish to first concentrate their mapping effort.
REQ: Indicate possible start points for the user, e.g. flag terms that have exact name matches, as these are generally the most straight-forward mappings to perform.

(#4.6) Progress feedback:
FP: Progress feedback facilitates planning, as it provides details about where the user is in the overall mapping process. This is also an indicator about the current verification state.
REQ: Provide progress feedback on the overall mapping process.

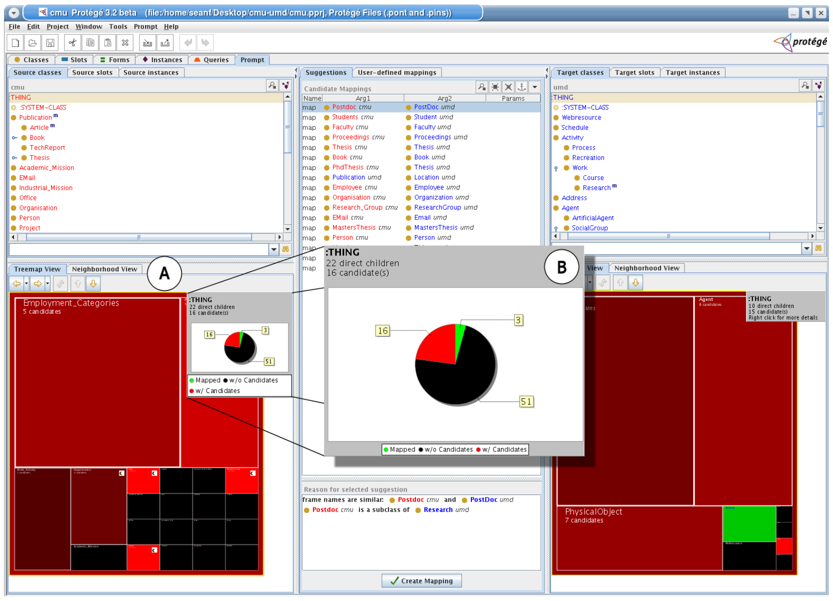
(#4.7) Reason for suggesting a mapping:
FP: Mappings auto-generated by the tool can support verification and understanding by "explaining" why the algorithm decided the two terms match. An explanation facility helps the user to decide on a mapping and also builds trust between the algorithm and the user.
REQ: Provide feedback explaining how the tool determined a potential mapping.

6 Using the framework to design a tool

In this section, we use the derived requirements to design a plugin for cognitive support in ontology mapping. Rather than building a tool from scratch, we decided to extend an existing mapping tool with a plugin for cognitive support. We recognized PROMPT as the best match for our cognitive support tool integration. This is because PROMPT already addresses some of the cognitive support requirements we defined and it is available as an open source tool. By working with the PROMPT developers, we created an extensive plugin architecture that allows researchers to easily plug-in their own algorithms, user interface components, and mapping file formats. Using this plugin architecture, researchers can extend many of PROMPT's user interface components. These extensions to PROMPT were first discussed at the "Ontology Matching Workshop 2006" [12].

We decompose the mapping process into steps: algorithm for comparison, presentation of mappings, fine-tuning and saving mappings, and execution of mappings. These represent plugin extension points in PROMPT. These extensions allow researchers to move their ideas from prototypes to fully implemented mapping tools, without recreating the entire user interface. These extensions to PROMPT provide the ontology engineering community with a consistent interface for mapping and give users access to a suite of tools and algorithms.

We developed a PROMPT plugin called COGZ (Cognitive Support and Visualization for Human-Guided Mapping Systems). COGZ was first introduced in [12] and contained only a neighborhood graph visualization. The latest version contains cognitive aids to address requirements derived from our framework. Because COGZ works as an extension to PROMPT, it can harness the features of PROMPT and enhance or support them with additional visual components. The plugin architecture also allows any algorithm plugin to indirectly benefit from the cognitive support provided by COGZ.



Name	Arg1	Arg2	Params
map	Postdoc *cmu*	PostDoc *umd*	
map	Students *cmu*	Student *umd*	
map	Faculty *cmu*	Faculty *umd*	
map	Proceedings *cmu*	Proceedings *umd*	
map	Thesis *cmu*	Thesis *umd*	
map	Book *cmu*	Book *umd*	
map	PhdThesis *cmu*	Thesis *umd*	
map	Publication *umd*	Location *umd*	
map	Employee *cmu*	Employee *umd*	
map	Organisation *cmu*	Organization *umd*	
map	Research_Group *cmu*	ResearchGroup *umd*	
map	EMail *cmu*	Email *umd*	
map	MastersThesis *cmu*	MastersThesis *umd*	
map	Person *cmu*	Person *umd*	

Reason for selected suggestion
frame names are similar: Postdoc *cmu* and PostDoc *umd*
Postdoc *cmu* is a subclass of Research *umd*

Fig. 2: CogZ TreeMap view (A) with enhanced pie chart view (B).

Figure 2 shows the Prompt+CogZ tool. TreeMaps [25] are used to provide an overview of the ontology and potential mappings (req. 2.1, 2.2, and 2.3). The TreeMap was chosen for several reasons. The overview needed to fit a small area of the user interface yet display a large amount of data. Since TreeMaps are space-filling, they take up the same amount of screen regardless of ontology size. Also, since ontologies can be very large, we needed a visualization that scales well; TreeMaps can visualize several thousand nodes [13]. Color intensity in the TreeMap helps identify candidate-heavy regions of the ontology and mapped regions (req. 4.4 and 4.3). The pie chart view provides details about the number of candidate mappings, mapped concepts, and concepts without an association within each branch of the ontology. This gives an overview about what has and has not been completed within a branch of the ontology (req. 4.6).

A visualization for comparing term neighborhoods is also available. The neighborhoods represent the "context" of the mapping terms, where the context is defined as the immediate structural relationships of an ontology term (req. 1.4). The generated context provides a visual structural comparison between two candidate terms. The CogZ plugin also provides mapping filters that can be used to reduce the number of mappings shown by Prompt and allow the user to focus on certain types of mappings (req. 2.4 and 4.5). The filters are based on the categories of potential mappings supported in Prompt (e.g., exact name matches and synonym matches). Users can also use hierarchical filters to display mappings within certain regions of the ontologies.

To support the user's working memory, we introduced temporary mappings by extending Prompt's candidate mapping list. The user can now flag a mapping as temporary, removing this candidate from the list. Temporary mappings can be viewed within the candidate list either by themselves or with the other candidates. When viewed within the existing candidate list, they are highlighted with a light-blue background. If the user

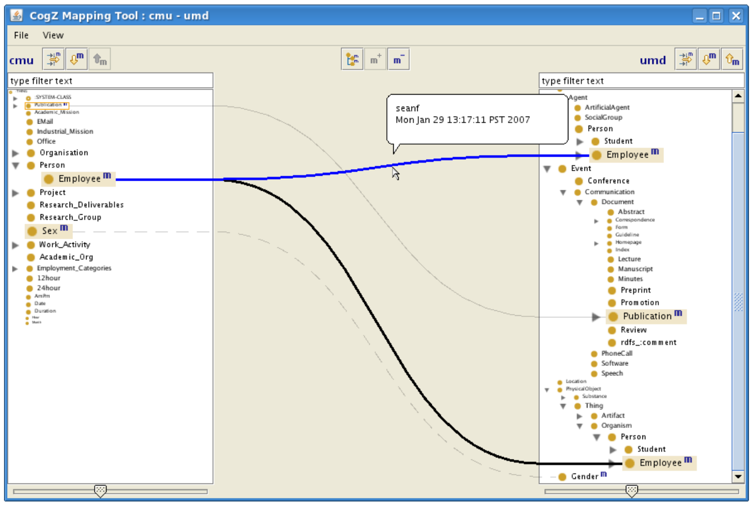


Fig. 3: COGZ view for the current mapping state.

performs a mapping with a concept that already has a temporary mapping, the user is reminded about this and asked if they wish to proceed. If they proceed, the corresponding temporary mappings are removed as possible candidate mappings (req. 1.1 and 1.2).

Figure 3 shows an example of COGZ's complete mapping interface (req. 2.3). Mappings are shown as edges drawn between the ontology trees. The view also displays a mapping annotation that can be used by users to explain why they chose to map two terms (req. 2.5). Temporary mappings are displayed as dashed lines between the source and target terms. The view supports semantic zooming or ”fisheye“ selection to highlight the current focus. The semantic zooming also effectively displays cases of multiple inheritance, as shown in the figure.

An interactive search is supported where the ontology trees are automatically filtered when a user types in a search query. The text of the query is highlighted in matched nodes. The trees can also be filtered to display only terms with or without mappings. This gives the user a quick overview of the specific mappings they've performed. These advanced filters combined with the semantic zooming help reduce the clutter that can occlude the display when there are a large number of mapping lines (req. 2.4).

While most of the requirements are supported by the PROMPT+COGZ tool suite, there are some limitations. PROMPT's support for executing mappings (req. 3.2), saving verification state (req. 3.3), manual creation of mappings (req. 2.5), and searching (req. 2.4) need to be further extended by COGZ.

7 Discussion and conclusion

The semantic web brings structure and formal semantics to web data. The vision is to create a globally linked database of information, where data can be shared between web pages and local data stores [22]. A prerequisite for information sharing is the mapping of independent data representations. This procedure is usually carried out offline and relies on the knowledge of domain experts. In this paper, we advocate for cognitive support in ontology mapping tools. Existing research points to a tendency to think of the underlying ontology mapping algorithm as mostly independent from the user. We strongly believe that by embracing a unified view of human and machine, cognitive aids introduced to the mapping process will enhance the quality of mappings.

We introduced a cognitive support framework for mapping tools based on existing literature, theories of cognitive support and decision making, our experience, and an observational case study. The framework describes the relationship between user and tool in the mapping process. From the framework, we see that the automated generation of mappings is a small part of the entire mapping procedure. Moreover, based on our observational study, the problems users experience go beyond the processing of the algorithms. Users have trouble remembering what they have looked at and executed, understanding output from the algorithm, remembering why they performed an operation, reversing their decisions, and gathering evidence to support their decisions. We believe addressing these problems is the key to improving the productivity of the users.

The requirements from this framework were used to develop COGZ, a user-interface plugin for the ontology management suite PROMPT. This tool introduces visualizations to support user cognition, filters to reduce mapping scope, mapping annotations, and the novel cognitive aid of a temporary mapping. To support COGZ as an extension to PROMPT, we enhanced PROMPT by developing a plugin architecture to support algorithm plugins, user-interface plugins, and mapping file extensions. This architecture allows us to harness PROMPT's existing framework and to add mapping algorithm tools to PROMPT. This enables researchers to harness PROMPT's architecture by plugging in their tools and ideas, quickly moving from prototype to full implementation. This approach also makes COGZ completely algorithm independent.

In the future, we plan to carry out a larger usability study to refine the cognitive support framework and enhance the features of COGZ. The PROMPT development team is also enhancing mapping file extension support and other PROMPT features including conflict resolution and mapping verification.

References

1. Performance, parsing and pragmatics. http://www.phon.ucl.ac.uk/home/marco/.
2. P. A. Bernstein and S. Melnik. Model management 2.0: Manipulating richer mappings. In *SIGNMOD'07*, June 2007.
3. L. Brainbridge. Ironies of automation. *Automatica*, 19:775–779, 1983.
4. F. Detienne. *Software Design - Cognitive Aspects*. Springer-Verlang.
5. H.-H. Do. *Schema Matching and Mapping-based Data Integration*. PhD thesis, Department of Computer Science, Universität Leipzig, 2006.
6. M. Ehrig and Y. Sure. Ontology mapping - an integrated approach. In *1st European Semantic Web Symposium*, Heraklion, Greece, 2004. Springer, LNCS.
7. M. Ehrig and Y. Sure. FOAM–framework for ontology alignment and mapping; results of the ontology alignment initiative. In *Proceedings of the Workshop on Integrating Ontologies*, volume 156, pages 72–76, October 2005.
8. H. J. Einhorn and R. M. Hogarth. Behavioral decision theory: Processes of judgement and choice. *Annual Review of Psychology*, 32:53–88, 1981.
9. J. Euzénat. Eon ontology alignment contest. http://oaei.inrialpes.fr/2004/Contest/.
10. J. Euzénat. *An API for ontology alignment (version 2.1)*, Feb. 2006.

11. J. Euzénat, D. Loup, M. Touzani, and P. Valtchev. Ontology alignment with ola. In *Proceedings of the 3rd EON Workshop, 3rd International Semantic Web Conference*, Nov. 2004.
12. S. M. Falconer, N. F. Noy, and M.-A. Storey. Towards understanding the needs of cognitive support for ontology mapping. In *Ontology Matching Workshop*, 2006.
13. J.-D. Fekete and C. Plaisant. Interactive information visualization of a million items. In *INFOVIS '02: Proceedings of the IEEE Symposium on Information Visualization (InfoVis'02)*, page 117, Washington, DC, USA, 2002. IEEE Computer Society.
14. A. C. Kerckhoff and K. E. Davis. Value consensus and need complementarity in mate selection. *American Sociological Review*, 27:295–303, 1962.
15. A. Kirlik. *Global Perspectives on the Ecology of Human-Machine Systems*, chapter 4, pages 68–120. Requirements for psychological models to support design: Toward ecological task analysis. Lawrence Erlbaum Assocatiates, 1995.
16. P. Lambrix and A. Edberg. Evaluation of ontology merging tools in bioinformatics. In *Proceedings Pacific Symposium on Biocomputing*, pages 589–600, Kauai, Hawaii, USA, 2003.
17. M. Lanzenberger and J. Sampson. Alviz - a tool for visual ontology alignment. In *IV '06: Proceedings of the conference on Information Visualization*, pages 430–440, Washington, DC, USA, 2006. IEEE Computer Society.
18. J. Löwgren. *Human-computer Interaction. What every system developer should know.* Studentlitteratur, Lund, 1993.
19. D. L. McGuinness, R. Fikes, J. Rice, and S. Wilder. The chimaera ontology environment. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 1123–1124, 2000.
20. N. F. Noy and M. A. Musen. Evaluating ontology-mapping tools: Requirements and experience. In *Proceedings of OntoWeb-SIG3 Workshop at the 13th International Conference on Knowledge Engineering and Knowledge Management*, pages 1–14, 2002.
21. N. F. Noy and M. A. Musen. The PROMPT suite: Interactive tools for ontology merging and mapping. *International Journal of Human-Computer Studies*, 59(6):983–1024, 2003.
22. S. B. Palmer. The semantic web: An introduction. http://infomesh.net/2001/swintro/, 2001.
23. G. G. Robertson, M. P. Czerwinski, and J. E. Churchill. Visualization of mappings between schemas. In *CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 431–439, New York, NY, USA, 2005. ACM Press.
24. M. Sherif, D. Taub, and C. Hovland. Assimilation and contrast effects of anchoring stimuli on judgements. *Journal of Experimental Psychology*, 55:150–155, 1958.
25. B. Shneiderman. Tree visualization with tree-maps: 2-d space-filling approach. *ACM Trans. Graph.*, 11(1):92–99, 1992.
26. H. A. Simon, G. B. Dantzig, R. Hogarth, C. R. Piott, H. Raiffa, T. C. Schelling, and et al. Decision making and problem solving. *National Academy of Sciences*, 1986.
27. M. Snyder and N. Cantor. Testing hypotheses about other people:the use of historical knowledge. *Journal of Experimental Social Psychology*, 15:330–342, 1979.
28. Syque. Theories about decision-making. http://changingminds.org/explanations/theories/a_decision.htm.
29. B. Victor. Magic ink: Information software and the graphical interface. http://worrydream.com/MagicInk/.
30. A. Walenstein. *Cognitive support in software engineering tools: A distributed cognition framework*. PhD thesis, Simon Fraser University, Vancouver, BC, 2002.