## Open Immunization Software

# Immunization Data Quality Assurance

VERSION: 1.0 REVISION DATE: June 16, 2011

| Approver Name | Title | Signature | Date |
|---|---|---|---|
| | | | |
| | | | |
| | | | |

# Contents

# Section 1. Overview

## 1.1 Purpose

Data Quality is just one step in a larger process of accepting, matching and importing data. The goal of Data Quality is to standardize and automate data quality functions in order to improve overall through put while increasing data quality.

## 1.2 Message Validation

Every message received by the IIS must be verified to meet minimum registry standards. This process is called Message Validation. The table below shows typically tasks performed in validation and examples of problems that can be screened in this process:

| **Verify Message:** | **Example of Potential Problem** |
|---|---|
| Is in HL7 format. | Message received is comma separated value format (CSV). |
| Has all fields required by HL7. | The patient first name field is empty. |
| Has all fields required by IIS. | The vaccination location facility id is not indicated. |
| Is internally consistent. | Vaccination is administered before patient's date of birth. |

Message Validation is well handled by integration engines such as Rhapsody. They can verify and ensure that the minimum standards are maintained for every message received.

## 1.3 Data Quality

Data Quality takes validation to the next step. Many issues that affect data quality cannot be seen in just one message but instead become clear when looking at a cohort of submitted records. The aim of Data Quality is to spot trends and larger patterns that point to areas that need attention. In addition Data Quality has more information about specific submitter practices and tailors the analysis appropriately. Here is a table showing the different areas data quality looks:

| **Review Message Batch:** | **Example of Potential Problem** |
|---|---|
| Ensure that all required but may be empty fields are sent as often as expected for a particular submitter. | One submitter never sends patient phone number although they are known to record it in their EHR. |
| Ensure data being recorded reflects expected vaccination practice. | One submitter consistently reports historical OPV vaccinations but never reports historical IPV. OPV is not longer administered in the US, probably a coding issue and OPV is really IPV. |
| Ensure that serious problems with a batch of messages can prevent that batch from being submitted. | One submitter sends in over 1,000 messages but nearly all the messages fail because of unrecognized vaccination codes. The message should be blocked automatically and the problem solved before continuing. |

| | |
|---|---|
| Enforce data quality process by only allowing through batches that come from submitters approved for production. | One submitter is found to have serious data quality problems and has their account suspended. They can still submit data normally but it is not being processed into production until the problems are fixed. |

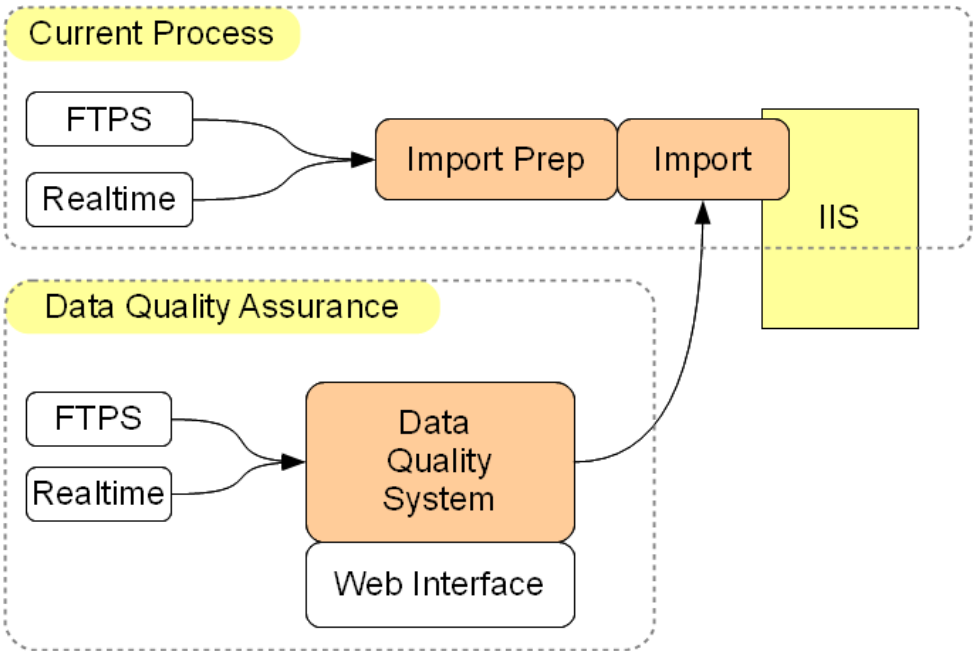
The data quality process has a different paradigm than message validation that allows it to:

- Handle data from different providers differently depending on data quality issues encountered in the past and what kind of data is expected.
- Review an entire batch for problems.
- Apply more vaccination specific rules and knowledge to a single message.
- Monitor and report back to submitters and IIS admin staff on the status of imports on a submitter level.

In addition the data quality process can be easily adapted to:

- Map submitter code values (such as provider ids and vaccinator ids) into IIS required values.
- Map non-standard format variations to IIS standards.

## 1.3 Overview Diagram

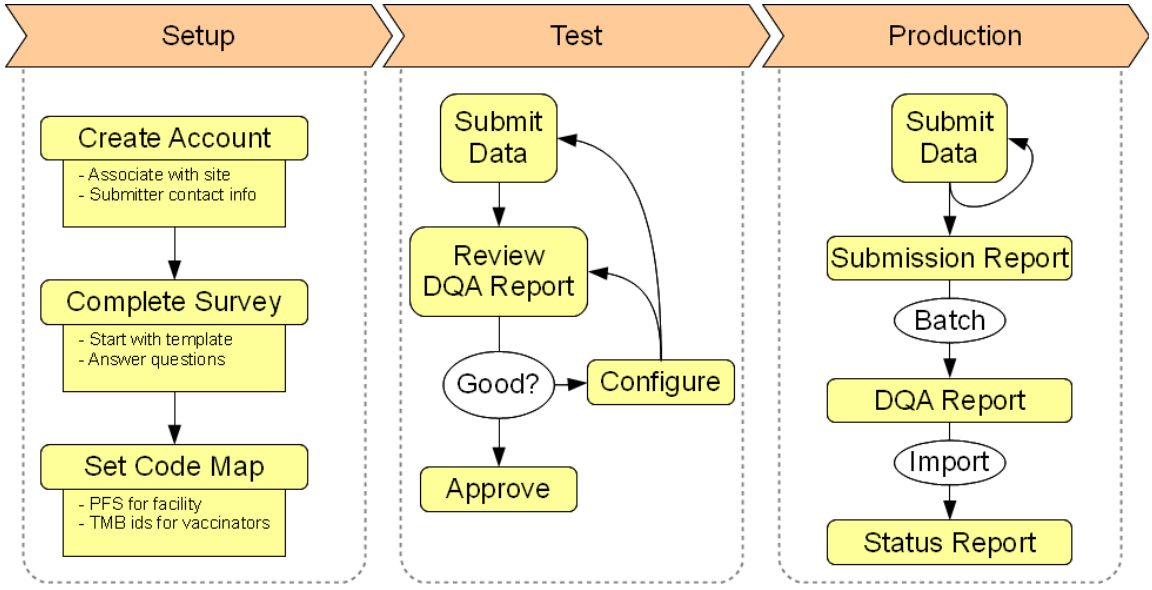



## 1.4 Current Process

The current process is manual and requires moving a set of batch files through several

# Section 2. DQA Setup and Configuration Flow

## 2.1 DQA Setup and Configuration Diagram




## 2.2 Setup Phase

The setup phase includes a process for collecting data about the providers interface and setting up the interface profile. At this point no data has yet been submitted and little is known about the interface. Ideally this process can be completed in large part by the provider but in most cases this will probably be completed by IIS admin staff in consultation with the provider.

## 2.3 Test Phase

Once the account is setup data should be submitted. At this point the data cannot be accepted until it has been thoroughly reviewed. From a software interface point this stage looks like the next stage, Live. The data should come from the provider's production system and should be sent on a regular basis, as if they were indeed reporting to the registry. This gives the IIS the opportunity to iron out all the interface configuration issues and work with the provider to fix any problems found.

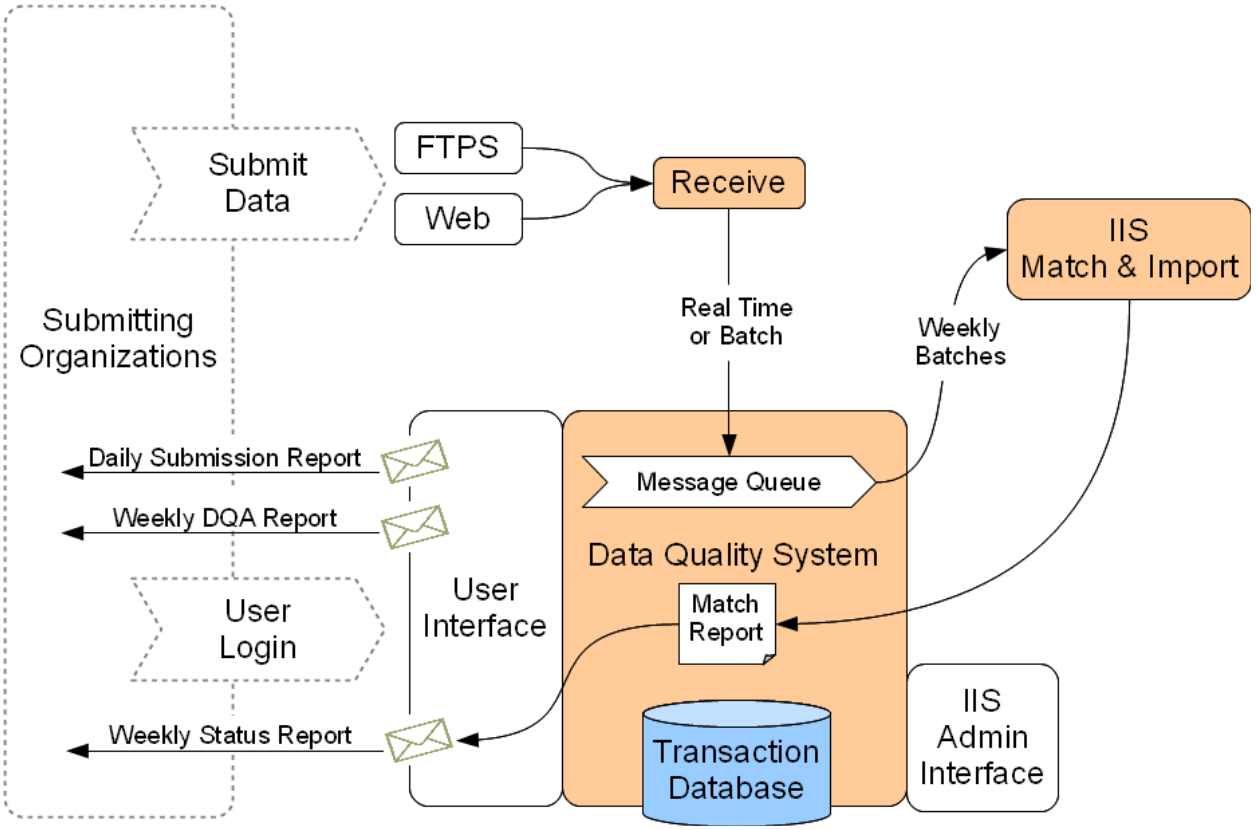
## 2.4 Production Phase

Once the interface is configured properly and all the data quality issues have been resolved, IIS admin staff can “flip a switch” and send the data to production. (IIS admin staff can decide exactly what data submitted previously is submitted.) At this point the process should move ahead

smoothly as a series of submissions, daily reports, data quality reports, and status reports. When new problems are seen the interface may taken back to the review stage.

# Section 3: System Integration

## 3.1 System Integration Diagram




## 3.2 Submitting Organization View

The submitting organization needs a unified interface that:

1. Easily accepts data for processing.
2. Gives clear reports on status or processing and if there are any errors and issues.
3. Provides clear direction on how to solve problems when they occur.

DQA would be design to handle the bulk of interaction with the provider. The data would be submitted to Rhapsody through FTP or Web upload. In the future a real-time interface to Rhapsody may be available. This data would then be sent to DQA for initial receiving.

The submitter would be updated on the submission the next day with details on what was submitted and its processing status. This email would indicate how many records were received, if there were any errors, and a note about when the data would actually be processed. No message specific data would be in the email but the provider could login to DQA to get more details in a secure manner.

The next week, when all the submitted data is batched together and prepared for IIS submission, the provider would get another email with a data quality report with detailed information about how the data is looking. Soon after this, perhaps the same day, the data would be submitted to the IIS for import.

After the IIS imports the data a status report would be sent back to DQA to be linked and saved with the submitted data. The submitter would be notified by email with a final status report and would know that the match report could now be downloaded from DQA.

This process would be built to keep submitters informed and involved. That way they can be the first to see and respond to data quality issues that are found. All the information that submitters see would also be available to IIS admin staff who can assist in the process.

## 3.3 Rhapsody Integration with DQA

Rhapsody would receive the data normally and then pass it directly to DQA without modification. This interface would be able to process batch or real-time and would send immediately to DQA.

DQA would release in batches data received as appropriate back to Rhapsody. The data would be guaranteed to meet the IIS standards and should pass through the message checks without a problem. The Rhapsody message checks developed in Stage 1 will remain in place to ensure that DQA is operating correctly.

## 3.4 DQA Process

Internally DQA will place all messages in a Message Queue. Not all messages placed in the queue are destined to be submitted to the IIS. The following situations will cause a message to be held in the queue and not batched to be sent to the IIS:

- The message failed validation checks and does not meet the IIS minimum standards.
- The message is from a submitter who has not been approved for production. The data will be included in data quality reports but no messages will be submitted.
- The message is from a submitter who has been approved for production, but the entire batch has too many data quality issues to pass minimum standards for the IIS, and is being held until reviewed by IIS admin staff.

DQA will be a web application with views for both submitters and the IIS import managing staff. This interface will allow for viewing reports, status and for stopping or allowing batches to proceed.

## 3.5 IIS Match Report

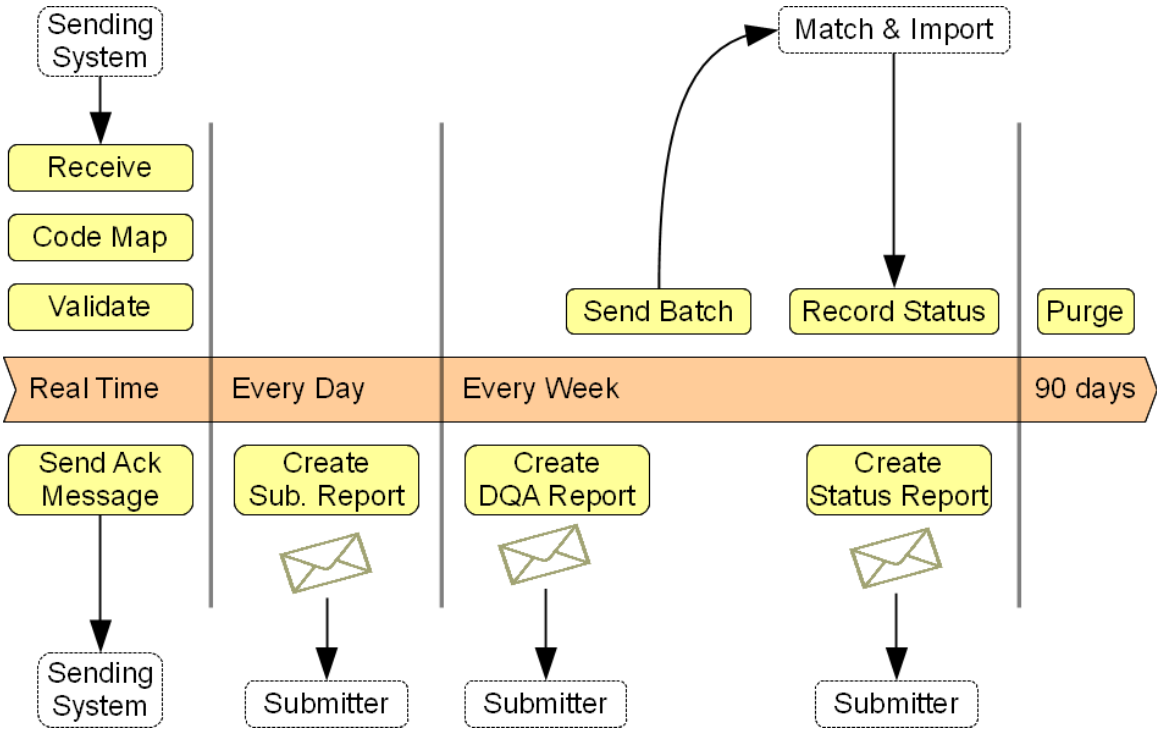
Some IIS can supply a report indicating the match success for each patient. DQA will be able to accept these match reports and record them for each message submitted. The submitter will be able to download this report or view details about how the matching went. DQA will be able to show trends on how the matching is going and highlight submitters who have poor match rates. The IIS can then use DQA to see where data quality problems are and point the way for the submitter to fix the problems.

# Section 4. DQA Record Flow

Providers submitting to the IIS need a real-time interface but some registries require or prefer batched data. In addition, the Data Quality process requires that data be put in batches in order to find batch level data quality problems. The solution is for DQA to have a real-time front end which collects the messages into weekly batches. These batches would form the basis for DQA reports and import batches. This section shows the how the records will flow through the message queue.

**Note**: Originally this was developed to support ImmTrac, which required weekly batches of data. This is not a requirement of DQA. DQA will be configurable so that data can be sent for import monthly, weekly, daily or immediately. This section assumes a weekly submission for all submitters.

## 4.1 Record Flow in Message Queue




## 4.2 Initial Inbound Process

Providers could send data whenever they wanted; once a week, once a day, or throughout the day. All messages would be replied to immediately and saved. Here is how the process would work:

- Data would be received in real-time from Rhapsody.
- Each file would be processed based on the settings made for each provider.
- Local values can be code mapped into IIS standard values. (This is needed for facility ids and vaccinator ids.)
- Each message would be validated for minimum registry requirements for format and content.
- An acknowledgment message would be created and sent back indicating if the message was accepted or not.
- The original message texts are saved but only those that were accepted are considered as candidates for being passed to Match & Import.

## 4.3 Daily Processing

Providers who submitted data will receive a short email report the next day indicating that data was received and a short report on its status. This information will also be available to view on the DQA application. This report:

- Will be sent to every provider who submitted data the previous day unless that provider account is configured not to do this.
- Report will indicate how many messages were sent in, how many were accepted, and other high level details.
- No patient data or other specific information will be included.
- Submitter can go to the DQA interface, login and view details.
- IIS admin staff can review daily reports, logs and details.
- IIS admin staff can receive summary email indicating all who submitted during previous day.

## 4.4 Weekly Processing

At the start of a new week all the data will be prepared for import. Every provider that submitted data will receive a Data Quality report that summarizes what was submitted and the level of data quality found. If the interface is live then the data will be queued for sending to Match & Import. The timing of this will depend on the status of the provider's data quality:

- Providers who have consistently demonstrated high data quality can be configured by the IIS to have their data imported immediately or very shortly after the Data Quality report is created.

- Providers who have had serious data quality problems can be automatically delayed up to 4 days to allow IIS admin staff time to review the Data Quality report.
- Batches that have serious data quality problems can be delayed or removed from the processing queue.

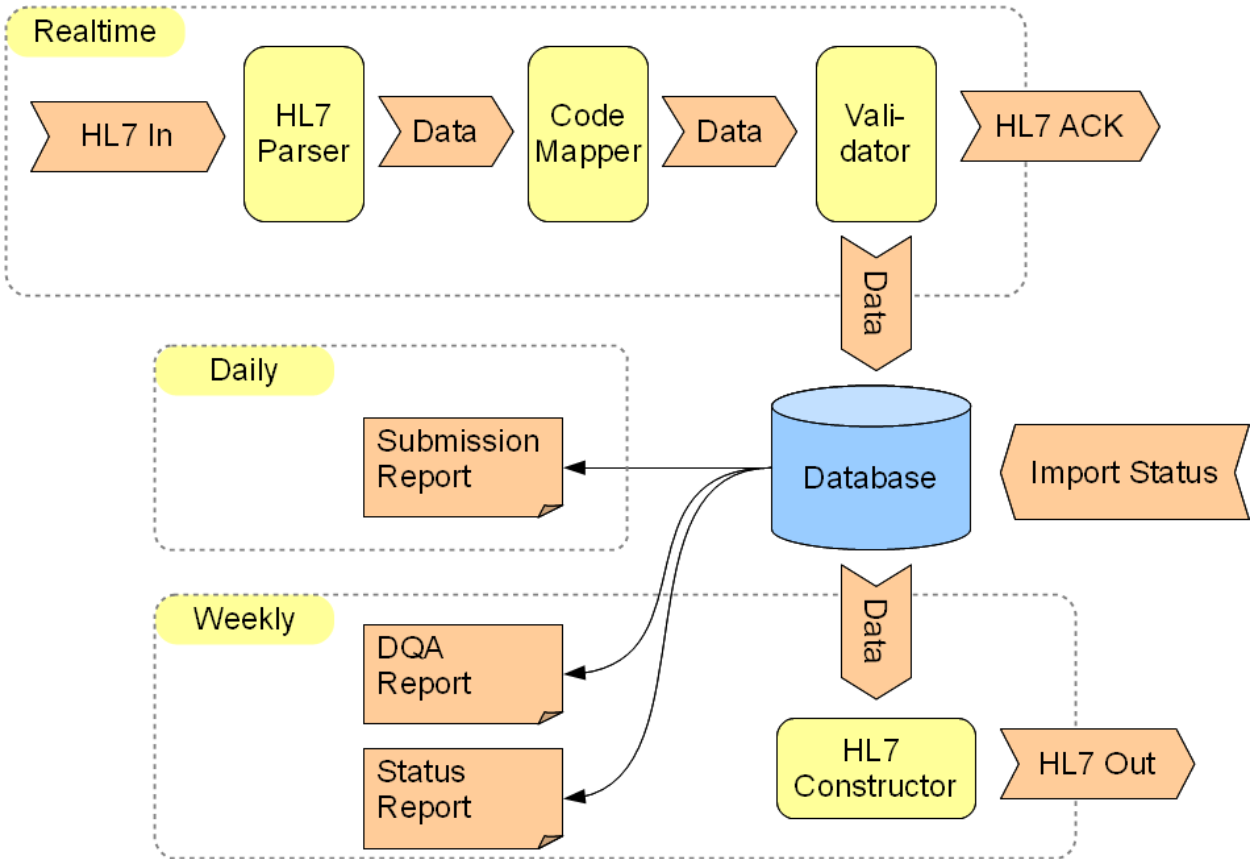
This process will ensure that all data received for Match & Import is thoroughly reviewed. The Match & Import process will return a file indicating any errors and how the matching went. This data will be feed back into DQA in order to complete the data quality picture. A final report with the status will be sent to the provider. The provider may login to DQA and download the match report.

## 4.5 Purge Processing

According to the IIS data retention policies, the received messages and specific data will be deleted at 90 days. Summaries of the weekly data quality will be kept so that trend reporting can be kept.

# Section 5. DQA Internal Process

## 5.1 Diagram




## 5.2 Database

The database will store HL7 messages, message data, provider configuration details, submitter profile information, interface status, and DQA summary details. Message details will be deleted at 90-days according to the IIS's record retention policy, but summary details will be retained in order to support trend reports.

## 5.3 Initial Inbound Processing

The realtime interface will receive and respond to messages individually and message validation based on the configuration for each provider. This will replicate some of the validation done in Rhapsody but can be fine tuned to meet what is expected from each provider. This will ensure a higher level of message compliance.

## 5.4 Daily Processing

Every day DQA would notify providers about the data that was sent the day before. This will provide a critical communication point back to the provider who may not be reviewing or seeing the acknowledgements returned.

## 5.5 Weekly Processing

A set of processes will run each week, starting with a DQA report, then a submission of live data for Match & Import, and then finished by storing the import status from the Match & Import process.

# Section 6. Development Process

The development of this system should be done in functional stages. Each improvement adding functionality to the system, but at each point the system continues to operate.

## 6.1 Project Phases

The DQA development will be seen has having three distinct phases:

- Phase 1: Create core architecture and basic functionality.
- Phase 2: Complete vision – build out core desired functionality
- Phase 3: Minor improvement to functionality, system wide performance tuning, long term maintenance.

Phase 1 is currently underway with one developer on the project. Phase 2 can include additional developers and the amount of work completed can be accelerated. Phase 3 will require consistent IT support but a software developer resource that is permanently tasked to the project at least part time.

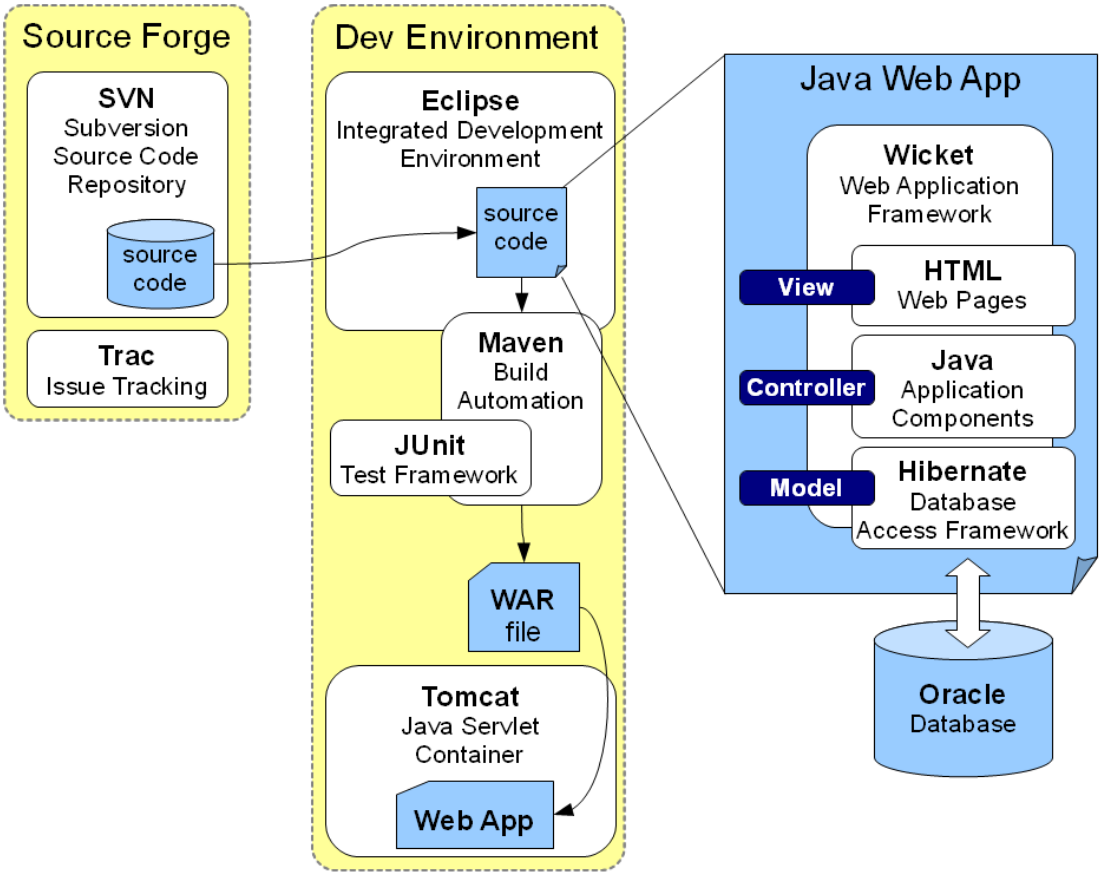
## 6.2 Phase 1: Build Core Application

The core architecture is currently being built. A summary of the tasks is given below:

| Functional Stage | Description |
|---|---|
| A: Establish Technical Architecture<br>**Completed** | The messages are accepted and saved in real-time in the database. DQA is not a part of the import flow, but rather receives a copy of the processed data. The data is not parsed, mapped or validated. No acknowledgment message is generated. |
| B: HL7 Message Parser for DQA<br>**Completed** | Message can be parsed and saved to database in its individual components. IIS admin staff can now start to run ad hoc queries to produce reports of data received. |
| C: Customized Configuration by Provider<br>**In Development** | Provider custom codes can be mapped to standard values. |
| D: Message Validation<br>**In Development** | Message can now be validated against a basic set of rules and an acknowledgment message can be generated. |
| E: Rebuild Message for Rhapsody<br>**July 5, 2011** | Messages can be batched and sent out in HL7 format for processing. Only providers configured as live are exported. This functional level will allow the DQA to be brought in as part of the import process and used as a batching mechanism and as a place to review data quality using ad hoc reports. |
| F: Generate DQA<br>**July 19, 2011** | Now a DQA report can be created. At this level it is not automatically sent but must be viewed on the DQA web application itself. This replaces the ad hoc querying to verify data quality. IIS admin staff will |

| | |
|---|---|
| | still have to run report and verify each week. |

**Technologies Used**




| | |
|---|---|
| Source Code Repository | • Subversion source code repository, final location has not yet been determined, perhaps SourceForge |
| Issue Tracking | • Trac on SourceForge is currently being used |
| Integrated Development Environment | • Eclipse is currently being used, but development is not locked into this, other dev environments would be equally acceptable. |
| Build Automation | • Maven provides an automated build process, simplifies development setup as well |
| Unit Test | • JUnit is a standard for Java applications and is well supported by Maven and other tools |
| Web Server | • Targeted for deployment on Tomcat but should work on most java servlet engines |
| Software Language | • Java has wide support and can be easily supported by most modern operating systems |

| | |
|---|---|
| Web Application Framework | • Wicket is an Apache open source framework that cleanly separates the MVC layers. This framework will separate the business logic from the view so it is easier to “dress up” the application without knowing Java. Will also dovetail with Hibernate. |
| Database Access | • Hibernate connects the java object world with the database relational mapping, speeding up development. More importantly it will help application support different types of databases with minimal work. |
| Database | • Oracle is the target for this application but it will also run on an alternate database for standalone use. Hibernate will support a simple migration to other databases. |

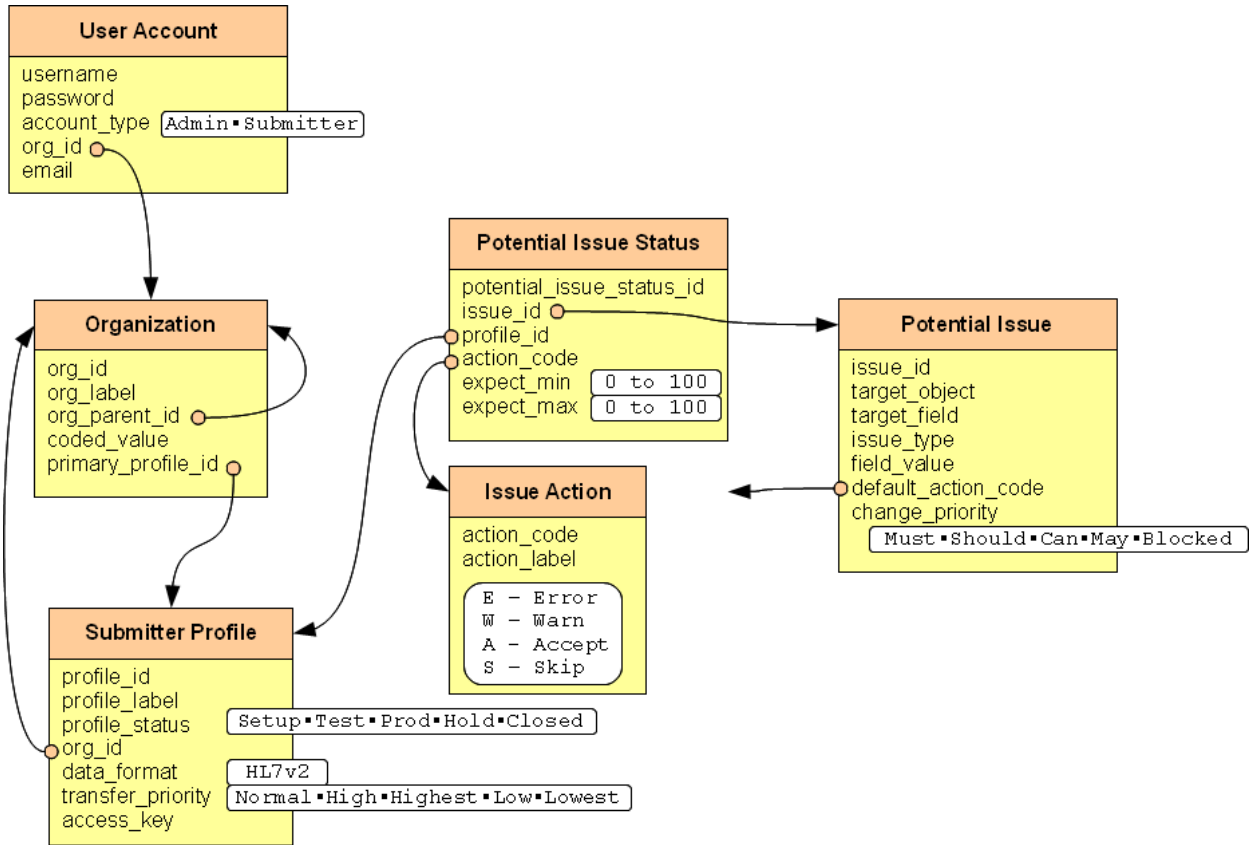
**Release Cycle**

Development is following a rapid application design pattern with 2 week functional releases. Every two weeks a new functional version of the application is completed and ready to be installed and tested. Functionality that cannot be completed in two weeks is broken into smaller functional pieces that are put together over several cycles. At each stage requirements can be modified as the application evolves.

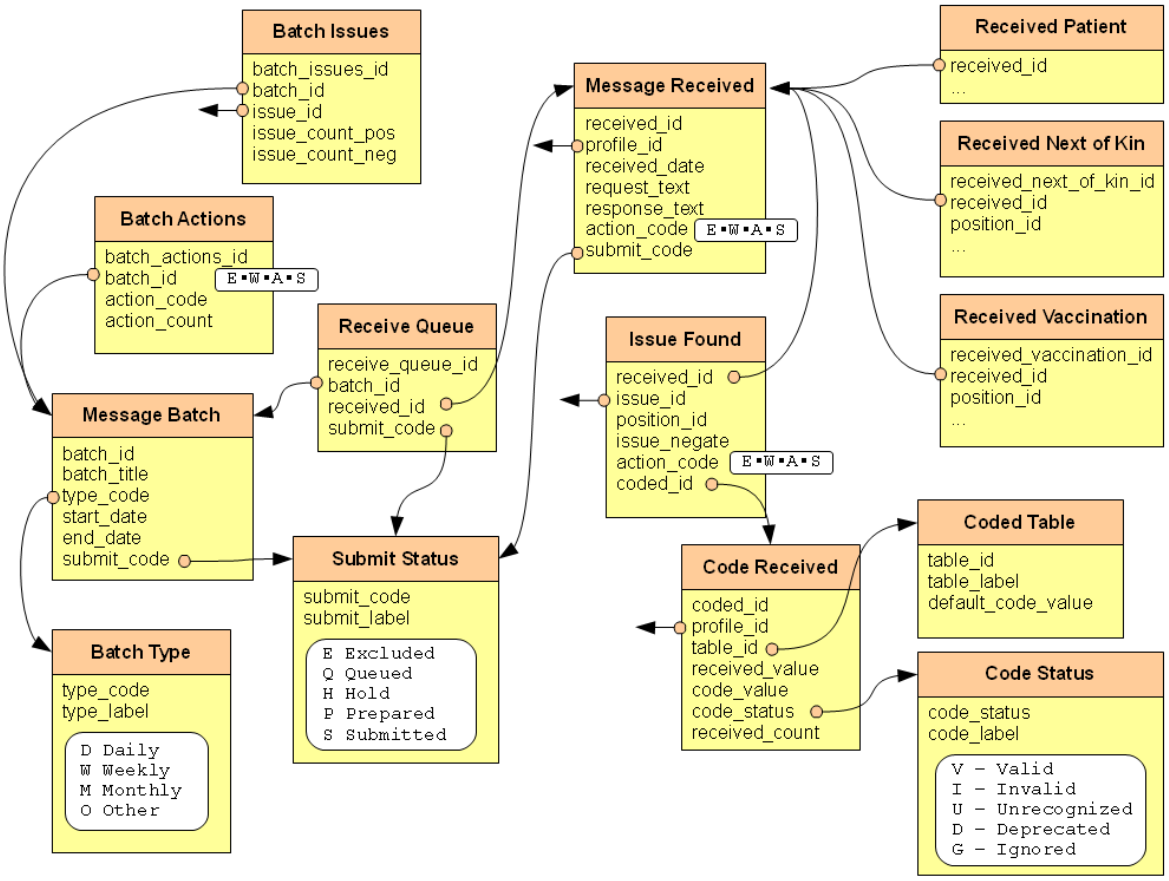
### Database Schema

The initial database schema has been designed, and is ready to support 90% of the expected functionality for Phase 1 and Phase 2. Here is the current database design:

ERD Page 1



ERD Page 2




## 6.3 Phase 2: Complete Vision

### User Interface

| Database Object | Associated UI Function |
|---|---|
| User Account | • Login<br>• Admin select organization<br>• Password reset<br>• Admin create/edit user account |
| Organization | • Add/edit organization<br>• Organization import or linking functionality |
| Submitter Profile | • Create/edit submitter profile<br>• Download/upload profile settings in XML |
| Potential Issue | • Edit default action and change priority of issues |
| Potential Issue Status | • Profile survey to ask expected issue levels |

| | |
|---|---|
| Message Received<br>Patient<br>Vaccination<br>Next of Kin | • View lists of messages received by profile by day<br>• View summary of messages received by profile<br>• View message details and data received |
| Issue Found | • View issues found for a given message |
| Code Received | • View list of codes received by profile by code table<br>• Edit code received status<br>• Reset received counts<br>• Add code received status one-by-one<br>• Upload codes received<br>• Download codes received<br>• View messages with issues associated with a code received |
| Code Table | • View list of code tables<br>• Edit default value<br>• View default code values |
| Message Batch | • View a list of batches for a profile for a particular status<br>• View list of messages received in a batch<br>• Change message batch status |
| Batch Issues<br>Batch Actions | • View list of batch summaries for profile |
| Vaccine Product | • Edit/add vaccine products<br>• Download/upload XML formatted values |
| Vaccine CPT | • Edit/update CPT codes<br>• Download/upload XML formatted values |
| Vaccine CVX | • Edit/update CVX codes<br>• Download/upload XML formatted values |
| Vaccine MVX | • Edit/update MVX codes<br>• Download/upload XML formatted values |
| Application Settings | • Edit/update application settings |

**Functionality Changes**

| | |
|---|---|
| Message Acceptance | • Create standard Web Service interface (as defined by CDC)<br>• Create other customer incoming interfaces as needed |
| Message Sending | • Create custom outgoing interfaces as needed<br>• Automatic message holding for batches that exceed threshold for warnings/errors |
| Emails | • Daily submission email summaries<br>• Weekly data quality summaries |

| | |
|---|---|
| | • Status reports when appropriate |
| Purge | • Automatic purging of personal identifiable information |
| | • |

**ImmTrac Specific Improvements Not Currently Planned**

| | |
|---|---|
| ImmTrac Match Status | • Interface for accepting match reports and saving them on the original messages sent in<br>• Interface to allow providers to view and download match reports<br>• Status email to be sent when match report is available<br>• Report summarizing how well a submitted data is matching |
| Submitter management | • Better integration with current process for tracking submitters |

**Possible Improvements for NMSIIS**

| | |
|---|---|
| HL7 Message | • Support specific NM requirements (such as insurance segment) |
| Submission Process | • Integrate with ideal submission process |

## 6.4 Phase 2 Estimated Level of Effort

| | |
|---|---|
| Phase 1 Work Completed so far | 114 hours – ImmTrac development project, work completed from April – June 16, 2011. |
| Phase 1 Work Estimate for remaining development | 180 hours – ImmTrac development project, all work under this contract must be completed by July 31, 2011. Final system must be functional and integrate into import process. |
| Phase 2 Development | Estimated 300 hours – Not currently scheduled, assumes build out of at least 90% of the complete vision detailed in this document. Partial development is possible, for core development assume that each bullet point is 7 hours of work. Some items are longer and some are shorter but this is a good rule of thumb. |
| Phase 2 Integration | Estimated 100 hours for developer – Some time will need to be given to address specific issues of integration when integrating with the next registry after ImmTrac in Phase 1. Once DQA has been integrated with several registries this integration time should be quite reduced. |

| | |
|---|---|
| Phase 2 Documentation | Estimated 100 hours – Not currently scheduled, the application needs to be documented and user guides written. |
| Phase 2 Software Quality Assurance | Estimated 100 hours – Not currently scheduled, software needs to be tested to assure it is working correctly. This task could be taken on by state health department |

.

# Section 7. Revision History

| Version | Date | Name | Description |
|---|---|---|---|
| 1.0 | 06/16/2011 | Nathan Bunker | Created document |
| | | | |
| | | | |
| | | | |