UNIVERSIDAD CATOLICA DE SANTA MARIA

FACULTAD DE CIENCIAS E INGENIERIAS FÍSICAS Y FORMALES

PROGRAMA PROFESIONAL DE INGENIERIA DE SISTEMAS

"EXPERIENCIA PROFESIONAL EN LA DIVISIÓN SISTEMAS DEL BANCO DE CRÉDITO DEL PERÚ DURANTE LOS AÑOS 2008 AL 2012"

Trabajo Informe presentado por:

Br. Quiroz Pilco, Lenin Edison

Para optar el Título Profesional de:

INGENIERO DE SISTEMAS

AREQUIPA - PERÚ

2014

PRESENTACIÓN

Sra. Directora del Programa Profesional de Ingeniería de Sistemas.

Srs. Miembros del Jurado.

De conformidad con las disposiciones del Reglamento de Grados y Títulos del Programa Profesional de ingeniería de Sistemas, pongo a vuestra consideración el presente Plan de Trabajo Informe titulado:

"Experiencia profesional en la división sistemas del Banco de Crédito del Perú durante los años 2008 al 2012", el mismo que de ser aprobado me permitirá optar el Título Profesional de Ingeniería de Sistemas.

Quiroz Pilco, Lenin Edison

AGRADECIMIENTOS

Agradezco a Dios por los padres que me dio, que siempre me apoyaron en todos mis proyectos y que sin su apoyo no habría podido cumplir con esta meta, a mi esposa por darme las fuerzas para finalizar el presente trabajo y a los docentes de la Universidad que me apoyaron con sus consejos y experiencia.

DEDICATORIA

El presente trabajo se lo quiero dedicar a mis padres Lenin Sergio e Ysabel por todo su sacrificio en darme la oportunidad de estudiar una carrera y por darme todos esos buenos consejos que han hecho de mí una persona de bien.

INDICE

INDICE DE FIGURAS

RESUMEN

Actualmente para el Banco de Crédito del Perú es importante contar con un proceso de verificación y validación de estándares en todos los desarrollos de software que se dan dentro de la empresa, porque aumenta la confianza y garantiza un producto de calidad, también le permite hacer mejoras en los aplicativos en caso sean necesarios.

La empresa maneja un alto número de transacciones e información, por tal motivo un gran numero de las aplicaciones importantes dentro de la empresa fue desarrollado con el lenguaje de programación COBOL BATCH, debido a la importancia de las aplicaciones desarrolladas con este lenguaje es que se elaboró estándares de programación para garantizar su calidad en el desarrollo y para permitir su mantenimiento, también se vio necesaria la elaboración de una herramienta que ayude a la revisión de los estándares; se desarrollo la herramienta Host Code Reviewer(HCR) la cual automatiza el proceso de verificación y validación realizando una inspección de código en programas desarrollados en COBOL BATCH, desplegando resultados sobre dicha validación, otorgando una puntuación que evaluará el nivel de cumplimiento de estándares antes de su implementación en ambientes de producción

El objetivo de este trabajo es mejorar la herramienta HCR existente, permitiendo la validación de JOBs y Procedimientos basado en los estándares de programación definidos dentro de la empresa.

ABSTRACT

Currently the Banco de Credito del Peru is important to have a process of verification and validation of standards in all software development that occur within the company, because it increases confidence and ensures a quality product, also allows you to make improvements in applications where necessary.

The company handles a large number of transactions and information, as such a large number of important applications within the company was developed with the programming language COBOL BATCH, because of the importance of the applications developed with this language is to be developed programming standards to ensure quality in development and to allow for maintenance, was also necessary to develop a tool to assist the review of the standards development tool is Host Code Reviewer (HCR) which automates the process of verification and validation by inspection of code in programs COBOL BATCH, displaying results on this validation, giving a score to assess the level of compliance with standards before deployment in production environments.

The objective of this work is to improve the existing HCR tool, allowing validation of JOBs and Procedures based on programming standards defined within the company.

INTRODUCCIÓN

El presente trabajo tiene como objetivo agregar las funcionalidades de validación de JOBs y Procedimientos en la herramienta HCR.

El alcance del trabajo consiste en exponer el trabajo más importante desarrollado dentro del Área de Sistemas del Banco de Crédito del Perú, mostrando los principales aspectos de su implementación, resaltando los puntos más importantes como Metodología empleada para el análisis, Estándares utilizados para la implementación y los Casos de prueba utilizados durante el desarrollo para la verificación del aplicativo terminado.

En el capítulo 1 se podrá ver la memoria descriptiva donde se verá información sobre la empresa y las funciones realizadas dentro de ella.

En el capítulo 2 se encuentra el marco teórico donde se muestra el proceso empleado para el desarrollo de software dentro de la empresa.

En el capítulo 3 se muestra el análisis realizado para la implantación del trabajo más importante desarrollado dentro de la empresa.

En el capítulo 4 se mostrara los principales aspectos de la implementación de la herramienta Host Code Reviewer, resaltando los puntos más importantes como es la identificación de los estándares a implementar, la forma en que se aplican las reglas en el código fuente de los programas y los casos de prueba aplicados sobre la herramienta.

CAPÍTULO I

MEMORIA DESCRIPTIVA

1.1. Empresa

1.1.1. Descripción de la empresa[URL01]

La empresa, llamada durante sus primeros 52 años Banco Italiano, inició sus actividades el 9 de abril de 1889, adoptando una política crediticia inspirada en los principios que habrían de guiar su comportamiento institucional en el futuro. El 01 de febrero de 1942, se acordó sustituir la antigua denominación social, por la de Banco de Crédito del Perú.

Así, el Banco Italiano, el primero en el país, cerró su eficiente labor después de haber obtenido los más altos resultados de la institución. Con el propósito de conseguir un mayor peso internacional, se instalaron sucursales en Nassau y en Nueva York, hecho que convirtió a la empresa en el único Banco peruano presente en dos de las plazas financieras más importantes del mundo. La expansión de sus actividades creó la necesidad de una nueva sede para la dirección central. Con ese fin se construyó un edificio de 30,000 m2, aproximadamente, en el distrito de La Molina. Luego, con el objetivo de mejorar sus servicios, establecieron la Red Nacional de Tele Proceso, que a fines de 1988 conectaba casi todas las oficinas del país con el computador

central de Lima; asimismo, crearon la Cuenta Corriente y Libreta de Ahorro Nacional, e instalaron una extensa red de cajeros automáticos.

En 1993, se adquirió el Banco Popular de Bolivia, hoy Banco de Crédito de Bolivia. Un año más tarde, con el fin de brindar una atención aún más especializada, crearon Credifondo, una nueva empresa subsidiaria dedicada a la promoción de los fondos mutuos; al año siguiente se estableció Credileasing, empresa dedicada a la promoción del arrendamiento financiero. Durante los '90, la oficina de representación en Santiago de Chile desarrolló una interesante actividad, dado el notable incremento de los capitales chilenos invertidos en empresas peruanas. La recuperación de los jóvenes talentos que emigraron entre 1970 y 1990 al extranjero, fue otro aspecto importante de esa década. Esos profesionales, sólidamente formados en centros académicos y empresas importantes de los Estados Unidos y Europa, han contribuido a confirmar la imagen que siempre se tuvo: un Banco antiguo con espíritu siempre moderno. Al cumplir los 120 años de existencia, la Institución cuenta con 330 oficinas, 1300 cajeros automáticos, 4,000 Agentes BCP y 14,311 empleados; y bancos corresponsales en todo el mundo.

1.1.2. Tipo de empresa

Financiera.

1.1.3. Ubicación[URL01]

La oficina principal de la empresa se encuentra ubicada en la ciudad de Lima en el distrito de la Molina, Calle Centenario N° 166 Urb. Las Laderas de Melgarejo. El centro de desarrollo de Sistemas principal se encuentra ubicado en Chorrillos, como sucursales de desarrollo de sistemas se tienen2 oficinas ubicadas en la ciudad de Arequipa, una en la Calle Perú 207 – Cercado y la otra en la Calle Urubamba 301 – Cayma.

1.1.4. Organigrama

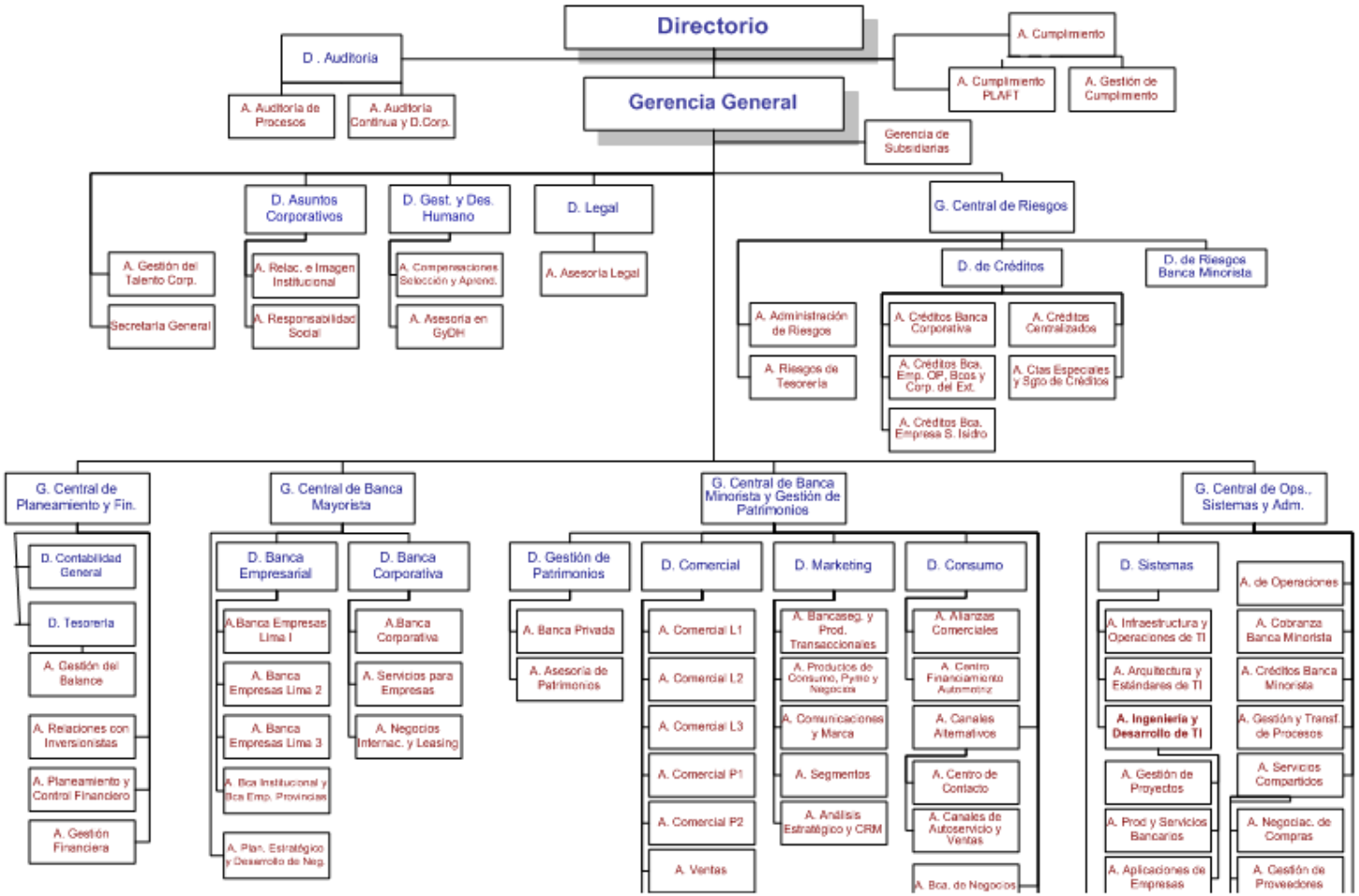

Directorio
D . Auditoria
A. Auditoria de Procesos
A. Auditoria Continua y D.Corp.
A. Cumplimiento
A. Cumplimiento PLAFT
A. Gestión de Cumplimiento
Gerencia General
Gerencia de Subsidiarias
A. Gestión del Talento Corp.
Secretaria General
D. Asuntos Corporativos
A. Relac. e Imagen Institucional
A. Responsabilidad Social
D. Gest. y Des. Humano
A. Compensaciones Selección y Aprend.
A. Asesoría en GyDH
D. Legal
A. Asesoría Legal
G. Central de Riesgos
D. de Créditos
D. de Riesgos Banca Minorista
A. Administración de Riesgos
A. Riesgos de Tesorería
A. Créditos Banca Corporativa
A. Créditos Centralizados
A. Créditos Bca. Emp. OP, Bcos y Corp. del Ext.
A. Ctas Especiales y Sgto de Créditos
A. Créditos Bca. Empresa S. Isidro
G. Central de Planeamiento y Fin.
D. Contabilidad General
D. Tesorería
A. Gestión del Balance
A. Relaciones con Inversionistas
A. Planeamiento y Control Financiero
A. Gestión Financiera
G. Central de Banca Mayorista
D. Banca Empresarial
A.Banca Empresas Lima I
A. Banca Empresas Lima 2
A. Banca Empresas Lima 3
A. Bca Institucional y Bca Emp. Provincias
A. Plan. Estratégico y Desarrollo de Neg.
D. Banca Corporativa
A.Banca Corporativa
A. Servicios para Empresas
A. Negocios Internac. y Leasing
G. Central de Banca Minorista y Gestión de Patrimonios
D. Gestión de Patrimonios
A. Banca Privada
A. Asesoría de Patrimonios
D. Comercial
A. Comercial L1
A. Comercial L2
A. Comercial L3
A. Comercial P1
A. Comercial P2
A. Ventas
D. Marketing
A. Bancaseg. y Prod. Transaccionales
A. Productos de Consumo, Pyme y Negocios
A. Comunicaciones y Marca
A. Segmentos
A. Análisis Estratégico y CRM
D. Consumo
A. Alianzas Comerciales
A. Centro Financiamiento Automotriz
A. Canales Alternativos
A. Centro de Contacto
A. Canales de Autoservicio y Ventas
A. Bca. de Negocios
G. Central de Ops., Sistemas y Adm.
D. Sistemas
A. Infraestructura y Operaciones de TI
A. Arquitectura y Estándares de TI
A. Ingeniería y Desarrollo de TI
A. Gestión de Proyectos
A. Prod y Servicios Bancarios
A. Aplicaciones de Empresas
A. de Operaciones
A. Cobranza Banca Minorista
A. Créditos Banca Minorista
A. Gestión y Transf. de Procesos
A. Servicios Compartidos
A. Negociac. de Compras
A. Gestión de Proveedores

Figura Nº 1

1.1.5. Visión

Ser el Banco líder en todos los segmentos y productos que ofrecemos

1.1.6. Misión

Promover el éxito de nuestros clientes con soluciones financieras adecuadas para sus necesidades, facilitar el desarrollo de nuestros colaboradores, generar valor para nuestros accionistas y apoyar el desarrollo sostenido del país.

1.2. Actividad Profesional

1.2.1. Cargo desempeñado

Programador.

1.2.2. Funciones del programador

Las principales funciones del programador en la empresa son las siguientes:

- Producir software para el BCP y las diversas empresas o instituciones que así lo requieran del grupo Crédito.
- Elaborar sistemas que optimicen las aplicaciones del negocio y las tareas administrativas de todas las áreas de la empresa.
- Mantener y actualizar los sistemas automatizados.
- Programar sistemas en los diferentes lenguajes de programación con los que cuente la empresa.
- Elaborar diagramas de lógica o bloques de aplicación.
- Realizar el análisis y optimización de algoritmos para su aplicación en tiempo real.
- Participar en la solución de problemas de soporte de software.
- Formular manuales de operación y programación.
- Realizar pruebas de calidad del nuevo software que se vaya a implantar en los servidores o equipos de la empresa.
- Elaborar informes del desarrollo de su labor.
- Realizar otras funciones que le sean asignadas.

- Colaborar y realizar trabajos relacionados con su especialidad.

1.2.3. Descripción de los trabajos desarrollados

1.2.3.1. Refactoring de la Herramienta HCR (Host Code Reviewer)

Requerimiento: La Herramienta HCR (Host Code Reviewer), es una aplicación encargada de validar el cumplimiento de los estándares de programación para programas COBOL.

Utiliza 3 interfaces principales, desarrolladas en 2 lenguajes: Una DLL de nombre ErrorFinder.dll, la cual es la encargada de realizar el proceso de revisión de estándares propiamente dicho, dicha DLL se encuentra desarrollada en Visual C++ 6.0, Un "Agente" de conexión con Mainframe, para la recepción de los programas a ser validados, dicho agente utiliza una conexión XCOM, dicho agente se encuentra desarrollado en Visual Basic 6.0. Y finalmente la interfaz grafica de la aplicación la cual se encuentra desarrollada en Visual Basic 6.0. Se cuenta con una base de datos SQL 2000 que almacena información de reglas, validaciones, usuarios etc.
Se desea disminuir la complejidad del aplicativo y el nivel soporte que se le da a la herramienta.

Alternativa de solución implementada: Implementar la Herramienta desde cero, utilizando C# .NET 2003 para el desarrollo, como gestor de Base de Datos utilizar el Microsoft SQL Server 2008, para la implementación de las reglas de validación utilizar expresiones regulares para que su mantenimiento sea por Base de Datos.

Herramientas utilizadas: C# .NET 2003 y Microsoft SQL Server 2008.

1.2.3.2. Desacoplamiento TOLD II de Letras

Requerimiento: La aplicación Letras TP debe exponer los servicios comunes que contemplen la funcionalidad ofrecida por la interfaz ad hoc LERE.

El servicio común debe implementarse desacoplado de TOLD II.

Contempla: La validación de datos de letras, la generación del código de planilla, código de Remesa, la grabación de archivos de Letras, grabación de log y manejo de extorno.

Alternativa de solución implementada: Clonar programas que forman parte del proceso actual (actualización de tablas CONSIST) a fin de que éste se convierta en Servicio Común.

Se reutilizo la tabla de Productos de Letras a fin de registrar ciertos parámetros que permitan evitar el "HARD CODE" en el nuevo programa CBZPILER y clon del CBZPIFUE. Este programa CBZPILER fue implementado utilizando los estándares definidos y evitando las sentencias GO TO que contenía el programa original CBZPIFUE y también tenía que considerar los estándares definidos para el uso de Servicios Comunes. Se implemento un COPY CBZXMFUE que debía contener la estructura definida para los datos de input del programa CBZPILER.

Herramientas utilizadas: COBOL CICS y sentencias FIO para el uso de DATACOM.

1.2.3.3. Controlar Relación con Grupos Económicos – CONSIST Clientes

Requerimiento: Restringir la asociación de un cliente a más de un grupo económico mediante la pantalla REC (Relaciones entre Clientes.).

Generar un reporte que permita identificar a los clientes que están asociados a más de un grupo económico, el reporte debe incluir todos los campos de la pantalla REC.

Alternativa de solución implementada: Restringir en la pantalla Online de CONSIST (Modulo de clientes), pantalla REC para que no permita relacionar a clientes con más de un grupo económico.

Herramientas utilizadas: COBOL CICS, COLDVIEW (Herramienta Administradora de Documentos) y DATACOM.

1.2.3.4. Implementar Tarjeta Mastercard en SAPP(Sistema Administrador de Productos Pasivos)

El Sistema Administrador de Productos Pasivos (SAPP) es un sistema cliente/servidor propietario de BCP que permite la administración de la información de clientes, la apertura de productos pasivos y operaciones con tarjetas de débito, así como las consultas a diferentes sistemas del BCP.

Requerimiento: Permitir la atención del cliente de una Tarjeta Mastercard a través de SAP Pasivos.

Requisitos funcionales:

- Consulta de datos en CONSIST: Debe mostrar la información de los productos TC Mastercard que tenga afiliados el cliente persona natural.
- Consulta de datos en VPLUS (Aplicativo de Tarjetas de Crédito): Debe mostrar la información de los productos TC Mastercard que tenga afiliados el cliente persona natural.
- Afiliación a una Tarjeta de Débito: El aplicativo SAPP permite la afiliación de un producto pasivo (Ahorros, Corriente, Maestra, CTS) o activo (Tarjeta de crédito) de un cliente a una tarjeta de débito.
- Desafiliación de una Tarjeta de Débito: Permitir la desafiliación de productos de la tarjeta de débito seleccionada, actualizando la información en SAT (Sistema Administrador de Tarjetas).
- Modificación de datos de una Tarjeta de Débito: Mostrar el producto en pantalla en caso se encuentre afiliado a una tarjeta de débito y asimismo mostrar el producto afiliado en el contrato de modificación.
- Cambio de una Tarjeta de Débito: Se muestra el producto en pantalla en caso esté afiliado a la tarjeta de débito anterior, y asimismo se muestre en el contrato de cambio de la nueva tarjeta de débito. Se debe tener en cuenta que la opción de afiliación de productos TC Mastercard sólo estará disponible para clientes persona natural.

- Bloqueo de una Tarjeta de Débito: Se muestra el producto en pantalla en caso se encuentre afiliado a una tarjeta de débito y asimismo mostrar el producto afiliado en el contrato de bloqueo de la tarjeta de débito.

- Consulta de Productos por Tarjeta: Se debe mostrar la información de los saldos y movimientos de los productos pasivos que están afiliados a una tarjeta de débito Credimás.

- Consulta de Tarjetas por Producto: Se debe mostrar la información de las tarjetas de débito Credimás que tienen afiliado el producto TC Mastercard.

- Cambio de Clave Personal de una Tarjeta: La opción de cambio de la clave personal de una tarjeta de crédito Mastercard debe permitir el cambio del PIN de 4 dígitos, tan igual como se realiza con una tarjeta de crédito VISA

- Impresión de Contratos: Se requiere modificar el Contrato A para que se muestre el número de tarjeta TC Mastercard afiliado a la tarjeta de débito en la impresión del contrato, en original y/o copia.

- Registro de Control de Bloqueo: Se requiere modificar las pantallas para incluir las operaciones de bloqueo de las nuevas tarjetas TC Mastercard.

- Impresión de Reportes de Bloqueos: Se requiere modificar las pantallas y reportes para incluir las operaciones de bloqueo de las nuevas tarjetas TC Mastercard.

Alternativa de solución implementada: Realizar las modificaciones en SAPP para reconocer a la Tarjeta Mastercard como un producto distinto a las tarjetas de otras marcas (VISA, AMEX). La implementación de esta alternativa es de complejidad media debido a la necesidad de modificar los formularios donde se muestre información de tarjeta de crédito, los contratos que se imprimen para entregar al cliente y los reportes administrativos.

Herramientas utilizadas: Visual Basic 6.0 y SQL Server 2000.

1.2.3.5. Respaldo de Data Histórica RRHH BCP y Subsidiarias

Requerimiento: Generar reportes históricos para las siguientes compañías:

- 01 – BCP.
- 02 – Credibolsa.
- 03 – Credifondo.
- 04 – Solución Financiera de Crédito.
- 06 - Soluciones en Procesamiento S.A.
- 14 - Banco Santander Central Hispano.
- 15 - Santander Central Hispano S.A.F.
- 19 – Prima AFP.
- 12 – Pacifico Peruano Suiza Empleados.
- 20 – Pacifico Peruano Suiza Funcionarios.
- 21 – Pacifico S.A. Entidad Prestadora de Salud.
- 22 – Asociación Civil Asistencia Social Cristal.

Los reportes que se generaron por cada compañía son los siguientes:

- Reporte de boletas de pago de haberes.
- Reporte de Datos de Trabajador.
- Reporte de Datos Laborales de Trabajador.
- Reporte de Datos Financieros de Trabajador.
- Reporte de Datos Familiares de Trabajador.
- Reporte de Estudios de Trabajador.
- Reporte de Datos de Ausencia de Trabajador.
- Reporte de Datos de Tardanza de Trabajador.
- Reporte de Datos de Licencia de Trabajador.
- Reporte de Balance de Prestamos.
- Reporte de Detalle de Prestamos Vigente.
- Reporte de Rol y record Vacacional del Trabajador.
- Reporte Experiencia Laboral de Trabajador.
- Reporte Misiones de Trabajador.
- Reporte de Movilidad Comisiones del Trabajador.
- Reporte de Estructura de Organización.
- Reporte de Puestos de la Organización.

- Reporte de Grado Salarial y Aumentos de Trabajadores.
- Reporte de Movimiento de Personal.
- Reporte de Prestamos Administrativos Desembolsados.

Alternativa de solución implementada: Esta alternativa consiste en migrar los reportes solicitados por el usuario de los antiguos aplicativos de RRHH BCP y subsidiarias a COLDVIEW (Herramienta Administradora de Documentos) bajo el formato que actualmente se visualiza en los sistemas antiguos de RRHH. Para ello se hará uso de la herramienta Microsoft SQL Server 2000, para crear la rutina en el servidor de base de datos de RRHH, cuyo resultado será exportado a la herramienta COLDVIEW.

Se generaron DTSs para generar los reportes en archivos TXT en un servidor Windows Server 2003, en una ruta compartida para ser cargados en la herramienta COLDVIEW.

Herramientas utilizadas: Microsoft SQL Server 2000 y COLDVIEW.

1.2.3.6. Migración de DTS a SSIS – Suite Cambios

Requerimiento: Migrar 12 DTSs a SISS del aplicativo Suite Cambios que se encuentran SQL Server 2008 en modo compatibilidad considerando que se debe modificar sentencias SQL y procedimientos almacenados para que no tengan sentencias no permitidas y la comunicación con la BD sea correcta. También es necesario crear archivos .BAT para ejecutar los siguientes Jobs SQL desde la rutina Host [PERE11]:

- MCAM-IPM6AMM5-LeeTotalesCOMEVEMEHOST
- MCAM-IPM6AMM4-LeeDetallesCOMEVEMEHOST
- MCAM-IPM0AMCD-EnviaDataGLHOST desde Host.

Los DTS a migrar son los siguientes:

- MCAM-IPM0AMCD-EnviaDataGL.dts
- MCAM-IPM6AMM1-EnviaDataForwards.dts

- MCAM-IPM6AMM11-CargarCarteraTrader.dts
- MCAM-IPM6AMM12-EnviaMailRiesgos.dts
- MCAM-IPM6AMM2-GeneraFormato108Sucave.dts
- MCAM-IPM6AMM3-GeneraFormato108HistSucave.dts
- MCAM-IPM6AMM4-LeeDetallesCOMEVEME.dts
- MCAM-IPM6AMM5-LeeTotalesCOMEVEME.dts
- MCAM-IPM6AMM6-GeneraFormato205Sucave.dts
- MCAM-IPM6AMM7-GeneraCircular10BCR.dts
- MCAM-IPM6AMM8-GeneraReporteMovimientos.dts
- MCAM-IPM6AMM9-GeneraReporteInventario.dts

Alternativa de solución implementada: Migración con la opcion Migrate DTS 2000 Package del Visual Studio 2008, porque es una estrategia utilizada por el equipo de Mejoras Tecnológicas dando resultados satisfactorios, está permite disminuir el riesgo de migrar las tareas manualmente donde existe el riesgos de omitir alguna funcionalidad pero con la herramienta de migración se evita ese riesgo.

Herramientas utilizadas: Visual Studio 2008, Microsoft SQL Server 2008 y Cobol Batch.

CAPÍTULO II

MARCO TEORICO

Para la atención de requerimientos dentro de la empresa se cuenta con un proceso llamado PAR (Proceso de Atención de Requerimientos), proceso en el cual se indican las fases por las cuales pasa una solicitud del cliente hasta su implementación y entrega final.

2.1. Fases del PAR[URL02]

2.1.1. Análisis y diseño

El análisis facilita la especificación de la función y comportamiento de los programas, además de indicar la interfaz con otros elementos del sistema y establecer las ligaduras de diseño que se debe cumplir en el programa. **[PRES98].**

En esta fase del PAR (Proceso de Atención de Requerimientos) se realiza el análisis funcional y técnico de acuerdo a lo solicitado por el usuario, para el desarrollo y construcción del requerimiento, generándose el entregable DAD (Documento de Análisis y Diseño), en ese formato se especifica cuáles serán los cambios en la funcionalidad del aplicativo impactado y los detalles técnico para su implementación, como el lenguaje de programación a utilizar,

gestor de base de Datos, servidor o servidores impactados, permisos de usuario, etc.

2.1.2. Construcción

En esta fase se realiza la codificación de todo lo indicado en el DAD, se realizan las pruebas unitarias e integrales antes de pasar a la siguiente fase, también se generan los entregables necesarios para la implementación de los cambios en los ambientes de Certificación y Producción.

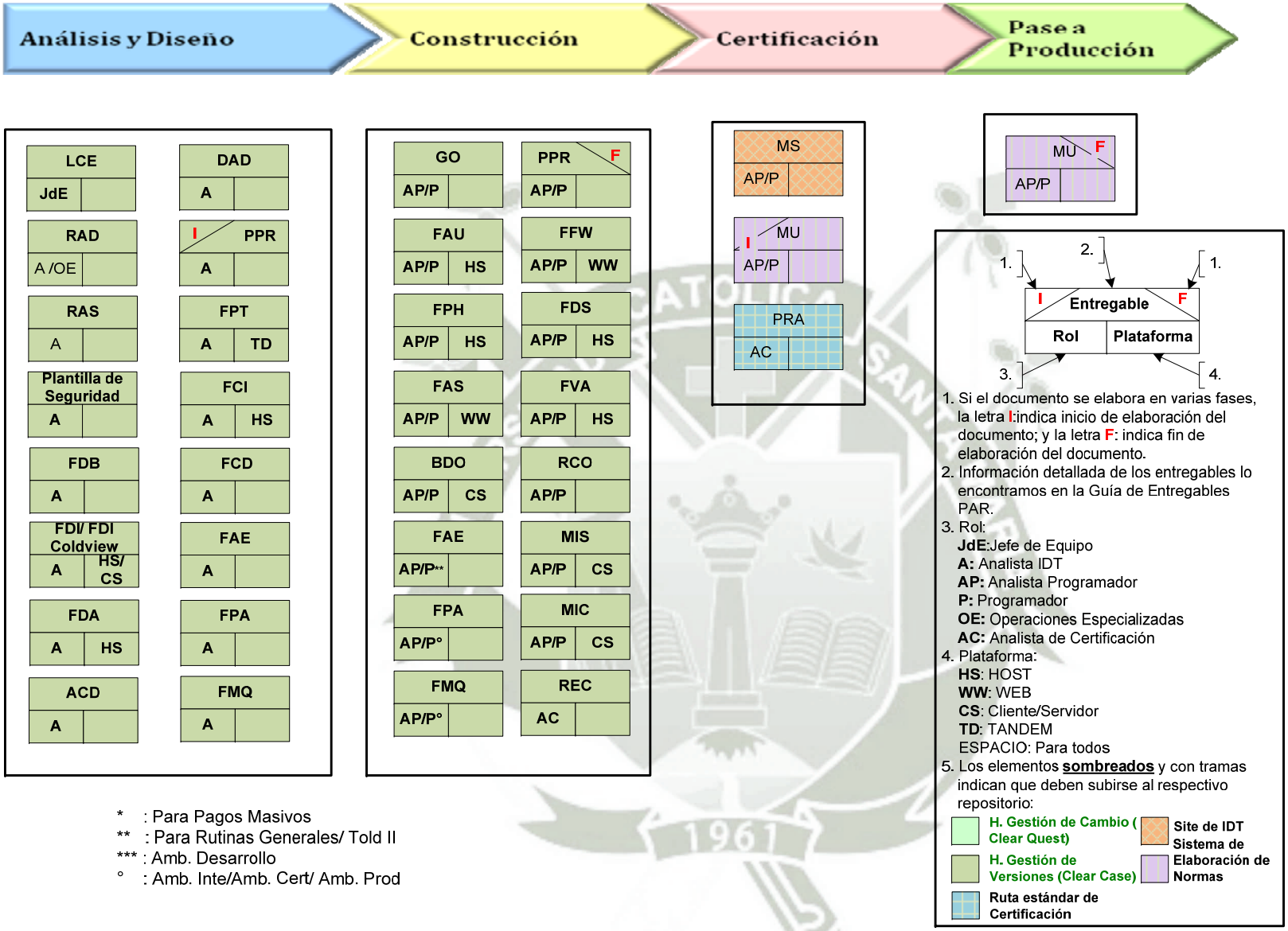
2.1.3. Certificación

En esta fase se ejecutan los casos de prueba para verificar que todo lo indicado en el DAD se desarrolló correctamente y que no se hayan impactado otras funcionalidades del aplicativo que debieron mantenerse. Esta fase es la que se encarga de verificar la calidad de lo desarrollado y también la calidad de los entregables generados para la implementación en certificación y producción.

2.1.4. Pase a Producción

Es la última fase del proceso PAR, y es la fase en la cual todos los cambios realizados pasan al ambiente del usuario y empiezan a funcionar en el área de negocio al cual corresponde el aplicativo modificado.

En el siguiente grafico se puede apreciar las fases del PAR, los entregables que se generan en cada una de las fases y los responsables de generar dichos entregables.



* : Para Pagos Masivos
** : Para Rutinas Generales/ Told II
*** : Amb. Desarrollo
° : Amb. Inte/Amb. Cert/ Amb. Prod

Figura Nº 2

2.2. Roles involucrados y sus funciones[URL02]

2.2.1. Analista de IDT (Funcional / Técnico)

PAR	
Análisis y Diseño	- Revisar la documentación generada durante el análisis de la necesidad para elaborar la documentación de análisis y diseño. - Solicitar Preparación de Ambiente de Desarrollo, en caso se requiera. - Proponer soluciones que se estén de acuerdo a los lineamientos y estándares definidos en el BCP. - Coordinar con otras unidades para evaluar el impacto del requerimiento-aplicativo. - Enviar al usuario la documentación y solicitar su aprobación. - Definir los casos de prueba de integración. - Coordinar la revisión de pares del documento de análisis y diseño y atender las observaciones. - Elaborar o actualizar la información de los documentos/plantillas de seguridad. - Elaborar los formatos AIO que sean necesarios, que haya identificado durante su análisis y que están bajo su responsabilidad, según lo definido en el proceso. - Elaborar, actualizar y versionar la documentación de la fase, según lo definido en el proceso. - Enviar a Seguridad la información del documento de análisis y diseño. - Congelar y versionar en la herramienta definida, los formatos AIO elaborados, como versión inicial, los cuales pueden ser modificados durante la construcción por el responsable de dicha fase. - Asesorar al proveedor, en caso sea este quien elabore la

	documentación de análisis y diseño. - Informar el inicio de la transferencia. - Transferir la documentación de análisis y diseño a las personas que participarán en la fase de construcción. - Informar sobre la finalización del documento de análisis y diseño. - Dejar el ambiente de desarrollo listo para la fase de construcción.
Construcción	- Revisar la documentación/formatos según lo definido en el proceso para esta fase.

2.2.2. Analista Programador / Programador

PAR	
Análisis y Diseño	- Realizar revisión de pares del documento de análisis y diseño. - Recibir transferencia del documento de análisis y diseño.
Construcción	- Codificar/programar unidades de programación de requerimientos, considera casos de prueba y lineamientos y estándares definidos en el BCP. - Realizar revisiones de pares de código y atender las observaciones. - Preparar la información/data para las pruebas unitarias. - Ejecutar pruebas unitarias. - Integrar componentes de la aplicación. - Ejecutar y registrar de Pruebas de integración (considera casos de Calidad de Software, casos de prueba que el analista técnico colocó en la herramienta de Gestión de CDS y casos que el mismo considere). - Generar/actualizar la documentación/formatos necesarios, según lo definido en el proceso, en la herramienta de versionamiento.

	- Proponer nuevos casos de Pruebas para la Certificación, si lo considera necesario. - Obtener transferencia del documento de Calidad de Software. - Preparar el congelamiento e informar el estado del mismo. - Corregir las observaciones recibidas luego de la solicitud de Preparación de Ambiente y Congelamiento, si los hubiera. - En caso el código sea realizado por el Proveedor, orientarlo durante esta fase, coordinar la revisión de pares que debe efectuar al proveedor y validar la calidad del código.
Certificación	- Corregir Defectos. - En caso ocurra un problema de ambiente en certificación en un requerimiento-aplicativo en curso, corregir el problema. Solicitar recongelamiento de corrección de defectos - Elaborar/Actualizar documentación de la fase, según lo definido en el proceso. - Informar en caso haya documentación desactualizada y/o incompleta.
Pase a Producción	- Coordinar reunión con unidades involucradas para revisar el plan de pase y reversión. - Verificar las conformidades. - Verificar que los formatos estén atendidos. - Brindar soporte post-producción y realizar el traspaso de entregables al Analista de Operaciones Especializadas (dentro del período de estabilización).

2.2.3. Analista Operaciones Especializadas

PAR	
Análisis y Diseño	- Estar informado de los cambios en el aplicativo a nivel funcional, debido a los distintos requerimientos. - Realizar revisión de pares (opcional de acuerdo a disponibilidad).
Construcción	- Elaborar la documentación de la fase. - Codificar o realizar los cambios. - Solicitar preparación de ambientes y congelamiento.
Certificación	- Corregir defectos y solicitar congelamiento para las correcciones. - Informar el fin del congelamiento planificado. - Solicitar preparación ambiente de producción.
Pase a Producción	- Asistir a reunión de transferencia durante el período de estabilización. - Estar informado con respecto a los cambios en los entregables. - Brindar soporte ante cualquier problema que suceda como consecuencia del pase a producción.

2.2.4. Analista de Calidad de Software

PAR	
Análisis y Diseño	- Revisar documento de análisis con el objetivo de conocer detalles sobre el alcance del requerimiento así como para despejar dudas, sugerir cambio o solicitar aclaraciones en el documento. Todo esto en el tiempo y momento del proceso. - Recibir la transferencia de información funcional y técnica. - Elaborar la documentación preliminar, según lo indicado

	en el proceso.
Construcción	- Elaborar e informar los lineamientos de pruebas. - Actualizar y enviar documentación para su aprobación, según lo definido en el proceso. - Actualizar los documentos/formatos en la herramienta de gestión de versiones y rutas definidas. - Realizar la transferencia del documento de análisis para la certificación. - Verificar los ambientes y estrategias de pruebas. - Notificar la finalización del documento de análisis que servirá para la certificación. - Elaborar, registrar y asignar casos de prueba.
Certificación	- Brindar asesoría relacionada a los casos de Pruebas - Coordinar la creación de los casos de pruebas para el siguiente ciclo. - Actualizar la asignación de los casos de prueba del siguiente ciclo. - Hacer seguimiento de Ejecución de Pruebas - Coordinar la solución de defectos. - Revisar la información de cierre de ciclo y conformidad de calidad de Software.
Pase a Producción	- Solicitar los accesos a producción para la ratificación. - Revisar el plan de pase a producción/reversión. - Coordinar la búsqueda de la data a ser utilizada en la Ratificación. - Coordinar la ratificación.

2.2.5. Certificador de Calidad de Software

PAR	
Construcción	- Leer la documentación que sirve para el análisis de certificación. - Revisar casos de Prueba asignados y registrar casos adicionales si lo considera necesario. - Identificar la data de prueba a utilizar. - Identificar los accesos y verificaciones de ambientes que necesita. - Solicitar la conformidad para los accesos. - Gestionar la configuración de ambiente de pruebas y coordinar posibles problemas que pudieran ocurrir. - Obtener data de prueba para la certificación. - Recibir capacitación de Herramientas. - Realizar revisión de pares del documento de análisis de Certificación.
Certificación	- Verificar en el ambiente de pruebas que los elementos/aplicaciones congeladas en el ambiente de certificación estén correctamente instaladas y operativas. - Preparar, seleccionar y adecuar la data asegurando que cumpla con las condiciones requeridas para la verificación de cada uno de los casos a probar. - Ejecutar los Casos de Pruebas en el ambiente de Certificación, realizando los escenarios establecidos en la secuencia definida, para dicho ciclo para la posterior verificación de resultados esperados. - Usar la herramienta definida para la ejecución de los casos de pruebas. - Documentar casos de prueba ejecutados y actualizar los estados de los mismos. - Notificar el fin de ciclo. - Elaborar conformidad de Calidad de SW y comunicar la

	culminación de los casos de prueba. - Completar/actualizar la documentación, según lo definido en el proceso.
Pase a producción	- Ejecutar los casos de prueba definidos para la etapa de ratificación. - Elaborar el informe de Ratificación. - Informar el término de las pruebas de ratificación.

CAPÍTULO III

INGENIERIA

3.1. Introducción

3.1.1. Antecedentes

La Herramienta HCR (Host Code Reviewer), es una aplicación encargada de validar el cumplimiento de los estándares de programación para programas COBOL, actualmente en su versión 2.3.

3.1.2. Objetivos

3.1.2.1. General

- Migrar y agregar nuevas opciones de validación de elementos a la herramienta HCR.

3.1.2.2. Específicos

- Migrar la herramienta HCR a C# manteniendo las mismas funcionalidades de la herramienta actual y utilizar SQL Server 2008 para el manejo de la Base de Datos.
- Adicionar la validación de estándares en Jobs Host.
- Adicionar la validación de estándares en Procedimientos Host
- Analizar, diseñar, desarrollar y documentar la implementación de las nuevas validaciones en HCR.

3.1.3. Descripción de la situación actual

Actualmente Utiliza 3 interfaces principales, desarrolladas en 2 lenguajes:

- Una DLL de nombre ErrorFinder.dll, la cual es la encargada de realizar el proceso de revisión de estándares propiamente dicho, dicha DLL se encuentra desarrollada en Visual C++ 6.0.
- Un "Agente" de conexión con Mainframe, para la recepción de los programas a ser validados, dicho agente utiliza una conexión XCOM, dicho agente se encuentra desarrollado en Visual Basic 6.0.
- Finalmente la interfaz grafica de la aplicación la cual se encuentra desarrollada en Visual Basic 6.0.

Se cuenta con una base de datos SQL 2000 que almacena información de reglas, validaciones, usuarios y puntajes.

3.1.4. Evaluación técnica preliminar

3.1.4.1. Descripción de alternativas de solución preliminares

Se propone 2 alternativas:

a) Basándose en el código actual migrar el mismo a un lenguaje estándar y moderno como lo es C#.**[URL03]**

b) Crear un nuevo aplicativo basándose en las funcionalidades del aplicativo actual, mas no en el código que se maneja actualmente. Esto en un lenguaje estándar y moderno como lo es C#.

3.1.4.2. Conclusiones derivadas de las evaluaciones técnicas preliminares.

Debido a la complejidad, manejo de estándares estático, errores existentes e inestabilidad del código actual, es que se optó por la alternativa 2, es decir, crear un nuevo aplicativo que tendrá las mismas funcionalidades (y algunas extras) de la versión de HCR actual, garantizando la estabilidad, el correcto manejo de estándares, así como la disminución del nivel de Soporte que actualmente se brinda a HCR (que es bastante elevado).

3.2. Descripción general de la solución

3.2.1. Descripción

En esta aplicación se crearán las funcionalidades necesarias para incluir mejoras, en la carga correcta de los elementos de una hoja de pase así como la correcta validación de la misma, la validación de reglas y una correcta descripción del error, la seguridad del aplicativo.

3.2.2. Descripción de los resultados de cada proceso

El proceso se inicia con la Carga de la hoja de pase, esto implica validar el archivo XML, los campos clave: **[URL04]**

Numero de ticket, las cabeceras de los elementos de la hoja y luego cada elemento como una unidad de programación.

En este proceso de carga de hoja de pase, se forma la librería donde debe ubicarse al elemento de acuerdo al Sistema, Subsistema y Tipo. En caso la hoja de pase tenga algún error léxico o no exista un Sistema o Subsistema, se enviara el mensaje de error correspondiente.

A continuación se muestran los diferentes Sistemas, Subsistemas y Tipos que existen definidos dentro del Banco para los elementos COBOL:

SISTEMA	SUBSISTEMA	TIPO
ADMIN	ABA	APACOBII
ADMPROPE	ABONHABE	APBCOBII
ADUANAS	AHORROS	APCCOBII
AHBAPE	ALS	BFSASMB
ALSBO	AM	BOTASMB
ALSCO	ANEXO5	BOTCOBII
ALSGNBO	ANEXO5S	CD2BA
ALSGNCO	ANEXO9	CD2BARXX
ALSGNPE	ANEXOS	CD2TP
ALSPE	ANEXOSFZ	CICPIN
AMQHCO	ASM	CO2BA
ANEXO5PE	ATM	CO2BAVP
ATOKCO	AUDITEXT	CO2BAVP1

BANKTRAD	AUDITSBS	CO2BAVP3
CAJERO	BANGIR	CO2BAVP4
CAJEROBO	BC	CO2SBVP
CAJEROCO	BCAPERS	CO2TP4
CAJEROPE	BCR	CO2TP4BT
CAMBCO	BKTD	CO2TP4SP
CANJECO	BONUPUNT	CO2TP4VP
CANJEPE	BRP	CO2TPTMQ
CATGPE	BTBCP	COMASMB
CBME	BTCSI	COMCDTII
CDPGCO	BTFUSION	COMCOB
CDPGPE	BTJPC	COMCOBII
CIF	BTJPCPRE	COPY
COMPVENT	BTVAR	CPYASMB
COMUNICA	CANJE	CPYCOBII
CONSISBO	CARDPAC	CPYMAP
CONSISCO	CARGA	CTCASMB
CONSISPE	CARGABON	CTCCOBII
CONSIST	CATG	JOB
CONTABBO	CBZ	JOB1
CONTABCO	CC	MACASMB
CONTABIL	CCARDPAC	MAPA
CONTABPE	CENTRAL	MAPASMB
CONTEMBO	CFONDO	OBJDATA
CONTEMCO	CMS	OBODATA
CONTEMPE	COL	PGMASM
CONTEMPO	COMPAGO	PGMASMB
CONTINGE	CONCILIA	PGMCDTII
CONTINPE	CONTEMP	PGMCOB
CORPORAT	CONTEN	PGMCOBDS
COVEBO	CONVERSI	PGMCOBII
COVECO	COVE	PGMCOBSB
COVEPE	CP	PROC
CREDCARG	CT	RELASMB
CREDPAGO	CTA	ROLASMB
CRMPE	CTAPLAZO	RPICOBII
CRWRKFLW	CTS	TOLASMB
CTASCTES	CUADRE	
CUADRE	CUSTOMIZ	
CUADRECO	CV	
CUADREPE	DEU	
CUSTODIA	DEUMENOR	
DATAWARE	DRIVER	
DWEDWPE	EDITPAC3	
DWETLCO	EDITPACK	

DWETLPE	ENCAJE	
DYNAMICO	ETLGEN	
ECSWCO	ETLVPL	
ECSWPE	EXCL	
ENCAJECO	EXDAT	
ENCAJEPE	EXPEXT	
ERPCO	FAS	
ERPPE	FONDSEGU	
EXPEXTPE	FUE	
FCTELEPE	FUSION	
FEDCO	GAFI	
FEDPE	GAMM	
FILNEGBO	GAP	
FILNEGCO	GAR	
FILNEGPE	GENERAL	
FINANZAS	GL	
FONSEGPE	GNS	
FUECO	HOST	
FUEPE	IATA	
GENERABO	ICC	
GENERACO	IGC	
GENERAL	IM	
GENERAPE	INHOUSE	
GESTION	INTERFAS	
GGYTTBO	INTERFAZ	
GGYTTCO	INTG	
GGYTTPE	ISP	
GIROSYTT	ITS	
HABERES	LAVADINE	
HCR	LDC	
IDS	LEAS	
IDSPE	LET	
IMPACSBO	LH	
IMPACSCO	LIQ	
IMPACSPE	MAESTRO	
INFCONTR	MASIVO	
INFCTRPE	MBS	
ITBO	MCCAMBIO	
ITCO	MCCUSVAL	
ITPE	MDCSMFMU	
IVCO	MDCSMGRL	
KOBO	MICTRANS	
KOCO	MILL	
KOPE	MONEDERO	
KSYSTE	MOTOR	

LAVADO	MQ	
LEASING	NEWTLC	
LETHIPO	NMPM	
LETRAS	NOINTERF	
LETRASCO	NOTAPEND	
LIQTESPE	NSAT	
LIQUIDAC	NX	
LMTECO	OLA	
MCAPITAL	OPECRUZA	
MDCSUICO	PACI	
MDCSUIPE	PAGOS	
MDOCAPPE	PASE	
MEDELEBO	PC	
MIGCOBR4	PH	
MPAGMABO	POS	
MPAGMACO	PRDCAJER	
MPAGMAPE	PRICE	
MPMGNCO	PROVEED	
MSCO	RCD	
MSPE	REGISTRO	
NOTASBO	RENGLNPC	
NOTASCO	RENTVSGL	
NOTASPE	REPO14	
NXBO	RTBCP	
NXCO	RTC	
NXPE	RTJPC	
ORQSCO	RTVAR	
ORQSPE	SAP	
PAGONET	SAPCIF	
PAGOSERV	SAPSYST	
PARTCIUD	SAT	
PASEPE	SBS	
PEADMPRO	SCBME	
PEADUANA	SEGURIMA	
PEBANKTR	SGC	
PEBCPIAT	SIN	
PEBONUS	SMS	
PEBRPIAT	SNT	
PECAJERO	SSC	
PECBME	ST	
PECIF	STANDARD	
PECOMUNI	SWBCP	
PECOMVEN	SWIFT	
PECONSIS	SWJPC	
PECONTAB	TAB	

PECONTEC	TABLAS	
PECONTEM	TARJCRED	
PECONTFZ	TC	
PECREDCR	TELEPROC	
PECREDPG	TEST	
PECRWKFW	TII	
PECTACTE	TOLDII	
PECUADRE	TRAINT	
PECUSTOD	TRANELEC	
PEDATAWR	TRASPASE	
PEEDITRA	TRM	
PEFINANZ	TSDIR	
PEGENRAL	ULTIMAS	
PEGESTIO	ULTIMOD	
PEGIRYTT	ULTIMOM	
PEHABERE	UNICO	
PEINFCTR	USOGEN	
PEINFOPA	UTIL	
PELAVDIN	VALISALD	
PELEASIN	VANILLA	
PELETHIP	VAR	
PELETRAS	VARIOS	
PELIQUID	VPLUS	
PEMDOCAP	XNTERFAZ	
PEMECCO	XOINTERF	
PEMECPE	XROVEED	
PEMIC		
PEPAGNET		
PEPAGSRV		
PEPARCIU		
PEPNDDC		
PEPOS		
PEREMITT		
PERENTAB		
PERHUMAN		
PESAP		
PESEGMEN		
PESUNAT		
PESWIFT		
PETELCRE		
PETOLDII		
PETRJCRE		
PETRNINT		
PEVARIOS		
PEVIABCP		

PEVPLUS		
PFTJBO		
PFTJCO		
PGNTPE		
PNDDC		
POBLAMCO		
POBLAMPE		
POS		
POSBO		
POSCO		
POSPE		
POSSCO		
POSSPE		
RCDPE		
RCPSCO		
RCPSPE		
REMICO		
REMIPE		
REMITTAN		
RENTABCO		
RENTABIL		
RENTABPE		
REPEXBO		
REPEXOPE		
REPEXPE		
RGENCO		
RHUMANOS		
RMCO		
RTCPE		
SALCO		
SALPE		
SAP		
SAPBO		
SAPCO		
SAPPE		
SATBO		
SATCO		
SATPE		
SAVINGBO		
SAVINGCO		
SAVINGEB		
SAVINGEP		
SAVINGPE		
SEGMENTA		
SEGURINF		

SERVICE		
SUNAT		
SUNATPE		
SWIFT		
SYSTCO		
SYSTEMAT		
SYSTPE		
TARJCRE3		
TARJCRED		
TBIECO		
TELCRECO		
TELCREPE		
TELECRED		
TOLDGNPE		
TOLDII		
TOLDIIBO		
TOLDIICO		
TOLDIIPE		
TRNICO		
TRNIPE		
TSBO		
TSCO		
TSEB		
TSEP		
TSPE		
VIABCP		
VPLCZ2CO		
VPLUS		
VPLUS2CO		
VPLUSBO		
VPLUSCO		
VPLUSPE		
WLEASPE		

Para la verificación de los programas:

a) Se divide el código en DIVISIONES utilizando una expresión regular.

b) Tomando como base las divisiones se divide en SECCIONES con otra expresión regular.

c) Luego de tener las secciones, se cargan las reglas con sus respectivas expresiones regulares y estas a su vez sus respectivas expresiones regulares de error**.[URL05]**

d) Validar regla, para este paso se entrega la SECCION a la regla, esta es evaluada y devuelve los errores.

Para la verificación de JOBs y Procedimientos se toma todo el código del elemento transferido, obviando las líneas de comentarios, ya que la verificación abarca solo la parte de código.

Las reglas de validación se obtienen desde la tabla de reglas.

3.2.3. Diagrama de Proceso

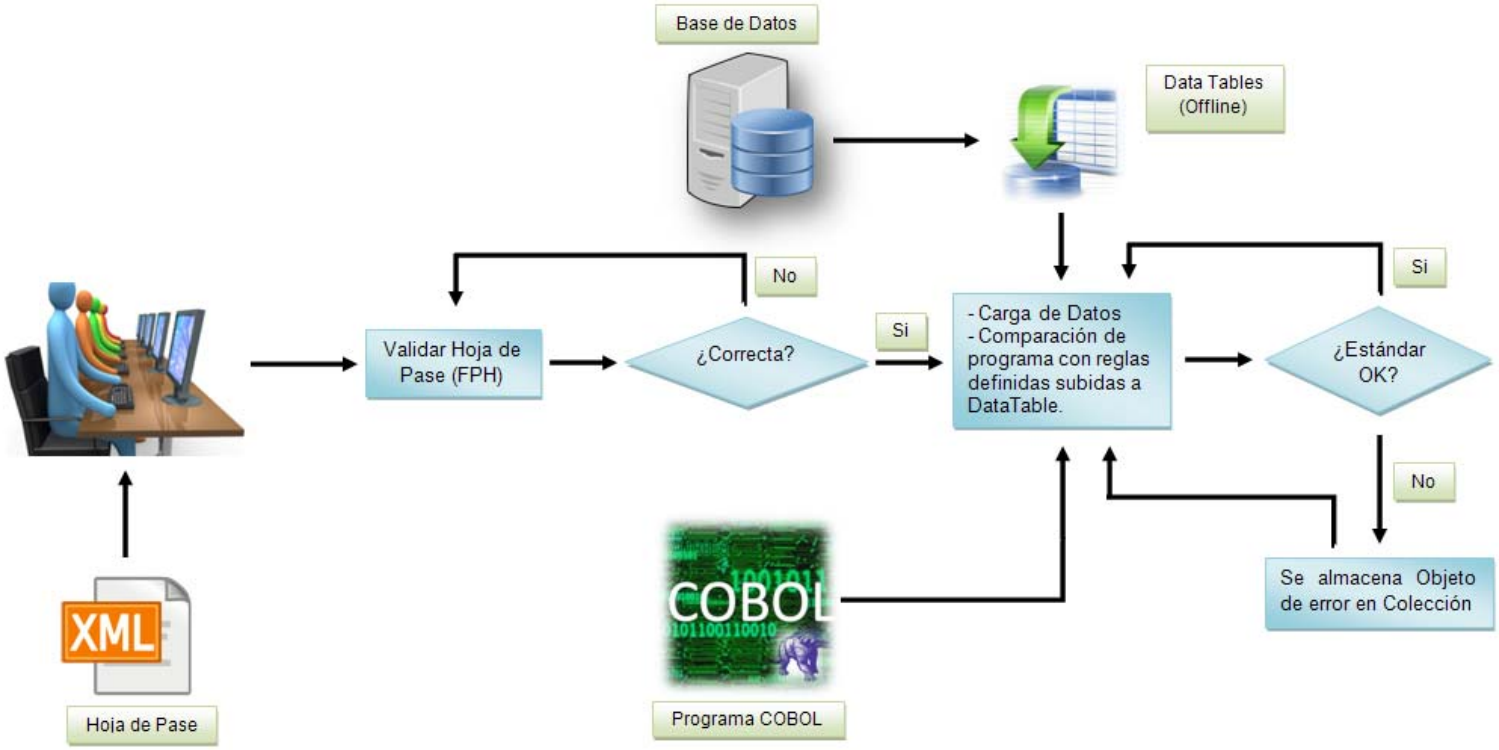


Figura Nº 3

3.2.4. Descripción del proceso de Carga de Hoja de Pase

En el proceso intervendrá el usuario HCR, la hoja de pase deberá tener el formato correcto. De lo contrario se avisará con un mensaje de error al usuario.

"Formato de hoja de pase no valido"

La secuencia operativa será:

a) El usuario HCR ingresara a la opción "Cargar Hoja de Pase" para ello podrá hacer uso de un Menú principal donde figurará esta opción, o directamente desde un botón ubicado bajo el menú principal.
b) Se abrirá un cuadro de selección de elementos (Abrir), solo elementos del tipo XML. El aplicativo deberá validar que el tipo de archivo a abrir es el correcto.

3.2.5. Validación de la hoja de pase

Proceso transparente para el usuario en el que se validará que la hoja de pase sea válido.

3.2.5.1. Descripción de Validación de hoja de pase

Luego de seleccionar la hoja de pase a cargar y validar que sea un archivo del tipo XML, el aplicativo deberá validar que la hoja de pase esté alineada con el último formato dado por Arquitectura e integración Tecnológica.

Esto implica:
Verificar que el numero de ticket, elemento, sistema, subsistema y demás columnas y filas de la hoja de pase se encuentren en la posición correcta, tal y como el formato de Arquitectura lo especifica, de lo contrario, la hoja de pase no es válida y no se permitirá su carga y se avisará con un mensaje de error al usuario.

"La Hoja de Pase no es válida"

3.2.6. Validación de los elementos de la hoja de pase

Se deberá validar que los elementos incluidos en la hoja de pase sean programas, Jobs y Procedimientos, además de que estos pertenezcan a un sistema y subsistema correcto.

3.2.6.1. Descripción del Proceso Validación de los elementos de la hoja de pase

Este proceso deberá ser transparente para el usuario, y se ejecutará inmediatamente después de hacer la validación de la hoja de pase (en caso esta sea válida).
Se validará que el sistema y subsistema indicados en la hoja de pase existan y sean los correctos para el elemento.

El aplicativo deberá filtrar únicamente los elementos de los tipos Job, Procedimiento y Programa.

Y no se tomarán en cuenta CTL's ya que a este tipo de elementos no es necesario hacerles la correspondiente validación de estándares.

El aplicativo deberá identificar la librería en la que se encuentra el elemento, basándose para ello en el sistema y subsistema indicados en la hoja de pase.

3.2.7. Transferencia de los elementos

Los elementos a ser validados por la Herramienta deberán ser transferidos a PC desde Mainframe y también se pueden tomar los elementos que se tengan en la PC.

3.2.8. Alcances

3.2.8.1. Alcance de la presente versión [CEBA92]

- Carga de Hoja de pase.
- Validación de la hoja de pase.
- Validación de los elementos de la hoja de pase.
- Validar Contenido.
- Llenar árbol con los elementos de la hoja de pase validadas.

- Validación de Errores.
- División de archivo COBOL.
- Administración de la conexión con la base de datos.
- Cargar valores hoja de Pase
- Validación de Errores.
- Obtención de Promedio.

3.2.9. Funciones Principales [JOYA96]

3.2.9.1. Carga de Hoja de pase:

Nombre / Clase:	***Cargar HojaXML***
Descripción/ Objetivo:	Carga los datos del archivo XML para ser comparados contra el modelo de la base de datos.
Entradas:	String **Nombre Archivo**, Entero **Nro Hoja**
Lógica de programa (host) / Métodos:	1. Cargar HojaXML New Version 2. Cargar Unidades XML
Salidas:	
Interfaces usadas:	

3.2.9.2. Validación de la hoja de pase:

Nombre / Clase:	***ValidarHojaXML***
Descripción/ Objetivo:	Valida el contenido y la estructura de la hoja de pase.
Entradas:	string **NombreArchivo**, int **NroHoja**
Lógica de programa (host) /	

Métodos:	
1. CargarHojaXML 2. ValidarXML	
Salidas:	CodigoValidacion
Interfaces usadas:	

3.2.9.3. Validación de los elementos de la hoja de pase:

Nombre / Clase:	***ValidarEstructura***
Descripción/ Objetivo:	Valida la estructura de la hoja de pase: Elemento, Sistema, subsistema, tipo, número de ticket.
Entradas:	
Lógica de programa (host) / Métodos:	
1. LeerCelda	
Salidas:	Int **Estado**
Interfaces usadas:	

3.2.9.4. Validar Contenido:

Nombre / Clase:	***ValidarContenido***
Descripción/ Objetivo:	Valida que exista el numero de ticket, los elementos, sistemas, subsistemas y tipos.
Entradas:	
Lógica de programa (host) / Métodos:	

1. ValidarUnidad 2. ObtenerElemento 3. EstablecerUbicacion	
Salidas:	Entero **Estado**
Interfaces usadas:	

3.2.9.5. Llenar árbol con los elementos de la hoja de pase validadas:

Nombre / Clase:	***CargarArbol***
Descripción/ Objetivo:	Obtiene las unidades de la hoja de pase y las va almacenando en los nodos del árbol en la interfaz.
Entradas:	clsHojaPase **HojaDePase**
Lógica de programa (host) / Métodos:	1. GetTicket 2. GetUnidades 3. ObtenerSistema 4. EstadoValidacion 5. ObtenerElemento
Salidas:	
Interfaces usadas:	

3.2.9.6. Validación de Errores

Nombre / Clase:	***VerficarEstandar***
Descripción/ Objetivo:	Verifica que se hayan cumplido los estándares de programación del programa cobol.
Entradas:	string **xstrCodFue**, bool **bolpractica**
Lógica de	Se basa en las expresiones regulares ingresadas en

programa (host) / Métodos:	Base de Datos, para validar los estándares. Para ello el programa deberá partirse en 4 (las 4 divisiones de un programa cobol) y validará para cada división únicamente las reglas que pertenecen a la misma.
1. DividirParCodFue 2. ValidarRegla 3. ObtenerReglasXDivision 4. clsErrorRegla	
Salidas:	Arraylist de errores
Interfaces usadas:	

Nombre / Clase:	***VerficarEstandar***
Descripción/ Objetivo:	Sobrecarga del anterior método que verifica los estándares de programación del programa cobol.
Entradas:	string **xstrCodFue**, string **xstrCodFueProd** , bool **bolpractica**
Lógica de programa (host) / Métodos:	Funciona de la misma forma que el anterior método , con la diferencia que solo ejecuta las reglas correspondientes a la División especificada por el segundo parámetro.
1. ObtenerExpRegCorte 2. DividirParCodFue 3. FiltrarCodigo 4. ValidarRegla 5. ObtenerReglasXDivision 6. clsErrorRegla	
Salidas:	Arraylist de errores
Interfaces usadas:	

3.2.9.7. División de archivo COBOL:

Nombre / Clase:	***ObtenerExpRegCorte***
Descripción/ Objetivo:	Esta función obtiene las expresiones regulares para efectuar los cortes por división del código, haciendo uso del DataTable.
Entradas:	*Entero* ***xintNroDivsion***
Lógica de programa (host) / Métodos:	
Salidas:	ArrayList ***expresionesCorte***
Interfaces usadas:	

3.2.9.8. Administración de la conexión con la base de datos

Nombre / Clase:	***RealizarConexion***
Descripción/ Objetivo:	Carga la cadena de conexión de acuerdo al servidor que se le asigne como parámetro.
Entradas:	string **servidor**
Lógica de programa (host) / Métodos:	
Salidas:	
Interfaces usadas:	

Nombre / Clase:	***VerificarSistema***
Descripción/ Objetivo:	Verifica que el nombre del sistema especificado en la hoja de pase exista en la base de datos.

Entradas:	string **sistema**
Lógica de programa (host) / Métodos:	
1. AbrirConexion	
Salidas:	Entero **id**
Interfaces usadas:	

Nombre / Clase:	***verificarSubSistema***
Descripción/ Objetivo:	Verifica que el nombre del subsistema especificado en la hoja de pase exista en la base de datos.
Entradas:	string ***subsistema***, int ***idSistema***
Lógica de programa (host) / Métodos:	
1. AbrirConexion	
Salidas:	Entero **id**
Interfaces usadas:	

Nombre / Clase:	***verificarTipo***
Descripción/ Objetivo:	Verifica que el nombre del tipo especificado en la hoja de pase exista en la base de datos.
Entradas:	string **tipo**
Lógica de programa (host) / Métodos:	

1. Abrir Conexion	
Salidas:	Entero
Interfaces usadas:	

Nombre / Clase:	***BuscarLibreria***
Descripción/ Objetivo:	Busca el nombre de la librería para concatenar con el nombre del elemento.
Entradas:	int **IdSistema**, int **IdLibreria**
Lógica de programa (host) / Métodos:	1. BuscarLibreria
Salidas:	String **Librería**
Interfaces usadas:	

Nombre / Clase:	***BuscarReglas***
Descripción/ Objetivo:	Busca las reglas que le corresponden a una división en la base de datos
Entradas:	int **NroDivision**
Lógica de programa (host) / Métodos:	1. SP BuscarLibreria
Salidas:	ArrayList **Reglas**
Interfaces usadas:	

Nombre / Clase:	***CargarExpresiones***
Descripción/ Objetivo:	Carga la tabla de expresiones regulares de la Base de Datos y lo almacena en un DataTable.
Entradas:	
Lógica de programa (host) / Métodos:	1. SP CagarExpresiones
Salidas:	DataTable **TablaExpresiones**
Interfaces usadas:	

3.2.9.9. Cargar valores hoja de Pase

Nombre / Clase:	***LeerNroTicket***
Descripción/ Objetivo:	Extrae el número de ticket de la hoja de pase.
Entradas:	string **LetraColunma**, int **NroFila**
Lógica de programa (host) / Métodos:	
Salidas:	String
Interfaces usadas:	

Nombre / Clase:	***CargarUnidades***
Descripción/ Objetivo:	Devuelve un array de los elementos de la hoja de pase
Entradas:	

Lógica de programa (host) / Métodos:	

1. UnidadPrograma
2. CargarUnidad

Salidas:	
Interfaces usadas:	

Nombre / Clase:	***FinHoja***
Descripción/ Objetivo:	Identifica cual es el fin de hoja
Entradas:	int **Fila**, string **Columna**, int **NroElementos**
Lógica de programa (host) / Métodos:	
Salidas:	Bool
Interfaces usadas:	

Nombre / Clase:	***CargarValorEtiqueta***
Descripción/ Objetivo:	Carga el numero de ticket
Entradas:	String **Columna**, int **Fila**
Lógica de programa (host) / Métodos:	

Salidas:	String
Interfaces usadas:	

Nombre / Clase:	***CargarCabeceras***
Descripción/ Objetivo:	Carga las cabeceras modelo a partir una fila, columna y numero de cabeceras
Entradas:	String **Columna**, int **Fila**, int **NroCabeceras**
Lógica de programa (host) / Métodos:	
Salidas:	string[]
Interfaces usadas:	

Nombre / Clase:	***ValidarUnidad***
Descripción/ Objetivo:	Carga las cabeceras modelo a partir una fila, columna y numero de cabeceras
Entradas:	clsConexionBD **Administrador**
Lógica de programa (host) / Métodos:	1. VerificarTipo 2. ObtenerTipoLibreria 3. VerificarSistema 4. VerificarSubSistema
Salidas:	Int
Interfaces usadas:	

3.2.9.10. Validación de Errores

Nombre / Clase:	***VerificarEstandar***
Descripción/ Objetivo:	Validará los estándares de programación del programa cobol
Entradas:	**Cadena** a revisar, **división** / **sección**
Lógica de programa (host) / Métodos:	Se basa en las expresiones regulares ingresadas en Base de Datos, para validar los estándares. Para ello el programa deberá partirse en 4 (las 4 divisiones de un programa cobol) y validará para cada división únicamente las reglas que pertenecen a la misma.
Salidas:	Arraylist de errores
Interfaces usadas:	

3.2.9.11. Obtención de Promedio

Nombre / Clase:	***ObtenerPromedio***
Descripción/ Objetivo:	Obtendrá el promedio de estándares por programa
Entradas:	Array con peso de las reglas, numero de líneas evaluadas
Lógica de programa (host) / Métodos:	El promedio se obtiene de la formula:

(∑ del peso de las reglas encontradas / Total de líneas evaluadas) * 100

De esta fórmula se obtendrá un valor, el cual para efectos de aprobar o no al elemento en sus estándares, deberá ser evaluado de la siguiente forma.

Promedio = 0 el elemento en el árbol de la parte izquierda se pintará de verde y estará aprobado.

Promedio mayor a 0 y menor que 20 el elemento en el árbol de la parte izquierda se pintará de amarillo y estará aprobado. Promedio igual o mayor que 20 el elemento en el árbol de la parte izquierda se pintará de rojo y estará desaprobado. *En caso el promedio obtenido sea mayor a 100 se deberá mostrar como igual a 100, es decir la nota promedio más alta que se puede dar es de 100.*	
Salidas:	Numeric
Interfaces usadas:	

Los métodos que serán expuestos desde nuestra Clase **HCR** son:

public ArrayList VerificarEstandar (string xstrCodFue)

public ArrayList VerificarEstandarJob(string xstrCodFue)

public ArrayList VerificarEstandarProc(string xstrCodFue)

Parámetros:

xstrCodFue: Que representa(en esta sobrecarga) todo el código cobol, de un elemento.

public ArrayList VerificarEstandar (string xstrCodFue, int xstrCodFueProd)

public ArrayList VerificarEstandarJob(string xstrCodFue, int xstrCodFueProd)

public ArrayList VerificarEstandarProc(string xstrCodFue, int xstrCodFueProd)

Parámetros:

xstrCodFue: Representa el fragmente de código cobol perteneciente a una División.

xstrCodFueProd: Representa el número de división propietario del código.

3.2.10. Base de Datos: [DATE01]

3.2.10.1. Definición de entidades y atributos

CONTADOR

Atributo	Tipo	Longitud
IDCONTADOR	int	
MATRICULA	nchar	6
FECHA	datetime	

ESTRUCTURAHOJAPASE

Atributo	Tipo	Longitud
IDESTRUCTURA	Int	
DESCESTRUCTURA	varchar	100
POSCFILA	Int	
POSCCOLUMNA	Char	1
NROSECUENCIA	numeric	
TIPO	Char	1

DIVISION

Atributo	Tipo	Longitud
IDDIVISION	int	
NOMBREDIVISION	char	32

LIBRERIAHOST

Atributo	Tipo	Longitud
IDLIBRERIA	int	
DESCLIBRERIA	char	64
IDSISTEMA	int	
IDTIPOLIBRERIA	int	

ESTADO

Atributo	Tipo	Longitud
IDESTADO	char	1
DESCRIPCION	char	10

PALABRARESERVADA

Atributo	Tipo	Longitud
IDPALABRA	Int	
DESCPALABRA	varchar	100
IDDIVISION	Int	
IDSECCION	Int	
POSCFILAPALABRA	Int	
POSCCOLUMPALABRA	Int	

EXPRESIONREGULAR

Atributo	Tipo	Longitud
IDEXPRESION	int	
IDDIVISION	int	
IDREGLA	int	
NROSECUENCIA	int	
EXPREG	varchar	1000
IDTIPO	varchar	3
IDEXPREGORI	int	
MSJEXPREG	varchar	1000

PARAMETROS

Atributo	Tipo	Descripción
IDPAR	int	
NOMBRE	varchar	20
VALOR	varchar	255
DESCRIPCION	varchar	255

REGLA

Atributo	Tipo	Longitud
IDREGLA	Int	
IDDIVISION	Int	
DESCREGLA	Char	100
PESOREGLA	Int	

SECCION

Atributo	Tipo	Longitud
IDSECCION	int	
IDDIVISION	int	
IDPALABRA	int	
NOMBRESECCION	char	32

SISTEMA

Atributo	Tipo	Longitud
IDSISTEMA	Int	
DESCSISTEMA	char	32
POSCFILASISTEMA	Int	
POSCCOLUMSISTEMA	char	1

SUBSISTEMA

Atributo	Tipo	Longitud
IDSUBSISTEMA	int	
DESCSUBSISTEMA	char	32

IDSISTEMA	int	
POSCFILASUBSISTEMA	int	
POSCCOLUMSUBSISTEMA	char	1

TIPOEXPRESION

Atributo	Tipo	Longitud
IDTIPO	Varchar	3
DESCTIPO	Char	50

TIPOARCHIVO

Atributo	Tipo	Longitud
IDTIPO	Int	
EXTENSION	Char	8
IDTIPOLIBRERIA	Int	

TIPOLIBRERIA

Atributo	Tipo	Longitud
IDTIPOLIBRERIA	Int	
DESCTIPOLIBRERIA	Char	16

VERIFICACION

Atributo	Tipo	Longitud
IDVERIFICACION	Int	
NUMEROTICKET	Varchar	15
ELEMENTO	Varchar	10
FECHAVERIF	Varchar	30
PROMEDIO	Real	
IDESTADO	Char	1
TOTALLINEAS	Int	
PROMVOL	Real	

CAPÍTULO IV

IMPLEMENTACIÓN DEL SISTEMA

4.1. Hoja de pase

La Hoja de Pase es el elemento principal para poder trabajar con HCR. Con este se puede crear proyectos para realizar la verificación de estándares de programación COBOL. El formato debe de tener el formato estándar, es decir tener la cabecera de la lista de elementos como se muestra en la **Figura Nº 4:**

ID	Cambio	Elemento	System	Subsys	Tipo	Estado N/M	Elementos para	Institución	Número de elementos que contiene (Este Campo sólo se habilita para JCL(Jobs, Proc y CTL) y Mapas y es obligatorio)
	Sin filtro...					Sin filtro...		Sin Filtro	
1	Si Congelar	BCXRTP11	CAMBCO	INHOUSE	COMCOBII	M	No adecuación	Corporativo	

Figura Nº 4

Para indicar que un elemento tiene una versión anterior debe de colocar al inicio del campo de **ESTADO** la letra **M** de Modificado en la Hoja de Pase en caso de que sea un elemento nuevo deberá colocar la letra **N** de Nuevo después debe guardar y actualizar el proyecto desde la Hoja de Pase modificada.

4.2. Ingreso al sistema

a) Se tiene dos opciones básicas para ingresar a la aplicación:

- Opción 1: Busque en la parte inferior izquierda de su pantalla, el botón Start, Programs, HCR 2.3 y seleccione la opción "HCR 2.3" como se muestra en la **Figura Nº 5**.

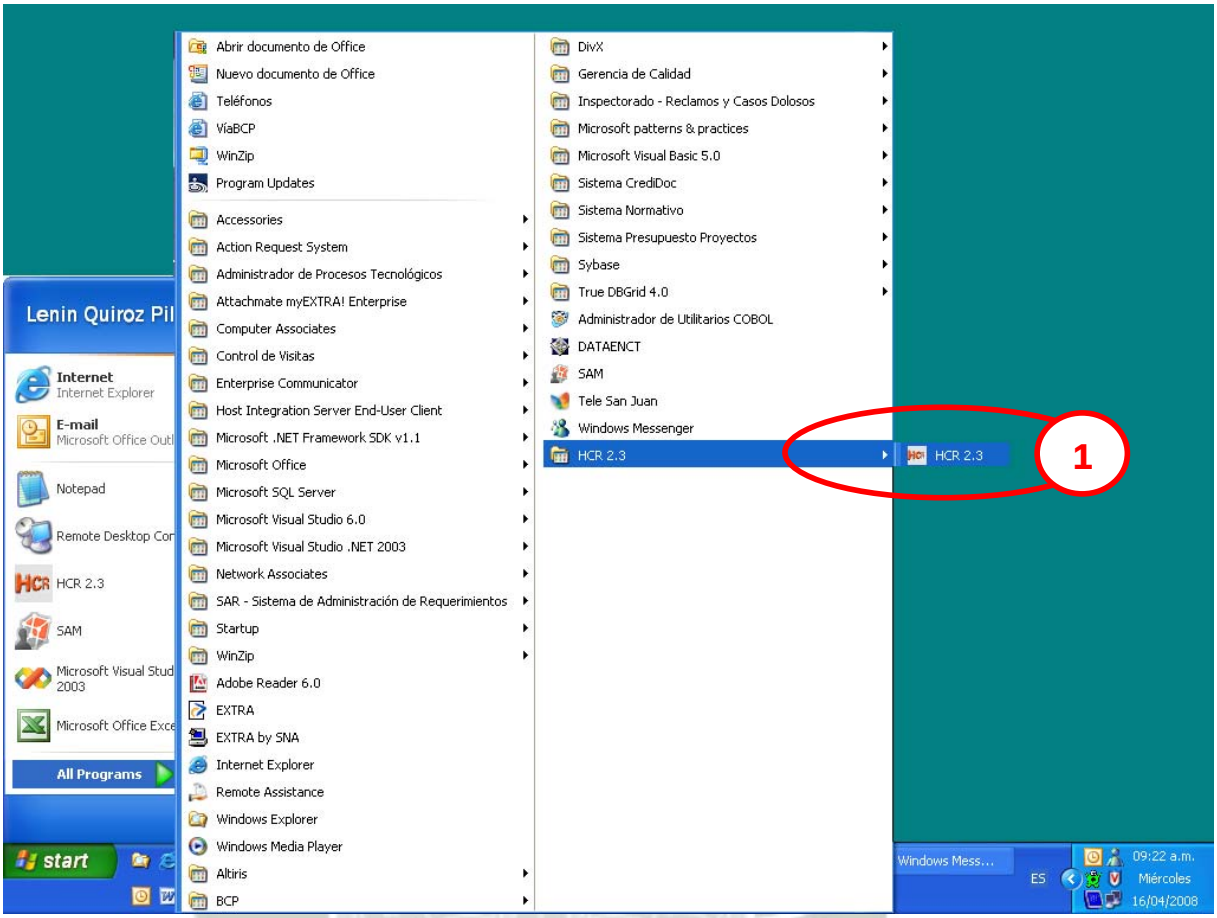


Figura Nº 5

- Opción 2: Seleccione el acceso directo de HCR 2.3 desde el desktop.

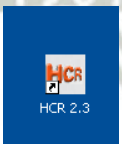


Figura Nº 6

Codificación

Al momento de ingresar al aplicativo se realiza la conexión a la Base de Datos “HCR” mediante la siguiente función:

```
public void RealizarConexion(string servidor,string
basedatos,string usuario, string contraseña)
{
 conexion=new SqlConnection();
cadenaConexion= "Persist Security Info=False;User ID=" + usuario +
";password= " + contraseña + ";Initial Catalog=" + basedatos +
";Data Source="+ servidor+ ";";
 conexion.ConnectionString=cadenaConexion;
}
```

4.3. Pantalla Principal

Barra de herramientas con todas las opciones que también podemos encontrar en el menú de la aplicación.

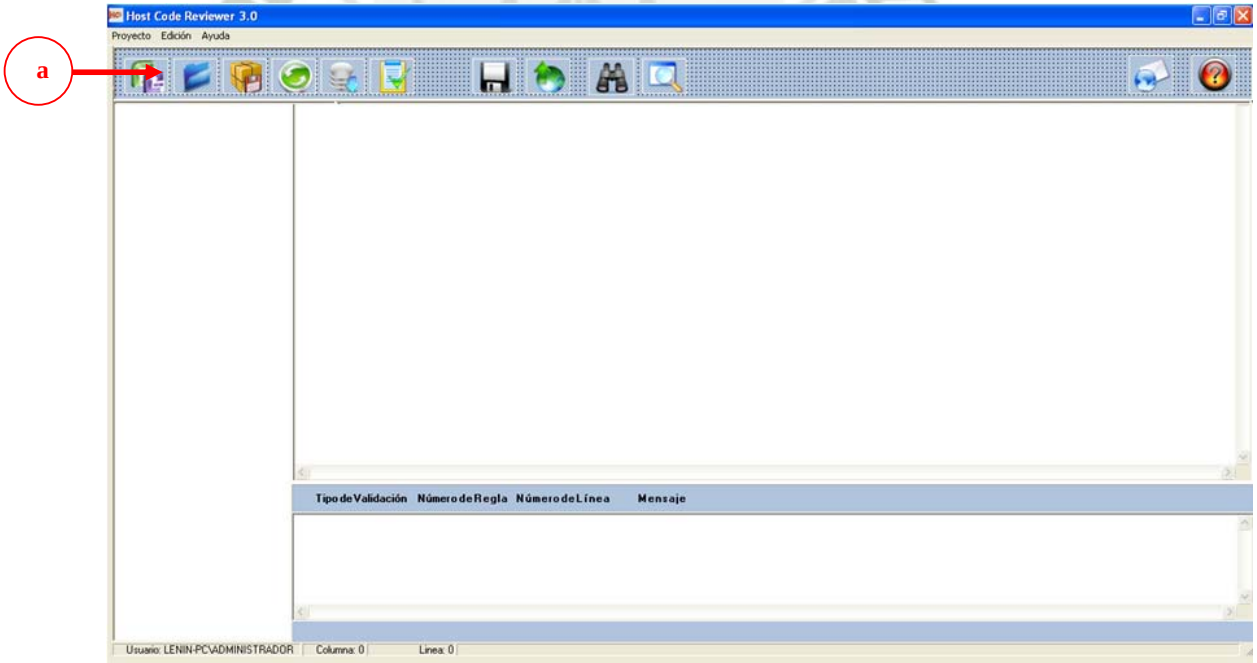


Figura Nº 7

Las opciones que se encuentran son las siguientes:

- **Creación de un proyecto HCR[KEND11]**

  Un proyecto se crea a través de una Hoja de Pase. Seleccione la opción **Proyecto**, **Cargar Hoja de Pase** y aparecerá el diálogo para abrir la hoja de pase a partir de la cual se creará el proyecto.

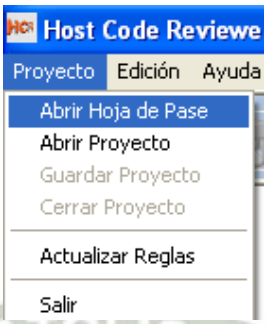


Figura Nº 8

También puede hacer clic en el icono  para abrir el diálogo.

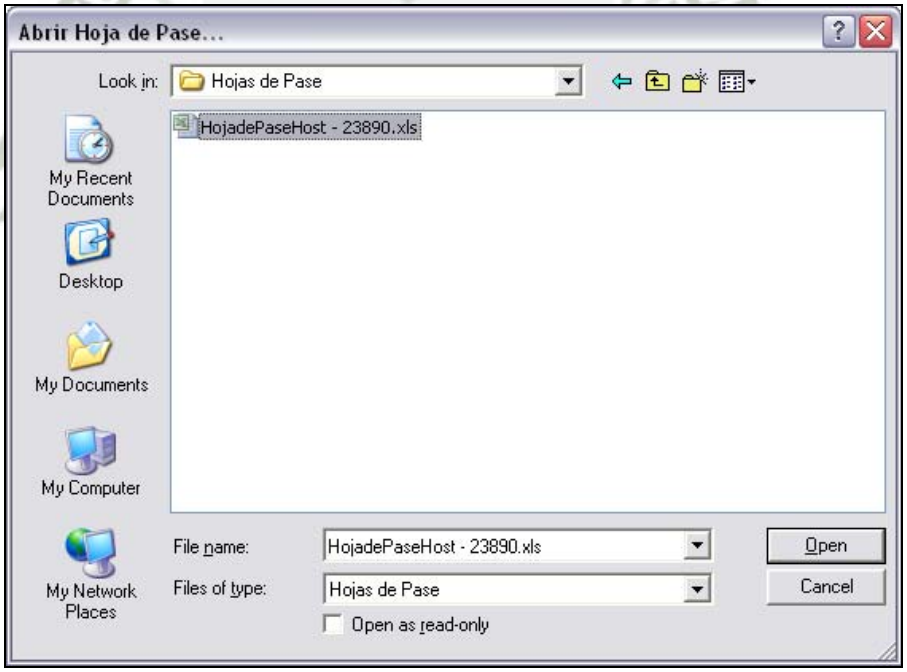


Figura Nº 9

Codificación

Una vez seleccionado el documento XML se llama a la función ValidarHojaXML(), para validar el documento y verificar que cumpla con la estructura estándar definida en la empresa:

```
private void validarHojaXML()
{
      string xstrErrores = String.Empty;
      this.BarraProgreso.Visible=true;
      int Estado =
      NucleoHCR.ValidarHojaXML(RutaArchivo,1,ref
      this.BarraProgreso);

      if(Estado.Equals(2201))
      {
            MessageBox.Show("No se puede cargar el FPH
            porque no existe número de ticket", "HCR",
            MessageBoxButtons.OK,
            MessageBoxIcon.Information);
      }
      else if(Estado.Equals(2205))
      {
            MessageBox.Show("No se puede cargar el FPH
            porque no hay elementos para validar", "HCR",
            MessageBoxButtons.OK,
            MessageBoxIcon.Information);
      }
      else
      {
            CargarArbol(NucleoHCR.ObtenerHojaPase());
            if(Estado.Equals(2202))
            {
                  ArrayList aErrores =
                  NucleoHCR.ObtenerHojaPase().ObtenerUnidad
                  es();
                  for(int i=0;i<aErrores.Count;i++)
                  {
                        if(((BibliotecaClases.UnidadProgram
                        a)aErrores[i]).EstadoValidacion.Equ
                        als(-1))
```

```
                    xstrErrores +=
                    ((BibliotecaClases.UnidadPrograma)a
                    Errores[i]).ObtenerElemento() + " -
                    no se encuentra sistema\n";

                    if(((BibliotecaClases.UnidadProgram
                    a)aErrores[i]).EstadoValidacion.Equ
                    als(-2))
                    xstrErrores +=
                    ((BibliotecaClases.UnidadPrograma)a
                    Errores[i]).ObtenerElemento() + " -
                    no se encuentra sub-sistema\n";
                }
                MessageBox.Show("Validación de FPH con
                errores.\nSe encontraron las siguientes
                observaciones:\n" + xstrErrores, "HCR",
                MessageBoxButtons.OK,
                MessageBoxIcon.Information);
            }
            NucleoHCR.CargarTablaExp();
            NucleoHCR.CargarTablaReg();
        }
        this.BarraProgreso.Visible=false;
    }
```

Una vez validada la Hoja de pase se carga el panel Izquierdo con los elementos encontrados en la hoja de pase como se puede ver en la siguiente figura:

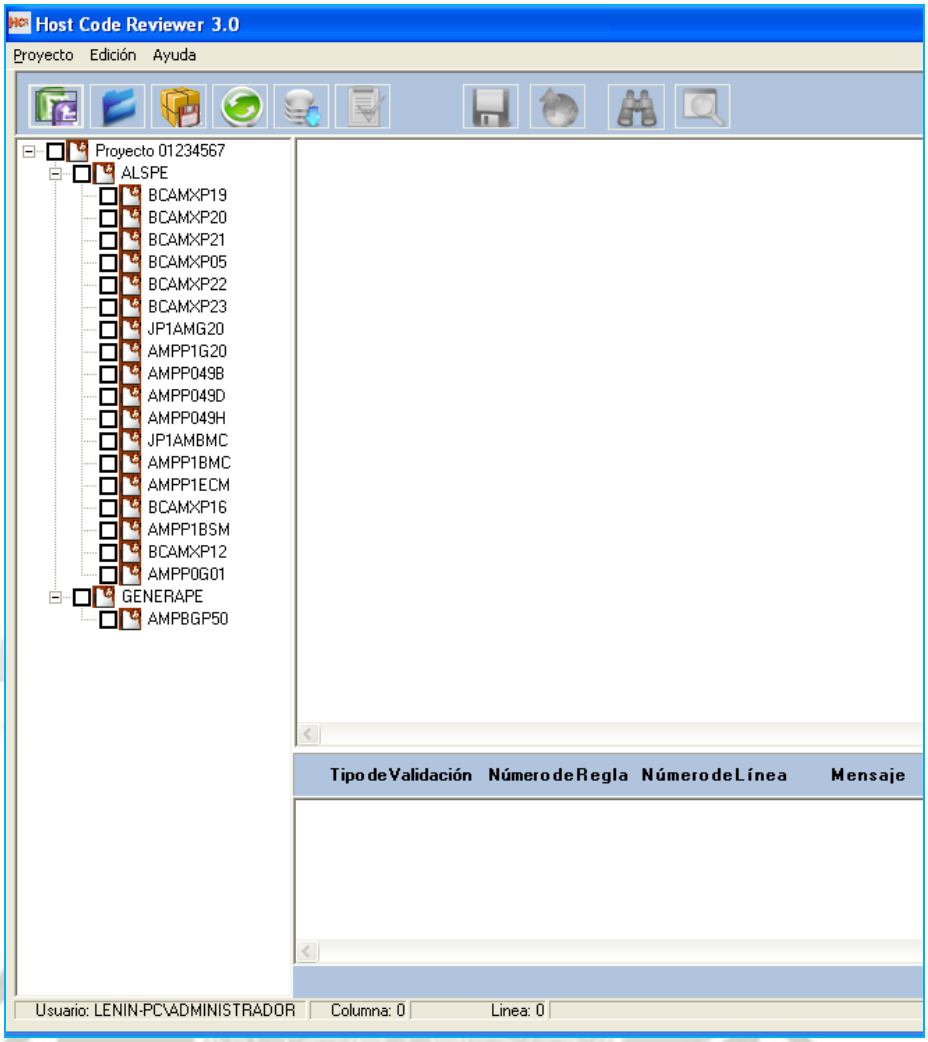


Figura Nº 10

Codificación:

```
private void CargarArbol(clsHojaPase HojaDePase)
{
   if(arbol.Nodes.Count>0)
         arbol.Nodes.Clear();
   NodoRaiz = new TreeNode();
   NodoPadre = new TreeNode();
   NodoRaiz.Text="Proyecto " + HojaDePase.GetTicket();
   arbol.Nodes.Add(NodoRaiz);
   ElementosPrograma=HojaDePase.GetUnidades();
   UnidadPrograma unidad = null;

   for(int i=0;i<ElementosPrograma.Count;i++)
```

```
if(((UnidadPrograma)ElementosPrograma[i]).EstadoValid
acion.Equals(0))
unidad=(UnidadPrograma)ElementosPrograma[i];
if(unidad!=null)
{
      NodoPadre.Text = unidad.ObtenerSistema();
      NodoRaiz.Nodes.Add(NodoPadre);
      NodoRaiz.Expand();

      string sistema="";

      //LLENA EL ARBOL CON LOS ELEMENTOS EN LA UNIDAD
      for(int NroUnidad=0; NroUnidad <
      ElementosPrograma.Count; NroUnidad++)
      {
            unidad=(UnidadPrograma)
            ElementosPrograma[NroUnidad];
            if(unidad.EstadoValidacion.Equals(0))
            {
                  NodoHijo=new TreeNode();
                  NodoHijo.Text =
                  unidad.ObtenerElemento();
                  NodoHijo.Tag=unidad;
                  sistema=unidad.ObtenerSistema();
                  NodoPadre=null;

                  foreach(TreeNode NodoSistema in
                  NodoRaiz.Nodes)
                  {
                        if(NodoSistema.Text==sistema)
                        {
                              NodoPadre=NodoSistema;
                        }
                  }

                  if(NodoPadre==null)
                  {
                        NodoPadre=new TreeNode();
                        NodoPadre.Text=sistema;
                        NodoRaiz.Nodes.Add
                        (NodoPadre);
```

```
                        }
                        NodoPadre.Nodes.Add(NodoHijo);
                    }
                }
            }
            arbol.ExpandAll();
        }
```

- **Verificación de elementos[KLAU12]**

  Para realizar la verificación de estándares de los elementos primero debe de marcar los elementos que desea verificar en el árbol de la izquierda y luego hacer clic en el icono  para iniciar la transferencia del código de los elementos seleccionados ya sea desde la PC o desde Host.

Cuando haya finalizado la transferencia le mostrara un cuadro indicándole si se realizo la transferencia con éxito, como se muestra en la **Figura Nº 11**:

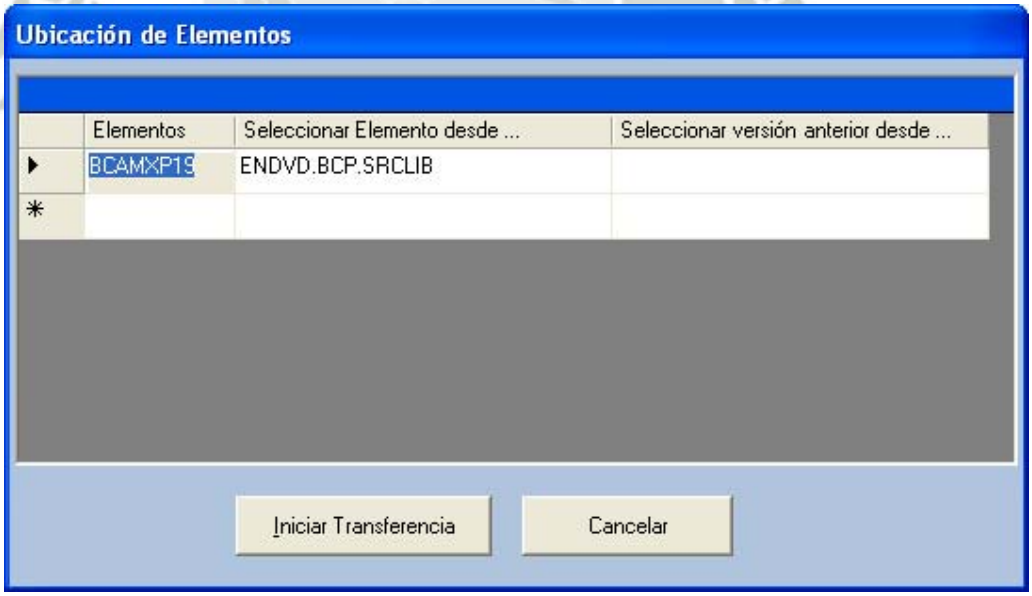


Figura Nº 11

Hacer clic en el Botón Aceptar. Hacer doble clic sobre el elemento seleccionado en el árbol de la izquierda y se mostrara el código del elemento como se muestra en la **Figura 12**:

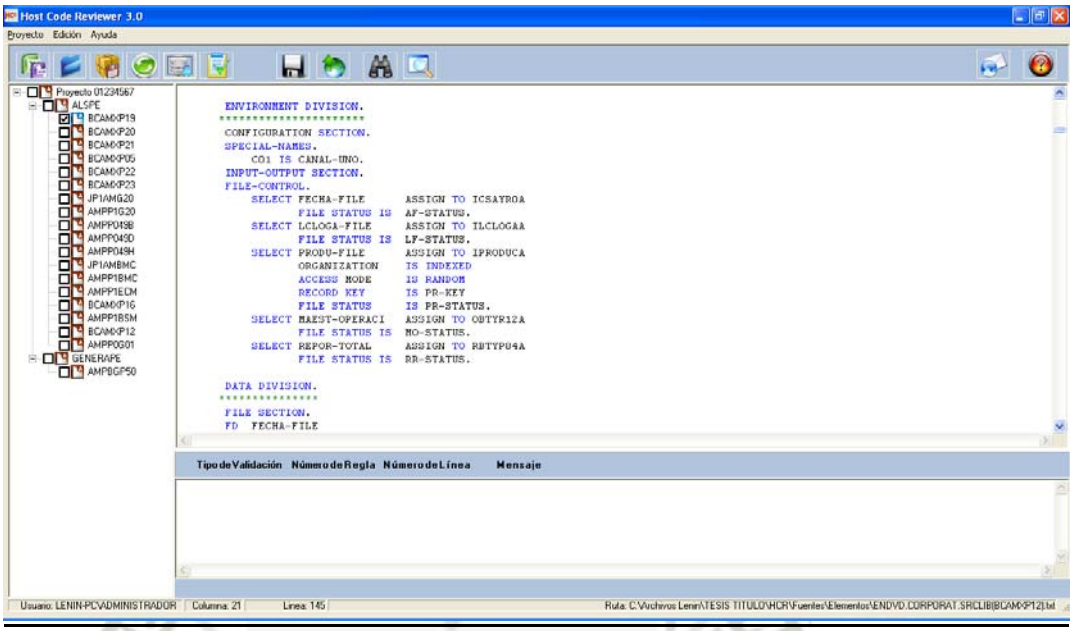


Figura Nº 12

Para iniciar la verificación de los estándares en el programa cobol debe de hacer clic en el icono  , si el código tiene el mínimo de errores le mostrara la **Figura Nº 13**:

Figura Nº 13

En caso de que haya superado el máximo de errores se mostrara la **Figura Nº 14:**

Figura Nº 14

Al hacer clic sobre algunas de las figuras anteriores se mostraran la lista de errores encontrados en la parte inferior de la ventana principal del HCR como se muestra en la **Figura Nº 15**:

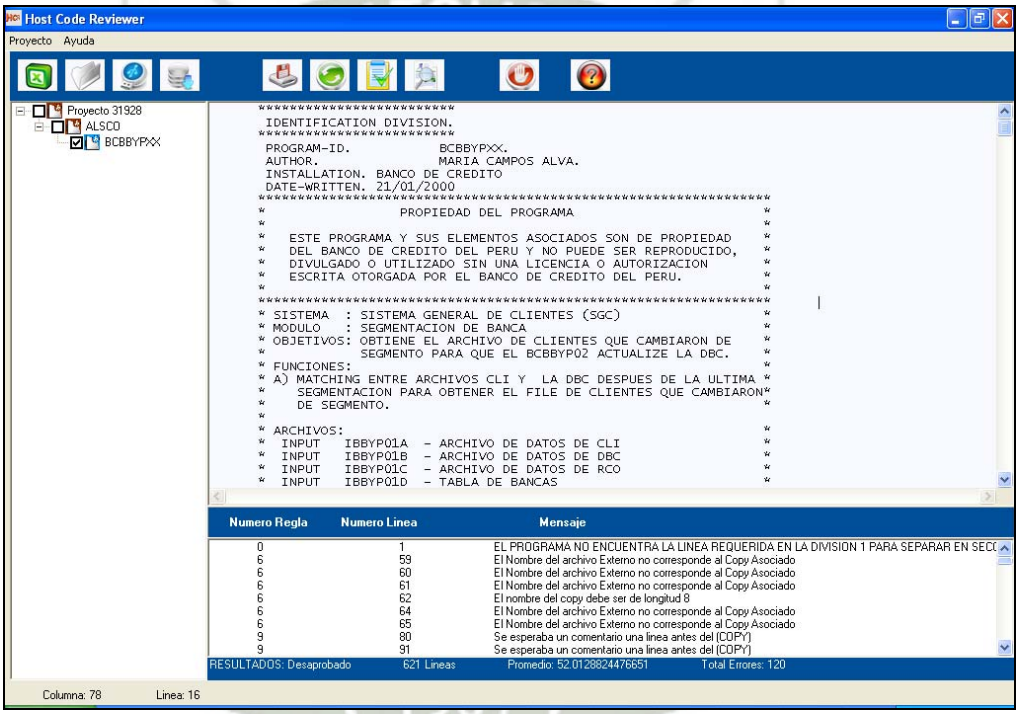


Figura Nº 15

Codificación:

La función empleada para la validación del código fuente es la siguiente:

```
private void Verificar(bool SilentMode)
{
   string strLineasError = ",";
   int idTipoLibreria =0;

   lblResultados.Text = "";
```

```
lstErrores.Items.Clear();
String Ruta="";

if(txtEditor.Text.Trim().Length > 0)
{
      Cursor.Current = Cursors.WaitCursor;
      lstErrores.Items.Clear();

      if(ElementoActual==null)
            MessageBox.Show("El elemento es invalido",
            "HCR", MessageBoxButtons.OK,
            MessageBoxIcon.Information);
      else
      {
            //OBTIENE LOS DATOS DEL ELEMENTO PARA REGISTRAR
            LA VERIFICACION
            NucleoHCR.EstablecerElemento (ElementoActual);

            //DETERMINA EL ESTADO DEL ELEMENTO
            SELECCIONADO(NUEVO O MODIFICADO)
            if (ElementoActual.ObtenerEstado().Equals("M"))
            {
                  RichTextBox txtCodProd = new
                  RichTextBox();

                  if (ElementoActual.
                  ObtenerFlagUbicacionDiscoProduccion ())
                        Ruta = ElementoActual.
                        ObtenerUbicacionDiscoProduccion();
                  else
                        Ruta = System.Windows.Forms.
                        Application.StartupPath. ToString()
                        + @"\Elementos\" +
                        ElementoActual.ObtenerUbicacionProd
                        uccion() + ".txt";

                  txtCodProd.LoadFile (Ruta,
                  System.Windows.Forms.
                  RichTextBoxStreamType.PlainText);
```

```
//VALIDACIÓN DE JOB Y PROCEDIMIENTOS
MODIFICADOS
idTipoLibreria = ElementoActual.
ObtenerIdTipo();

//SI ES PROCEDIMIENTO
if(idTipoLibreria == 3)
      Errores =
      NucleoHCR.VerificarEstandarProc
      (txtEditor.Text, txtCodProd.Text);
else
{
      //SI ES JOB
      if(idTipoLibreria == 4)
            Errores = NucleoHCR.
            VerificarEstandarJob
            (txtEditor.Text,
            txtCodProd.Text);
      else
      {
            //VALIDACION DE PROGRAMA HOST
            MODIFICADO
            Errores =
            NucleoHCR.VerificarEstandar
            (txtEditor.Text,
            txtCodProd.Text,
            this.practicas);
      }
}
}
else
{
      //VALIDACION DE JOB Y PROCEDIMIENTOS
      NUEVOS
      idTipoLibreria = ElementoActual.
      ObtenerIdTipo();

      //SI ES PROCEDIMIENTO
      if(idTipoLibreria == 3)
```

```
                    Errores =
                    NucleoHCR.VerificarEstandarProc
                    (txtEditor.Text);
                else
                {
                    //SI ES JOB
                    if(idTipoLibreria == 4)
                        Errores =
                        NucleoHCR.VerificarEstandarJo
                        b (txtEditor.Text);
                    else
                    {
                        //VALIDACION DE PROGRAMA HOST
                        NUEVO
                        Errores=NucleoHCR.VerificarEs
                        tandar(txtEditor.Text,this.pr
                        acticas);
                    }
                }
                Errores.Sort();
            }
        }

        try
        {
            Promedio PromEval = NucleoHCR.ObtenerPromedio
            (Errores);
            Promedio PromLinea =
            NucleoHCR.ObtenerPromedioLineamientos
            (Errores);

            string Resultado = String.Empty;
            Resultado += " | Total Lineas: " +
            PromEval.NumLin;
            Resultado += " | Promedio Estándares: " +
            Convert.ToSingle
            (PromEval.PromEval).ToString("#.00");
            Resultado += " | Estado: " + PromEval.Estado;

            if (this.practicas)
            {
```

```
                Resultado += " | Promedio Lineamientos: "
                + Convert.ToSingle(PromLinea.PromEval).
                ToString("#.00");
                Resultado += " | Estado: " +
                PromLinea.Estado;
            }

            Resultado += " | Total Errores: " +
            Errores.Count.ToString();
            CambiarIconos(PromEval.PromEval,
            PromLinea.PromEval,
            ElementoActual.EstadoRevision, true,
            SilentMode);
            lblResultados.Text = Resultado;
            ElementoActual.NotaRevision=PromEval.PromEval;
            ElementoActual.NotaRevisionLineamientos =
            PromLinea.PromEval;
            ElementoActual.NroLineas=PromEval.NumLin;
            ElementoActual.NumErrores = int.Parse
            (Errores.Count. ToString());

            if(PromEval.PromEval == 0)
                ElementoActual.EstadoRevision=0;
            else
            {
                if(PromEval.PromEval == 0)
                    ElementoActual.EstadoRevision=0;
                else if(PromEval.PromEval <= 20)
                    ElementoActual.EstadoRevision=1;
                else
                    ElementoActual.EstadoRevision=2;
            }
        }

        catch(Exception ExepcionPromedio)
        {
            MessageBox.Show(ExepcionPromedio.Message);
        }

        txtEditor.HideSelection = true;
```

```
            for(int NroError=0;NroError<Errores.Count;NroError++)
            {
                  clsErrorRegla Error =
                  (clsErrorRegla)Errores[NroError];

                  if (Error.NumeroRegla < 100)
                  {
                        lstErrores.Items.Add("\t" + "Estándar:
                        " + "\t\t"+ Error.NumeroRegla +  "\t" +
                        "\t" + Error.NumeroLinea.ToString() +
                        "\t" + "\t" +Error.MensajeError );
                  }

                  if (Error.NumeroRegla > 100)
                  {
                        lstErrores.Items.Add("\t" + "Lineamiento:
                        "+  "\t\t" + Error.NumeroRegla +  "\t" +
                        "\t" + Error.NumeroLinea.ToString() +
                        "\t" + "\t" +Error.MensajeError );
                  }
                  strLineasError += Error.NumeroLinea.ToString()
                  + ",";
            }

            txtEditor.HideSelection = false;

            //LÍNEAS CON ERRORES PARA QUE SEAN RESALTADAS DURANTE
            EL PINTADO
            txtEditor.LineasError = strLineasError;
            txtEditor.PuedePintarControl = true;
            txtEditor.Lines = txtEditor.Lines;
            txtEditor.PuedePintarControl = false;
            Cursor.Current = Cursors.Default;
      }
}
```

- **Interpretación de los resultados de la verificación.**

Los iconos de cada elemento significa el estado en que se encuentra cada elemento del proyecto. Así tenemos que:

- El icono representa que el elemento está sin transferir.
- El icono representa que el elemento fue transferido.
- El icono representa que el elemento no tiene errores de estándares.
- El icono representa que el elemento tiene un nivel aceptable de errores de acuerdo al promedio permitido.
- El icono representa que el elemento tiene algún error que no esta permitido o que tiene más errores que el nivel aceptable.

La columna de promedio es un cálculo que se realiza tomando en consideración la cantidad de errores y los tipos de errores cometidos sobre el total de líneas del programa. Para un programa modificado es sobre el total de líneas modificadas. Los errores que tienen el máximo peso en los estándares de codificación son los que están relacionados a la IDENTIFICATION DIVISION. Por lo que un error en esta sección, no será aceptado con un nivel aceptable por HCR.

- **Visualización de los errores de un programa**

Cuando verifica un programa, la lista de errores se mostrará en el panel inferior, para poder ubicar el error dentro del código y poder corregirlo debe hacer doble clic sobre el error en la lista de errores como se muestra en la **Figura Nº 16:**

Al haber hecho doble clic sobre el error se sombreará en el código el error al que se hace referencia en la lista de errores

Lista de errores encontrados

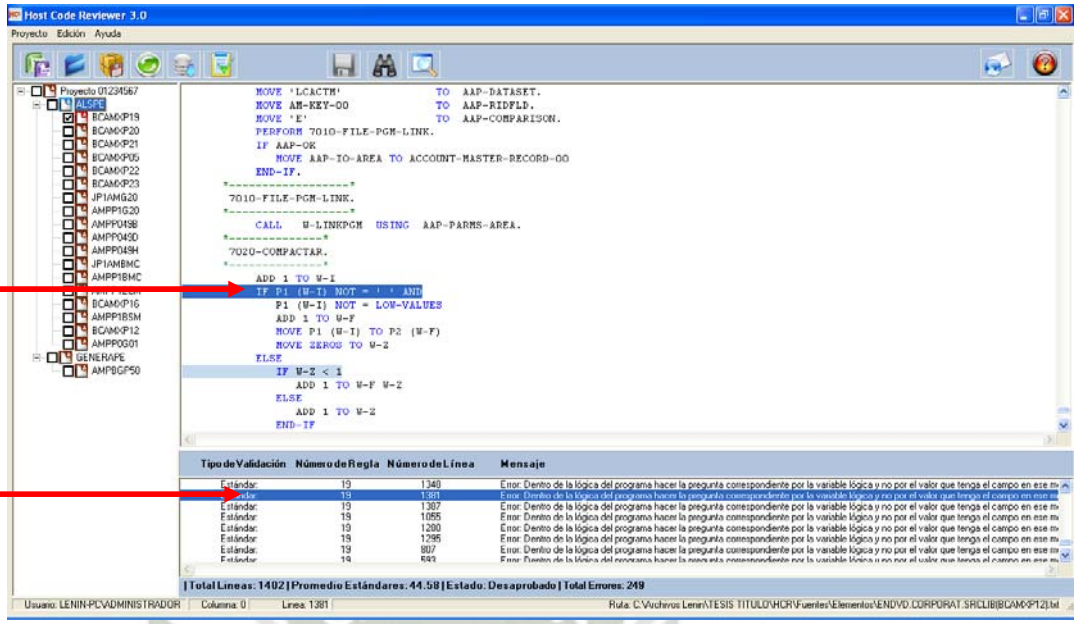


Figura Nº 16

- **Corregir y guardar los errores desde el editor de HCR**

El usuario podrá modificar los errores en estándares que encuentre, directamente desde la herramienta HCR, realizando esta tarea se corre el riesgo de incluir errores de programación, por lo que el elemento deberá ser nuevamente compilado en COBOL.

CONCLUSIONES

1. Se debe invertir más tiempo en las etapas iníciales del proyecto como: el análisis de requerimientos y el análisis del sistema, para evitar errores en etapas posteriores, que aumentarían el costo del proyecto.
2. El uso de estándares permite una comunicación más clara entre los integrantes de un proyecto y los nuevos integrantes que se suman al desarrollo en etapas intermedias del ciclo de vida del software.
3. Como parte de la experiencia que desarrolle en mi trabajo, tuve la oportunidad de aprender lo importante que es el uso de estándares durante el desarrollo, ya que es una buena práctica para el mantenimiento del aplicativo.
4. El trabajo desempeñado en la empresa me ha permitido aprender nuevos lenguajes de programación como COBOL, que a pesar de ser un lenguaje antiguo aun se sigue utilizando ya que es rápido y permite las administración de millones de transacciones, así como las metodologías como el CMMI e ITIL.
5. Por último, la experiencia adquirida en los años de permanencia en el BCP y actualmente en TCS, me han permitido ampliar mis conocimientos en las diferentes ramas de mi carrera y apuntar hacia temas relacionados a proyectos en un corto plazo.

RECOMENDACIONES

- Implementar la herramienta HCR en Web, utilizando ASP.NET. para que se puedan validar elementos desde cualquier PC que tenga una conexión a internet.
- Implementar una herramienta de validación para Consultas SQL, utilizando las premisas utilizadas en este proyecto como las Expresiones Regulares, Estándares y mejores prácticas.

BIBLIOGRAFÍA

[CEBA92] Francisco Javier Ceballos Sierra (1992), CURSO DE PROGRAMACION RM COBOL 85.

[DATE01] Date C.J. (2001). Introducción a los Sistemas de Base de Datos, México: Editorial Addison Wesley Iberoamericana S.A.

[HANS98] Gary W. Hansen, James V. Hansen: Diseño y Administración de Base de Datos, 1998, Prentice Hall.

[KEND11] Simon Kendal (2011). Object Oriented Programming using C#. Primera Edición.

[KLAU12] Poul Klausen (2012). Introduction to programming and the C# language. Primera Edición.

[JOYA96] Joyanes, Aguilar Luis (1996). Fundamentos de programación, Algoritmos y Estructuras de Datos. Segunda Edición. España: McGraw-Hill Interamericana de España, S.A.

[LAND10] Nicolás Arrioja Landa Cosio (2010). C# Guía Total del Programador. Primera Edición. Buenos Aires.

[PERE11] Tania Pérez Vázquez (2011). JCL Básico.

[PRES98] Pressman, Roger (1993). Ingeniería del Software, Un enfoque practico. España: McGraw-Hill Interamericana de España, S.A.

URLS:

[URL01] Banco de Crédito:

https://www.viabcp.com

[URL02] Procesos PAR

url intranet

[URL03] Visual C#

http://msdn.microsoft.com/es-es/library/kx37x362(v=vs.90).aspx

[URL04] Extensible Markup Language (XML)

http://www.w3.org/XML/

[URL05] Introducción a las expresiones regulares

http://msdn.microsoft.com/es-es/library/28hw3sce(v=vs.90).aspx

APÉNDICE(S)

PLAN DE TRABAJO INFORME

1. Titulo del Trabajo Informe

"EXPERIENCIA PROFESIONAL EN LA DIVISIÓN SISTEMAS DEL BANCO DE CRÉDITO DEL PERÚ DURANTE LOS AÑOS 2008 AL 2012"

2. Objetivos

2.1 General

- Exponer el trabajo desarrollado en el Área de Sistemas del Banco de Crédito del Perú.

2.2 Específicos

- Exponer los trabajos desarrollados para plataforma Mainframe y Cliente/Servidor.
- Exponer las metodologías y estándares utilizados para el desarrollo de Software dentro del Área de Sistemas de la empresa.

3. Marco Teórico

El alcance del trabajo consiste en exponer los trabajos más importantes desarrollados dentro del Área de Sistemas del Banco de Crédito del Perú, mostrando los principales aspectos de su implementación, resaltando los puntos más importantes como Metodología empleada para el análisis, Estándares utilizados para la implementación y los Casos de prueba utilizados durante el desarrollo para la verificación de los aplicativos terminados.

4. Memoria Profesional

4.1 Empresa

4.1.1 Descripción de la empresa

La empresa, llamada durante sus primeros 52 años Banco Italiano, inició sus actividades el 9 de abril de 1889, adoptando una política crediticia inspirada en los principios que habrían de guiar su comportamiento institucional en el futuro. El 01 de febrero de 1942, se acordó sustituir la antigua denominación social, por la de Banco de Crédito del Perú.

Así, el Banco Italiano, el primero en el país, cerró su eficiente labor después de haber obtenido los más altos resultados de la institución. Con el propósito de conseguir un mayor peso internacional, se instalaron sucursales en Nassau y en Nueva York, hecho que convirtió a la empresa en el único Banco peruano presente en dos de las plazas financieras más importantes del mundo. La expansión de sus actividades creó la necesidad de una nueva sede para la dirección central. Con ese fin se construyó un edificio de 30,000 m2, aproximadamente, en el distrito de La Molina. Luego, con el objetivo de mejorar sus servicios, establecieron la Red Nacional de Tele Proceso, que a fines de 1988 conectaba casi todas las oficinas del país con el computador central de Lima; asimismo, crearon la Cuenta Corriente y Libreta de Ahorro Nacional, e instalaron una extensa red de cajeros automáticos.

En 1993, se adquirió el Banco Popular de Bolivia, hoy Banco de Crédito de Bolivia. Un año más tarde, con el fin de brindar una atención aún más especializada, crearon Credifondo, una nueva empresa subsidiaria dedicada a la promoción de los fondos mutuos; al año siguiente se estableció Credileasing, empresa dedicada a la promoción del arrendamiento financiero. Durante los ’90, la oficina de representación en Santiago de Chile desarrolló una interesante actividad, dado el notable incremento de los capitales chilenos invertidos en empresas peruanas. La recuperación de los jóvenes talentos que emigraron entre 1970 y 1990 al extranjero, fue otro aspecto importante de esa década. Esos profesionales, sólidamente formados en centros académicos y empresas importantes de los Estados Unidos y Europa, han contribuido a confirmar la imagen que siempre se tuvo: un Banco antiguo con espíritu siempre moderno. Al cumplir nuestros 123 años de existencia, la Institución cuenta con 346 Agencias, 1,526 cajeros automáticos, 5,400 Agentes BCP y 15,564 empleados; y bancos corresponsales en todo el mundo.

4.1.2 Tipo de empresa

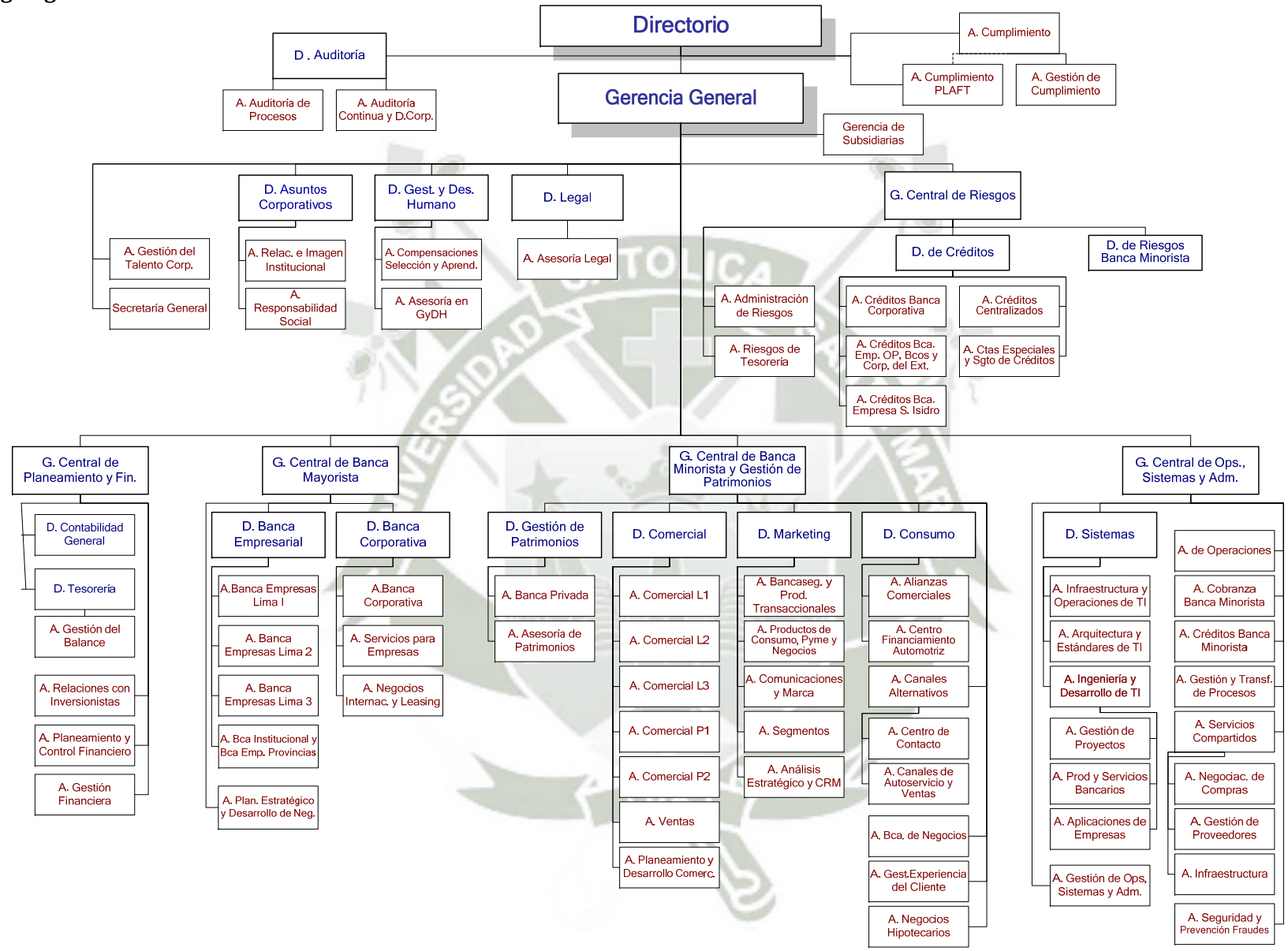
Financiera.

4.1.3 Ubicación

La oficina principal de la empresa se encuentra ubicada en la ciudad de Lima en el distrito de la Molina, Calle Centenario N° 166 Urb. Las Laderas de Melgarejo.

El centro de desarrollo de Sistemas principal se encuentra ubicado en Chorrillos, como sucursales de desarrollo de sistemas se tienen dos oficinas ubicadas en la ciudad de Arequipa, una en la Calle Perú 207 – Cercado y la otra en la Calle Urubamba 301 – Cayma.

4.1.4 Organigrama



4.1.5 Visión

Ser el Banco líder en todos los segmentos y productos que ofrecemos.

4.1.6 Misión

Promover el éxito de nuestros clientes con soluciones financieras adecuadas para sus necesidades, facilitar el desarrollo de nuestros colaboradores, generar valor para nuestros accionistas y apoyar el desarrollo sostenido del país.

4.2 Actividad Profesional

4.2.1 Cargo desempeñado

Programador.

4.2.2 Funciones del programador

- Producir software para el BCP y las diversas empresas o instituciones que así lo requieran del grupo Crédito.
- Elaborar sistemas que optimicen las aplicaciones del negocio y las tareas administrativas de todas las áreas de la empresa.
- Mantener y actualizar los sistemas automatizados.
- Programar sistemas en los diferentes lenguajes de programación con los que cuente la empresa.
- Elaborar diagramas de lógica o bloques de aplicación.
- Realizar el análisis y optimización de algoritmos para su aplicación en tiempo real.
- Participar en la solución de problemas de soporte de software.
- Formular manuales de operación y programación.
- Realizar pruebas de calidad del nuevo software que se vaya a implantar en los servidores o equipos de la empresa.
- Elaborar informes del desarrollo de su labor.

- Realizar otras funciones que le sean asignadas.

- Colaborar y realizar trabajos relacionados con su especialidad.

4.2.3 Descripción de los trabajos desarrollados

4.2.3.1 Refactoring de la Herramienta HCR(Host Code Reviewer)

Requerimiento: La Herramienta HCR (Host Code Reviewer), es una aplicación encargada de validar el cumplimiento de los estándares de programación para programas COBOL.

Utiliza 3 interfaces principales, desarrolladas en 2 lenguajes: Una dll de nombre ErrorFinder.dll, la cual es la encargada de realizar el proceso de revisión de estándares propiamente dicho, dicha dll se encuentra desarrollada en Visual C++ 6.0, Un "Agente" de conexión con Mainframe, para la recepción de los programas a ser validados, dicho agente utiliza una conexión XCOM, dicho agente se encuentra desarrollado en Visual Basic 6.0. Y finalmente la interfaz grafica de la aplicación la cual se encuentra desarrollada en Visual Basic 6.0. Se cuenta con una base de datos SQL 2000 que almacena información de reglas, validaciones, usuarios etc.

Se desea disminuir la complejidad del aplicativo y el nivel soporte que se le da a la herramienta.

Alternativa de solución implementada: Implementar la Herramienta desde cero, utilizando C# .NET 2003 para el desarrollo, como gestor de Base de Datos utilizar el Microsoft SQL Server 2008, para la implementación de las reglas de validación utilizar expresiones regulares para que su mantenimiento sea por Base de Datos.

Herramientas utilizadas: C# .NET 2003 y Microsoft SQL Server 2008.

4.2.3.2 Desacoplamiento TOLD II de Letras

Requerimiento: La aplicación Letras TP debe exponer los servicios comunes que contemplen la funcionalidad ofrecida por la interfaz ad hoc LERE(Recepción de Letras).

El servicio común debe implementarse desacoplado de TOLD II (Ruteador de transacciones).

Contempla: La validación de datos de letras, la generación del código de planilla, código de Remesa, la grabación de archivos de Letras, grabación de log y manejo de extorno.

Alternativa de solución implementada: Clonar programas que forman parte del proceso actual (actualización de tablas CONSIST – Modulo de Clientes) a fin de que éste se convierta en Servicio Común.

Se reutilizo la tabla de Productos de Letras a fin de registrar ciertos parámetros que permitan evitar el "Hard Code" en el nuevo programa CBZPILER y clon del CBZPIFUE. Este programa CBZPILER fue implementado utilizando los estándares definidos y evitando las sentencias GO TO que contenía el programa original CBZPIFUE y también tenía que considerar los estándares definidos para el uso de Servicios Comunes. Se implemento un copy CBZXMFUE que debía contener la estructura definida para los datos de input del programa CBZPILER.

Herramientas utilizadas: COBOL CICS y sentencias FIO(File Input Output) para el uso de DATACOM(Sistema Administrador de Base de Datos Relacional).

4.2.3.3 Controlar Relación con Grupos Económicos – CONSIST Clientes

Requerimiento: Restringir la asociación de un cliente a más de un grupo económico mediante la pantalla REC (Relaciones entre Clientes.).

Generar un reporte que permita identificar a los clientes que están asociados a más de un grupo económico, el reporte debe incluir todos los campos de la pantalla REC.

Alternativa de solución implementada: Restringir en la pantalla Online de CONSIST (Modulo de clientes), pantalla REC para que no permita relacionar a clientes con más de un grupo económico.

Herramientas utilizadas: COBOL CICS, COLDVIEW (Herramienta Administradora de Documentos) y DATACOM

4.2.3.4 Implementar Tarjeta Mastercard en SAPP(Sistema Administrador de Productos Pasivos)

El Sistema Administrador de Productos Pasivos (SAPP) es un sistema cliente/servidor propietario de BCP que permite la administración de la información de clientes, la apertura de productos pasivos y operaciones con tarjetas de débito, así como las consultas a diferentes sistemas del BCP.

Requerimiento: Permitir la atención del cliente de una Tarjeta Mastercard a través de SAP Pasivos.

Requisitos funcionales:

- Consulta de datos en CONSIST: Debe mostrar la información de los productos TC Mastercard que tenga afiliados el cliente persona natural.
- Consulta de datos en VPLUS (Aplicativo de Tarjetas de Credito): Debe mostrar la información de los productos TC Mastercard que tenga afiliados el cliente persona natural.
- Afiliación a una Tarjeta de Débito: El aplicativo SAPP permite la afiliación de un producto pasivo (Ahorros, Corriente, Maestra, CTS) o activo (Tarjeta de crédito) de un cliente a una tarjeta de débito.
- Desafiliación de una Tarjeta de Débito: Permitir la desafiliación de productos de la tarjeta de débito seleccionada, actualizando la información en SAT (Sistema Administrador de Tarjetas).
- Modificación de datos de una Tarjeta de Débito: Mostrar el producto en pantalla en caso se encuentre afiliado a una tarjeta de débito y asimismo mostrar el producto afiliado en el contrato de modificación.
- Cambio de una Tarjeta de Débito: Se muestra el producto en pantalla en caso esté afiliado a la tarjeta de débito anterior, y asimismo se muestre en el contrato de cambio de la nueva tarjeta de débito. Se debe tener en

cuenta que la opción de afiliación de productos TC Mastercard sólo estará disponible para clientes persona natural.

- Bloqueo de una Tarjeta de Débito: Se muestra el producto en pantalla en caso se encuentre afiliado a una tarjeta de débito y asimismo mostrar el producto afiliado en el contrato de bloqueo de la tarjeta de débito.

- Consulta de Productos por Tarjeta: Se debe mostrar la información de los saldos y movimientos de los productos pasivos que están afiliados a una tarjeta de débito Credimás.

- Consulta de Tarjetas por Producto: Se debe mostrar la información de las tarjetas de débito Credimás que tienen afiliado el producto TC Mastercard.

- Cambio de Clave Personal de una Tarjeta: La opción de cambio de la clave personal de una tarjeta de crédito Mastercard debe permitir el cambio del PIN de 4 dígitos, tan igual como se realiza con una tarjeta de crédito VISA

- Impresión de Contratos: Se requiere modificar el Contrato A para que se muestre el número de tarjeta TC Mastercard afiliado a la tarjeta de débito en la impresión del contrato, en original y/o copia.

- Registro de Control de Bloqueo: Se requiere modificar las pantallas para incluir las operaciones de bloqueo de las nuevas tarjetas TC Mastercard.

- Impresión de Reportes de Bloqueos: Se requiere modificar las pantallas y reportes para incluir las operaciones de bloqueo de las nuevas tarjetas TC Mastercard.

Alternativa de solución implementada: Realizar las modificaciones en SAPP para reconocer a la Tarjeta Mastercard como un producto distinto a las tarjetas de otras marcas (VISA, AMEX). La implementación de esta alternativa es de complejidad media debido a la necesidad de modificar los formularios donde se muestre información de tarjeta de crédito, los contratos que se imprimen para entregar al cliente y los reportes administrativos.

Herramientas utilizadas: Visual Basic 6.0 y SQL Server 2000.

4.2.3.5 Respaldo de Data Histórica RRHH BCP y Subsidiarias

Requerimiento: Generar reportes históricos para las siguientes compañías:

- 01 – BCP.
- 02 – Credibolsa.
- 03 – Credifondo.
- 04 – Solución Financiera de Crédito.
- 06 - Soluciones en Procesamiento S.A.
- 14 - Banco Santander Central Hispano.
- 15 - Santander Central Hispano S.A.F.
- 19 – Prima AFP.
- 12 – Pacifico Peruano Suiza Empleados.
- 20 – Pacifico Peruano Suiza Funcionarios.
- 21 – Pacifico S.A. Entidad Prestadora de Salud.
- 22 – Asociación Civil Asistencia Social Cristal.

Los reportes que se generaron por cada compañía son los siguientes:

- Reporte de boletas de pago de haberes.
- Reporte de Datos de Trabajador.
- Reporte de Datos Laborales de Trabajador.
- Reporte de Datos Financieros de Trabajador.
- Reporte de Datos Familiares de Trabajador.
- Reporte de Estudios de Trabajador.
- Reporte de Datos de Ausencia de Trabajador.

- Reporte de Datos de Tardanza de Trabajador.
- Reporte de Datos de Licencia de Trabajador.
- Reporte de Balance de Prestamos.
- Reporte de Detalle de Prestamos Vigente.
- Reporte de Rol y record Vacacional del Trabajador.
- Reporte Experiencia Laboral de Trabajador.
- Reporte Misiones de Trabajador.
- Reporte de Movilidad Comisiones del Trabajador.
- Reporte de Estructura de Organización.
- Reporte de Puestos de la Organización.
- Reporte de Grado Salarial y Aumentos de Trabajadores.
- Reporte de Movimiento de Personal.
- Reporte de Prestamos Administrativos Desembolsados.

Alternativa de solución implementada: Esta alternativa consiste en migrar los reportes solicitados por el usuario de los antiguos aplicativos de RRHH BCP y subsidiarias a COLDVIEW (Herramienta Administradora de Documentos) bajo el formato que actualmente se visualiza en los sistemas antiguos de RRHH. Para ello se hará uso de la herramienta Microsoft SQL Server 2000, para crear la rutina en el servidor de base de datos de RRHH, cuyo resultado será exportado a la herramienta COLDVIEW.

Se generaron DTSs para generar los reportes en archivos TXT en un servidor Windows Server 2003, en una ruta compartida para ser cargados en la herramienta COLDVIEW.

Herramientas utilizadas: Microsoft SQL Server 2000 y COLDVIEW.

4.2.3.6 Migracion de DTS a SSIS – Suite Cambios

Requerimiento: Migrar 12 DTSs a SISS del aplicativo Suite Cambios que se encuentran SQL Server 2008 en modo compatibilidad considerando que se debe modificar sentencias SQL y procedimientos almacenados para que no tengan sentencias no permitidas y la comunicación con la BD sea correcta. También es necesario crear archivos .BAT para ejecutar los siguientes Jobs SQL desde la rutina Host:

- MCAM-IPM6AMM5-LeeTotalesCOMEVEMEHOST
- MCAM-IPM6AMM4-LeeDetallesCOMEVEMEHOST
- MCAM-IPM0AMCD-EnviaDataGLHOST desde Host.

Los DTS a migrar son los siguientes:

- MCAM-IPM0AMCD-EnviaDataGL.dts
- MCAM-IPM6AMM1-EnviaDataForwards.dts
- MCAM-IPM6AMM11-CargarCarteraTrader.dts
- MCAM-IPM6AMM12-EnviaMailRiesgos.dts
- MCAM-IPM6AMM2-GeneraFormato108Sucave.dts
- MCAM-IPM6AMM3-GeneraFormato108HistoricoSucave.dts
- MCAM-IPM6AMM4-LeeDetallesCOMEVEME.dts
- MCAM-IPM6AMM5-LeeTotalesCOMEVEME.dts
- MCAM-IPM6AMM6-GeneraFormato205Sucave.dts
- MCAM-IPM6AMM7-GeneraCircular10BCR.dts
- MCAM-IPM6AMM8-GeneraReporteMovimientos.dts
- MCAM-IPM6AMM9-GeneraReporteInventario.dts

Alternativa de solución implementada: Migración con la opción Migrate DTS 2000 Package del Visual Studio 2008, porque es una estrategia utilizada por el equipo de Mejoras Tecnológicas dando resultados satisfactorios, está

permite disminuir el riesgo de migrar las tareas manualmente donde existe el riesgos de omitir alguna funcionalidad pero con la herramienta de migración se evita ese riesgo.

Herramientas utilizadas: Visual Studio 2008, Microsoft SQL Server 2008 y Cobol Batch

4.2.4 Resumen de trabajo más importante desarrollado en el BCP

Actualmente para el Banco de Crédito del Perú es importante contar con un proceso de verificación y validación de estándares en todos los desarrollos de software que se dan dentro de la empresa, porque aumenta la confianza y garantiza un producto de calidad, también le permite hacer mejoras en los aplicativos en caso sean necesarios.

La empresa maneja un alto número de transacciones e información, por tal motivo un gran numero de las aplicaciones importantes dentro de la empresa fue desarrollado con el lenguaje de programación COBOL BATCH, debido a la importancia de las aplicaciones desarrolladas con este lenguaje es que se elaboró estándares de programación para garantizar su calidad en el desarrollo y para permitir su mantenimiento, también se vio necesaria la elaboración de una herramienta que ayude a la revisión de los estándares; se desarrollo la herramienta Host Code Reviewer (HCR), la cual automatiza el proceso de verificación y validación realizando una inspección de código en programas desarrollados en COBOL BATCH, desplegando resultados sobre dicha validación, otorgando una puntuación que evaluará el nivel de cumplimiento de estándares antes de su implementación en ambientes de producción.

El objetivo de este trabajo es mejorar la herramienta HCR existente, permitiendo la validación de JOBs y Procedimientos basado en los estándares de programación definidos dentro de la empresa.

5. Metodología Empleada[1]

Para la atención de requerimientos dentro de la empresa se cuenta con un proceso llamado PAR (Proceso de Atención de Requerimientos), proceso en

el cual se indican las fases por las cuales pasa una solicitud del cliente hasta su implementación y entrega final.

Las fases con las que se cuentan son:

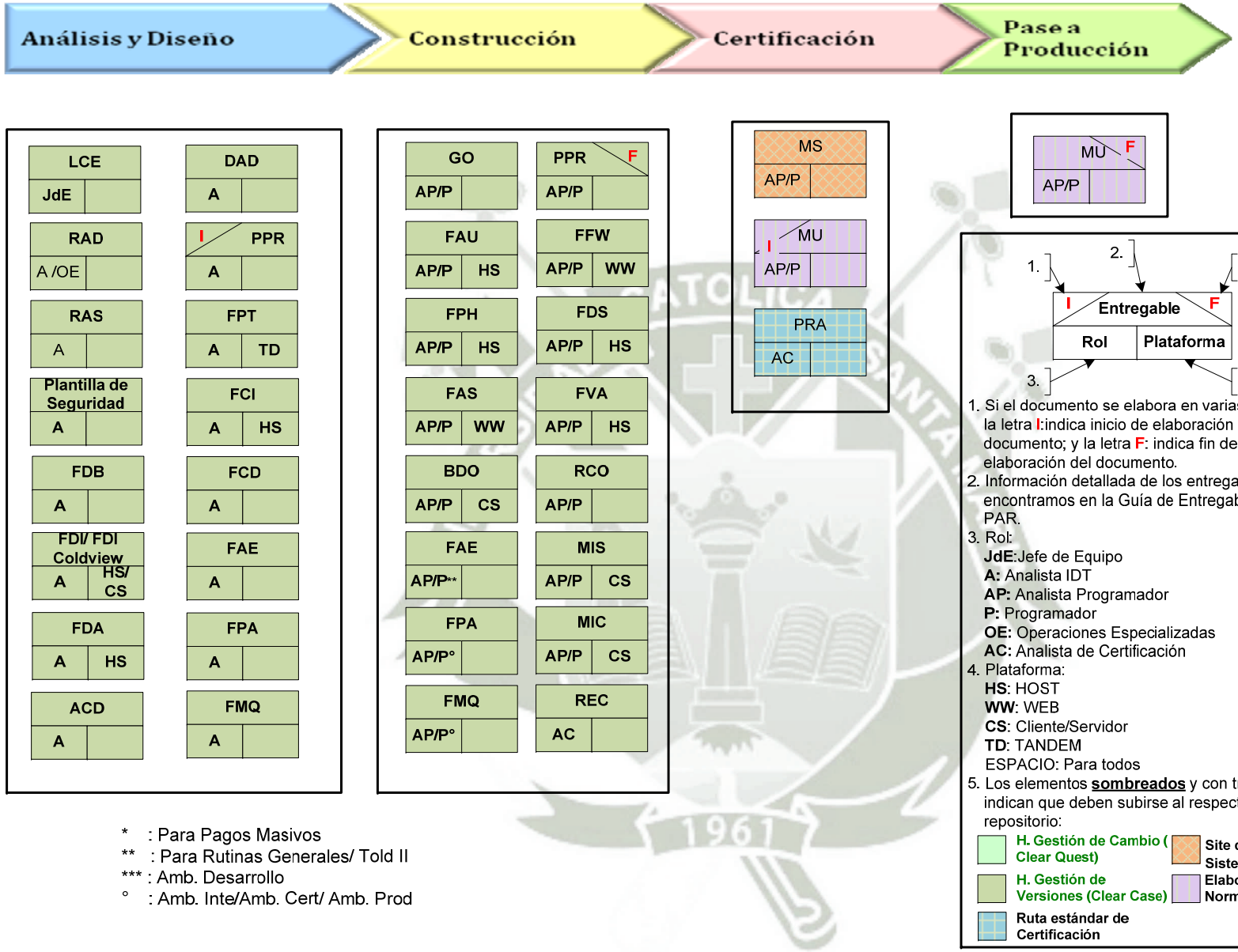
Análisis y diseño: En esta fase del PAR (Proceso de Atención de Requerimientos) se realiza el análisis funcional y técnico de acuerdo a lo solicitado por el usuario, para el desarrollo y construcción del requerimiento, generándose el entregable DAD (Documento de Análisis y Diseño), en ese formato se especifica cuáles serán los cambios en la funcionalidad del aplicativo impactado y los detalles técnico para su implementación, como el lenguaje de programación a utilizar, gestor de base de Datos, servidor o servidores impactados, permisos de usuario, etc.

Construcción: En esta fase se realiza la codificación de todo lo indicado en el DAD, se realizan las pruebas unitarias e integrales antes de pasar a la siguiente fase, también se generan los entregables necesarios para la implementación de los cambios en los ambientes de Certificación y Producción.

Certificación: En esta fase se ejecutan los casos de prueba para verificar que todo lo indicado en el DAD se desarrolló correctamente y que no se hayan impactado otras funcionalidades del aplicativo que debieron mantenerse. Esta fase es la que se encarga de verificar la calidad de lo desarrollado y también de que los entregables generados para la implementación en certificación y producción.

Pase a Producción: Es la última fase del proceso PAR, y es la fase en la cual todos los cambios realizados pasan al ambiente del usuario y empiezan a funcionar en el área de negocio al cual corresponde el aplicativo modificado.

En el siguiente grafico se puede apreciar las fases del PAR, los entregables que se generan en cada una de las fases y los responsables de generar dichos entregables.



* : Para Pagos Masivos
** : Para Rutinas Generales/ Told II
*** : Amb. Desarrollo
° : Amb. Inte/Amb. Cert/ Amb. Prod

6. Plan de Trabajo

Se elaboro el siguiente plan de trabajo para el Informe y el proyecto:

Nombre de tarea	% completado	Duración	Comienzo	Fin
PROYECTO IIIFORME	**0%**	**312 horas**	**mar 17/09/13**	**jue 21/11/13**
Elaboracion de Capitulo I	0%	40 horas	mar 17/09/13	mié 25/09/13
Elaboracion de Capitulo II	0%	40 horas	mié 25/09/13	jue 03/10/13
Elaboracion de Capitulo III	0%	40 horas	jue 03/10/13	vie 11/10/13
Implementacion de Proyecto	**0%**	**192 horas**	**vie 11/10/13**	**jue 21/11/13**
Agregar Validacion de Jobs y Proc a Herramienta HCR	**0%**	**192 horas**	**vie 11/10/13**	**jue 21/11/13**
Análisis	**0%**	**32 horas**	**vie 11/10/13**	**vie 18/10/13**
Validar Jobs	**0%**	**16 horas**	**vie 11/10/13**	**mar 15/10/13**
Validar Jobs Nuevos	0%	8 horas	vie 11/10/13	lun 14/10/13
Validar Jobs Modificados	0%	8 horas	lun 14/10/13	mar 15/10/13
Validar Procedimeintos	**0%**	**16 horas**	**mar 15/10/13**	**vie 18/10/13**
Validar Procedimientos Nuevos	0%	8 horas	mar 15/10/13	jue 17/10/13
Validar Procedimientos Modificados	0%	8 horas	jue 17/10/13	vie 18/10/13
Construcción	**0%**	**160 horas**	**vie 18/10/13**	**jue 21/11/13**
Codificar las Unidades de Programación	**0%**	**72 horas**	**vie 18/10/13**	**lun 04/11/13**
Validar Jobs y Procedimientos	**0%**	**72 horas**	**vie 18/10/13**	**lun 04/11/13**
Modificar carga de elementos	0%	24 horas	vie 18/10/13	jue 24/10/13
Modificar Validador de elementos	0%	24 horas	jue 24/10/13	mar 29/10/13
Agregar Expresiones rgulares en BD	0%	24 horas	mar 29/10/13	lun 04/11/13
Pruebas Unitarias: Preparar Ejecutar	0%	24 horas	lun 04/11/13	vie 08/11/13
Pruebas de Integración: Ejecutar casos	**0%**	**64 horas**	**vie 08/11/13**	**jue 21/11/13**
Validar Jobs	0%	32 horas	vie 08/11/13	vie 15/11/13
Validar Procedimientos	0%	32 horas	vie 15/11/13	jue 21/11/13

7. Posible Tabla de Contenidos

Introducción

Cap. 1. Aspectos Generales

Objetivos

Memoria Profesional

Marco Teórico

Técnicas y Herramientas

Cap. 2. Aspectos Relevantes del Desarrollo Profesional

Aspectos Relevantes

Bibliografía

Conclusiones

Recomendaciones

Apéndice(s)

8. Referencias Bibliográficas

[1] PAR-PPT-Guía_Gráfica_Entregables_PAR, Documentación BCP.