Short URL to this doc: <http://bit.ly/xOqHPj>

# National Consolidated Fingerprinting Design Specification

[Summary](#_crfhn0x1zj9f)

[Basic Architecture](#_y7gyxmfsh3m6)

[Synchronization Processing Steps](#_clcv0r9x6axd)

[De-Duplication Functionality](#_9j02kcpjtkki)

[Detailed Design for the Synchronization Tasks](#_c9fjlvcuhjgh)

[Setup](#_6f4aac2u5mbi)

[1. Windows Side](#_8974ol57hozs)

[2. iSanté Side](#_9p6utnzb04kd)

[Algorithm](#_v2afpu4wvfzo)

[Folder where DuplicateTemplateFound.log files are stored](#_5m9rjuj3odcs)

[Logs](#_vkv28f1y52td)

[1) a log of the execution in /var/log/itech/](#_vt5rqwlukngm)

[2) log of event in table eventLog in database](#_w6msdmaviqc5)

[3) Archives](#_m2aqa2h6f7kc)

[Steps executed daily](#_kb4aigb8vtcb)

[Detailed Design for National De-Duplication](#_d5gwgalo6ifn)

[Alternative Design Considerations](#_lc7elfarc816)

[National Fingerprint Server Location](#_ebtk0l8lefvd)

[Increasing the Size of the patientID Field](#_y5yfvkoml8km)

[Interactive Synchronization](#_82kl28mwosar)

[National Fingerprint Database Replication](#_tdlcvqxhkc7a)

[HTTP POST rather than SFTP (Suggestion by Eric Webster)](#_s5dqbebi5jdv)

### Summary

Patients are always assigned a local site identifier (patientID) when initially registered. This identifier is also the ID associated with their fingerprints in the local fingerprint database and the ID that is transmitted to the national fingerprint database. For national synchronization purposes, a patient’s first **local identifer** becomes her **master patient identifier** (masterPID). The national fingerprint server (administered by UGP in Port-au-Prince) stores and supplies this master identifier to additional sites the patient may visit. Local identifiers are associated with master identifiers whenever a local fingerprint database is synchronized with the national fingerprint database. Frequency of synchronization is proposed to be daily or weekly, depending upon ongoing experience with the volume of transactions and the speed of synchronization processing.

### Basic Architecture

In iSanté, patients are managed via the internal patientID, which is the primary key in iSanté’s patient table and a foreign key in most of the other tables in the database. The patientID is a varchar(11) field consisting of the first five characters of the local site’s sitecode concatenated with the auto-generated person\_id (a sequential number). Thus, a typical patientID, created when a patient is initially registered at a site is, for example, 7110025, site 71100’s 25th registered patient. The patient table also stores this information redundantly in location\_id and person\_id, respectively, for compatibility with OpenMRS and its obs table.

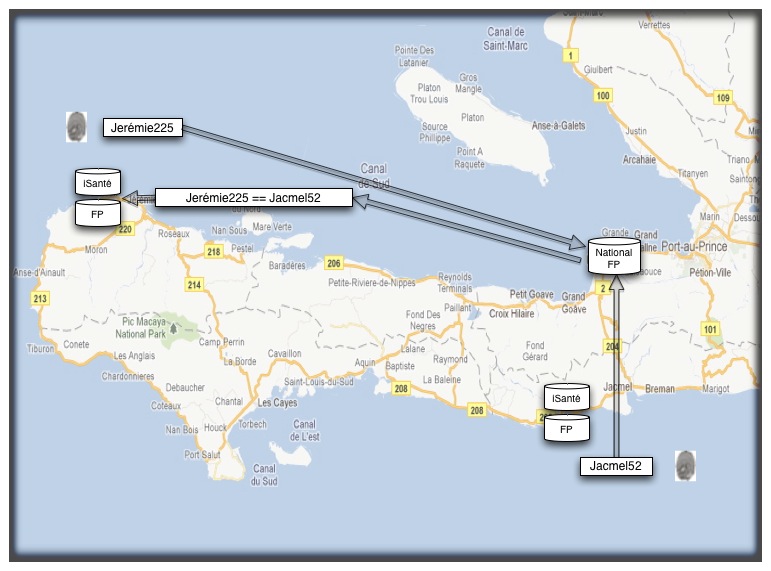
The patient table also contains the column masterPID. MasterPID was establish when patient transfer functionality was added to the system so that ***transferred from***information could be retained. Initially, the masterPID has the same value as patientID. Upon a patient being transferred, the ***transferred to*** site generates a new patientID for the patient and stores the ***transferred from*** site’s patientID as its masterPID. Here is an example:

|  |  |  |
| --- | --- | --- |
| Step | patientID | masterPID |
| registered at site1 | 7110025 | 7110025 |
| transferred to site2 | 21100645 | 7110025 |

Because fingerprinting forces uniqueness, the masterPID will be used as the fingerprint key in the local and national consolidated databases. Local sites will submit their patientID/fingerprint pairs to the national consolidated database. New prints will be successfully added with the local site’s patientID as the national, masterPID fingerprint key. When duplicate prints are submitted, they are detected as duplicates and the current national-level fingerprint key is returned, becoming the masterPID at the submitting site, since that patient had already submitted prints at another site. The process is analogous to the patient transfer scenario above:

|  |  |  |
| --- | --- | --- |
| Step | patientID | fingerprint key (masterPID) |
| Submit patientA from site1 | 7110025 | 7110025 |
| Submit patientA from site2 | 21100645 | 7110025 |

Also, see the diagram below, where a patient is first seen in Jacmel and later seen in Jerémie. Note that the Jacmel patientID becomes the masterPID for the patient in Jerémie and it is associated with the patientID within iSanté. It also becomes the key associated with the patient’s prints in the Jerémie fingerprint database.



### 

### 

### 

### Synchronization Processing Steps

The processing steps above, required to synchronize the national consolidated fingerprint database with local site patient information, are proposed here as a daily or weekly task rather than a task that could potentially hinder the registration process. There are at least three reasons for this approach:

1. Sites need to continue operations when network services are unavailable.
2. Local site - national site interconnect performance is likely to be slow even in good conditions and even in areas less challenged than Haiti.
3. Determination of duplicate patients is not time-critical for clinical care.

The batch task involves a number of steps which utilize the M2Sys merge capability--software that can copy the contents of the local fingerprint database into the national database. These steps are:

1. Copy the local database to the national database.
2. Receive resultant files containing inserted, matched, and duplicate records, respectively, for each local fingerprint.
3. Process the records in the duplicates file to update local masterPIDs and local fingerprint database fingerprint keys.
4. Optionally request patient transfers for duplicate records via alerts and/or a transfers request page in iSanté.

The merge program runs on the local fingerprinting server (Windows) and the resultant files are returned there. A Perl program on the iSanté server (Linux) reads those files and performs the appropriate updates to the local iSanté database and the local fingerprint database.

Alerts and/or a pending transfer requests page in iSanté would show the patients that were detected as duplicates in the national fingerprint database and provide a simple interface for requesting their clinical summary or records transfer.

### De-Duplication Functionality

De-duplication is dealt with differently at the local vs. the national level. Local de-duplication consists of making sure that patients aren’t registered more than once at the local site and that all visits are recorded against that single registration. This could occur if the registration clerk doesn’t search in iSanté for the patient prior to registering, or if a search fails to find a patient that was already registered because of a misspelled name or incorrect site identifier. This data quality problem will be detectable via fingerprinting, since searching by print instead of by name or identifier should always find previously registered patients. iSanté also has a data quality report for finding possible duplicate registrations. In any case, iSanté Version 9.0 RC1 included the ability to merge records when they had been entered against two or more separate registrations of the same patient. In addition, as of iSanté Version 13.3, whenever a patient with an existing fingerprint is merged to a designated patient without a print, the designated patient’s masterPID becomes the fingerprint handle in the local database. At the same time, the handle must be changed in the national database as well -- if it isn’t, the synchronization process will set it back to the merged patient’s patientID.

The second type of de-duplication is at the national level. National de-duplication is really a misnomer here, because at the national level the informational goal is to account for the fact that the same patient may visit and be treated at multiple sites within Haiti. So the primary de-duplication goals at the national level are:

1. Getting accurate counts of unique individual patients at the national level
2. Determining the most recent demographic and geographical information for each patient
3. Determining accurate adherence and treatment time-series information for patients across sites

The synchronization tasks above will result in masterPID values in the iSanté consolidated databases that are accurate pointers into all records for unique individuals in Haiti and thus provide a straightforward way to satisfy the de-duplicated informational goals needed at the national level.

### Detailed Design for the Synchronization Tasks

#### Setup

##### 1. Windows Side

The goal is to setup a task that will merge the local fingerprint database with the national consolidate one, retrieve the duplicate fingerprint entries and upload that list to the iSanté server for deduplication.

1. copy the fpmerge folder and its content from ~/support/fpmerge in the iSanté code to C:\Program Files\BioPlugin\fpmerge

2. Replace the existing curl.exe with the appropriate one for your operating system (<http://curl.haxx.se/download.html>)

3. Update the "hybrid\_datamerger.ini" file according to your requirement. It contains PORT and IP of Destination BioPlugin Server. It also links to your local Database.

a. PORT: Port of the destination BioPlugin Server

b. IP: IP of the destination BioPlugin Server

c. ConnectStr: ODBC Connection to the source (local) BioPlugin Database

4. Update the "hybrid\_datamerger.cmd" file with your login information to the iSanté Server (lines 20-22)

5. Verify the installation:

a. Execute "hybrid\_datamerger.cmd"

b. Consult the hybrid\_datamerger.log to see if the transaction worked or curlErrorFile.htm if it exists to debug

c. The uploaded DuplicateTemplateFound.log should be found in to iSanteMergers\_done folder with today's date

6. Set up a Scheduled task in Windows that will execute C:\Program Files\BioPlugin\fpmerge\hybrid\_datamerger.cmd daily.

IMPORTANT >>> In destination machine, BioPlugin Server needs to RUN. Otherwise it won't be work.

--- Files in this folder ---

Read Me First.txt: documentation

curlSendFile.cmd: secondary executable that uploads a file to the serveur via HTTP with cURL and calls the iSante side merging process

hybrid\_datamerger.cmd: main executable file that starts the fingerprint database merge process with hybrid\_datamerger.exe and then calls curlSendFile.cmd for every DuplicateTemplateFound.log in the iSanteMergers\_todo folder

hybrid\_datamerger.exe: executable file created by M2Sys that merges the local database with a destination database. It uses the hybrid\_datamerger.ini file for its configuration.

hybrid\_datamerger.ini: configuration file for hybrid\_datamerger.exe.

hybrid\_datamerger.log: log of the previous executions of hybrid\_datamerger.cmd

dataMerge.txt: file created by the hybrid\_datamerger.exe (not used in our process)

DuplicateIDFound.txt: file created by the hybrid\_datamerger.exe (not used in our process)

MergedRecords.txt: file created by the hybrid\_datamerger.exe (not used in our process)

##### 2. iSanté Side

Everything should run without any additional setup.

The batch file that runs mergeFingerprintDate() is ~/patientStatusBatch.php

You can verify that the merger process works by monitoring the various logs created by the mergeFingerprintDate() function.

#### Algorithm

Here is pseudocode for what will probably be Perl code in the actual implementation:

#Copy the local database to the national database.

## an ODBC reference to the MySql database must be set up

## an ini file (hybrid\_datamerger.ini must be set up to point at the MySql DB via the ODBC reference

hybrid\_datamerger.ini:

[Server]

PORT=1200

IP=localhost

ConnectStr=Driver={MySQL ODBC 3.51 Driver};Server=localhost;Database=myDataBase; User=myUsername;Password=myPassword;Option=3;

## on the Windows server, perhaps via a scheduled job, run the following command

system ('ssh <windowsServer> hybrid\_datamerger.exe FP1');

## copy the file DuplicateTemplateFound.txt to the iSanté server

system ('sftp <windowsServer>'; 'get DuplicateTemplateFound.txt'; )

##Read and process DuplicateTemplateFound file:

foreach (record) {

//Update patient.masterPID with value from file

UPDATE patient SET masterPID = <masterpid> WHERE patientID = '<patientID>';

//Update registration record with current date so that masterPID is pushed to consolidated DB

UPDATE encounter SET lastModified = now() WHERE patientID '<patientID> AND encounterType in (10,15)';

//Update local FP ID with value from file

changeFingerprintID (<patientID>, <masterPID>);

//Issue alert or listing/report or automatically request patient transfer for these patientIDs

execWithProgress('someMessage','transfer-request.php',param1);

param1:{"report":"214","format":"html","siteName":"11100","pid":"1110014654","lang":"en","site":"11100","transferTo":"35101","source":"natlServerProxy.php"}')

}

## if this process proves to be too slow with large local databases, we can send an incremental fingerprint database based upon only new records

## probably don't attempt this in the first release, but see how performance goes

## for reference, here are the tables in the fingerprint database:

CREATE TABLE TEMPLATES (

PERSONID varchar(50) NOT NULL default '0',

ENGINETYPE varchar(10) default NULL,

TEMPLATEID bigint(11) NOT NULL default '0',

TEMPLATETYPE bigint(11) default NULL,

TEMPLATEFEATURE blob,

TEMPLATESIZE bigint(20) unsigned NOT NULL default '0'

) TYPE=MyISAM;

CREATE TABLE PERSON (

PERSONID varchar(50) NOT NULL default '0',

REGISTRATIONNO varchar(100) default NULL,

TYPEID bigint(11) default NULL,

REGISTEREDDATE datetime NOT NULL default '0000-00-00 00:00:00',

STATUS bigint(11) default NULL

) TYPE=MyISAM;

CREATE TABLE EXPERT\_DATA (

ENTRY\_ID bigint(15) NOT NULL default '0',

ENTRY\_DATE datetime NOT NULL default '0000-00-00 00:00:00',

FINGER\_DATA blob,

FINGER\_TEMPLATE\_SIZE bigint(11) default NULL,

FINGER\_TEMPLATE\_VALUE bigint(11) default NULL,

PERSON\_REGID varchar(100) default NULL,

PERSON\_TEMPLATE\_SIMILARITY bigint(11) default NULL,

SIBLING\_ENTRY\_ID bigint(15) default '0',

SIBLING\_TEMPLATE\_SIMILARITY bigint(11) default NULL

) TYPE=MyISAM;

## Run a query against the fingerprint db to generate a new table containing only recent inserts:

system ('mysql fpDB

truncate lastPerson;

insert into lastPerson select \* from person where registereddate >= '<lastrun>';

exit;')

## Clone the local db and replace the full table with the table generated in #1

system ('export entire db to tempdb'

'mysql tempdb

truncate person;

insert into person values lastPerson;'

)

## Now run the original merge (see above) with the cloned db (tempdb) as input

#### Folder where DuplicateTemplateFound.log files are stored

/var/backups/itech/fpDuplicateLogs/

(accessible by php server because it is owned by www-data)

#### Logs

##### 1) a log of the execution in /var/log/itech/

Done via the ~/batch-jobs.php file

##### 2) log of event in table eventLog in database

a record is added to the table eventLog in the database on completion of the process with function recordEvent().

##### 3) Archives

/var/backups/itech/fpDuplicateLogs/processed

the file ~/config-linux/scripts/110var.sh has been updated accordingly with the following lines:

#this directory holds fingerprint duplicates data

mkdir -p /var/backups/itech/fpDuplicateLogs/processed

chown -R itech:www-data /var/backups/itech/fpDuplicateLogs

chmod -R 771 /var/backups/itech/fpDuplicateLogs

#### Steps executed daily

1. Scheduled task **C:\Program Files\BioPlugin\fpmerge\hybrid\_datamerger.cmd** on local windows computer hosting fingerprint database.
   1. :: hybrid\_datamerger.exe FP1 Merges Local Fingerprint Database with distant Fingerprint Database defined in hybrid\_datamerger.ini
   2. It imports the DuplicateTemplateFound.log file to the current folder
   3. DuplicateTemplateFound.log is copied to the iSanteMergers\_todo folder and renamed with today’s date iSanteMergers\_todo.
   4. For each DuplicateTemplateFound.log in iSanteMergers\_todo, executes curlSendFile.cmd that uploads the file and calls the iSante server side merging function.
   5. exits
2. The cURL script uploads a DuplicateTemplateFound.log with today’s date (eg DuplicateTemplateFound-20120120.log) and calls **~/upload.php**. Function mergeFingerprintData(‘NameOfTheFile.log’); from **~/backend/materializedViews.php** is executed.
   1. It output the received data from the cURL script to **/var/backups/itech/fpDuplicateLogs/NameOfTheFile.log** (eg /var/backups/itech/fpDuplicateLogs/DuplicateTemplateFound-20120120.log)
   2. It opens the ‘**NameOfTheFile.log**’ file if it exists
   3. For each duplicate
      1. it updates masterID in patient
      2. it updates lstModified in encounter
      3. it applies changeFingerprintID()
   4. it moves the **NameOfTheFile.log** file to **/var/backups/itech/fpDuplicateLogs/processed** folder and adds the day of processing (example: DuplicateTemplateFound-20120120.log)
   5. It adds an event in table eventLog and saves the duplicateIDs with function recordEvent()

### Detailed Design for National De-Duplication

Because the basic architecture of iSanté, at the local level, supports elimination of duplicated patient via both a report identifying duplicates and a merge process to combine duplicates; and at the national level, identifies duplicates across sites via fingerprint synchronization, what remains is to rework existing reports to use masterPID rather than patientID. The most simple example of this, supposing that the national-level patient table contains only the two records discussed in the opening section above:

|  |  |  |
| --- | --- | --- |
| Step | patientID | fingerprint key (masterPID) |
| Submit patientA from site1 | 7110025 | 7110025 |
| Submit patientA from site2 | 21100645 | 7110025 |

If we select for a count of patients via patientID:

SELECT COUNT(DISTINCT patientID) FROM patient;

We get a count of two (2). If we use masterPID:

SELECT COUNT(DISTINCT masterPID) FROM patient;

We get a count of one (1).

Expanding upon this, let’s take a more complicated query (report) in the existing system:

### Alternative Design Considerations

#### National Fingerprint Server Location

It is proposed that the national fingerprinting server be located in Port-au-Prince and administered by UGP. However, this should be discussed and perhaps modeled to detect potential performance issues.

#### Increasing the Size of the patientID Field

The current size of the patientID field is varchar(11). This allows for a maximum number of patients at a single site of one million (999,999). If this is judged to be too small a number at a particular site, it could be increased. The population of Haiti is currently nearly ten million.

If patientID was increased to varchar(12) or varchar(13), this would correspond to a capacity at each sitecode of 10 million (9,999,999) and 100 million (99,999,999), respectively.

The person\_id field is currently an unsigned int, which can store the above maximums (9,999,999, 99,999,999).

There are something less than 60 base tables that contain patientID and would need to be altered to the larger size. However, no other database changes would be required to make this change.

#### Interactive Synchronization

As an alternative to daily or weekly synchronization, the system could be designed to synchronize at point of care, providing both masterPID information and immediate patient summary information. This approach is not considered feasible at this time.

#### National Fingerprint Database Replication

Point of care synchronization could also be achieved by frequent distribution of the national fingerprint database to each local site. This approach is not considered feasible at this time.

#### HTTP POST rather than SFTP (Suggestion by Eric Webster)

Everything looks pretty good but I have a suggestion. Instead of using SCP to copy a file over and then process it I think it would be a lot more reliable if the Windows task use HTTP POST to send the duplicate data directly to a PHP script on the server.

One problem with the SCP + CRON approach is that it relies on the two systems getting the timing just right every time. What happens if the SCP is successful but for some reason the CRON script never runs? Will the log file be overwritten by the next execution of SCP?

There is also the matter of what account would need to be used. Will an administrator need to setup a new system account on the Linux side and install SSH keys in order to get this to work? The is quite a lot of administrative burden. It would be much better if credentials or authorization information for this could all be entered through the web interface of the application.