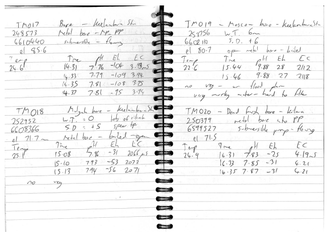
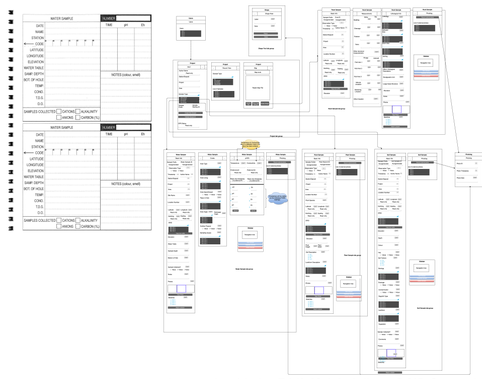
1 Data Recording

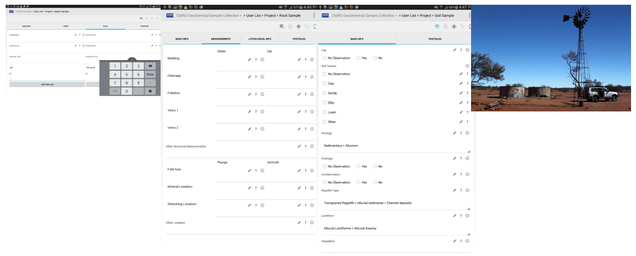


Geosampling data was initially collected via notebook.

CSIRO Researchers then moved to pre-printed field notebooks for more accurate data entry.



Imported into FAIMS Mobile. CSIRO workflow modeled in the generalised field recording app. Once approved, this workflow



After extensive testing, an app was deployed. These are three screens, designated by the wireframe and scripted by the files before.

Data was collected on multiple tablets in the field

FAIMS is designed to work completely offline, allowing asynchronous work on multiple tablets with eventual sync. The server, h

FAIMS Mobile is a FOSS Software platform (comprising an android client and ruby server on ubuntu) funded by the Australian Research Council designed to provide a means of collecting rich, geospatial, and multi-media field data on multiple tablets with no network connectivity in the middle of nowhere. While originally intended to support archaeologists, FAIMS Mobile provided a sufficiently general field recording framework to allow for geochemical and biological sampling by multiple teams of CSIRO researchers.

Code for this module is available at: https://github.com/FAIMS/CSIRO-Geochemistry-Sampling. The module can be explored by downloading FAIMS from google play and downloading the *CSIRO Geochemical Sample Collection - Sydney Maps* module from the demo server. A getting started guide is available at: http://faims.edu.au

2 Details of FAIMS

Internal Infrastructure

The fundamental innovation of FAIMS Mobile is the domain-key normal form append only datastore. Each record is identified by two naturally-unique identifiers: the user and the time of creation. Beyond that, every action is timestamped to allow for full histories and audit trails. Because every record and every action is timestamped and unique to a user, we can combine different versions of the database (i.e. those created on multiple devices over a week's trek through the Australian Outback) without any risk of clobbering edits or data. The most recent activity is 'true' and instances where multiple users edited the same data at the same time are flagged for human review.

DKNF Design

!(dknf)(dknf.png) (Image by Geoff Matheson).

Records are defined in 3 logical tables. `Rows' are defined in the ArchEntity table, which also holds cruical GIS data. `Columns' are defined by the attribute and Ideal Entity tables. The Attribute table defines what attributes are possible, their names, and their list/export formats. The Ideal Entity table defines which attributes belong to which entity. By defining these tables in DML (data *manipulation* language) rather than DDL (Data *Definition* Language), the structure of the database remains consistent. This consistent structure allows for significant query reuse and allows us to dynamically script the fields of a workflow *after* all the fundamental data interactions of the app have been rewritten.

Each attribute has four sub-attributes reflecting the needs of field data recording and can be multivalued if multiple rows *share* a timestamp. An attribute can comprise a set of: * a constrained vocabulary (how we implement dropdowns, checkboxes, and radio buttons); * a unconstrained measurement; * an annotation (to represent a way of scribbling in the margins without contaminating the data); and * a certainty (to reflect scribbling question marks or to otherwise rate physical uncertainty of the data reliability).

By combining these in a single `measurement'. Highly nuanced but *machine readable* data can be recorded in such a way as to fit the needs of the recording workflow.

Append only design

!(history)(eventLog.png)

This domain-key structure is also necessary to support the append only design. As each `event' (insertion, updates, `deletion') occupies its own row, we use GROUP BY and HAVING max(timestamp) to emit the latest versions of each attribute. Event uniqueness is guaranteed by UUID (user creation + time of creation due to the length limits of integers as primary keys in Sqlite), acting userid, and time of event. Therefore, by virtue of the need of having `eventually consistent' data stores, we also have a complete action log for every record: it shows when each attribute was edited and by whom. This allows granular control and review of records, as individual attributes can be `rolled back' to a more authoritative/correct state by users on the server.

The append-only design also protects against data-loss, as `deletions'" are merely a flag on the record which hides it from normal view. Thus, this database is designed to preserve user actions at all costs, allowing differences in datastores to be sent to the server and thereby distributed to all devices. This also has the virtue, so long as devices sync relatively often, of creating a complete backup of the datastore on every device, further armouring the database against mischance.

OSS Heritage

!(chart)(test.png)

The FAIMS Mobile app stands on the shoulders of giants. The only way an app this complex would be possible would be via the contributions of many open source projects – which is one of the primary reasons why our own work is released under the GPLv3. Crucial technologies include:

* JavaRosa which allows us to parse XML into native android widgets, allowing us to avoid the overheads of html webviews for every single dynamic component. * NativeCSS which allows us to include some stylings for elements, defined at module load, rather than at compile; * BeanShell which allows us to dynamically include a java-like scripting language, to allow for runtime logic changes. * Antlr3 which allows

for highly sophisticated parsing of strings, allowing for highly customisable record identifiers; * Spatialite which allows GIS operations inside the database. * Sqlite which is a supremely stable single-user database, which is perfect for offline operation * Ruby, Apache, Linux which allows us to write a sophisticated server running on a completely open source stack.

Choices of the fundamental libraries were made in 2012, when the project was commissioned as part of NeCTAR (National eResearch Collaboration Tools and Resources project) eResearch tools initiative and the libraries remain solid (if slower than some newer iterations) for the next few years to come.

3 CSIRO Workflow

4 Module Details

5 Faims internal architecture

6 OSS Heritage graph