MerchantOS-Highrise-Sync

Functional Specification & Design Documentation

Author: Erika Ellison

Last Updated: 2012-06-08

Functional Specification

Overview:

MerchantOS-Highrise-Sync syncs contact data stored in MerchantOS, a point-of-sale web application, and Highrise, a customer relations management web application.

An account management page allows users of MerchantOS to sign up for the sync service by providing their Highrise API credentials (which they can also update at any time on this page).

An initial sync is performed on sign-up, moving any Customer contact information from MerchantOS into Highrise, and any People contact information from Highrise into MerchantOS.

At regular intervals (likely every night), an incremental sync is performed for all subscribers, which looks for any new or modified contacts in either system to update in the other.

Scenarios/Use Cases:

A user would like to start using Highrise with the contact data in their existing MerchantOS account.

A new user signs up for a MerchantOS account and wants to import their contacts from their existing Highrise account.

Non-Goals:

- checking for duplicate contacts (contact data that is already entered in both applications).
- merging changes made to the same contact in both applications in the same period between syncs.

For the above reasons, it is strongly encouraged that users of this service add and modify contact data in only one of the two applications.

Desirable Additional Features:

- automated tests
- automatic notification when a sync attempt fails due to invalid API credentials.

Design Documentation

Requirements:

A web server running PHP 5.3 or newer (with curl module installed and xsl enabled), and MySQL 5.1 or newer.

Assumptions:

The sync will be run when the user is not making changes to any of the contact data.
The user will only make changes to a particular contact in one application or the other between syncs.

Data Mapping:

See Figure 1.

Highrise requires a location for each address. Valid values are [work, home, other]. MerchantOS has no analogous field and only allows one address per Customer. The MerchantOS-Highrise-Sync app maps the Highrise address with the location “work” to the single MerchantOS address.

Highrise requires a location [work, home, other] for email addresses. MerchantOS uses a useType [primary, secondary] for email addresses. The MerchantOS-Highrise-Sync app maps primary to work, secondary to home, and does not map Highrise's other to MerchantOS.

MerchantOS can only accept two email addresses per Customer. Highrise can have an unlimited number of email address per Person. If there are more than two email addresses for a contact in Highrise, the MerchantOS-Highrise-Sync app transfers only the email addresses with locations “work” and “home”.

MerchantOS customers that have been flagged noEmail/noPhone/noMail will still have their contact information transferred to Highrise by the MerchantOS-Highrise-Sync app, but these MerchantOS fields will have no analogous values in Highrise.

Architecture

See Figure 2 for a class diagram. In the initial sync, the service first creates a custom field in Highrise with which to keep track of MerchantOS's customer number. All customers and people are read in first, before any creation work is done (to avoid having to read in and check for already-synced contacts when syncing in the second direction). From the set of customers read in, a person is created from each customer's data, including their customer number, and sent to Highrise. From the set of people read in, a customer is created from each person's data, and sent to MerchantOS. From the response, the service then updates the Highrise entry with the new MerchantOS customer number. In the incremental sync, the service reads in all modified or newly created contacts, and syncs them in the same order as above. After each call to sync, the last_synced_on field in the database record representing that account is updated with the current date and time. All exceptions are thrown up to the SyncAccount class, where they are sent to the DAO to be logged with contextual information.

See Figure 3 for an activity diagram of account_management.php, which is written procedurally with decision trees and subroutines.

Possible Problems:

Scaling: An sync of 10 contacts total takes approximately 2 seconds, so accounts with multiple thousands of contacts to sync could take several minutes to an hour.

User interference: any modification to the custom field created and used by this service will result in the sync service not performing as intended.

Timing: if the user adds or modifies contacts in either application too soon after sync() is called on their account (within a few seconds, as suggested by testing), it is possible that those contacts will not be synced appropriately.

Figure 1, Data Mapping

MerchantOS	Highrise	Notes
Customer	**Person**	
firstName	first-name	
lastName	last-name	
title	title	
company	company-name	
Contact	contact-data	
Addresses	addresses	
ContactAddress	address	MerchantOS allows only 1 ContactAddress, Highrise allows one address for each location
address1	street	
address2		
city	city	
state	state	
zip	zip	
country	country	
	location	Highrise valid location values { Work || Home || Other }
Phones	phone-numbers	
ContactPhone	phone-number	
number	number	
useType		MerchantOS valid useType values { Home || Work || Pager || Mobile || Fax }
	location	Highrise valid location values {Work || Mobile || Fax || Pager || Home || Skype || Other }
Emails	email-addresses	
ContactEmail	email-address	
address	address	
useType		MerchantOS valid useType values { Primary || Secondary }
	location	Highrise valid location values { Work || Home || Other }
Websites	web-addresses	
ContactWebsites	web-address	
url	url	
	location	Highrise valid location values { Work || Personal || Other }
Note		
note	background	
isPublic		only put note in background if isPublic == true ??
	subject_datas	
	subject_data	
customerID	value	
	subject_field	

Figure 2
Class Diagram

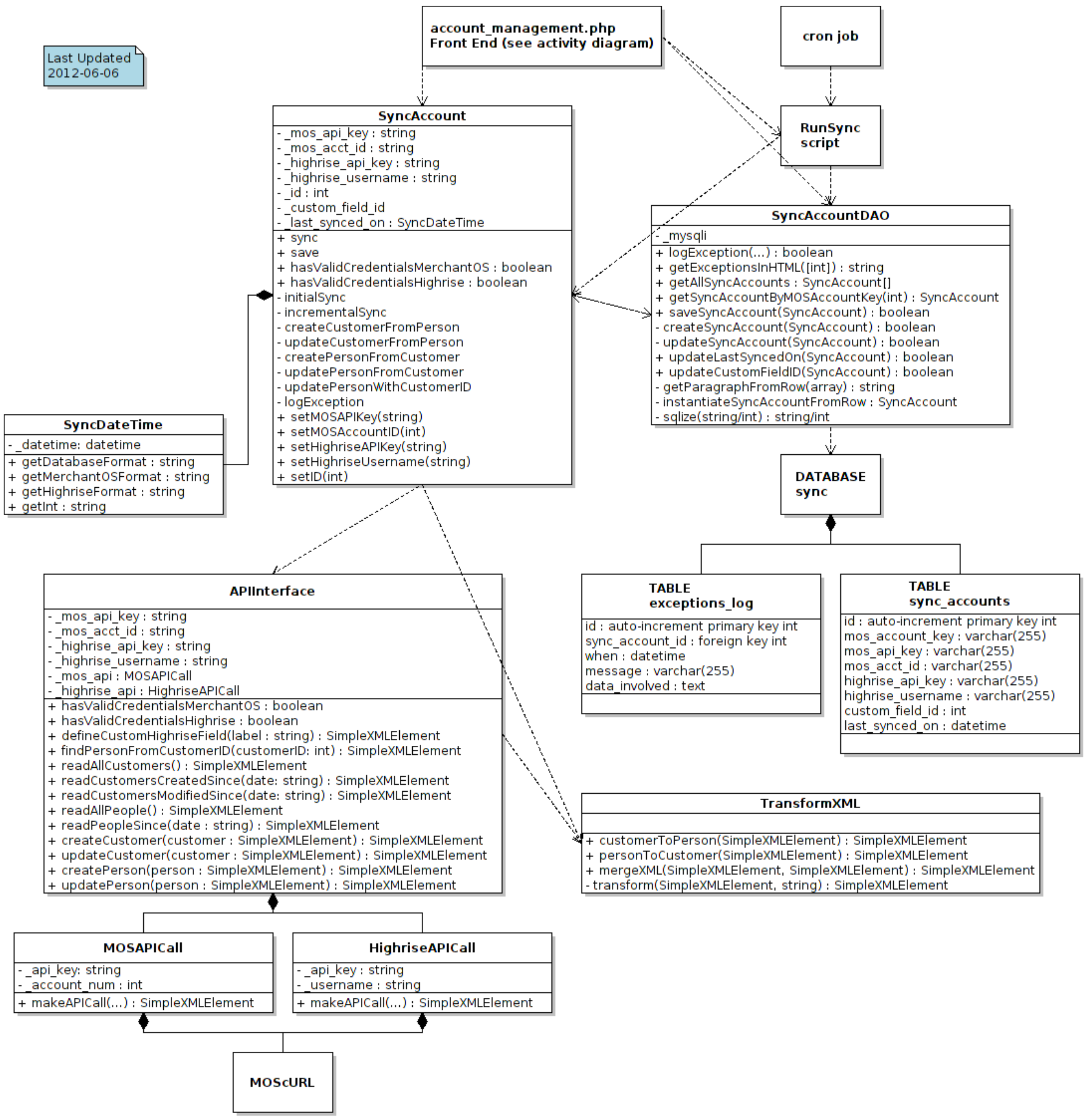
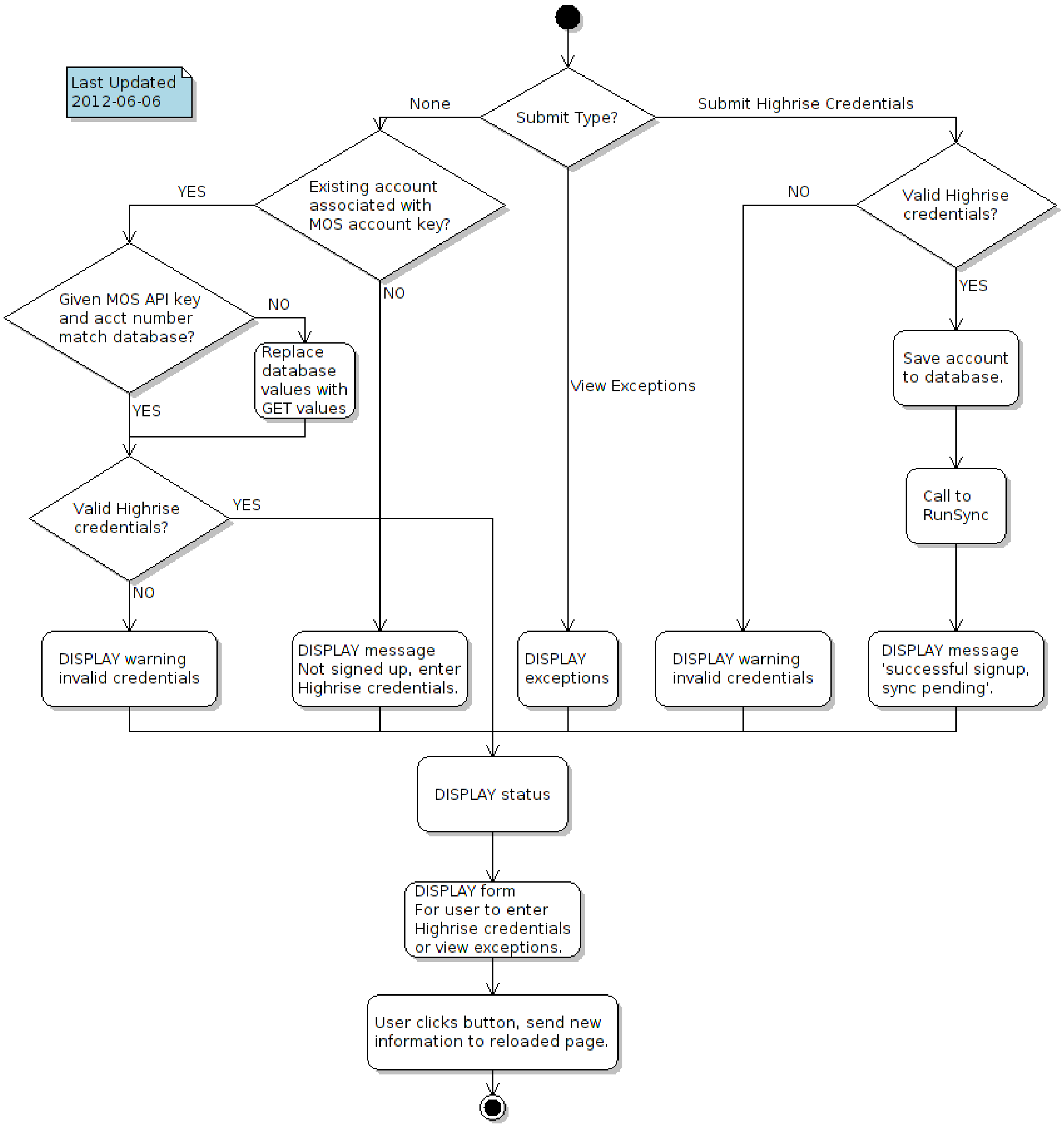

Last Updated 2012-06-06
account_management.php
Front End (see activity diagram)
cron job
RunSync script
SyncAccount
- _mos_api_key : string
- _mos_acct_id : string
- _highrise_api_key : string
- _highrise_username : string
- _id : int
- _custom_field_id
- _last_synced_on : SyncDateTime
+ sync
+ save
+ hasValidCredentialsMerchantOS : boolean
+ hasValidCredentialsHighrise : boolean
- initialSync
- incrementalSync
- createCustomerFromPerson
- updateCustomerFromPerson
- createPersonFromCustomer
- updatePersonFromCustomer
- updatePersonWithCustomerID
- logException
+ setMOSAPIKey(string)
+ setMOSAccountID(int)
+ setHighriseAPIKey(string)
+ setHighriseUsername(string)
+ setID(int)
SyncAccountDAO
- _mysqli
+ logException(...) : boolean
+ getExceptionsInHTML([int]) : string
+ getAllSyncAccounts : SyncAccount[]
+ getSyncAccountByMOSAccountKey(int) : SyncAccount
+ saveSyncAccount(SyncAccount) : boolean
- createSyncAccount(SyncAccount) : boolean
- updateSyncAccount(SyncAccount) : boolean
+ updateLastSyncedOn(SyncAccount) : boolean
+ updateCustomFieldID(SyncAccount) : boolean
- getParagraphFromRow(array) : string
- instantiateSyncAccountFromRow : SyncAccount
- sqlize(string/int) : string/int
SyncDateTime
- _datetime: datetime
+ getDatabaseFormat : string
+ getMerchantOSFormat : string
+ getHighriseFormat : string
+ getInt : string
DATABASE sync
TABLE exceptions_log
id : auto-increment primary key int
sync_account_id : foreign key int
when : datetime
message : varchar(255)
data_involved : text
TABLE sync_accounts
id : auto-increment primary key int
mos_account_key : varchar(255)
mos_api_key : varchar(255)
mos_acct_id : varchar(255)
highrise_api_key : varchar(255)
highrise_username : varchar(255)
custom_field_id : int
last_synced_on : datetime
APIInterface
- _mos_api_key : string
- _mos_acct_id : string
- _highrise_api_key : string
- _highrise_username : string
- _mos_api : MOSAPICall
- _highrise_api : HighriseAPICall
+ hasValidCredentialsMerchantOS : boolean
+ hasValidCredentialsHighrise : boolean
+ defineCustomHighriseField(label : string) : SimpleXMLElement
+ findPersonFromCustomerID(customerID: int) : SimpleXMLElement
+ readAllCustomers() : SimpleXMLElement
+ readCustomersCreatedSince(date: string) : SimpleXMLElement
+ readCustomersModifiedSince(date: string) : SimpleXMLElement
+ readAllPeople() : SimpleXMLElement
+ readPeopleSince(date : string) : SimpleXMLElement
+ createCustomer(customer : SimpleXMLElement) : SimpleXMLElement
+ updateCustomer(customer : SimpleXMLElement) : SimpleXMLElement
+ createPerson(person : SimpleXMLElement) : SimpleXMLElement
+ updatePerson(person : SimpleXMLElement) : SimpleXMLElement
TransformXML
+ customerToPerson(SimpleXMLElement) : SimpleXMLElement
+ personToCustomer(SimpleXMLElement) : SimpleXMLElement
+ mergeXML(SimpleXMLElement, SimpleXMLElement) : SimpleXMLElement
- transform(SimpleXMLElement, string) : SimpleXMLElement
MOSAPICall
- _api_key: string
- _account_num : int
+ makeAPICall(...) : SimpleXMLElement
HighriseAPICall
- _api_key : string
- _username : string
+ makeAPICall(...) : SimpleXMLElement
MOScURL

Figure 3
Acitivty Diagram of account_management.php



Last Updated 2012-06-06
Submit Type?
None
Submit Highrise Credentials
Existing account associated with MOS account key?
YES
NO
Valid Highrise credentials?
NO
YES
Given MOS API key and acct number match database?
NO
Replace database values with GET values
YES
View Exceptions
Save account to database.
Call to RunSync
Valid Highrise credentials?
YES
NO
DISPLAY warning invalid credentials
DISPLAY message Not signed up, enter Highrise credentials.
DISPLAY exceptions
DISPLAY warning invalid credentials
DISPLAY message 'successful signup, sync pending'.
DISPLAY status
DISPLAY form For user to enter Highrise credentials or view exceptions.
User clicks button, send new information to reloaded page.