



Eusko Jauriaritzaren Informatika Elkarte
Sociedad Informática del Gobierno Vasco

Estándares

Estándares de desarrollo de sistemas software

Fecha: 30/06/2011

Referencia:

EJIE S.A.
Mediterráneo, 14
Tel. 945 01 73 00*
Fax. 945 01 73 01
01010 Vitoria-Gasteiz
Posta-kutxatila / Apartado: 809
01080 Vitoria-Gasteiz
www.ejie.es

Este documento es propiedad de EJIE, S.A. y su contenido es confidencial. Este documento no puede ser reproducido, en su totalidad o parcialmente, ni mostrado a otros, ni utilizado para otros propósitos que los que han originado su entrega, sin el previo permiso escrito de EJIE, S.A.. En el caso de ser entregado en virtud de un contrato, su utilización estará limitada a lo expresamente autorizado en dicho contrato. EJIE, S.A. no podrá ser considerada responsable de eventuales errores u omisiones en la edición del documento.

Control de documentación

Título de documento:

Histórico de versiones

Código:

Versión: 2.0

Fecha: 30/06/2011

Resumen de cambios:

- Se incluyen las actividades y tareas derivadas del modelo estándar de aseguramiento de la calidad.
- Se especifican los roles identificados y se completa el flujo de ejecución de procesos
- Se asignan las tareas más importantes del análisis al personal de EJIE
- Se incorporan las matizaciones y cambios propuestos por DIT
- Se añade la necesidad de validar las aplicaciones en otros navegadores
- Se actualiza para contemplar la metodología de pruebas Probamet y el modelo de aseguramiento de la calidad (modelo SQA)

Cambios producidos desde la última versión

Primera versión.

Control de difusión

Responsable:

Aprobado por:

Firma:

Fecha: 30/06/2011

Distribución:

Referencias de archivo

Autor: Ander Martínez

Nombre archivo:

Localización:

Contenido

	Capítulo/sección	Página
1	Introducción	2
2	Objetivo	2
3	Estructura	4
4	Procesos organizativos	5
4.1	Proceso: Preparación y provisión	5
4.1.1.	Actividad: Adquisición	5
4.1.2.	Actividad: Equipamiento	6
4.1.3.	Actividad: Infraestructura técnica	9
4.2	Proceso: Administración del proyecto	10
4.2.1.	Actividad: Gestión del proyecto	11
4.2.2.	Actividad: Gestión del cambio	11
5	Procesos operativos	11
5.1	Proceso: Análisis	12
5.2	Proceso: Diseño	14
5.3	Proceso: Implementación	15
5.4	Proceso: Implantación y aceptación del sistema	18
5.5	Proceso: Aseguramiento de la calidad	19
5.5.1.	Actividad: Verificación documental	20
5.5.2.	Actividad: Auditorías SQA	20
6	Flujo de procesos	23
7	Anexo I. Arquitectura	24
8	Anexo II. Contextos de albergue	26
9	Anexo III Herramientas de desarrollo	27
10	Anexo IV. Sistemas horizontales	28

1 Introducción

En el entorno de GV-EJIE para cada proyecto que es adjudicado y construido el proceso de desarrollo del software viene siendo aplicado bajo distintos criterios acordados entre proveedor y contratante. En las distintas fases de desarrollo del producto o servicio, la sistemática de trabajo y traspaso de los entregables resultado ha podido ser resuelta en muchos casos con distintas herramientas con el mismo cometido, incluso se han aplicado distintos modelos de trabajo que en la práctica han hecho dificultar el propio proceso de desarrollo.

Aunque ya están definidas las distintas normativas a seguir para cada problemática y fase del desarrollo, éstas acusan cierta dispersión y falta de coherencia en su conjunto, no por ello dejando de ser perfectamente válidas.

Desde una perspectiva más tecnológica es conveniente que todas las actividades y tareas necesarias en el proceso de desarrollo del software sean estandarizadas e interconectadas entre sí facilitando y acelerando de este modo ya no solo dicho proceso sino también la propia operativa productiva de los sistemas.

2 Objetivo

Es objeto del presente documento:

- Establecer el conjunto mínimo de requerimientos y recomendaciones técnicas que estandaricen el proceso de desarrollo de software en las fases definidas por las metodologías de aplicación.
- Definir una serie de instrucciones de trabajo estandarizadas y coherentes en dicho proceso.
- Proporcionar un marco de referencia de terminología y vocabulario común para el desarrollo del software.
- Identificar las principales expectativas a gestionar en las distintas modalidades de contratación del desarrollo. Con recursos humanos y/o materiales propios y/o ajenos, etc. Se fijará especialmente en los entregables (código, ejecutables, documentación, etc.) a exigir y auditar.
- Estandarizar las herramientas de gestión asociadas a las fases del ciclo de vida del proceso de desarrollo del software.
- Enfatizar las necesidades de gestión de la calidad de los productos a implantar orientadas a minimizar al máximo los fallos del servicio en el entorno de producción.
- Limitar las posibles arquitecturas software y hardware, ajustándolas al modelo de albergue de GV-EJIE.

El estándar contiene un conjunto de procesos, actividades y tareas diseñadas para ser utilizadas en los proyectos de desarrollo software, alineadas en todos los casos con la metodología normalizada.

Se ha tomado como referencia el estándar para los procesos de ciclo de vida del software de la organización ISO, “**ISO/IEC 12207** Information Technology / Software Life Cycle Processes”, el cual establece un proceso de ciclo de vida para el software que incluye procesos y actividades que se aplican desde la definición de requisitos, pasando por la adquisición y configuración de los servicios del sistema, hasta la finalización de su uso. Este estándar tiene como objetivo principal proporcionar una estructura común para que compradores, proveedores, desarrolladores, personal de mantenimiento, operadores, gestores y técnicos involucrados en el desarrollo de software usen un lenguaje común. Este lenguaje común se establece en forma de procesos bien definidos.

El estándar no define explícitamente el contenido, nombre o formato de los entregables a producir.

El estándar será aplicable en la provisión del desarrollo y mantenimiento de un sistema software.








No contendrá el detalle de las normas definidas y aprobadas, sólo será un compendio ordenado de éstas dentro de la secuencia de fases del ciclo de vida del desarrollo software. Las normativas sólo estarán referenciadas pero no descritas íntegramente.

El ámbito de aplicación del presente documento se circunscribe tanto a los desarrollos gestionados desde GV como a los contratados desde EJIE.

El estándar identifica al menos los siguientes roles:

- **Jefe de proyecto de EJIE.** Responsable de la provisión del servicio de desarrollo o de mantenimiento de aplicaciones demandado por el cliente y de la calidad de los productos obtenidos.
- **Cliente.** El cliente es la persona que define y acuerda con el jefe de proyecto de EJIE la provisión del servicio de desarrollo o de mantenimiento de una aplicación. Se refiere al director de servicios del departamento que demanda la prestación.
- **Usuario final.** La persona que utiliza y explota el producto o aplicación obtenido como resultado del servicio contratado.
- **Proveedor.** Tercero responsable de suministrar todo o parte del servicio de desarrollo de aplicaciones solicitado por el cliente.
- **Analista responsable del proyecto de EJIE.** Responsable de la entrega del servicio de desarrollo de aplicaciones contratado por el cliente. Se refiere al técnico de análisis de EJIE que gestiona y participa en la provisión del servicio.
- **Responsable del sistema.** Responsable de la entrega del servicio de desarrollo o de mantenimiento de aplicaciones convenido. Se refiere al jefe de proyecto asignado por el proveedor externo como responsable último del servicio ante el jefe de proyecto de EJIE.
- **Responsable del módulo.** Responsable del desarrollo de un módulo o subsistema en los que se descompone el sistema software. Se refiere al analista o analista-programador asignado por el proveedor externo para la construcción de un módulo.
- **Desarrollador.** Responsable del desarrollo de un conjunto de artefactos que componen un módulo o subsistema. Se refiere al analista-programador o programador asignado por el proveedor externo para la construcción de un módulo.
- **Tester.** Responsable del diseño y ejecución de las pruebas, y del cálculo de los indicadores. Se refiere al equipo de pruebas (responsable de pruebas e ingenieros de pruebas).
- **Oficina técnica de calidad (OTC).** Vela por la ejecución de todas las actividades relacionadas con el aseguramiento de la calidad del producto software a obtener y definidas en el modelo SQA (Software Quality Assurance). Verifica el cumplimiento de las metodologías de aplicación (ARINbide y Probamet), y por ende, de los entregables documentales que estas establecen. Se refiere al equipo de SQA (responsable SQA e ingenieros SQA).
- **Oficina técnica de calidad de EJIE (OTC-EJIE).** Supervisa las actividades realizadas por las oficinas técnicas de calidad de cada proyecto.
- **Oficina de evaluación.** Evalúa los niveles de calidad obtenidos para el producto software obtenido e informa al CAB sobre dichos resultados.
- **CAB (Change advisory board).** Personal que asesora al gerente de cambios en la valoración, priorización y planificación de los cambios.

El lenguaje utilizado en la redacción de los procesos, actividades y tareas de los estándares indicará el grado de exigencia para su conformidad:

	Deberá	Indica un requisito obligatorio para su conformidad.
	Debería	Indica una fuerte recomendación que no es obligatoria para su conformidad.
	Puede	Indica una forma autorizada de cumplir un requisito o de evitar la necesidad de satisfacer la conformidad.
	Si...entonces:  deberá  debería  puede	Indica que el grado de conformidad indicado está sujeto al cumplimiento de ciertas condiciones.



Del mismo modo se remarcan aquellos aspectos que deben ser tratados y cumplidos de manera rigurosa

3 Estructura

El estándar se descompone en:

Procesos organizativos

Engloba los procesos más relacionados con la gestión y preparación, necesarios para el desarrollo del sistema software.

- Proceso: **Preparación y provisión.**
Definir el alcance del desarrollo a realizar, las infraestructuras técnicas necesarias para su ejecución y los acuerdos de gestión del proceso.
- Proceso: **Administración del proyecto.**
Comprende las actividades y tareas necesarias para la correcta administración del proyecto.

Procesos operativos

Define los procesos a realizar desde el punto de vista de la construcción del sistema software.

- Proceso: **Análisis**
Comprende las actividades y tareas dictadas por la metodología de aplicación ARINbide en la fase de *análisis del sistema de información* (ASI), y las definidas por Probamet en la fase de *planificación de pruebas* (PPB1), y parcialmente, en la fase de *análisis y diseño de las pruebas* (APB), indicando además las herramientas de uso en cada actividad y tarea.
- Proceso: **Diseño**
Comprende las actividades y tareas dictadas por la metodología de aplicación ARINbide en la fase de *diseño del sistema de información* (DSI), y parcialmente, las definidas por Probamet en la fase de *análisis y diseño de las pruebas* (APB)., indicando además las herramientas de uso en cada actividad y tarea.
- Proceso: **Implementación**
Comprende las actividades y tareas dictadas por la metodología de aplicación ARINbide en la fase de *construcción del sistema de información* (CSI), y parcialmente, las definidas por Probamet en la fase de *ejecución de pruebas* (EPB), indicando además las herramientas de uso en cada actividad y tarea.
Se indican también las mejores prácticas de aplicación en dicha fase, orientadas a mejorar tanto la ejecución del propio proceso, como a optimizar la tarea de entrega del sistema software.

- Proceso: **Implantación y aceptación del sistema**
Comprende las actividades y tareas dictadas por la metodología de aplicación ARINbide en la fase de implantación y aceptación del sistema (IAS), y parcialmente, las definidas por Probamet en la fase de *ejecución de pruebas* (EPB).
- Proceso: **Aseguramiento de la calidad**
Asegurará que los productos obtenidos y los procesos del ciclo de vida del proyecto cumplen con los requerimientos y los planes establecidos.

4 Procesos organizativos

- ↑ En el proceso de desarrollo de los sistemas software se **deberá** aplicar:
- La metodología de desarrollo de aplicaciones ARINbide.
 - La metodología de pruebas Probamet.
 - El modelo de aseguramiento de la calidad estándar de GV-EJIE (Modelo SQA)

4.1 Proceso: Preparación y provisión

Definir el alcance del desarrollo a realizar, de las infraestructuras técnicas necesarias para su ejecución y de los acuerdos de gestión del proceso.

Este proceso se descompone en las actividades descritas a continuación.

4.1.1. Actividad: Adquisición

↑ El jefe de proyecto de EJIE **deberá** definir las necesidades del sistema a construir, adquirir o mantener, indicando además en el pliego de bases técnicas la obligatoriedad de adscripción a los presentes estándares, incluyendo entonces la exigencia en la aplicación de las metodologías ARINbide y Probamet, y del modelo de aseguramiento de la calidad del software (modelo SQA).

↑ Se **deberá** indicar también el valor NAC¹ (nivel de aseguramiento de la calidad) asignado al proyecto.



↑ El jefe de proyecto de EJIE **deberá** señalar en el pliego de bases técnicas las implicaciones emanadas de las siguientes obligaciones y recomendaciones relativas a las actividades marcadas por la metodología de desarrollo ARINbide:

- La fase de *Requisitos de usuario* que consta de las actividades de *Definición del Sistema* (ASI 1) y la de *Establecimiento de Requisitos* (ASI 2) ↑ **deberán** ser realizadas por el analista responsable del proyecto de EJIE
- Y al menos las actividades de *Identificación de subsistemas de análisis* (ASI 3), *Análisis de casos de uso* (ASI 4) y la de *Análisis de clases* (ASI 5), ↻ **deberían** también ser realizadas por el analista responsable del proyecto de EJIE

↑ El jefe de proyecto de EJIE **deberá** decidir la idoneidad de la contratación de un equipo adicional que actúe como Oficina Técnica de Calidad para el proyecto, por lo tanto, distinto al que realizará las actividades de ingeniería, y al que realizará las tareas de pruebas. Incluso se podría decidir que las tareas de pruebas pasen a ser responsabilidad de dicha Oficina Técnica de Calidad. En cualquier caso se **deberá** asegurar que dicho rol queda cubierto.

¹ El valor NAC (Nivel de Aseguramiento de la Calidad) permite determinar el grado de calidad necesario para un proyecto de desarrollo software, y en consecuencia, identificar qué controles mínimos serán de obligado cumplimiento, las tipologías y niveles de pruebas a ejecutar, y los umbrales admitidos para los indicadores identificados en el modelo de aseguramiento de la calidad.

⬆ El jefe de proyecto de EJIE **deberá** realizar el proceso de selección del proveedor, acordando los modelos de prestación del servicio más adecuados a las necesidades demandadas por el cliente, y proponiendo así la contratación del proveedor más apropiado.

4.1.2. **Actividad: Equipamiento**

Proveer de los bienes básicos para la ejecución del proyecto de desarrollo de software. Comprende tanto el hardware, el software, las herramientas, los estándares y las facilidades para el desarrollo, operación y mantenimiento.

Este proceso se descompone en las tareas descritas a continuación.

Tarea: Provisión del puesto de trabajo.

⬆ Sea cual sea la modalidad de la contratación, los miembros de los equipos de desarrollo y mantenimiento **deberán** proveerse del mismo conjunto de herramientas y recursos hardware homologados en el entorno de GV-EJIE.

⚡ O **si** se acordó que el puesto de trabajo estuviese en los locales del proveedor, **entonces** éste **debería** proveerse del mismo conjunto de herramientas y hardware que las especificadas para GV-EJIE, o bien de aquellas que considere suficientes para cubrir adecuadamente el proceso de desarrollo software para el que ha sido contratado, facilitando y acelerando así la entrega de los productos construidos.

Recursos software:

⬆ Se ha establecido el conjunto mínimo de herramientas que se **deberá** disponer en base al perfil del usuario asignado. Todos los perfiles (jefe de proyecto, analista, o desarrollador) contarán con un mismo conjunto básico de herramientas, ampliándose éste para el perfil de desarrollador.

Herramientas comunes para jefe de proyecto, analista o desarrollador:

- Software de base:
 - Windows XP
 - Microsoft Office 2003 u OpenOffice
- Navegadores:
 - Internet Explorer
 - Mozilla Firefox
 - Chrome
 - Opera
- Diseño de imágenes:
 - Gimp
- Utilidades para desarrollo:
 - Framework .NET. Necesario para ejecución aplicaciones .NET en local
 - Java JRE: Necesario para ejecución aplicaciones .Java en local
- Bases de datos Oracle:
 - Cliente Oracle. Herramientas administrativas cliente de Oracle
 - Oracle SQL Developer. Gestión de base de datos Oracle (v>=9)

- Bases de datos SQL-Server:
 - Cliente SQL-Server. Herramientas administrativas cliente de SQL-Server
- Herramientas de desarrollo y pruebas:
 - Microsoft Project. Planificación de proyectos.
 - Enterprise Architect. Modelado UML, diagramación, documentación, e ingeniería inversa.
 - XML Spy. Edición, depuración y tratamiento de XML y tecnologías asociadas.
 - Subversion: Sistema open-source para control de versiones.
 - AIS web accessibility (AWA) Toolbar para Internet Explorer. Utilidades para análisis de normas WAI, XHTML, CSS
 - Web Developer extensión (WDE) Toolbar para Firefox. Utilidades para análisis de normas WAI, XHTML, CSS
 - Badboy y Selenium. Utilidad para generación de scripts de navegación para un sitio web.
 - JMeter. Pruebas de rendimiento y stress sobre aplicaciones web (sólo para cliente)
 - KeyToolUI: Herramienta gráfica para la manipulación de formatos criptográficos. (Certificados, keystores, firmas, claves en diversos formatos).
 - Xolido Sign: Herramienta gráfica de firma electrónica.
 - SoapUI: Herramientas para testeo de servicios web.
 - Portal SQA: Aplicación web corporativa de EJIE de gestión y consulta de resultados del Modelo SQA
- Gestión de errores e incidencias: Registro, control y gestión de las incidencias generadas en un proyecto:
 - Mantis: Aplicación web para el registro y gestión de bugs.
- Gestión de pruebas: Registro, control y gestión de pruebas en un proyecto:
 - TestLink: Aplicación web para la gestión de pruebas (requisitos, planes de pruebas, casos de prueba).
- Gestión de librerías y dependencias: control de uso de librerías de aplicación:
 - Maven: repositorio de librerías y utilidades para la descarga y publicación
 - Archiva: herramienta con interfaz gráfica para gestión de repositorios propios (departamental o de aplicaciones horizontales)

Solo para perfil desarrollador:

- Software de desarrollo:
 - Firebug: Extensión de Firefox compuesta por un paquete de utilidades para editar, monitorizar y depurar el código fuente, CSS, HTML y JavaScript de una página web.
 - Fiddler: proxy para capturar y analizar el tráfico de las peticiones web
- Herramientas de desarrollo y pruebas:
 - QuickREx: Plugin de ayuda a la creación de expresiones regulares.

Solo para perfil desarrollador J2EE/JEE:

- Software de desarrollo:
 - Entorno desarrollo java. Oracle Weblogic Server
 - JDK
 - Eclipse OEPE. Entorno integrado de desarrollo J2EE

- Plugin UDA: plugin corporativo para desarrollo rápido de aplicaciones
- Hibernate Tools: plugin de Eclipse que permite generar el modelo de datos a partir de una base de datos relacional
- QuickREx: gestión de expresiones regulares
- Subversive: El proyecto Subversive se ocupa de facilitar la integración de Subversion para Eclipse.
- JUnit. Plugin para Eclipse, para pruebas unitarias.
- Checkstyle. Plugin para Eclipse, para pruebas de chequeo de código, en base a reglas de codificación.
- PMD. Plugin para Eclipse, para pruebas de chequeo de código, detectando código muerto, código repetido, etc.
- FindBugs. Plugin para Eclipse de detección de errores de código Java.
- jQuery: librería Javascript para desarrollo de aplicaciones web enriquecidas

Solo para perfil desarrollador .NET:

- Software de desarrollo:
 - Visual Studio .NET. Entorno integrado de desarrollo .NET
 - AnkhSVN: Plug-in para Visual Studio para trabajo desde puesto local de desarrollo con el repositorio de versiones Subversion.
- Herramientas de desarrollo y pruebas:
 - NUnit: Permite realizar pruebas unitarias para cualquier lenguaje de .Net.

Usuarios

⬆ Se **deberá** solicitar la creación de los usuarios de red y configuraciones XLNetS necesarios en el entorno de desarrollo, y en el de pruebas cuando se estime conveniente.

Tarea: Provisión del canal de acceso.

⚠ Si se acordó que el puesto de trabajo estuviese en los locales del proveedor, **entonces** se ⬆ **deberá** proveer el canal de comunicaciones entre dicho puesto y los servidores de desarrollo a los que sea preciso acceder.

⬆ En este supuesto, el jefe de proyecto de EJIE **deberá** asegurar que el proveedor dispone de acceso desde sus oficinas a otras herramientas corporativas de desarrollo tales como el repositorio de modelos UML “Enterprise Architect”, el de gestión de versiones “Subversion”, la herramienta de gestión de errores “Mantis” (incluido el repositorio de conocimiento de EJIE), y la de gestión de pruebas “TestLink”.

⬆ Preferentemente se **deberá** optar por una conexión VPN entre el proveedor y EJIE con las siguientes prestaciones mínimas recomendadas:

- Tipo de conexión: ADSL Simétrico con IP fija
- Velocidad: 2 MB
- Software de cliente VPN (proporcionado por EJIE)

⬆ Esta conexión vía VPN **deberá** condicionarse con un acuerdo previo de compromisos entre EJIE y la empresa colaboradora, e incluyendo además la provisión por parte de EJIE del software de cliente VPN y la creación de los usuarios de conexión que se hayan acordado.

Tarea: Emulación.

💡 **Si** se acordó que el puesto de trabajo estuviese en los locales del proveedor, **entonces** el adjudicatario **debería** implantar en sus oficinas las soluciones de servidor que permitan agilizar el desarrollo del sistema software y su implantación posterior en los servidores de EJIE. ➡

Existen diversas maneras de facilitar los desarrollos de aplicaciones. En general éstas pasan por intentar emular el entorno existente en EJIE, así como sus aplicaciones horizontales (emulando-virtualizando máquinas o bien con otras técnicas).

Algunas de estas sugerencias se detallan a continuación:

- **Emulación – virtualización de máquinas**

Se reduce a intentar emular los sistemas básicos existentes en el entorno de EJIE; para ello se exponen dos de las posibilidades que existen: las imágenes virtuales y las distribuciones arrancables.

- Imágenes virtuales: Se trataría de un fichero con una máquina virtual que simulase el entorno de desarrollo en servidor de EJIE; este fichero contendría todos los sistemas utilizados en EJIE con su misma configuración.
- Distribución arrancable: El sistema que estaría implantado en esta distribución arrancable sería el mismo que la imagen virtual con la diferencia de que no se necesitaría el programa base de la imagen virtual para su ejecución sino que se ejecutaría en el arranque del sistema.

- **Emulación de aplicaciones horizontales**

Esta sugerencia consiste en buscar una fórmula que permita emular de alguna manera las aplicaciones horizontales para evitar la dependencia exclusiva de estos sistemas de EJIE y que puedan provocar retrasos en los desarrollos.

Una de las posibilidades sería utilizando clases “huecas” que devuelvan unos valores predefinidos acordes con los valores válidos que retornarían esas aplicaciones horizontales y que permitan no tener que modificar el sistema a la hora de implantar la aplicación en los servidores de desarrollo de EJIE; un ejemplo práctico puede ser emular la API de XLNets mediante unas clases “huecas” y un xml que emule un token de sesión que es interrogado desde dichas clases.

4.1.3. **Actividad: Infraestructura técnica**

Establecer y mantener la infraestructura técnica necesaria para el resto de procesos. Comprende tanto el hardware, el software, las herramientas, los estándares y las facilidades para el desarrollo, operación y mantenimiento, desde la perspectiva del uso de los sistemas de infraestructura y de su albergue.

Este proceso se descompone al menos en las tareas descritas a continuación.

Tarea: Sistema de infraestructura XLNetS y PKI

⬆ Los procesos de autenticación de usuarios mediante pares usuario-password o mediante certificado electrónico, así como la gestión de autorizaciones de acceso a los recursos de la aplicación **deberán** valerse del sistema de seguridad corporativo XLNetS.

⬆ Se **deberá** solicitar la creación de los recursos y configuraciones XLNetS del entorno de desarrollo, pruebas, y producción necesarios para el sistema software (cadena de conexión a BD, usuarios, unidades orgánicas, inclusión de los PCs de trabajo en dominios, etc.)

- ⬆ Se **deberá** seguir el procedimiento de administración de usuarios definido para la organización; SCP-34.
- ⬆ Los certificados electrónicos de uso de la aplicación **deberán** solicitarse a Izenpe.

Tarea: Sistema de infraestructura PLATEA-Tramitación

- ⬆ Se **deberá** solicitar la creación de los recursos PLATEA-Tramitación del entorno de desarrollo, pruebas, y producción necesarios para el sistema software (alta del procedimiento en el catálogo de procedimientos, alta en el catálogo de trámites y tareas, etc.)

Tarea: Sistema de infraestructura PLATEA-Presencia en Internet

- ⬆ Se **deberá** solicitar la creación de los recursos PLATEA-Presencia en Internet del entorno de desarrollo, pruebas y producción necesarios para el sistema software (workareas de aplicación, usuarios Interwoven y Open Deploy, etc.)

Tarea: Albergue

- ⬆ Para el sistema a construir, y en base a la combinación de colectivos usuarios y sus posibles canales de acceso, se **deberá** determinar el contexto o contextos de albergue así como su disposición en una o varias aplicaciones. Se **deberá** definir además que aplicación o aplicaciones deben desplegarse en los contextos de Internet, Intranet, Extranet-Jakinapulus o Extranet-Jaso. El proceso de desarrollo de cada aplicación **deberá** adscribirse en cada caso a los modelos de despliegue establecidos y a los libros de estilo vigentes.

- ⬆ En cualquier caso se **deberán** considerar las reglas de visibilidad y acceso impuestas por el sistema de seguridad de red definido en GV, tanto para el caso de invocaciones entre distintos contextos como para peticiones dentro del mismo. Cualquier requerimiento de arquitectura de sistema que no cumpla el modelo de seguridad de red **deberá** ser estudiado y aprobado, si procede, por EJIE, siempre como paso previo al inicio de cualquiera de los procesos operativos.

Tarea: Documentación, normativas y notas técnicas

- ⬆ El jefe de proyecto de EJIE **deberá** facilitar al proveedor el conjunto de las normativas vigentes, y de toda la documentación que se considere de interés en la construcción del sistema software, así como de todas aquellas notas técnicas internas que sean publicadas. **Deberá** prestar especial atención a las actualizaciones que vayan surgiendo de modo que el proveedor esté continuamente informado de las novedades o cambios que se produzcan.
- ⬆ **Si** se acordó la provisión del puesto de trabajo en los locales del proveedor **entonces** el jefe de proyecto de EJIE ⬆ **deberá** asegurar que éste dispone de acceso al repositorio de conocimiento técnico de EJIE, al cual podrá acudir ante cualquier eventualidad técnica que surja.

4.2 Proceso: Administración del proyecto

Comprende las actividades y tareas necesarias para la correcta administración del proyecto.

Este proceso se descompone en las actividades descritas a continuación.

4.2.1. **Actividad: Gestión del proyecto**

Comprende las actividades y tareas dictadas por la metodología de aplicación ARINbide en la fase de *gestión del proyecto* (GPR), las definidas por Probamet en su fase de *seguimiento de pruebas* (PPB2), y las marcadas por el modelo de aseguramiento de la calidad (Modelo SQA) en su fase de *ejecución y seguimiento de las actividades SQA* (SQA 3).

- ⬆ Se **deberán** ejecutar las actividades y tareas definidas por ARINbide en el módulo de *gestión de proyectos* (GPR), así como las indicadas por Probamet en su fase de *seguimiento de pruebas* (PPB2), y las establecidas por el Modelo SQA en su fase de *ejecución y seguimiento de las actividades SQA* (SQA 3).
- ⬆ Se **deberá** seguir el procedimiento de gestión de proyectos definido para la organización; SCP-60 para el caso de contrataciones gestionadas por EJIE.
- ⬆ Para la generación de la documentación se **deberán** utilizar las herramientas ofimáticas homologadas en GV-EJIE, Microsoft Office u OpenOffice.
- ⬆ Para elaborar los entregables de planificación se **deberá** utilizar Microsoft Project o cualquier otra herramienta que permita su importación posterior en ésta.

4.2.2. **Actividad: Gestión del cambio**

- ⬆ Se **deberá** seguir el procedimiento de gestión de cambios definido para la organización.

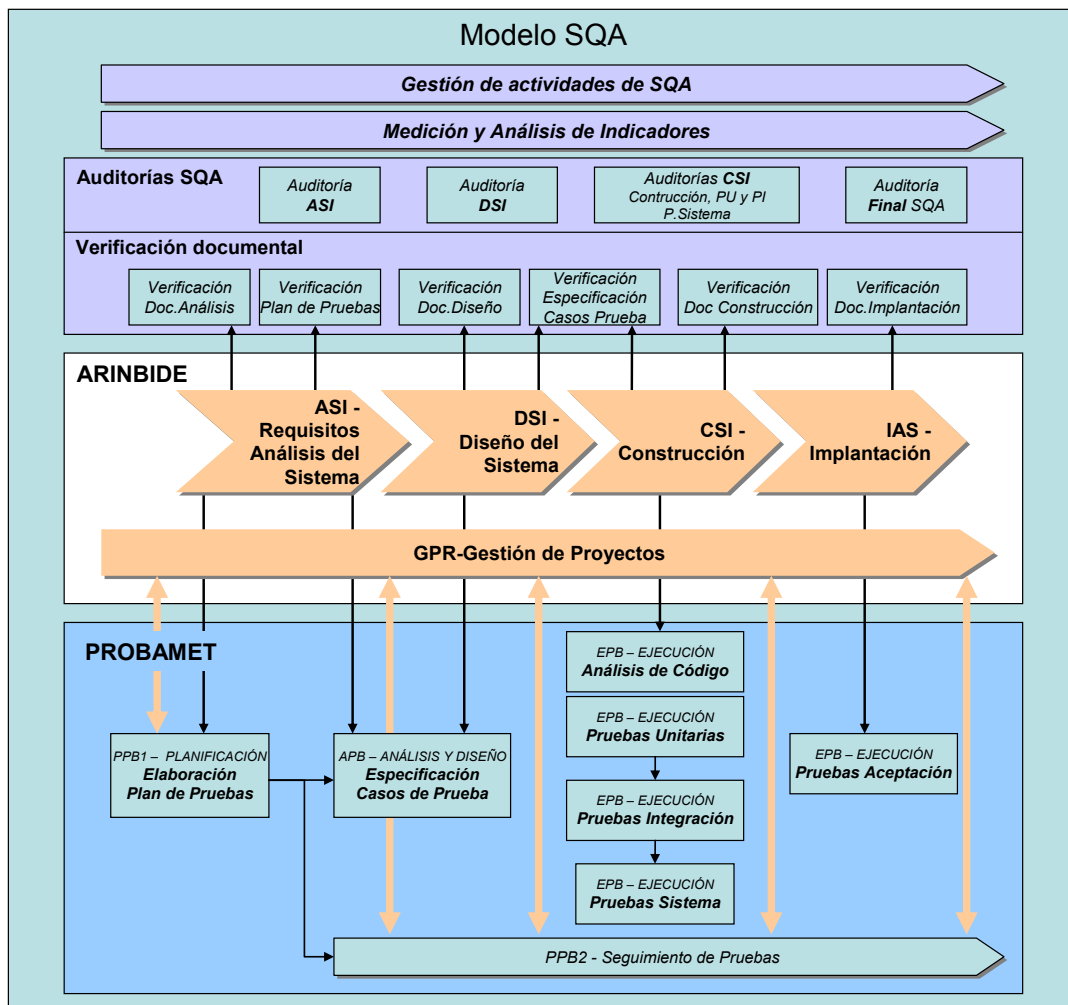


Los traspasos entre entornos (desde desarrollo a pruebas, y desde pruebas a producción) ⬆ **deberán** ser planificados y propuestos con antelación suficiente para que los servicios de gestión del cambio y gestión del despliegue de EJIE puedan planear adecuadamente los trabajos a realizar.

- ⬆ Para la generación de la documentación se **deberá** proveer de Microsoft Office o de OpenOffice como herramienta ofimática homologada en GV-EJIE.

5 **Procesos operativos**

- ⬆ Se **deberá** aplicar:
 - La metodología de desarrollo de aplicaciones ARINbide.
 - La metodología de pruebas Probamet.
 - El modelo de aseguramiento de la calidad estándar de GV-EJIE (Modelo SQA)



5.1 Proceso: Análisis

Comprende las actividades y tareas dictadas por la metodología de aplicación ARINbide en la fase de *análisis del sistema de información* (ASI), y las definidas por Probamet en la fase de *planificación de pruebas* (PPB1), y parcialmente, en la fase de *análisis y diseño de las pruebas* (APB). Se indica para cada caso las herramientas de uso en cada actividad y tarea.

Puesto que el proceso se corresponde con las fases de ARINbide y de Probamet ya indicadas, no se detallan las actividades y tareas en las que se descompone, sólo se remarcan ciertos aspectos especialmente importantes que **deberán** o **deberían** cumplirse.

⬆ Se **deberán** ejecutar las actividades y tareas definidas por ARINbide en esta fase, así como las dictadas por Probamet para este ciclo. Estas se **deberán** realizar considerando el uso de sólo las herramientas, librerías y sistemas especificados en el documento de estándares tecnológicos publicado por la Dirección de Informática y Telecomunicaciones del GV. Ante cualquier otra nueva necesidad, EJIE **deberá** realizar un estudio detallado determinándose la idoneidad o no de la propuesta.

⬆ Las actividades de *Definición del Sistema* (ASI 1) y la de *Establecimiento de Requisitos* (ASI 2) definidas por la metodología de desarrollo ARINbide en su fase de *Requisitos de usuario*, ⬆ **deberán** ser realizadas por

el analista responsable del proyecto de EJIE.

➡ Y al menos las actividades de *Identificación de subsistemas de análisis* (ASI 3), *Análisis de casos de uso* (ASI 4) y la de *Análisis de clases* (ASI 5), también ➡ **deberían** ser realizadas por el analista responsable del proyecto de EJIE.

⬆ Para la generación de la documentación se **deberá** utilizar Microsoft Office u OpenOffice como herramienta ofimática homologada en GV-EJIE. ⚡ Y **si** se acordó que el puesto de trabajo estuviese en los locales del proveedor, **entonces** éste ➡ **puede** utilizar cualquier otra herramienta que permita generar los formatos compatibles que puedan ser gestionados por alguna de estas.

⬆ Para la creación de los modelos UML y de sus descripciones detalladas a incluir en la documentación, se **deberá** utilizar "Enterprise Architect". ⚡ Y **si** se acordó que el puesto de trabajo estuviese en los locales del proveedor, **entonces** éste ➡ **puede** utilizar cualquier otra herramienta que permita generar el formato estándar XMI, de modo que dichos modelos resultado puedan ser gestionados por esta.

⬆ Se **deberá** utilizar un único repositorio de modelos UML. Si el proyecto es de cierta envergadura dicho repositorio se **deberá** crear en el servidor remoto habilitado a tal efecto, facilitando así el trabajo en equipo

⬆ En la actividad de definición de interfaces de usuario, los prototipos **deberán** construirse con un software de prototipado al uso (como por ejemplo Pencil o Microsoft Web Express) si se opta por la opción de posibilidad de navegación, o con GIMP si se prefieren modelos más estáticos. Y **si** se acordó que el puesto de trabajo estuviese en los locales del proveedor, **entonces** éste ➡ **puede** utilizar cualquier otra herramienta que permita generar los formatos compatibles con estas.

⬆ Se **deberá** poner especial atención al proceso de revisión y aprobación de requisitos definidos y especificados para el sistema software.



- ⬆ El valor NAC asignado al proyecto determinará las tipologías y niveles de pruebas a ejecutar. Así, el plan de pruebas **deberá** contemplar la ejecución del conjunto de:
 - Pruebas de aceptación. Que deberá realizar el usuario en el entorno de pruebas, y que permitirán determinar la aceptación o no del sistema construido.
 - Pruebas de sistema. Conjunto de pruebas del sistema software que aseguren la cobertura completa de la especificación de requisitos.
 - Pruebas funcionales. Que permitan asegurar que se cumplen los requisitos funcionales definidos para cada uno de los módulos de análisis del sistema.
 - Pruebas no funcionales. Los ensayos de instalación y configuración, de consistencia de datos, de seguridad, de prestaciones (rendimiento, escalabilidad, capacidad, carga, estrés, volumen y estabilidad), de fallo y recuperación del sistema, de accesibilidad y de usabilidad.
- ⬆ El plan de pruebas **deberá** identificar también:
 - Pruebas de integración. Se **deberá** establecer al menos la estrategia de pruebas de integración a realizar.
 - Pruebas unitarias. En esta fase se **deberán** identificar al menos los caminos y procesos críticos (riesgos) del sistema a partir de los cuales en la fase de implementación se crearán los artefactos de pruebas básicas de componentes.
- ⬆ En esta fase se **deberán** definir al menos a alto nivel los conjuntos de casos de prueba

emanados de las pruebas de aceptación y de sistema.

5.2 Proceso: Diseño

Comprende las actividades y tareas dictadas por la metodología de aplicación ARINbide en la fase de *diseño del sistema de información* (DSI), y parcialmente, las definidas por Probamet en la fase de *análisis y diseño de las pruebas* (APB). Se indican para cada caso las herramientas de uso en cada actividad y tarea.

Puesto que el proceso se corresponde con las fases de ARINbide y de Probamet ya indicadas, no se detallan las actividades y tareas en las que se descompone, sólo se remarcen ciertos aspectos especialmente importantes que **deberán** o **deberían** cumplirse.

⬆ Se **deberán** ejecutar las actividades y tareas definidas por ARINbide en esta fase, así como las dictadas por Probamet para este ciclo. Éstas se **deberán** realizar considerando el uso de sólo las herramientas, librerías y sistemas especificados en el documento de estándares tecnológicos publicado por la Dirección de informática y telecomunicaciones del GV. Ante cualquier otra nueva necesidad, EJIE **deberá** realizar un estudio detallado determinándose la idoneidad de la propuesta.

⬆ Para la generación de la documentación se **deberá** utilizar Microsoft Office u OpenOffice como herramienta ofimática homologada en GV-EJIE. ⚡ Y **si** se acordó que el puesto de trabajo estuviese en los locales del proveedor, **entonces** éste **puede** utilizar cualquier otra herramienta que permita generar los formatos compatibles que puedan ser gestionados por alguna de estas.

⬆ Para la creación de los modelos UML y de sus descripciones detalladas a incluir en la documentación, se **deberá** utilizar "Enterprise Architect". ⚡ Y **si** se acordó que el puesto de trabajo estuviese en los locales del proveedor, **entonces** éste **puede** utilizar cualquier otra herramienta que permita generar el formato estándar XMI, de modo que dichos modelos resultado puedan ser gestionados por esta.

⬆ Se **deberá** utilizar el mismo repositorio de modelos UML y definiciones detalladas que el solicitado en la fase de análisis.



⬆ Se **deberá** completar exhaustivamente el plan de pruebas establecido en el proceso de análisis y almacenado en el repositorio "TestLink", ampliando y detallando además el conjunto de casos de prueba a realizar.

⬆ Si se trata de un aplicativo con interfaz web que será desplegado en el contexto de internet, las pruebas del sistema **deberán** contemplar los ensayos necesarios para garantizar que el sistema se comporta correctamente en los navegadores más utilizados (Internet Explorer, Firefox, Chrome, Opera, y Safari)². Mientras que si el contexto de despliegue es intranet (con PCs gestionados), el sistema **deberá** soportar los navegadores homologados por los estándares tecnológicos publicados por la Dirección de informática y telecomunicaciones del GV para dicho contexto (Internet Explorer y Firefox). Para el caso de extranet, la lista de navegadores a soportar se **deberá** decidir en función de los colectivos usuarios de la aplicación y del uso o no de PCs gestionados.

⬆ Se **deberá** establecer perfectamente la matriz de trazabilidad requisito – casos de prueba que permitirá validar que éstos se cumplen y se satisfacen adecuadamente.

² Se han incluido los navegadores más utilizados a nivel mundial a fecha de Enero de 2010. No obstante, y con objeto de dar cobertura al máximo número de usuarios en el momento de realizar el proyecto, se recomienda consultar las estadísticas de uso de los navegadores web en el contexto de los interesados de la aplicación que se va a construir, para así ampliar o reducir para cada caso la lista de navegadores a soportar.

↑ Con objeto de minimizar el esfuerzo a realizar, y haciendo uso del conjunto de herramientas homologadas, se **deberá** sistematizar al máximo el proceso de ejecución de las pruebas que se hayan definido.

- ↑ Se **deberán** tener en consideración las restricciones impuestas por el modelo de seguridad de red y por ende las arquitecturas lógicas y físicas que este impone. Para más información consultar el *Anexo I. Arquitectura*.
- ↑ Se **deberá** considerar especialmente la información necesaria a reflejar en el manual de implantación del sistema software. Para más información consultar el *proceso de implantación y aceptación del sistema (5.4)*
- ↑ Las tareas relacionadas con la gestión de pruebas establecidas por Probamet **deberán** realizarse con la herramienta de gestión de pruebas estándar, TestLink, conteniendo por lo tanto el conjunto de requisitos definidos para el proyecto, así como el detalle de los casos de prueba que los validan.

5.3 Proceso: Implementación

Comprende las actividades y tareas dictadas por la metodología de aplicación ARINbide en la fase de *construcción del sistema de información (CSI)*, y parcialmente, las definidas por Probamet en la fase de *ejecución de pruebas (EPB)*. Se indica para cada caso las herramientas de uso en cada actividad y tarea.

Puesto que el proceso se corresponde con las fases de ARINbide y de Probamet ya indicadas, no se detallan las actividades y tareas en las que se descompone, sólo se remarcan ciertos aspectos especialmente importantes que **deberán** o **deberían** cumplirse.

Se indican también las mejores prácticas de aplicación en dichas fases, orientadas a mejorar tanto la ejecución de los propios procesos como a optimizar la tarea de entrega del sistema software.

- ↑ Se **deberán** ejecutar las actividades y tareas definidas por ARINbide en esta fase, así como las dictadas por Probamet para este ciclo. Éstas se **deberán** realizar considerando el uso de sólo las herramientas, librerías y sistemas especificados en el documento de estándares tecnológicos publicado por la Dirección de informática y telecomunicaciones del GV. Ante cualquier otra nueva necesidad, EJIE **deberá** realizar un estudio detallado determinándose la idoneidad de la propuesta.
- ↑ Para la generación de la documentación se **deberá** utilizar Microsoft Office u OpenOffice como herramienta ofimática homologada en GV-EJIE. 💡 Y **si** se acordó que el puesto de trabajo estuviese en los locales del proveedor, **entonces** éste 🚫 **puede** utilizar cualquier otra herramienta que permita generar los formatos compatibles que puedan ser gestionados por alguna de estas.
- ↑ Para la creación de los modelos UML a incluir en la documentación se **deberá** utilizar “Enterprise Architect”. 💡 Y **si** se acordó que el puesto de trabajo estuviese en los locales del proveedor, **entonces** éste 🚫 **puede** utilizar cualquier otra herramienta que permita generar el formato estándar XMI, de modo que dichos modelos resultado puedan ser gestionados por esta.
- ↑ Se **deberá** utilizar el mismo repositorio UML que el solicitado en el proceso de análisis (5.1).
- ↑ El procedimiento de catalogación y gestión de errores que puedan producirse durante el proceso de implementación de cualquiera de las áreas del sistema software (código ejecutable, elementos estáticos, scripts de creación de BD, ejecución del plan de pruebas, etc.) **deberán** tratarse utilizando la herramienta “Mantis”.

Previamente se **deberá** crear dicho repositorio y asignar los roles (informador, validador, resolutor, etc.) y usuarios que se consideren necesarios.

- ⬆ Las tareas relacionadas con la gestión de pruebas establecidas por Probamet **deberán** realizarse con la herramienta de gestión de pruebas estándar, TestLink, conteniendo por lo tanto el conjunto de requisitos definidos para el proyecto, el detalle de los casos de prueba que los validan, y los resultados obtenidos para cada caso de prueba.
- ⬆ El tratamiento de los ficheros “planos” se **puede** realizar con la herramienta “UltraEdit32”.
- ⬆ El diseño de los ficheros XML se **puede** realizar con la herramienta “XMLSpy”.
- ⬆ El acceso al servidor de desarrollo de EJIE se **deberá** realizará con la herramienta “SecureCRT”, o con “SecureFX” si lo que se desea es hacer una carga FTP. ⚡ Y **si** se acordó que el puesto de trabajo estuviese en los locales del proveedor, **entonces** éste **puede** utilizar cualquier otra herramienta compatible con la configuración del servidor.



⬆ Se **deberán** generar y ejecutar intensivamente los casos de prueba definidos en los procesos anteriores para el sistema software.

Contenidos estáticos

- ⬆ El diseño de las interfaces gráficas en formato HTML junto con el resto de artefactos web (CSS, javascript, etc.) se **deberá** realizar con “Microsoft Web Designer”
- ⬆ Las imágenes se **deberán** crear con la herramienta “GIMP”. ⚡ Y **si** se acordó que el puesto de trabajo estuviese en los locales del proveedor, **entonces** éste **puede** utilizar cualquier otra herramienta que permita generar los formatos compatibles con esta.
- ⬆ Las entregas de contenidos estáticos se **deberán** realizar siempre a través del repositorio de versionado “Subversion”, en la rama de “estatico”.
- ⬆ Para sistemas a desplegar en el contexto de Internet, los contenidos estáticos **deberán** ser cargados en la “workarea” de aplicación del servidor de contenidos “Interwoven”. Posteriormente se **deberán** desplegar en los servidores web Apache con la herramienta de “Open Deploy”. Este proceso se repetirá en los entornos de pruebas y de producción.
- ⬆ En el resto de contextos (Intranet y Extranet) se **deberán** copiar estos contenidos en el servidor web “Apache” del entorno correspondiente haciendo una carga FTP. Y para el paso a producción se **deberá** utilizar la aplicación M26 para su despliegue definitivo en dicho entorno.

Código, configuración y pruebas

- ⬆ **Si** se trata de un sistema con tecnología JEE **entonces** se **deberá** utilizar “UDA” como utilidad de generación de código básico a partir del cual se construirá el sistema.
- ⬆ Como herramienta de desarrollo IDE en entorno J2EE se **deberá** utilizar “Eclipse” junto con el plugin “UDA”.



⬆ El proceso de codificación y el de generación del conjunto de pruebas unitarias se **deberá** realizar **en el PC local**. **No se permite** desplegar en el servidor de desarrollo ningún código que no haya sido perfectamente probado y validado previamente en el entorno de PC local.

⬆ **No se permite** utilizar el servidor de desarrollo como única herramienta o entorno de compilación y pruebas unitarias.

⬆ Las entregas de código, de los ficheros de configuración en el servidor de desarrollo y de los conjuntos de pruebas automatizadas (unitarias, integración y sistema) se **deberán** realizar siempre a través del repositorio de versionado “Subversion” en la rama de “codigo”, en la de “config”, y en las de “test” (test_unit, test_integration y test_system) respectivamente. Es decir, desde el PC de desarrollo se alimentará el repositorio, y desde éste y haciendo uso de las tareas habilitadas a tal efecto se descargará en el servidor de desarrollo para su compilación y despliegue.

⬆ Haciendo uso de la herramienta para la automatización de la ejecución de pruebas, “Hudson”, el desarrollador **deberá** lanzar el proceso de análisis estático del código, así como todas aquellas pruebas unitarias y de integración automáticas que se hayan configuradas. Este proceso se podrá ejecutar tantas veces como se considere necesario. Los resultados obtenidos se recogerán automáticamente en la herramienta de gestión de pruebas “TestLink” y las incidencias producidas se darán de alta en la herramienta de gestión de errores “Mantis”.

⬆ Sólo el responsable del módulo **podrá** realizar el despliegue de dicho módulo en el servidor de desarrollo. Esta acción se **deberá** hacer de manera programada y coordinada con el resto de responsables de módulos, idealmente un máximo de una vez al día hasta que el módulo testeado se considere correcto.



⬆ Se **deberá** reducir por tanto al máximo el número de despliegues a realizar en el servidor de desarrollo.

⬆ Cualquier código que esté generando problemas de estabilidad en el servidor de desarrollo **deberá** ser descargado y subsanado en el entorno PC local, nunca en el servidor.

✓ **Una vez desplegado el módulo:**

⬆ Haciendo uso de la herramienta de generación de scripts automáticos de pruebas de sistema-funcionales, “Selenium”, el responsable del módulo **deberá** generar los escenarios de validación y pruebas de cada uno de los módulos contenidos en su ámbito de responsabilidad.

⬆ El responsable del módulo y/o el desarrollador **deberá** generar también el conjunto de pruebas de sistema no funcionales: de instalación y configuración, de consistencia de datos, de seguridad, de prestaciones (rendimiento, escalabilidad, capacidad, carga, estrés, volumen y estabilidad), de fallo y recuperación del sistema, de accesibilidad y de usabilidad. Se **deberá** utilizar el conjunto de herramientas estandarizadas para cada caso (“Ais Web Accessibility For Internet Explorer”, soapUI, etc.). Para más información consultar el *Anexo III. Herramientas de desarrollo*.

⬆ Haciendo uso de la herramienta para la automatización de la ejecución de pruebas, “Hudson”, el responsable del módulo **deberá** lanzar, para cada uno de sus módulos, el proceso de validación de todas aquellas pruebas de sistema automáticas que se hayan configurado. Este proceso se podrá ejecutar tantas veces como se considere necesario. Los resultados obtenidos se **deberán** recoger en la herramienta de gestión de pruebas “TestLink” y las incidencias producidas se **deberán** dar de alta en la herramienta de gestión de errores “Mantis”.

⬆ Los procesos de ejecución de pruebas que no sea posible automatizar con el uso de las herramientas **deberán** realizarse de manera “manual”.

✓ **Sólo cuando ya estén validados los distintos módulos de la aplicación:**

⬆ De modo similar al estadio anterior (pruebas de módulos), el responsable del sistema y/o el desarrollador **deberán** generar, configurar y ejecutar las pruebas de integración entre módulos, y las de sistema (funcionales y no funcionales), pero esta vez para todo el sistema.

Datos

⬆ Las tareas de gestión de los datos albergados en base de datos se **deberán** realizar con “SQLDeveloper” si la base de datos es Oracle y con la “Consola SQLServer” si la base de datos es SQLServer.

⬆ Las entregas de scripts de creación de base de datos se **deberán** realizar siempre a través del repositorio de versionado “Subversion” en la rama de “scripts”, y los datos en la rama de “datos”.

⬆ Se **deberá** definir adecuadamente la volumetría de datos del sistema, cumplimentando para ello la hoja de cálculo estándar en EJIE.

💡 En lo referente al código fuente, los contenidos estáticos y scripts de creación de BD, **si** se acordó que el puesto de trabajo estuviese en los locales del proveedor, **entonces**, y tras acuerdo por ambas partes, éste **puede** utilizar cualquier conjunto de herramientas de sus elección siempre y cuando permitan entregar a EJIE los formatos de soporte definidos para los servidores de desarrollo de GV-EJIE.

5.4 Proceso: Implantación y aceptación del sistema

Comprende las actividades y tareas dictadas por la metodología de aplicación ARINbide en la fase de implantación y aceptación del sistema (IAS), y parcialmente, las definidas por Probamet en la fase de *ejecución de pruebas* (EPB). Se indica para cada caso las herramientas de uso en cada actividad y tarea.

Puesto que el proceso se corresponde con las fases de ARINbide y de Probamet ya indicadas, no se detallan las actividades y tareas en las que se descompone, sólo se remarcan ciertos aspectos especialmente importantes que **deberán** o **deberían** cumplirse.

⬆ Se **deberán** ejecutar las actividades y tareas definidas por ARINbide en esta fase, así como las dictadas por Probamet para este ciclo. Éstas se **deberán** realizar considerando el uso de sólo las herramientas, librerías y sistemas especificados en el documento de estándares tecnológicos publicado por la Dirección de informática y telecomunicaciones del GV. Ante cualquier otra nueva necesidad, EJIE **deberá** realizar un estudio detallado determinándose la idoneidad de la propuesta.

⬆ De modo similar al estadio anterior (pruebas de sistema), el usuario final y/o el responsable del sistema **deberán** ejecutar las pruebas de aceptación de usuario que se hayan definido. Con el fin de realizar estas pruebas en un entorno más similar al de producción, estas **deberán** realizarse obligatoriamente en el entorno de pruebas.

⬆ Las incidencias y posibles mejoras detectadas por el usuario se **deberán** registrar y tratar a través de la herramienta “Mantis”, siguiendo siempre el proceso de resolución del ciclo completo de creación de código.



⬆ Como paso previo al despliegue definitivo del sistema en el entorno de producción, y obligatoriamente en el entorno de pruebas, se **deberán** ejecutar el conjunto total de pruebas (análisis estático del código, pruebas unitarias, de integración y de sistema), validando que el resultado de dichas pruebas es el esperado, y que por lo tanto el sistema cumple con los indicadores de calidad establecidos.



Se **deberá** reflejar en el manual de implantación del sistema software:

- Necesidades de visibilidad y acceso entre los componentes del sistema y entre máquinas físicas y otros sistemas de GV o externos (DDFF, Ministerios –red SARA-, etc.), puertos de escucha (si es que se conocen)
- Protocolos de comunicación utilizados (HTTP, HTTPS sobre SSL one-way o two-way, FTP, NFS, IMAP, POP3, etc.)
- Software o librerías homologadas necesarias junto con sus versiones
- Parámetros de configuración de:
 - Pooles (JDBC) de conexión a BD
 - Descriptores de despliegue de los EJBs de la aplicación
 - Colas JMS
 - Etc.
- Relación de scripts de creación de objetos de BD u otros, que se deben ejecutar.
- Configuraciones a realizar en otros sistemas de infraestructura o de plataforma como puede ser PLATEA, correo Exchange, etc.
- Otras necesidades especiales

En definitiva, todo aquello que se considere relevante para la correcta instalación del sistema en los distintos entornos. Se **deberán** elaborar entonces:

- Esquema detallado de la arquitectura física del sistema
- Esquema detallado en el que se refleje su arquitectura lógica



En consonancia con la actividad de gestión del cambio (4.2.2) definida como parte del proceso de administración del proyecto (4.2), se **deberá** elaborar detalladamente el plan de implantación del sistema, que **deberá** contemplar las especificaciones y requisitos definidos en el manual de implantación. Se asegura así la correcta instalación del producto en los entornos de pruebas y de producción.

5.5 Proceso: Aseguramiento de la calidad

Asegurará que los productos obtenidos y los procesos del ciclo de vida del proyecto cumplen con los requerimientos y los planes establecidos.



Se **deberá** aplicar el modelo de aseguramiento de la calidad estándar en GV-EJIE (Modelo SQA).



⬆ El proceso de obtención de los indicadores que realmente determinan el grado de calidad del producto software se **deberá** realizar **en el entorno de pruebas**, puesto que se entiende que este entorno cuenta con una configuración y recursos más similares a los disponibles en el entorno de producción.



Todo proyecto de construcción o mantenimiento de un sistema software **deberá** incluir obligatoriamente el rol de Oficina Técnica de Calidad.



El proceso de aseguramiento de la calidad **deberá** ser ejecutado por la Oficina Técnica de Calidad del proyecto, que **deberá**:

- Asegurar que se ejecutan todas las actividades relacionadas con el aseguramiento de la calidad

- del producto software definidas en el modelo SQA
- Verificar el cumplimiento de las metodologías de aplicación: ARINbide y Probamet, y por ende, que los entregables documentales que estas establecen son correctos y suficientes.
- Verificar que se superan los umbrales establecidos por los indicadores de calidad.

⬆ El Jefe de proyecto de EJIE, como responsable último del proyecto, **deberá** asegurar que el sistema software construido cumple con los estándares de calidad definidos.

⬆ El Modelo SQA se fundamenta en un calendario de entregas **versionadas** del producto (documentación y ejecutables) y en el **plan SQA** que **deberá** ser elaborado y acordado previamente entre los equipos de desarrollo y de pruebas, y la oficina técnica de calidad.

Este proceso se descompone en las actividades descritas a continuación.

5.5.1. Actividad: Verificación documental

En el tiempo, se ejecutará en paralelo a cada uno de los proceso de desarrollo y pruebas del sistema software.

En función del NAC asignado al proyecto se habrán determinado ya:

- Los controles de calidad (SQA) a realizar durante el desarrollo del sistema.
- Las tipologías de pruebas a ejecutar.
- Los indicadores estándar y sus umbrales permitidos

⬆ En las entregas de la documentación asociada al proyecto se **deberá** asegurar que se cumplen los controles de calidad (SQA) definidos, garantizando así el cumplimiento de la metodología de desarrollo, ARINbide, y de pruebas, Probamet, así como la calidad de dicha documentación:

- Verificación documentación de análisis
- Verificación del plan de pruebas
- Verificación documentación de diseño
- Verificación de la especificación de casos de prueba
- Verificación documentación de construcción
- Verificación documentación de implantación

⬆ Los resultados de la verificación realizada **deberán** recogerse en el “portal SQA” de EJIE, formando parte entonces de los indicadores de calidad calculados para el sistema software.

⬆ El analista responsable del proyecto de EJIE **deberá** asegurar que se cumplen los controles de calidad (SQA) definidos.

5.5.2. Actividad: Auditorías SQA

En el tiempo, se ejecutará en paralelo a cada uno de los proceso de desarrollo y pruebas del sistema software.

⬆ Al finalizar cada una de las fases del desarrollo y pruebas del sistema software indicadas por ARINbide (ASI; DSI, CSI e IAS) y por Probamet se **deberá** ejecutar la auditoría de fin de fase, obteniendo así el grado de calidad del sistema según los indicadores recogidos:

- Indicadores de pruebas
 - Análisis estático de código

- Pruebas unitarias
- Pruebas de integración
- Pruebas de sistema
 - Pruebas basadas en requisitos (tanto funcionales como no funcionales)
 - Rendimiento (prestaciones):
 - ✓ Cumplimiento global indicadores prestaciones
 - ✓ Indicadores servidor web
 - ✓ Indicadores servidor de aplicaciones
 - ✓ Indicadores servidor bbdd
 - Seguridad grado cumplimiento owasp top 10
 - Usabilidad grado cumplimiento evaluación heurística
 - Accesibilidad:
- Indicadores de calidad SQA
 - Indicadores de Fin de Fase



⬆ Se **deberá** prestar especial atención a los indicadores de rendimiento obtenidos (uso de la memoria del servidor, de la CPU, tiempo de respuesta de las peticiones http, parámetros de configuración de pools de conexión a base de datos, descriptores de despliegue de EJBS, colas JMS, etc.) los cuales normalmente tienden a descuidarse.

⬆ En el entorno de pruebas, haciendo uso de la herramienta de pruebas de carga “LoadRunner” se **debería** asegurar que el sistema software construido cumple con las especificaciones de rendimiento definidas. Pero **si** se entiende que el sistema debe soportar una gran carga de trabajo, bien por el tipo de proceso que resuelve, bien por el número de usuarios concurrentes o por cualquier otra circunstancia, **entonces** las pruebas de carga ⬆ **deberán** realizarse de manera obligatoria.

- ⬆ Los resultados obtenidos **deberán** recogerse en el portal SQA de EJIE, formado parte entonces de los indicadores de calidad calculados para el sistema software.
- ⬆ Las incidencias y posibles mejoras detectadas se **deberán** registrar y tratar a través de la herramienta “Mantis”. Estas **deberán** ser solventadas en los PCs de desarrollo, nunca en el servidor y siguiendo siempre el proceso de resolución del ciclo completo del código.
- ⬆ Ante cualquier incidente externo al sistema software, se **deberá** seguir el procedimiento de gestión de incidencias definido para la organización; SCP-32.
- ⬆ El analista responsable del proyecto de EJIE **deberá** asegurar que se cumplen los controles de calidad (SQA) definidos, garantizando así el cumplimiento de los indicadores de calidad establecidos y por tanto, con los requisitos del sistema.
- ⬆ Durante la fase de construcción del sistema (CSI) en los PCs locales se **deberá** asegurar que el código construido cumple con las reglas de calidad de código estático definidas por las herramientas estandarizadas: “CheckStyle”, “PMD” y “FindBugs”.

Traspaso al entorno de Producción

- ⬆ Una vez que el usuario final acepta el sistema software construido, y siguiendo la normativa de traspasos vigente, se **deberá** realizar la implantación del aplicativo en el entorno de producción.
- ⬆ Como paso previo al entorno de producción se **deberá** ejecutar la auditoría final SQA el cual permitirá consolidar los resultados de las actividades de calidad realizadas en el proyecto y el grado de calidad obtenido

en los productos generados. Esta auditoría establece además conclusiones y recomendaciones para el CAB en la toma de decisiones para el paso a producción del sistema.

↑ El analista responsable del proyecto de EJIE **deberá** crear en “Subversion” una versión estable del sistema software para el entorno de producción.

↑ El sistema software **deberá** ser traspasado desde el entorno de pruebas ajustando las configuraciones óptimas calculadas y especificadas en el manual de implantación



↑ Durante los dos primeros meses en los que el sistema esté operativo en el entorno de producción se **deberá** realizar un seguimiento exhaustivo del mismo asegurando así que éste funciona de manera adecuada. Se **deberá** comprobar especialmente todo aquello que pueda provocar la inestabilidad del sistema y por extensión, de los servidores en los que se encuentren desplegados

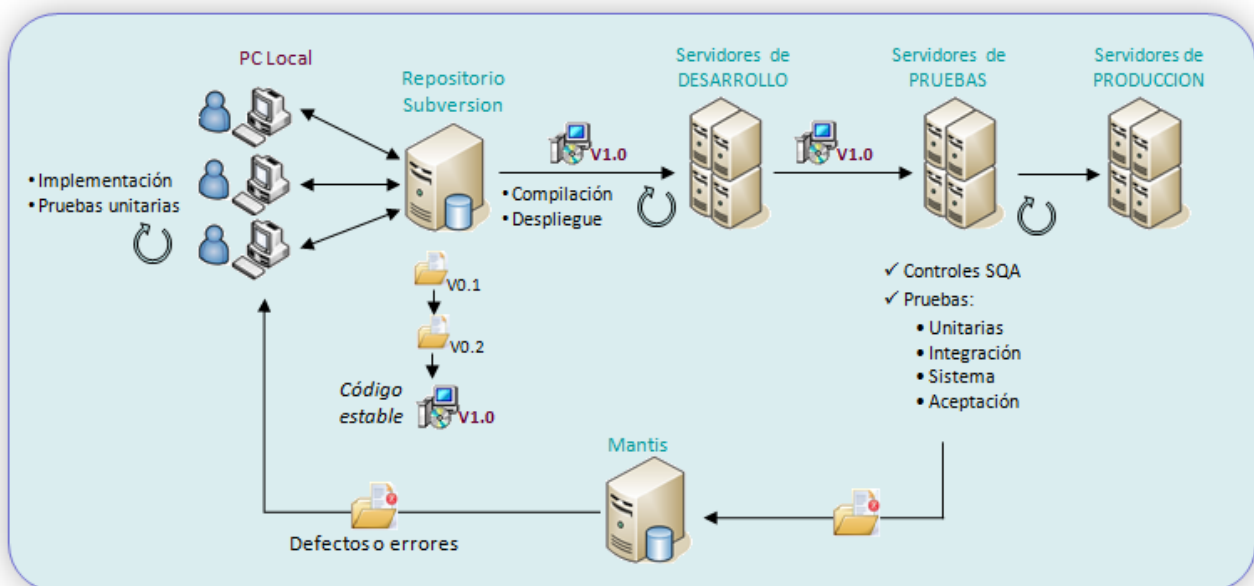
↑ Transcurrido el período de depuración y ajuste final en el entorno de producción, si no se tiene previsto que el sistema software sufra ningún cambio, adaptación o mejora, este **deberá** deshabilitarse del servidor de desarrollo y de pruebas, evitando así un consumo innecesario de recursos.

6 Flujo de procesos

El flujo de ejecución de procesos **deberá** seguir el siguiente esquema:



Y para el proceso de implementación:



Si se acordó la provisión del puesto de trabajo en los locales del proveedor **entonces** éste **deberá** entregar el sistema software debidamente probado y validado previamente en sus instalaciones. No obstante se **deberán** realizar entregas parciales por módulos o fases para poder validar y asegurar de manera temprana y continua la calidad del sistema que se está construyendo. Todas las entregas de contenidos estáticos, código, scripts de base de datos, etc. **deberán** realizarse sobre el repositorio de versiones “Subversion”, y por tanto correctamente etiquetados e identificados, evitando siempre la carga directa por SFTP sobre los servidores de desarrollo.

- ⬆ El equipo de desarrollo **deberá** realizar la implementación del sistema y la codificación de las pruebas unitarias en los PCs locales.
- ⬆ Puesto que el código generado se **deberá** compilar y desplegar en el servidor de desarrollo, este se **deberá** descargar siempre desde el repositorio “Subversion” al disco del servidor, para así poder ser compilado y desplegado. Un vez realizado se podrán ejecutar todas las pruebas que se considere necesarias.
- ⬆ El desarrollador **podrá** dejar en el repositorio de “Subversion” tantas versiones como considere necesario.
- ⬆ Se **deberá** realizar la subida y ejecución de los scripts del modelo de datos en los servidores de back-end.
- ⬆ Se **deberá** realizar la subida del contenido estático mediante la aplicación M26 (aplicación intranet) o a través de Interwoven y Open Deploy (aplicación Internet)
- ⬆ En el entorno de desarrollo, una vez realizadas el conjunto de pruebas básicas que garanticen al menos el funcionamiento elemental del sistema, este se **podrá** promocionar al entorno de pruebas.
- ⬆ Se **deberán** definir y generar los scripts de simulación de escenarios, pudiendo ejecutar así las pruebas de sistema.
- ⬆ En el entorno de pruebas, el equipo de pruebas **deberá** obtener los valores de las métricas marcadas por los indicadores definidos por el modelo de aseguramiento de la calidad (análisis estático del código, pruebas unitarias, pruebas de integración y pruebas de sistema) reflejando dicha información en el “Portal SQA”.
- ⬆ En el entorno de pruebas se **deberían** realizar las pruebas de carga del sistema que se consideren necesarias.
- ⬆ Se **deberá** asegurar también que la documentación asociada al proyecto cumplen los controles de calidad definidos, garantizando así el cumplimiento de la metodología de desarrollo, ARINbide, y de pruebas, Probamet.
- ⬆ La Oficina técnica de calidad **deberá** velar por la ejecución de todas las actividades relacionadas con el aseguramiento de la calidad del producto software a obtener y definidas en el modelo SQA, y comprobar que los resultados obtenidos están dentro de los umbrales aceptados.
- ⬆ El usuario final **deberá** realizar las pruebas del sistema. Los defectos o errores, y por tanto las modificaciones, **deberán** seguir el mismo proceso que el resto de incidencias detectados en las fases de desarrollo del sistema.
- ⬆ En todos los casos, las incidencias o errores **deberán** reflejarse y tramitarse a través de “Mantis”, asignándose así la subsanación al desarrollador correspondiente
- ⬆ Una vez aceptado el sistema software en el entorno de pruebas por parte del usuario final, se **deberá** establecer como versión para producción la que tenga actualmente el entorno de pruebas, es decir, la aceptada por el usuario, siendo por lo tanto la que se traspasará al entorno de producción.
- ⬆ Se **deberá** realizar el traspaso del sistema software al entorno de producción.
- ⬆ Como paso previo a la implantación definitiva en el entorno de producción, la Oficina de evaluación **deberá** informar al CAB al respecto de las conclusiones y recomendaciones obtenidas en lo relativo a los niveles de calidad del sistema software a implantar.

7 Anexo I. Arquitectura

En el ámbito de GV-EJIE, la arquitectura física y por ende la arquitectura lógica de alto nivel es idéntica para cada uno de los entornos existentes, desarrollo, pruebas y producción.

En el entorno de desarrollo se realizará la primera integración de lo desarrollado en PC local a la arquitectura y el entorno tecnológico que sustentará el aplicativo. En el entorno de pruebas se realizarán las pruebas de usuario planteadas como requisitos del aplicativo como paso previo al entorno productivo. El entorno de producción será el entorno final donde la aplicación se ejecutará con los usuarios finales y los datos reales.

El paso entre los diferentes entornos se realizará mediante una sistemática de traspasos.

Plataforma multicapa

La arquitectura física impuesta por el modelo de seguridad de red define un modelo de tres capas. Dicho modelo presenta además una distribución de componentes software, distinguiéndose tres niveles lógicos con cometidos distintos: uno para los servicios de usuario, otro para los servicios de la lógica de negocio (la lógica principal de la aplicación), y otro nivel para los servicios de datos.

Estas capas están físicamente distribuidas de tal manera que únicamente poseen conectividad uno a uno, es decir, la capa de presentación solo tiene acceso a la de negocio, y ésta sólo tendrá acceso a la de datos.

• Presentación

La lógica de presentación es básicamente la encargada de proporcionar la interfaz necesaria para presentar la información y recoger las solicitudes de proceso de los usuarios. Es la parte de funcionalidad de la aplicación que permite al usuario interactuar con la misma.

En la capa de presentación se engloban al menos los contenidos estáticos necesarios para la aplicación (CSSs, javascript, imágenes, etc.) que se desplegarán en el servidor web Apache del contexto y entorno correspondiente.

• Negocio o aplicación

La lógica de negocio es el puente entre el usuario y los servicios de datos. Estos servicios de negocio responden a peticiones del usuario, y de otros servicios de negocio, implementando procedimientos y tratamientos sobre los datos que comprenden las principales funcionalidades y objetivos de la aplicación.

Como medio para la implementación y soporte de esta capa lógica, así como para el funcionamiento de toda esta arquitectura, estará el servidor de aplicaciones Oracle Weblogic Server para tecnología Java, y Microsoft IIS para tecnología .NET. El servidor proporciona los servicios necesarios para gestionar el tratamiento de los componentes de negocio desarrollados así como su control transaccional.

• Datos o Back-end

La capa de datos representa los servicios de datos para la aplicación. Esta capa es la responsable de proporcionar medios para recuperar y actualizar la información y mantener la integridad de datos.

La capa de datos estará sustentada en un sistema de gestión de base de datos.

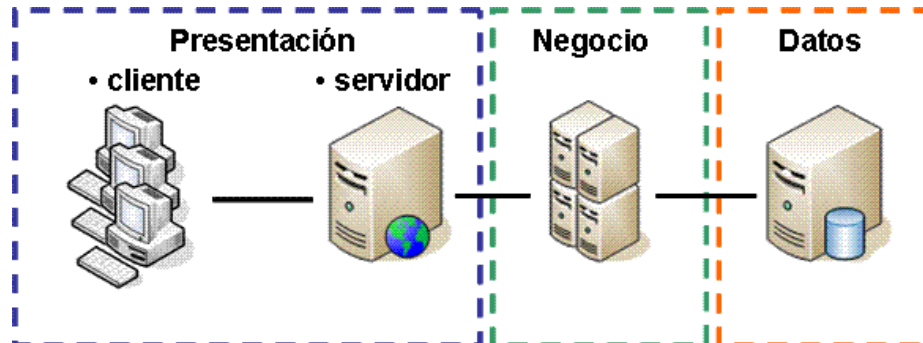
Esta capa sustentará aquellos procesos batch programados para su ejecución con Control-M y gestionados por la aplicación K31/O75

• Integración

Se añade una capa de integración como:

- Intermediario y agregador de funciones de interés general suministrados por distintos sistemas para su exposición y consumo
- Suministrador de servicios con formato de servicio web
- Ámbito de despliegue de procesos de negocio inter-sistemas (orquestración de servicios)
- Proveedor de adaptadores que faciliten el acceso a distintas tecnologías subyacentes
- Propuesta de ámbitos de publicación y consumo de eventos con un único punto de entrada pero con diversas tecnologías de publicación y recolección
- Acelerador de reutilización de sistemas y de interoperabilidad técnica

Todos estos componentes se distribuyen entonces en capas lógicas, pero con disposición en forma de servicios para aquellas funciones de interés general (SOA)



En el proceso de desarrollo del software en el ámbito de GV-EJIE, estos componentes *deberán* de implantarse en el entorno de desarrollo para posteriormente ir traspasándose a los entornos de pruebas y de producción. El paso de desarrollo a pruebas se hará a través de las zonas de traspaso, mientras que para el paso a producción se utilizará lo existente en las zonas de pruebas, es decir, todo paso a producción *deberá* ser transferido y validado previamente en el entorno de pruebas.

8 Anexo II. Contextos de albergue

Los sistemas software o aplicaciones se van a clasificar en los diferentes contextos de albergue según sus características de negocio.

Una aplicación podrá tener módulos en diferentes contextos en función de las necesidades existentes aunque el tratamiento para los procesos de desarrollo e implantación será como si de aplicaciones independientes se tratara.

Estos contextos de albergue se corresponden con los contextos de Internet, de Intranet y de Extranet.

- El contexto de Internet está asociado a aquellas aplicaciones que requieren un acceso de los usuarios desde Internet. Este contexto tiene connotaciones especiales en lo referente a la publicación de los contenidos estáticos y estilos.
- El contexto de Intranet está asociado a aquellas aplicaciones que requieren únicamente un acceso de los usuarios de la intranet de GV. En este contexto estará ubicada la plataforma de Integración que actúa como intermediario y agregador de funciones de interés general suministrados por distintos sistemas para su exposición y consumo.
- El contexto de Extranet está asociado a aquellas aplicaciones que requieren exponer todo o parte de su negocio a un grupo limitado y definido de usuarios apoyándose en otras redes que no se corresponden con la intranet de GV, habitualmente Internet. Actualmente existen dos tipos de extranet; la extranet de “jakinaplan” y la extranet de “Jaso”.

En lo referente a la arquitectura de los diferentes contextos, se trata de una arquitectura replicada para las capas de presentación y de negocio compartiendo la capa de Back-End. Las diferencias básicas entre los diferentes contextos de albergue se limitan, sobre todo, en los aspectos referentes a la seguridad y la conectividad.

9 Anexo III Herramientas de desarrollo

- **AWA:** Ais Web Accessibility es una barra de herramientas que ha sido desarrollada para ayudar a examinar las páginas Web de forma manual para una gran variedad de aspectos de accesibilidad.
- **WDE:** Web Developer es una extensión para el navegador Mozilla Firefox, que consta de una barra de herramientas que ha sido desarrollada para ayudar a examinar las páginas Web de forma manual contemplando una gran variedad de aspectos de accesibilidad.
- **Firebug:** Firebug es una extensión (add-on) de Firefox creada y diseñada especialmente para desarrolladores y programadores web. Es un paquete de utilidades con el que se puede analizar (revisar velocidad de carga, estructura DOM), editar, monitorizar y depurar el código fuente, CSS, HTML y JavaScript de una página web de manera instantánea e "inline".
- **BadBoy:** Permite efectuar el testeo de aplicaciones web, incluyendo una interfaz simple, fácil e intuitiva, mediante los métodos de captura y repetición, siendo una gran ayuda para la prueba de carga de gran alcance, informes detallados, gráficos, etc...
- **Selenium:** Permite efectuar pruebas funcionales de aplicaciones web, mediante la grabación de escenarios funcionales (navegación), facilitando además la automatización de pruebas contra distintas plataformas y navegadores web
- **Enterprise Architect:** Herramienta CASE para el diseño y construcción de sistemas software. Permite definir y gestionar la creación de modelos UML.
- **GIMP:** Editor de imágenes similar a Adobe Photoshop o Corel Photopaint.
- **Mantis:** Aplicación web para el registro y control de bugs.
- **TestLink:** Herramienta de gestión de las pruebas. Permite crear diferentes proyectos de pruebas gestionando para cada uno de ellos el plan de pruebas, los requisitos, casos de prueba y defectos y el análisis de los resultados.
- **Hudson:** Herramienta para la integración y automatización de tareas. Permite automatizar el despliegue y compilación de aplicaciones y la ejecución de pruebas en servidor.
- **Sonar:** Herramienta para validación de calidad del software. Ofrece un Cuadro de mando para visualización de métricas de calidad
- **Oracle SQL Developer:** Herramienta gráfica para el desarrollo en bases de datos Oracle. Permite visualizar objetos de base de datos, ejecutar sentencias SQL y scripts SQL, y editar y depurar sentencias PL/SQL. También permite ejecutar informes ya proporcionados o los creados y salvados por el usuario. SQL Developer simplifica y mejora la productividad a la hora de desarrollar sobre bases de datos Oracle.
- **Subversion:** Sistema open-source escalable de control de versiones, muy potente, usable y flexible, que ha sido diseñado para sustituir a CVS. Para ello trata de proporcionar funcionalidades similares al CVS preservando su filosofía de desarrollo y de dar solución a los principales defectos del CVS.
- **Tortoise SVN:** Cliente gratuito de código abierto para el sistema de control de versiones Subversion. Está desarrollado bajo la Licencia Pública General GNU (GPL).
- **CollabNet Subversion Command-Line Client:** Cliente de código abierto para el sistema de control de versiones Subversion, desarrollado bajo la Licencia Pública General GNU (GPL). Aporta la posibilidad de trabajar desde línea de comandos.
- **XMLSpy:** Altova XMLSpy es el estándar en el entorno de desarrollo para el modelado, edición, depuración y transformación de todas las tecnologías XML. Por eso XMLSpy es ideal para los desarrolladores de J2EE,

.NET, Eclipse y de bases de datos que necesiten estas tecnologías.

- **CheckStyle:** Herramienta de desarrollo que ayuda a los programadores a escribir código Java adscrito a estándares de codificación establecidos, facilitando para ello la automatización del proceso de chequeo del código generado.
- **KeyToolUi:** Herramienta gráfica para la manipulación de formatos criptográficos. (Certificados, keystores, firmas, claves en diversos formatos).
- **Xolido Sign:** Herramienta gráfica para la manipulación de formatos criptográficos. (Certificados, keystores, firmas, claves en diversos formatos).
- **JUnit:** Paquete Java utilizado para automatizar los procesos de prueba. Mediante la creación de Tests, JUnit realizará pruebas unitarias del código que indique el usuario.
- **HP Diagnostics:** herramienta utilizada para monitorizar el rendimiento y consumos de recursos de las aplicaciones en servidor.
- **JMeter:** Apache JMeter es una herramienta de carga diseñada para realizar Pruebas de Rendimiento y Pruebas Funcionales sobre Aplicaciones Web.
- **PMD:** PMD es una herramienta de auditoría y verificación de código estático, que permite detectar errores potenciales en las aplicaciones, en base a un conjunto de reglas parametrizables.
- **Subversive:** El proyecto Subversive se ocupa de facilitar la integración de Subversion para Eclipse. El plugin instalado se integra en el entorno de desarrollo y nos permite trabajar con los repositorios Subversion. Se podrán utilizar los repositorios Subversion casi de la misma manera que se utilizan los repositorios CVS mediante el plugin CVS incluido en la distribución estándar de Eclipse.
- **SoapUI:** Herramienta para testeo de servicios web. Permite la realización de tests funcionales, de carga, simulación de servicios web (mock webservice), integración con herramientas de generación de código, etc. a través de una interfaz gráfica. Está escrito en java, es multiplataforma e integrable en los entornos de desarrollo más comunes (eclipse, netbeans e IntelliJ).
- **QuickRex:** Plug-in para Eclipse cuyo objetivo es facilitar la creación de expresiones regulares.
- **NUnit:** Alternativa a JUnit para .Net. Permite realizar pruebas unitarias para cualquier lenguaje de .Net.
- **AnkhSVN:** Plug-in para Visual Studio para trabajo desde puesto local de desarrollo con el repositorio de versiones Subversion.
- **Eclipse:** Herramienta para el desarrollo de aplicaciones Java y J2EE (IDE de desarrollo).
- **OEPE:** Conjunto gratuito plug-ins que permiten a los desarrolladores de WebLogic trabajar con Java EE y Servicios Web Estándares. Oracle Enterprise Pack para Eclipse permite el desarrollo bases de datos, Java SE, Java EE, Web Services, XML, y Spring.
- **JReport:** Aplicación para la generación de informes para aplicaciones JAVA.
- **JasperReports:** Librería para la generación de informes para aplicaciones JAVA.
- **Maven:** Sistema de gestión de repositorio de librerías de aplicación y utilidades para la descarga y publicación
- **Archiva:** herramienta con interfaz gráfica para gestión de repositorios propios

10 Anexo IV. Sistemas horizontales

- **XLNETS**: Sistema general de autenticación de Gobierno Vasco basado en LDAP.
- **PLATEA**: Plataforma para la e-administración de Gobierno Vasco.
- **Control M**: Aplicación para la planificación y ejecución de procesos batch.
- **K31**: Gestor de procesos batch.
- **O75**: Interfaz web y API para interactuar con el gestor de procesos batch (K31).
- **Pasarela de pagos**: Plataforma que integra la gestión de los pagos en Gobierno Vasco.
- **NORA**: Aplicativo para la gestión actualizada de los datos de localización, incluyendo utilidades geográficas como Visor GIS y Geolocalizador
- **GIS Corporativo**: Plataforma de información geográfica y servicios GIS
- **DOKUSI (Sistema Integral de Gestión Documental)**: Plataforma para la gestión de todos los documentos almacenados electrónicamente por el Gobierno Vasco.
- **SGA (Sistema de Gestión de Archivos)**: Plataforma para la gestión de todos los documentos archivados por el Gobierno Vasco.
- **SMS Corporativo**: Sistema para la gestión y envío de mensajes a móviles, basado en la plataforma Latinia
- **UDA (Utilidades de desarrollo de aplicaciones)**: Conjunto de utilidades, herramientas, librerías, plugins, guías, y recomendaciones funcionales y técnicas que permiten acelerar el proceso de desarrollo de sistemas software con tecnología Java.