ResourceMapper

Goal: collect structured data through freeform SMS and make that data available in multiple formats, at the terminal and over the internet.

Quick Intro

If a user was trying to report that a hospital had been damaged by water, they might send this SMS to a central hub: "The hospital has taken water damage". NLP techniques could be used to extract meaning from the message, but doing so is computationally expensive and only occasionally successful.

ResourceMapper takes a different approach and exposes an easy-to use SMS syntax that allows users to send in responses to a pre-defined set of fields. In the damaged hospital example, there would be a field called "Damage" where the choices are "Water Damage", "Fire Damage", "Earthquake Damage", and "No Damage". To submit a response to this field, the user would send an SMS containing a textable abbreviation of the "damage" field, along with their response, e.g. "dmg fire". This is the core of ResourceMapper: providing a user-friendly way for users to submit structured data in plaintext SMS.

Original Use Case

Originally, this system was designed for information gathering in disaster response scenarios where SMS is often the first communication technology to come back online. Specifically, it was designed to gather hospital information after the 2010 Haiti earthquake.

In a large disaster, many hospitals will quickly reach their maximum capacity for care, and many patients will not be adequately treated because the hospital does not have the proper facilities to handle their need. ResourceMapper is designed to collect information about the availability, status, and capabilities of each hospital, and forward that information to map-based app on Google App Engine. Once the data is inside the app it can be viewed by logistics coordinators or hospital staff to determine patient transfer opportunities and coordinate assistance.

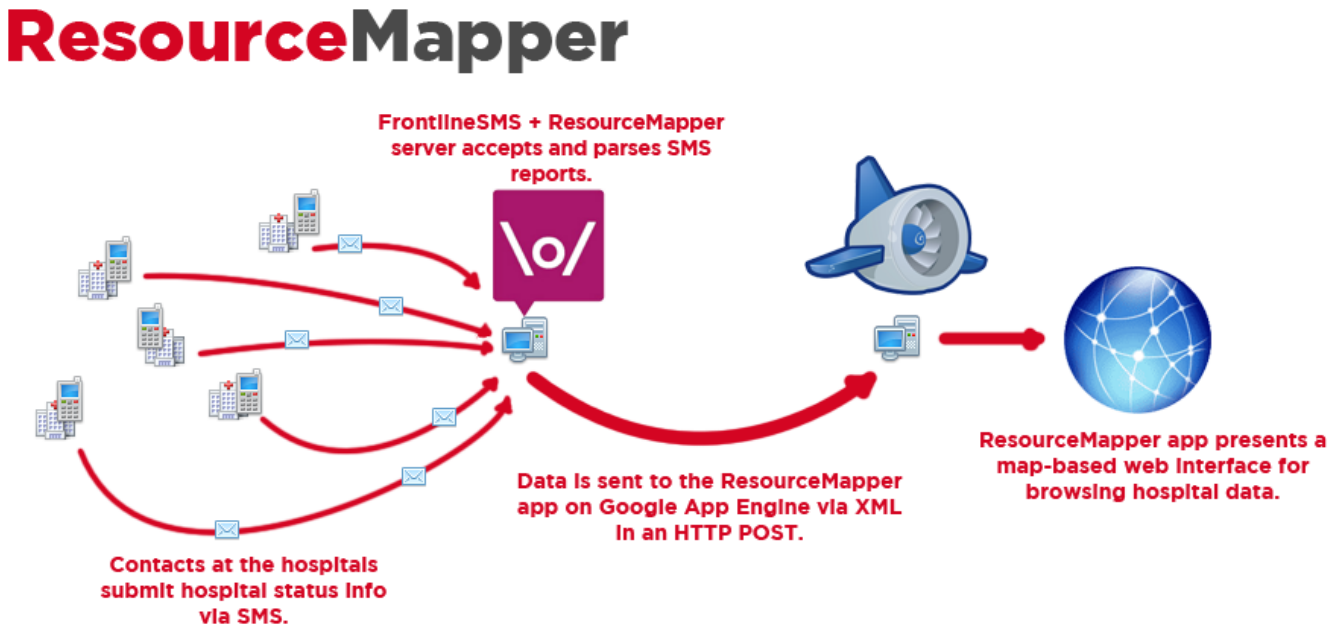
Expanded Use Case

There is need for structured data collection over freeform SMS beyond the disaster relief arena. ResourceMapper has several advantages over mobile-application-based data collection: there is nothing to install, the cheapest phones can use it, and the data fields being collected can be changed at the terminal.

Because of the usefulness of ResourceMapper in other areas, it should be developed in a modular way that allows it to satisfy the original use case, but also allows others to use and build on it.

Architecture

The architecture of the original system is explained in the diagram below.



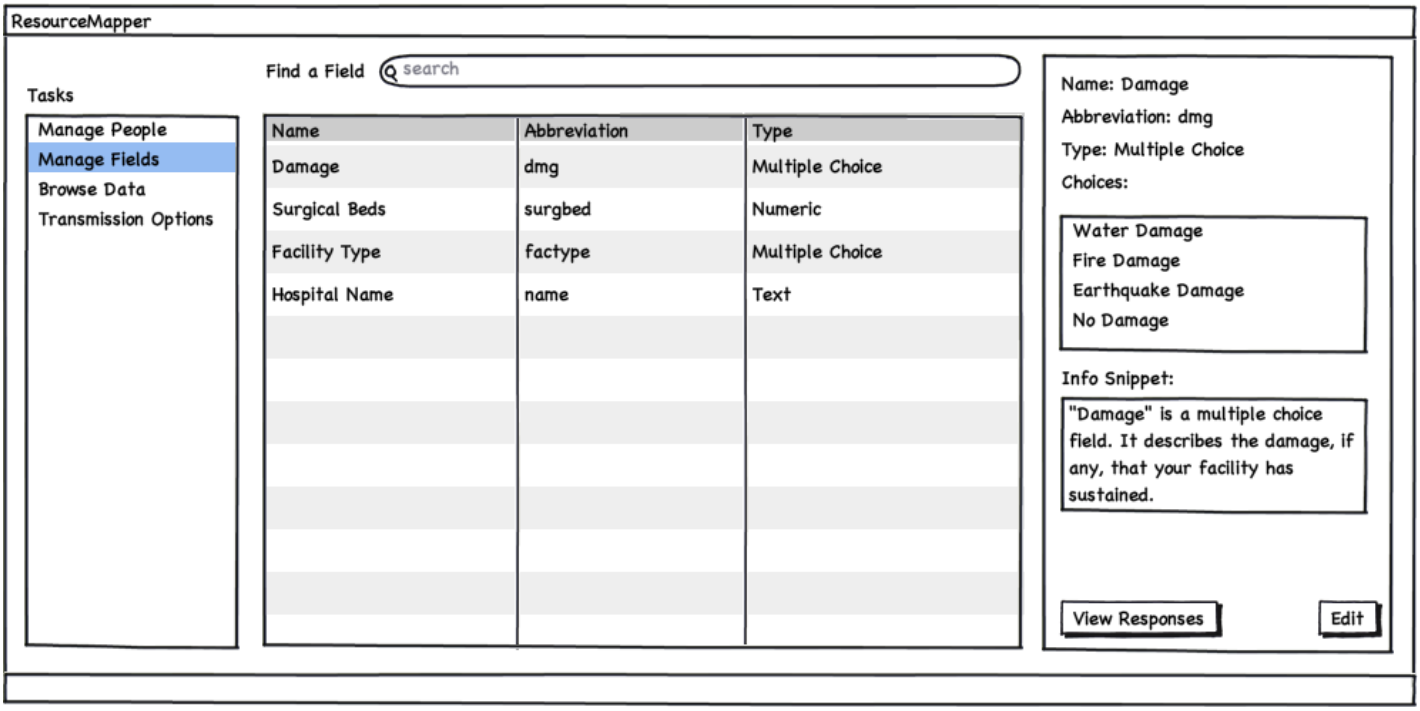
For modularity's sake, the system should be split up into two halves – data collection and data transmission. Data collection will always remain the same, but not all users will want their data POSTed to Google App Engine – some will want to just view it at the FrontlineSMS terminal, and some will want other options.

Part 1: Data Collection

The fields that can be submitted will be defined at the ResourceMapper server. The different field types are as follows:

Plain Text – anything goes
Numeric Text – numbers only
Boolean – yes or no
Checklist - collection of boolean values, can choose more than one answer
Multiple Choice - plain text, can only have one answer, answers are restricted to predefined choices

Here is a mockup of the UI for managing fields:



Once the fields are defined at the terminal, users can submit responses to them. Each datatype has a different workflow and different allowable responses.

Plain Text

The user begins the SMS with the field name and everything after the field name (space delimited) is considered to be the response.

Example: `name St. Vincent's Hospital`

However, if a user wants to include multiple plain text fields in one SMS, they must prefix the next field with pound sign (#).

Example: `name St. Vincent's Hospital #phone +1806.470.2435`

Numeric Text

It's the same as plain text, except the user will receive an error if they enter something besides numbers.

Examples

```
bedav 100
bedav 100 #name St. Vincent's Hospital #phone +18064702435
```

Error Example
User: `bedav lots`
ResourceMapper: `"Bed Availability" is a numeric field, and 'lots' is not a valid response.`

Boolean

Similar to plain text and numeric fields, except only accepts boolean responses. The acceptable responses should be definable at the terminal in an option panel, and could include y, yes, yeah, yea, ye, 1, n, no, and 0.

Example
`heli y #bedav 100`

Error Example
User: `heli cool`
RM: `"Helicopter" is a yes/no field and "cool" is not a valid response. Please respond with 'yes' or 'no'.`

Checklist

Checklist fields are the first 'multi-part' fields that we have discussed. They are made up of a collection of previously-defined boolean fields. Below is an example using the field "Services" which represents the different services that a hospital could offer.

User: `serv`
RM:

```
Services
Respond with the numbers of your selections separated by comma:
1 Neurosurgery
2 Corpse Removal
3 MRI
4 CT Scan
5 Ambulance
```

User: `1-3,5`

In the above example the user was presented with the options and then selected neurosurgery, corpse removal, MRI, and ambulance. The selection system is similar to git's, and allows dashes for 'through' and commas for 'and'. The above example could be read "1 through 3, and five".

Additional Considerations

There are several things that complicate multi-step responses. First, SMS messages may be delayed for hours or not delivered at all. If a user starts a response to a multi-part field, the system should still accept the final response hours later. A time-out period should be definable at the terminal.

Secondly, the user could enter an invalid response after beginning the answering process. The system should handle this gracefully.

Example
User: `serv`
RM:
```
Services
Respond with the numbers of your selections separated by commas
1 Neurosurgery
2 Corpse Removal
3 MRI
4 CT Scan
5 Ambulance
```
User: `all of the above`
RM: `There was a problem with you answer to "Services". Please look at our previous text and respond with the numbers of your selections separated by commas.`
User: `1,4,5`

There should be a configurable 'fail limit' so that if the user does this too many times they just get a response like: "That was an invalid response for the field 'Services'" and the system should reset.

Lastly, since checklists are made up of many boolean fields, users could also respond to each field individually.

Example: `neurosurg yes`

Multiple Choice

Multiple choice fields are like checklists, except that users can only choose one of the multiple options. Thus, instead of being based on a collection of boolean fields, they are simply a text field with a discrete set of valid responses.

Example
User: `hosptype`
RM:
```
Hospital Type
Respond with the number of your choice
1 Military
2 University
3 Clinic
4 Large public
```
User: 3

Multiple choice error handling is similar to checklist error handling.

Example
User: `hosptype`
RM:
```
Hospital Type
Respond with the number of your choice
1 Military
2 University
3 Clinic
4 Large public
```
User: `Large`
RM: `There was a problem with your answer to "Hospital Type". Please look at our previous SMS and respond to it with the number of your selection.`

In addition to the numeric response method, users that already know the possible responses should be able to respond to multiple-choice fields as they would to plain text fields.

Example: `hosptype military`

Additionally, if the user is just a few letters off, fuzzy string matching algorithms should automatically 'figure out' what the user was attempting to input.

Example
User: `hosptype unversity`
RM: `"Unversity" was interpreted as "University"`

Additional Commands

Info

In addition to answering questions, users should be able to get information about the questions that they are answering. Each field has an attached 'info snippet' that will be displayed if the user asks for it in one of the following manners

```
info surgbedav
help surgbedav
? surgbedav
```

For plain text, numeric, and boolean fields only, the user should be able to text in only the name of the field and receive info about it like this:

```
surgbedav
```

Beyond information about specific fields, the user should also be able to get information about how to use the system. This information should start when the user submits the `'help'` or `'info'` commands. For example:

User: `afioejww`
RM: `Your response could not be interpreted. Respond to this SMS with 'help' to get more information about this system.`
User: `help`
RM: `This is a data collection system for the hospitals in Port-au-Prince, Haiti. For information on how to submit data, respond to this SMS with 'data'.`
User: `data`
RM: `You can submit data by sending an SMS with a field's abbreviation followed by the response you wish to send. For a list of fields, respond with "fields".`

Etc…

This section should be spec'd out more.

Haiti-Specific Commands
In addition to all of these commands, the situation in Haiti requires special, additional functionality because the users are tracking data about hospitals. The users need to be able to indicate which hospital they are submitting data about. Hospital identification is achieved through a web-style namespace system where dots are used to separate hierarchies.

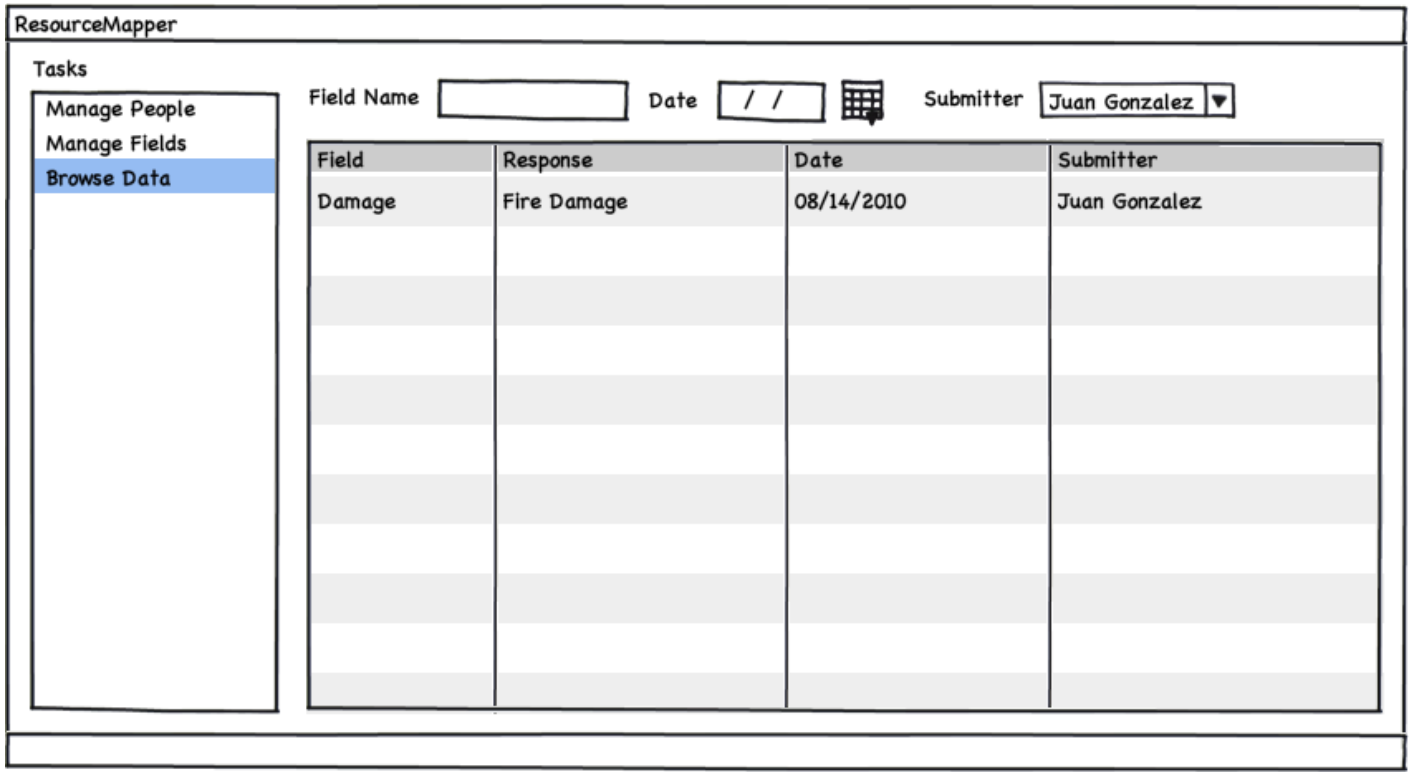
Example
`PaP.downtown.stvincents.pub`
`paho.org/HealthC_ID/1234567`

As you can see, the dots & slashes separate hierarchies of identifying information. ResourceMapper does not need to validate or parse these IDs, but it does need to allow the users to modify their hospital id. This should be done by designating a field like 'homehosp' or 'hospid' that the users can submit like a plain text field. These hospital ids are transmitted with the rest of the information in the transmission step.

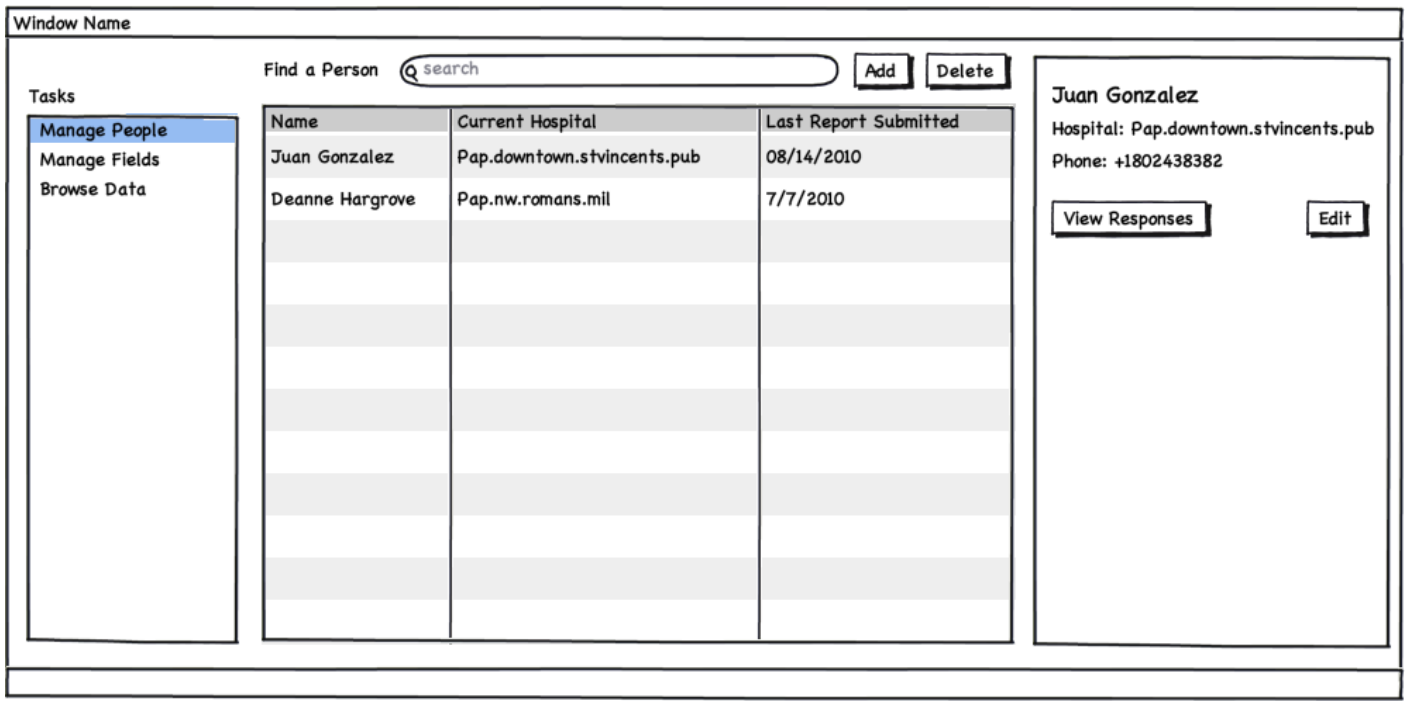
Lastly, it is possible that a hospital system will have a namespace prefix and assign numeric IDs per hospital (like in the second example). In this case we need to provide the ability to add a prefix to user's hospital IDs.

Part 2: At the Terminal
The users of ResourceMapper will require a basic functionality at the terminal, including the ability to manage their contacts, manage their fields, and view their data. A mockup of managing fields has already been shown, but the remaining mockups are below.

Browse Data – screen for viewing and sorting responses



Manage People – screen for managing your contacts. Haiti specific.



The point of this screen is to allow terminal admins to change a user's default hospital id.

Part 3: Transmission

The instance of ResourceMapper that we are creating with Google requires that the submitted data be forwarded to an app on Google App Engine. To do this, fields will be mapped to elements in an XML schema. When ResourceMapper receives a new response, an XML document will be generated and sent to Google App Engine via HTTP POST.

Google and Medic have already defined a set of fields to be collected in the installation in Haiti. It is important to keep in mind that this field list is only for the Haiti installtion, and could change. Below is a sample XML document that contains all possible fields.

```
<?xml version="1.0" encoding="utf-8"?>
<entry xmlns="http://www.w3.org/2005/Atom"
    xmlns:status="http://schemas.google.com/status/2010"
    xmlns:gs="http://schemas.google.com/spreadsheets/2006">
  <author>
<!-- Always a globally unique number, including "+" and country code.      -->
    <uri>tel:+50912345678</uri>
  <author>

<!-- Unique ID of the facility to be updated.  Haiti users should mostly   -->
<!-- only deal with the numeric part, so the producer of this XML entry    -->
<!-- will generally prepend "paho.org/HealthC_ID/" for transmission.       -->
  <status:subject>paho.org/HealthC_ID/1234567</status:subject>

<!-- Always a complete RFC 3339 timestamp in UTC.                          -->
  <status:observed>2010-06-29T15:27:39Z</status:observed>

<!-- Every entry has exactly one <status:report> element, which contains   -->
<!-- exactly one <status:record> element, which contains the data fields.  -->
  <status:report type="{http://schemas.google.com/status/2010}record">
    <status:record>

<!-- All field values are specified with the <gs:field> tag.  Each field   -->
<!-- can appear at most once, and all are optional.  Any field can be      -->
<!-- present and empty, which indicates that this update should erase the  -->
<!-- existing contents the field.  Field values are freeform strings       -->
<!-- unless otherwise indicated in comments below.                         -->

      <gs:field name="title">Title</gs:field>

      <gs:field name="alt_title">Alternate Title</gs:field>

<!-- The value of "available_beds", if present, must be an integer.        -->
      <gs:field name="available_beds">55</gs:field>

<!-- The value of "total_beds", if present, must be an integer.            -->
      <gs:field name="total_beds">66</gs:field>

<!-- "services" contains zero or more values separated by commas.          -->
<!-- Permitted values are:                                                 -->
<!--     GENERAL_SURGERY                                                   -->
<!--     ORTHOPEDICS                                                       -->
<!--     NEUROSURGERY                                                      -->
<!--     VASCULAR_SURGERY                                                  -->
<!--     INTERNAL_MEDICINE                                                 -->
<!--     CARDIOLOGY                                                        -->
<!--     INFECTIOUS_DISEASE                                                -->
<!--     PEDIATRICS                                                        -->
```

```
<!--     POSTOPERATIVE_CARE                                                 -->
<!--     REHABILITATION                                                     -->
<!--     OBSTETRICS_GYNECOLOGY                                              -->
<!--     MENTAL_HEALTH                                                      -->
<!--     DIALYSIS                                                           -->
<!--     LAB                                                                -->
<!--     X_RAY                                                              -->
<!--     CT_SCAN                                                            -->
<!--     BLOOD_BANK                                                         -->
<!--     MORTUARY_SERVICES                                                  -->
      <gs:field name="services">ORTHOPEDICS,CARDIOLOGY,X_RAY</gs:field>

      <gs:field name="contact_name">Name</gs:field>

      <gs:field name="phone">+12 345-678-90</gs:field>

      <gs:field name="email">user@example.com</gs:field>

      <gs:field name="department">Ouest</gs:field>

      <gs:field name="district">Léogâne</gs:field>

      <gs:field name="commune">Petit-Goâve</gs:field>

      <gs:field name="address">123 Example Street</gs:field>

      <gs:field name="location">18.3037,-72.8636</gs:field>

      <gs:field name="accuracy">Description of location accuracy.</gs:field>

      <gs:field name="organization">Example Organization</gs:field>

<!-- "organization_type" contains one of these permitted values:          -->
<!--     PUBLIC                                                             -->
<!--     FOR_PROFIT                                                         -->
<!--     UNIVERSITY                                                         -->
<!--     COMMUNITY                                                          -->
<!--     NGO                                                                -->
<!--     FAITH_BASED                                                        -->
<!--     MILITARY                                                           -->
<!--     MIXED                                                              -->
      <gs:field name="organization_type">PUBLIC</gs:field>

<!-- "category" contains one of these permitted values:                   -->
<!--     HOSPITAL                                                           -->
<!--     CLINIC                                                             -->
<!--     MOBILE_CLINIC                                                      -->
<!--     DISPENSARY                                                         -->
      <gs:field name="category">HOSPITAL</gs:field>

<!-- "construction" contains one of these permitted values:               -->
<!--     REINFORCED_CONCRETE                                                -->
<!--     UNREINFORCED_MASONRY                                               -->
<!--     WOOD_FRAME                                                         -->
<!--     ADOBE                                                              -->
      <gs:field name="construction">REINFORCED_CONCRETE</gs:field>

      <gs:field name="damage">Text describing building damage.</gs:field>

<!-- "operational_status" contains one of these permitted values:         -->
<!--     OPERATIONAL                                                        -->
```

```
<!--     NO_SURGICAL_CAPACITY                                              -->
<!--     FIELD_HOSPITAL                                                    -->
<!--     FIELD_WITH_HOSPITAL                                               -->
<!--     CLOSED_OR_CLOSING                                                 -->
<!-- This field indicates the level of functionality.  "OPERATIONAL" means -->
<!-- fully operational; "NO_SURGICAL_CAPACITY" means unable to perform     -->
<!-- surgeries; "FIELD_HOSPITAL" means as functional as a field hospital;  -->
<!-- "FIELD_WITH_HOSPITAL" means as functional as a field hospital that is -->
<!-- co-located with a hospital; CLOSED_OR_CLOSING" means closed or in the -->
<!-- process of closing.                                                   -->
      <gs:field name="operational_status">OPERATIONAL</gs:field>

      <gs:field name="comments">Arbitrary comment text goes here.</gs:field>

<!-- "reachable_by_road" contains one of these permitted values:          -->
<!--     TRUE                                                              -->
<!--     FALSE                                                             -->
      <gs:field name="reachable_by_road">TRUE</gs:field>

<!-- "can_pick_up_patients" contains one of these permitted values:       -->
<!--     TRUE                                                              -->
<!--     FALSE                                                             -->
      <gs:field name="can_pick_up_patients">FALSE</gs:field>

    </status:record>
  </status:report>
</entry>
```

Notice that the hospital information XML is wrapped inside of an Atom <entry> element. The phone number of the submitter is included in the entry/author/uri element and the actual data is further packaged inside of a <status:report> element, with header info included in the <status:subject> and <status:observed> elements.

Our current plan is to distribute the Google version of ResourceMapper with all these fields and mappings already in the system, while planning for change. As such, we need a way to specify what the 'Google schema name' of each field is. This simply requires another label & text field in the rightmost panel on the "Manage Fields" screen.

Lastly, since internet service may be sporadic at the site of the FrontlineSMS:Server, ResourceMapper should be able to stockpile reports and send them en masse when internet is available. In all other situations, the reports should be POSTed as soon as they are received, to keep information as up-to-date as possible.

Part 4: Web App

Once the data is received by the ResourceMapper app on Google App engine, it will be accessible to logistics coordinators and others that need information about the status of hospitals. Below is a mockup of the web app.

Resource Mapper

Show: All specialties

Facility name	Facility specialites	Open/total beds	
Achahaie Hospital Saint Joseph De Galilee	Cardiovascular, OB/GYN, Oncology	90	100
Adventiste Diquini	Neurosurgery, Pediatrics	72	280
Albert Schweitzer	Radiology	47	50
Anse A Foleur	OB/GYN, Oncology	38	45
Anse Rouge (A.F.M.E)	Neurosurgery	32	50
Argentinian Hospital	Orthopaedics	24	30
Asile Franrais	Pediatrics, Radiology	15	15
B-FAST Field Hospital	Rheumatology, OB/GYN	12	20
Bassin Blue Hospital Health Center	Oncology	10	50
Bainet	Neurosurgery	8	40
Baraderes	Orthopaedics	6	100
Beraca La Pointe	Pediatrics	5	80
Beudet (Defilee)	Radiology	5	32
Bombardopolis	Rheumatology	4	15
Bon Samaritain	OB/GYN	3	20
Bonne Fin	Oncology	3	100
Bonneau St. Joseph	Neurosurgery	2	20
Brazillian Field Hospital	Orthopaedics	1	300
C.M.S.	Pediatrics	1	40
Cerca La Source Hospital Health Center	Radiology, Pediatrics Cardiovascular	0	800
Chantal Dispensary Saint Jeanne	OB/GYN	0	200
Climapev	Oncology	0	15
Corail Hospital Corail	Neurosurgery	-	-
Canadian Field Hospital	Orthopaedics	-	-