

Architecture Notebook

Contact Center Operations

|  |  |  |
| --- | --- | --- |
| Revision History | | |
| Revision | Date of Release | Purpose |
| 1.0 | 4/16/2014 | Initial Release |
| 2 | 5/9/2014 | OY 2 Update |
| 2.1 | 7/16/14 | Updated for Supervisor and Quality eCL |
| 3.0 | 5/19/2017 | TFS 6620 - Updated to follow the new Architecture Notebook template |
| 3.1 | 5/30/2017 | TFS 6620 – Updated to follow the new Architecture Notebook template;  More update after Suzy’s review. |
| 4.0 | 8/18/2017 | TFS 7109 – Updated SQL Server version to 2012 |

Table of Contents

[1. Purpose 3](#_Toc483892754)

[2. Architectural goals and philosophy 3](#_Toc483892755)

[3. Assumptions and dependencies 3](#_Toc483892756)

[3.1 Assumptions 3](#_Toc483892757)

[3.2 Dependencies 3](#_Toc483892758)

[4. Architecturally significant requirements 4](#_Toc483892759)

[5. Decisions, constraints, and justifications 4](#_Toc483892760)

[6. Architectural patterns 5](#_Toc483892761)

[7. Key abstractions 6](#_Toc483892762)

[8. Key architectural frameworks 6](#_Toc483892763)

[9. Architectural views 6](#_Toc483892764)

[9.1 Logical View 8](#_Toc483892765)

[9.2 Development View 9](#_Toc483892766)

[9.3 Physical View 10](#_Toc483892767)

[9.4 Process View 11](#_Toc483892768)

[9.5 Security Diagram 12](#_Toc483892769)

[10. Architecture History 12](#_Toc483892770)

[11. Product Integration 12](#_Toc483892771)

[11.1 Integration Stragegy 12](#_Toc483892772)

[11.2 Integration Sequence 12](#_Toc483892773)

[11.3 Integration Process/Procedures 13](#_Toc483892774)

[11.4 Integration Environment 13](#_Toc483892775)

eCoaching Log

Architecture Notebook

There is guidance within this template that appears in a style named InfoBlue. This style has a hidden font attribute that allows you to toggle whether it is visible or hidden in this template. Use the Microsoft® Word® menu **Tools > Options > View > Hidden Text** check box to toggle this setting. There is also an option for printing: **Tools > Options > Print**.

# Purpose

This document describes the philosophy, decisions, constraints, justifications, significant elements, and any other overarching aspects of the system that shape the design and implementation.

The eCoaching Log is designed to provide quality feedback at all levels of the Contact Center Operations (CCO)

* Phone and Alternate Channels Customer Service Representatives (CSRs), including Advanced Resolution Center, Web Chat, Written Correspondence and Email
* Supervisors of CSRs
* Quality Specialists
* LSAs
* Trainers

The coaching log builds on our goal of continuous improvement to achieve sustainable results.

# Architectural goals and philosophy

* The system should utilize existing systems and infrastructure to the greatest extent possible.
* Utilize data from existing data sources
* Separate the various layers of the system from each other to be in line with a n-tier architecture
* Limit the amount of data that can be manually entered to reduce / eliminate the possibility of beneficiary PII/PHI being entered into the system.

# Assumptions and dependencies

## Assumptions

* Team has experience using Microsoft SQL Server and associated tools including but not limited to:
  + SQL Server Management Studio
  + SQL Server Reporting Services
  + SQL Server Integration Services
* Team has experience using ASP.Net and related technologies from Microsoft
* Team has the required access to the resources and tools to do their job
* Stable and suitable hardware is available to support the application and monitored for health
* Data is backed up on a regular basis
* Code is stored in a Central repository
* System can be recreated from scratch
* Application is for DDIT employees only. Subcontractors are not supported.
* Application is not considered critical and Outage Notifications will not be communicated to CMS
* Customers are internal GDIT staff

## Dependencies

* Data is available from the following in a timely manner and is accurate.
  + Data feeds
    - Attendance Tracking System
    - ETS
    - Outliers from Performance Management
    - Quality
    - Training
  + Employee feeds
    - Employee Information from Aspect
    - PeopleSoft
    - HR Employee Information from HR

# Architecturally significant requirements

All requirements for the eCoaching log can be found in TFS (Team Foundation Server). No requirements are defined as architecturally significant.

# Decisions, constraints, and justifications

| **Decision, Constraint** | **Justification** |
| --- | --- |
| All users of the system must have an active directory account to authenticate to the application. | Centralizes account authentication; the application does not need to perform authentication outside of Active Directory. Ensures users are GDIT CCO employees in good standing. |
| Roles will be used to control the functions a user is authorized to access. | This enforces the concept of least privilege. |
| Keep all pages simple to walk the user through the steps needed to complete a transaction. Utilize plain writing. Use clear headings and subheadings. | Supports usability Quality Attribute. |
| The user interface should be consistent across all pages; page layout, fonts, colors, etc. This can be accomplished through the use of Cascading style sheets (CSS) and Master pages. | The use of style sheets improves page rendering.  Usability Quality Attribute |
| Session lifetime should be limited to conform to program security requirements. Session termination will route the user a timed-out message and prevent page back | Security |
| Limit the use of parameters in the URL string to prevent manipulation. | Security |
| Menus and links to other sites should be easy for the user to find and use | Usability Quality Attribute |
| SSIS packages should be used to pull data from across multiple database servers and to produce reports. | Good programming technique  Security |
| Use Windows authentication for the database connection with an implementation of the trusted subsystem model. | Good programming technique  Security |

# Architectural patterns

The solution for the web application will use the Model-View-Presenter pattern. The following diagram demonstrates how this will be implemented within eCoaching Log system.

Archpattern

The ASP.NET code-behind technology, which uses partial classes, provides a natural implementation of the MVP pattern. The code-behind file (the Presenter) contains all the logic and processing code, and populates the page (the View). Event handlers within the code-behind file handle events raised by controls, or by the page itself, to perform actions when a postback to the server occurs. To complete this pattern, the code-behind file can use a separate data access layer or component (the Model) to read from, write to, and expose the source data - usually accessed through built-in providers that are part of the .NET Framework.

The solution will make use of the most of the following design patterns:

* Observer
* Iterator
* Decorator
* Adapter
* Factory
* Strategy
* Template Method
* Intercepting Filter Pattern
* Page Controller

Please see [Discovering the Design Patterns You're Already Using in the .Net Framework](http://msdn.microsoft.com/en-us/magazine/cc188707.aspx) by Rob Pierry for a detailed description of how these design patterns apply to the design of the project.

# Key abstractions

* CSRs - Loaded into the pull down menus
* Supervisors – Loaded into the pull down menus
* Quality Specialists – Loaded into the pull down menus
* LSAs – Loaded into the pull down menus
* Trainers – Loaded into the pull down menus
* Hierarchy - Employee hierarchical tables are current and accurate

# Key architectural frameworks

Quality attributes describe the system’s intended behavior. It provides a mechanism to determine the fitness and suitability of the system. By defining the quality attributes of the system, the architecture designed to fulfill these goals.

The high level quality attributes for this system include the following:

* Correctness – the system does what is expected of it
* Availability – the percentage of the time the system will be available for use by the user community.
* Manageability – how easy it will be for the system administrators to manage the operation and installation of the system.
* Performance – the responsiveness of the system to perform a given transaction.
* Reliability – the ability of the system to remain operational over time.
* Scalability – the ability of the system to scale with the number of users of the system.
* Security – the ability of the system to prevent malicious or accidental attacks on the system.
* Testability – the ease with which can be made to demonstrate its faults through testing.
* Maintainability – the ability of the system to undergo changes over time.
* Usability – defines how well the system meets the user requirements by being intuitive, easy to locate functions and providing good access to the disabled.

# Architectural views

Views are used to present the system to the stakeholders on the solution. A view is a representation of a set of system elements and the relationships associated with them. The following is a modified version of Philippe Kruchten‘s 4+1 view to describe architecture in multiple views. This modified view is composed of six main views:

* The logical view, which shows documents the logical layers of the solution.
* The process view which captures the concurrency and synchronization aspects of the design
* The physical view which describes the mappings of the software onto the hardware.
* The development view describes the static organization of the software in its development environment.
* Scenarios view describes the functionality of the system
* Security documenting how the security will be implemented to support the solution.



## Logical View

The Logical view describes the structure and behavior of architecturally significant portions of the system. This might include the package structure, critical interfaces, important classes and subsystems, and the relationships between these elements. It also includes physical and logical views of persistent data, if persistence will be built into the system. This is a documented subset of the design.



**Client – The client is the end user’s web browser.**

**Presentation Layer -** The presentation layer contains the components that implement and display the user interface and manage user interaction. This layer includes controls for user input and display.

**UI Components** - The visual elements used to display information to the user and accept user input.

**UI Process Components** - Presentation logic is the application code that defines the logical behavior and structure of the application in a way that is independent of any specific user interface implementation.

**Business Layer** – The business layer contains all business logic, workflow and rules associated with the application. It is concerned with the processing of the application data. It is designed to maximize reuse across the application.

**Application Façade** – This is a simplified interface to the business logic components. It combines multiple processes into a single operation.

**Workflow Components** – Workflow components define and coordinate long running, multistep business processes.

**Business Rules** – Encapsulates the business logic and data necessary to represent real world objects and processes in the application.

**Data Layer** – This layer contains the interface between the application and the data stores in the system. It centralizes all data access components into one layer and facilitates re-use of application code.

**Data Access Components** – These components are the logic required to access the data stores (databases and file system).

**Data Sources** – This contains all the data files and database files associated with the application.

**System Services** – This contains all services and functions provided by the operating system and other low-level applications that comprise the system.

## Development View

A definition of the software layers associated with the system.



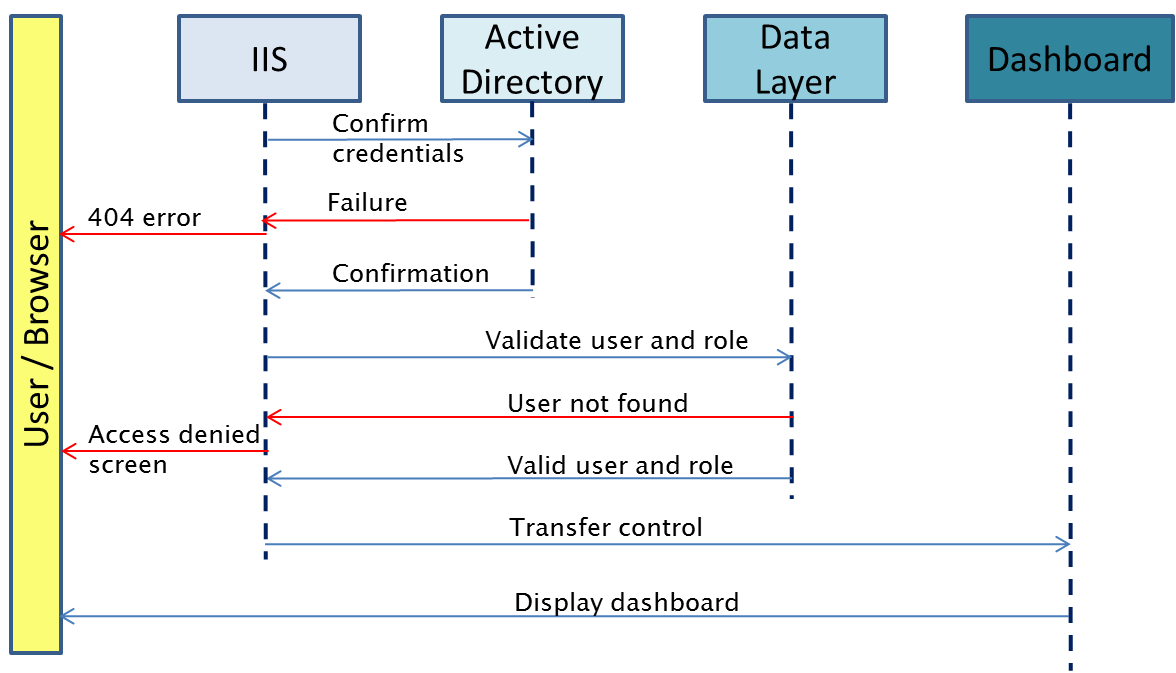
## Physical View

The physical view describes the physical nodes of the system and the processes, threads, and components that run on those physical nodes. This view isn’t necessary if the system runs in a single process and thread.



## Process View

The process view documents the sequences of events within the eCoaching Log intranet web application system.



## Security Diagram

The following security diagram describes how security is layered within the system.



# Architecture History

This was a new architecture. Subsequent changes will be documented in this section.

# Product Integration

External data feeds will be loaded daily to keep data current within the system. Complete details of the integration activity will be include in the project plan.

## Integration Stragegy

The components that will be integrated with the eCoaching Log system include:

* Attendance Tracking System
* ETS
* Outliers from Performance Management
* Quality
* Training
* Employee information from People Soft
* Employee information from Aspect
* HR employee information from HR

## Integration Sequence

Install database

Install UI Code

Setup required access and permissions

Setup for receiving Feed files

Stage SSIS Packages

* Schedule Jobs

## Integration Process/Procedures

Integration steps will be documented in runbooks that will be maintained in the <F3420-TFSASP01\CCO Program> team project in Team Foundation Server at \eCoaching\_V2\Runbook.

## Integration Environment

The eCoaching Log application will be using a development, test and production environment owned and operated by GDIT.

The following environment must be in place for the integration of the application:

* Web server running Windows 2008 or later with IIS and the .Net framework installed
* Database server with Microsoft SQL Server 2012 SP3, SQL Server Agent, SQL Server Integration Services and SQL Server Reporting Services installed and configured.
* A file server with a share enabled for file delivery and staging
* An Encryption service for Encrypting and Decrypting files at rest on the staging server

Engineers must have access to the servers and databases to the extent needed to perform the job.