Department of Veterans Affairs:

**Benefits Claims Decision Support (BCDSS) System**

Contractor Project Management Plan (CPMP)



**September 2016**

Version 1.11



**1101 King Street, Suite 200 Alexandria, VA 22314**

Revision History

| Date | Version | Description | Author |
| --- | --- | --- | --- |
| 09/20/2016 | 1.11 | [Updated Stakeholders and Key Personnel section](#_Stakeholders_and_Key)  [Updated Communication Matrix](#_Communication_Matrix)  [Updated Staffing Assignment Matrix](#_Staffing_Assignment_Matrix)  [Updated section 7.2 Schedule](#_Schedule)  [Updated section 8.1 Project Deliverables](#_Project_Deliverables)  [Updated section 8.2 Project Milestones](#_Project_Milestones) | BCDSS Team |
| 08/23/2016 | 1.10 | [Update section 7.2 Schedule](#_Schedule) | BCDSS Team |
| 07/20/2016 | 1.9 | [Updated Avi Moses in Stakeholders and Key Personnel section](#_Stakeholders_and_Key)  [Updated Avi Moses in Communication Matrix](#_Communication_Matrix)  [Updated Avi Moses in Staffing Assignment Matrix](#_Staffing_Assignment_Matrix) | BCDSS Team |
| 06/24/2016 | 1.8 | Updated Sections 1.4 Stakeholders and Key Personnel, 2.2 Organizational Boundaries and Interfaces, 7.2 Schedule, 8.1 Project Deliverables, 8.2 Project Milestones, 9. Communication Matrix, 14.1 Roles and Responsibilities Matrix and 14.2 Staffing Assignment Matrix | BCDSS Team |
| 05/20/2016 | 1.7 | Updated various sections, changing PST to BCSS.  Removing Pete’s name from the list. Adding Heath Forney. | BCDSS Team |
| 04/27/2016 | 1.6 | Updates on Sections 1.4,6.1.1,7.2,8.1,8.2,9,14.2 | BCDSS Team |
| 03/26/2016 | 1.5 | Updates on Sections 1.4,6.1.1,7.2,8.1,8.2,9,14.2 | BCDSS Team |
| 2/07 and 2/08 | 1.2/.3/.4 | Updated various sections bring current  through the month of January | BCDSS Team |
| 11/27/2015 | 1.1 | Updated Sections 1.4 Stakeholders and Key Personnel, 2.2 Organizational Boundaries and Interfaces, 6.1.2 Tools, Programming Languages and Methods, 7.2 Schedule, 9. Communication Matrix, and 14.2 Staffing Assignment Matrix | BCDSS Team |
| 10/26/2015 | 1.0 | Initial | BCDSS Team |

Table of Contents

[1. Introduction 1](#_Toc462143729)

[1.1 Project Overview 1](#_Toc462143730)

[1.2 Scope Statements 1](#_Toc462143731)

[1.3 Goals and Objectives 2](#_Toc462143732)

[1.4 Stakeholders and Key Personnel 2](#_Toc462143733)

[1.5 Technical Approach 3](#_Toc462143734)

[1.5.1 BCDSS System Functional Requirements 4](#_Toc462143735)

[1.5.2 Design BCDSS System 6](#_Toc462143736)

[1.5.3 Develop BCDSS Predictive Models 9](#_Toc462143737)

[1.5.4 Develop BCDSS Systems 12](#_Toc462143738)

[1.5.5 BCDSS Pilot 16](#_Toc462143739)

[2. Project Organization 17](#_Toc462143740)

[2.1 Organizational Structure 18](#_Toc462143741)

[2.2 Organizational Boundaries and Interfaces 18](#_Toc462143742)

[3. Acquisition Process 19](#_Toc462143743)

[4. Monitoring and Control Mechanisms 19](#_Toc462143744)

[5. Systems Security Plans and Requirements 21](#_Toc462143745)

[6. Technical Process 21](#_Toc462143746)

[6.1 Methods, Tools, and Techniques 21](#_Toc462143747)

[6.1.1 Development Methodology 21](#_Toc462143748)

[6.1.2 Tools, Programming Languages and Methods 23](#_Toc462143749)

[6.1.3 Technical Standards, Policies, Procedures, and Guidelines 23](#_Toc462143750)

[6.2 Testing Support 23](#_Toc462143751)

[6.3 Product Component Test (Unit Testing) 24](#_Toc462143752)

[6.4 System Testing 24](#_Toc462143753)

[6.5 User Acceptance Testing 25](#_Toc462143754)

[7. Work Breakdown Structure (WBS) and Schedule 25](#_Toc462143755)

[7.1 WBS 26](#_Toc462143756)

[7.2 Schedule 27](#_Toc462143757)

[8. Project Success Criteria 28](#_Toc462143758)

[8.1 Project Deliverables 28](#_Toc462143759)

[8.2 Project Milestones 29](#_Toc462143760)

[9. Communication Matrix 30](#_Toc462143761)

[10. Risk Management Plan 31](#_Toc462143762)

[10.1 Risk Identification and Analysis 31](#_Toc462143763)

[10.2 Risk Assessment 31](#_Toc462143764)

[10.3 Risk Action Planning 32](#_Toc462143765)

[11. Software Configuration Management (SCM) Plan 32](#_Toc462143766)

[12. Quality Assurance Plan 33](#_Toc462143767)

[13. Project Measurement Plan 34](#_Toc462143768)

[14. Resource Plan 34](#_Toc462143769)

[14.1 Roles and Responsibilities Matrix 34](#_Toc462143770)

[14.2 Staffing Assignment Matrix 35](#_Toc462143771)

[15. Requirements Traceability Matrix (RTM) 35](#_Toc462143772)

[16. Disaster Recovery Plan 35](#_Toc462143773)

[17. Reference Materials 35](#_Toc462143774)

# **Introduction**

This Contractor Project Management Plan (CPMP) describes the project management guidelines, best practices, and processes that Team ProSphere will follow during execution of the Benefits Claims Decision Support (BCDSS) System project. This CPMP will govern the management practices across the life of the project. As those practices evolve, this document will be updated to reflect the changes.

## **Project Overview**

The mission of the Department of Veterans Affairs (VA) is to provide a seamless experience of benefits and services to Veterans of the United States. VA, Veterans Benefits Administration (VBA) is undergoing transformation to significantly modernize and improve the delivery of benefits to the nation’s Veterans. In meeting these goals, VA seeks to utilize advanced data science techniques to enhance service delivery and integrate these techniques into the existing service delivery and underlying technology application and data infrastructure. To drive the success of transformation objectives, VA, in partnership with industry, must provide extensive strategic, technical, and program management direction to guide all transformation activities. That direction must be supported by advanced technology and data science expertise.

One such strategy is to automate the process for adjudication of claims. Doing so will allow VBA adjudicate claims faster and reduce the overall claim volume by avoiding future claims for rating increases, all while simultaneously assuring that the Veteran receives the full compensation benefit he or she is due. VBA previously conducted analyses to determine the feasibility of using existing (original) disability claims data to anticipate future claims for adjustments based on the correlation between specific hearing and musculoskeletal conditions and the likelihood that a Veteran will file an adjustment. VBA is now seeking to determine whether this analytical methodology can serve as the foundation for developing a solution (the BCDSS System) capable of accurately deriving disability rating determinations with little or no human intervention.

## **Scope Statements**

In 2013, Compensation Service initiated a research effort to determine the feasibility of using predictive models to automate disability claims processing and focused on requests for increase claims. This research developed predictive models to forecast disability ratings based on the type of claim, Veteran characteristics, military service, demographic data, previous claims history and ratings, and other key parameters.

The research to date has been focused on producing predictive models for rating disability claims for the hearing and knee body systems which represent the highest frequency of disabilities claimed. VBA seeks to create a software platform to implement predictive models to rate historical disability claims. This document contains an overview of the architecture components which combine to form the BCDSS and VBA system. This document describes the system design, and the strategy for realizing the architecture vision.

## **Goals and Objectives**

The goal of the Benefits Claims Decision Support System (a.k.a. BCDSS) is to demonstrate the feasibility of automatically determining the level of disability and adjudicating a claim for benefits based on predictive modeling of historical claims analyses.

* To develop two models within an Operating Platform in a Prototype Solution based in the boundaries, conditions, expectations and statements contained within: contract number VA118A-12-D-1015, order number VA118-1015-0036
* To demonstrate a BCDSS Prototype that successfully fulfills contract number VA118A-12-D-1015, order number VA118-1015-0036
* To deploy BCDSS Prototype Solution into a sandbox environment to properly evaluate the capability, results and speed of operations in order to determine the feasibility of deploying such a solution into a Pilot.
* To conduct Pilot

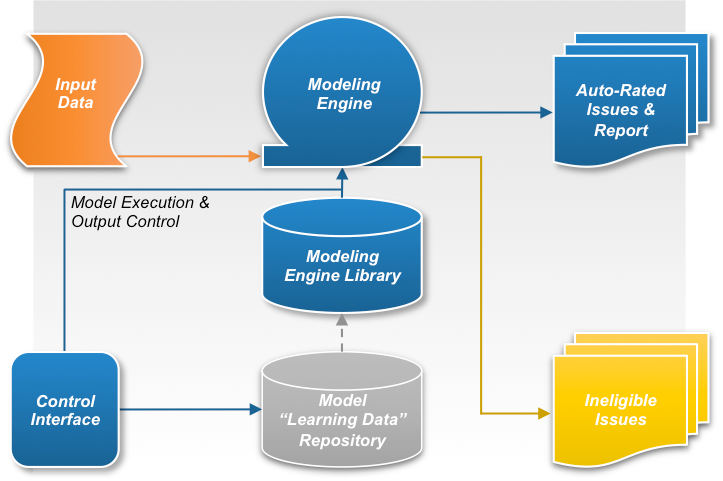
## **Stakeholders and Key Personnel**

**Table 1: Stakeholders and Key Personnel**

|  |  |
| --- | --- |
| **Name** | **Title/Role** |
| Veteran | VA End User Community |
| Summer Spalliero | VA Contracting Officer |
| Tinamarie Giraud | VA Contract Specialist |
| Heath Forney | VA Contracting Officer Representative & Project Manager |
| Elizabeth Wollin | VA Project Manager |
| Erik Rothwell | BCDSS Project Manager |
| David Teague | BCDSS Ear and Knee Modeling SME |
| Avi E Moses | BCDSS Ear and Knee Modeling Analyst |
| Chiranjeevi Puttaswamy | BCDSS Requirements Analyst |
| Darrell Dorman | BCDSS Configuration Manager |
| Dominic Yeh | BCDSS Developer |
| Masud Habib | BCDSS Developer |
| Vasudeva Rayapati | BCDSS Software Architect |
| Jeffrey Bamba | BCDSS Database Analyst |
| Ganesh Panneer | BCDSS SCRUM Master |
| Bhupinder Pal Singh | BCDSS PST Test/QA |
| Evan Weber | BCDSS Technical Writer |

## Technical Approach

Team ProSphere’s proposed solution architecture is depicted **Figure 1**. As this figure suggests, the BCDSS System will be composed of four primary components:

***Predictive Model “Model Formulation Data” Repository:*** This component stores statistically relevant enterprise data supporting current and future predictive models. These data will be used to formulate, validate, and refine current and future predictive models;

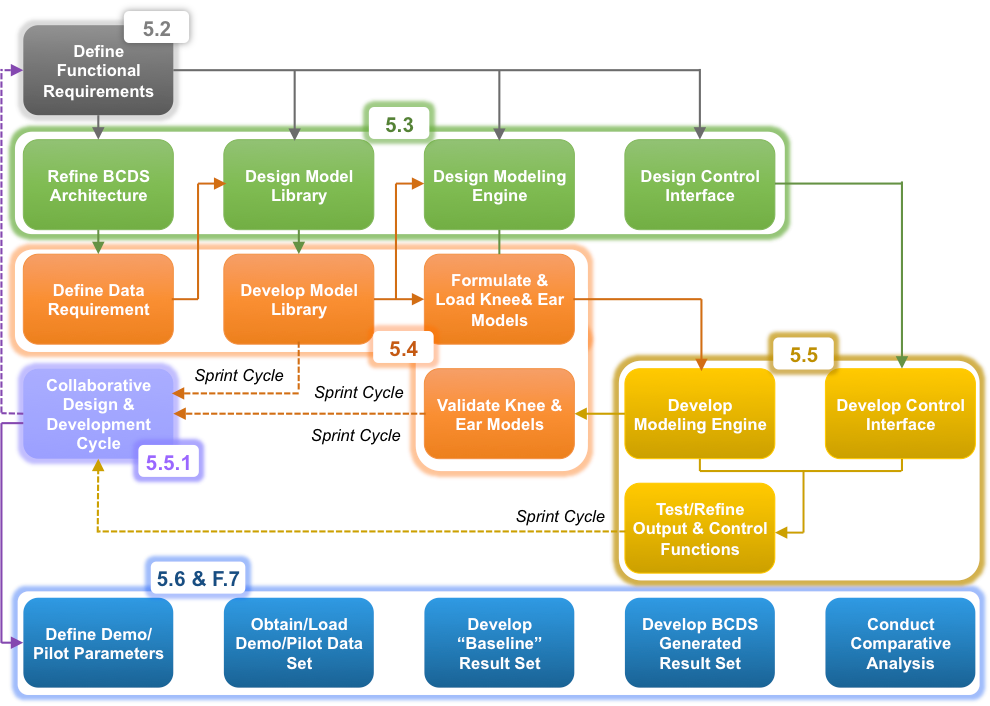
***Predictive Model Library:*** This component stores the established predictive models for use by the modeling engine to automatically determine ratings for issues contained within relevant claimant data sets.

***Modeling Rules Engine:*** This component provides the platform and modeling logic for ingesting claimant data sets, determining whether the predictive models can be applied, applying the predictive models, and outputting either auto-rated issues and the related report for the claimant data set, or the ineligible issues for the claimant data set;

***Control Interface:*** This component allows authorized users to control how the data ingest and auto-adjudication system operates, provides means to control the format and content of the output, and provides the mechanism to control the predictive model library and refinement of the underlying data.

Team ProSphere’s development approach for the BCDSS is depicted in **Figure 2**. We provide a detailed description of the activities conducted in each iterative development cycle in the sections that follow.

**Figure 2: Team ProSphere’s BCDSS Development Approach**



### BCDSS System Functional Requirements

Team ProSphere will employ an Agile-based iterative design and development process. A key component of this methodology is to capture and translate business and functional requirements into user scenarios or “stories” that provide a realistic depiction of how users will interact with the system, and their expectations for its performance.

VACI has established high-level BCDSS functional requirements in Section 5.2 of the PWS, including seven specific functionalities. Consequently, Team ProSphere’s approach to functional requirements focuses principally on translating these summary level requirements, in collaboration with the open source community, into discrete “buildable” tasks through a series of Agile sprint cycles. Resulting tasks form the basis of a build/test backlog. Initially, these sprint cycles are divided into four logical groups, sequenced in accordance with the initial architecture:

1. Validation and refinement of the architecture and build-out of the technical infrastructure, as well as data requirements associated with replicating the initial predictive models,
2. Design and development of the models and associated library,
3. Design and development of the modeling engine, and finally,
4. Design and development of the system controls, output formats, and user interfaces.

A summary of our approach to each high-level requirement is provided in **Table 2**.

**Table 2: Team ProSphere’s Summary of Approach**

| **Req. No.** | **Requirement Description** | **Team ProSphere Approach** |
| --- | --- | --- |
| 0-A | A Rating Engine that fetches, populates, tests, and stores predictive models and utilizes them to rate VA disability claims | Our BCDSS solution architecture includes a modeling engine capable of retrieving, verifying (for format, completeness, and integrity) specified predictive models, and storing the existing Knee and Ear predictive models, as well as models that may be developed in the future. The modeling engine will apply the models to claimant data to rate specific knee or ear issues that conform to the modeling parameters and determine the appropriate rating. |
| 0-B | A model repository capability to create, test, and store the predictive models | Our BCDSS solution architecture includes a model “Model Formulation Data ” repository – this database will allow the team’s analytical experts to verify/refine the existing analytical basis for the Knee and Ear predictive models, as well as create new models using appropriate algorithms (e.g., Random Forest) or classifiers and that conform to VBA specific accuracy and through-put thresholds. Once created, the predictive models will be available for ingestion into the modeling engine. |
| 0-C | A rating engine which consists of a software platform, separate from the BCDSS model repository which will receive claims and orchestrate the appropriate models and data to rate the incoming claims. | Our BCDSS solution architecture incorporates the rating engine into the modeling engine – this is separate from the model repository. As indicated earlier, the modeling engine determines what incoming claim data is eligible for auto-rating (as well as that which is ineligible), applies the appropriate predictive model, and determines the rating for the specific contention. It then orchestrates output and reporting. |
| 1 | The capability to provide predictive models for providing a recommended rating for the contentions on Veterans’ disability claims. | Our BCDSS modeling engine will first evaluate specific claimant data for contentions with corresponding predictive models. It will then apply the predictive models and generate issue-specific ratings where applicable. An updated claimant data set and accompanying report will be generated, identifying ineligible contentions and eligible contentions, along with the automated rating that was generated, and the analytical basis for that rating. |
| 2 | The capability to receive feeds from new supplemental claims requesting an increase to the Veteran’s rating. | Our BCDSS solution will evaluate the entirety of a claimant’s data set to identify relevant contentions, and consequently it is agnostic as to the type of claim, (irrespective of end-product). Relevant “profile” information will be retained to provide a traceability mechanism to origination data. |
| 3 | The capability to determine which contentions on each claim can be rated by predictive models which exist in the BCDSS Model Repository. | See No. 1. This is a key element of the process for applying predictive models within the BCDSS modeling engine, as current conceived by Team ProSphere |
| 4 | The capability to retrieve data from the Veteran’s claims history, claims text, and Veteran demographic information as detailed in the Engineering Notebooks indicated in the Applicable Documents 2.0 section of this PWS. | The model “learning data” repository, as envisioned in our design, will serve as a storage medium for evaluating claimant data and configuring it for analysis consistent with the Engineering Notebooks (Applicable Documents 2.0). The system will be capable of retrieving and uploading this data within the repository. This will also enable further analysis and refinements of the existing models, and development of new models. |
| 5 | The capability to orchestrate the models and corresponding data and produce a recommended rating for the contentions on the Veteran’s claim. | As currently envisioned, the BCDSS modeling engine will identify applicable contentions, evaluate and recommend the appropriate ratings for those contentions, and re-integrate the auto-rated contentions with the all other claimant data to form a completed data set. A report will accompany the re-integrated data set identifying which contentions were auto-rated, and the basis for the ratings. This Report can be made available (in the future) to the rater and subsequently uploaded to the eFolder. |
| 6 | The capability to separate model logic from platform code and have the capability to be directly authored, modified and managed by the line-of-business independent of the underlying BCDSS System. | The system will allow predictive models to be altered independent of the platform code residing within the modeling engine, and will prevent alterations that will compromise model data integrity or conformance with modeling engine requirements. The analytical requirements associated with verifying the accuracy and throughput of models imposes specific limitations on automating the evaluation process. The system will not be capable of controlling for analytical validity, or to ensure the resulting model conforms to the original throughput or accuracy thresholds. |
| 7 | The capability to produce a report for each rated contention that provides an analysis of how the rating was determined by the models and be made available to the rater and subsequently be uploaded to the Veteran’s eFolder | See No. 5. Each claimant data set processed by the system will be accompanied by an auto-generated report identifying which issues were rated by the system, and the basis for that rating. |

### Design BCDSS System

Specific design activities will be conducted to establish the overall system architecture. Different stakeholders have unique interests and concerns with respect to various aspects of the BCDSS system. Team ProSphere will leverage elements of IEEE 1016, Software Design Descriptions and IEEE 42010, Architecture Descriptions to produce a comprehensive document referred to as the Software Design Document (SDD) that addresses these stakeholder concerns and interests. Software design is an iterative process by which elements of the design are altered to align with the customer expectations. Assumptions, constraints, desires and understanding change over time and the design document should reflect these evolutions.

BCDSS system design activities are closely integrated with the process for developing and building the related BCDSS predictive models being conducted in parallel. These activities also overlap with BCDSS development activities. As a result, the SDD will be treated as a living document with established baselines approved through a Configuration Management process. The BCDSS design will be formulated in five phases, each aligned towards different stakeholders of the BCDSS system. These include: Data, Logical, Process, Development, and Physical, This approach is rooted in the Department of Defense Architecture Framework (DoDAF) 2.0 which has a greater emphasis on data.

**Figure 3: BCDSS System Design Activities**



* The Data phase will produce designs covering data models, data flow and data dictionaries that support both BCDSS specific functional requirements, as well as non-functional requirements (e.g. security). The focus of this data design effort is to establish a comprehensive data ontology of sufficient detail to codify the BCDSS data ecosystem.
* The Logical phase aligns stakeholder/user interests and concerns (e.g. usability considerations, reporting needs, responsiveness/performance interests, etc.) with system design parameters and the logical construction and inter-relationship between target BCDSS components.
* The Process phase produces the design specifications for control flow of the system, and communication interfaces between system components and business workflow.
* The Development phase decomposes the system into subsystems, software modules and classes or compile-time units consistent with the specifications of the preceding phases. Core artifacts are then provided to the software development team responsible for implementation.
* The Physical phase addresses performance constraints, environmental and logistic issues, as well as deployment and maintenance of the system.

**Data Phase**

Data architecture addresses what data is accessed, how it is accessed, how it is organized, how it is represented and how it is stored. The majority of the data functional requirements, and data analysis, have been provided within the Engineering Notebooks, developed as part of the proof-of-concept effort. The Engineering Notebook entitled ‘Data Preparation’ addresses what data will be accessed. This is detailed in a data dictionary which includes the field name, field type and description, of the Claims and Veterans data within Appendix A & B.

Working in collaboration with our predictive modeling team, Team ProSphere will first validate the data specifications provided in the engineering notebooks. We will next address how best to access/obtain the necessary data, particularly the interface controlling inbound data to determine security access protocols, the sequence of calls required to produce a comprehensive dataset, and volumetric constraints on data access. Additional service level agreements (SLA) with the VA data provider will impose constraints and performance limitations on the model repository; whether VACI chooses to have the team load the data from the VA provider at the outset, or wait until the Pilot is complete.

Existing Engineering Notebooks, based upon hearing and musculoskeletal conditions, provide a roadmap to the data organization and representation necessary for future predictive models. Team ProSphere will then produce detailed data flows, data models and data dictionaries for each predictive model consistent with the procedures governing how the predictive models were initially developed for hearing and musculoskeletal conditions.  Because the transformation of the incoming dataset into the feature vector data set required by the predictive modeling is a multi-step process, this process will also be defined as part of the data design. This process will be focused on capturing the intermediate data flows and interfaces between each processing stage of the transformational data processing chain.

Data analysis will also yield the appropriate data storage layout for the training, testing, and operational data for the predictive models. A key design consideration impacting the storage structure of the both the repository and model library relate to the extent and characteristics of changes in the structure of the predictive models (either expansion, or a fixed data set that is replaced/replenished at fixed intervals). If the “Model Formulation Data set” contained within the repository is to grow through accumulation, then the design must accommodate increasing capacity and indexing requirements to ensure search performance specifications can be satisfied. However, if the “learning data set” is fixed in size but replenished at set intervals, then the design must address data purging, retention, and archiving protocols and strategies. The data design must also determine which persistence model is best suited for the BCDSS system. For example, VACI appears to indicate that inbound data will come from a traditional RDBMS. However, an open source MongoDB option may provide a better option given the analytical needs of BCDSS.

**Logical Phase**

Too often usability is an afterthought - resulting in a dejected user base, poor system adoption rates, and a system plagued by a fractured design and workflow problems. Team ProSphere will address usability as a core part of the overall system design. We start with user analysis and field research to elicit user interface requirements. In this case, Team ProSphere’s design team will be working, as part of our collaborative development methodology, through a series of engagements with the open source community, VBA stakeholders, and VACI to obtain direct input and feedback on design considerations. Additionally, information architecture is applied to the requirements to determine optimal data input, output and workflow, as well as expected behavior, alternative operations and the error-handling (i.e., “out-of-bounds” data and events).

Team ProSphere will next develop a comprehensive set of requirements for human-computer interfaces. A key component of our process is the development wireframe layouts, or screen designs, using an interactive methodology to produce a well aligned interface. Historically, wireframes were snapshots, or even drawings, representing a static view of the visual layout of key user interfaces. Our use of interactive wireframes allows Team ProSphere to incorporate usability testing directly into the design process, validating information and process flow prior to full-scale development of the system component, thereby increasing development efficiency, reducing downstream operational errors, reducing training requirements and ultimately increasing the adoption rate of the software upon release.

Team ProSphere will also use a series of performance questions, including response and processing times, to ensure the capture of critical nonfunctional requirements during the elicitation process. To assign responsibility to the correct artifact scope and stakeholder, Team ProSphere will perform analysis to categorize performance requirements. Properly aligning performance requirements to the appropriate stakeholder will ensure performance needs are addressed within the design specifications, while eliminating possible misinterpretation and/or rejection by the stakeholder. We also follow a rigorous reporting process to document thoroughly document design considerations and map them to the originating stakeholder/ customer. This process ensures our design remains aligned with, and fully supports the customer’s business goals.

**Process Phase**

Process architecture deals with the control flow of the system, communication between systems, and business process workflow. Each of these components addresses a unique customer interest or concern. We will develop an Activity diagram modeled in UML to detail the processing logic of internal systems, subsystems, packages and classes. External system interaction is captured as the Communication diagram artifact, again following industry best practices by utilizing UML. We will use Business Process Modeling Notation (BPMN) to model business workflow involving human-computer interaction and the orchestration of those resources through complex control states.

**Development Phase**

The architecture of the BCDSS system will follow the principle of Separation of Concerns throughout the decomposition and modeling of the system from the definition of subsystems through packaging and class structure. The degree of decomposition determines whether subsystems and components fall within the Architecture Description or the Software Design Description.

The first stage is to take the functional requirements and turn these into a series of user stories (a type of use case). The Team ProSphere employs the following template to develop user statements: “As a <type of user>, I want <some goal> so that <some reason>”.  This template captures critical elements for use within the software specification, namely, the actor or actionable user, system functionality and business value. In following the tenets of Behavioral Driven Development (BDD), acceptance criteria can be captured as a series of scenarios, represented as: “Given <some context>, when <event occurs>, then <expected outcome>.  Acceptance criteria is critical when determining if a user story is complete from a development perspective and provides an actionable list for QA to produce a test suite.

An SDD needs to have several artifacts, or models, produced as part of the design and analysis activities within the Development phase if it is to fully serve the needs of the development team. Team ProSphere will create a series of diagrams decomposing system interactions at various levels of detail - from high level subsystem interaction to low level class input and output parameters. These diagrams are tailored to illustrate, for specific system users and stakeholders, how system interactions, behaviors, and results address their interests or concerns, and consequently how the design reflects their input. These diagrams will be produced using UML, an industry best practice, selected to represent three standard “views”: Sequence, Component and Object diagrams. The sequence diagram shows the message or data flow between interfaces representing subsystems, services, packages and classes; collectively referred to as software modules. The component and object diagrams represent dialogue specifications, input parameters, and output parameters that serve as a descriptor of compile and load units.

Critical to the success of any managed software development project is traceability between implemented code and a requirement. A Requirements Traceability Matrix (RTM) is the cornerstone of several industry best practices and certifying bodies, such as CMMI. The RTM provides traceability through all phases of the development lifecycle. From a design perspective it will show a relationship between a representation of the requirement within a system of record to an artifact within the SDD. Team ProSphere will leverage the RTM throughout all phases of the development lifecycle to show full traceability of each phased artifact from requirement to implementation. Another critical use of a managed RTM is the ability to identify work products of a user story, referred to as a “project package”, which have test cases or test suites designed to allow user validation and acceptance at periodic points along the project continuum.

**Physical Phase**

Team ProSphere recognizes the need to consider constraints, limitations and trade-offs when developing hardware and software specifications, also known as the platform architecture. A failure to do so results in a solution that fails to address overall system requirements, or one that is “over-engineered” and unnecessarily exceeds the needs of the user. Another benefit of a comprehensive platform architecture is the ability to address discovered limitations with stakeholders prior to initiation of development.

Environmental concerns are paramount, and are not just limited to the execution platform. Team ProSphere considers the development environment when designing the platform architecture. Leveraging DevOps best practices, Team ProSphere can define and develop a homogenous development environment that supports all environments from development and testing, through production, to ensure the code is developed and tested against the same performance needs of the production system. Similarly, too often, logistics and maintenance of a system are a post deployment consideration. Team ProSphere addresses logistics and maintenance needs as part of the overall system design. For example, we evaluate and develop a disaster recovery plan inclusive of data recoverability, operational availability, environmental concerns, and security considerations.

### Develop BCDSS Predictive Models

VBA’s original disability claims analysis evaluated the characteristics of “original” claims for disability benefits for specific body systems, identifying specific fact-patterns (the nature of specific contentions in conjunction with descriptive phrases provided by the Veterans, input from the evaluating physician, etc.) that collectively provided a basis for determining the likely disability rating of the Veterans – both at the time of rating, as well as in the future (essentially, anticipating future claims for adjustments to the disability rating). Following specific data preparation protocols, VBA extracted contention and diagnosis text data from digital claimant files and identified those “features” within these fields for relevance and frequency to specific conditions (in this case, ear and knee injuries). Selected computational algorithms (e.g., Random Forest, logistic regression, ordinal logistic regression, auto-encoder) were then applied repeatedly to identify patterns that allowed ratings to be determined within a computed level of accuracy for a given “throughput” (or volume of claims).

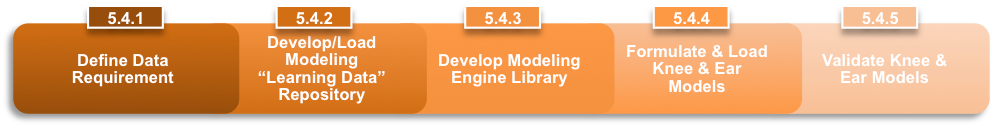
Team ProSphere’s approach to developing the models builds on the “output” of these analyses – the specific fact-patterns that produced ratings within specific accuracy thresholds. Our approach is tightly integrated with our overall BCDSS design and development. Because the system itself is driven by these models, development of the BCDSS infrastructure, data flow from the CDW (or in this case, anonymized data from VBA PA&I), and database architecture is developed in conjunction with the BCDSS Design team. Doing so provides a basis early in the development process to test data infrastructure and processing performance.

As noted earlier, adding additional information beyond contention and diagnostic data, may provide feature vector data sets that can expand the applicability of the models to a broader set of claims (e.g., original claims without contentions translating into diagnostic codes during the manual rating process). This is distinctly different from adding volume (throughput) or expanding the feature set for specified contentions or diagnostic data – as previously demonstrated statistically, there is a clear knee-in-the-curve at ~50 percent throughput where large reductions in throughput do not produce proportional improvements in accuracy. This is the operating point at which the team recommends that confidence thresholds be set for classification.

Team ProSphere, working in collaboration with experts from VBA’s Compensation service, will build on the original methodology to identify and verify claimant demographic, military service and historical transaction data (e.g., end product, frequency, and timing of previously submitted updates to claimant records) that may allow ratings to be determined automatically for an expanded set of claims using refined predictive models. We will set, as a starting point, the basic performance parameters originally identified within Mitre’s analysis. “While the classifier for the knee can be relied upon to make correct decisions with 85% confidence on 29% of the claims; classifiers for the ear can be relied upon to make correct decisions with 90% accuracy on 50% of the claims.”

Team ProSphere’s approach is illustrated in **Figure 4**.

**Figure 4: Approach to Model Development**

******

**Define Data Requirements**

During this step, Team ProSphere’s Requirements Analyst will define the specific data requirements that support both the model repository and modeling engine library. The analysis will also develop data transfer, performance specifications, and integrity protocols for the databases. As stated earlier, we will rely extensively on data specifications and translation procedures identified in “D-Engineering Notebook, Data Preparation” to expedite this process. However, a thorough validation of the field structure will be necessary to ensure BCDSS solution design considerations relating to performance are appropriately factored into the design.

**Develop/Load Modeling “Model Formulation Data” Repository**

Upon completion of the database structure, a “baseline” predictive Model Formulation Data set will be obtained and validated for integrity, business compliance, and form. This data will then be uploaded to the repository. As currently envisioned, this baseline data will be identical to (or closely approximate) the data set used previously to develop the initial analysis (e.g., he referenced “12 million anonymized historical claims records.”) Use of this data set will allow the team to reproduce the predictive models from the same underlying data and provide a basis for testing and calibrating each model to similar accuracy and throughput specifications.

**Develop Modeling Engine Library**

Systems and database development engineers will next develop the initial predictive modeling engine library. This library will be developed in close coordination with the BCDSS engine, and related business rules that govern identification of eligible claimant data sets, and application of the models to the data sets, and result set (and report) production following application.

**Formulate and Load BCDSS Predictive Models**

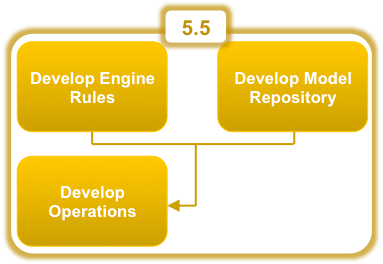
The Modeling SME and Analyst, working closely with VBA and the VACI, will next use the baseline data to generate and test the initial Knee and Ear predictive disability determination models in accordance with the Engineering Notebooks. These models are specific fact sets (a series of 20 or greater sets of contention and diagnostic features with a correlation of sufficient strength to be considered valid for predicting a specific rating for the contention) that can be thought of as forming an array of the various permutations of 32 or more features, narrowed first by frequency and use, and then by the strength of the correlation to the specific rating. Theoretically, a new series of claims data (with similar contention and diagnostic features) can be compared to this template to determine a correlated rating.

Key elements of the model development process will be evaluated by the open source community during Agile sprint cycles as part of the collaborative development process. Upon completion, the predictive models, they will be uploaded to the library for use by the engine. As part of this process, the models be tested against test claimant data (independent of the engine itself) to evaluate potential automation issues or challenges that may confront solution engineers during system programming.

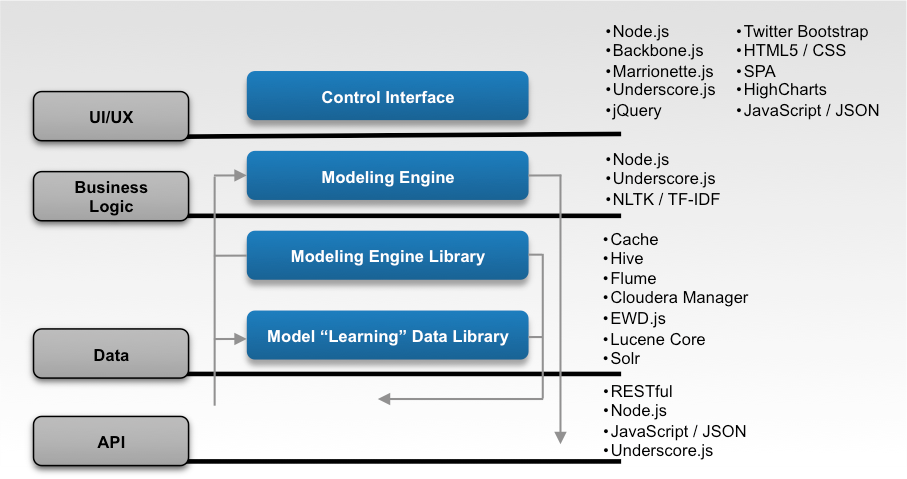
**Validate and Verify Knee and Ear Predictive Models**

Finally, the models will be applied, following completion of modeling engine development, to the baseline data to verify that the team’s formulated knee and ear predictive models are returning results similar to those produced by VBA in its initial analysis.

### Develop BCDSS Systems

Team ProSphere will employ a phased approach to building the BCDSS System, focusing on developing the Rules Engine, developing the Model Repository and implementing a strong and focused Development Operations (DevOps) process.  As such, Team ProSphere will develop functional, logical, and physical data models and strategies to support data interoperability in accordance with COR approved baseline SDD and BCDSS Predictive Models, **Figure 5**.

To execute our interrelated RAD phase approach, Team ProSphere will employ a Service Oriented Architecture (SOA) using industry standards, including the World Wide Web Consortium (W3C).  This will allow each independent system to be technology agnostic. Furthermore, our team will implement architectural principles - Loose Coupling, Separation of Concerns and Layers of Abstraction via RESTful APIs - to deliver a higher degree of software component reusability, increased system integration, enhanced quality control and assurance, and streamlined deployment, maintenance, and sustainment.  This SOA based approach, in conjunction with industry standards and best practices, also helps ensure compatibility across the system and provides a framework and architectural baseline for growth; a development characteristic especially important given the anticipated rapid application growth and maturation of the BCDSS prototype and component predictive models.

Team ProSphere utilizes a holistic Software Engineering methodology that includes designing, developing, implementing, testing, securing and maintaining the necessary data integration infrastructure. Our methodology also includes configuration baselines, source code control and related policies, build validations and defect resolution. Team ProSphere’s design envisions that the BCDSS Rules Engine and Model Repository will consist of four tiers or abstraction layers, each providing a logical separation of concerns so that new functionality can be added to and deployed with minimal impact to the BCDSS System, **Figure 6**.

These four tiers include the User Interface/User Experience layer (UI/UX), Rules/Business Logic layer, Data layer and the Application Programming Interface (API layer):

* Tier one will be the User Interface, UI/UX layer, where the user will interact with the system.  This layer will allow the end-user to view the Ratings output of the system, see graphs and reports pertaining to the data, and ultimately provide an interface for updating and changing the Predictive Models.
* The second tier will be the Rules/Business Logic layer.  This layer will be the central engine where target data ingression will be mapped against the models.
* The BCDSS Models will be stored in the third tier, Data layer.  This layer will serve as the central repository.  Further, operational data, such as logs, will be stored in the data layer in addition to textual content.
* The fourth tier, the API layer, includes all inputs into the system and outputs from the system, and will occur through an API interface.

**Develop the Rules Engine**

In the first phase, Team ProSphere will develop the Rules Engine which can subsequently be broken down into the UI/UX layer, Rules/Business Logic layer, and the Data layer.

**Tier 1 UI/UX Layer:** The UI/UX, will employ a Single Page Application (SPA) approach, simplifying document traversal and manipulation, event handling, UI and client-server interactions. The UI/UX will be built with Node.js. Node.js is known for building fast and scalable network applications. Backbone.js will be employed to provide the ability for the SPA approach by providing a rich feature set, including key-value data binding, custom event for user interaction, and a rich interface for iterations over collections of data. Backbone will be made easier to use, and allow for more rapid development by the supporting libraries of Marionette which provides additional behavioral and component utilities and view templates. Additionally, Underscore will provide functional methods for working with collections of data, further decreasing development time. Using standard web technologies and approaches, jQuery will be used for client-side JavaScript interactions, Hypertext Markup Language (HTML) 5 for markup and CSS for styling and layout. By combining the latter three technologies with Twitter Bootstrap and JavaScript, the UI will be highly portable, responsive, and modern; allowing for displaying data, as well as, for allowing for a rich and interactive user experience by targeting a multitude of browsers (including Internet Explorer, Firefox, Chrome, and Safari) and mobile phones and tablets. For additional, interactive graphing and displaying of data, HighCharts will be leveraged.

By using frameworks such as Twitter Bootstrap and technologies such as JavaScript, HTML5 and CSS, we will be able to rapidly iterate on and develop, design concepts, and deliver an experience that more accurately meets the need of users. Additionally, these technologies will allow Team ProSphere to develop software platforms using the best design patterns and practices for web development, ensure maintainability, and allow for future extensibility and reuse of portions of the BCDSS system. They also help us design the platform using best practice responsive web design and techniques.

**Tier 2 Rules/Business Logic Layer:** By employing Node.js’s event-driven, non-blocking I/O model, we will be able to take advantage of its lightweight, efficient and proven ability to perform extremely well with data-intensive, real-time applications at the Business logic layer.  Once again, this layer serves as the BCDSS Rules Engine.  Using Node.js for this layer will allow the system to ingress target data, load models, compare data, calculate Ratings, and egress results with speed and confidence.  Should any unstructured data or text come in the system that needs Natural Language Processing, such as Sentiment Analysis or Named Entity Recognition, libraries such as the Natural Language Toolkit (NLTK) and statistical analysis such as Term Frequency-Inverse Document Frequency (TF-IDF) will be employed.

**Tier 3 Data Layer:** There will be multiple datastores in play for the overall system, which as a whole will constitute the Data Layer.  For the Rules Engine, the models will be persisted across the MongoDB database Cache and will be accessed by using the Node.js-based Application Framework and Application Server/Container Edsger W Dijkstra (EWD).js.  This datastore will serve as the BCDSS Model Repository from which we pull known models as well as update and add new ones.  Further, the Apache web server and Lucene Core & Solr will be employed as a persistence store for fast and efficient retrieval of textual based information.  Hive will be used as the datastore for Flume logging.

**Develop the Model Repository**

Another phase of development for the BCDSS System centers on the developing the Model Repository and can be broken down into Database Management and the API layer.

**Tier 4 API Layer:** The API layer will consist of server-side Node.js in all cases possible.  All APIs we create will adhere to RESTful standards and will use JavaScript Object Notation, JSON, as the format for data exchange.  By staying consistent with a JavaScript technology stack (UI, Rules/Business Logic, API), our development team will be able to scale across tiers, and hence feature sets, as the project demands; embracing a full-stack developer concept.  This will be especially useful for testing, debugging, and enhancement efforts.  As such, our development team will use Jasmine for unit-testing; tying into our DevOps and CI&D process of Test Driven Development (TDD).

**Database Management:** As alluded to above, the repository models will be stored in a MongoDB database, as opposed to a relational one. This configuration has several benefits, including: new versions of the models can be easily created while retraining the previous versions; the need for ACID compliance as found in traditional relational databases is reduced, thus removing unnecessary bottlenecks or locks to processing the target data against the models; and a framework is provided in which development can occur in a much more rapid fashion as new models and changes to existing models are introduced.

Team ProSphere will baseline all model definitions and schemas and follow a repeatable revisions process for updates and modifications. Backups of the database will occur on a scheduled basis to prevent data loss in the event of corruption or unforeseen downtime. Developers will connect to the database via APIs and database security will be employed to prevent direct access. All data input will be validated before attempting to persist and all data retrieval will be validated against security permissions before retrieval to prevent unauthorized access.

Team ProSphere will manage any large-scale data analysis or map/reduce functionality via Hadoop and use Cloudera Manager for the administration of Hadoop. This will allow for reduced deployment times via automating the installation process and provide a range of reporting tools.

**Development Operations**

As part of Development Operations, the system will use Flume for logging and Hive for interrogating the logging data.  Our developers will use Vagrant for their local environments.  This will allow for a consistent development environment across all developers and thus produce a truer “production-like” environment.  Also, developers will use Grunt for local install and deployments tasks.  This will reduce errors and bugs associated with inconsistent versions of technologies, and will allow developer to rapidly develop and test new functionality locally.  GitHub will be the source code repository tool and all code will exist in a public facing repository.  Further, we will use industry standard and best of breed tools as part of its DevOps lifecycle; reducing deployment errors and minimizing downtime associated with moving from development to production systems.  By using infrastructure management tools such as Chef, data center orchestration software such as Puppet, which provides automation configuration and management of machines and software, as well as, virtualization containerization technology such as Vagrant, the DevOps team will ensure a consistent, rapid and highly controlled deployment process.  Access to all network servers will occur over Secure Shell (SSH), ensuring security and compliance.

**Quality Assurance:** QA Engineers and QA Analysts, embedded in the development Scrum team, will not only perform manual testing, but also, automated testing for function, regression, performance and security cases as defined by Test Plans and Test Cases developed during the gathering of requirement and within each sprint story’s definition of acceptance.  This testing in concert with the use of TDD, will serve as the backbone for quality throughout the Software Development Life Cycle (SDLC). Another component of the QA approach will include daily regression testing and automated testing. We will spearhead the creation of daily operating procedures, where we exercise our latest development build against a standard set of manual and automated tests; testing for features, security and performance. Through this regression testing, we are able to ensure a build of consistently high quality and are able to quickly identify functionality that has broken since the prior test. This activity has been a key factor in reducing the number of defects in the application and delivering high quality software to our customers.

**Continuous Integration and Delivery:**CI&D will serve as an essential component of software delivery, providing capabilities such as source code management, build and deployment automation, environment configuration and administration, and automated test execution. A well-defined and mature CI&D processes help accelerate the delivery of software.

Team ProSphere’s approach to building the BCDSS System and subsystems also includes Test Driven Development (TDD). This methodology allows our developers to write clean code that works. By writing a test before developing just enough production code, we are able to fulfill that test and report results back to the development scrum team to refactor the code. This process allows our team to fully assess and design requirements prior to creating the functional code.  Using Jasmine, a behavior-driven development framework for testing JavaScript, for writing our unit tests will allow us to keep a consistent approach, methodology and framework across all our tiers. Jenkins will be employed as the CI&D tool of choice for automated and manual builds and deployments.

We will develop extensive build scripts to compile source code, execute unit testing, create code coverage reports, automatically deploy into lower environments, and run functional and regression test cases in order to immediately identify errors in the code. An automated build framework not only helps solve integration issues early, but also helps build and deploy the complex products quickly, all while ensuring that the build processes are transparent, auditable and repeatable.   As Jenkins will be the build server with automated daily builds and notifications and because all code will be hosted in a public facing repository on GitHub, we will be able to take advantage of GitHub’s post-commit hooks, and via automation, we will be able get the latest checked-in code, run all unit tests against it, verify passing states, and if all successful, deploy to the lower environments. Vagrant will be used for local environments ensuring a consistent environment reducing complexity and time to deliver for users and clients.

All of these enhanced CI&D processes for checking code quality, auto build/code compilation and deployment will be documented and will provide detailed instructions and guides for the development Scrum teams to implement, such as code check-in procedures, developer guidelines around TDD relating to integration, build compilation design and workflows, operational instructions for deployments, branch naming conventions, branch schedules, release numbers and corresponding dates, and versioning conventions.

### BCDSS Pilot

Following completion of system development, Team ProSphere will Pilot BCDSS to evaluate performance and accuracy relative to a specific Pilot Plan developed in conjunction with VACI. Our team will employ a five phase piloting approach as depicted in **Figure 7** below.

**Figure 7: Team ProSphere 5 Phase Pilot Approach**



**Phase 1 – Define:** During Phase 1, we will work with VACI to define the objectives and performance parameters of the pilot. This includes development of the timeline, scale (number of claimant data sets or eFolders to be evaluated), evaluation metrics (or standard for measuring success), and primary decision structure. These will major parameters and will be documented in the Pilot Plan.

**Phase 2 – Obtain/Load:** The team will next obtain historical the record set corresponding to the parameters identified in the Pilot Plan. As currently envisioned, this data set will cover sufficient time for the claimants involved that BCDSS generated rating determinations can be compared to Veterans who received changes in the level of disability rating based on claims for increases.

**Phase 3 – Develop Baseline:** During this phase, a select set of claimant data sets (or eFolders) will be identified as the “Pilot Baseline.” The historical results will be catalogued and marked to support subsequent comparative analysis.

**Phase 4 – Develop Results Set:** The model will next be run against the complete database. Doing so provides the most statistically relevant basis for conducting a comparative analysis system performance – including, the extent to which all/only applicable contentions are being rated by the system.

**Phase 5 – Conduct Analysis:** A comparative analysis will then be performed to measure BCDSS performance relative to the metrics defined in the Pilot Plan. Included in this comparison is a direct comparison between the actual historical rating for “pilot baseline” claimant data sets determined by VBA rating staff, and those determined by BCDSS for the same claimant datasets. This comparison will also identify unforeseen benefits and challenges posed by the system’s performance. The results of the analysis will be documented in the ***Model Rating Pilot Report.***

# **Project Organization**

The VA Project Manager (PM) and Contracting Officer Representative (COR) along with Team ProSphere’s Project Manager (PM) will be responsible for coordinating and maintaining all project efforts, services, and solutions. This will be accomplished by ensuring that management and technical strategies and plans are communicated and that appropriate staff are integrated in the project to provide collaboration, advice, recommendations, and/or support. All other team members will be responsible for their respective development, testing, analyst, and coordination tasks.

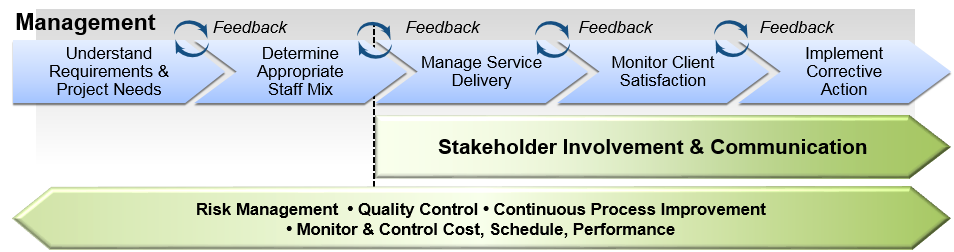
Team ProSphere’s management methodology approach is based on industry best practices, including PMBoK® for program management, Agile methodology for technical support, and is the basis of our International Organization for Standardization (ISO) 9001:2008 certification and aligned processes.

Our Team fosters effective communications through collaborative tools like SharePoint, supporting the team’s tasks and constant reporting and touch points (daily, weekly, monthly basis, depending on the priority level). We manage the CPMP and its integrated schedule of the teams’ progress with consistent and periodic monitoring of deliverables and other reporting mechanisms. Additionally, we mitigate risks and manage changes to the project through proactive monitoring of risk issues and collaboration with VA staff daily.

Team ProSphere follows the “management by deliverables” project management principles. This principle is predicated upon the production of visible work products. The benefit of instituting this approach is to gauge a percent complete criterion (quantitative value) based on physical evidence and to provide transparency into project progress. We will adhere to all inspection and acceptance guidelines in ushering deliverables through the program review cycle. Team ProSphere will continuously monitor performance and report to the VA PM and COR progress and all deviations and issues affecting performance on the project during routine, regular communications and in the monthly updated CPMP. This will include any deviations from the IMS or previous CPMP. We provide input to the COR/VA PM to address ad hoc requests from VA for project status information.

Our approach, as depicted in **Figure 8**, addresses all the required functions necessary to monitor the health of the overall Task Order (TO).

**Figure 8: Management Process**

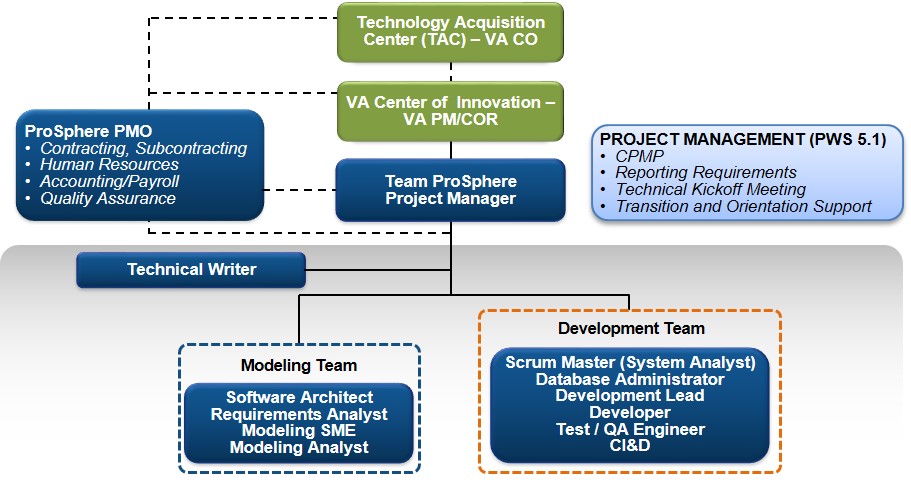


We believe in open, accurate, and timely communications; we understand it is important to immediately report significant issues and a corrective timeline to the COR, PM, and VA stakeholders. As a result, stakeholders have the opportunity to implement corrective actions and make sure that other internal or external groups receive early notification to execute proactive and preventative actions. As with many projects, the need for expedited communications is required when a serious risk or identified issue impacts approved project plans and processes. In these situations, we notify the COR by e-mail within 2 hours of identifying the issue, with the word “ALERT” in the subject line. As a regular practice, we also call the COR with a recommended resolution or a timeline for providing a resolution and follow up with an ALERT e-mail.

## **Organizational Structure**

Team ProSphere’s approach to the organizational structure and staffing of our team is intended to simultaneously provide structure and accountability, and yet offer flexibility and adaptability to support the BCDSS System project needs. **Figure 9** shows the structure of our delivery organization from a functional perspective, demonstrating our ability to manage all the detailed tasks listed under PWS.

**Figure 9: Functional Organizational Chart**



## **Organizational Boundaries and Interfaces**

Members of Team ProSphere have comprehensive work experience with the VA gained from past projects and know the VA environment. We offer solid project management and quality assurance processes to provide reliable services with a high degree of customer satisfaction.

Table 3profiles the three companies that comprise Team ProSphere: Pro-Sphere Tek, Inc. (ProSphere) and PWC.

Table 3: Assignment of Team Effort

| Team Pro-Sphere  Contractor | Role | Responsibilities |
| --- | --- | --- |
|  | Project Management and Statistical Adjudication Models Support (PWS Tasks 5.1 through 5.3) | ProSphere will lead an integrated team to execute the project mission objectives and all Performance Work Statement (PWS) requirements. Additionally, ProSphere will design the overall architecture of the BCDSS; as well as, design and develop the model repository and BCDSS rules engine that supports the predictive models used in rating VA disability claims. |
|  | BCDSS Design, Predictive Model Development, and BCDSS Pilot (PWS Tasks 5.3, 5.4 and 5.7) | PwC will provide the expertise needed to design and develop the predictive models for rating VA disability claims, help calibrate the BCDSS System once developed, and assist in the planning, implementation, and evaluation of the BCDSS Pilot. |

# **Acquisition Process**

The project management process will implement, maintain, and support the BCDSS System project maintenance policies, processes, and procedures in accordance with the PWS, and with the following primary purposes in mind:

* Estimate the scope of work that needs to be performed for the project
* Develop mechanisms to acquire and manage identified products and services
* Implement progressive elaboration for all artifacts, including documentation and source code
* Monitor progress against the CPMP and Project Schedule, and take action to address significant deviations from these documents
* Identify and analyze project-level risks and taking action to appropriately mitigate those risks and manage issues

# **Monitoring and Control Mechanisms**

This section establishes the process for Project Monitoring and Control for the BCDSS System project. Project Monitoring and Control is performed to provide understanding and insight into the project’s progress so that appropriate corrective actions can be taken before the project’s performance deviates significantly from the plan. Aspects of a project’s progress include interfaces to other organizations, deliverables, schedules, cost, effort, risk reviews, verification, validation, and amount of supporting services. Planned management of these aspects is captured in one or more project and system management plans.

**Major Tasks:**

* Monitor progress against the schedule by periodically measuring actual completion of activities and milestones; compare this progress against the planned documented schedule and identify significant deviations and trends
* Monitor resources provided and used by the project; compare the resources to the planned documented estimates and identify significant deviations and trends
* Monitor documented risks in the context of the project’s current status and circumstances; if the project circumstances change, which could give rise to new risk(s), incorporate relevant risk information into the Risk Identification process
* Revise the documentation on risks as additional information becomes available to incorporate changes; the project’s Risk Management Plan (RMP) details the process steps required to support risk identification and mitigation and is consistent with VA Risk Management (RM) policies
* Monitor the project’s work products and tasks by periodically measuring the actual characteristics of the work products and task, e.g. size, complexity, quality, security, etc.; compare the actual characteristics to estimates documented in the CPMP and Project Schedule, and identify significant deviations and trends
* Monitor commitments (internal and external) against the plan; monitor the status of stakeholder involvement against the plan and identify those commitments that have not been satisfied or those that are at significant risk of not being satisfied
* Manage corrective actions; gather issues for analysis developed during the previous tasks or input from other process areas
* Analyze issues to determine need for corrective actions; document the analysis and appropriate actions needed to address the identified issues
* Review and get agreement with the relevant stakeholders on the actions to be taken and the priority to be assigned for their completion
* Assemble project measures and the identified significant deviations and trends from what was planned in the CPMP and Project Schedule
* Communicate status on assigned activities, generate project status reports, and review progress; examples of progress reviews include: reviews with Team ProSphere; reviews with project management and suppliers; reviews with stakeholders and end users
* Conduct milestone reviews at meaningful points in the project’s schedule with relevant stakeholders
* Review the commitments, plan, status, and risks of the project
* Collect and document significant issues and their impacts as action items
* Track action items and issues to closure
* Collect and document issues that are found to have had a significant positive or negative impact on the project; if possible, provide a suggestion for improvement to processes and submit these significant issues (lessons learned) for communication to relevant stakeholders

# **Systems Security Plans and Requirements**

Team ProSphere will maintain a comprehensive Security Plan for both the facility and IT resources that adheres to strict enforcement of Federal laws and VA regulations concerning data security, including the Freedom of Information Act (FOIA), the Privacy Act, and the Federal Information Security Management Act (FISMA). Team ProSphere’s Security Plan was developed to ensure compliance with the above regulations. It also provides detailed policy statements that govern the authorization of use for Team ProSphere’s computer, network, and communications resources. Team ProSphere’s Security Plan is the primary source for information regarding security controls or information for the BCDSS System project. As required by VA, the security plan will follow the format prescribed in National Institute of Standards and Technology (NIST) Special Publication (SP) 800-18 “Guide for Developing Security Plans for Information Technology Systems,” with sections providing background and administrative information, as well as describing the managerial, operational and technical security controls required to protect the system. By following this approved format, the security plan will provide a consistent layout of information and assists in ensuring all required information is included within the plan. The system security plan is a living document, with constant updates as the system moves through its life cycle and changes occur to the system’s security controls.

Security Managementfunctions include:

* Protection Management – Defines the systems, resources, standards, and actions that prevent hardware, software, and system security breaches
* Risk Management – Determines the opportunity for and consequences of a system security compromise, accidental service disruption, or unwanted technical effects (i.e. vulnerabilities and risks of exposure)
* Access and Authentication – Establishes and manages guidelines concerning the authorized use of computers and data

# **Technical Process**

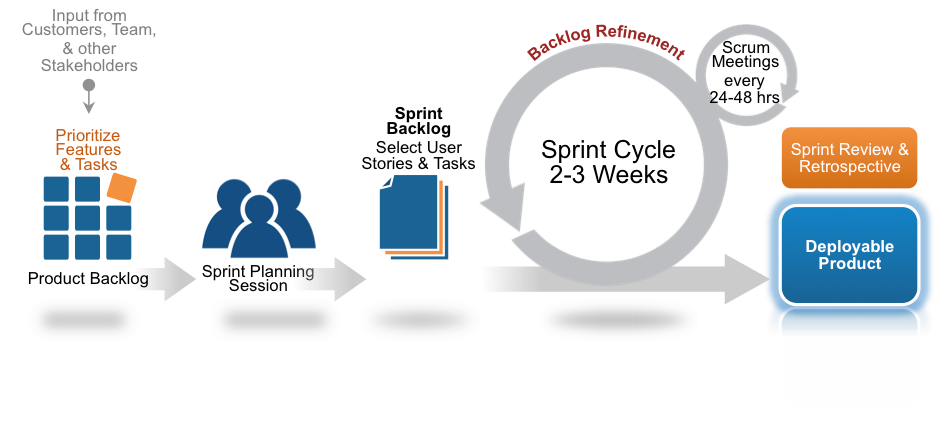
## Methods, Tools, and Techniques

The following are major process steps and tools that must be executed to successfully drive the BCDSS System project:

### Development Methodology

Team ProSphere’s Agile methodology provides both accountability and transparency in our efforts, while also reducing project risk, including risk introduced through unexpected challenges and changes in requirements. The effort will be comprised of iterative assessments, planning, execution, verification, and analysis. The Agile methodology as shown in **Figure 10** below enables iterative progressions of work with the means to frequently assess and measure results, and adjust tasks to maintain alignment with priorities – while working within VACI’s software development lifecycle framework, delivering required artifacts against the established schedule.

Figure 10: Team ProSphere’s Agile Methodology



Consistent with an Agile methodology, Team ProSphere will break individual requirements into user stories and tasks. A set of tasks will be developed during a pre-defined sprint increment and progress will be presented to the stakeholder group at regular sprint reviews. Feedback from the stakeholder group at the sprint reviews will be incorporated into the Product Backlog and will provide a basis for continuous improvement to the system. Our approach will enable impediments to be identified early, refinement of requirements to occur frequently, and allow project progress and challenges transparent across the stakeholder group.

Team ProSphere has defined the following standards for Scrum execution:

* Each Development Cycle will be composed of two sprints.
* Sprints will be two weeks in duration.
* On the second sprint of each month the team will work on the demo and video for the presentation to the customer, in addition to the user stories assigned for the sprint.
* The team will hold a Sprint Planning Session on the first day of the Sprint, typically a Monday except on holidays or other extenuating circumstances.
* The team will hold a Sprint Review and Retrospective meeting on the next Monday after finishing a sprint, except on holidays or other extenuating circumstances.
* The Scrum team holds daily Scrum meetings.
* Scrum calls may be cancelled on Sprint Review or Sprint Planning days per the discretion of project leadership.
* All Scrum calls will be strictly time-boxed to 15 minutes.
* The team will manage and groom all Product Backlog items via the Jira tool.

This approach empowers the team to deliver quality solutions that effectively and responsively meet business requirements and scheduling constraints. In addition to the Agile software development methodology, our approach is tailored to specifically address VA requirements, including: 1) automated testing for increased speed and efficiencies with issue identification and resolution; 2) a structured approach to configuration management that accounts for the servers within the IT environments; and 3) deployment and post-production support of IT products.

VA and the open source community will be providing collaborative input to this effort.  We will utilize a GitHub repository, provided by the VA, to maintain the Collaborative Deliverable Package as defined by VA. At the end of each sprint cycle, project artifacts will be delivered into the assigned project GitHub repository as part of the ***Collaborative Deliverable Package.***  Items included in the Collaborative Deliverable Package will meet VA Innovations Document Standards. We will review all collaborative input with VA for determination of applicability to the VA requirement. Since this code is made available to the public on the public GitHub repository, we will verify that the source code and test data do not include VA-sensitive information. We will provide a separate ***Internal VA-Sensitive Information Configuration File*** for information that may be required to support integration with VA production systems at a later date.   This file will not be included in the source code delivered to the GitHub repository or contributed to the open source community.  Proprietary software components of the overall solution such as binaries, libraries and installers will be published to a private GitHub repository provided by the VA.

### Tools, Programming Languages and Methods

Team ProSphere (PST) will use, to the maximum extent practical, open source tools in creating all Deliverables, specifically thoughts referenced in Section 5.5 and 5.5.1 of the PWS. If PST anticipates an instance where open source tools will not be used or may not be practical, the PM will notify the COR/PM both by email. Our approach is to use only software that is on the VA Technical Reference Model Technology/Standard List located at <http://www.va.gov/TRM/ToolListSummaryPage.asp>. PST understands the Government must approve any dependencies that our proposed solution has on licensed software. PST also understand that we must provide all licenses for software that is required in the development of our solution.

In addition to the tools specified in PWS, Team ProSphere will use the tools specified in **Table 4**.

Table 4: Additional Project Tools

|  |  |
| --- | --- |
| **Tool Category** | **Tool Name(s)** |
| Agile Management Tool | Atlassian JIRA |
| Software IDE | Eclipse IDE or equivalent |
| Programming Language | Java, Structure Query Language (SQL) |
| Defect Management | Atlassian JIRA |
| Operating Systems | Windows, Windows Server, Mac, Linux |
| Documentation | Microsoft Office |
| Project Management | Microsoft Projects |

### Technical Standards, Policies, Procedures, and Guidelines

Deliverables for this project will follow any applicable specifications as published by VA. Standard Operating Procedures (SOPs) and VA-established guidelines for process and project management will be followed.

## **Testing Support**

QA Engineers and QA Analysts, embedded in the development Scrum team, will not only perform manual testing, but also, automated testing for function, regression, performance and security cases as defined by Test Plans and Test Cases developed during the gathering of requirement and within each sprint story’s definition of acceptance. This testing in concert with the use of TDD, will serve as the backbone for quality throughout the Software Development Life Cycle (SDLC). Another component of the QA approach will include daily regression testing and automated testing. We will spearhead the creation of daily operating procedures, where we exercise our latest development build against a standard set of manual and automated tests; testing for features, security and performance. Through this regression testing, we are able to ensure a build of consistently high quality and are able to quickly identify functionality that has broken since the prior test. This activity has been a key factor in reducing the number of defects in the application and delivering high quality software to our customers.

## **Product Component Test (Unit Testing)**

Prior to the delivery of the build to the Test team, the developers will perform Product Component Testing, also known as Unit Testing, on code that has been developed. The objective of this testing event is to verify that the requirements defined in the detailed design specifications have been successfully applied to the module/component.

Utilizing commonly associated JSON testing platform for services which supports a number of parameters, will be used. As in most testing processes, the testing team will first create a Test Plan. This plan will include a proper set of benchmark numbers (similar to those in a Service Level Agreement) along with the client.

For purpose of this document, benchmark examples would be: Response time, Throughput and CPU Memory usage and so on. In this lowest-level testing, each unit (basic component) of the software is tested to verify that the detailed design for the unit has been correctly implemented with respect to design specifications.

## **System Testing**

Agile methodology will be followed for the development, therefore the development team will frequently deliver a new build to the test team, as new builds become readily available and stable enough to promote to test. System testing will be executed by the test team, using the FTL environment and the contractor’s test accounts.

The objective of this test event is to exercise all parts of an integrated system and provide quality assurance level verification of the code. This test event provides an opportunity to detect defects in the interfaces and interactions between the integrated components, as well as, to detect possible defects that may occur in the functionality of the code.

Testing will include:

* Features driven by business logic:
  + Business logic is the purpose. For whatever qualities exist or don’t, purpose is the reason (e.g. inputs/ outputs and which actors are involved in which ways).
* Inputs controlled by technical logic:
  + The technical components of the interface such as variable types and ranges.
* Service to perform at various load levels:
  + Multi-threading, well enough to validate the system under test (SUT) is handling multiple concurrent clients.
* Professional performance testing:
  + Normally associated performance tests from: shoe-string, (e.g. work flows and data flows), to creation of logs indicating what errors have been realized and how long transactions are taking, to work flow scripts.
* Service that is appropriately secure:
  + At minimum, as stated and/ or defined within the TO.

When completed, this testing event will verify that the product components, having been verified against the requirements, perform according to functional specifications and any defects found have been addressed to the satisfaction of the customer.

## User Acceptance Testing

The objective of User Acceptance Testing (UAT) is to:

* Ensure compliance of the application with business requirements
* Check the business functionalities or logic from a risk perspective and
* Certify that the system meets all business requirements

UAT includes testing requirements for functional, operational, performance and interface areas. The fact that UAT takes place after the development of an application in the project lifecycle makes it even more critical, as it’s the last chance to screen and solve issues which were not detected earlier in the development lifecycle. UAT is about identifying the gaps between how the completed application works and how it’s expected to perform in a business environment. These acceptance criteria could also include additional requirements to the feature, both functional and non-functional.

**Definition of Done**

When Agile teams are working on a product, it’s important for them and the Product Owner to have a mutual understanding of what it means when a User Story from the Product Backlog is done and make this as transparent as possible. They come to an agreement of what the definition of done is and agree to only call a User Story done when it checks all the boxes on the list. An increment should only be called done when it is a working, fully tested and meets all acceptance criteria.

The Product Owner is the person responsible for maximizing the value of the product. By collaborating closely with the Development Team, the Product Owner can continuously deliver feedback about the product and can accept a user story as done when it meets the definition of done. During Sprint Review, feedback will be collected from the client and all stakeholders as input for the Product Backlog. This ensures a continuous feedback loop that enables the Development team to always deliver exactly what the Product Owner and stakeholders want.

Transparency and client collaboration ensure that all checks in the definition of UAT are done. This improves throughput, which in turn reduces the time of the feedback loop making the product a higher value to the client.

Pairing UAT with process-streamlining Agile development practices, will ensure that applications will be used effectively once deployed, and Agile promises to shorten the UAT process.

# **Work Breakdown Structure (WBS) and Schedule**

The BCDSS System project team uses an integrated planning approach to track elements in this section.

## WBS

Our high level WBS is presented in **Table 5**.

**Table 5: WBS**

|  |  |
| --- | --- |
| WBS | Task |
| 1 | Project Management |
| 2 | Functional Requirements |
| 3 | BCDSS Software Design Document (SDD) Phase |
| 4 | BCDSS Predictive Models |
| 5 | BCDSS Development |
| 5.1 | Configure Development Environment |
| 5.2 | Develop Model Repository |
| 5.2.1 | Sprint A – Development |
| 5.2.2 | Sprint B – Development |
| 5.3 | BCDSS R1.0 Release Management |
| 5.3.1 | Develop Rules Engine |
| 5.3.1.1 | Sprint 0 – Development |
| 5.3.1.2 | Sprint 1 – Development |
| 5.3.1.3 | Sprint 2 – Development |
| 5.3.1.4 | Sprint 3 – Development |
| 5.3.1.5 | Sprint 4 – Development |
| 5.3.2 | Development Operations |
| 5.3.2.1 | Sprint 5 – Development |
| 5.3.2.2 | Sprint 6 – Development |
| 5.3.2.3 | Sprint 7 - Development |
| 5.3.2.4 | Sprint 8 - Development |
| 5.3.2.5 | Sprint 9 - Development |
| 5.3.2.6 | Sprint 10 - Development |
| 5.3.2.7 | Sprint 11 - Development |
| 5.3.2.8 | Sprint 12 - Development |
| 5.3.2.9 | Sprint 13 - Development |
| 5.3.2.10 | Sprint 14 - Development |
| 5.3.2.11 | Sprint 15 - Development |
| 5.3.2.12 | Sprint 16 - Development |
| 5.3.2.13 | Sprint 17 - Development |
| 5.3.2.14 | Sprint 18 - Development |
| 5.3.2.15 | Sprint 19 - Development |
| 5.3.2.16 | Sprint 20 - Development |
| 5.3.2.17 | Sprint 21 - Development |
| 5.3.2.18 | Sprint 22 - Development |
| 5.3.2.19 | Sprint 23 - Development |
| 5.3.2.20 | Sprint 24 - Development |
| 5.3.2.21 | Sprint 25 - Development |
| 6 | BCDSS Pilot |
|  | Model Rating Pilot Report |
| 7 | BCDSS Project Closeout |
|  | Project Summary and Production Integration/Transition Report |

## Schedule

Team ProSphere’s Project Plan Schedule, presented in the Gantt chart, and was developed by deconstructing the major WBS detailed in the approach into subtasks that directly contribute to either task order deliverables or major events (Milestones). Milestones and Deliverable dates were then set to correspond with the delivery dates established in the Task Order. Incremental sub-tasks were synchronized with the milestone and delivery dates.

A Contract Mod was issued on February 2016.

A Contract Mod was issued on September 2016 for Ear and Knee Models.

This incorporates:

|  |  |  |
| --- | --- | --- |
| CLIN/SLIN | From | To |
| 0002 BCDSS System Design | March 27, 2016 | April 30, 2016 |
| 0002AA Software Development Document | January 27, 2016 | April 15, 2016 |
| 0002ABRequirements Traceability Matrix | January 27, 2016 | April 15, 2016 |
| 0003 Develop Models | March 27, 2016 | May 27, 2016 |
| 0003AA Prediction of the Disabilities of the Ear | January 27, 2016 | April 15, 2016 |
| 0003AB Prediction of the Disabilities of the Knee | January 27, 2016 | April 15, 2016 |
| 00031 Reverse/Enhance Predictive Models | September 30, 2016 | December 30, 2016 |
| 00031AA Predictive Model of the Ear – Tinnitus | September 30, 2016 | December 30, 2016 |
| 00031AA Predictive Model of the Ear – Hearing Loss | September 30, 2016 | December 30, 2016 |
| 00031AB Comparative Performance Analysis of Ear Models | September 30, 2016 | December 30, 2016 |
| 00031AC Knee Model Decomposition Analysis and Recommendations for future models | September 30, 2016 | December 30, 2016 |

The Modified BCDSS Project Schedule is available below:



**Figure 11A: BCDSS Modified Project Schedule to 9/27/2016**

# **Project Success Criteria**

The BCDSS System project’s success criteria can be described as follows:

* Estimate the scope and work that need to be performed, develop a project plan, and clearly define roles and responsibilities
* Identify, analyze, and mitigate risks to enable the project to proactively identify and reduce risks that may jeopardize achieving project objectives
* Establish and maintain the integrity of work products
* Provide objective visibility into, and feedback on, processes and associated work products throughout the lifecycle of the project
* Determine which decisions should use a formal evaluation process, and develop and sustain a measurement capability that is used to support management information needs
* Analyze possible decisions using a formal evaluation process that identifies alternatives against established criteria
* Ensure that selected work products meet their specified requirements and demonstrate that a product or product component fulfills its intended use when placed in its intended environment
* Establish an infrastructure for continuous process improvement by identifying assets and creating guidelines to incorporate lessons learned
* Establish and manage the project and all relevant stakeholders according to an integrated and defined process that is tailored from standard processes
* Deliver deliverables on-time, on-target, and within budget

## Project Deliverables

Critical project deliverables and their respective dates are maintained and tracked in the Project Schedule and included in **Table 6**.

**Table 6: Project Deliverables**

|  |  |  |
| --- | --- | --- |
| **CLIN #** | **Deliverables** | **Due Date/Recurrence** |
| 0001AA | Contractor Project Management Plan | Due October 27, 2015 and updated monthly thereafter throughout the PoP. |
| 0001AB | Final Section 508 Compliance Test Results | Due in five days after testing is completed. |
| 0002AA | Software Design Document (SDD) | Draft due April 15, 2016. Baseline due five business days after receipt of Government comments. Update due five business days after each sprint cycle. |
| 0002AB | Requirement Traceability Matrix (RTM) | Draft due April 15, 2016. Baseline due five business days after receipt of Government comments. Update due five business days after each sprint cycle. |
| 0003AA | Prediction of Disabilities of the Ear | Draft due April 15, 2016. Final due thirty (30) days after receipt of Government comments. |
| 0003AB | Prediction of Disabilities of the Knee | Draft due April 15, 2016. Final due thirty (30) days after receipt of Government comments. |
| 0004AA | Collaborative Deliverable Package | Due thirty (30) days after approval of the SDD, baseline is due five business days after receipt of Government comments. Update is due five business days after each development cycle. |
| ~~0004AB~~ | ~~Internal VA-Sensitive Information Configuration File~~ | ~~Draft due thirty (30) days after approval of the SDD, baseline is due five business days after receipt of Government comments. Update is due five business days after each development cycle.~~ |
| 0005AA | Electronic BCDSS Demonstration Reports | Due five business days after each demonstration. |
| 0005AB | BCDSS Demonstration Video | Due five business days after each demonstration. |
| 0006AA | Model Rating Pilot Report | Draft due thirty (30) days prior to the end of the Pilot, final due five days after receipt of Government comments. |
| 0007AA | Project Summary and Production Integration/Transition Report | Due 10 days prior to the end of the task. |
| 00031 | Reverse/Enhance Predictive Models | The Period of Performance for this task is from September 30, 2016 through December 30, 2016 |
| 00031AA | Predictive Model of the Ear – Tinnitus | Draft due October 28 2016, Final due 10 days after the receipt of Government comments. |
| 00031AA | Predictive Model of the Ear – Hearing Loss | Draft due October 28 2016, Final due 10 days after the receipt of Government comments. |
| 00031AB | Comparative Performance Analysis of Ear Models | Draft due November 29 2016, Final due 10 days after the receipt of Government comments. |
| 00031AC | Knee Model Decomposition Analysis and Recommendations for future models | Draft due December 15 2016, Final due 10 days after the receipt of Government comments. |

## Project Milestones

Key project milestones and respective dates are maintained and tracked in the Project Schedule and included in **Table 7**.

**Table 7: Key Project Milestones**

|  |  |
| --- | --- |
| **Milestone** | **Date** |
| Project Award | September 28, 2015 |
| Kick-Off Meeting | October 5, 2015 |
| Contractor Project Management Plan | October 27, 2015 |
| Draft SDD | April 15,2016 |
| Draft RTM | April 15, 2016 |
| Draft Prediction of Disabilities of the Ear Model | April 15, 2016 |
| Draft Prediction of Disabilities of the Knee Model | April 15, 2016 |
| Final Prediction of Disabilities of the Ear Model | May 15, 2016 |
| Final Prediction of Disabilities of the Knee Model | May 15, 2016 |
| Collaborative Deliverable Package | May 27, 2016 |
| Draft Predictive Model of the Ear – Tinnitus | October 28, 2016 |
| Draft Predictive Model of the Ear – Hearing Loss | October 28, 2016 |
| Final Predictive Model of the Ear – Tinnitus | November 11, 2016 |
| Final Predictive Model of the Ear – Hearing Loss | November 11, 2016 |
| Draft Comparative Performance Analysis of Ear Models | November 29, 2016 |
| Final Comparative Performance Analysis of Ear Models | December 13, 2016 |
| Draft Knee Model Decomposition Analysis and Recommendations for future models | December 15, 2016 |
| Final Knee Model Decomposition Analysis and Recommendations for future models | December 29, 2016 |
| Model Rating Pilot Report | August 23, 2017 |
| Project Summary and Production Integration/Transition Report | September 14, 2017 |

# **Communication Matrix**

The Communications Matrix is a consolidated list of current communication items within the plan. The matrix serves as a quick reference for communication items. Project documentation, test documentation, and ad hoc materials will be added to this matrix as the project progresses.

**Table 8: Communication Artifacts**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Communication Item | Delivered To | Media | Frequency | Provider |
| Weekly Status Meeting | VA PM/COR & CO | Document | Weekly | Project Manager |
| SME Meeting Minutes | VA PM/COR & CO | Document | Weekly | Project Manager |
| Backlog Report | VA PM/COR & CO | Document | Bi-Weekly | Project Manager |
| Sprint Retrospective & Planning | VA PM/COR & CO | Document | Bi-Weekly | Project Manager |
| CPMP Updates | VA PM/COR & CO | Document | Monthly | Project Manager |
| BCDSS SDD | VA PM/COR & CO | Document | Monthly | Project Manager |
| BCDSS RTM | VA PM/ COR & CO | Document | Coincides with above SDD | Project Manager |
| Monthly T4 Report | VA PM/COR & CO | Document | Monthly | Project Manager |
| BCDSS Status Report | VA PM/COR & CO | Document | Monthly | Project Manager |
| Model Rating Pilot Report | VA PM/COR & CO | Document | On-time | Project Manager |

Table 9: Contact Information

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Group | Email | Office Phone |
| Summer N. Spalliero | VA CO | [Summer.Spalliero@va.gov](mailto:Summer.Spalliero@va.gov) | 732-440-9609 |
| Tinamarie Giraud | VA CO Specialist | [Tinamarie.Giraud@va.gov](mailto:Tinamarie.Giraud@va.gov) | 732-440-9641 |
| Heath Forney | VA COR/PM | [Heath.Forney@va.gov](mailto:Heath.Forney@va.gov) | 619-500-4402 |
| Elizabeth Wollin | VA PM | [Elizabeth.Wollin2@va.gov](mailto:Elizabeth.Wollin2@va.gov) | 703-507-0922 |
| Erik Rothwell | PST Project Manager | [Erik.Rothwell@pro-spheretek.com](mailto:pete.Erik.Rothwell@pro-spheretek.com) | 843-343-2608 |
| Vasudeva Rayapati | PST Software Architect | [Vasudeva.Rayapati@pro-spheretek.com](mailto:Vasudeva.Rayapati@pro-spheretek.com) | 512-788-1317 |
| Chiranjeevi Puttaswamy | PST Requirements Analyst | [Chiranjeevi.Puttaswamy@pro-spheretek.com](mailto:Chiranjeevi.Puttaswamy@pro-spheretek.com) | 571-224-6672 |
| David Teague | PST Modeling SME | david.c.teague@us.pwc.com | 202-730-4435 |
| Avi E. Moses | PST Modeling Analyst | avi.e.moses@us.pwc.com | 202-730-4193 |
| Jeffrey Bamba | PST Database SME | [jeffrey.bamba@pro-spheretek.com](mailto:jeffrey.bamba@pro-spheretek.com) | 240-678-0034 |
| Ganesh Panneer | PST Scrum Master | [ganesh.panneer@pro-spheretek.com](mailto:ganesh.panneer@pro-spheretek.com) | 703-810-3115 |
| Bhupinder Pal Singh | PST Test/QA | [Bhupinder.Singh@Pro-spheretek.com](mailto:Bhupinder.Singh@Pro-spheretek.com) | 703 980-9344 |
| Evan Weber | PST Tech. Writer | Evan.Weber@pro-spheretek.com | 703-810-5733 |
| Darrell Dorman | PST Configuration Manager | darrell.dorman@pro-spheretek.com | 703-810-3079 |
| Erik Rothwell | PST Dev Lead | [Erik.Rothwell@pro-spheretek.com](mailto:pete.Erik.Rothwell@pro-spheretek.com) | 843-343-2608 |
| Masud Habib | PST Developer | [Masud.Habib@pro-spheretek.com](mailto:Masud.Habib@pro-spheretek.com) | 906-370-8595 |
| Dominic Yeh | PST Developer | [Dominic.Yeh@pro-spheretek.com](mailto:Dominic.Yeh@pro-spheretek.com) | 703-810-3029 |

# **Risk Management Plan**

The RMP defines how the BCDSS System project team risks will be identified, analyzed, prioritized, mitigated, monitored, tracked, and reported. This plan contains descriptions of major elements to be used in this process:

Risk Identification

Risk Analysis – Assessment and Rating

Risk Responses – Contingency and Mitigation Strategies

Risk Monitoring

This plan also assigns specific roles and responsibilities for the management of risk and describes the documenting, monitoring, and reporting processes to be followed.

## Risk Identification and Analysis

Risk identification attempts to identify risks before they can impact the project. Risk identification begins during initial proposal preparation (i.e., the initial risk assessment) and continues on a regular basis (e.g., monthly) for the duration of the project. Risk analysis follows risk identification and involves analyzing each risk’s probability and impact to determine risk criticality.

## Risk Assessment

Once the risks have been identified, the state of risk is identified as below with the appropriate communication to the appropriate parties. The outline below shows the realist parameters for a project of this nature (Development for Pilot in a Sandbox environment) for determining project risk that will be used from this document point forward:

Major – Impacts Overall Cost & Schedule

* Identify Risk(s), provide dependencies, provide mitigation for or the resources needed
* Report same week to Mgmt (VA PM/ Director)
* Report within Monthly Report
* Track Weekly with Update

Medium – Impacts Overall Schedule Only

* Identify Risk(s), provide dependencies, provide mitigation for – or – the resources needed
* Report same week to Mgmt (VA PM/ Director)
* Report within Monthly Report
* Track bi-weekly and Update

Low

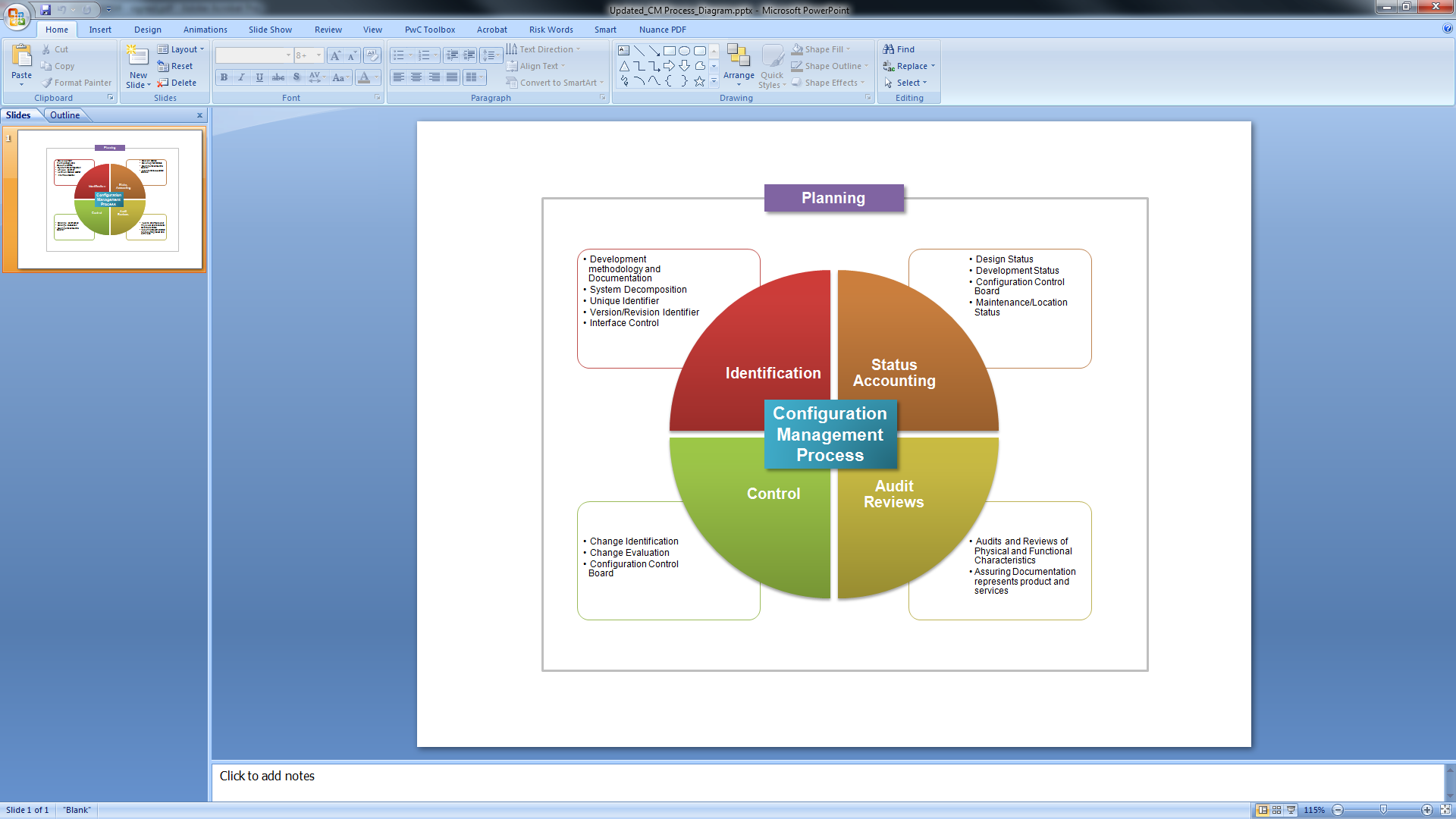
* Impacts next Milestone / Deliverable(s) Date Only
* Identify Risk(s), provide dependencies, provide mitigation for – or – the resources needed
* Report same Month to Mgmt ( VA PM/ COR)
* Track bi-weekly with Update

## Risk Action Planning

The BCDSS has implemented JIRA, utilizing this tool, any and all risks will be tracked accordingly starting the month of February. Risks will be entered into and kept within JIRA being tracked and reported/ updated according to the outline above (communication sent from within JIRA via email to those outside of the JIRA environment.

# **Software Configuration Management (SCM) Plan**

A SCM Plan entails establishing and maintaining the integrity of work products or supporting artifacts. In addition, it provides objective visibility into, and feedback on, processes and associated artifacts throughout the life of the project to support delivering high-quality products and services.

Figure 13: Configuration Management

BCDSS System software configuration control requirements will be specified in the SDD.

# **Quality Assurance Plan**

Team ProSphere will execute internal QA procedures of the BCDSS System project work and all associated deliverables to evaluate the quality of performance.

QA and Quality Control (QC) support and confirm the consistent and effective application of testing methodologies during the course of the project. They provide process discipline for Risk Management, CM, structured deliverable reviews, and documentation. The goal of the project quality process is to provide a quality surveillance plan for Team ProSphere services performed for the BCDSS System project program. This includes implementing quality reviews, maintaining Project Schedules, and ensuring the quality and timeliness of project deliverables.

Creating the project QC structure consists of planning quality features and designing the project QC plan to ensure that all personnel, materials, and IT support are in place to perform project QC activities. The desired outcome of project QC is a project completed on time, on target, and within budget. Team ProSphere desires not only to complete the project, but also to deliver products and services that have value and serve the needs of the Client.

The BCDSS System project team does not anticipate any deviation from the VA Quality Assurance Plan, which is located on the following link to the [VA Quality Assurance Standard](http://www.osehra.org/document/department-veterans-affairs-quality-assurance-standard).

# **Project Measurement Plan**

Team ProSphere will monitor performance against the established schedule, milestones, risks, and resource support outlined in the approved CPMP; any deviations will be reported in the monthly updated CPMP.

**Table 11: Performance Metrics**

|  |  |  |
| --- | --- | --- |
| **Performance Objective** | **Performance Standard** | **Acceptable Performance Levels** |
| **1. Technical Needs** | * Shows understanding of requirements * Offers quality services/products | Satisfactory or higher |
| **2. Project Milestones and Schedule** | * Quick response capability * Products completed, reviewed, and delivered in timely manner * Notifies customer in advance of potential problems | Satisfactory or higher |
| **3. Project Staffing** | * Currency of expertise * Personnel possess necessary knowledge, skills, and abilities to perform tasks | Satisfactory or higher |
| **4. Value Added** | * Provided valuable service to Government * Services/products delivered were of desired quality | Satisfactory or higher |

# **Resource Plan**

## **Roles and Responsibilities Matrix**

The roles and responsibilities matrix will be broken down to individual team member in future versions of the CPMP.

**Table 12: Roles and Responsibilities**

| Team Pro-Sphere  Contractor | Role | Responsibilities |
| --- | --- | --- |
|  | Project Management and Statistical Adjudication Models Support (PWS Tasks 5.1 through 5.3) | ProSphere will lead an integrated team to execute the project mission objectives and all Performance Work Statement (PWS) requirements. ProSphere will develop the BCDSS rules engine and model repository that are supported by the predictive models used in rating VA disability claims. |
|  | BCDSS Design, Predictive Model Development, and BCDSS Pilot (PWS Tasks 5.3, 5.4 and 5.7) | PwPPlProvides the expertise needed to develop the predictive models for rating VA disability claims, help calibrate the BCDSS System once developed, and assist in the planning, implementation, and evaluation of the BCDSS Pilot. |

## **Staffing Assignment Matrix**

**Table 13: Staffing Assignment Matrix**

|  |  |  |
| --- | --- | --- |
| Name | Group | Office Phone |
| Summer N. Spalliero | VA CO | 732-440-9609 |
| Tinamarie Giraud | VA CO Specialist | 732-440-9641 |
| Elizabeth Wollin | VA PM | 703-507-0922 |
| Heath Forney | VA COR/PM | 619-500-4402 |
| Erik Rothwell | BCDSS Project Manager | 843-343-2608 |
| Chiranjeevi Puttaswamy | BCDSS Requirements Analyst | 571-224-6672 |
| Vasudeva Rayapati | BCDSS Software Architect | 512-788-1317 |
| David Teague | BCDSS Modeling SME | 202-730-4435 |
| Avi E. Moses | BCDSS Modeling Analyst | 202-730-4193 |
| Jeffrey Bamba | BCDSS Database SME | 240-678-0034 |
| Ganesh Panneer | BCDSS Scrum Master | 703-810-3115 |
| Dominic Yeh | BCDSS Developer | 703-810-3115 |
| Masud Habib | BCDSS Developer | 906-370-8595 |
| Erik Rothwell | BCDSS Developer Lead | 843-343-2608 |
| Evan Weber | BCDSS Tech. Writer | 703-810-5733 |
| Bhupinder Pal Singh | BCDSS Test/QA | 703 980-9344 |
| Darrell Dorman | BCDSS Configuration Manager | 703-810-3079 |

# **Requirements Traceability Matrix (RTM)**

Team ProSphere will create, continually monitor and update the Requirements Traceability Matrix (RTM) of all functional requirements for the BCDSS System project. ProSphere will also track the source of the requirement, the date the requirement was identified, the system application/module assigned, and test status.

# **Disaster Recovery Plan**

Disaster recovery requirements will be addressed in the SDD.

# **Reference Materials**

* J. 1-Attachment A: Statistical Adjudication Final Report 9
* J. 2-Attachment B: VA Center of Innovation (VACI) Executive Board (EB) Deck 2015 v2 excerpt
* J. 3- Attachment C: Statistical Adjudication Engineering Notebook – Preliminary Data
* Exploration and Prediction of Disabilities of the Ear
* J. 4- Attachment D: Veterans Administration Disability Rating Engineering Notebook – Data Preparation
* J. 5- Attachment E: Veterans Administration Disability Rating Engineering Notebook 1 – Recommendations Based on Initial Classification Experiments
* J. 6-Attachment F: Veterans Administration Disability Rating Engineering Notebook 5 – Analysis of Rating Performance Differences and Sensitivity to Training S
* J. 7- Attachment G: Veterans Administration Disability Rating Engineering Notebook 6 – Analysis of the Suitability of Rating and Rating Change Random Forest Classifiers
* J. 8- Attachment H: Veterans Administration Disability Rating Engineering Notebook 7 –Analysis of Random Forest Meta Classifier for Tinnitus

Approval Signatures

This section is used to document the approval of this version of the Contractor Project Management Plan during the Formal Review. The review should be ideally conducted face to face where signatures can be obtained ‘live’ during the review however the following forms of approval are acceptable:

* Physical signatures obtained face to face or via fax
* Digital signatures tied cryptographically to the signer
* /es/ in the signature block provided that a separate digitally signed e-mail indicating the signer’s approval is provided and kept with the document

The following members of the governing Integrated Project Team (IPT) are required to sign. Please annotate signature blocks accordingly.

September 28, 2016:

Signed: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Integrated Project Team Chair Date

Signed: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Project Manager Date:

Signed: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Business Sponsor Date