DDE2 migration steps checklist

Task #	Sub Task	Description	Sample Command	Expected Outcome	Done
1.		Clone the migration tool from Github to a folder on the target machine	git clone https://github.com/BaobabHealthTrust/dde2_migration_tool.git migration/	A folder will be created with the name passed for the target folder which in this case is “migration/”	
2.		Clone the migration validation tool from Github to a folder on the target machine	git clone https://github.com/BaobabHealthTrust/dde2_migration_validator.git migration_validator/	A folder will be created with the name passed for the target folder which in this case is “migration_validator/”	
3.		Install couchDB	$> sudo apt-get install couchdb		
4.		Load DDE1 integration data in a MySQL server if it does not exist already	DDE 1 Master MySQL database		
5.		Load DDE1 site data on a MySQL server	DDE 1 proxy “site” MySQL database. e.g dde_likuni		
6.		Load OpenMRS application data on a MySQL server	e.g BART2 application.		
7.		Open a command-line terminal and navigate to the migration tool “code” folder	$> cd {MIGRATION FOLDER}/code		

8.		Make sure the environment has Ruby version 2.1.2 running			
9.		Install mysql-client	$> sudo apt-get install libmysqlclient18 libmysqlclient-dev		
10.		Install the gems required to run the tool	{MIGRATION FOLDER}/code$> bundle install --local		
11.		Create databases configuration file based on “database.yml.example”	cp databases.yml.example databases.yml		
12.		Change settings in “database.yml” to match your settings	vim database.yml		
	a.	Set DDE1 aggregate database details in the “npids_mysql_source” section as listed in the sample configuration. (**Note: *The database referred to in this section is the database created in STEP 3 above.***) This is the database with all valid distributed national patient identifiers at the time of migration. The expected default value for “table” in this section is “national_patient_identifiers” whereas the default value for “field” entry is “value”			
	b.	Set the database credentials for the source MySQL database where the	e.g. The source MySQL database could hold all target sites with individualised		

		individual DDE1 instances for site data are loaded in the “mysql” section	databases like “dde_area_18” for Area 18 HC DDE1 MySQL data instance and so on		
	c.	Set the credentials for the target destination CouchDB DDE2 databases in the “couchdb” section as well as the database names for the 2 main target databases for Npids and Person.			
	d.	Set the corresponding target DDE1 databases to migrate to DDE2 in the “target” section. Ideally, one database will be configured at a time, however, it’s also possible to have multiple databases chained back-to-back by separating with commas(“,”)			
	e.	For merging, define application OpenMRS databases using similar titles as defined in (d.) for each corresponding MySQL database instance			
13.		Change “main.rb” file to executable	$> chmod +x main.rb		
14.		Initialize the database defined in 10.c if it does not exist	{MIGRATION FOLDER}/code$> ./main.rb -i **OR** {MIGRATION FOLDER}/code$> ./main.rb --initialize-only	Target CouchDB databases created as well as views	
15.		Open a command-line terminal and navigate to the migration validator tool folder	$> cd {VALIDATOR FOLDER}/		

16.		Setup the gem file environment for the tool	$> bundle install --local		
17.		Setup validator tool configuration files			
	a.i.	Create a copy of config/couchdb.yml.example as config/couchdb.yml	{VALIDATOR FOLDER}$> cp config/couchdb.yml.example config/couchdb.yml		
	a.ii	Configure config/couchdb.yml settings to point to the target DDE2 server instance. Set the “suffix” as the first part of the CouchDB database and “prefix” as the second part of the database name. For example, for a database name “dde_migration”, “suffix” is “dde” and “prefix” is “migration”			
	b.i	Create a copy of config/secrets.yml.example as config/secrets.yml **OR**	{VALIDATOR FOLDER}$> cp config/secrets.yml.example config/secrets.yml		
	b.ii	Generate a secret for secrets.yml if you want to run the application in production	{VALIDATOR FOLDER}$> rake secret		
	c.i	Create a copy of config/database.yml.example as config/database.yml	{VALIDATOR FOLDER}$> cp config/database.yml.example config/database.yml		
	c.ii	Configure config/database.yml			

		to point to the source MySQL DDE1 database for the target site			
	d.i	Create a copy of config/site_config.yml.example as config/site_config.yml	{VALIDATOR FOLDER}$> cp config/site_config.yml.example config/site_config.yml		
	d.ii	Configure the “site_code” to match the code for the target site for migration			
18.		Replicate current version of DDE2 master databases to the target Proxy databases	$> curl -H "Content-Type: application/json" -X POST -d '{"target":"http://TARGET_PROXY_IP_ADDRESS:PORT/PROXY_DB","source":"http://MASTER_IP_ADDRESS:PORT/MASTER_DB", "create_target": true}' "http://REPLICATION_HOST_IP:PORT/_replicate"		
19.		Run the validation tool as a normal Rails application on port of choice	{VALIDATOR FOLDER}$> rails s -p {PORT}		
20.		Run pre-migration report by selecting the Pre-Migration report path in a browser	http://{SERVER}:{PORT}/people/premigration		
21.		Run migration task (**Warning**: *This is usually a long task*)	{MIGRATION FOLDER}/code$> ./main.rb -m **OR** {MIGRATION FOLDER}/code$> ./main.rb --migrate-only	Data migrated from DDE1 to DDE2 and report generated	

22.		Run post-migration analysis and generate report by selecting the Post-Migration report path in a browser	http://{SERVER}:{PORT}/people/postmigration		
23.		Run pre-merge report	{MIGRATION FOLDER}/code$> ./main.rb -p **OR** {MIGRATION FOLDER}/code$> ./main.rb --pre-merge-report-only	Report on expected merge impact. No changes are made to the data source and destination	
24.		Run merge task	{MIGRATION FOLDER}/code$> ./main.rb -o **OR** {MIGRATION FOLDER}/code$> ./main.rb --merge-only	Data from OpenMRS application merged with DDE2 data and a report generated	