

Navigating Virtual Information Sources with Know-ME

Xufei Qian¹ Bertram Ludäscher¹ Maryann E. Martone²
Amarnath Gupta¹

San Diego Supercomputer Center, University of California San Diego, USA
{xqian,ludaesch,gupta}@sdsc.edu

Department of Neuroscience, University of California San Diego, USA
mmartone@ucsd.edu

In many application domains such as biological sciences, information integration faces a challenge usually not observed in simpler applications. Here, the to-be-integrated information sources come from very different sub-specialties (e.g., anatomy and behavioral neuroscience) and have widely diverse schema, often with little or no overlap in attributes. Yet, they can be *conceptually* integrated because they refer to different aspects of the same physical objects or phenomena. We have proposed **model-based mediation** (MBM) as an information integration paradigm where information sources with hard-to-correlate schemas may be integrated using *auxiliary expert knowledge* to hold together widely different data schemas [GLM00, LGM00, LGM01]. The expert knowledge is captured in a graph structure called the **Knowledge Map**. In MBM, we extend the global-as-view architecture by lifting exported source data to conceptual models (CMs) that represent more source specific knowledge than a logical schema. The mediator’s IVDs are defined in terms of source CMs and make use of a semantically richer model involving class hierarchies, complex object structure, and rule-defined semantic integrity constraints. Additionally, sources specify *object contexts*, i.e., formulas that relate a source’s conceptual schema with the global domain knowledge maintained at the mediator. In this paper, we introduce a tool called Knowledge Map Explorer (Know-ME) for a user to explore both the domain knowledge, and all data sources that have been integrated using it.

Knowledge Map. A Knowledge map consists of four components:

A *domain map* (DM) is an edge-labeled directed graph whose nodes C are called concepts, and whose edge labels R are called relationships. For example, a DM may contain the labeled edge $map\ kinase \xrightarrow{isa} enzyme$ specifying that *map kinase* is a subconcept of *enzyme*. Note that *isa* is a special relationship and defines the concept hierarchy. Other relationships may define a “has a” hierarchy, spatial relationships such as “inside”, and domain-specific relationships such as “projects to”. Concepts C and relationships R can be constant symbols or ground atomic formulas.

A *process map* (PM) is an edge-labeled directed graph whose nodes S are called states, and whose edge labels P are called processes or events. Like relationships R of a domain map, labels P can be ground atomic formulas. For example, a PM may contain the labeled edge $s_i \xrightarrow{activates(map_kinase, protein_kinase_A)} s_j$ specifying a transition from state s_i to s_j under a parameterized process called *activates(map_kinase, protein_kinase_A)*. Such a transition can change the truth value of fluent predicates. For example, a temporal logic axiom like “*IF [S] not active(Y), activates(X,Y) THEN [S+1] active(Y)*” asserts that when the above transition is considered, the fluent *active(protein_kinase_A)* will be true in s_j .

Links between Conceptual and Procedural Knowledge. The knowledge represented in DMs and

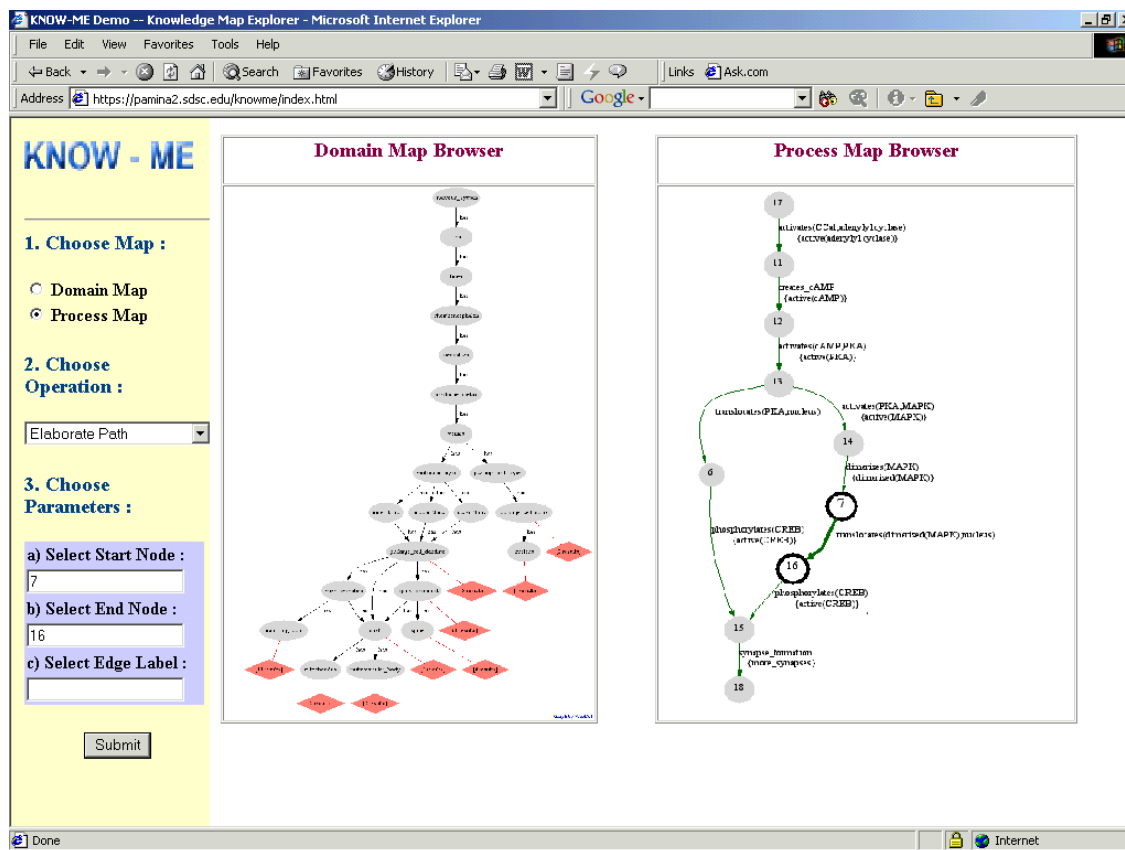


Figure 1: The prototype Know-ME interface

in PMs can be linked by using processes as concepts, i.e., allowing that process labels P and concepts C come from a common namespace $N = C \cap P$. This means that the PM contains the procedural knowledge about concepts in N , while a DM describes how a process in N relates to other processes. For example, L_LTP (late long-term potentiation) is a process $state_i \xrightarrow{E_LTP} state_j \xrightarrow{L_LTP} state_k$ comprising two intermediate steps early and late phase LTP. From the DM, we may know that $L_LTP \xrightarrow{occurs_in} hippocampus$ i.e., this process occurs in the hippocampus.

Links from Knowledge Maps to Data. Nodes in DMs and edges in PMs are linked to actual data via *context edges*. For example, a source Src_1 that has immunolabeling images that may serve as “evidence” of the process $activates(\dots)$ can declare this fact using a *context edge* of the form $Src_1 \xrightarrow{has_evidence(immunolabeling_image)} activates(\dots)$ linking source data to a process. The declarative semantics of knowledge maps is given via a logic program.

The Demonstration System. The Knowledge Map database stores the graph structure of domain and process maps. It includes the UMLS ontology from the National Library of Medicine and the Gene Ontology from the Gene Ontology Consortium. In the demo system, the wrappers connect to web-accessible Neuroscience information sources. The logic engine is FLORA, an F-Logic preprocessor on top of XSB Prolog. The rules specifying the semantics of Knowledge Maps are implemented in F-Logic. The query engine uses a graph processor and a logic engine as required. The KNOW-ME tool is a query formulation front-end that drives the query engine by sending it relativized and possibly parameterized queries.

The KNOW-ME Tool. The KNOW-ME tool presents to the user a two-window interface

showing a DM and a PM, respectively. In either window, the user needs to create an initial graph from which she starts exploration. In the default case, the system presents the user with a preset initial DM and PM – the user starts expanding the graph by querying on their neighboring nodes and edges. In a more involved case, the user selects from a number of concepts, relationships and processes, and asks the system to compute a connected subgraph that contains the chosen nodes and edges. For this request, the graph processor *constructs* the initial graph, and the user may select a node, edge or subgraph to launch the next “relativized query”, applied only to the active subgraph. The KNOW-ME tool internally maintains a list of all visible, active and response subgraphs. A visible subgraph is a fragment of a Knowledge Map that appears on the screen at the time of the next operation; an active subgraph is one that has been highlighted by the user as the “argument” of the next operation; and a response subgraph is returned as the answer to a query. These are represented with different shapes (for nodes) and colors (for edges) on the interface. The user may manually deactivate an active subgraph or set a preference to automatically deselect the active subgraph after a query has been evaluated and make the response subgraph active. After selecting the active subgraph the user can right-click to open a query menu and perform either a basic operation or a predefined parameterized query. An active subgraph of the PM can be seen (in bold) in Figure 1, while the diamonds in the DM shows evidence edges leading to data.

Operations. The basic operations for programming graph queries for the KNOW-ME tool are expressed as generalized path expressions. Two important operations are *elaboration* and *abstraction*. The first substitutes the edge e with a path $\langle e_1, e_2, \dots, e_k \rangle$ that has been logically defined as a more elaborate description of the process denoted by e . The second is the reverse of above operation. The user can formulate more complex queries by selecting the “create new query” option in the query menu. Once created a query can be saved for future use. The demo will exhibit a number of pre-defined complex queries.

References

- [GLM00] A. Gupta, B. Ludäscher, and M. E. Martone. Knowledge-Based Integration of Neuroscience Data Sources. In *12th Intl. Conference on Scientific and Statistical Database Management (SSDBM)*, pp. 39–52, Berlin, Germany, July 2000. IEEE Computer Society.
- [LGM00] B. Ludäscher, A. Gupta, and M. E. Martone. Model-Based Information Integration in a Neuroscience Mediator System. In *26th Conf. on Very Large Data Bases (VLDB)*, pp. 639–642, Cairo, Egypt, 2000. Morgan Kaufmann. demonstration session.
- [LGM01] B. Ludäscher, A. Gupta, and M. E. Martone. Model-Based Mediation with Domain Maps. In *17th Intl. Conf. on Data Engineering (ICDE)*, Heidelberg, Germany, 2001. IEEE Computer Society.