Desarrollo de Componentes en el Proceso de Generación de una Solución de Software para la Gestión de Nómina en la Universidad Distrital Francisco José de Caldas

Proyecto de Grado

Juan David Lara Rodríguez
William Alfredo Linares Gordillo

Universidad Distrital Francisco José de Caldas
Facultad de Ingeniería
Ingeniería de Sistemas
Proyecto de Grado
Bogotá D.C.
2016

Desarrollo de Componentes en el Proceso de Generación de una Solución de Software para la Gestión de Nómina en la Universidad Distrital Francisco José de Caldas

Proyecto de Grado

Juan David Lara Rodríguez
William Alfredo Linares Gordillo

Proyecto de Grado Pasantía para optar por el título de Ingeniero de Sistemas

Director:
Ing. John Fredy Parra Peña

Universidad Distrital Francisco José de Caldas
Facultad de Ingeniería
Ingeniería de Sistemas
Proyecto de Grado
Bogotá D.C.
2016

Índice general

Índice de figuras

Índice de cuadros

1 INTRODUCCIÓN

La Universidad Distrital Francisco José de Caldas es un ente autónomo de Educación Superior fundada en el año 1948 cuya misión se centra en las funciones básicas de docencia, investigación y extensión y para cumplir a cabalidad con estas funciones, requiere del apoyo de talento humano como docentes, administrativos y contratistas, quienes deben tener una vinculación contractual con la Institución para que puedan ser retribuidos monetariamente por sus servicios.

Cualquiera que sea la vinculación contractual que tenga una persona natural con la Universidad debe efectuarse un proceso periódico de gestión de nómina integral de liquidación de salarios, honorarios, prestaciones y aportes patronales, seguridad social, parafiscales y prestaciones sociales según sea el caso.

Actualmente en la Institución se cuenta con varios procesos de gestión de nómina de acuerdo a los tipos de vinculación contractual existentes apoyados por diferentes herramientas informáticas, que van desde hojas de cálculo hasta soluciones más robustas soportadas en bases de datos oracle e integradas a los sistemas de información misionales de la Universidad, teniendo como común denominador la baja flexibilidad a nuevos requerimientos legales o tributarios del entorno externo o institucional, así como la baja interoperabilidad con otros componentes de software requeridos para efectuar los pagos y causaciones financieras generadas en los períodos fiscales.

El presente documento se realiza desde el enfoque de desarrollador, el cual hace parte de un proyecto mayor que tiene como objetivo analizar, diseñar y construir una solución modular, integral y escalable que permita apoyar los procesos relacionados a la gestión de nóminas de la Universidad Distrital, soportado en tecnologías libres siguiendo el proceso de desarrollo OPENUP/OAS el cual es aplicado en la Oficina Asesora de Sistemas, área donde se desarrollará la presente pasantía y por lo cual nuestra metodología se ajusta a su política de desarrollo, y a partir de dicha política de trabajo desarrollar una parte de la solución de software que genere una única herramienta tecnológica que soporte cualquier tipo de vinculación contractual con la Institución.

2 PLANTEAMIENTO DEL PROBLEMA

El presente proyecto hace parte de un proyecto mayor que busca la obtención de un Sistema de Manejo de Nómina para el uso interno de la Universidad Distrital Francisco José de Caldas, para tal fin se trabajará en colaboración con otros equipos de pasantes, los cuales estarán encargados de áreas como la obtención de requerimientos y la arquitectura de software, a continuación se relaciona una tabla que detalla de mejor manera el ámbito del problema que afecta el actual anteproyecto.

Tabla 2.1: Planteamiento del Problema

El Problema	Proporcionar componentes de software que respondan al diseño realizado por otros equipos y a la gran variedad de procesos presentes en la liquidación de nóminas a las personas naturales que tienen algún vínculo contractual con la Universidad, además sustituir las herramientas informáticas obsoletas y corregir la baja interoperabilidad de dichas herramientas con otros componentes de software proveedores o destinatarios.
Afecta	▪ Las personas naturales que tiene alguna vinculación contractual con la Universidad (Funcionarios, docentes de vinculación especial, Contratistas). ▪ Los procesos de liquidación y pago de nómina. Personal a cargo de los procesos de liquidación y pago en las áreas de la División de Recursos Humanos y la División de Recursos Financieros.

El Impacto del Problema es	▪ No confiabilidad, disponibilidad e integridad de los datos. ▪ Duplicidad de las actividades y tareas y registro de información. ▪ Demoras innecesarias en la liquidación y pago de honorarios o sueldos a funcionarios, contratistas y docentes de vinculación especial. ▪ Traumatismo en los procesos internos de la División de Recursos Financieros y Recursos Humanos. ▪ Revelación inadecuada y poco fidedigna de la información financiera. ▪ Falta de información oportuna para la presentación de informes a los diferentes a entes de control.
Una Solución con exito debería ser	Una solución de software modular, integral y escalable, que permita apoyar los procesos relacionados a la gestión de nóminas de la Universidad Distrital, cualquiera que sea el tipo de vinculación contractual.

8

3 OBJETIVOS

3.1. Objetivo General

Desarrollo de una solución de software modular, integral y escalable para la gestión de nóminas de la Universidad Distrital, tomando como base las especificaciones de casos de uso y la arquitectura definida por el proyecto "*Sistema Integral de Información de Nómina de la Universidad Distrital Francisco José de Caldas*".

3.2. Objetivos Específicos

- Generar prototipos acordes a la arquitectura del sistema empleando un estilo modular, basado en patrones y orientado a servicios que garantice el cumplimiento de las funcionalidades requeridas desde la perspectiva de usuario final.
- Desarrollar de manera paralela al código fuente la documentación técnica requerida para su entendimiento a nivel de la funcionalidad del software.
- Aplicar pruebas funcionales y no funcionales para determinar hasta qué punto el prototipo desarrollado cumple con los requerimientos de los usuarios, con los productos propios del análisis (casos de uso) y con la arquitectura desarrollada para el proyecto.
- Establecer una comunicación continua con los demás miembros del proyecto general, con lo cual se desea gestionar los cambios que pueda llegar a tener cualquier área relacionada con la construcción de la solución de software.

4 JUSTIFICACIÓN

Debido a que la Universidad Distrital no cuenta con un sistema integral, unificado y escalable que permita soportar el proceso de gestión de nómina de las personas naturales que tiene algún vínculo contractual con la Universidad, se hace necesario el análisis, diseño arquitectónico y desarrollo de una solución de software que permita unificar los diferentes tipos nóminas existentes en una sola herramienta web, cuyas características sea una alta interoperabilidad con otros componentes de software, adaptable a los cambios normativos y legales del entorno interno o externo y cumpla requisitos de alta calidad en capacidades de usabilidad, fiabilidad, rendimiento y mantenimiento del software.

La solución integral de software permitirá reducir el tiempo invertido en el proceso de la liquidación de la nómina en cualquier tipo de vinculación contractual en la Universidad, con lo que se facilitarán los pagos oportunos a los funcionarios, docentes y contratistas de la Institución. Por otro lado y por ser un sistema altamente interoperable permitirá lograr una sincronización precisa de datos con otros sistemas proveedores o destinatarios y armonizar la información organizacional.

Adicionalmente es prudente indicar que el actual anteproyecto hace parte de un proyecto más grande que incluye todos los tópicos profesionales propios de la Ingeniería de Sistemas, presentado equipos que trabajan paralelamente en el análisis de requerimientos, y la formulación del diseño arquitectural del macro proyecto que nos involucra a nosotros como desarrolladores, por lo cual el trabajo en equipo será un aspecto importante del trabajo con el cual se busca obtener una herramienta de software que solucione los problemas de nómina mencionados anteriormente, y a manera de consecuencia se obtenga el desarrollo estipulado de una pasantía como desarrolladores en un equipo profesional y bajo metodologías propias del campo.

5 MARCO TEÓRICO

5.1. El Proceso OPENUP/OAS

En el marco de la resolución 461 de Rectoría del 29 de Julio de 2011, la Universidad Distrital Francisco José de Caldas avaló el método de desarrollo OPENUP/OAS, como el marco de trabajo institucional en el análisis, diseño, desarrollo e implementación de productos de software al interior de la Universidad[1]. A continuación se describen sus principales directrices y fundamentos.

5.1.1. Generalidades

El proceso openUP/OAS es un método de trabajo que involucra un conjunto mínimo de prácticas tendientes a guiar a un equipo de trabajo pequeño en el análisis, diseño, desarrollo y despliegue de un producto de software[2]. Los objetivos que persiguen son:

- Promover la colaboración y compartir conocimientos alineando intereses del equipo de trabajo y los usuarios.
- Ayudar al equipo a enfocarse en la arquitectura de forma rápida; de tal forma que se minimicen los riesgos y se organice el desarrollo.
- Ayudar al equipo a balancear prioridades en conflicto para maximizar el valor obtenido por los interesados en el proyecto.
- Ayudar al equipo en la evolución continua del producto para obtener retroalimentación continua y fomentar el mejoramiento.
- Permitir a los administradores del proyecto realizar seguimientos a las avances basados en metas e indicadores.
- Permitir que los integrantes del equipo entiendan rápidamente como realizar el trabajo para alcanzar los objetivos y metas proyectadas.

[1] BAHAMON CALDERON, Inocencio. *Resolución de Rectoría N° 461*, SISGRAL, p. 1, 2011, `http://sgral.udistrital.edu.co/xdata/rec/res_2011-461.pdf` [Consulta: Jueves, 11 de Septiembre de 2015]

[2] OFICINA ASESORA DE SISTEMAS, OAS. *Guía Rápida Proceso de Desarrollo OPENUP/OAS*, Generalidades, Versión: 0.0.0.7, pág. 2, 2011, `https://www.udistrital.edu.co/files/dependencias/oas/GuiaRapidaOpenUPOAS.pdf` [Consulta: Jueves, 11 de Septiembre de 2015]

Los principios en que se enmarca el método de trabajo OPENUP/OAS son [3]:

a. Conocer a los Interesados: Se deben identificar, conocer a los grupos de interés y trabajar de cerca con ellos para asegurarse que sus necesidades son claramente definidas e incrementalmente satisfechas a medida que se evoluciona en el desarrollo de la solución. Debe mantenerse una comunicación abierta y frecuente además de una colaboración entre ellos y el equipo de trabajo.

b. Separar el Problema de la Solución: Se debe estar seguro que se conoce el problema (o una parte de él) antes de definir una solución (o una parte de ella). Al separar claramente el problema (que necesita el cliente - no que necesita el equipo de desarrollo) de la solución (el sistema que tiene que hacer), es fácil mantener un enfoque y encontrar vías alternativas para solucionar el problema.

c. Crear un conocimiento compartido del dominio: Se debe fomentar un ambiente de intercambio y trabajo en el que todos los involucrados puedan obtener constantemente la información adecuada para lograr tener una visión compartida de lo que se debe hacer, el por qué hacerlo y como se está haciendo.

d. Usar escenarios y casos de uso para capturar requerimientos: Hacer uso de escenarios y casos de uso para capturar los requerimientos funcionales del sistema permiten que los interesados alcancen rápidamente un consenso acerca de sus necesidades e intereses.

e. Establecer y mantener contratos de prioridades: Se deben priorizar los requisitos y requerimientos de implementación basado en un trabajo continuo con los grupos de interés y tomar decisiones que lleven a que el sistema siempre incremente los beneficios ofrecidos y reduzca los riesgos.

f. Realizar negociaciones que maximicen el beneficio obtenido: Las negociaciones costo beneficio dentro del proyecto no pueden ser independientes de la arquitectura. Los requisitos y requerimientos establecen los beneficios que se deben alcanzar al implementar el sistema mientras que la arquitectura es una medida base para calcular el costo del mismo. El costo asociado con un beneficio puede influenciar en gran medida la percepción del usuario acerca del valor real obtenido.

g. Gestionar el entorno: El cambio es inevitable, y aunque presenta oportunidades para mejorar los beneficios dados a los grupos de interés, un entorno incontrolado de cambios fácilmente decantará en sistemas deficientes, sobredimensionados y que no satisfacen las necesidades reales de los clientes. Se debe gestionar los cambios manteniendo contratos específicos con los grupos de interés.

h. Conocer cuándo se debe parar: Sobrerecargar de características un sistema no sólo es una pérdida de tiempo y recursos sino que conduce a sistemas innecesariamente complejos. El desarrollo debe parar cuando la calidad esperada del sistema se alcanza.

[3]OFICINA ASESORA DE SISTEMAS, OAS. *Guía Rápida Proceso de Desarrollo OPENUP/OAS*, Generalidades, Versión: 0.0.0.7, pág. 2, 2011, `https://www.udistrital.edu.co/files/dependencias/oas/GuiaRapidaOpenUPOAS.pdf` [Consulta: Martes, 9 de Septiembre de 2015]

i. Mantenga un entendimiento común: Sea proactivo comunicando y compartiendo información con los participantes del proyecto y no asuma que todos y cada uno encontrarán justo lo que ellos necesitan saber o que cada persona tiene la misma comprensión del proyecto que todos los demás.

j. Aprender continuamente: Desarrolle continuamente sus habilidades técnicas e interpersonales, aprenda de los ejemplos de sus colegas, aproveche la oportunidad, tanto de ser un estudiante de sus colegas, así como maestro de ellos. Siempre incremente su habilidad personal para sobrellevar su propio antagonismo hacia otros miembros del equipo.

k. Organice alrededor de la arquitectura: La comunicación entre los miembros del equipo empieza a ser compleja incrementalmente. Por consiguiente, organice el equipo alrededor de la arquitectura, el vocabulario y el modelo mental compartido del sistema.

l. Desarrolle su proyecto en iteraciones: Divida su proyecto en una serie de iteraciones encajadas en el tiempo y planee su proyecto iterativamente. Esta estrategia iterativa lo habilita para entregar capacidades incrementalmente, como un conjunto ejecutable, subconjunto utilizable de requisitos y requerimientos probados e implementados, que pueden ser evaluados por los interesados al final de cada iteración.

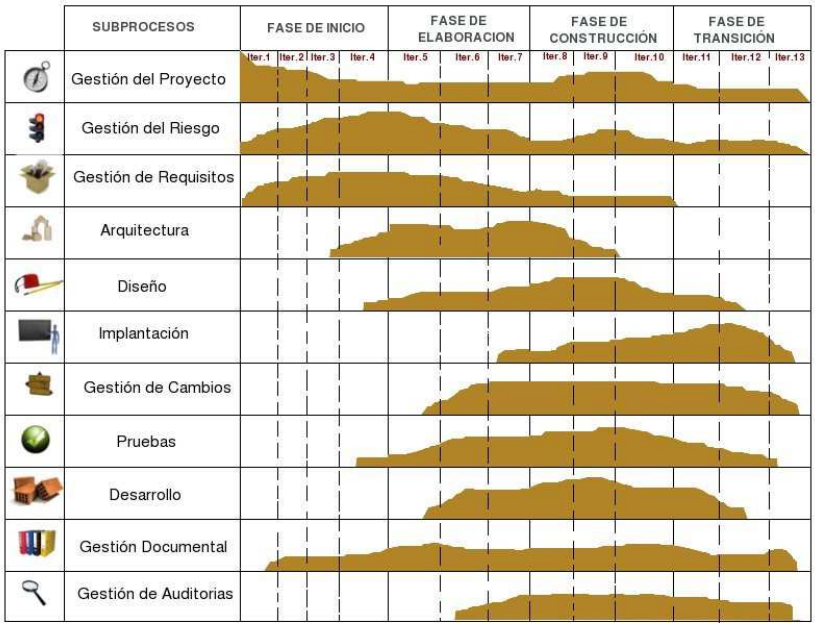
m. Gestione los riesgos: Ataque tempranamente los riesgos que atacarán el proyecto. Continuamente identifique y priorice los riesgos y entonces idee estrategias para mitigarlos.

n. Adopte y gestione el cambio: Adoptar los cambios ayuda a construir un sistema que se encamina a las necesidades de los interesados y manejar los cambios permite reducir costos y mejorar la predicción de estos cambios. Los cambios hechos tempranamente en el proyecto se pueden hacer usualmente a bajo costo. A medida que usted avanza en el proyecto, los cambios pueden empezar a incrementarse en términos de costos.

o. Mida el progreso objetivamente: Si no conoce objetivamente cómo su proyecto está progresando, no sabe si éste falla o tiene éxito. La incertidumbre y los cambios a un proyecto de software en progreso dificultan medirlo objetivamente, en tanto que las personas tienen una habilidad muy asombrosa para creer que todo esta bien ante la catástrofe.

5.1.2. El Método de Trabajo

El OPENUP/OAS es un proceso iterativo e incremental que se distribuyen a través de cuatro fases: Inicio, Elaboración, Construcción y Transición. En las cuales se desarrollan transversalmente una serie de subprocesos entendiéndose estos últimos como un conjunto de actividades, personas (Roles), prácticas (Guías) y productos de trabajo (Artefactos) que orientan el desarrollo de software a través del tiempo.



Fuente: Tomado de [6]

Figura 5.1: Método de Trabajo OPENUP/OAS.

Cada fase puede tener tantas iteraciones como se requiera, dependiendo del grado de complejidad y desconocimiento del dominio, la tecnología a ser usada, la complejidad arquitectónica y el tamaño del proyecto, por nombrar algunos factores[4].

5.1.2.1. Fases del Proceso OPENUP/OAS

Fase de Inicio: Primera fase del proceso, donde los interesados (stakeholders) y los integrantes del equipo de desarrollo, colaboran para determinar el ámbito del proyecto, sus objetivos y determinar si el proyecto es viable. Las iteraciones de esta fase enfocan el esfuerzo de trabajo en las siguientes actividades y resultados:

[4]OFICINA ASESORA DE SISTEMAS, OAS. *Guía Rápida Proceso de Desarrollo OPENUP/OAS*, Fases OPENUP/OAS, Versión: 0.0.0.7, págs. 4-5, 2011, `https://www.udistrital.edu.co/files/dependencias/oas/GuiaRapidaOpenUPOAS.pdf` [Consulta: Martes, 9 de Septiembre de 2015]

16

Actividad	Resultados
Iniciar el proyecto	* Elaborar el documento Visión * Elaborar el Plan General del Proyecto * Elaborar el documento de Análisis de riesgo
Planear y gestionar la iteración	* Elaborar el Plan de Iteración * Elaborar el documento de evaluación de la iteración * Elaborar el documento de valoración de resultados de la iteración
Identificar y refinar los requerimientos y requisitos	* Elaborar la especificación de casos de uso * Elaborar el documento de requisitos de soporte * Elaborar el documento casos de prueba
Llegar a un acuerdo sobre el enfoque técnico	* Elaborar el documento Bloc de Notas de la Arquitectura

Fuente: Tomado de [6]

Tabla 5.1: Actividades Fase de Inicio.

Al final de esta fase, como mínimo, el proyecto:

- Ha definido el ámbito
- Tiene un estimado inicial de los costos y el cronograma
- Ha definido y priorizado un conjunto inicial de requerimientos funcionales y no funcionales
- Ha identificado un conjunto de riesgos y haya propuesto las estrategias de mitigación.
- Ha identificado un conjunto de interesados.
- Ha creado un bosquejo de arquitectura.

Fase de Elaboración: La segunda fase dentro del ciclo de vida del proyecto. En ella los riesgos significativos que influyen en la arquitectura son identificados y considerados. En esta fase:

- Se obtiene un entendimiento más detallado de los requerimientos y requisitos
- Se diseña, implementa válida y establece la línea base de la arquitectura.
- Se mitigan los riesgos esenciales.
- Se produce un cronograma detallado.
- Se realiza una mejor estimación de costos.

Las iteraciones de esta fase enfocan el esfuerzo de trabajo en las siguientes actividades y resultados:

Actividad	**Tareas/Resultados**
Planear y gestionar la iteración	- Elaborar el Plan de Iteración - Elaborar el documento de evaluación de la iteración - Elaborar el documento de valoración de resultados de la iteración
Identificar y refinar los requerimientos	- Actualizar, depurar y aumentar el contenido de la especificación de casos de uso - Actualizar, depurar y aumentar el contenido del documento de Requerimientos de soporte - Actualizar, depurar y aumentar el contenido del documento Casos de prueba
Desarrollar la arquitectura	Agregar las vistas de arquitectura al documento Bloc de Notas de la Arquitectura
Desarrollar un incremento en la solución	- Actualizar, depurar y aumentar el contenido del documento Especificación de Diseño - Actualizar, depurar y aumentar el contenido del documento Pruebas efectuadas por el Realizador - Obtener el código fuente que realiza uno o varios elementos de diseño - Elaboración de una Construcción del Sistema que integre nuevos elementos (componentes desarrollados, clases, etc) - Elaborar el artefacto Registro de Pruebas que contenga los resultados de la ejecución de las pruebas hechas por el realizador.
Probar la Solución Construida	Elaborar el artefacto Script de Prueba Elaborar el artefacto Registro de Pruebas que contenga los resultados de la ejecución de las pruebas.
Gestionar las peticiones de cambio	Actualizar, depurar y aumentar el contenido del documento Lista de Unidades de Trabajo.

Fuente: Tomado de [6]

Tabla 5.2: Actividades Fase de Elaboración.

Fase de Construcción: Esta es la tercera fase del proceso, se enfoca en detallar los requisitos y requerimientos, diseñar, implementar y probar el grueso del software y completar el desarrollo del sistema basado en la arquitectura.

18

Se describen los requisitos y requerimientos restantes Se completan en detalles los diseños, la implementación y las pruebas del software. Se libera la primera versión operativa del software (beta) del sistema.

Las actividades de esta fase son:

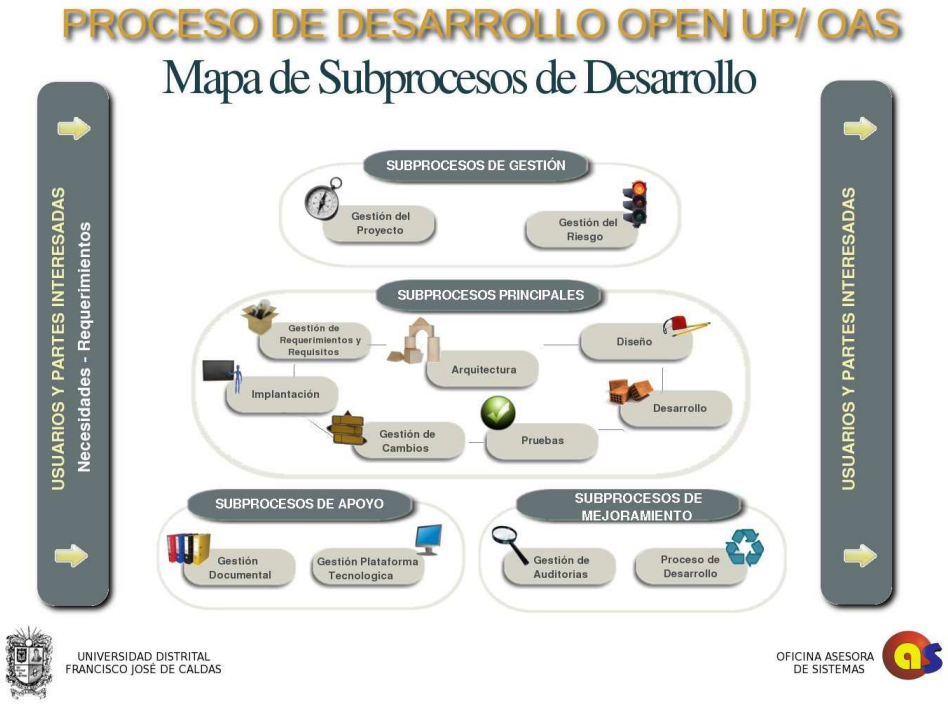
1. Planificación y gestión de la iteración
2. Identificar y refinar requisitos y requerimientos
3. Desarrollar un incremento de solución
4. Probar la solución construida

Fase de Transición: Es la cuarta fase del proceso. Se enfoca en la transición del producto de software a la plataforma tecnológica del cliente logrando que los interesados convengan que el desarrollo del producto cumple con los requerimientos planteados. Los objetivos de esta fase son lograr:

- La prueba beta valida que satisfaga las expectativas del usuario. Esto típicamente requiere algunas actividades de afinamiento, tales como depuración de errores y mejora del desempeño y la usabilidad.
- El consentimiento de los interesados en que el desarrollo está completo. Esto puede involucrar varios niveles de pruebas para la aceptación del producto, incluyendo pruebas formales e informales y pruebas beta.
- Mejorar el desempeño en futuros proyectos a través de lecciones aprendidas.
- Documentar las lecciones aprendidas y mejorar el ambiente de los procesos y las Herramientas para el proyecto.

5.1.2.2. Subprocesos OPENUP/OAS

Como se había señalado anteriormente un subproceso es un conjunto de actividades desarrolladas por personas con unos roles determinados, las cuales se guían por medio de una serie de prácticas o guías para obtener unos productos de trabajo denominados Artefactos y que permiten cumplir direccionar las fases y actividades propuestas en las cuatro fases del proceso de desarrollo de software openup/oas. Estos subprocesos se relacionan entre sí, siendo unos entradas o insumos iniciales para que otros subprocesos se puedan desarrollar.



Fuente: Tomado de [6]

Figura 5.2: Mapa de Subprocesos OPENUP/OAS.

Subproceso	Objetivo	Artefactos de Salida
Gestión de Requerimientos y Requisitos	Recolectar, analizar, aprobar y seguir la evolución de los requerimientos funcionales del Cliente o interesado y los requisitos del software a través de la vida del producto y/o servicio.	- Visión - Especificaciones de Casos de Uso - Glosario -Requisitos de Soporte - Actas de Trabajo - Listado de requerimientos funcionales y no funcionales
Gestión del Proyecto	Planear, ejecutar, controlar y socializar las actividades y resultados de un proyecto de software.	Plan General del proyecto Plan de Iteración Cierre de iteración Lista de Unidades de trabajo
Gestión del Riesgo	Identificación, valoración, relevancia, prevención, mitigación, control y respuesta a posibles riesgos que se generen en un proyecto de software.	Plan y tratamiento de Riesgos

20

Arquitectura y diseño	Transformar los requerimientos y requisitos significativos en una arquitectura que describa su estructura e identifique los componentes del software.	Diagramas de Clases, Diagrama de componentes Diagramas de Secuencia Diagramas de Colaboración Arquitectura de Datos. Bloc de Notas de la Arquitectura
Desarrollo	Implementar una solución técnica que cumpla con la arquitectura definida y soporte los requerimientos de los grupos interesados.	Código Fuente
Gestión de Pruebas	Diseñar, implementar, ejecutar y evaluar pruebas en cada uno de los componentes desarrollados.	Casos de Prueba Resultados casos de prueba
Gestión de Cambios	Registrar, revisar y llevar a cabo solicitudes de cambios generadas en un proceso de desarrollo de software.	Control de Cambios
Implantación	Planificar y llevar a cabo la producción de una solución de software mediante el alineamiento de las necesidades de capacitación de los usuarios y el desarrollo de pruebas de funcionamiento.	Plan de despliegue socialización, capacitación o acompañamiento

Fuente: Tomado de [6]

Tabla 5.3: Subprocesos OPENUP/OAS.

5.1.2.3. Roles OPENUP/OAS

Los productos de software los crean personas con diferentes intereses y competencias. Un ambiente de grupo saludable potencia la colaboración efectiva requiriendo una cultura compartida que fomente la creatividad y el cambio positivo.

Los roles son el rostro humano del proceso de desarrollo de software. Dependiendo del número de personas que conforman el equipo de trabajo y las condiciones del proyecto una persona puede asumir uno o varios roles.

Fuente: Tomado de [6]

Figura 5.3: Estructura del Equipo de Trabajo OPENUP/OAS.

Rol	Función Principal
Director del Proyecto	Este rol garantiza la continuidad del proyecto al gestionar los recursos necesarios y mantener el interés institucional en el proyecto.
Jefe de Proyecto	Este rol se encarga de la supervisión y dirección directa de las actividades y resultados de cada uno de los miembros del equipo de desarrollo.
Líder del Proyecto	Lidera la planeación del proyecto, coordina interacciones con los interesados y conserva el equipo del proyecto enfocado en alcanzar los objetivos del proyecto
Analista	Realizar tareas de relevamiento, análisis y diseño de los requerimientos y requisitos en el proyecto.
Arquitecto	Responsable de diseñar la arquitectura del software, la cual incluye tomar las principales decisiones técnicas que condicionan globalmente el diseño y la implementación del proyecto.
Realizador o desarrollador	Es responsable de desarrollar una parte del sistema, incluyendo diseñar esta, para que se ajuste a la arquitectura
Inspector de Pruebas	Identificar, definir, implementar y dirigir las pruebas necesarias, así como verificar y analizar sus resultados.

Fuente: Tomado de [6]

Tabla 5.4: Roles OPENUP/OAS.

22

5.1.2.4. Artefactos del Proceso OPENUP/OAS

Los Artefactos relacionados son:

Nombre	Descripción
Plan General del Proyecto	Este artefacto define los parámetros para realizar el direccionamiento y seguimiento al proyecto. Especifica los objetivos de alto nivel de las iteraciones y sus correspondientes hitos.
Plan de Iteración	Comunica los objetivos, la asignación de tareas y los criterios de evaluación para una iteración dada.
Unidades de Trabajo Diarias	Contiene una lista de los trabajos programados diariamente y que responden a los objetivos definidos en la Iteración y en el proyecto
Cierre de iteración	Este documento registra los resultados de una iteración
Visión	Contiene los lineamientos de los requerimientos nucleares visionados del sistema, especificado las necesidades y características claves de los Interesados.
Requisitos de soporte	Captura requisitos en el ámbito del sistema que no hayan sido capturados en escenarios o casos de uso, incluye requisitos sobre atributos de calidad y de desempeño global.
Especificación de Casos de Uso	Captura la secuencia de acciones que un sistema realiza y que genera un resultado observable que es de valor para aquellos que interactúan con el sistema.
Glosario	Este artefacto define términos importantes usados en el proyecto
Listado de requerimientos y requisitos	En este documento se registran los requerimientos y requisitos que surjan a lo largo del proyecto, y sirve para priorizar y organizar las tareas, objetivos y metas del mismo.
Acta de Trabajo	Registra los acuerdos o compromisos definidos entre los interesados y el equipo de desarrollo
Plan de Riesgos	Contiene la identificación, valoración, relevancia, prevención, mitigación, control y respuesta a posibles riesgos que se generen en un proyecto de software.
Bloc de notas de la Arquitectura	Contiene las decisiones, razonamientos, asunciones, explicaciones e implicaciones sobre la arquitectura en formación.
Documento de Diseño	Artefacto documenta las especificaciones técnicas en cuanto al diseño del software y se complementa con diagramas de clases, diagramas de colaboración, diagramas de secuencia entre otros.

Control de Cambios	Este artefacto es utilizado para documentar las solicitudes de cambio de los diferentes subprocesos que surgen al interior del proyecto por parte de los interesados o miembros del equipo del proyecto.
Caso de prueba	Son la especificación de un conjunto de pruebas de entradas, condiciones de ejecución y resultados esperados, los cuales son identificados para el propósito de realizar una evaluación de una aspecto particular en un escenario específico.
Registro de Pruebas	Este artefacto recolecta los resultados de la ejecución de una o más pruebas en un ciclo completo de pruebas.

Fuente: Tomado de [6]

Tabla 5.5: Artefactos OPENUP/OAS.

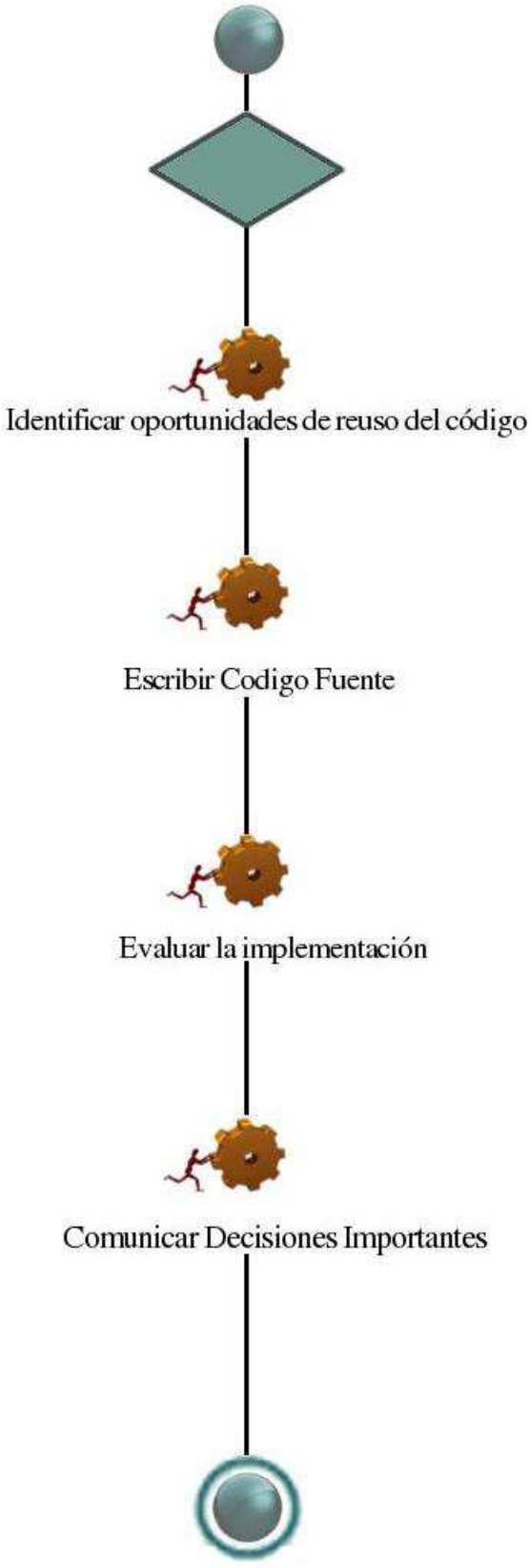
5.2. El Rol de Desarrollo OPENUP/OAS: Funciones y Tareas

En el marco del actual anteproyecto, el énfasis central del desarrollo de la pasantía está en el **Rol de Desarrollador**, por consiguiente a continuación se desarrolla todo el Marco Teórico relacionado con esta área en el ámbito del Proceso de Desarrollo OPENUP/OAS.

5.2.1. Subproceso de Desarrollo

A continuación se puede obervar un grafico donde se permite apreciar las labores comprendidas en el desarrollo.

24



Fuente: Tomado de [5]

Figura 5.4: Subproceso de Desarrollo OPENUP/OAS.

5.2.2. Actividades del Desarrollo

Identificar las oportunidades de reutilización de código: Identificar código existente u otros elementos de puesta en funcionamiento que puedan ser utilizados. Un manejo adecuado del diseño siempre será útil ya que las oportunidades de reciclaje son más evidentes cuando se tiene una visión clara de lo que debe tener la solución.

Se debe tener siempre en consideración elementos de software libre o de propiedad de la institución manteniendo un especial cuidado por el respeto de las normas y legislación relacionada con el derecho de autor. El responsable de esta actividad es siempre el desarrollador.

Transformar el diseño en puesta en funcionamiento: si se usan herramientas de modelado se puede obtener gran parte del código al transformar los modelos, pero aun así siempre se deben tener tareas de codificación. Esta tarea es inherente del diseñador junto con el programador.

Escribir el código fuente: Escribir el código fuente para hacer una puesta en funcionamiento que cumpla con el diseño y el comportamiento esperado. Se debe procurar la reutilización o generación a partir de modelos pero aún en estos casos se requerirá de algún tipo de programación. Adicionalmente en algunos casos se deben examinar los requerimientos para ver si son acordes al modelo o si se debe aplicar una programación directa, siempre se debe apuntar a piezas reutilizables de código que estén en constante evolución. Esta actividad es propia del programador y produce como producto el código fuente del proyecto.

Comunicar decisiones importantes: Comunicar rápidamente el impacto de cambios no esperados en el diseño o los requerimientos. Los problemas y restricciones que no se hayan tenido en consideración deben ser comunicados al equipo de desarrollo.[5]

5.3. La Nómina: Estructura, Partes y Elementos Básicos.

La nómina es el sistema contable de la empresa para mantener un registro con el salario, cargo, deducciones, así como otros gastos y rendimientos que genera cada uno de sus empleados. La nómina es el documento que se entrega mensualmente a todos los trabajadores en el que aparece el detalle del salario que recibe, junto con las deducciones que se le practican de dicho salario, bien sea por descuentos obligatorios marcados por la legislación vigente, bien sea por otro tipo de descuentos como anticipos, o deducciones para seguros de salud[6].

[5]OFICINA ASESORA DE SISTEMAS, OAS. *Cápitulo 9, Desarrollo de la Solución*, Proceso de Desarrollo OPENUP/OAS, págs. 4-6, 2011, `http://www.udistrital.edu.co:8080/documents/276352/356568/Cap9Desarrollo.pdf` [Consulta: Miércoles, 10 de Septiembre de 2015]

[6]TRECET, José. *Financialred, Red de Blogs Especializados en Finanzas, Economía y Bolsa*, La Nómina, `http://www.finanzzas.com/la-nomina` [Consulta: Martes, 1 de Septiembre de 2015]

26

El formato estándar de una nómina está regulado por la legislación vigente y se marca una estructura y contenido mínimo que se debe respetar en todo caso. Este contenido mínimo de la nómina debe incluir al menos:

- Datos identificativos de la empresa, dirección del centro de trabajo y código cuenta cotización en el que está el trabajador incluido.
- Datos básicos del trabajador, tipo de contrato, categoría, antigüedad en la empresa.
- Periodo de liquidación al que corresponde dicha nómina.
- Detalle de las percepciones salariales y extrasalariales que componen la retribución bruta del trabajador.
- Detalle de las deducciones que se le practican al salario bruto, bien marcadas por la legislación vigente, bien por otro tipo de deducciones que haya que aplicarle a la nómina, como anticipos o, embargos en la nómina.
- Líquido a percibir, dado que la nómina tiene consideración de documento acreditativo del pago de salarios cerrando los pagos pendientes al trabajador para el periodo estipulado.
- Detalle de las bases de cotización de la nómina
- Lugar de emisión y firma y sello por la empresa y trabajador.

Remarcar que aunque estos elementos sean mínimos, existen algunas excepciones como el caso de la firma del trabajador. No es necesaria dicha firma si el pago de la nómina se ha realizado por medios bancarios que puedan demostrar la percepción salarial por parte del trabajador.

5.3.1. Retribución Bruta: Conceptos que Conforman el Salario Bruto

La suma total de todos los conceptos que hay que abonar al trabajador dan origen a la retribución bruta mensual. La nómina tiene por norma general periodos de liquidación mensuales con las siguientes excepciones:

- Entrada o salida del trabajador de la empresa, sin que estas fechas correspondan con el mes natural.
- Nóminas de paga extra.

Dentro de las percepciones que se suman para dar lugar al salario bruto tenemos dos grupos diferenciados:

Percepciones salariales, que lo conforman todos aquellos conceptos que están fijados por el convenio colectivo de aplicación en la empresa

Percepciones extrasalariales, en el que se incluyen conceptos que no tienen la consideración pura de salario, como pueden ser dietas, gastos de locomoción o pluses por retribuciones en especie como el plus por desgaste de herramientas

5.3.1.1. Percepciones salariales

Tal y como hemos definido anteriormente, la cantidad de conceptos que se pueden incluir dentro de las percepciones salariales es muy amplia y no es genérica, porque cada empresa tiene un convenio colectivo distinto y cada uno de estos textos incluyen una distribución distinta de los pagos que hay que realizar.

Normalmente, nos podemos encontrar con conceptos como:

- Salario Base, que corresponde al pago mensual mínimo que se tiene que realizar para un trabajador dentro de la categoría en la que está encuadrado.
- Complementos; como cantidades adicionales que complementan al salario base en el caso de que se cumplan unos determinados requisitos de productividad, cumplimiento horario, productividad, trabajos penosos, nocturnos en días festivos.
- Parte proporcional de paga extra. Todos los convenios colectivos regulan el pago de una o varias pagas extras. Normalmente se abonan entre dos y cuatro pagas extras anuales y puede darse el caso de que la paga extra se encuentre prorrateada.

5.3.1.2. Percepciones extrasalariales

En este apartado se engloban todos aquellos conceptos que no constituyen un pago directo por el trabajo desempeñado, sino que cubren otro tipo de gastos que se le puedan originar a un trabajador. Por ejemplo, tienen esta consideración los pagos por dietas, desgaste de herramientas o cantidades indemnizatorias por cambio de centro de trabajo o de localidad.

5.3.2. Descuentos en la Nómina

Tal y como hemos explicado anteriormente, en la nómina nos podemos encontrar dos tipos de descuentos diferentes, bien los descuentos obligatorios por ley o bien los descuentos que se deban aplicar por cualquier otro tipo de normativas.

En el caso de los descuentos por ley, tenemos dos grupos de deducciones diferentes, que se destinan al pago de la seguridad social a cargo del trabajador, como los pagos a cuenta que el propio trabajador tiene que realizar por impuestos u otros conceptos.

28

5.4. Estructura Legal de la Nómina de la Universidad Distrital

5.4.1. Funcionarios

5.4.1.1. Empleados Públicos Administrativos

Se estipulan las siguientes consideraciones de nómina[7]:

Ley 4 de 1992 "Mediante la cual se señalan las normas, objetivos y criterios que debe observar elGobierno Nacional para la fijación del régimen salarial y prestacional de los empleados públicos, de los miembros del Congreso Nacional y De la Fuerza Pública y para la fijación de las prestaciones sociales de los Trabajadores Oficiales y se dictan otras disposiciones, de conformidad con lo establecido en el artículo 150, numeral 19, literales e) y f) de la Constitución Política"

Decreto 1042 de 1978 "Por el cual se establece el sistema de nomenclatura y clasificación de los empleos de los ministerios, departamentos administrativos, superintendencias, establecimientos públicos y unidades administrativas especiales del orden nacional, se fijan las escalas de remuneración correspondientes a dichos empleos y se dictan otras disposiciones"

Ley 30 de 1992 "Por el cual se organiza el servicio público de la Educación Superior".

Acuerdo No.003 de 1997 "Por el cual se expide el Estatuto General de la universidad Distrital Francisco José de Caldas"

Acuerdo No.008 de 1983 "Por el cual se determina salarios y otros derechos a los empleados públicos administrativos"

Acuerdo 003 de 2003 "Por medio del cual se fija el régimen salarial para los empleados públicos de la Universidad"

Resolución 004 de 2003 "Por la cual se establecen las escalas salariales de las distintas categorías de empleos de la Universidad Distrital Francisco José de Caldas y se dictan otras disposiciones"

5.4.1.2. Docentes Acuerdo 003 de 1997

Para las personas que se rigen por este acuerdo se determina[8]:

[7] OFICINA ASESORA DE SISTEMAS, OAS. *Levantamiento de la primera versión del proceso de parametrización y liquidación de nómina soportado en el aplicativo de nómina de funcionarios y pensionados de la Universidad Distrital*, Régimen Salarial, Empleados Públicos Administrativos, págs. 10-21, 2013.

[8] OFICINA ASESORA DE SISTEMAS, OAS. *Levantamiento de la primera versión del proceso de parametrización y liquidación de nómina soportado en el aplicativo de nómina de funcionarios y pensionados de la Universidad Distrital*, Régimen Salarial, Docentes Acuerdo 003 de 1997, págs. 22-25, 2013.

Ley 4 de 1992 "Mediante la cual se señalan las normas, objetivos y criterios que debe observar el Gobierno Nacional para la fijación del régimen salarial y prestacional de los empleados públicos, de los miembros del Congreso Nacional y de la Fuerza Pública y para la fijación de las prestaciones sociales de los Trabajadores Oficiales y se dictan otras disposiciones, de conformidad con lo establecido en el artículo 150, numeral 19, literales e) y f) de la Constitución Política"

Decreto 1042 de 1978 "Por el cual se establece el sistema de nomenclatura y clasificación de los empleos de los ministerios, departamentos administrativos, superintendencias, establecimientos públicos y unidades administrativas especiales del orden nacional, se fijan las escalas de remuneración correspondientes a dichos empleos y se dictan otras disposiciones"

Ley 30 de 1992 "Por el cual se organiza el servicio público de la Educación Superior".

Acuerdo No.003 de 1997 "Por el cual se expide el Estatuto General de la universidad Distrital Francisco José de Caldas"

Acuerdo No.008 de 1983 "Por el cual se determina salarios y otros derechos a los empleados públicos administrativos"

Acuerdo 003 de 2003 "Por medio del cual se fija el régimen salarial para los empleados públicos de la Universidad"

Resolución 004 de 2003 "Por la cual se establecen las escalas salariales de las distintas categorías de empleos de la Universidad Distrital Francisco José de Caldas y se dictan otras disposiciones" El régimen salarial de los docentes de la Universidad distrital Francisco José de Caldas está determinada por aquellos que fueron vinculados en vigencia del Acuerdo No.003 de 1997 (Salario fijo durante el año) del Consejo Superior Universitario.

5.4.1.3. Docentes Decreto 1279 de 2002

Para las personas que se rigen por este acuerdo se determina[9]:

Ley 4 de 1992 "Mediante la cual se señalan las normas, objetivos y criterios que debe observar el Gobierno Nacional para la fijación del régimen salarial y prestacional de los empleados públicos, de los miembros del Congreso Nacional y de la Fuerza Pública y para la fijación de las prestaciones sociales de los Trabajadores Oficiales y se dictan otras disposiciones, de conformidad con lo establecido en el artículo 150, numeral 19, literales e) y f) de la Constitución Política"

Ley 30 de 1992 "Por el cual se organiza el servicio público de la Educación Superior".

Decreto 1279 de 2002 "Por el cual se establece el régimen salarial y prestacional de los docentes de las Universidades Estatales"

[9]OFICINA ASESORA DE SISTEMAS, OAS. *Levantamiento de la primera versión del proceso de parametrización y liquidación de nómina soportado en el aplicativo de nómina de funcionarios y pensionados de la Universidad Distrital*, Régimen Salarial, Docentes Decreto 1279 de 2002, págs. 26-39, 2013.

30

5.4.1.4. Liquidación de Mesada Pensional

Para este aspecto se tienen en cuenta las siguientes reglamentaciones[10]:

Ley 69 de 1923 "Por el cual se fija el personal de una oficinas de Hacienda y se adoptan algunas disposiciones fiscales".

Decreto No.0277 de 1958 "Por el cual se establece el régimen jurídico de las Universidades oficiales y departamentales".

Decreto 3135 de 1968 "Por el cual se prevé la integración de la seguridad social entre el sector público y el privado y se regula el régimen prestacional de los empleados públicos y trabajadores oficiales.

Ley 1848 de 1969 "por el cual se reglamenta el decreto 3135 de 1968"

Acuerdo No.003 de 1973 "Por medio del cual se reforma el Estatuto de Profesores de la Universidad Distrital Francisco José de Caldas"

Ley 4 de 1976 "Por la cual se dictan normas sobre materia pensional de los sectores público, oficial, semioficial y privado y se dictan otras disposiciones.

Ley 71 de 1988 "Por la cual se expiden normas sobre pensiones y se dictan otras disposiciones"

Acuerdo No.24 de 1989 "Por el cual se normaliza el procedimiento de liquidación de prestaciones sociales para los empleados públicos docentes y se fijan otros derechos salariales"

Decreto No.1444 de 1992 "Por el cual se dictan disposiciones en materia salarial y prestacional para los empleados públicos docentes de las Universidades Públicas del Orden Nacional"

Convenciones Colectivas

Ley 100 de 1993 "Por la cual se crea el sistema de seguridad social integral y se dictan otras disposiciones"

Concepto No.732 del 1995, Sala de Consulta y Servicio Civil del Consejo de Estado.

Laudos Arbitrales

Decreto 1919 de 2002 "Por el cual se fija el Régimen de prestaciones sociales para los empleados públicos y se regula el régimen mínimo prestacional de los trabajadores oficiales del nivel territorial".

Ley 1066 de 2006 "Por la cual se dictan normas para la normalización de la cartera pública y se dictan otras disposiciones".

[10] OFICINA ASESORA DE SISTEMAS, OAS. *Levantamiento de la primera versión del proceso de parametrización y liquidación de nómina soportado en el aplicativo de nómina de funcionarios y pensionados de la Universidad Distrital*, Régimen Salarial, Liquidación de Mesada Pensional, págs. 40-49, 2013.

5.4.1.5. Trabajadores Oficiales

Para los trabajadores oficiales la nómina se reglamenta por[11]:

Ley 4 de 1992 "Mediante la cual se señalan las normas, objetivos y criterios que debe observar el Gobierno Nacional para la fijación del régimen salarial y prestacional de los empleados públicos, de los miembros del Congreso Nacional y de la Fuerza Pública y para la fijación de las prestaciones sociales de los Trabajadores Oficiales y se dictan otras disposiciones, de conformidad con lo establecido en el artículo 150, numeral 19, literales e) y f) de la Constitución Política"

Convenciones Colectivas de Trabajo desde los años 1974 hasta la presente Vigencia

Acuerdo 010 de 1986 "Por la cual se expide la planta de personal de la universidad Distrital que regirá a partir del 1 Enero de 1986"

Acuerdo 041 de 1988 "Por el cual se aprueba una reclasificación de trabajadores oficiales y se modifica la planta de personal"

Ley 30 de 1992 "Por el cual se organiza el servicio de la educación superior" Posteriormente, con la expedición del Acuerdo 003 de 1997 en su artículo 49, se precisa que ".... las personas que desempeñan labores de aseo, mantenimiento y jardinería, son trabajadores oficiales".

5.4.2. Vinculación Especial

El marco jurídico para este tipo de vinculación está establecido desde la Ley 30 de .006 del 1o. Agosto del 2001)"Por el cual se establece y se fija el límite máximo de cupos para cada una de las Facultades Académicas en la contratación de docentes de VINCULACIÓN ESPECIAL, que prestan servicios en los Programas de Pregrado en la Universidad Distrital Francisco José de Caldas"[12]:

- Acuerdo No.07 del 28 de Diciembre de 2001"Por el cual se modifica el Acuerdo 005 de Julio 27 de 2001"
- Acuerdo No.08 del 28 de Diciembre de 2001 "Por el cual se modifica un artículo y se añaden dos parágrafos al Acuerdo 006 de Agosto 1 del 2001"
- Acuerdo No.11 del 15 de Noviembre de 2002 "Por el cual se expide el Estatuto Docente de Carrera de la Universidad Distrital Francisco José de Caldas"
- Acuerdo N.12 del 15 de Noviembre de 2002 "Por medio del cual se fijan los factores para el reconocimiento y pago de la hora para los docentes que prestan servicios a la Universidad Distrital en la modalidad de vinculación Especial en los programas de pregrado"

[11] OFICINA ASESORA DE SISTEMAS, OAS. *Levantamiento de la primera versión del proceso de parametrización y liquidación de nómina soportado en el aplicativo de nómina de funcionarios y pensionados de la Universidad Distrital*, Régimen Salarial, Trabajadores Oficiales, págs. 50-57, 2013.

[12] OFICINA ASESORA DE SISTEMAS, OAS. *Modelo de Negocio Nómina de Docentes de Vinculación Especial*, págs. 2-14, 2013.

- Resolución No. 317 del 8 de Septiembre de 2006 “Por medio de la cual se fija el escalafón equivalente para los profesores de VINCULACIÓN ESPECIAL, que prestan servicios a la Universidad Distrital Francisco José de Caldas”

Y las expedidas para la presente vigencia así:

- Decreto No. 1003 DE 2013 “Por el cual se dictan disposiciones en materia salarial y prestacional para los empleados públicos docentes y administrativos de las Universidades Estatales u Oficiales”.
- Resolución No.352 del 24 de junio de 2013 “por medio de la cual se acoge y aplica el Decreto 1003 de 21 de mayo de 2013 a los docentes de Vinculación Especial Hora Cátedra, Medio tiempo Ocasional y Tiempo Completo Ocasional en Pregrado en lo pertinente únicamente al valor del punto”. 1992, por el cual se organiza el servicio público de la Educación Superior y posteriormente se han generado una serie de normas entre las cuales podemos encontrar:
    - Acuerdo No.005 del 27 de Julio 27 de 2001"Por medio del cual se fija el valor de la Hora Cátedra y se establece el número máximo de horas para los docentes que prestan servicios a la Universidad Distrital Francisco José Caldas, en los Programas de Posgrado”

5.4.3. Contratistas

De conformidad con lo establecido en las normas tanto a nivel nacional como institucional, la Universidad Distrital Francisco José de Caldas puede celebrar contratos en cumplimiento de las normas de derecho público, civiles y comerciales.[13]

Teniendo en cuenta lo anterior, Consejo Superior Universitario suscribió el Acuerdo 08 de 2003 el cual establece que podrán contratar con la Universidad Distrital las personas naturales y jurídicas.

Posteriormente se expide la Resolución 014 del 5 de febrero de 2004, estableciendo que cuando la Universidad o una de sus dependencias, requiera la prestación directos de un servicio profesional, técnico o asistencial, podrá mediante orden de prestación de servicios, contratarlo. La misma resolución estableció que los pagos para este concepto se harían por honorarios y se incrementará de conformidad con el aumento anual del salario

Durante la vigencia 2006 se expidió la Resolución de Rectoría No.04 por la cual "Por media de la cual se precisa la modalidad de contratación mediante Órdenes de Prestación de Servicios y se deroga la Resolución de Rectoría 03 de 2005"

En concordancia con dicha norma se expidió la Resolución 031 de 2008 “por la cual se ajusta y se precisa el procedimiento para la selección en la contratación de órdenes de prestación de servicios (OPS) en la Universidad Distrital Francisco José de Caldas”

[13] OFICINA ASESORA DE SISTEMAS, OAS. *Modelo de Dominio Nómina de Contratistas*, págs. 2-12, 2013.

Mediante resolución No.143 de 2009, en su artículo 1o. numeral a. delega en la Vicerrectoría Administrativa y Financiera la competencia contractual, la ordenación del gesto y el pago, entre otros de los rubros de Honorarios y Remuneración de Servicios Técnicos.

Ya para finalizar, la Resolución 049 de 2011, ajusta el procedimiento para la contratación de órdenes de prestación de servicios (OPS) y se dictan otras disposiciones.

5.5. El Software Libre

5.5.1. La Filosofía del Software Libre

Es necesario mencionar los conceptos que se estipulan en un desarrollo de software libre, para ello *la definición de software libre estipula los criterios que se tienen que cumplir para que un programa sea considerado libre. De vez en cuando se modifica esta definición para clarificarla o para resolver problemas sobre cuestiones delicadas.*

«Software libre» es el software que respeta la libertad de los usuarios y la comunidad. A grandes rasgos, significa que **los usuarios tienen la libertad de ejecutar, copiar, distribuir, estudiar, modificar y mejorar el software**. Es decir, el «software libre» es una cuestión de libertad, no de precio. Para entender el concepto, piense en «libre» como en «libre expresión», no como en «barra libre». En inglés a veces decimos «libre software», en lugar de «free software», para mostrar que no queremos decir que es gratuito.

Se proveen estas libertades porque todos merecen tenerlas. Con estas libertades, los usuarios (tanto individualmente como en forma colectiva) controlan el programa y lo que este hace. Cuando los usuarios no controlan el programa, decimos que dicho programa «no es libre», o que es «privativo». Un programa que no es libre controla a los usuarios, y el programador controla el programa, con lo cual el programa resulta ser un instrumento de poder injusto.

Un *programa es software libre si los usuarios tienen las cuatro libertades esenciales*[14]:

- La libertad de ejecutar el programa como se desea, con cualquier propósito (libertad 0).
- La libertad de estudiar cómo funciona el programa, y cambiarlo para que haga lo que usted quiera (libertad 1). El acceso al código fuente es una condición necesaria para ello.
- La libertad de redistribuir copias para ayudar a su prójimo (libertad 2).
- La libertad de distribuir copias de sus versiones modificadas a terceros (libertad 3). Esto le permite ofrecer a toda la comunidad la oportunidad de beneficiarse de las modificaciones. El acceso al código fuente es una condición necesaria para ello.

[14]OFICINA DE LICENCIAS Y CUMPLIMIENTO DE LA FSF, *¿Qué es el Software Libre?, Free Software Foundation*, 2015, `http://www.gnu.org/philosophy/free-sw.es.html` [Consulta: Miércoles, 9 de Septiembre de 2015]

«Software libre» no significa que «no es comercial». Un programa libre debe estar disponible para el uso comercial, la programación comercial y la distribución comercial. La programación comercial de software libre ya no es inusual; el software libre comercial es muy importante. Puede haber pagado dinero para obtener copias de software libre, o puede haber obtenido copias sin costo. Pero sin tener en cuenta cómo obtuvo sus copias, siempre tiene la libertad de copiar y modificar el software, incluso de vender copias.

Si una modificación constituye o no una mejora, es un asunto subjetivo. Si su derecho a modificar un programa se limita, básicamente, a modificaciones que alguna otra persona considera una mejora, el programa no es libre.

Un problema particular se presenta cuando la licencia requiere que a un programa se le cambie el nombre con el cual será invocado por otros programas. De hecho este requisito dificulta la publicación de la versión modificada para reemplazar al original cuando sea invocado por esos otros programas. Este tipo de requisitos es aceptable únicamente cuando exista un instrumento adecuado para la asignación de alias que permite especificar el nombre del programa original como un alias de la versión modificada.

Una licencia libre no puede exigir la conformidad con la licencia de un programa que no es libre. Así, por ejemplo, si una licencia requiere que se cumpla con las licencias de «todos los programas que se usan», en el caso de un usuario que ejecuta programas que no son libres este requisito implicaría cumplir con las licencias de esos programas privativos, lo cual hace que la licencia no sea libre.

Cuando se habla de software libre, es mejor evitar usar términos como «regalar» o «gratuito», porque dichos términos implican que el asunto es el precio, no la libertad.

5.5.2. El Software Libre en Bogotá D.C.

Para tener presente las políticas del Distrito en relación con el software libre tomamos como base la Directiva 011 de 2012, de Noviembre 1, promulgada por el Alcalde Mayor de Bogotá D.C., el señor Gustavo Petro, esta directiva tiene por asunto "PROMOCIÓN Y USO DE SOFTWARE LIBRE EN EL DISTRITO CAPITAL", con lo cual nos permitimos contextualizar la filosofía del distrito al respecto.

Así pues para tener claro la filosofía política el distrito estipula en la Directiva 011 lo siguiente[15]:

> Al tenor de las disposiciones previstas en el Acuerdo Distrital 279 de 2007 *"Por el cual se dictan los lineamientos para la Política de Promoción y Uso del Software libre en el Sector Central, el Sector Descentralizado y el Sector de las Localidades del Distrito Capital"* el Honorable Concejo de Bogotá D.C., definió los lineamientos guía de la reglamentación que a través de la Resolución N° 305

[15] PETRO, Gustavo. Directiva 011 de 2012 (Noviembre 1), *Promoción y uso de Software libre en el Distrito Capital*, 2012, Obtenido de: `http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=50214` [Consulta: Miércoles, 9 de Septiembre de 2015]

de 2008' profirió la Comisión Distrital de Sistemas -CDS-, específicamente en lo contemplado por los artículos 64° a 68° referidos a la Política para Promocionar el Uso del Software Libre en las Entidades del Distrito Capital, como una estrategia orientada a la racionalización del gasto público y a la búsqueda de soluciones alternativas que proveen funcionalidades que el Distrito requiere.

Fundamentado en lo anterior, de conformidad con los lineamientos que define el Artículo 68° de la Resolución Nº 305 de 2008, el Jefe de cada Entidad y Organismo "*debe acoger, difundir y aplicar las políticas para promocionar el uso del software libre en las entidades del Distrito Capital, la cual será difundida y aplicada por el área que éste designe. (...). Parágrafo: corresponde a los Jefes de dependencia, responsables de área, grupos de trabajo e intervinientes en los procesos y procedimientos asociados con las Tecnologías de Información y Comunicaciones (TIC), garantizar la implementación, la divulgación, la aplicación y el seguimiento de la política de promoción y uso del software libre prevista en este capítulo*".

Con base en lo expuesto, se solicita a cada entidad del Distrito Capital, adelantar las siguientes acciones:

1. Registrar los datos del talento humano experto en software libre en el sitio web que para el efecto publicará la Comisión Distrital de Sistemas de la Alcaldía Mayor de Bogotá, información que deberá permanecer actualizada.
2. Documentar los proyectos exitosos que se hayan adelantado en implementación de software libre.
3. Remitir a la Oficina de Alta Consejería Distrital de Tecnologías de Información y Comunicaciones - TIC-, dentro de los treinta (30) días hábiles siguientes a la expedición de esta Directiva, un plan de acción con el correspondiente cronograma que identifique para las vigencias 2013 a 2016, los proyectos y recursos que serán destinados para efectos de implementar y aplicar la política de promoción y uso de Software Libre en el marco de lo establecido en la Resolución No. 305 de 2008 y su Anexo 18 (el cual puede ser consultado en la página www.cds.gov.co).
4. Con el objeto de generar la racionalización del gasto en materia tecnológica, las entidades deben dar cumplimiento al numeral 4.5 del Anexo 18 de la Resolución Nº 305 de 2008, que formula las "Políticas sobre la disposición de orientaciones tendientes a que en los estudios de mercado, que soportan los procesos de contratación del Distrito, se incluya la valoración y evaluación de herramientas tecnológicas basadas en software libre, en los casos pertinentes'; en consecuencia, las entidades deben darle prioridad a la adquisición de soluciones de software libre y adquirir software privativo solo para los casos en que sea plenamente justificado.

36

5.6. El Framework SARA

5.6.1. Definiendo el Framework

Para el desarrollo se utilizara un framework gestionado por la Oficina Asesora de Sistemas, por lo cual es necesario identificar y definir con claridad lo que es este framework, así pues SARA es un Sistema para la Articulación Rápida de Aplicaciones (System for Addressing the Rapid development of Applications), es un marco de trabajo para el desarrollo de aplicaciones orientadas a la web.

Está escrito en lenguaje PHP y propende por una arquitectura dividida en capas.

La Instalación de SARA requiere para su funcionamiento un servidor web, preferiblemente Apache HTTP (con soporte para PHP), un motor de bases de datos con soporte para PostgreSQL, MySQL, Oracle. Se recomienda que el acceso a las carpetas de SARA sea deshabilitado desde la configuración del servidor

5.6.2. Antecedentes

SARA es una evolución del marco de desarrollo que desde el año 2005 se ha venido trabajando en la Universidad Distrital Francisco José de Caldas. Desde el año 2008, es dirigido conforme a los lineamientos de la Oficina Asesora de Sistemas y se distribuye como software libre como muestra del compromiso institucional con la Política Distrital de Fomento al Software Libre.

5.6.3. Derechos de Autor

En la información recopilada se determinó que el framework tiene una licencia de software libre como se ha indicado e indica textualmente[16]:

> *"SARA This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.*
>
> *This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details."*

[16] OFICINA ASESORA DE SISTEMAS, *GitHub, Framework SARA: Readme.txt*, Universidad Distrital Francisco José de Caldas, 2012, `https://github.com/frameworksara/sara` [Consulta: Miércoles, 9 de Septiembre de 2015]

6 METODOLOGÍA USADA

El método a aplicar en este proyecto es el proceso de desarrollo OPENUP/OAS, que se encuentra debidamente aprobado en la Universidad Distrital Francisco José de Caldas, mediante la Resolución de Rectoría número 461 de 2011[17]. Bajo esta estipulación en la pasantía se estará sujeto a la metodología que promueve el OpenUP en el rol de desarrollador de tal manera que a continuación se permitirá mostrar la forma de trabajo que como desarrolladores debemos cumplir en la pasantía.

Como se ha indicado la metodología que promueve el OPENUP/OAS es un desarrollo iterativo, por tanto las tareas de desarrollo son continuas y los productos y tareas son incrementales y están sujetos a lineamientos del OPENUP que se describieron con anterioridad en el marco teórico. Así pues lo que sigue será aclarar la metodología que concierne al desarrollo en este proyecto, por ser el objetivo de la pasantía de la que se ocupa el presente anteproyecto.

6.1. Identificar las Oportunidades de Reutilzación de Código

Parte importante de la metodología es identificar código existente u otros elementos de puesta en funcionamiento que puedan ser utilizados como parte de las tareas de puesta en funcionamiento. Un manejo adecuado del diseño siempre será útil ya que las oportunidades de reciclaje son más evidentes cuando se tiene una visión clara de lo que debe tener la solución.

Se debe tener siempre en consideración elementos de software libre o de propiedad de la institución manteniendo un especial cuidado por el respeto de las normas y legislación relacionada con el derecho de autor. Tener cuidado con licencias virales, freeware o shareware.

6.2. Transformar Diseños de Software con Funcionalidad

Es de vital importancia dotar de funcionalidad los diseños arquitecturales y diagramas de casos de usos, de secuencia y similares con funcionalidad para lograr los productos

[17]BAHAMON CALDERON, Inocencio. Resolución de Rectoría N° 461, SISGRAL, p. 1-2, 2011, `http://sgral.udistrital.edu.co/xdata/rec/res_2011-461.pdf` [Consulta: Viernes, 12 de Septiembre de 2015]

de software que son claramente el objeto de todo el proceso de desarrollo, no obstante, si se utilizan herramientas sofisticadas de modelado es relativamente fácil obtener gran cantidad de código fuente directamente de la transformación de los modelos. Sin embargo hay que notar que aún en casos ideales se requerirán tareas de programación sobre el código generado para completar la puesta en funcionamiento, y que estos se ajusten a los requerimientos que fueron elaborados por los Ingenieros de Requerimientos.

6.3. Escribir el Código Fuente

La tarea principal del rol de desarrollo es claramente generar codigo fuente de la solución, es decir se debe escribir el código fuente para hacer una puesta en funcionamiento que cumpla con el diseño y el comportamiento esperado. Se debe procurar la reutilización o generación a partir de modelos pero aún en estos casos se requerirá de algún tipo de programación. Para ejecutar esta tarea con éxito por lo tanto la metodología apropiada para conseguir esto es:

- **Examinar los requerimientos**. No toda la información de los requerimientos se puede colocar en los modelos por tanto se debe verificar que parte de la implementación no cumple con ellos y así depurarla a través de programación directa.
- **Reconstruir el código para mejorar el diseño del mismo**. La readaptación es una técnica con la que se mejora la calidad del código a partir de pequeños, y seguros, cambios. La idea es tratar de crear piezas reutilizables.
- **Afinar los resultados de implementaciones existentes** mejorando el desempeño, la interfaz de usuario, la seguridad y otras áreas no funcionales. La idea es reutilice y mejore, con esto se logra una evolución del código.
- **Agregar detalles faltantes**, tales como completar la lógica de ciertas operaciones, adicionar clases de soporte, estructuras de datos, etc.
- **Manejar condiciones de frontera**.
- **Administrar estados de error o circunstancias inusuales**. En todo caso tratar de crear elementos genéricos que puedan ser utilizados en la mayor cantidad de circunstancias. • Restringir el comportamiento. (previniendo que el código de usuario pueda ejecutar flujos o escenarios ilegales).

6.4. Utilizar Estándares para la Escritura de Código

Para realizar desarrollos profesionales se utilizan estándares que garanticen una construción adecuada por parte de un equipo de trabajo, el cual se enfoque en una solución de software, por lo cual la metodología dicta que se deberan describir varias convenciones acerca de como escribir el código fuente. Su principal tarea es asegurar la consistencia, calidad y una puesta en funcionamiento fácil de entender.

40

El uso de los estándares para la escritura de código es una de las prácticas más extendidas en la práctica de realización de software y se vuelve imperativa cuando se labora en ámbitos de alta colaboración. Los grupos de desarrollo deben tener una forma estándar para nombrar y dar formato a las cosas de tal manera que el código fuente se pueda entender rápidamente y la propiedad del mismo pueda ser cambiada sin detrimento de la calidad.

Idealmente los estándares para la escritura de código debe ser el resultado de un consenso entre el equipo de trabajo ya que esta tarea ayuda a la rápida adopción de los estándares.

Los estándares mencionados pueden cubrir áreas como:

- **Estándares para asignación de nombres:** Esto incluye la forma en que se le da nombre a todos los elementos dentro del código. Cuando se cubren elementos de gran escala pueden solapar los estándares de diseño.
- **Organización de los archivos:** Incluye convenciones para colocar nombres a los archivos y como éstos deben ser organizados dentro del árbol de directorios del sistema.
- **Estándares para los comentarios:** Poner demasiado énfasis en los comentarios denota una pérdida de confianza en cuanto la calidad del software que se está escribiendo. Además, se tendrá siempre una impresión de que los comentarios estarán desactualizados. La idea es estandarizar la forma en la cual se comenta el código para soportar la capacidad de soporte y la capacidad de generar documentación a partir del código.
- **Convenciones para la escritura de código:** Aplicación de convenciones específicas a nivel de código.
- **Espacio en blanco:** Aunque algunos autores lo consideran como de menor impacto es un hecho que el manejo adecuado de los espacios en blanco, los saltos de línea, las sangrías y las lineas en blanco facilitan la lectura del código.

Cada ítem en el estándar debe alcanzar una o más metas siendo la más importante la de **mejorar la comunicación entre los integrantes del equipo**. Una vez que el equipo se pone de acuerdo con el estándar todos ellos deben seguir estrictamente las reglas marcadas. Con el tiempo se espera que el equipo modifique el estándar de acuerdo a las experiencias de uso y se cree después de varias iteraciones uno que se amolde al contexto.

Aunque algunos estándares pueden trascender y ser utilizados para cualquier lenguaje lo más comúnes que éstos sean específicos para un lenguaje en particular, y la metodología que se nos presenta nos permitira utilizar de estás herramientas de coordinación de equipo.

6.5. Evaluar la Implementación

Para tal fin se realizaran pruebas unitarias entre otras, esto con el objetivo de verificar que la puesta en funcionamiento está de acuerdo al propósito por el cual fue construida.

Examinar el código para comprobar el cumplimiento de la funcionalidad por la cual fue construido. Lo anterior constituye un paso necesario para verificar la calidad del producto y debe complementar las pruebas que han sido descritas en otras tareas. Al realizar esto se debera tener en cuenta:

- **Programación en parejas:** Al trabajar por parejas se puede ir evaluando la calidad del código a medida que se escribe (al tener que llegar a acuerdos con la pareja se garantiza que el código es completo, entendible, claro, organizado, cumple con las normas de codificación, etc)
- **Leer el código buscando fallos comunes**. Considerar el uso de listas de verificación que estén a la mano para referencia.
- **Usar herramientas para verificar errores de puesta en funcionamiento o código inapropiado**. Por ejemplo, usar herramientas de estilo, sintaxis o colocar el compilador (depurador) en el nivel más detallado de advertencias.
- **Usar herramientas que permitan visualizar el código** para poder detectar fácilmente acoplamientos excesivos entre elementos o referencias circulares.
- **Realizar pequeñas inspecciones de código informales y personalizadas**. Convoque a colegas para que revisen secciones críticas del código. Evitar porciones extensas. En todo caso siempre realizar un modelo del código que permita ubicar al grupo de pruebas
- **Mejorar la puesta en funcionamiento** de acuerdo a los requerimientos.

6.6. Comunicar Decisiones Importantes (Control de Cambios)

En la metodología se establece como un aspecto de retroalimentación que requiere un proceso satisfactorio iterativo, donde se permitirá comunicar rápidamente el impacto de cambios no esperados en el diseño o los requerimientos.

Los problemas y restricciones que no se hayan tenido en consideración deben ser comunicados al equipo de desarrollo. El impacto de problemas descubiertos durante la puesta en funcionamiento debe ser incorporado para las decisiones futuras. Se es apropiado se deben actualizar los requerimientos para reflejar las ambigüedades identificadas y resueltas durante la puesta en funcionamiento de tal forma que puedan ser verificadas y así manejar correctamente las expectativas de los interesados. De forma similar actualice el diseño para reflejar nuevos problemas y restricciones descubiertas durante la puesta en funcionamiento asegurando de esta forma que la información es comunicada a otros realizadores.

Si los cambios y actualizaciones son de gran impacto o trascienden el ámbito del desarrollo personal se requiere que se comuniquen efectivamente a todo el equipo siendo lo más adecuado crear peticiones de cambio.

7 DESARROLLO DEL PROYECTO

Aqui yo creo que describir lo que utilizamos para desarrollar... que los analisistas generaban casos de uso y mostrar el modelo ese que tienen con toda la relacion entre los modulos luego.... poner la arquitectura o lon que haya de eso como general... decir que eran las entradas esos pauqtes de desarrollo y luego se relaizaban las iteraciones y eso..... no se que mas colocar jajajajajajaja

7.1. Requerimientos del Sistema

7.2. Arquitectura del Sistema

7.3. Realización de las Iteraciones

7.4. Control de Cambios

8 CONSTRUCCIÓN DE LA SOLUCIÓN

A conti

8.1. Componentes Desarrollados

Los Componentes desarrollados atacan los siguientes Modulos del Sistema de Nomina TITAN:

Grafico de los Modulos de todo el sistema resaltando los que desarrollamos nosotros directamente....

8.2. Módulo Gestión Persona

8.3. Módulo Novedades - Datos Funcionario

8.4. Módulo Párametros - Fondo de Pensión

8.5. Módulo Párametros - Caja de Compensación

8.6. Módulo Párametros - Cargo

8.7. Módulo Párametros - Nivel Cargo

8.8. Módulo Párametros - Cátegoria Párametros

8.9. Módulo Conceptos - Conceptos de Nómina

8.10. Módulo Conceptos - Cátegoria de Conceptos

8.11. Módulo Liquidación - Tipo de Nomina

8.12. Módulo Liquidación - Preliquidación

8.13. Módulo Reportes - Plantillas de Reporte

8.14. Módulo Reportes - Reporte

46

9 TERMINACIÓN DEL PROYECTO

A continuación

9.1. Entrega del Proyecto

9.2. Conclusiones

10 RESULTADOS

El proyecto

Bibliografía

[1] Bahamon Calderon, Inocencio: Rectoría Resolución N° 461, Por medio de la cual se adopta el Método del Proceso de Desarrollo OPENUP/OAS como Marco de Trabajo Institucional en el Analisis, Diseño, Desarrollo e Implantación de Productos de Software al Interior de la Universidad Distrital Francisco Jose de Caldas. En: *SISGRAL* 461 (2011), p. 2

[2] Oficina de Licencias y Cumplimiento de la FSF, OLC. *¿Qué es el Software Libre?, Free Software Foundation*. 9 2015

[3] Petro, Gustavo: Promoción y uso de Software libre en el Distrito Capital. En: *Alcaldia de Bogotá* Directiva 011 de 2012 (Noviembre 1) (2012), p. 2

[4] Project Eclipse, EPFP. *Eclipse Process Framework Project*. Julio 2015

[5] Oficina Asesora de Sistemas, OAS: Capitulo 9, Desarrollo Solución. En: *Proceso de Desarrollo OPENUP/OAS* Oficina Asesora de Sistemas, Versión:0.0.0.7 (2011), p. 23

[6] Oficina Asesora de Sistemas, OAS: Guía Rápida Proceso de Desarrollo OPENUP/OAS. En: *Proceso de Desarrollo OPENUP/OAS, Guía Rápida* Oficina Asesora de Sistemas, Versión:0.0.0.7 (2011), p. 11

[7] Oficina Asesora de Sistemas, OAS. *GitHub, Framework SARA: Readme.txt*. 11 2012

[8] Oficina Asesora de Sistemas, OAS. ; de Nómina de la Universidad Distrital., Proyecto Sistema I. (Ed.): *Levantamiento de la primera versión del proceso de parametrización y liquidación de nómina soportado en el aplicativo de nómina de funcionarios y pensionados de la Universidad Distrital*. Universidad Distrital Francisco José de Caldas, 2013

[9] Oficina Asesora de Sistemas, OAS. ; de Nómina de la Universidad Distrital., Proyecto Sistema I. (Ed.): *Modelo de Dominio Nómina de Contratistas*. Proyecto Sistema Integral de Nómina de la Universidad Distrital., 2013

[10] Oficina Asesora de Sistemas, OAS. ; de Nómina de la Universidad Distrital., Proyecto Sistema I. (Ed.): *Modelo de Negocio Nómina de Docentes de Vinculación Especial*. Universidad Distrital Francisco José de Caldas, 2013

[11] Trecet, José. *Financialred, Red de Blogs Especializados en Finanzas, Economía y Bolsa*. Julio 2015

ANEXOS

DOCUMENTACIÓN DE PASANTÍA (ACUERDO 002/2002)