U.S. Department of Health and Human Services
Assistant Secretary for Planning and Evaluation
Office of Disability, Aging and Long-Term Care Policy

Development of a National Adult Protective Services Data System: NAMRS Pilot Final Report

Volume 2: System Documentation

September 2015

Office of the Assistant Secretary for Planning and Evaluation

The Office of the Assistant Secretary for Planning and Evaluation (ASPE) is the principal advisor to the Secretary of the Department of Health and Human Services (HHS) on policy development issues, and is responsible for major activities in the areas of legislative and budget development, strategic planning, policy research and evaluation, and economic analysis.

ASPE develops or reviews issues from the viewpoint of the Secretary, providing a perspective that is broader in scope than the specific focus of the various operating agencies. ASPE also works closely with the HHS operating agencies. It assists these agencies in developing policies, and planning policy research, evaluation and data collection within broad HHS and administration initiatives. ASPE often serves a coordinating role for crosscutting policy and administrative activities.

ASPE plans and conducts evaluations and research--both in-house and through support of projects by external researchers--of current and proposed programs and topics of particular interest to the Secretary, the Administration and the Congress.

Office of Disability, Aging and Long-Term Care Policy

The Office of Disability, Aging and Long-Term Care Policy (DALTCP), within ASPE, is responsible for the development, coordination, analysis, research and evaluation of HHS policies and programs which support the independence, health and long-term care of persons with disabilities--children, working aging adults, and older persons. DALTCP is also responsible for policy coordination and research to promote the economic and social well-being of the elderly.

In particular, DALTCP addresses policies concerning: nursing home and community-based services, informal caregiving, the integration of acute and long-term care, Medicare post-acute services and home care, managed care for people with disabilities, long-term rehabilitation services, children's disability, and linkages between employment and health policies. These activities are carried out through policy planning, policy and program analysis, regulatory reviews, formulation of legislative proposals, policy research, evaluation and data planning.

This report was prepared under contract #HHSP23320095656WC between HHS's ASPE/DALTCP and Walter R. McDonald and Associates. For additional information about this subject, you can visit the DALTCP home page at https://aspe.hhs.gov/office-disability-aging-and-long-term-care-policy-daltcp or contact the ASPE Project Officer, Helen Lamont, at HHS/ASPE/DALTCP, Room 424E, H.H. Humphrey Building, 200 Independence Avenue, S.W., Washington, D.C. 20201. Her e-mail address is: Helen.Lamont@hhs.gov.

NAMRS PILOT FINAL REPORT
Volume 2: System Documentation

Y. Yuan
S. Leelaram
S. Dahbour
M. Greene
A. Acker
E. Swartz

WRMA, Inc.

September 25, 2015

Prepared for
Office of Disability, Aging and Long-Term Care Policy
Office of the Assistant Secretary for Planning and Evaluation
U.S. Department of Health and Human Services
Contract # HHSP23320095656WC

TABLE OF CONTENTS

LIST OF FIGURES

ACRONYMS

The following acronyms are mentioned in this report and/or appendices.

ACL	Access Control List
App	Application
API	Application Programming Interfact
APS	Adult Protective Services
CIDR	Classless Inter-Domain Routing
DBO	Database Object
ELMAH	Error Logging Modules and Handlers
ERD	Entity Relationship Diagram
HTTP	Hypertext Transfer Protocol
HTTPS	HTTP Secure
ID	Identifier
IP	Internet Protocol
LINQ	Language-Integrated Query
NAMRS	National Adult Maltreatment Reporting System
PDF	Portable Document Format
SHA	Secure Hash Algorithm
SMTP	Simple Mail Transfer Protocol
SQL	Structured Query Language
SSL	Security Sockets Layer
TLS	Transport Layer Security
URL	Uniform Resource Locator
VHD	Virtual Hard Disk
VM	Virtual Machine
XML	Extensible Markup Language
XSD	XML Schema Definition

1. INTRODUCTION

The National Adult Maltreatment Reporting System (NAMRS) Pilot was designed and developed to test the capacity to collect administrative data from state adult protective services (APS) agencies, which in turn could be transformed into information that would provide knowledge to the field on the extent and underlying features of abuse, neglect, mistreatment, and exploitation of vulnerable adults. The NAMRS Pilot design was founded upon several technical development concepts and strategies. These strategies provided insight into the key abstractions and mechanisms used in the systems architecture. The NAMRS Pilot was a modern data system utilizing state-of-the-art concepts and technologies. The major concepts used strongly support data quality, data integrity, data security, ease of operation, and effective access to the data.

Purpose of the Document

The purpose of this document is to provide technical details for the NAMRS Pilot that will serve as a guide for the operation and maintenance of the system. This document addresses the details of how aggregate and case-level administrative data were submitted, validated, and stored in the NAMRS Pilot system. The guide is focused on helping the system architects, developers, and designers understand the technical details and concepts of the NAMRS Pilot.

Organization of the Document

This document describes the main components of the NAMRS Pilot. Each component will be discussed in detail and the discussion will be framed in four parts:

- **Function**--What was the function of the component and how was it implemented?
- **Access**--Which users/components could access the component, and which users/components did the component access?
- **Security**--What security measures were implemented by the component?
- **Configuration**--Where appropriate, there is a section that explains how configuration data was stored.

2. SYSTEM ARCHITECTURE

The NAMRS Pilot was a multi-tiered cloud application consisting of the user interface layer, business services layer, and the data layer. The user interface layer provided the web pages with control elements to access the various functionalities. The business rules and data processing were carried out in the business services layer, with each service implementing a particular function. Data were stored in the data layer using an advanced database management system. Figure 2.1 illustrates the NAMRS Pilot architecture on the cloud platform.

The NAMRS Pilot consisted of four main system components: (1) a website that users interacted with to upload data and obtain results; (2) an application called the "Loader" for validating the Case Component data files in Extensible Markup Language (XML) format and loading them into the database; (3) a database that stored all the data for the system; and (4) a data warehouse that stored all data to be accessed by analytical users. Additional cloud components that supported the operation of the main system components were the shared storage, email client, and the temporary cache.

The NAMRS Pilot was hosted in the Microsoft Azure Cloud service. It took advantage of the services that are provided in the cloud:


Azure Web Apps--This is a managed, secure, scalable, highly-available, load-balanced, and geographically load-balanced web application hosting service. It removes the need to implement a web server or virtual machine (VM).


Azure Redis Cache--This is a managed implementation of the Redis Cache service. This provides secure, scalable, caching inside the Azure Cloud.


Azure Virtual Machine--This is a scalable, highly-available, secure, virtual server located inside the Azure Cloud. This server can be loaded with any operating system, software, and virtual networking/hardware resources needed.

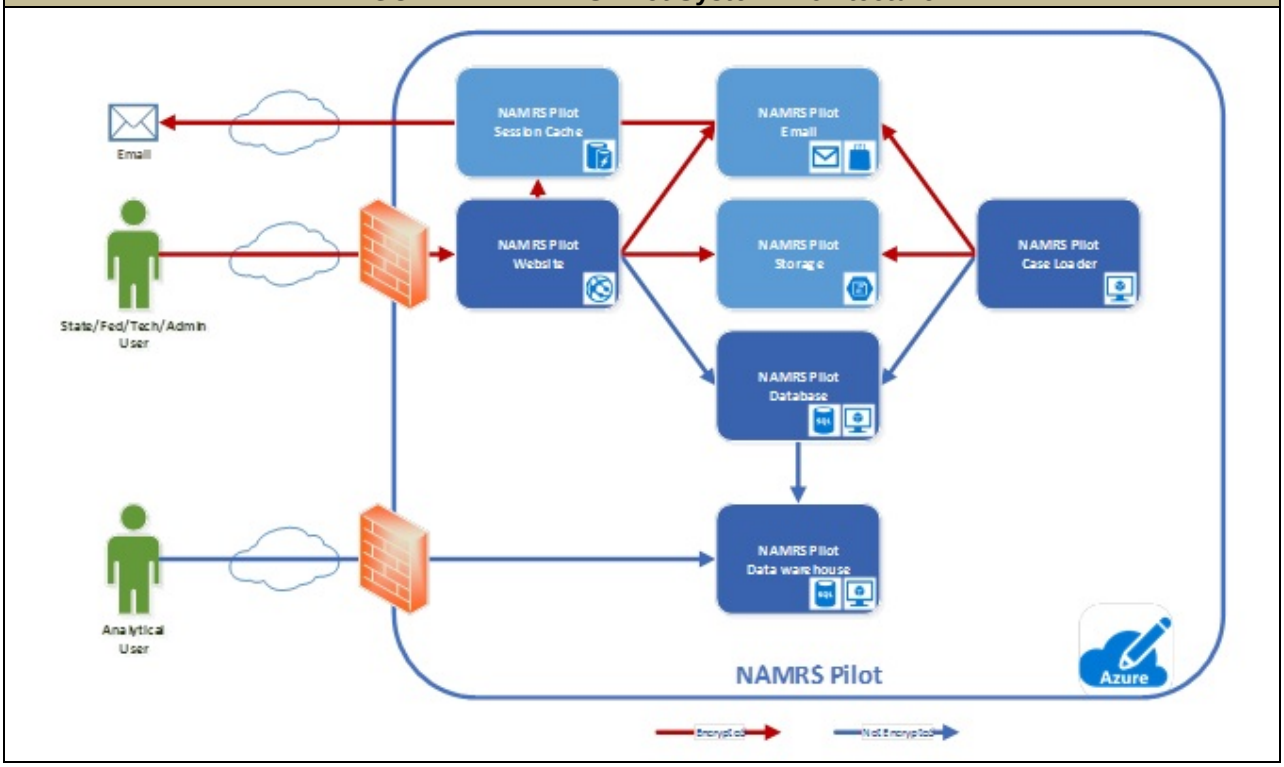

Azure Blob Storage--This is a scalable, highly-available, geographically redundant data storage service for files inside the Azure Cloud.


Azure Service Marketplace--This is a marketplace for services from 3rd party vendors that are available inside the Azure Cloud.

There are many other services available inside the Microsoft Azure Cloud, but the ones above are the set that were used for the NAMRS Pilot.

The data centers for the Microsoft Azure Cloud are extremely secure, and they many different compliance certifications. More about their compliance can be found here: http://azure.microsoft.com/en-us/support/trust-center/compliance/.

FIGURE 2.1. NAMRS Pilot System Architecture



3. NAMRS PILOT WEBSITE

The NAMRS Pilot Website was the main web interface that users interacted with. This website allowed state users, federal users, technical users, and administrators to log in, view/edit data, upload XML files, download reports, etc.

Function

There were several functional types of pages on the website:

- **Administrative**--These pages allowed a technical user or administrative user to manage (i.e., create/update/disable) resources, announcements, and users.

- **Account**--These pages allowed any user to change their own password or log off.

- **Component Data**--These pages allowed state users to enter/upload, change, and view their Agency, Case, and Key Indicators data. Federal users could view these pages for any state (although they could not change any data). Technical users and administrators could update data for any state and change the workflow status (i.e., accept/reject data).

- **Informational**--These pages allowed any user to view information about NAMRS.

The website was built as a Microsoft ASP.Net MVC Web Application. The site was written in C# and used Entity Framework and LINQ for all database access. The site used HTML5, CSS3, and JavaScript, as well as many open-source components and frameworks, including jQuery, jQuery UI, Bootstrap, ELMAH, Unity, iTextSharp, and other components.

The website was hosted in the Microsoft Azure Cloud as a “Web App.” A Web App (previously called an Azure Website) is a scalable, highly-available website. They provide a directory to upload the website into. They manage the all the underlying servers, security, server updates, networking, etc. Like all cloud services, if underlying hardware fails, the site is immediately moved and will become available immediately. Because of the redundant nature of the Web App service, there is no downtime for hardware upgrades and patches. If the site becomes slow because of traffic, it can be scaled to multiple server instances, and it can even spread those instances around the country and geographical load balancing will occur. The site can also be automatically scaled out based on traffic and scaled back in as traffic goes down (after hours) to keep costs down.

Access

The site was accessed directly by users, like any other website. The user logged in to access any pages--there were no pages accessible without logging in, other than the login page itself and the NAMRS XML Schema Definition (XSD) file (the XSD file that needed to be accessed by users submitting Case Component data). The reason that the XSD file was accessible was that many (most) XML validators do not have a way of logging into a website before they retrieve an XSD file. Therefore, the general paradigm on the web is to make XSD files available without login through HTTP. There was no sensitive data in the XSD file.

The site also accessed several other components in the Azure Cloud (each component is covered in more detail later):

- **Session Data**--There were just a few bytes of session data used across the site. An example of this data is that when a state user logged in, their assigned state was stored into session data for convenience. Session data was generally used for performance and convenience. The NAMRS Pilot used very little session data. Session data was stored in the Azure Redis Cache service and was very fast and very secure.

- **Website Data**--All data for the NAMRS Pilot Website, including the state's Agency, Key Indicators, and Case Components, announcements and resources that were displayed on the website, login data, and other data was all stored in the NAMRS Pilot Database. This database used Microsoft SQL Server running on an Azure VM. (We looked at using SQL Azure, which (like Web Apps) was a managed SQL Service, but the space required and the performance levels would have been prohibitive for our purposes.)

- **Email**--The NAMRS Pilot Website sent emails to state users and technical users as the status of state data was updated. For example, when a state user submitted their Agency Component data, an email was sent to a technical user to inform them that they needed to inspect this data. When the technical user approves/rejects the data, an email was sent to the state users to let them know the updated status.

  Obviously, sending email requires an SMTP server, however Azure does not have an SMTP service available. However, Microsoft has contracted with a company called SendGrid to supply SMTP service through the Azure Marketplace. SendGrid offers an SMTP service (the service is also called SendGrid) to Azure customers for free. The service can send up to 25,000 emails a month for free, and additional credits can be purchased very inexpensively.

Security

There were multiple security measures in the website, as described below.

Login

As stated above, the entire site (other than the login page and XSD file) was protected by the requirement to login.

Encryption

All communications between the user's browser and the website were encrypted with SSL. (The lock was displayed in the user's browser.) The site was configured to immediately switch to HTTPS if the user attempted to access through HTTP, so the connection was always encrypted.

The only exception to this was the XSD file. Users could access the XSD file using HTTP without being switched to HTTPS--this was the only file on the site that could be accessed through HTTP. The reason for this is that many (most) XML Editors/Validators cannot handle HTTPS.

Hashed Passwords

The site did not store user passwords in the database. Instead, only password hashes were stored. The password hash algorithm used in Microsoft Identity 2 was SHA1--more specifically, it ran 1000 iterations of a salted SHA1 (where the salt becomes part of the actual hash). Therefore, even if someone gained access to the database, it would require tremendous work to figure out the passwords (if it was possible at all). When a user entered their password, the password was sent to the website, where it was hashed and compared to the hashed value stored in the database. Since the website used HTTPS for all communications, the actual password was encrypted while it is being transported.

Roles

Every user was assigned a role when their account was created by an administrator. There were four roles:

- **State User**--State users were also assigned to a state. State users could access data only for their state. There was no way for a state user to access data for a different state--there were checks in the site to block this and log it if a user attempted to do so.

- **Federal User**--Federal users could access data for all states. They could view data but could not change any data on the site (except their own password).

- **Technical User**--Technical users could access data for all states. They could view and change data for any state. They could also change the workflow status (i.e., accept/reject data components for each state). They could also manage (i.e., create/update/disable) resources and announcements.

- **Administrators**--Administrators could do everything that technical users could do. They were also able to manage (i.e., create/update/disable) users.

Firewall

The Web Apps service was firewalled from the Internet, and all physical and network security was handled by Azure.

Data Precautions

There was no personally identifiable information in the NAMRS Pilot, so there was no way to link any particular piece of data to an individual person. States always encrypted all their database IDs (using their preferred encryption methodology) that could potentially be tracked to an individual.

Configuration

Like all ASP.Net Apps, configuration was done through the web.config file, located in the root of the App. All the parameters were documented in the web.config file.

4. NAMRS PILOT DATABASE

The NAMRS Pilot Database stored all the data for the NAMRS Pilot.

Function

There were several functional types of data that were stored in the database:

- **NAMRS Component Data**--This included the Agency, Case, and Key Indicators Component datasets that were entered/uploaded by the states.
- **Announcements/Resources**--These were the announcements and resource files that were displayed in the NAMRS Pilot Website. These items were loaded into the database through the website by administrators and technical users.
- **Case Component XML Validation Rules**--These were the data validation rules that were used to validate the Case Component XML files that were uploaded by state users.
- **Error Logs**--These were "ELMAH logs" that tracked errors, exceptions, and security violation attempts in the website.
- **User Profiles**--This was contact and other information about NAMRS Pilot users.
- **Credentials and Roles**--This data included the credentials and roles for all users. This was standard Microsoft Identity 2 tables.

Schemas

The six functional types above comprised the 95 different tables that made up the database. Those tables were broken up into four schemas:

- **DBO**--The DBO schema contained the identity and ELMAH tables.
- **Core**--The core schema contained all the tables for data that was not submitted by states, such as the announcements, resources, user profiles, and validation rules.
- **Lookup**--The lookup schema contained all the lookup tables.
- **NAMRS**--The NAMRS schema contained all the tables for the Agency, Case, and Key Indicators Component datasets.

Appendix A provides ERD for the physical database structure. The ERD is provided as a PDF and can be modified using Adobe Acrobat software. Appendix B lists the definition of all the database tables and data elements.

Views

There were nine different views in the database that were used only for data extract functionality (which was a manual process run by a system administrator). These views were not accessed by the website.

Stored Procedures

There were only five stored procedures in the database. Three of them were used by the ELMAH component in the site. One was used to build the data extracts and one was used to load the data warehouse. The last two were both manual processes run by a system administrator.

Custom Functions

There were only two user-defined (scalar-valued) functions in the database. Both were used in the data extract views.

The database ran on Microsoft SQL Server 2014 (Web Edition), installed on a VM in the Microsoft Azure Cloud. Since this was a VM, normal maintenance had to be performed. In addition to manual service pack updates, there were two scheduled maintenance plans:

- **Nightly**--The nightly maintenance plan ran at 1 a.m. every night and backed up all databases on the server (to Azure Storage), checked database integrity, reorganized indexes, and updated statistics.

- **Weekly**--The weekly maintenance plan ran at 3 a.m. on Saturdays and rebuilt all indexes on all databases on the server.

Access

The NAMRS Pilot Database was accessed by the following:

- **NAMRS Pilot Website**--The website accessed the database so that it could store and retrieve data for the website.

- **NAMRS Pilot Case Loader**--The case loader application accessed the database so that it could load data from Case Component XML files.

The NAMRS Pilot Database accessed the following:

- **NAMRS Pilot Storage**--The nightly backup process created a full database backup each night. The backup file was stored in NAMRS Pilot Storage.

- **NAMRS Pilot Data Warehouse**--Data from the database was stored into the data warehouse. This was a manual process performed by a system administrator.

Security

There were two levels of security for the NAMRS Pilot Database.

Network Security

An Azure VM basically ran behind a firewall. One must expose “endpoints” which are essentially TCP ports. Each endpoint could be protected with an Access Control List (ACL) which allows one to limit access to the endpoint to one or more IP addresses (or address ranges that are specified as classless inter-domain routing [CIDR] addresses).

The VM that ran this SQL Server only has the following endpoints exposed:

- **Powershell**--This allowed powershell scripting against the VM.

- **Remote Desktop**--This allowed a system administrator to get remote access to the server. Two-factor authentication was enabled for all remote desktop access.

- **SQL Server**--This allowed the SQL to be queried and managed.

Each of the endpoints was open only to a single IP address--the WRMA office. These servers could not be accessed from any other IP address on the Internet other than the WRMA office. Also, the SQL Server endpoint was open to IP addresses for Azure Web Apps. This was required so that the NAMRS Pilot Website (which is an Azure Web App) could access the database. Since a Web App can change IP addresses without notice (as it moves around in the cloud), the database had to be open to the full range of IP addresses that could be used by Web Apps.

Database Security

Each person/component that accessed the database had database credentials. The only accounts available in SQL Server were for the system administrator and for the website to access the database.

5. NAMRS PILOT CASE LOADER

The NAMRS Pilot Case Loader was an application that does processing on the Case Component XML files that were uploaded by state users. These files had to be validated against the NAMRS Case Component XSD file. State users should have validated the file against the XSD before they uploaded it to the NAMRS Pilot Website (although the file was validated as part of processing). If the file was very large the state user could zip the file and upload the zipped file.

Function

The Case Component Loader performed the functions on all files waiting to be processed:

- Downloaded the file from NAMRS Pilot Storage to the local temp space.
- Unzipped the file if the file was zipped.
- Validated the file against the XSD. If it was not valid, saved an error message for the user and stopped.
- Ran the data validation rules against the file (and performed actions on rule violations) and built a list of warning messages (which could be quite long) for the user. If the file was not valid (i.e., it had no investigations left at the end of validation), then added an error message for the user and stopped. After the file had been validated and any invalid date had been removed (the actions mentioned above), a new version of the file without any invalid tags/data was saved.
- Loaded the data from the valid case XML file produced by the previous step into the database. This was the longest part of the process.
- Calculated the basic counts that were displayed on the website.
- Built the PDF files for the summary counts and frequency counts.
- Saved the computed Key Indicators to the database.

The Case XML Loader was a “console application” that could be run from the command line by executing the following command:
Namrs.WebJob.CaseXmlLoader.exe -- it did not take any parameters.

The Case XML Loader was built as a Microsoft Console Application. The site was written in C# and used Entity Framework and LINQ for all database access. The site used several open-source components and frameworks, including DotNetZip, iTextSharp, log4net, RazorEngine.

This application ran on an Azure VM. (It ran on the same VM that hosted the NAMRS Pilot Database.) It was executed as a Scheduled Task on the VM. It ran once a minute every day. So on average, it started processing an XML file within 30 seconds of the state user uploading the file. The scheduler was set so that it only ran one instance of the loader at a time.

The validation rules applied at the time of data entry of Agency Component data, validation rules applied at the time of data entry of Key Indicator data, and the validation rules applied on the Case Component XML file along with the action taken when invalid data were found--were all included as Appendix F in Volume 1.

Access

Since the Case XML Loader was a console application, it was not accessed by any other process (although one could say that it was accessed by the Task Scheduler, which runs it every minute).

The Case XML Loader accessed the following:

- **NAMRS Pilot Database**--The loader queried the database to find which Case Component XML files were awaiting processing, to save the Case Component data from the XML files into the database, to save other statistics and computed counts into the database, etc.

- **NAMRS Pilot Storage**--The loader downloaded the Case XML file from NAMRS Pilot Storage into a local disk so that it could be easily accessed (and unzipped if necessary).

- **NAMRS Pilot Email**--The loader would send an email to the state user when it was finished processing the file.

Security

The NAMRS Pilot Case Component XML Loader was a command line application and did not accept any incoming connections or parameters. It ran on the same VM that hosted the NAMRS Pilot Database, so all the same security measures for that VM were in place for this process.

Configuration

The NAMRS Pilot Case XML Loader was configured using the NAMRS.WebJob.CaseXmlLoader.exe.config file, located in the root of the App. All parameters were documented in the config file. The configuration parameters were similar to the parameters for the website, although there were not as many.

6. NAMRS PILOT DATA WAREHOUSE

The NAMRS Pilot Data Warehouse provided a way for analysts to query the Case Component data collected from the states using statistical analysis tools, data visualization tools, or any other type of tool that can query a SQL database. The data warehouse contained only the Case Component data, and was connected to the Internet.

Function

There were several functional types of data that were stored in the data warehouse:

- **CaseDataSet**--stored the various states/periods for the case data.
- **Investigation**--stored the investigations.
- **Clients**--stored the client data.
- **Maltreatments**--stored the data about the maltreatments.
- **Perpetrators**--stored the data about the perpetrators.
- **Relationships**--stored the data about the client-perpetrator relationships.

Schemas

There was a single schema in the data warehouse:

- **Data Warehouse**--The data warehouse schema contained all the objects that could be queried in the data warehouse.

Views

There were no views that could be accessed in the data warehouse. (There was a single view that could not be accessed through the data warehouse schema--it could be used for testing, but would probably be deleted.)

Stored Procedures

There were no stored procedures that could be accessed in the data warehouse.

Custom Functions

There were no user-defined functions that could be accessed in the data warehouse.

The database ran on Microsoft SQL Server 2014 (Web Edition), which was installed on a VM in the Microsoft Azure Cloud. Since this was a VM, normal maintenance had to be performed. In addition to manual service pack updates, there were two scheduled maintenance plans:

- **Nightly**--The nightly maintenance plan ran at 1 a.m. every night and backed up all databases on the server (to Azure Storage), checked database integrity, reorganized indexes, and updated statistics.

- **Weekly**--The weekly maintenance plan ran at 3 a.m. on Saturdays and rebuildt all indexes on all databases on the server.

Appendix C provides the ERD for the physical database structure. The ERD is provided as a PDF and can be modified using Adobe Acrobat software. Appendix D lists the definition of all the data warehouse tables and data elements.

Access

The NAMRS Pilot Data Warehouse was accessed by the following:

- **NAMRS Pilot Database**--The database loaded data into the data warehouse.

- **Analysts**--Analysts connected directly to this database to do queries.

The NAMRS Pilot Data Warehouse did not access any other components.

Security

There were two levels of security for the NAMRS Pilot Database.

Network Security

An Azure VM basically ran behind a firewall. One must expose “endpoints” which are essentially TCP ports. Each endpoint could be protected with an ACL which lets one limit access to the endpoint to one or more IP addresses (or address ranges that are specified as CIDR addresses).

The VM that ran this SQL Server only had the following endpoints exposed:

- **Powershell**--This allowed powershell scripting against the VM.
- **Remote Desktop**--This allowed a system administrator to get remote access to the server.
- **SQL Server**--This allowed the SQL to be queried and managed.

The Powershell and Remote Desktop endpoints were open only to a single IP address--the WRMA office. These endpoints could not be accessed from any other IP address on the Internet other than the WRMA office. The SQL Server endpoint was wide open to the Internet. This was required so that the analysts could log in to query the database.

Database Security

Each person/component that accessed the database had database credentials. The only accounts available in SQL Server were for the system administrator and for the NAMRS Pilot database (so that it could load data). A role called data warehouse users was created in SQL server and in the database. This role had been denied access to most objects in the master database, denied access to all databases on the server (except the data warehouse) and it had been granted read-only access to the data warehouse schema.

New analysts could be given access to the data warehouse as needed. Analyst accounts were assigned to the data warehouse user role. This effectively allowed analysts to query the data in the data warehouse while denying access to any other database on the server.

7. NAMRS PILOT STORAGE

The NAMRS Pilot utilized shared storage space where multiple components could access a file. In the Microsoft Azure Cloud, the mechanism for doing this is called Azure Storage.

Function

There were three types of storage mechanisms provided in Azure:

1. **Azure Blob Storage**--provided a way to store files.

2. **Azure Table Storage**--provided a way to store (unstructured) table rows.

3. **Azure Queues**--provided a common message queue--a way to share messages.

Azure Storage is a very economical, infinitely scalable storage service. You only pay for what you use, and there is (practically) no limit to the amount of data that can be stored.

The NAMRS Pilot only used Azure Blob Storage for storing and sharing files. Azure Blob Storage uses the following hierarchy:

- **Storage Account**--There must be at least one Storage Accounts.

- **Container**--Each storage account must have at least one Container. A container is analogous to a folder (or directory) on a disk.

- **Files**--A Container can have zero or more files in it.

The NAMRS Pilot used a single storage account that had the following containers in it:

- **CaseFiles**--This container had all the case files. This included the actual XML/zip files that were uploaded by the state users through the website. Once the XML file was processed by the NAMRS Case XML Loader, the following files were added for each case XML file: (1) a text file that had all the error/warning messages produced for the XML file; (2) a second version of the XML file without any invalid data; (3) a PDF file containing the frequency counts which was available on the website; and (4) a PDF file containing the summary counts which was available on the website.

- **ResourceFiles**--This container had all the resource files that were uploaded by technical users. These files were available to users on the website.

- **DbBackup**--This container had all the database backups created by SQL Server maintenance plans.

- **VHDs**--This container had all the “virtual hard disk” files that were used by the Azure VMs.

Azure Storage is a service provided by the Azure Cloud. It is a “geographically redundant,” meaning that files are not only backed up, but backed up across different regions. This means that if an entire data center was to ever go offline (or possibly be destroyed), the files will remain intact. (For the NAMRS Pilot, the primary region was in the Eastern United States region, and the secondary region was in Western United States.)

Access

The following components accessed NAMRS Pilot Storage:

- **NAMRS Pilot Website**--The website stored Case Component XML files that were uploaded by state users, as well as resource files uploaded by technical/administrative users. It also read PDF and text files that were created by the Case Component XML Loader, as well as resource files.

- **NAMRS Pilot Case Component Loader**--The loader retrieved the Case XML files, and stored PDF and text files.

- **NAMRS Pilot Database** (not shown on diagram)--The database saved its backup files.

- **NAMRS Pilot Data Warehouse** (not shown on diagram)--The data warehouse saved its backup files.

The NAMRS Pilot Storage is a service provided by the Azure Cloud. It did not access any components in the NAMRS Pilot.

Security

Security for NAMRS Pilot Storage was handled by the Microsoft Azure Cloud.

Azure Blob Storage is an application programming interface (API) and it is available from the Internet. It is available by HTTPS. There are many positive aspects to

this because Azure Blob Storage can be used for many different purposes. The files were not available on the Internet in the NAMRS Pilot.

Access to Azure Blob Storage API is through an URL and a secret access key. To use the API, one needs the access key. Access keys for Azure Storage are almost 90 characters long, so it was highly unlikely that anyone could guess this key.

All access to the Azure Blob Storage API uses SSL encryption over HTTPS. So, data was always encrypted as it moved between the web server (and the Case XML Loader) and Azure Storage.

8. NAMRS PILOT EMAIL

The NAMRS Pilot sent email alerts on a regular basis to support the various workflow and system processes.

Function

There were several types of emails that were sent:

- **Emails to Technical Users**--These emails informed technical users that they needed to take action. (For example, that a technical user needed to review a data component that had been entered/uploaded by a state user.)

- **Emails to State Users**--These emails informed state users that the system was awaiting input from them. (For example, that a technical user had accepted a data component for their state.)

- **Emails to System Administrators**--These emails informed a system administrator of some action. (For example, which nightly backup has completed successfully.)

Email required an SMTP service. The Azure Cloud does not have an SMTP service. However, as described earlier, there is an SMTP service called SendGrid that is available in the Azure Cloud. The NAMRS Pilot used SendGrid as the NAMRS Pilot email component.

Access

The following components access NAMRS Pilot email:

- **NAMRS Pilot Website**--The website sent email to both state users and technical users.

- **NAMRS Pilot Case Loader**--The loader sent email to state users.

- **NAMRS Pilot Database** (not shown on diagram)--The database sent email every time it completed a maintenance cycle.

- **NAMRS Pilot Data Warehouse** (not shown on diagram)--The data warehouse sent email every time it completed a maintenance cycle.

The NAMRS Pilot Storage was a service--it did not access any components in the NAMRS Pilot.

Security

Like any SMTP server, the SMTP Server from SendGrid uses a host, port, username, and password for security. The username is quite long (almost 50 random characters) and the password is strong. SendGrid is used port 587, so TLS encryption is in place inside the Azure Cloud. This means that data was encrypted as it moved between the web server (and Case Component XML Loader) and the SendGrid server.

If the user's server supported TLS, then SendGrid would use encryption when sending the email to the receiving SMTP server. (And if the user had an email client that supported encryption, the email would encrypt as it was sent.)

9. NAMRS PILOT SESSION CACHE

Like most websites, the NAMRS Pilot used session data to store data between requests for a particular user during a login session. NAMRS used very little session data--only a couple of items such as the state ID and state name.

Function

There were various ways to store session data. NAMRS used Azure Redis Cache to store its session data. This is a best practice when using SQL Azure. The NAMRS Pilot had been migrated from SQL Azure onto a VM running SQL Server and still used Redis for maximum portability.

Redis is an industry standard, open-source caching service. The website used a standard Redis Session Provider that plugged into the ASP.Net Provider Framework. This meant that there were no code changes if the session provider was changed. It would be very simple to switch from Redis session state to SQL session state, to even in-memory session state. (And this had all been tested--it is purely configuration without any code changes.)

Access

The NAMRS Pilot Session Cache was accessed by the NAMRS Pilot Website.

The NAMRS Pilot Session Cache was a service--it did not access any components in the NAMRS Pilot.

Security

Redis security was controlled through a host, port, and secret access key. The access key was almost 50 random characters long and it was unlikely that someone would guess the key.

Data were always encrypted as it moved across the network inside the Azure Cloud between the web server and the Azure Redis service.

10. WORKFLOWS

The following workflow took place when a state submitted any of the data Components. The approval process required a technical user review the data for the Agency and Key Indicators Components and the validation results and reports for the Case Component.

In this flow, the status for the data component started at not submitted, then became one of the following based on the state and technical user's action:

- **In Process**--Once the state had submitted the data, the technical user reviewed the data to confirm if the data quality requirements were met by the data submission. During this review period the data status remained marked as in process.

- **Resubmit**--In this case, the technical user had reviewed the data (Agency and Key Indicators Components) or results/report (Case Component) and decided that the data were not valid. The technical user requested the state to resubmit and needed to contact the state and discuss the issues with them.

- **Approved**--In this case, the technical user found no errors during the review of the data component and approved the data.

Automatic email alerts were sent to appropriate state and technical users to communicate the data status changes. Figure 10.1 provides a schematic representation of the workflow and interaction between the NAMRS, state user, and the technical team user.

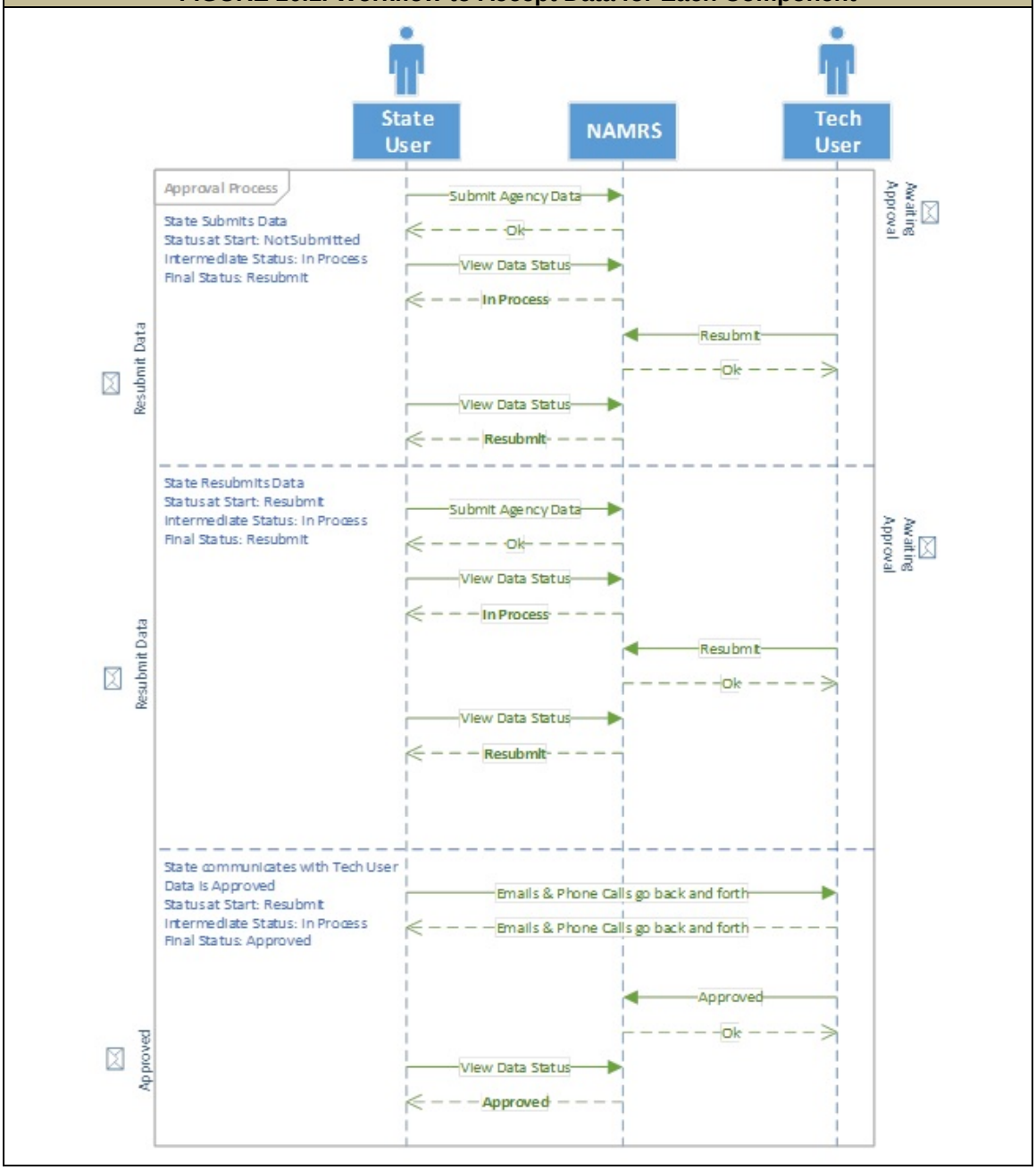
Agency Component Data Entry and Submission

State users filled a data entry form that contained fields for all the data points in the Agency Component. The user could fill out the form, save their data, and return later to work on it some more. Once they were satisfied with all the data, they could submit it. This page had basic type validation (i.e., if a field required numbers and the user entered letters in that field) and required field validation at the time of saving the data. If the user submitted their data, they may no longer edit their data unless data resubmission was requested.

A federal user could view data for the Agency Component for any state. A federal user would see the same page as the state user. However, the federal user could view the data for any state by toggling the list of states on the top of the page. All fields would be disabled, and there would be no way to save or submit data.

A technical user or administrator could submit or approve data for the Agency Component for any state. A technical or administrative user would see the same page as the state user. However, the user could view the data for any state by toggling the list of states on the top of the page. The user could edit/save data for any state. The user would see radio buttons for each status, with the current status selected. The user could change the data status and save. The appropriate emails were sent, depending on the new status. There would be no submit button for this user but data could be saved by changing the status to in process.

FIGURE 10.1. Workflow to Accept Data for Each Component



Key Indicators Component Data Entry and Submission

State users filled a data entry form that contained fields for all the data points in the Key Indicators Component data. The user could fill out the form, save their data, and return later to work on it more. Once they were satisfied with all the data, they could submit it. This page had basic type validation (i.e., if a field required numbers and the user entered letters in that field) and required field validation at the time of saving the data. If the user submitted their data they could no longer edit their data unless data resubmission was requested.

A federal user could view data for the Key Indicators Component for any state. A federal user would see the same page as the state user. However, the federal user could view the data for any state by toggling the list of states on the top of the page. All fields would be disabled, and there would be no way to save or submit data.

A technical user or administrator could submit or approve data for the Key Indicators Component for any state. A technical or administrator would see the same page as the state user. However, these user could view the data for any state by toggling the list of states on the top of the page. The user could edit/save data for any state. The user would see radio buttons for each status, with the current status selected. The user could change the data status and save. The appropriate emails were sent, depending on the new status. There would be no submit button for this user but data could be saved by changing the status to in process.

Case Component Data Entry and Submission

A state user generated the XML file. The data for the Case Component would be submitted in XML format. A state user would first export the data from the state information system into the required XML format. Following that, the state user validated for XML structure and data element characteristics requirements using the XSD file available for download on the NAMRS Pilot website. The state user could perform the validation using their favorite XML validator software. XmlValidator (Sourceforge) is an open-source XML validation software. XML Spy is a commercially available XML validator and editor. The XML file without any errors would be ready to be uploaded to NAMRS.

A state user could upload and submit data for the Case Component. A state user could upload an XML file containing the Case Component data on the file upload tab. When the file was uploaded, it was validated against an XSD by the NAMRS Pilot system to determine validity. If the XML was invalid (structure and field definition validation), validation error messages describing the issues were displayed and the entire XML file is considered invalid. The user could download the list of error

messages. The data status for the component remained as in process because NAMRS could not read the data in the file. The user has to upload a valid file to proceed further.

If the XML was valid, it proceeded to the next step which was the content validation. This was an asynchronous process where a number of validation rules across fields and records were applied and could take some time. The data status for the component changed to in validation while the file was being validated and a new file could not be uploaded during this time. When data validation was complete, the data status for the component changed to data valid when there were no errors and to data invalid when errors were identified.

If there were no errors the user could view the validation results on the data validation results tab. The results may have contained warnings showing where the data were missing. Basic counts like the number of investigations accepted would be displayed in the data report tab. The user could submit the data by clicking on the submit button or could upload another XML file and go through the entire upload process again.

If there were errors, the user could view the validation results on the data validation results tab. The results listed the data errors where the XML violated the validation rules. The data corresponding to the errors would not be saved in the database. The results could also contain warnings showing where the data were missing. A new XML file without any errors needed to be uploaded to proceed further. Figure 10.2 describes this process visually.

A federal user could view status and reports for the Case Component for any state. A federal user would see the same page as the state user but only the data validation results and the data report tabs would be visible. The file upload tab would not be visible. All fields would be disabled, and there would be no way to save or submit data.

A technical user or administrator could submit, view, and approve data for the Case Component for any state. A technical or administrative user would see the same page as the state user. There would also be a dropdown with all states in it so that the user could select which state they wanted to view. The user could upload an XML file for any state. The user would see radio buttons for each data status, with the current status selected. The user could change the status and save. There would be no submit button for this user--data could be submitted by changing the status to in process. Figure 10.2 shows the data submission and approval process as a flow chart.

FIGURE 10.2. Case Component Approval Workflow