MOBBED (Mobile Brain-Body-Environment Decision-making) Part I: Database Design

By

Jeremy Cockfield, Kyung Min Su, and Kay Robbins

Department of Computer Science

University of Texas at San Antonio

April 2013

UTSA CS Technical Report TR-2013-005

Updated from UTSA CS Technical Report TR-2012-006

A companion technical report entitled: MOBBED (Mobile Brain-Body-Environment Decision-making) Part II: User Guide is available as TR-2013-006.

This report was revised 7/4/2013 to reflect a minor reorganization in the database tables.

This report was revised 8/12/2013 to reflect some minor renaming of columns in the database tables.

Contents

1. Introduction

The US Army CAN-CTA project focuses on understanding the brain and body in action using a data-driven approach with a goal of developing assistive and monitoring technologies for soldiers in real world environments. Project scientists collect EEG, eye-tracking, motion capture, muscle activity, heart rate, force-plate responses, as well as sensory and galvanic skin responses from subjects performing a variety of tasks in realistic environments. Video and audio recordings, both of subjects and of the environment are also available. Some datasets, acquired in controlled simulation environments, have detailed simulation-specific information that describes and controls the context. Some subjects will undergo additional imaging modalities (such as diffusion tensor imaging or simultaneous fMRI and EEG recording) in fixed environments to obtain detailed structural brain information.

Brain-body imaging in realistic environments has only recently become feasible with the advent of portable dry-electrode technology. Various partners within the CAN-CTA are beginning to acquire datasets that combine various modalities. However, the equipment, data format, processing, and storage techniques vary considerably across groups. The CAN-CTA data corpus will be, for the most part, annotated streaming data characterized by diverse formats, high data rates, and complex interrelationships. Analysis and discovery in such a data corpus requires a data handling structure that is robust, extensible, and capable of extracting complex combinations of features for analysis and visualization.

Two fundamental assumptions have been widely articulated among CAN-CTA members. The first assumption is that effective data-driven approaches require a data corpus. Without training and test data that covers the range of possibilities, machine learning and other computational approaches fail. The second assumption is that the corpus of data acquired under the initial CAN-CTA can provide a lasting legacy with far-reaching potential beyond the individual experiments and particular research questions that initiated these experiments.

2. What is MOBBED and why does it need a data infrastructure?

The ultimate MOBBED (Mobile Brain-Body-Environment Decision-making) system is the human being --- compact, mobile, adaptive. Environmental observation occurs through the senses, while processing, command, and control functions occur in the brain. Human systems have limits in terms of bandwidth for sensory acquisition, processing capabilities, and decision-making acuity. Furthermore, humans are not machines --- they are subject to disease, injury, fatigue, and stress --- all of which can degrade their performance. In addition, humans have evolved under vastly different conditions than are typically present in a technological, information-rich environment. Assistive and monitoring technologies are needed to improve and stabilize human performance in these situations.

Fig. 1 shows a schematic of a typical real-time MOBBED system for assistance and monitoring. Examples of environmental acquisition include GPS positioning, information from cameras and microphones mounted inside or outside vehicles or robots, body mounted cameras. Feedback and instructions may come in the form of audio or visual cues or possibly cues from other sensory devices. The human instrumentation may include EEG, eye-tracking, voice tracks, motion capture, and physiological measures such as heart rate.

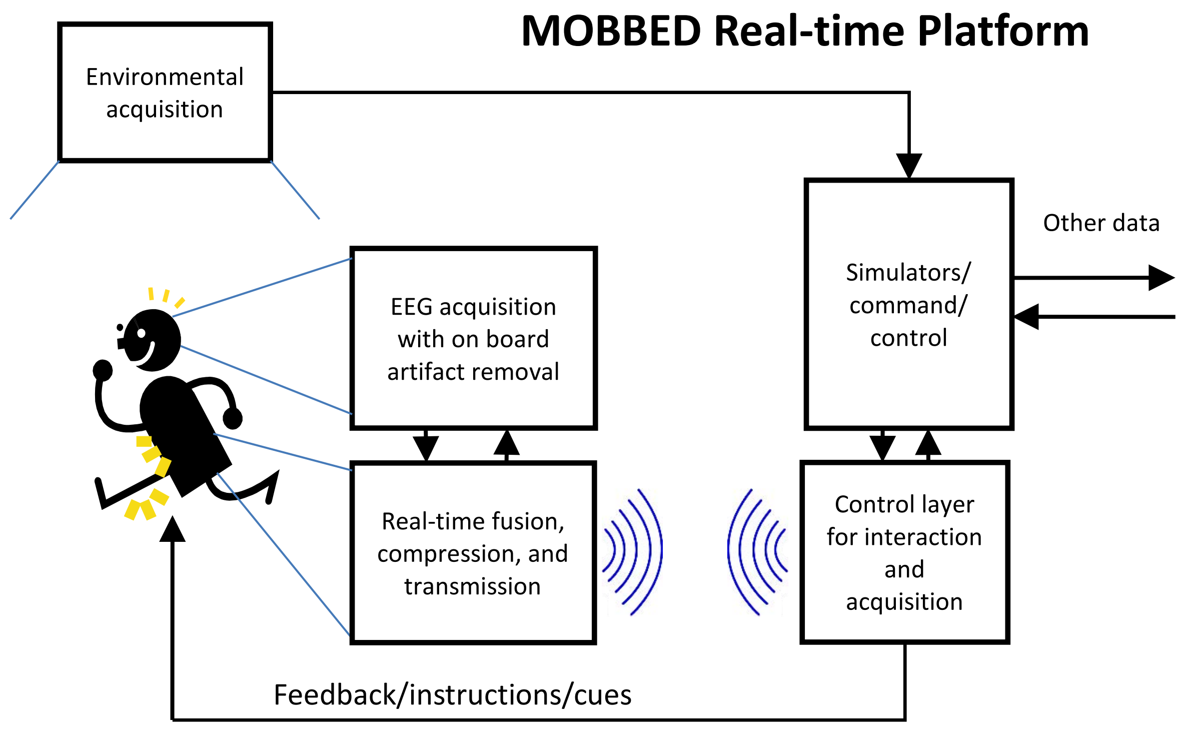
MOBBED Real-time Platform
Environmental acquisition
EEG acquisition with on board artifact removal
Real-time fusion, compression, and transmission
Simulators/ command/ control
Other data
Control layer for interaction and acquisition
Feedback/instructions/cues

Fig. 1: MOBBED real time platform.

.

Command and control may come from human commanders, automated assistants, or from simulators such as the DCS Crewstation Simulator. These entities require data from other sources to provide a context for decision-making. In training and testing, some of this data can come from preplanned scenarios. Automated assistants may need extensive data for making decisions. Human commanders need their own information assistants to integrate and summarize complex situations.

Fig. 2 shows a proposed CAN-CTA analysis platform dataflow. Data from laboratory instrumentation is stored in archive files that come in a variety of formats including proprietary instrumentation formats, the Data River XML[1] format, and laboratory-specific formats.

In the current incarnation of the typical data flow for CAN-CTA members, the analysis software consists of EEGLAB, BCILAB, and other software written in MATLAB. The access layer for EEG data is `EEG`, an instrumentation-independent MATLAB data structure used in most EEGLAB analysis. Original archival data is stored in various formats and in various states of processing by individual experimenters. There is currently no standard for integrating EEG data and other types of data for analysis. The current workflow does not contain either archival or computational databases.

To address the problem of fusing different types of data for analysis, members of SCCN at UCSD have proposed development of standardized MATLAB structures similar to `EEG` for other modalities [2]. The MOBBED analysis suite will need standardized structures for eye tracking, motion capture, and physiological indicators to permit the development of general purpose processing and machine learning algorithms.

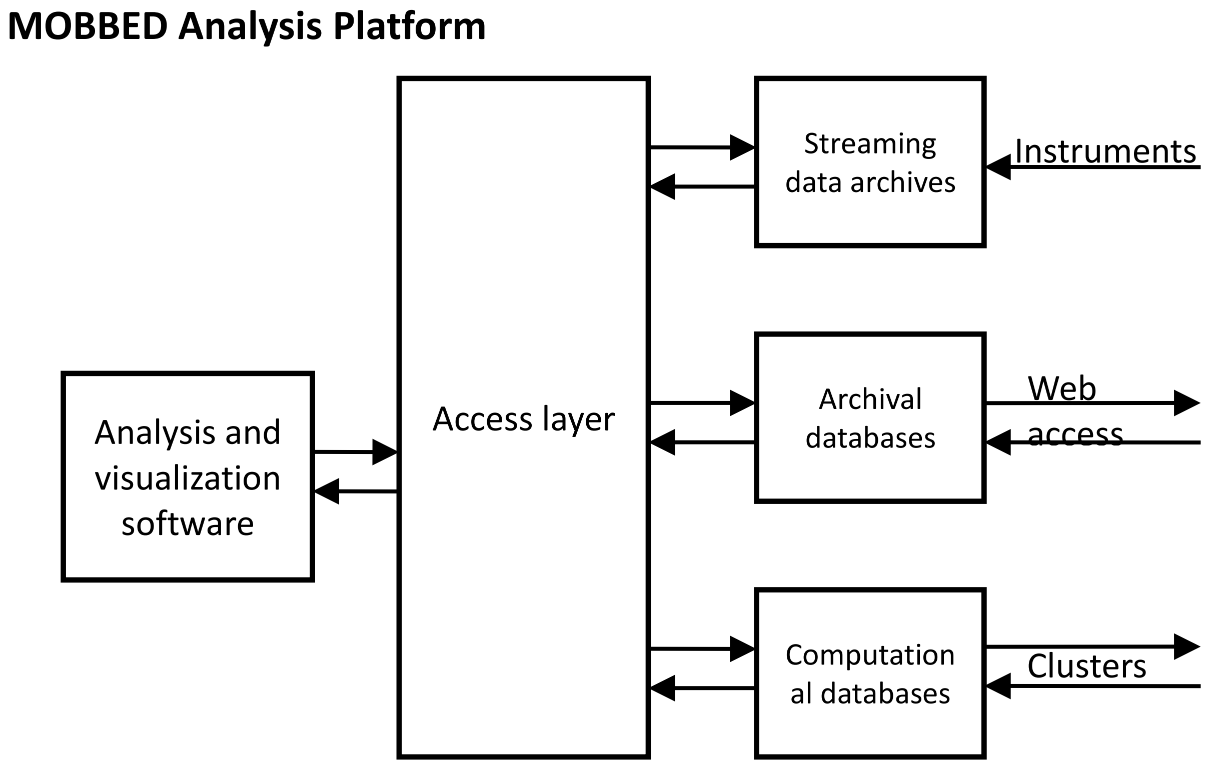

Fig. 2: Data flow for the MOBBED assistive technology.

The current approach of storing data in MATLAB structures provides an efficient and highly effective access layer that allows researchers to develop algorithms in MATLAB that are independent of the acquisition instrumentation. However, the approach also has some limitations in terms of scalability and access. Researchers must keep the entire dataset in memory during computation, which will become prohibitive as the datasets become larger and data mining requires combinations of multiple datasets. Furthermore, stored MATLAB structures are not searchable, making it hard to extract and combine data across multiple datasets without reading all of the datasets into memory and scanning them. Finally, because no central data repository exists, the `EEG` structures must be redundant, storing a copy of the original dataset along with the transformation, each time researchers perform an ICA or apply a different version of artifact removal. EEGLAB has standardized the MATLAB `EEG` structure for ICA transformations, so analysts must include their own non-standardized fields if they wish to perform a different type of transformation.

To address the memory issues, Nima Bigdely Shamlo, Christian Kothe, and Alejandro Ojeda of SCCN are developing an access layer infrastructure called Mobi based on memory-mapped files [3]. This development is an important step in improving the scalability of the MOBBED data flow. However, by its nature, memory-mapping is a local strategy. The data structures map to the local file system, and each machine must keep its own copy. One might envision an implementation that uses proxies to distribute pieces across the network on other machines, but it is not clear what the performance of such a strategy would be.

Memory mapped files will play a central role in the local caching structure needed to support future large-scale MOBBED computation. However, this strategy does not address some of the other limitations mentioned above. In an effort complementary to the UCSD work, UTSA is implementing an experimental database back end (MOBBED) to address the following issues:

- Accessibility of large sets of MOBBED data across research groups for searching, extraction, and mining.
- Separation of the original data from various transformations such as channel and epoch rejection or ICA while maintaining an association so that researchers know where the data came from and what transformations it has undergone (provenance).
- Storage of the data in a platform-independent way so that tool developers and researchers are not locked into a particular platform such as MATLAB, but still have the option of taking advantage of these platforms if they choose.
- Provision of the basic infrastructure needed to apply large scale distributed processing techniques using tools such as Apache Hadoop [4] to perform data mining.
- Data abstraction to facilitate retrieval and classification.

As shown in Fig. 2, the envisioned workflow has two different types of databases. Archival databases provide permanent storage and a high-level of security. Computational databases provide more support for data mining and large-scale computation. The HeadIT project at UCSD [5] and the IEEG.org project [6, 7] are examples of archival databases. Data coming from instrumentation would initially be stored in instrumentation-specific formats or in the Data River XML format. UCSD's MobiLab [8] is an access layer tool that provides drivers to store streaming data into structured local files. The access layer software would also provide drivers for reading and writing this data to the archival database(s) in an offline batch processing operation.

The MOBBED data infrastructure includes a computational database that provides an alternative to flat files for facilitating large-scale computation. We have implemented a prototype MOBBED database called MOBBED using PostgreSQL [9]. The analysis software is MATLAB and the access layer has a Java implementation. We have also implemented a graphical user interface for the database in MATLAB. The remainder of this white paper gives an overview of the prototype design and implementation. Section 3 gives an overview of the design strategy. Section 4 gives a more detailed description of the MOBBED design, and Section 5 describes the basic access-layer user interface. Section 6 provides some performance measurements for the prototype implementation, and Section 7 discusses a future design and implementation path. Appendix A shows some screenshots of MOBBED GUIs.

3. Implementation overview

MOBBED has a relational schema currently implemented as a PostgreSQL database, a well-established open source database that is widely used for information retrieval research. PostgreSQL has a number of data handling and query extensions not available in mySQL. The open source pgAdmin tools [10] provide a nice suite of database management and browsing capabilities outside of the MOBBED framework. MOBBED follows several design principles:

Principle 1: Disk storage is infinite

The design consequence of this principle is that large tables and database redundancy can pay for efficiency in extraction and in the capability of extension. The rapid decline in disk costs and the availability of federated servers affirms that this principle is practical for long-term design horizons. Every row and hence every entity in MOBBED is uniquely identified by a 128-bit UUID.

Principle 2: Extensibility is primary and trumps efficiency

One design consequence of this principle is that MOBBED tables are narrow and general rather wide and specific. As a result, more queries may be required to retrieve the required records than would ordinarily be required in a focused design. Additional assembly of information from diverse records may be required in such a design. A mitigating factor in this design choice is that analysts will access MOBBED databases through a caching middleware layer of assembled information specific to individual applications. Often MOBBED performs the assembly outside of the SQL query mechanism by copying records to a stream.

Principle 3: Structures should be controlled but extensible

Users work with a variety of structures to analyze their data and specification of these structures is necessary for storage and retrieval as well as searching. However, because of the complex and ever-evolving nature of MOBBED, designers cannot anticipate all future structure types. MOBBED treats structure layout as data and promotes definition of hierarchies of structure types.

Principle 4: Primary data acquired from experiments is distinct from user transformations

The `EEG` MATLAB structure in EEGLAB inter-mixes user transformations such as ICA, trial rejection, and event transformations with primary data. This approach makes the representation very compact and promotes efficiency of analysis for a single dataset. However, the approach has some drawbacks for large-scale analysis tasks. A researcher wishing to run an analysis with different epoching strategies or with different rejected channels must create two complete copies of the dataset. The current infrastructure does not provide an organized method of keeping track of the provenance of such operations. MOBBED stores primary datasets separately from user transformations, but links them to allow users to perform limited provenance tracing of the transformation sequence.

Initially, MOBBED will only store original data. However, as the CANCTA comes to an agreement on a standardized and automated preprocessing procedure (dereferencing, sampling, high pass filtering, artifact removal, saccade and blink separation, ICA transformation, etc.), the MOBBED will represent and store "standardized" versions of the data. MOBBED provides users with the capability of creating additional tables of transformed data and additional data mining capabilities. The next section outlines the basic structure of our initial MOBBED PostgreSQL design.

4. MOBBED entities

The MOBBED design includes eleven (11) different primary entities (e.g., attribute, comment, contact, data_def, dataset, device, element, event, subject, tag, and transform). Each entity has a universally unique 128-bit identifier (UUID) that serves as the unique key for the corresponding database table. The unique identification of primary entities, allows users to merge databases without conflict and to back track entity types based on their UUIDs. Section 6 provides more implementation details.

4.1 Datasets and their structure

A dataset represents a group of related data. A dataset may represent data collected in a single modality such as EEG or eye-tracking that is identified as a unit and represents a single experiment. A dataset may also represent a collection of other datasets. Datasets are the fundamental organizing entities in MOBBED. A dataset may have elements that represent streams of time-stamped measurements. They also may have events, attributes, metadata such as tags, and other types of data. Fig. 3 shows a schematic of the organization around a dataset.

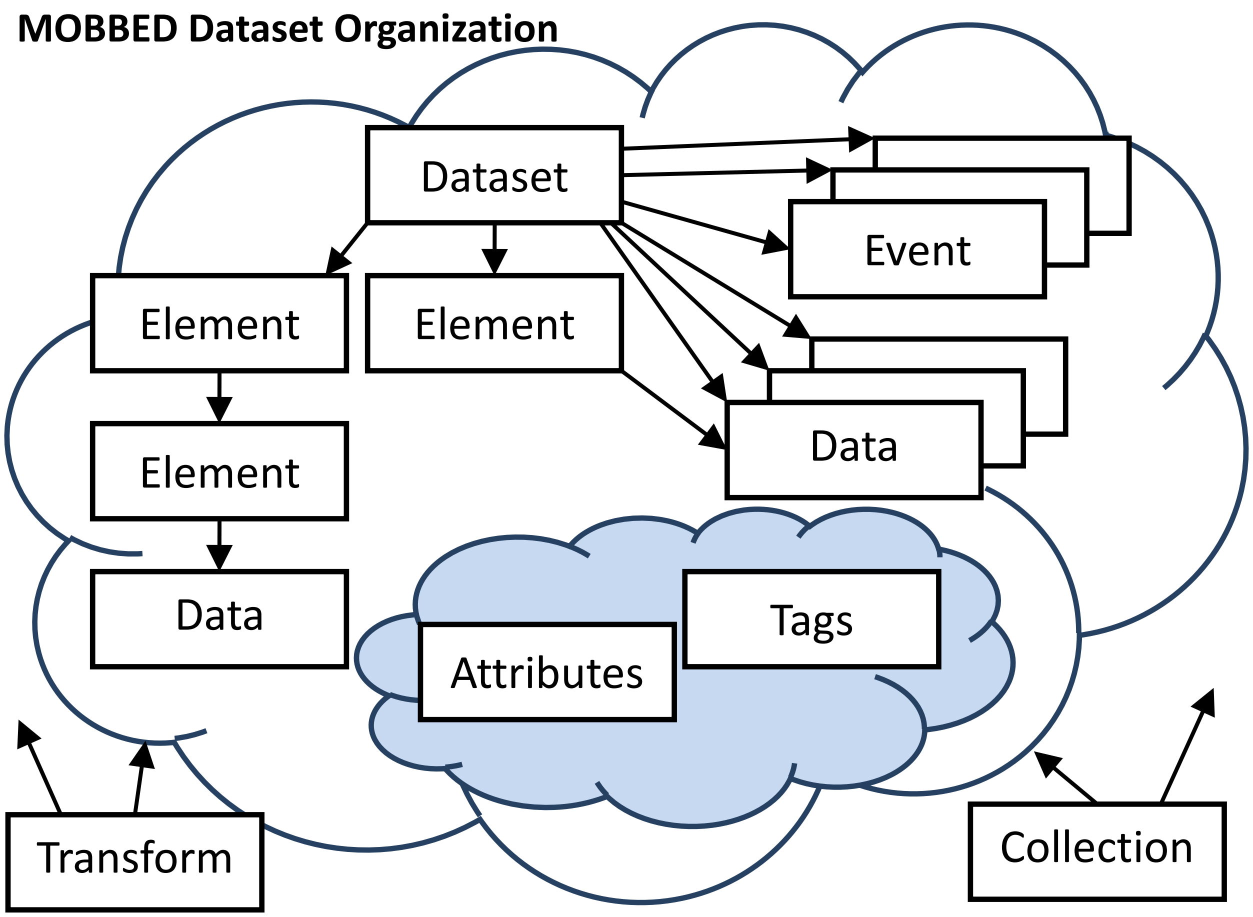

Fig. 3: Schematic organization of a dataset in MOBBED.

Elements represent sensors or other spatially distributed measurement devices. These elements may be organized in a hierarchy. For example, an element may represent an entire EEG cap or just a single channel in that cap. Consider for example, the `EEG` structure that represents EEG data in EEGLAB. The `EEG.chanlocs` field is a structure array holding channel locations and corresponds naturally to the dataset elements. The current mapping of the `EEG` structure to the database creates a single element corresponding to the EEG cap and identifies the EEG data with this cap. If the `EEG.chanlocs` is not empty, MOBBED creates additional elements corresponding to each channel. The value of a channel property for the i^{th} channel, such as `EEG.chanlocs(i).labels`, is mapped to an attribute. The attribute has a structure ID that encapsulates information about structure layout for retrieval purposes.

MOBBED treats events as primary entities rather than attributes because of their central role in searching and retrieval. Events are entities with a type, a starting time, an ending time, and a unique entity owner. MOBBED stores additional event properties as attributes. Events provide descriptive overlays for other MOBBED entities. Researches can code traditional EEG events such as the appearance of a target as event entities. However, the event infrastructure also supports the coding of more general features, e.g., a strong correlation of a given channel with another or the appearance of energy of a certain level in a specified frequency band. Researchers must first extract these implicit events and then code them in MOBBED by associating them with appropriate event types. Once coded, researchers can then retrieve a variety of signals associated with the occurrence of such events.

Elements or datasets or other entities may have data associated with them. The data may be stored in one of five ways: time-stamped numeric array, time-stamped XML blob, numeric-array, XML blob, or external file. Information and properties that do not fit into the event/data format are stored as attributes. Users may associate tags with any of the other entities and use the tags for searching and data mining.

The remainder of this section describes the representation of a dataset and its associated events and attributes.

4.1.1 The DATASETS Table

The DATASETS table identifies the datasets in the database. We assume that each dataset consists of a single modality (such as EEG data). MOBBED specifies the structures for extraction in type tables described in the next section. The DATASET_UUID identifies this dataset in operations and serves as the unique key for the DATASETS table. Primary datasets use their own DATASET_UUID as the DATASET_PARENT_UUID, while derived datasets use the DATASET_UUID of the dataset from which they were derived. The description may contain information about the transformation. Child datasets inherit the events and attributes of their parents.

DATASETS: [key: DATASET_UUID**]**

Table column	Meaning
DATASET_UUID	UUID identifying the dataset.
DATASET_SESSION_UUID	UUID identifying the experimental session.
DATASET_NAMESPACE	String identifying the namespace (e.g., lab URL) for this dataset.
DATASET_NAME	String name identifying this dataset.
DATASET_VERSION	Integer version number of this dataset.
DATASET_CONTACT_UUID	UUID of the contact person for this dataset.
DATASET_CREATION_DATE	Date that dataset was entered into the database.
DATASET_DESCRIPTION	Description of the dataset.
DATASET_PARENT_UUID	UUID of parent dataset (own UUID for primary datasets).
DATASET_MODALITY_UUID	UUID of the modality of dataset.
DATASET_OID	Object identifier of external file containing copy of the dataset.

A copy of the dataset is stored externally by the database as a large binary object and assigned an object identifier. The user may also store an arbitrary number of additional data items using the data definition mechanism. Some systems that interact with MOBBED require that datasets have a unique name. MOBBED provides unique naming through the combination of the namespace, the name, and the version. The DATASET_NAMESPACE is typically the URL or other string uniquely identifying the laboratory or the investigator. After they have specified a namespace, researchers only have to be concerned about unique naming within their own workspace. Depending on the type of insertion, MOBBED interface routines use a flag to indicate whether a duplicate insertion (without version number) should fail and return with an error or should complete correctly after incrementing the version number. Often datasets from multiple modalities will be acquired during a single experiment or session. The DATASET_SESSION_UUID identifies these datasets.

4.1.2 The ELEMENTS table

Element entities identify time stamped data streams. An element may correspond to a single entity or may contain sub elements. For example, researchers may choose to store the data corresponding to one EEG run using a cap with 36 channels as a single element with each time snapshot of the data stored in a single row of the numeric data table. However, because the individual channels of the cap have properties such as name and position, researchers could also create additional child elements for each channel and associate attributes with these elements. MOBBED can index element data for individual channels through the parent element data using the ELEMENT_POSITION value.

ELEMENTS [key: ELEMENT_UUID**]**

Table column	Meaning
ELEMENT_UUID	UUID identifying this element.
ELEMENT_LABEL	Name of the element (channel in EEG).
ELEMENT_DATASET_UUID	UUID of the dataset containing the element.
ELEMENT_PARENT_UUID	Parent element if this element should be grouped.
ELEMENT_POSITION	Position of this element in parent's group (-1 if all).
ELEMENT_DESCRIPTION	Description of what this element represents.

4.1.3 The EVENT_TYPES and EVENTS table

Event entities have a name, a starting time, and an ending time. MOBBED associates each event with a particular target entity. For example, EEG events from the EEGLAB `EEG.urevent` structure have entries in this table associated with a target dataset. This design decision allows fast extraction of events for an entity at the expense of space for duplication of similar events.

In order to facilitate tagging and association of events across multiple datasets, MOBBED keeps a separate EVENT_TYPES table and uses the EVENT_TYPE_UUID in the EVENTS table rather than the string representing the event directly. All event types are strings. This representation emphasizes the discrete nature of events and also promotes identification of comparable events across multiple datasets and studies. An event type such as 101 is not suitable for data mining when viewed in the context of a single experiment.

EVENT_TYPES [Key: EVENT_TYPE_UUID**]**

Table column	Meaning
EVENT_TYPE_UUID	UUID of the event type.
EVENT_TYPE	String event type (e.g., 'ButtonPress').
EVENT_TYPE_DESCRIPTION	Description of this event type.

EVENTS [Key: EVENT_UUID**]**

Table column	Meaning
EVENT_UUID	UUID of the event.
EVENT_DATASET_UUID	UUID of the dataset associated with this event
EVENT_TYPE_UUID	UUID of the event type.
EVENT_START_TIME	Double time in seconds of start of event.
EVENT_END_TIME	Double time in seconds of end of event.
EVENT_POSITION	64-bit index of the event.
EVENT_CERTAINTY	64-bit double value in [0, 1] indicating the certainty of an event.

4.1.4 The MODALITIES table

MOBBED organizes information into datasets representing a single modality such as EEG. MOBBED associates each modality with one or more standardized access layer structures. MOBBED supports three modalities: EEG, GENERIC, AND SIMPLE. The MOBBED infrastructure supports future extensions to different modalities and different standardized structures for given modalities. MOBBED tries not to duplicate data storage, but may duplicate metadata to allow multiple forms of access layer extraction.

MODALITIES: [key: MODALITY_ UUID**]**

Table column	Meaning
MODALITY_ UUID	UUID identifying the modality.
MODALITY_ NAME	Name of the modality (e.g., EEG).
MODALITY_PLATFORM	Access layer platform for extraction (e.g., EEGLAB).
MODALITY_DESCRIPTION	Description of the modality and access layer.

4.1.5 The TRANSFORMS table

Each MOBBED dataset corresponds to data collected from a single experiment and single modality. MOBBED can organize multiple datasets from the same experiment into a collection. During analysis these raw datasets may be transformed to produce new datasets. MOBBED uses transform entities to store and reuse transformed data. MOBBED transforms usually represent strings that the access layer can evaluate. For a MATLAB access layer, MOBBED would apply the MATLAB `eval` function to evaluate such a transform. In BCILAB [11], for example, this string is a fully parenthesized expression containing the full transformation history starting with the original data. The TRANSFORM_MD5_HASH is the MD5 hash of the transform string. The evaluation of the transformation string should result in a single output dataset whose UUID is the TRANSFORM _UUID. A user can fetch the dataset resulting from this transformation by querying with a specific MD5 hash value rather than recalculating.

TRANSFORMS: [key: TRANSFORM _UUID**]**

Table column	Meaning
TRANSFORM _UUID	UUID identifying the dataset resulting from the transform.
TRANSFORM_STRING	String specifying the transform (often executable command string).
TRANSFORM_MD5_HASH	MD5 hash of the transform string.
TRANSFORM_DESCRIPTION	Description of the transform.

4.2 Data

MOBBED allows a dataset to store its data in multiple ways. The simplest way is as a large binary object that is retrievable from the server through an OID (object ID). This method of storage is efficient for retrieval and storage. MOBBED also can store data directly in the database in one of four formats: a time-stamped array of double values, a time-stamped self-defining XML blob, an array of double values, or an XML blob. The representation choice involves a trade-off between search, extraction, and space efficiency. The researcher may choose not to store the data in the database at all but only use the flat file representation.

The DATADEFS table identifies a piece of data (an array, a blob, a stream, or a file). The description may include information about data provenance. Because a piece of data may be associated with many different entities and an entity may be associated with many different pieces of data, a separate DATAMAPS table provides the many-to-many associations. The

DATADEFS [key: DATADEF_UUID**]**

Table column	Meaning
DATADEF_UUID	UUID identifying the data.
DATADEF_FORMAT	NUMERIC_VALUE, NUMERIC_STREAM, XML_VALUE, XML_STREAM, or EXTERNAL.
DATADEF_SAMPLING_RATE	Sampling rate in Hz (-1 if not fixed or not time series).
DATADEF_OID	OID of the data file stored in the system table if external.
DATADEF_DESCRIPTION	Description of what this element represents.

DATAMAPS [key: DATAMAP_DEF_UUID, DATAMAP_ENTITY_UUID**]**

Table column	Meaning
DATAMAP_DATADEF_UUID	UUID of the data definition.
DATAMAP_ENTITY_UUID	UUID of the entity associated with data by this map entry.
DATAMAP_ENTITY_CLASS	Name of the table in which the entity is defined in.
DATAMAP_PATH	String path specifying structure for data on retrieval.

The DATADEFS table does not contain the actual data, but specifies how the data is stored using the DATADEF_FORMAT value. Numeric vectors are stored in the NUMERIC_VALUES or the XML_VALUES table depending on the format, while time sampled data or data streams are stored in the NUMERIC_STREAMS or XML_STREAMS table.

NUMERIC_VALUES [key: NUMERIC_VALUE_DATADEF _UUID**]**

Table column	Meaning
NUMERIC_VALUE_DATADEF_UUID	UUID of data definition associated with this value.
NUMERIC_VALUE	Vector of doubles containing the data.

NUMERIC_STREAMS [key: NUMERIC_STREAM_DATADEF_UUID, NUMERIC_STREAM_RECORD_POSITION**]**

Table column	Meaning
NUMERIC_STREAM_DATADEF_UUID	UUID of data definition associated with this value.
NUMERIC_STREAM_RECORD_POSITION	64-bit long indicating position in stream (starting from 1).
NUMERIC_STREAM_RECORD_TIME	Double value with time in seconds from start.
NUMERIC_STREAM	Vector of doubles containing the data for this time.

XML_VALUES [key: XML_VALUE_DATADEF_UUID**]**

Table column	Meaning
XML_VALUE_DATADEF_UUID	UUID of data definition associated with this value.
XML _VALUE	XML blob containing the data.

XML_STREAMS [key: XML_STREAM_DEF_UUID, XML_STREAM_RECORD_POSITION**]**

Table column	Meaning
XML_STREAM_DATADEF_UUID	UUID of data definition associated with this value.
XML_STREAM_RECORD_POSITION	64-bit long indicating position in stream (starting from 1).
XML_STREAM_RECORD_TIME	Double value giving time in seconds of this record from start.
XML_STREAM	XML blob containing the data for this time.

4.3 Attributes and structure

Attributes, which do not have timestamps, may encapsulate entity metadata or other characteristics as well as time independent data. The semi-structured nature of MOBBED data makes mapping of attribute data into a relational database more challenging, since MOBBED must recover the structures on retrieval. For example, EEG events (as encapsulated in the `EEG.urevent` and `EEG.event` structures) always have a type and a start time but may also have other properties. Researchers often use these additional properties to characterize the event or the experimental conditions under which the event was observed. MOBBED stores each extra property as an attribute of the event and associates a structure with the attribute.

The ATTRIBUTES table associates string values with entities. An attribute entity has a target ATTRIBUTE_ENTITY_UUID indicating the specific entity associated with this property.

ATTRIBUTES [Key: ATTRIBUTE_UUID**]**

Table column	Meaning
ATTRIBUTE_UUID	UUID of the attribute.
ATTRIBUTE_ENTITY_UUID	UUID of the entity that has this attribute.
ATTRIBUTE_ENTITY_CLASS	Name of table in which entity is defined.
ATTRIBUTE_ORGANIZATION_UUID	UUID of the organizational unit containing the entity.
ATTRIBUTE_PATH	String path specifying structure for attribute on retrieval.
ATTRIBUTE_NUMERIC_VALUE	Double value or null if attribute value is non-numeric.
ATTRIBUTE_VALUE	Value of the attribute (a string that may be an XML blob).

MOBBED always stores the attribute values as strings. However, some attributes are actually numeric. In order to support numeric range queries, the ATTRIBUTES table also includes an optional numeric value.

4.4 Collections

A single experimental run may acquire EEG, eye tracking, and motion capture as well as environmental information. Each of the measurement modalities (e.g., EEG, eye tracking, and motion capture) would appear as an individual dataset, but the three datasets form a group representing the experimental run. MOBBED organizes this group as a collection. Collections may contain any number of datasets and other entities**.** The COLLECTIONS table defines the associations between a collection and the entities they contain. Users should take care not to insert cycles into the map.

COLLECTIONS [Key: COLLECTION_UUID, COLLECTION_ENTITY_UUID**]**

Table column	Meaning
COLLECTION_UUID	UUID of the dataset representing the collection
COLLECTION_ENTITY_UUID	UUID of an entity in the collection
COLLECTION_ENTITY_CLASS	Name of table in which entity is defined

4.5 Tags

A tag is a word or flag that users can assign to any entity to facilitate searching and association. Effective searching requires users to reuse tags where possible, so rather than allowing arbitrary tag names, MOBBED requires users to explicitly create tags prior to using them. The TAGS table lists the available tags.

The TAGS table encapsulates the many-to-many relationships between tags and entities.

TAGS [Key: TAG_NAME, TAG_ENTITY _ UUID**]**

Table column	Meaning
TAG_ NAME	Name of the tag.
TAG_ENTITY_UUID	UUID of entity in this mapping.
TAG_ENTITY_CLASS	Name of table in which entity is defined

4.6 Miscellaneous entities

4.6.1 Comments

Comments are informational strings that users can associate with any entity. Comments serve to document the data and may contain references to more complex documents such as images, papers, code, and bibliographic references. Users may search comments.

Comment entities have a comment string for a specific entity along with the time and the UUID of the person who wrote it. This design allows users to add multiple comments for any entity.

COMMENTS [Key: COMMENT_UUID**]**

Table column	Meaning
COMMENT_UUID	UUID of the comment.
COMMENT_ENTITY_UUID	UUID of the entity associated with this comment.
COMMENT_ENTITY_CLASS	Name of table in which entity is defined.
COMMENT_CONTACT_UUID	UUID of the person entering the comments entity
COMMENT_TIME	Local time timestamp.
COMMENT_VALUE	String value of the comments entry.

4.6.2 Contacts

A contact entity represents the contact information for individuals associated with the data including the owner of the data or the creator of database entries such as comments. The CONTACTS table promotes central updating of contact information without modifying the rest of the database. The table keeps the basic required contact information. Other information such as institutional affiliations, web sites, and alternate phone numbers are stored as attributes.

CONTACTS [Key: CONTACT_UUID**]**

Table column	Meaning
CONTACT_UUID	UUID of the person represented by this contact.
CONTACT_FIRST_NAME	First name of this contact.
CONTACT_LAST_NAME	Last name of this contact.
CONTACT_MIDDLE_INITIAL	Middle initial.
CONTACT_ADDRESS_LINE_1	First line of postal address.
CONTACT_ADDRESS_LINE_2	Second line of postal address.
CONTACT_CITY	City.
CONTACT_STATE	State.
CONTACT_COUNTRY	Country.
CONTACT_POSTAL_CODE	Postal code.
CONTACT_TELEPHONE	Contact phone number.
CONTACT_EMAIL	Primary email address.

4.6.3 Subjects

The SUBJECTS table holds subject entities, which represent data sources such as instrumented humans, simulators, or robots. MOBBED represents additional information about each subject through its attributes, comments, and tags.

SUBJECTS [Key: SUBJECT_UUID**]**

Table column	Meaning
SUBJECT_UUID	UUID of the subject.
SUBJECT_DESCRIPTION	Description of the subject.

4.6.4 Devices

The DEVICES table holds entities that represent measuring devices such as a particular EEG cap. Researchers can identify the particular piece of equipment used to acquire a dataset and document its history and properties by assigning appropriate attributes and types.

DEVICES [Key: DEVICE_UUID**]**

Table column	Meaning
DEVICE_UUID	UUID of the piece of equipment or other device.
DEVICE_CONTACT_UUID	UUID of the contact person for this device.
DEVICE_DESCRIPTION	Description of the equipment and its history.

5. Basic use cases

This section presents several use cases that describe basic access layer functionality for MOBBED databases. Features supporting basic exploration and extraction are candidates for implementation in early releases. Functionality that is more sophisticated is possible as data mining approaches evolve. MOBBED provides a MATLAB access layer, which is described another technical report. To support another analysis platform, such as R, an equivalent programmatic interface would be required. These use cases do not support transformations of data or the creation, representation and extraction of feature sets for data mining.

The simplest use case is for tools that allow users to understand the structure and content of the database. PostgreSQL has an excellent tree view browser in the pgAdmin tool suite [10] for simple exploration. This tool displays the table structure and allows users to perform simple SQL queries. PostgreSQL also has a number of open source web interface tools such as phpPgAdmin [12] for remote exploration of the data base contents. Exploration is a useful prelude for users who want to insert their own data or extract data for analysis.

The toolbox needs to support creation and deletion of databases from within MATLAB to facilitate creation of computational databases. The toolbox should allow multiple simultaneous connections to the same database as well as to different databases. The toolbox should support access to database tables, both for storage and retrieval, without the user needing to use SQL. The toolbox should also provide simple mechanisms for common operations such as storage, retrieval, and tagging of datasets.

The toolbox should also provide facilities for searching of complex event scenarios and possibly extracting of data slices around events.

6. Status

An initial version of MOBBED is available at http://visual.cs.utsa.edu/mobbed with source available at http://vislab.github.com/mobbed. A companion technical report entitled: MOBBED (Mobile Brain-Body-Environment Decision-making) Part II: User Guide is available as TR-2013-006.

Acknowledgments

We wish to acknowledge Arif Hoissan for his contributions to the initial implementation of this toolbox. We wish to acknowledge useful discussions with Scott Makeig, Nima Bigdely Shamlo, Christian Kothe, Alejandro Ojeda, Kaleb McDowell, Scott Kerick, and Andre Vankov. This research was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-10-2-0022. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

References

[1] A. Vankov, DataRiver XML Technical Specification Draft.

[2] S. Makeig, SCCN, personal communication.

[3] N. Bigdely-Shamlo, C. Kothe, and A. Ojeda, SCCN, personal communication.

[4] Hadoop description: http://en.wikipedia.org/wiki/Apache_Hadoop.

[5] HeadIT project: http://sccn.ucsd.edu/wiki/HeadIT.

[6] The IEEG portal: http://braintrust.seas.upenn.edu/.

[7] Z. Ives, T. Green, G. Karvounarakis, N. Taylor, V. Tannen, P. Talukdar, M. Jacob, and F. Pereira. 2008. The ORCHESTRA Collaborative Data Sharing System. *SIGMOD Rec.* 37, 3 (September 2008), 26-32. DOI=10.1145/1462571.1462577 http://doi.acm.org/10.1145/1462571.1462577.

[8] MoBI lab homepage: http://sccn.ucsd.edu/wiki/MoBI_Lab.

[9] PostgreSQL homepage: http://www.postgresql.org/.

[10] pgAdmin PostgreSQL administration and management tool homepage: http://www.pgadmin.org/development/.

[11] A. Delorme, T. Mullen, C. Kothe, Z. Akalin, N. Bigdely-Shamlo, A. Vankov and S. Makeig (2011). EEGLAB, SIFT, NFT, BCILAB, and ERICA: New Tools for Advanced EEG Processing, Computational Intelligence and Neuroscience, Article ID 130714.

[12] phpPgAdmin homepage: http://phppgadmin.sourceforge.net/doku.php?id=start.