Proyecto Final de Carrera
Modalidad: Práctica en Empresa

Desarrollo de un sistema integrado de gestión de seguridad e inscripciones en el Instituto Provincial Autárquico de Vivienda (IPAV)

Carrera: Ingeniería en Sistemas (Plan 2004)

Alumno: CAVALLO, Bruno Sebastián

Legajo: 3671

D.N.I.: 34.124.834

Teléfono: 02302-15507898

E-mail: brunocavallo@hotmail.com

Tutor: Dr. DIVÁN, Mario José

Prefacio

A continuación se presenta el proyecto final de la carrera de Ingeniería en Sistemas (Plan 2004) perteneciente a la Universidad Nacional de La Pampa (UNLPam), el mismo consiste en detallar las actividades desarrolladas en el Instituto Provincial Autárquico de Vivienda (IPAV) en el cual me desempeñé como pasante; la finalidad de las mismas es el relevamiento, análisis y desarrollo de un nuevo sistema de seguridad e inscripciones que se relacionan directamente con la carrera cursada.

El proyecto tiene como modalidad "Práctica en Empresa" y se centra principalmente en las actividades efectuadas sobre el proyecto de inscripciones, el cual brinda la información para la gestión y monitoreo de inscripciones, irregularidades, impugnaciones, sorteo, pre-adjudicaciones, cancelaciones, permutas y renuncias.

El proyecto final de carrera tiene como tutor académico al Doctor Mario José Diván y como tutor por parte del IPAV al Contador Público Nacional Roberto Vassia.

Agradecimientos

En primer lugar quiero agradecer a mi **mamá** por estar presente en cada instante de mi vida, en los buenos y difíciles momentos, dando todo por mí y mis **hermanas** con esfuerzo y dedicación, me ha apoyado siempre para que yo pueda crecer y elegir mis propios caminos acompañándome durante toda la carrera. A mis **hermanas**, agradecerles por los momentos compartidos desde chicos, por estar siempre cuando las necesito y por interesarse en como avanzaba día a día en el transcurso de la carrera. A mi **abuela**, agradecerle por cuidarnos siempre, es agradable compartir diferentes momentos con ella, desde una comida los domingos, mirando un partido de futbol o las diferentes charlas con un mate de por medio. A mis **familiares**, que estando o no en Pico, sé que puedo contar con ellos siempre.

Gracias a mi **director**, por las oportunidades que me ha dado en esta pasantía, compartiendo sus experiencias y conocimientos durante el trascurso de la misma, y principalmente por el tiempo dedicado para resolver mis consultas y dudas del proyecto final. A los **miembros del tribunal** por el tiempo empleado en las correcciones.

A mis **amigos** de la vida y los que he conocido en la universidad, les agradezco por los momentos compartidos, salidas, horas de estudios, charlas, juntadas y por estar siempre cuando los necesito.

A mis **compañeros del Grupo de trabajo** del IPAV por la ayuda brindada cuando tenía algún problema al momento del desarrollo del sistema y por las sugerencias aportadas; por las charlas, momentos y mates compartidos en el trabajo, tanto con ellos como con el **personal del IPAV**.

A los **profesores** y **ayudantes de cátedra** por brindarme sus enseñanzas a lo largo de la carrera y al **personal no docente** por aconsejarme y/o ayudarme en cualquier problema que me ha surgido en la facultad.

Bruno

Índice

Lista de Figuras

Lista de Tablas

Resumen

Los sistemas actuales del Instituto Provincial Autárquico de Vivienda (IPAV) se encuentran desarrollados en base a Fox Pro 2.5 y Visual Basic 6.0, con el riesgo de constituirse en compartimientos estancos, es decir los mismos no son interoperables. Ello implica que el organismo posee una necesidad inminente de actualizar no solo tecnológicamente sino funcionalmente su operatoria, con vistas a mejorar la calidad de servicios brindados al ciudadano.

Luego de analizar los diferentes sistemas actuales del IPAV y haber realizado entrevistas a los empleados del organismo, pudo constatarse la necesidad de actualización de sus sistemas a partir de la perspectiva del usuario. De esta manera, y gracias a que el organismo posee los procesos de la Gerencia de Planificación y Adjudicación formalizados, es posible ahora avanzar sobre la automatización de los mismos.

En tal sentido, el IPAV comienza el desarrollo de un nuevo sistema para agilizar la gestión de la Gerencia de Planificación y Adjudicación, sobre la base de un nuevo sistema troncal de seguridad.

El nuevo módulo de seguridad troncal sobre el que se incorporará el Sistema de Gestión de la Gerencia de Planificación y Adjudicaciones, plantea una capa homogénea e integrada, que permite al usuario autenticarse ante un mismo sistema, y a partir de ello, determinar las autorizaciones disponibles en base a su perfil a lo largo de todos los sistemas del organismo. El objetivo de esta Práctica en Empresa fue integrar activamente una célula de desarrollo de software real en el marco de la implementación del sistema de seguridad e inscripciones, participando en cada una de las fases y etapas que conforman el ciclo de vida de un software.

La metodología de gestión utilizada para el desarrollo de los diferentes sistemas fue SCRUM, cuyo principal aspecto es la coordinación y sincronización de objetivos en ciclos cortos de trabajo. Otro aspecto importante de SCRUM es la determinación del orden de prioridades de las funcionalidades del sistema a desarrollar en base a la interacción con el cliente, como así también la entrega de revisiones parciales del software al usuario para minimizar riesgos y obtener su retroalimentación en forma continua.

Introducción

A continuación se detallan las actividades desarrolladas, durante el desarrollo del Proyecto Final de Carrera, en el Instituto Provincial Autárquico de Vivienda (IPAV) en el cual me desempeñé como pasante.

El desarrollo de este informe está organizado con la siguiente estructura:

• **Capítulo 1:** Inicio del Sistema de Seguridad e Inscripciones. Se introduce y fundamenta la necesidad de actualización de los sistemas del IPAV.

• **Capítulo 2:** Introducción a los Servicios RESTFul utilizados durante la pasantía.

• **Capítulo 3:** Etapas de desarrollo del sistema. Síntesis de las fases del proyecto que se realizaron con antelación al comienzo de la práctica en empresa.

• **Capítulo 4:** Bitácora de las actividades desarrolladas durante la práctica. Se enumeran y especifican las actividades efectuadas durante la práctica en empresa con una periodicidad semanal.

• **Capítulo 5:** Aplicaciones y Tecnologías empleadas durante la pasantía. Se detallan los distintos componentes tecnológicos que han sido utilizados durante la práctica en empresa del IPAV.

• **Capítulo 6:** Nociones aprendidas durante la práctica. Se sintetiza lo asimilado durante la realización de la pasantía que complementa los conocimientos ya adquiridos en asignaturas curriculares de la carrera de Ingeniería en Sistemas.

Desarrollo

Capítulo 1: Inicio del Sistema de seguridad e inscripciones

"EL Instituto Provincial Autárquico de Vivienda (IPAV) en sus inicios, comenzó siendo una Dirección dependiente del Ministerio de Bienestar Social, donde un grupo muy reducido de empleados, trabajaba para construir viviendas en todos los rincones de La Pampa. El 18 de agosto de 1977 se creó el INSTITUTO PROVINCIAL AUTÁRQUICO DE VIVIENDA con el fin de satisfacer la demanda habitacional de las familias de escasos recursos, cuyos ingresos no le permitían financiar con recursos propios el costo de las mismas" [1].

A la fecha, si bien este organismo tiene los procesos de adjudicaciones de viviendas, irregularidades, permutas y demás actividades, pertenecientes al área de planificación y adjudicación, formalizados, aún resta su automatización.

Por otro lado, los sistemas actuales no son interoperables y se manejan como compartimientos estancos, no permitiendo la integración entre los distintos componentes que conforman dichos sistemas, lo que dificulta el intercambio de datos de manera automatizada.

A partir de este contexto, surge la necesidad de desarrollar un nuevo sistema integrado que cumpla con la posibilidad de administrar y monitorear la actividad del IPAV mediante un único sistema.

De este modo, y dada la misión del organismo, se prioriza inicialmente la capa de seguridad y la automatización de los procesos de la Gerencia de Planificación y Adjudicaciones. Así, el nuevo sistema de inscripciones, que se encuentra en proceso de desarrollo, está integrado por los siguientes ítems:

- **Un módulo de seguridad común para todas las aplicaciones del IPAV:** Se desarrolla una estructura de seguridad común a todas las aplicaciones que se están desplegando [2]. Dicho módulo contempla una arquitectura orientada al servicio (Service-Oriented Architecture - SOA) [3] como estrategia de integración de las aplicaciones futuras del Instituto, y con respecto a otros organismos del Estado Municipal, Provincial y Nacional, manteniendo en forma permanente una política de seguridad uniforme sobre los distintos actores que pretendan usar las aplicaciones del organismo.

- **Automatización de los Procesos de la Gerencia de Planificación y Adjudicación:** A partir de la reingeniería de procesos realizada sobre la Gerencia de Planificación y Adjudicación, se desarrolla un software que automatiza los procesos de dicha Gerencia. El sistema brinda la información para la gestión y monitoreo de inscripciones, irregularidades, impugnaciones, sorteo, pre-adjudicaciones, cancelaciones, permutas y renuncias. El límite del software se acota desde la registración de las inscripciones, hasta la emisión de los listados pertinentes vinculados con las adjudicaciones.

Capítulo 2: Introducción a Servicios RESTFul

En los sistemas que se están desarrollando a la fecha en el IPAV, se emplean servicios Web del tipo RESTFul. REST (Representational State Transfer) define un estilo de arquitectura para desarrollar servicios Web haciendo foco en los recursos del sistema, incluyendo como se accede al estado de dichos recursos.

Una de las características claves de los servicios Web REST es el uso explícito de los métodos HTTP y adicionalmente, permiten brindar la respuesta mediante diferentes formatos, tales como XML, JSON, HTML, entre otros.

Principios de diseño de los servicios RESTFul

Una implementación concreta de un servicio Web REST sigue cuatro principios de diseño fundamentales [4]:

- Utiliza los métodos HTTP de manera explícita,
- No mantiene estado,
- Expone URIs con forma de directorios,
- Transfiere XML, JSON (JavaScript Object Notation), o ambos.

A continuación se explican cada uno de estos principios:

REST utiliza los métodos HTTP de manera explícita

Una de las características claves de los servicios Web REST es el uso explícito de los métodos HTTP. Así, REST hace que los desarrolladores usen los métodos HTTP de manera que resulte consistente con la definición del protocolo. Este principio de diseño básico establece una asociación uno-a-uno entre las operaciones de crear, leer, actualizar y borrar un recurso del servidor y los métodos HTTP. De acuerdo a esta asociación:

- ✓ se usa POST para crear un recurso en el servidor,
- ✓ se usa GET para obtener un recurso,
- ✓ se usa PUT para cambiar el estado de un recurso o actualizarlo,
- ✓ se usa DELETE para eliminar un recurso.

REST no mantiene estado

Son servicios Web que no mantienen estado asociado al cliente. Cada petición que se realiza a ellos es completamente independiente de la siguiente. Todas las llamadas al mismo servicio serán idénticas.

Una petición completa e independiente hace que el servidor no tenga que recuperar ninguna información de contexto o estado al procesar la petición. Una aplicación o cliente de servicio Web REST debe incluir dentro del encabezado y del cuerpo HTTP de la petición todos los parámetros, contexto y datos que necesita el servidor para generar la respuesta. De esta manera, el no mantener estado mejora el rendimiento de los servicios Web y simplifica el diseño e implementación de los componentes del servidor, ya que la ausencia de estado en el servidor elimina la necesidad de sincronizar los datos de la sesión con una aplicación externa. En un servicio Web REST el servidor es responsable de generar las respuestas y proveer una interfaz que le permita al cliente mantener el estado de la aplicación por su cuenta.

REST expone URIs con forma de directorios

La URI (Uniform Resource Identifier) de los servicios Web REST debe ser intuitiva. Se debe pensar en la URI como una interfaz auto-documentada que necesita de muy poca explicación o referencia, para que un desarrollador pueda comprender a lo que apunta, y a los recursos derivados relacionados.

Una forma de lograr este nivel de usabilidad es definir URIs con una estructura al estilo de directorios. Este tipo de URIs es jerárquica, con una única ruta raíz, y va abriendo ramas a través de las sub rutas para exponer las áreas principales del servicio. De acuerdo a esta definición, una URI no es solamente una cadena de caracteres delimitada por barras, sino más bien un conjunto de nodos en forma de árbol jerárquico, donde cada nodo representa un recurso del servidor.

REST transfiere XML, JSON, o ambos

La representación de un recurso en general refleja el estado actual del mismo y sus atributos al momento en que el cliente de la aplicación realiza la petición.

La última restricción al momento de diseñar un servicio Web REST tiene que ver con el formato de los datos que la aplicación y el servicio intercambian en las peticiones/respuesta. El formato puede ser XML, JSON o ambos.

Otras primicias que deben cumplir los servicios Web RESTful:

A continuación se detallan otros principios que deben cumplir los servicios RESTful [5].

Cliente/Servidor: Los servicios Web son cliente/servidor y definen una interface de comunicación entre ambos, separando completamente las responsabilidades entre las partes.

Cache: El contenido de los servicios Web REST se puede almacenar temporalmente, de tal forma que una vez realizada la primera petición al servicio, el resto de las peticiones pueden apoyarse en dicho almacenamiento, si fuera necesario.

Arquitectura en Capas: Todos los servicios REST están orientados hacia la escalabilidad. Así, un cliente REST no es capaz de distinguir si está realizando una petición directamente al servidor, si la respuesta proviene desde un sistema de almacenamiento intermedio, o bien, si existe un balanceador que se encarga de redirigir la petición a otro servidor.

Capítulo 3: Etapas de desarrollo del sistema

A continuación se describen cada una de las etapas que se efectuaron durante la pasantía, algunas de ellas previas al comienzo de ésta práctica. Las mismas estuvieron a cargo de los diferentes integrantes del equipo de trabajo, conformado por seis personas, donde cada uno realizaba diferentes tareas en forma coordinada para lograr un objetivo final [6], es decir, el desarrollo del sistema de seguridad e inscripciones. Algunas de estas tareas se efectuaron en la ciudad de Santa Rosa y otras en la ciudad de General Pico.

El sistema de inscripciones mencionado abarca todos los procesos de la Gerencia de Planificación y Adjudicación, es decir la gestión y monitoreo de inscripciones, irregularidades, impugnaciones, sorteo, pre-adjudicaciones, cancelaciones, permutas y renuncias.

Etapa 1: Realización de entrevistas y modelado de procesos

Antes del desarrollo del sistema de seguridad e inscripciones del IPAV se comienza con una serie de entrevistas a los diferentes empleados de la Gerencia de Planificación y Adjudicación; el objetivo de las mismas era adquirir conocimientos de las diferentes funciones que desarrollan, las actividades y procesos de la gerencia, los documentos con los que trabajan, las acciones que se realizan cuando se origina un suceso con una determinada vivienda, o cuando existe alguna irregularidad con la misma, entre otras cuestiones.

A partir de la información obtenida mediante las entrevistas, se documenta cada uno de los procesos mediante SPEM (Software & Systems Process Engineering Metamodel), para luego poder definir los diferentes componentes y procesos para el desarrollo del software.

SPEM se define como: *"un metamodelo de ingeniería de procesos, el cual proporciona los conceptos necesarios (marco conceptual) para modelar, documentar, presentar, gestionar, intercambiar y promulgar métodos de desarrollo y procesos. Una implementación de éste modelo está dirigida a los ingenieros, directores y gerentes de programa y proyecto quiénes son responsables de mantener e implementar procesos para sus organizaciones de desarrollo o proyectos individuales"* [7] y es utilizado: *"para definir los procesos de desarrollo de sistemas y software y sus componentes. El alcance de SPEM es limitado deliberadamente a los elementos mínimos necesarios para definir un proceso de desarrollo de software y sistemas, sin la adición de características específicas de dominios o estándares particulares de desarrollo (por ejemplo, gestión de proyectos). El objetivo entonces es acomodar una amplia gama de métodos de desarrollo y procesos de diferentes estilos; sin embargo,*

el foco de SPEM es el desarrollo de proyectos, no pretende ser un lenguaje de modelado de procesos genéricos, ni proporciona sus propios conceptos de modelado, sino que define la capacidad para que el implementador pueda elegir el enfoque de modelado que mejor se adapte a sus necesidades. También proporciona estructuras específicas para mejorar dichos modelos de procesos genéricos de desarrollo. En otras palabras, SPEM se centra en proporcionar información adicional para definir los procesos de modelado, a la brindada por UML (Unified Modeling Language) y así poder describir un proceso de desarrollo real." [7].

Etapa 2: Generación del módulo de seguridad del sistema

El desarrollo del sistema de seguridad constituye la primera etapa del ciclo de automatización, junto con su correspondiente base de datos. Dicho sistema es común a todas las aplicaciones del IPAV y puede integrarse en forma transparente con los futuros sistemas que se realicen en el organismo (Ver Figura 1). Adicionalmente, permite un núcleo de control unificado de acceso regulado a la información, que facilita la interoperabilidad de los datos del organismo en relación con otras instituciones.

Figura 1: Módulo de Seguridad - IPAV

El nuevo sistema de seguridad permite la administración de los diferentes usuarios y los permisos pertinentes de éstos sobre las diferentes funcionalidades de los sistemas del organismo.

Dentro de las funcionalidades contempladas en el módulo de seguridad, pueden enunciarse:

1. **Registrar nuevos usuarios:** Permite ingresar y guardar los datos de los usuarios; dichos usuarios corresponden a las personas que usarán los diferentes sistemas que se están desarrollando.
   Si un usuario desea usar una funcionalidad del sistema de inscripciones entonces se verifica, mediante el sistema de seguridad integrado, que ese usuario tenga los permisos necesarios para ejecutar dicha funcionalidad.

2. **Listar usuarios:** Permite emitir un listado con los diferentes usuarios ingresados al sistema, activar y desactivar los mismos. Un usuario desactivado no tiene el permiso para el ingreso a los diferentes sistemas del IPAV.

   Debido a que queda registrado qué operación realiza cada persona en los sistemas, no es posible eliminar los usuarios ya ingresados a la base de datos de seguridad (solamente desactivarlos).

3. **Buscar usuario:** Permite buscar un usuario en particular por perfil, rol, acción o tarea asociado/a a éste, para luego efectuar alguna actividad con el mismo: modificar, asociar/desasociar permisos, activar/desactivar, entre otras (Ver Figura 2).

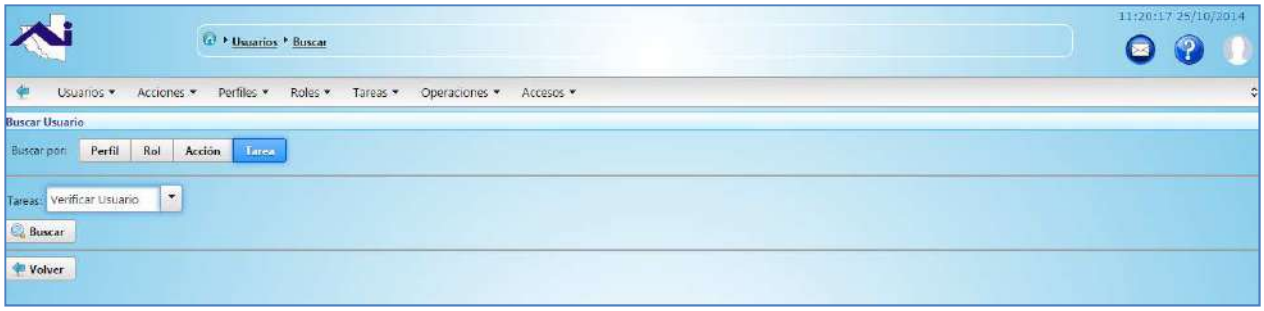


Figura 2: Interface Buscar Usuario. Módulo de Seguridad - IPAV

4. **Modificar usuarios:** Permite cambiar la información correspondiente a cada usuario (además de los datos personales, se puede cambiar su contraseña).

5. **Visualizar y Registrar permisos de usuarios:** Permite visualizar los permisos de un usuario. Un permiso se determina de acuerdo a los perfiles, roles, acciones o tareas que se asocian al mismo. Además se pueden agregar y quitar permisos de un usuario en particular.

6. **Registrar acciones:** Permite registrar las acciones que un usuario en particular puede realizar en los diferentes sistemas.

7. **Visualizar acciones:** Permite visualizar las diferentes acciones registradas y los usuarios, roles y perfiles con permisos para su ejecución.

8. **Asignar cotas:** El sistema permite asignar cota física a cada acción (direcciones IP de computadoras donde se puede realizar) y/o temporal (días y horas de las semana en que se puede realizar) de cada perfil, usuario y/o rol.

9. **Registrar nuevos perfiles de usuarios**: Permite ingresar el perfil de un usuario.

10. **Visualizar perfiles:** Permite visualizar los diferentes perfiles de usuarios registrados y la eliminación de los mismos. Además de la definición de las acciones, tareas y/o usuarios asociados al perfil.

11. **Modificar perfiles de usuarios:** Permite la modificación de cada perfil de usuario.

12. **Registrar nuevos roles de usuarios**: Permite ingresar un nuevo rol de usuario.

13. **Visualizar roles:** Permite visualizar los diferentes roles registrados y la eliminación de los mismos. Además de la definición de las acciones, tareas y/o usuarios asociados al rol.

14. **Modificar roles:** Permite la modificación de cada rol.

15. **Registrar nuevas tareas**: Permite ingresar una tarea que se puede realizar en un sistema en particular.

16. **Visualizar tareas:** Permite visualizar las diferentes tareas registradas y la eliminación de las mismas. Además de la definición de las acciones y/o usuarios asociados a la tarea mencionada.

17. **Modificar tarea:** Permite la modificación de cada tarea registrada en el sistema.

18. **Visualizar operaciones:** El sistema permite visualizar cada operación realizada en los sistemas.

    Cuando un usuario realiza una determina operación en los distintos sistemas que se están desarrollando en el IPAV, ésta queda registrada en el sistema de seguridad, por lo tanto en este apartado se puede visualizar cada operación efectuada y que usuario la realizó.

19. **Definir y visualizar los accesos:** Permite registrar y visualizar el acceso al sistema de cada usuario, perfil o rol. Los accesos se definen de manera temporal, es decir mediante los días de la semana e intervalo de horarios en que se está permitido el ingreso al sistema.

De este modo, el sistema administra los distintos perfiles de los usuarios, los roles, las acciones y las tareas que el usuario puede realizar en el sistema y los accesos permitidos; y también visualizar las diferentes operaciones ejecutadas en los distintos sistemas.

Figura 3: Pantalla de login de los sistemas desarrollados por el IPAV, Incluido el de Inscripciones.

El sistema al igual que todas las aplicaciones que está implementando el IPAV presenta una pantalla de login, y en base al perfil del usuario autenticado se determinan los permisos que posee, las acciones que puede realizar y la información que tiene permitido visualizar (Ver Figura 3).

Etapa 3: Relevamiento / Gestión de requerimientos:

En base a los procesos SPEM documentados para la Gerencia de Planificación y Adjudicación, se realiza un estudio para poder determinar sobre qué aspectos se debe profundizar en el desarrollo de la aplicación e identificar la funcionalidad esperada del sistema de inscripciones. De este modo, se participa del relevamiento sobre:

a) El área donde se utiliza el sistema (en éste caso la Gerencia de Planificación y Adjudicación).

b) Los usuarios finales de la aplicación: los gerentes y empleados de dicha gerencia.

c) La interacción con otros sistemas: provinciales y nacionales. Debido a que se ha desarrollado un módulo de seguridad orientado al servicio, éste sistema que se está programando se podrá integrar a otros sistemas que el IPAV desarrolle en el futuro.

d) La funcionalidad esperada: la gestión y monitoreo de inscripciones, irregularidades, impugnaciones, sorteo y las pre-adjudicaciones, entre otras funcionalidades.

Identificación de los requerimientos no funcionales:

Se identificaron los requerimientos no funcionales, los mismos hacen relación a las características del sistema que aplican de manera general como un todo, más que a rasgos particulares del mismo. Estos requerimientos son adicionales a los requerimientos funcionales que debe cumplir el sistema, y corresponden a aspectos tales como la disponibilidad, mantenibilidad, flexibilidad, seguridad, facilidad de uso, etc.

Los requerimientos no funcionales detectados se mencionan a continuación:

- El sistema debe dar respuesta al acceso de todos los usuarios con tiempo de respuesta aceptable. La información almacenada podrá ser consultada y actualizada permanente y simultáneamente, sin que se afecte el tiempo de respuesta.
- Estar disponible la mayor cantidad posible de tiempo, 100% o muy cercano del horario hábil laboral.
- Ser amigable, de fácil utilización y entrenamiento por parte de los usuarios de IPAV. Típicamente usuarios acostumbrados a entornos de programas Fox Pro 2.5 sobre DOS.
- Ser escalable. Es decir estar preparado para permitir en el futuro el desarrollo de nuevas funcionalidades, modificar o eliminar las mismas después de su construcción y puesta en marcha inicial.
- Presentar mensajes de error que permitan al usuario identificar el tipo de error y comunicarse con el administrador del sistema.
- Estar debidamente documentados el código fuente, los manuales de administración y de usuario, y cada uno de los componentes de software que forman parte de la solución propuesta.
- Contar con una interfaz de administración que incluya al menos la posibilidad de: administración de usuarios, roles, perfiles, control de acceso, y autenticación unificada. Cada una de estas secciones deberá ofrecer todas las opciones de gestión disponibles para cada uno (módulo de seguridad).

- Permitir la trazabilidad e identificación de errores ante funcionalidades actuales y-o futuras. Permitir en el futuro el mantenimiento y seguimiento con respecto a los posibles errores que se puedan presentar durante la operación del sistema.
- restringir el acceso mediante el uso de claves asignadas a cada uno de los usuarios. Sólo podrán ingresar al Sistema las personas que estén registradas, estos usuarios serán clasificados en varios tipos de roles con acceso a las opciones de trabajo definidas para cada rol (módulo de seguridad).
- Implementar un control de acceso que permita asignar los perfiles para cada uno de los roles identificados (módulo de seguridad).
- Rechazar accesos o modificaciones indebidos (no autorizados) a la información y proveer los servicios requeridos por los usuarios legítimos del sistema (módulo de seguridad).
- Contar con mecanismos que permitan el registro de actividades con identificación de los usuarios que las realizaron (módulo de seguridad).
- Validar automáticamente la información contenida en los formularios de ingreso.
- Implementar un sistema basado en la Web y realizar toda administración desde un navegador.
- Operar la solución de manera independiente del navegador que se utilice.
- Proveer mecanismos para generar backup´s periódicamente de la información que se mantiene en el sistema.
- Ser interoperable e integrable. Es decir ser capaz de interactuar con otros sistemas propios del organismo, provinciales y nacionales.

Identificación de los requerimientos funcionales:

Como se mencionó anteriormente se identificaron los requerimientos funcionales del sistema de inscripciones. Entre las funcionalidades identificadas se pueden mencionar las siguientes:

- Dar de alta a una persona para poder asignarle un turno para la posterior inscripción.
- Asignar turno a una persona para que luego, en dicho turno, pueda inscribirse en el sistema y aspirar a una vivienda.
- Informar los turnos asignados.
- Gestionar el cambio de titularidad de un grupo familiar.
- Gestionar la cancelación de una vivienda.
- Gestionar el comodato de una vivienda.
- Gestionar la permuta de viviendas entre adjudicados.
- Gestionar la renuncia a una vivienda.

- Visualizar adjudicaciones de viviendas.
- Gestionar las denuncias de vivienda.
- Inscribir a una persona que aspira a una vivienda social en el turno correspondiente.
- Gestionar las impugnaciones de las preadjudicaciones de viviendas.
- Emitir listados y certificados correspondientes de adjudicación.
- Gestionar los distintos informes con los que trabaja el IPAV: cancelación, asesoría legal y social, entre otros.
- Otras funcionalidades.

Para cada funcionalidad expresada se efectuó el correspondiente diagrama de casos de usos y se identificaron los actores asociados [8].

Para mantener un orden en el diagrama realizado, los diferentes casos de usos fueron separados por paquetes. Cada paquete mantiene un conjunto de casos de usos coordinados entre si y contiene un código representativo.

A continuación se mencionan los diferentes paquetes:

- **Adjudicaciones**: Se lleva a cabo la gestión de todo lo asociado con el cálculo de puntaje de los grupos familiares para acceder a una vivienda, el sorteo de las mismas, la preadjudicación de las viviendas a otorgar. Código del paquete: ADJ.
- **Inscripciones**: Se gestiona todo lo relacionado con el alta de la persona en el sistema de inscripciones y en caso de ya estar inscripto, actualizar la información asociada a dicha persona, como también dar de baja a una persona de la ficha de inscripción correspondiente. Código del paquete: INS.
- **Impugnaciones:** Se llevan a cabo la recepción y verificación de las denuncias recibidas sobre aquellas personas preadjudicadas de viviendas. Código del paquete: IMP.
- **Irregularidades:** Se gestiona todo lo referente con las denuncias de las personas a las familias cuya vivienda les fue adjudicada y no viven en la misma. Se llevan a cabo verificaciones de vivienda para corroborar la denuncia recibida. Código del paquete: IRG.
- **Operatorias de mesa:** En este paquete se concentran todas las operatorias de mesa, asociadas con el cambio de titular, la cancelación, comodato, permuta, renuncia, y demás procesos sobre las viviendas adjudicadas. Código del paquete: OPM.
- **Turnos:** Se lleva a cabo la gestión de la solicitud de turnos para que la persona pueda luego realizar la inscripción correspondiente. Código del paquete: TUR.

Para cada caso de uso se empleó una plantilla donde se incorpora información del mismo (Ver Tabla 1)

Actividad	**Código**	**Nombre**
Objetivo		
Pre-condiciones		
Especificación de la actividad		
Post-condiciones		
Requerimientos no funcionales asociados		
Observaciones		

Tabla 1: Plantilla asociada con el caso de uso correspondiente.

A continuación, se describe cada uno de los componentes que conforman la plantilla de casos de usos.

Actividad: Determina el código y nombre de caso de uso.

Código: Un código único que identifica al caso de uso, dicho código contiene la siguiente forma *"código del paquete"-"número del caso de uso dentro del paquete comenzando desde 0001"*. Ejemplo: INS-0001. Explicación: Caso de uso 0001 correspondiente al paquete de inscripciones.

Nombre: Es una descripción corta que da a entender el objetivo principal del caso de uso.

Objetivo: Una descripción más extensa que el nombre que indica lo que esperan obtener los actores con la realización de este caso de uso.

Pre-condiciones: Es el estado en que se debe encontrar el sistema para que el caso de uso pueda realizarse.

Especificación de la actividad: Describir como una secuencia de pasos la interacción entre los actores y el sistema para cumplir el objetivo.

Post-condiciones: Especifica lo que debe suceder luego de que las actividades del caso de uso se efectuaron.

Requerimientos no funcionales asociados: Especifica los requerimientos no funcionales asociados al caso de uso. Los requerimientos no funcionales hacen relación a las características del sistema que aplican de manera general como un todo, más que a rasgos particulares del mismo. Estos requerimientos son adicionales a los requerimientos funcionales que debe cumplir el sistema, y corresponden a aspectos tales como la disponibilidad, mantenibilidad, flexibilidad, seguridad, facilidad de uso, etc.

Observaciones: Problemas a resolver o cuestiones que son importantes mencionar porque están directamente relacionadas con alguno de los escenarios del caso de uso.

En este informe sólo se muestran los casos de usos correspondientes a los procesos de "Inscribir a una familia que aspira a una vivienda social en el turno correspondiente" (ver Figura 4) perteneciente al paquete de inscripciones y "Solicitar renuncia a una vivienda" (ver Figura 5) pertenecientes al paquete de Operatorias de mesa, ya que si bien participé en el desarrollo de varios casos de usos y de las plantillas asociadas, debido a la extensión de las mismas incorporo solamente las correspondientes a los procesos mencionados.

A continuación, se presenta un ejemplo del diagrama de casos de usos de "Inscribir a una familia que aspira a una vivienda social en el turno correspondiente":

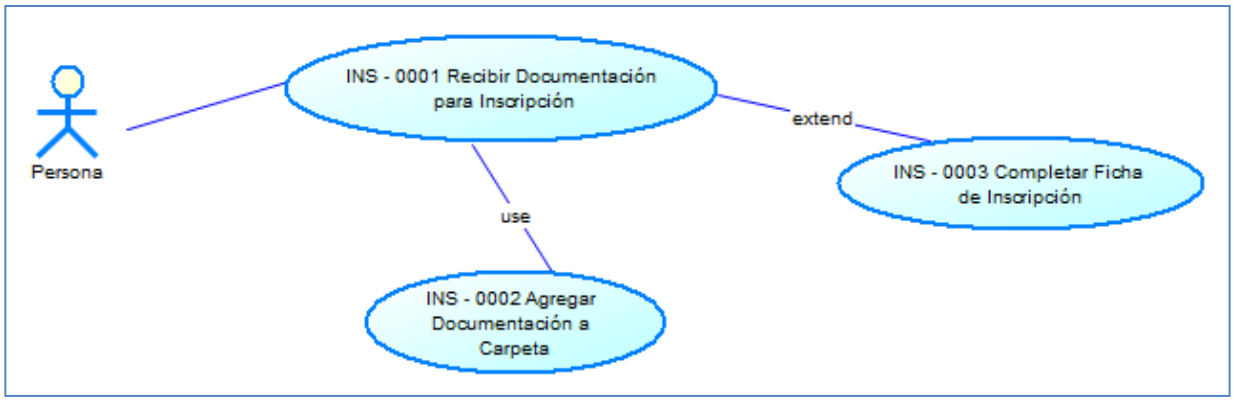


Figura 4: Fragmento del diagrama de casos de usos correspondiente a los procesos de inscripción.

A continuación, se muestran las planillas de los casos de usos del proceso de "Inscribir a una familia que aspira a una vivienda social en el turno correspondiente". Dicho proceso contiene los siguientes casos de usos:

- Recibir Documentación para inscripción (Ver Tabla 2).

- Agregar Documentación a carpeta (Ver Tabla 3).
- Completar Ficha de Inscripción (Ver Tabla 4).

Actividad	**Código**	**Nombre**
	INS - 0001	Recibir Documentación para Inscripción
Objetivo	Completar la documentación de inscripción y la ficha correspondiente.	
Pre-condiciones	Que se hayan informado los requisitos para la inscripción	
Especificación de la actividad	1. El responsable de Inscripción recibe documentación necesaria para la inscripción. 2. ¿La documentación está completa y es correcta? a. **Sí:** *i.* Se debe agregar la documentación en la carpeta correspondiente. *Se usa el caso de uso INS - 0002 Agregar Documentación a Carpeta.* ii. Continuar en paso 3. b. **No**: i. Se informa al postulante de la documentación faltante o errónea ii. Volver al paso 1. *3.* Responsable de Inscripción completa la ficha de inscripción. *Se extiende al caso de uso INS - 0003 Completar Ficha de Inscripción.* 4. Fin.	
Post-condiciones	No presenta.	
Req. no func. asoc.		
Observaciones		

Tabla 2: Recibir Documentación para inscripción.

Actividad	**Código**	**Nombre**
	INS – 0002	Agregar Documentación a Carpeta
Objetivo	Incorporar la documentación del inscripto en la carpeta correspondiente de la persona.	
Pre-condiciones	Que la persona haya presentado la documentación requerida.	
Especificación de la actividad	1. El Responsable de Inscripción ordena la documentación recibida por parte de la persona. 2. Incorpora toda la documentación requerida para la inscripción en la carpeta asociada a la persona. **3.** Fin.	
Post-condiciones	No presenta.	
Req. no func. asoc.		
Observaciones		

Tabla 3: Agregar Documentación a carpeta.

Actividad	Código	Nombre
	INS – 0003	Completar Ficha de Inscripción
Objetivo	Ingresar los datos correspondientes de la persona en la ficha de inscripción del sistema.	
Pre-condiciones	Que la persona haya presentado la documentación requerida.	
Especificación de la actividad	1. El Responsable de Inscripción debe completar la ficha de inscripción con los datos de la persona. 2. El Responsable de Inscripción confirma los datos ingresados en la ficha de inscripción. 3. Le comunica a la persona el número de inscripto asociado a la ficha. 4. La Persona recibe una copia de la ficha de inscripción y firma la misma. 5. Fin.	
Post-condiciones	No presenta.	
Req. no func. asoc.		
Observaciones		

Tabla 4: Completar Ficha de Inscripción.

A continuación se presenta el diagrama de casos de usos de "Solicitar renuncia a una vivienda" correspondiente al paquete de Operatorias de mesa:

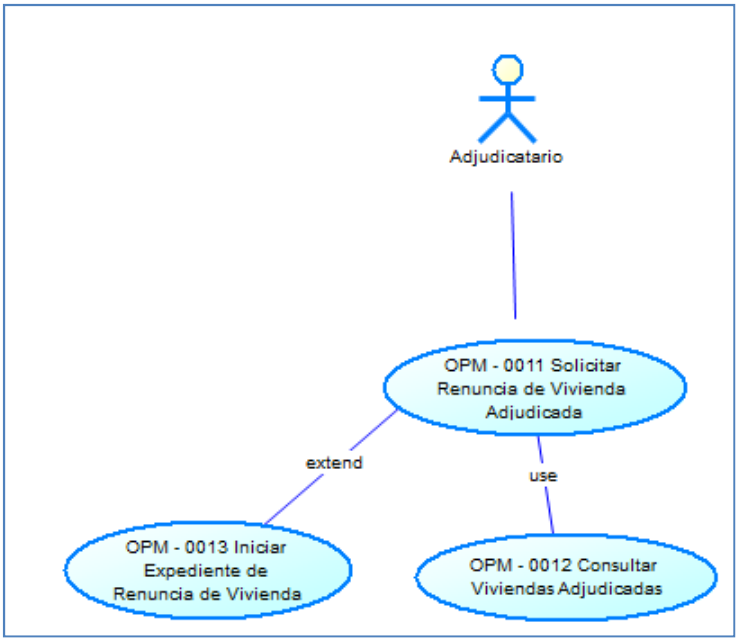


Figura 5: Fragmento del diagrama de caso de usos correspondiente al proceso de renuncia a una vivienda.

A continuación se muestran las planillas asociadas a los casos de usos del proceso de "Solicitar renuncia a una vivienda". Dicho proceso contiene las actividades

- Solicitar Renuncia de Vivienda Adjudicada (Ver Tabla 5).
- Consultar Viviendas Adjudicadas (Ver Tabla 6).
- Iniciar Expediente de Renuncia de Vivienda (Ver Tabla 7).

Actividad	**Código**	**Nombre**
	OPM– 0011	Solicitar Renuncia de Vivienda Adjudicada
Objetivo	Presentar en el organismo solicitud de renuncia a la vivienda social adjudicada	
Pre-condiciones	Que la persona sea un adjudicatario de vivienda y sea titular de la misma	
Especificación de la actividad	1. La encargada de Operatoria de Mesa recibe nota del adjudicatario para renunciar a la vivienda adjudicada. 2. Se debe buscar la vivienda adjudicada. *Se usa el caso de uso OPM - 0012 Consultar Viviendas Adjudicadas.* 3. ¿Se encontró la vivienda? a. **Sí:** i. Se debe actualizar los datos del titular de la vivienda. ii. Actualizar los datos de la vivienda a renunciar. iii. Continuar en paso 4. b. **No:** i. Se debe determinar porque la vivienda adjudicada no está cargada. ii. Continuar en paso 5. *i.* 4. La encargada de Operatoria de Mesa, debe crear o actualizar el expediente de la vivienda. *Se extiende al caso de uso OPM - 0013 Iniciar Expediente de Renuncia de Vivienda.* 5. Fin.	
Post-condiciones	No presenta	
Req. no func. asoc.		
Observaciones		

Tabla 5: Solicitar Renuncia de Vivienda Adjudicada.

Actividad	**Código**	**Nombre**
	OPM– 0012	Consultar Viviendas Adjudicadas
Objetivo	Llevar a cabo la búsqueda de las viviendas que fueron adjudicadas a las personas que salieron beneficiadas en el sorteo correspondiente	
Pre-condiciones	Que existan viviendas adjudicadas	
Especificación de la actividad	2. La encargada de Operatoria de Mesa ingresa al sistema para buscar la vivienda cuyo titular viene a renunciar. 3. Busca dicha vivienda a través del número de casa o de algún filtro de búsqueda especificado. 4. ¿Encontró vivienda? a. **Sí:** i. Continuar en paso 5. b. **No:** i. Informar que la vivienda no fue encontrada en el sistema. ii. Continuar en paso 6. 5. Encargada de Operatoria de Mesa, debe controlar que la información asociada a la vivienda se corresponda con el titular y grupo familiar del mismo. 6. Fin.	
Post-condiciones	No presenta	
Req. no func. asoc.		
Observaciones		

Tabla 6: Consultar Viviendas Adjudicadas.

Actividad	**Código**	**Nombre**
	OPM– 0013	Iniciar Expediente de Renuncia de Vivienda
Objetivo	Realizar la apertura del expediente correspondiente a la renuncia de una vivienda social que fue adjudicada	
Pre-condiciones	Que la persona se encuentre inscripta en el Organismo y corresponda a un adjudicatario de vivienda	
Especificación de la actividad	1. La encargada de Operatoria de Mesa, debe controlar en primer lugar si la vivienda adjudicada tiene iniciado un expediente. 2. ¿Existe expediente? a. **Sí:** i. Se incorpora la documentación asociada a la renuncia de la vivienda adjudicada. ii. Continuar en paso 3. b. **No:** i. Se inicia el expediente. ii. La encargada de Operatoria de Mesa solicita que se imprima carátula de renuncia y luego incorpora la documentación correspondiente. iii. Continuar en paso 3. 3. Se debe enviar luego el expediente a la Gerencia Técnica y Administrativa para que se realice la escritura correspondiente al nuevo titular de la vivienda renunciada. 4. Desde dicha Gerencia se debe notificar la finalización de la escritura al nuevo titular de la vivienda. 5. Fin.	
Post-condiciones	No presenta	
Req. no func. asoc.		
Observaciones		

Tabla 7: Iniciar Expediente de Renuncia de Vivienda.

Etapa 4: Relevamiento de la base de datos transaccional (origen de los datos):

El grupo de trabajo hizo un estudio del sistema de gestión de base de datos relacional a utilizar. En el caso en particular de los módulos que se están desarrollando, se eligió el sistema PostgreSQL v9.3 en combinación con el lenguaje imperativo propio del sistema PL/pgSQL. Este

sistema de gestión de base de datos tiene la ventaja de ser open source, contiene las funcionalidades requeridas para el sistema que se está desarrollando y junto a la herramienta de Interfaz gráfica PgAdmin III de PostgreSQL, que permite la administración y mantenimiento de las bases de datos, se crearon las diferentes tablas, vistas, índices, claves, entre otros, de la base de datos de inscripciones [9].

Se verificaron los resguardos que se tomarán en el momento de insertar, eliminar o modificar registros de la base de datos, ya sea en forma intencional o involuntaria, (verificación de datos entrantes, scripts, procedimientos, triggers, entre otros).

Se creó el Diagrama Entidad-Relación (DER) asociado a la base de datos del sistema.

Por directivas de confidencialidad asociadas con el IPAV, no es posible mostrar contenido alguno de la base de datos del sistema de inscripciones.

Capítulo 4: Bitácora de las actividades desarrolladas durante la práctica

El esquema de trabajo que se está utilizando para el desarrollo del sistema informático está basado en trabajo en célula, empleando SCRUM como esquema de gestión con ciclos cortos de 2 semanas.

SCRUM es una metodología ágil de gestión [10], cuyo principal punto es la coordinación y el trabajo colectivo de los integrantes del grupo de trabajo para lograr ciclos cortos, minimización de riesgos y generación periódica de entregables para el usuario. Otro punto importante de SCRUM es la determinación o re-determinación del orden de prioridades sobre las funcionalidades del sistema en desarrollo en forma coordinada con el cliente [11].

El equipo de trabajo está conformado por seis integrantes, incluido el Director del Proyecto. Durante cada ciclo el Director del Proyecto brinda las diferentes tareas a realizar durante el mismo.

Cada ciclo contiene los siguientes tipos de tareas:

- **Programación de la capa lógica y de los servicios Web**: Permite implementar las reglas que deben cumplirse a nivel funcional, la gestión de la información a partir de los usuarios y la base de datos, las funcionalidades involucradas y la creación de los distintos servicios Web RESTFul que serán utilizados desde la capa visual.

- **Diseño y desarrollo de los diferentes reportes del sistema:** Permite diseñar e implementar los diferentes reportes a mostrar a los distintos usuarios del sistema, además de la programación en la pantalla oportuna para que cuando el usuario acepte, puedan mostrarse los reportes desarrollados.

- **Definición de los permisos necesarios por rol**: Se definirán los permisos a nivel de base de datos para especificar qué roles pueden utilizar una determinada funcionalidad del sistema, o quienes tienen los permisos necesarios para realizar determinada acción (visualizar, registrar, modificar, borrar, etc.) sobre los datos de la base de datos; generalmente quien realiza la Programación de la capa lógica y de los servicios Web también efectúa esta tarea de manera conjunta.

- **Tareas de testing de funciones y servicios Web**: Se ejecutan diferentes tipos de pruebas unitarias sobre las distintas funciones y servicios Web que conforman el sistema en diferentes situaciones y bajo distintas condiciones, a los efectos de evaluar el accionar de la aplicación. Las situaciones propuestas de prueba unitaria sobre las componentes funcionales del sistema pueden basarse en: a) Analizar el comportamiento en estados esperados, o bien, b) Analizar el comportamiento en estados no esperados o anormales. De este modo, el objetivo de las pruebas es detectar errores en forma temprana (previo a la liberación del entregable del software), a los efectos de determinar las circunstancias en las que se produce la anomalía o comportamiento incorrecto.

  Una vez realizada la prueba unitaria, se documenta su resultado mediante el uso de plantillas especiales (Ver Tabla 8), la cual es supervisada por la responsable de Calidad. Posteriormente, la planilla supervisada es remitida al implementador original para que corrija los errores indicados, si es que existen. Corregidos los errores, vuelve para re-ejecutar los casos de prueba y si se verifica el éxito en todos los casos, se remite a la responsable de calidad para la revisión del documento. Verificado el documento de prueba y cumplido los casos de test, la funcionalidad se indica como satisfecha.

Nombre de Caso de Prueba:	
Objetivo:	
Pre – requisitos de ejecución:	
Tipo de Requerimiento:	
Éxito:	**No éxito:**
Descripción del caso de prueba:	
Resultado obtenido:	
Observaciones:	
Ejecutor del test case	
Responsable del test case	

Tabla 8: Plantilla usada para documentar un caso de prueba en particular.

Nombre de caso de Prueba: Es una descripción corta del caso de prueba que se está documentando (debe contener un código de caso de test). Un código contiene la siguiente forma:

#TC-"Una sigla del módulo del sistema que se está desarrollando"-"una sigla del servicio Web RESTFul programado"- "número del caso de prueba unitaria comenzando desde 001". Ejemplo: #TC-INSCWS-VIVPROY-001. Explicación: Caso de test número 001 correspondiente al módulo de inscripciones, del servicio Web RESTFul de viviendas de proyecto.

Objetivo: Es el propósito a alcanzar. Debe indicar, la función del servicio Web RESTFul que se está probando y los datos de entrada de la función indicada anteriormente.

Tipo de requerimiento: Determina si la función a probar es un requerimiento funcional o no funcional. En este caso los tests realizados pertenecen a requerimientos funcionales.

Pre – requisitos de ejecución: Es la condición necesaria que debe cumplirse para que el servicio Web pueda ejecutarse, siempre y cuando no tenga errores en su implementación.

Éxito: Resultado esperado de la función del servicio Web.

No éxito: Cualquier otro resultado que se obtiene que difiere al de éxito.

Descripción del caso de prueba: Es la secuencia de pasos a seguir para ejecutar la función del servicio Web que se está probando. (Debe indicar los tipos de datos de entrada y si los mismos son válidos, inválidos o null [sin valor]; el valor de un dato de entrada se elige de acuerdo a la prueba a realizar).

Resultado obtenido: Es un comentario que indica si la respuesta obtenida fue un éxito. Los valores posibles son "el esperado" y "no esperado".

Observaciones: Enumera la referencia al defecto implicado si un caso de prueba falla, o una explicación de las posibles causas de dicho fallo.

Ejecutor del test case: Integrante del equipo que realizó la prueba unitaria.

Responsable del test case: Integrante del equipo que realizó el servicio Web RESTFul que se está probando.

- **Programación de los diferentes prototipos de pantallas**: Se crean los prototipos de interfaces, es decir, se modela las distintas pantallas por las que eventualmente el sistema podría interactuar con el usuario. Se delimitan las condiciones que deben cumplirse para que el programa: a) muestre una interface en particular, b) se defina el intercambio de datos

entre pantallas (asociado con el diagrama de casos de uso) y c) se esboza el formato de la información a exponer.

- **Desarrollo de las funcionalidades de las pantallas:** Se desarrollan las distintas funcionalidades de la pantalla, de esta manera cuando el usuario ejecuta un comando de la misma, éste se ejecuta de acuerdo a lo esperado (según la dificultad de las funcionalidades a desarrollar esta tarea se realiza conjuntamente con la programación del prototipo de las pantallas).

- **Testing de los pantallas**: Se realizan pruebas sobre las distintas pantallas creadas, navegando sobre cada una de ellas y usando las diferentes características de la misma, verificando que las respuestas obtenidas sean las esperadas y los resultados que se muestran correspondan con lo que realmente se busca de cada pantalla, cada testing realizado sobre el prototipo es documentado con la misma plantilla que en el caso del testing de las funcionalidades del sistema y los servicios Web [Ver Tabla 8].

- **Documentación del sistema**: Cada etapa del desarrollo del sistema tiene su documentación técnica pertinente, la cual va constituyendo gradualmente la documentación técnica global del sistema que facilitará el posterior mantenimiento y el armado del manual para el usuario del sistema que se está desarrollando [6].

Actividades desarrolladas por lapsos temporales:

A continuación, y a los efectos de pormenorizar las tareas desarrolladas, se indican las actividades realizadas durante la práctica en empresa por lapsos temporales (semanal), con 20 horas por semana, iniciando desde el 6 de agosto del 2014 (Fecha de aprobación del anteproyecto del proyecto final "modalidad práctica en empresa") hasta el 17 de diciembre del mismo año (Fecha asociada con el último entregable del sistema presentado ante Presidencia).

[6-agosto-2014; 8-agosto-2014]

Tareas de Testings de funciones y servicios Web

Durante esta semana realicé tareas de prueba unitaria sobre las siguientes implementaciones del sistema, con la correspondiente documentación de las mismas.

- **Viviendas de proyecto**: Administra la información de las diferentes viviendas correspondiente a cada proyecto definido. El sistema desarrollado permite gestionar diferentes proyectos de casas a entregar.

  A través del software de gestión de tareas mantis [12], el integrante del trabajo que desarrolló esta implementación me derivó la incidencia para que hiciera las pruebas correspondientes de la misma; utilizando el marco JUnit [13], que son un conjunto de aplicaciones para hacer testing de aplicaciones Java, se realizaron las pruebas unitarias a través de un cliente de servicios Web RESTFul, las mismas se documentaron y fueron revisadas por la responsable de calidad; se derivaron al desarrollador para que hiciera las correcciones correspondientes y así sucesivamente hasta que el desarrollo de los servicios Web RESTFul correspondientes a Viviendas de proyectos funcione de manera correcta .

[11-agosto-2014; 15-agosto-2014]

Tareas de Testings de funciones y servicios Web

Durante esta semana realicé tareas de prueba unitaria:

- **Viviendas de proyecto**: Administra la información de las diferentes viviendas correspondiente a cada proyecto definido. El sistema desarrollado permite gestionar diferentes proyectos de casas a entregar.

  A través del software de gestión de tareas mantis [12], el integrante del trabajo que desarrolló esta implementación me derivó la incidencia para que hiciera las pruebas correspondientes de la misma.

Personas por localidad: Permite gestionar datos sobre las personas inscriptas relacionados a la localidad de residencia. Se utilizó el marco JUnit para realizar las pruebas unitarias de la implementación de personas por localidad, también para cada testing se utiliza la librería PostgreSQL JDBC Driver requerida por los servicios Web RESTFul para la correspondiente conexión a los registros de la base de datos. Asimismo durante las pruebas se determina si los permisos (triggers, funciones, entre otros) definidos en la base de datos funcionan correctamente.

- **Estados de ficha**: Permite administrar los diferentes estados que puede asumir una ficha de inscripción y/o integrante de la misma. Se efectuó el testing correspondiente de los servicios Web RESTFul de Estados de ficha y se documentó el mismo. Además durante las pruebas unitarias se examinaron los recursos de los servicios Web RESTFul. Un servicio Web RESTful expone un conjunto de recursos (resource) que identifican la manera en que éste interactuará con sus clientes. A través de estos recursos el cliente (aplicación) puede acceder a los diferentes servicios Web RESTFul.

Programación de la capa lógica y de los servicios Web

Durante la semana se realizaron las siguientes implementaciones de la capa lógica, de los diferentes servicios Web RESTFul y la definición de los permisos necesarios por rol para el acceso a la base de datos:

- **Modalidad de Entrega**: Permite administrar las diferentes modalidades de entrega de una vivienda. Se efectuaron los servicios Web RESTFul de Modalidad de entrega de una vivienda utilizando las librerías JAXB, PostgreSQL JDBC Driver y Jercey.

  Para tener servicios Web RESTFul es necesario poder transformar los objetos de java a una representación que pueda ser entendida por muchos lenguajes y de vuelta de esa representación a objetos java, para ello se crean los objetos del modelo y se los mapea para XML. Los documentos XML (Extensible Markup Language) se han convertido en un estándar para intercambio de datos entre programas y aplicaciones. En un esquema XML (o XSD) podemos definir los elementos que pueden aparecer en un documento XML así como las relaciones entre los mismos.

  A continuación se especifican los pasos para el desarrollo del servicio Web RESTFul de modalidad de entrega [14]:

En primer lugar se programó en java la clase **ModalidadEntrega** usando la librería JAXB que permite convertir los objetos de la clase a XML y luego leer esos mismos XML para obtener objetos Java (Ver Figura 6).

```
@XmlRootElement(name="Modalidadentrega")
@XmlType(propOrder={"idmodalidadengrega","modalidadnombre",
    "modalidaddescripcion",
    "registrologin","registrofechahora"})
@XmlAccessorType(XmlAccessType.NONE)
public class Modalidadentrega implements Serializable {
                        ...
/**
 * @return the idmodalidadengrega
 */
@XmlElement
public Long getIdmodalidadengrega() {...3 lines }

/**
 * @param idmodalidadengrega the idmodalidadengrega to set
 */
public void setIdmodalidadengrega(Long idmodalidadengrega) {...3 lines }
                        ...
 }
```

Figura 6; clase ModalidadEntrega, la cual utiliza propiedades de la librería JAXB

@XMLRootElement: Asigna una clase o un tipo enum a un elemento XML.

Name: nombre local del elemento XML.

@XMLType: Asigna una clase o un tipo enum a un tipo de esquema (schema) XML.

PropOrder: es una lista de nombres de propiedades JavaBean en la clase. El orden en que se enumeran las propiedades JavaBean es el orden de los elementos del esquema XML cuando dichas propiedades se mapean.

@XMLAccesorType: Controla si los campos o propiedades JavaBean son serializados por defecto.

XMLAccessType: Utilizado por XMLAccessorType para controlar la serialización de campos o propiedades.

None: Ninguno de los campos o propiedades se mapeará a XML a menos que estén especificados con algunas de las anotaciones JAXB.

@XMLElement: Mapea una propiedad JavaBean a un elemento XML derivando el nombre de la propiedad [15].

Luego se creó la clase **ModalidadesEntrega** que contiene una lista de ModalidadEntrega, y la clase **SesionModalidadEntregaParams** que contiene los parámetros necesarios para las funciones del servicio Web RESTFul de modalidad de entrega, también se utilizó para esta implementación las librerías de java y JAXB.

Seguidamente se desarrolló la interfaz de java con las funciones (sin implementar) del servicio Web de modalidad de entrega (Ver Figura 7).

```
public interface IModalidadEntrega {
    /** Permite la consulta por patrón ...7 lines */
    String consultarModalidadesEntrega(String idsesion, String patronName);

    /** Busca la instancia de modalidad de entrega vinculada con el ID indicado ...6 lines */
    String consultarModalidadEntregaByID(String idsesion, Long idmodalidad);

    /** Permite registrar diferentes Modalidades de entrega ...6 lines */
    String registrarModalidadesEntrega(String idsesion, ModalidadesEntrega lista);

    /** Permite modificar los datos de las modalidades de entrega indicadas ...6 lines */
    String modificarModalidadesEntrega(String idsesion, ModalidadesEntrega lista);

    /** Permite eliminar las modalidades de entrega indicadas en la medida que se tenga
    permisos y sea posible en base a las relaciones ...6 lines */
    String borrarModalidadesEntrega(String idsesion, ModalidadesEntrega lista);
}
```

Figura 7: Interfaz del servicio Web RESTFul de Modalidad de entrega

Luego se implementó cada una de las funciones de la interfaz del servicio Web de Modalidad de entrega, para ello se necesitó de la librería de servicios Web RESTFul de Jercey (Ver Figura 8):

```
/**
 * Clase que implementa los servicios web IModalidadEntregaImpl
 *
 * @author Bruno
 * @version 1.0
 */
public class IModalidadEntregaImpl extends BaseInscWSImpl implements IModalidadEntrega {

    public String consultarModalidadesEntrega(@Context ServletContext pcontexto,
            @Context HttpServletRequest prequest,String idsesion, String patronName) {...9 lines }

    @Override
    public String consultarModalidadesEntrega(String idsesion, String patronName) {...97 lines }

        ...

    public String registrarModalidadesEntrega(@Context ServletContext pcontexto,
            @Context HttpServletRequest prequest,String idsesion, ModalidadesEntrega lista) {...8 lines }

    @Override
    public String registrarModalidadesEntrega(String idsesion, ModalidadesEntrega lista) {...93 lines }

        ...
}
```

Figura 8: Implementación del servicio Web de Modalidad de entrega.

Se desarrollaron los recursos de los servicios Web de Modalidad de Entrega, también usando la librería Jercey (Ver Figura 9).

Como se mencionó anteriormente un servicio Web RESTful expone un conjunto de recursos que identifican la manera en que éste interactuará con sus clientes. A través de estos recursos el cliente puede acceder a los diferentes servicios Web RESTFul.

```
/**
 *
 * @author bruno
 */
@Path("/insc-ws-bruno")
public class InscWSResourceBruno {

 @Path("/consultarModalidadesEntrega")
 @GET
 @Produces(MediaType.TEXT_PLAIN)
 public String consultarModalidadesEntrega(@QueryParam("idsesion") String idsesion,
        @QueryParam("patronName") String patronName) {...5 lines }

 ...


 @Path("/registrarModalidadesEntrega")
 @POST
 @Produces(MediaType.APPLICATION_XML)
 @Consumes(MediaType.APPLICATION_XML)
 public String registrarModalidadesEntrega(SesionModalidadEntregaParams xml) {...9 lines }

 ...
```

Figura 9: Resources del servicio Web de Modalidad de entrega.

Luego se crearon los clientes de las funciones de los servicios Web [ver Figura 10]. Un cliente de servicios Web es cualquier componente o aplicación que se comunica con éstos.

```
  /** Permite la modificacion de Modalidad de entrega ...9 lines */
 public String modificarModalidadesEntrega(SesionModalidadEntregaParams requestEntity) throws ClientErrorException {...3 lines }

 /** Busca la instancia de Modalidad de entrega vinculada con el ID indicado ...10 lines */
 public String consultarModalidadEntregaByID(String idsesion, Long idmodalidad) throws ClientErrorException {...11 lines }

 /** Permite consultar las modalidades de entrega mediante un patron de busqueda ...10 lines */
 public String consultarModalidadesEntrega(String idsesion, String patronName) throws ClientErrorException {...11 lines }

  /** Permite el registro de Modalidad de entrega ...9 lines */
 public String registrarModalidadesEntrega(SesionModalidadEntregaParams requestEntity) throws ClientErrorException {...3 lines }

  /** Permite la modificacion de modalidad de entrega ...9 lines */
 public String borrarModalidadesEntrega(SesionModalidadEntregaParams requestEntity) throws ClientErrorException {...3 lines }
```

Figura 10: Clientes del servicio Web de modalidad de entrega.

Documentación del sistema

Durante esta semana realicé la siguiente documentación del sistema:

- **Diagramas de casos de uso del sistema:** Participé en el armado de los diferentes diagramas de casos de uso del sistema mediante la utilización del software Power Designer y las planillas asociadas que detalla cada caso de uso identificado usando Microsoft Office Word. Los casos de usos pueden ir actualizándose o pueden agregarse nuevos a medida que se desarrolla el sistema.

[19-agosto-2014; 22-agosto-2014]

Tareas de Testings de funciones y servicios Web

Durante esta semana realicé tareas de prueba unitaria:

- **Estado del grupo familiar**: Gestiona los estados por los que puede atravesar un integrante de la ficha, el cual puede ser independiente del estado de la ficha en sí. Se efectuó el testing conveniente de los servicios Web RESTFul correspondientes a Estado del grupo familiar, desarrollado por otro integrante del equipo de trabajo y se documentó el mismo. Las pruebas se repitieron hasta que las respuestas del servicio Web fueran las esperadas.
- **Informes contables de comodato**: Permite especificar los informes contables específicamente de comodato. Como en los casos anteriores otro participante del proyecto realizó la implementación de Informes contables de comodato y mediante el uso de un cliente de servicios Web RESTFul se realizaron las pruebas unitarias de la misma. Dichas pruebas fueron documentadas con la plantilla correspondiente, en la misma se debía indicar el resultado obtenido; "No esperado" si había algún error en la implementación (Ver Tabla 9) y "El esperado" si el caso de test era correcto (Ver Tabla 10).

Nombre de Caso de Prueba: Consultar Informes contable comodatos por filtros de búsqueda con parámetros válidos (#TC-INSCWS-INFCONCOM-0002)	
Objetivo: Probar la consulta de Informes contable comodatos por filtros de búsqueda usando el Webservice /consultarInformescontablecomodato, cuando se ingresan todos los parámetros válidos	
Pre – requisitos de ejecución: Haber iniciado sesión con permisos de consulta WS_INSCRIPCIONES_INFCTBLECOMODATO_CONSULTAR	
Tipo de Requerimiento: - Funcional	
Éxito: *Retorna un XML bajo el esquema Informescontablecomodato*	**No éxito:** *Cualquier otra salida*
Descripción del caso de prueba:	
Consultar Webservice con los parámetros: IPAV.ws.insc.jaxb.params.InformescontablecomodatoParams Filtros = válido java.lang.String idsesion = válido Mostrar la respuesta	
Resultado obtenido:	No esperado
Observaciones:	Error 1: en patronencabezado faltó agregar la sentencia `toUpperCase() de la siguiente manera: filtros. getPatronencabezado().toUpperCase())` Error 2: `Cuando se chequea si consolidalogin no es vacío aparece filtros.getCierralogin (). Debería aparecer if(!Ustring.isEmpty(filtros. getConsolidalogin()))`
Ejecutor del testcase	Bruno Cavallo
Responsable del test case	Integrante que implementó los servicios Web

Tabla 9: Ejemplo de plantilla con resultado no esperado

Nombre de Caso de Prueba: Consultar Viviendas de proyecto por su id con un id de proyecto nulo y un id de sesión inválido (#TC-INSCWS-VIVPROY-0020)	
Objetivo: Probar la consulta de Viviendas de proyecto por id usando el Webservice /consultarViviendaproyectoById, cuando se ingresa un id de proyecto nulo y un id de sesión inválido.	
Pre – requisitos de ejecución: Haber iniciado sesión con usuario con permiso de consulta WS_INSCRIPCIONES_VIVIENDAPROYECTO_CONSULTAR	
Tipo de Requerimiento: - Funcional	
Éxito: *Retorna un XML bajo el esquema MensajeIPAV, con código "GE-0005"*	**No éxito:** *Cualquier otra salida*
Descripción del caso de prueba:	
Consultar Webservice con los parámetros: java.lang.Integer idvivienda = válido java.lang.Long idproyecto = null java.lang.String idsesion = inválido Mostrar la respuesta	
Resultado obtenido:	**El esperado**
Observaciones:	Ninguna
Ejecutor del testcase	Bruno Cavallo
Responsable del test case	Integrante que implementó los servicios Web

Tabla 10: Ejemplo de plantilla con resultado El esperado

- **Informes asesoría legal para cancelación de viviendas**: Contiene informes de Asesoría Legal específicos para una cancelación. Se efectuaron las pruebas unitarias de los servicios Web RESTFul de Informes asesoría legal para cancelación de viviendas utilizando las librerías JUnit y PostgreSQL JDBC Driver.

Programación de la capa lógica y de los servicios Web

Durante la semana se realizaron las siguientes implementaciones de la capa lógica:

- **Motivos de devolución**: Permite especificar los distintos motivos de devolución de una vivienda.

  Se efectuaron los servicios Web RESTFul de motivos de devolución de una vivienda utilizando las librerías JAXB (Java Architecture for XML Binding), PostgreSQL JDBC Driver y Jercey [16].

Documentación del sistema

Durante esta semana realicé la siguiente documentación del sistema:

- **Diagrama de navegación del sistema de seguridad:** Desarrollé el diagrama de navegación del sistema de seguridad mediante el uso del software Power Designer.
  Un mapa de navegación es la representación gráfica de la organización de la información de una estructura Web. Expresa todas las relaciones de secuencia y permite elaborar escenarios de comportamiento de los usuarios. Para ello se usó el software Power Designer que permite, entre otras características el desarrollo del mapa de navegación. A continuación se muestra un fragmento del diagrama correspondiente a Buscar Usuario (Ver Figura 11).

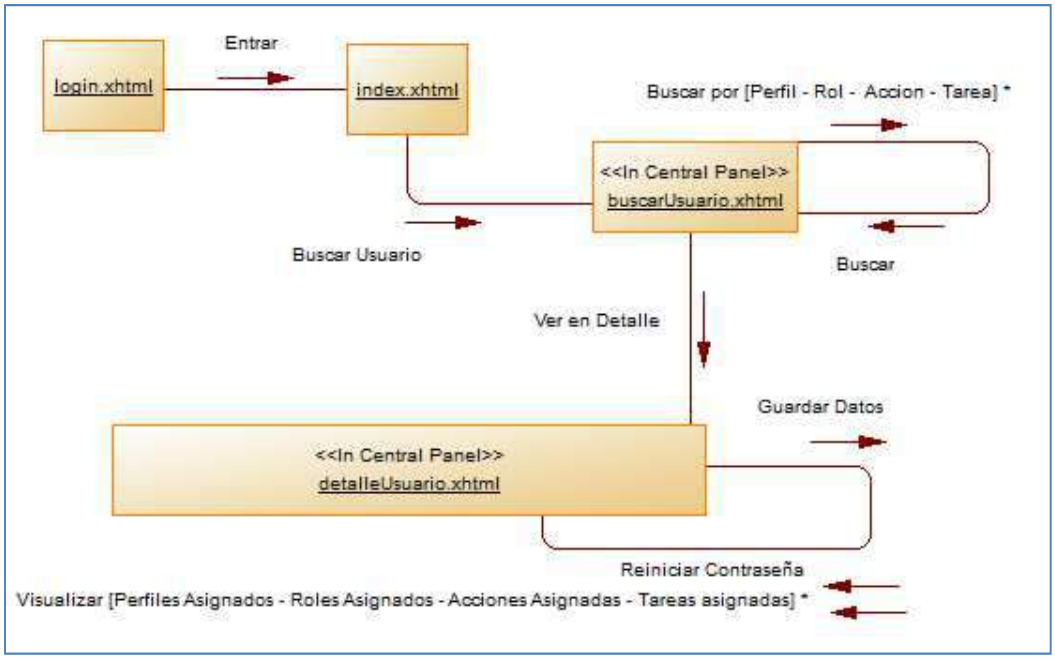


Figura 11: Fragmento del mapa de navegación de Seguridad correspondiente a Buscar usuario.

[25-agosto-2014; 29-agosto-2014]

Tareas de Testings de funciones y servicios Web

Durante esta semana realicé tareas de prueba unitaria:

- **Segmentos**: Determina los segmentos en los que se dividen las viviendas a entregar, según las características de los grupos familiares. El integrante del equipo de trabajo que realizó la implementación de segmentos creó una incidencia en el mantis, la misma me fue derivada para que hiciera las pruebas unitarias correspondientes utilizando las librerías ya mencionadas.

- **Histórico de segmento**: Permite administrar el Histórico con los diferentes segmentos en los que se dividen las viviendas a entregar. Realicé las pruebas unitarias correspondientes a Histórico de segmento utilizando un cliente de servicios Web RESTFul.
- **Anteriores planes de vivienda**: Permite administrar los distintos planes de viviendas en que eventualmente una persona podría haber sido adjudicataria anteriormente, de acuerdo a lo informado en la ficha de inscripción. Realicé las pruebas unitarias correspondientes a anteriores planes de viviendas utilizando las librerías adecuadas, dichas pruebas fueron documentadas con la plantilla correspondiente.
- **Planes vivienda**: Permite especificar los planes de viviendas que posee el IPAV. El integrante del equipo que realizó la implementación me derivó la incidencia a través del mantis para que efectuara las pruebas correspondientes y poder encontrar los posibles errores.

Programación de la capa lógica y de los servicios Web

Durante la semana se realizaron las siguientes implementaciones de la capa lógica:

- **Segmentos por proyectos**: Permite determinar las viviendas correspondientes en cada proyecto para cada uno de los segmentos. Mediante el uso de las librerías ya nombradas para la programación de los diferentes servicios Web se efectuó Segmentos por proyectos, el mismo fue derivado a otro integrante del equipo para que efectúe los casos de pruebas correspondientes y así encontrar posibles errores, para que luego pudiera corregirlos.

Documentación del sistema

Durante esta semana realicé la siguiente documentación del sistema:

- **Diagrama de navegación del sistema de seguridad (finalización):** Finalicé con el desarrollo del diagrama de navegación del sistema de seguridad mediante el uso del software Power Designer.

Desarrollo de las funcionalidades de las pantallas:

El diseño de pantallas consiste en la creación de diferentes pantallas que visualizará el usuario mientras utiliza el sistema y el desarrollo de sus funcionalidades .Durante esta semana, realicé las siguientes tareas de programación de pantallas y las funcionalidades respectivas:

- **Aspirantes / Denunciantes**: Permite el Alta, la Modificación y la Baja de las personas que aspiran un turno para poder inscribirse en el sistema, y de las personas que denuncian una

irregularidad en la entrega de una vivienda. Para el desarrollo de las pantallas de Aspirantes/Denunciantes se usó JSF (java server faces, tecnología y framework para aplicaciones Java basadas en la Web que simplifica el desarrollo de interfaces de usuario [17]), XHTML (HTML expresado como XML válido), Primefaces (es una librería de componentes para JSF de código abierto que cuenta con un conjunto de componentes enriquecidos que facilitan la creación de las aplicaciones Web) [18] y las librerías JAXB, PostgreSQL JDBC Driver y Jercey.

[01-septiembre-2014; 05-septiembre-2014]

Tareas de Testings de funciones y servicios Web

Durante esta semana realicé tareas de prueba unitaria:

- **Informes sociales**: Permite administrar los diferentes informes sociales generados. Realicé las pruebas de los servicios Web RESTFul de informes sociales, y completé las plantillas correspondientes.
- **Histórico de persona por discapacidad**: Histórico con los cambios de Discapacidades que presenta la persona. Desplegué las pruebas de Históricos persona por discapacidad, de los permisos de la base de datos, los recursos correspondientes y los clientes mediante el uso de las librerías correspondientes.
- **Histórico de persona**: Histórico que contiene la información sobre las personas que aspiran a una casa, o bien la tienen otorgada (contiene la información histórica de todas las personas). Utilizando el marco JUnit y la librería PostgreSQL JDBC Driver realicé las correspondientes pruebas unitarias de los servicios Web de Histórico de persona.

Desarrollo de las funcionalidades de las pantallas:

Durante esta semana, realicé las siguientes tareas de programación de pantallas y las funcionalidades respectivas:

Correcciones en Aspirantes / Denunciantes: La encargada de calidad, mediante la realización de testings de pantallas (usando las plantillas asociadas), me indicó y/o detectó un conjunto de observaciones y/o errores en las pantallas asociadas a Aspirantes / Denunciantes. Dichas observaciones fueron efectuadas y los errores corregidos, y así sucesivamente hasta que la pantallas y las funcionalidades correspondientes a Aspirantes/Denunciantes fueran las correctas.

[08-septiembre-2014; 12-septiembre-2014]

Tareas de Testings de funciones y servicios Web

Durante esta semana realicé tareas de prueba unitaria:

- **Adjudicados**: Permite administrar los diferentes adjudicatarios cuando ya les han entregado la llave y Acta de tenencia precaria. Mediante el uso de las librerías ya nombradas realicé el testing correspondiente y le derivé por Mantis la incidencia a la encargada de calidad, quién corrigió las planillas asociadas a las pruebas y le derivó la incidencia al implementador, el cual tuvo que corregir los errores encontrados.
- **Sorteos**: Permite especificar la información relacionada con cada sorteo. Se realizó el testing junto a las planillas asociadas, para que el implementador pudiera realizar las correcciones necesarias.
- **Detalle de Sorteo**: Guarda la información detallada de los titulares al momento del sorteo de viviendas. Cuando se terminaba el testing de los servicios web y el implementador corregía los errores encontrados, entonces se cerraba la incidencia en el Mantis, indicando que las pruebas habían finalizado.

Desarrollo de las funcionalidades de las pantallas:

Durante esta semana, realicé las siguientes tareas de programación de pantallas y las funcionalidades respectivas:

- **Tipos de viviendas**: Permite el Alta, la Modificación y la Baja de los tipos de viviendas sociales que pueden adjudicarse a una familia. Mediante el uso de XHTML y Primefaces se realizó el prototipo de pantalla de tipo de viviendas, se programó la misma (el Bean de JSF) utilizando los servicios Web RESTFul, hasta que la funcionalidad completa del caso de uso quedase satisfecha. Culminada la funcionalidad, se derivó la implementación de tipo de viviendas para que se proceda a la prueba funcional y de integración con el resto del sistema.

[15-septiembre-2014; 19-septiembre-2014]

Diseño de prototipos:

Durante esta semana realicé tareas de diseño de prototipos (esquemas visuales representativos de la funcionalidad pero no funcionales).

- **Informes sociales:** Permite el Alta, la Modificación y la Baja de los informes que las asistentes sociales elaboran al momento de visitar una vivienda. Utilizando Primefaces y el servidor de

aplicaciones Tomcat, se procedió a esquematizar cómo debiera lucir la pantalla Web de informes sociales en base a los casos de usos y requerimientos. Cuando el esquema quedó desarrollado, se deriva para implementación funcional (Ver Figura 12).

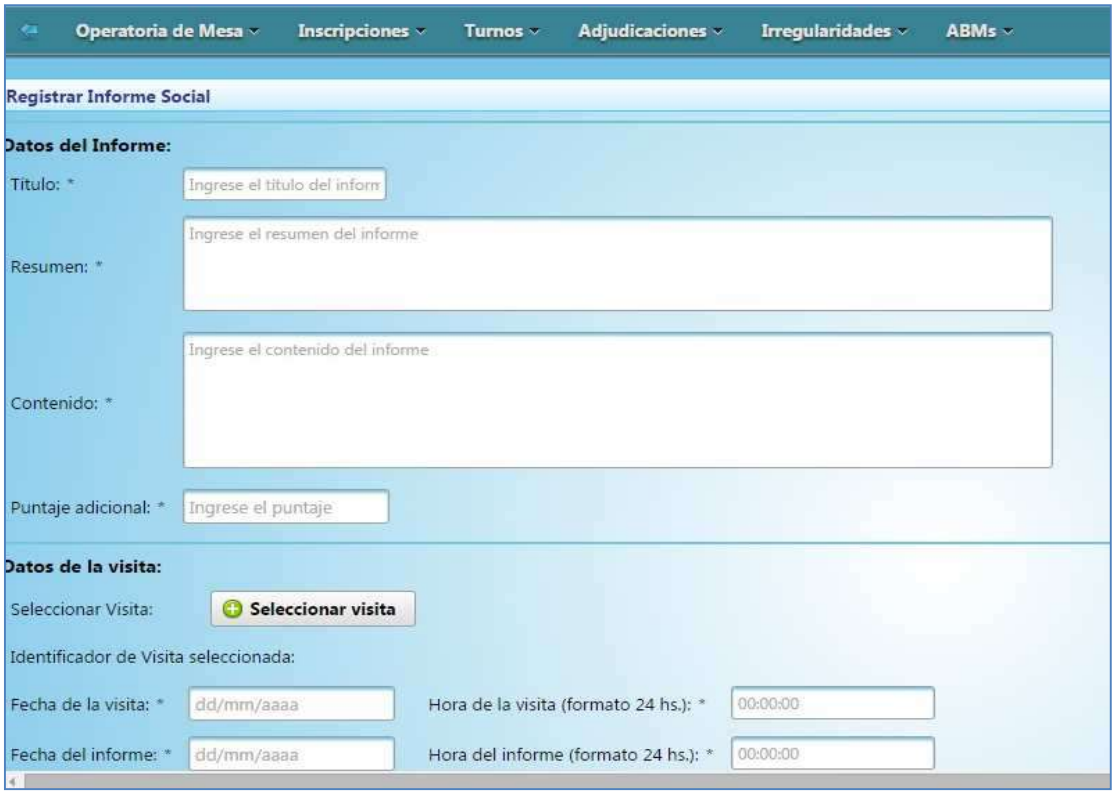


Figura 12: Prototipo de pantalla de registrar un nuevo informe social.

- **Visitas planificadas**: Contiene la planificación de visitas al grupo familiar por parte de las asistentes sociales (Ver Figura 13); usando la librerías y el servicio Web ya mencionados en NetBeans, se procedió a esquematizar la pantalla Web de visitas planificadas en base a los casos de usos y requerimientos.

Figura 13: Prototipo de pantalla de visualizar visitas planificadas a una familia.

[22-septiembre-2014; 26-septiembre-2014]

Diseño de prototipos:

Durante esta semana realicé tareas de diseño de prototipos:

- **Continuación de Informes sociales:** Se corrigieron errores del prototipo de pantallas de informes sociales detectados por la encargada de calidad; cuando dicho prototipo quedó finalizado, otro integrante del equipo realizó la programación de las funcionalidades de la pantalla consumiendo los servicios Web RESTFul.
- **Continuación de Visitas planificadas:** Utilizando la librería de componentes Primefaces continué con el prototipo y la encargada de calidad procedió a indicarme distintas observaciones y/o errores encontrados en el mismo. A continuación se muestra el prototipo de pantalla de modificar visita planificada (Ver Figura 14).

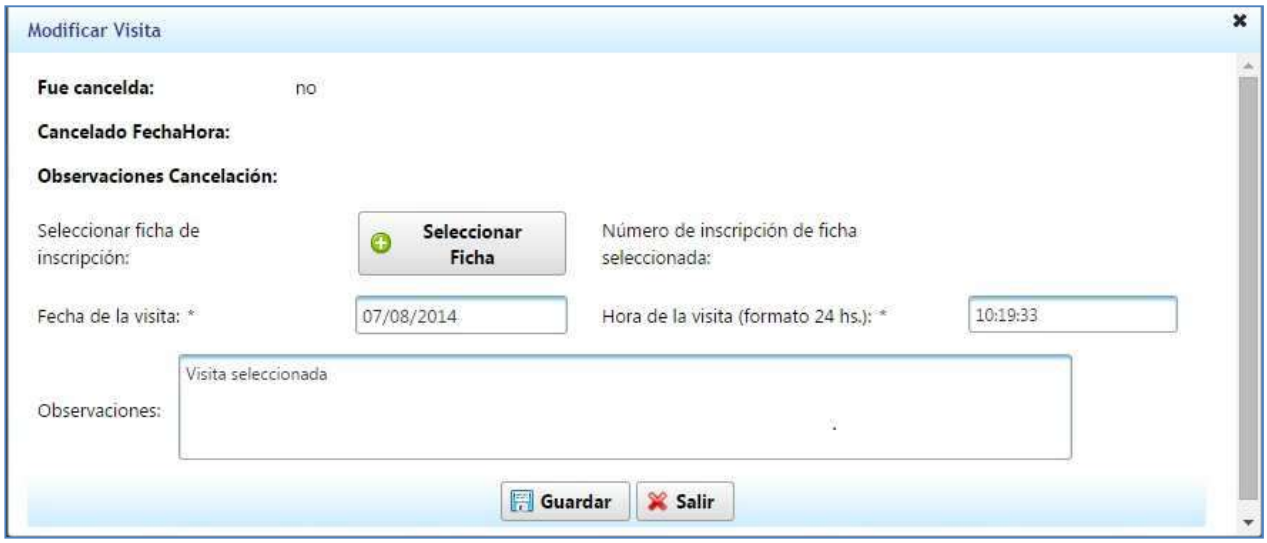


Figura 14: Modificar una visita planificada.

[29-septiembre-2014; 03-octubre-2014]

Diseño de prototipos:

Durante esta semana realicé tareas de diseño de prototipos:

- **Visitas planificadas (Correcciones de observaciones realizadas por la encargada de calidad):** Después de haber realizado el prototipo de visitas planificadas, la semana anterior la encargada de calidad me efectuó un conjunto de observaciones y/o detectó errores, de ésta manera modifiqué el prototipo de acuerdo a lo informado por ella.

- **Informes sociales (Correcciones de observaciones realizadas por la encargada de calidad):** Además de corregir los errores y efectuar las observaciones realizadas por la encargada de calidad de visitas planificadas, también modifiqué lo que me observó de informes sociales.
- **Viviendas de proyecto:** Administra la información de las diferentes viviendas correspondiente a cada proyecto definido. El sistema desarrollado permite gestionar diferentes proyectos de casas a entregar. Me encargué del prototipo de pantallas de viviendas de proyecto, mediante el uso de Primefaces y JSF, y luego mediante mantis le pasé la incidencia a la encargada de calidad para que hiciese las observaciones pertinentes, para posteriormente se pueda proseguir con la programación de las distintas funcionalidades de las pantallas prototipadas.

[6-octubre-2014; 10-octubre-2014]

Diseño de prototipos:

Durante esta semana realicé tareas de diseño de prototipos:

Viviendas de proyecto (Correcciones de observaciones realizadas por la encargada de calidad): Luego de que la encargada de calidad me expresara diferentes observaciones sobre los prototipos de viviendas de proyecto, realicé los cambios en los mismos de acuerdo a dichas observaciones. A continuación se puede observar un esquema de como el usuario interactúa en las diferentes pantallas de viviendas de proyecto, cuando dicho usuario ingresa a "Viviendas de Proyectos" desde el menú de opciones (Ver Figura 15).

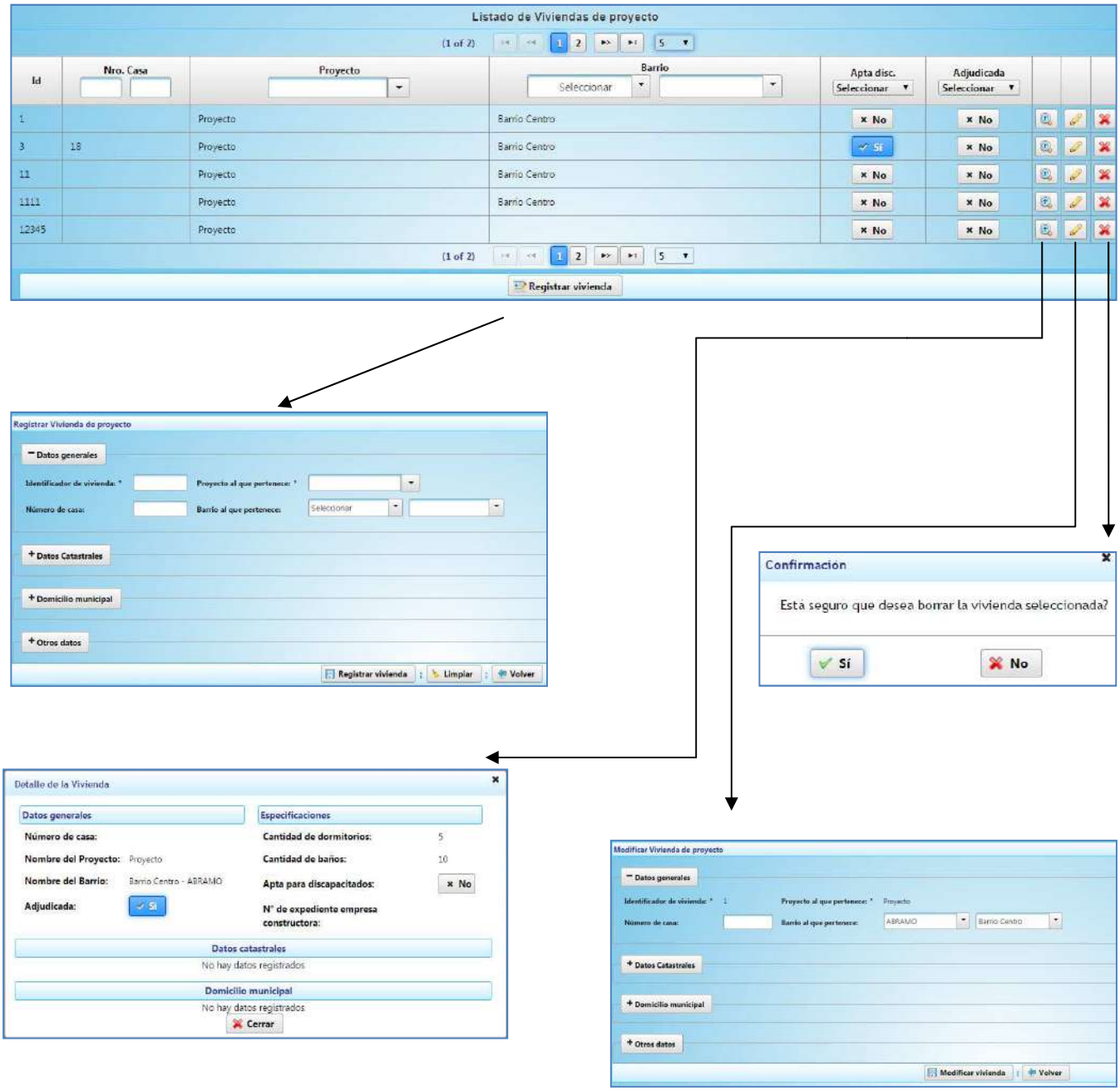


Figura 15: Esquema de interacción del usuario con las diferentes pantallas de viviendas de proyecto.

- **Impugnaciones por notas (inicio)**: Permite incorporar notas a la impugnación para mantener actualizada la información por si fuere requerida. Mediante el uso de las librerías mencionadas anteriormente, comencé con el prototipo de Impugnaciones por notas.

[14-octubre-2014; 17-octubre-2014]

Tareas de Testings de funciones y servicios Web

Durante esta semana realicé tareas de prueba unitaria:

- **Renuncias:** Guarda registro de las solicitudes de renuncias y resoluciones sobre las viviendas que han sido adjudicadas. Utilizando el marco JUnit y la librería PostgreSQL JDBC Driver realicé las correspondientes pruebas unitarias de los servicios Web de Renuncias hasta que éstos quedaran sin errores.

Diseño de prototipos:

Durante esta semana realicé tareas de diseño de prototipos:

Impugnaciones por notas (Continuación): Utilizando Primefaces, JSF y el servidor de aplicaciones Tomcat, se procedió a continuar con el prototipo de la pantalla Web de Impugnaciones por notas en base a los casos de usos y requerimientos.

[20-octubre-2014; 24-octubre-2014]

Tareas de Testings de funciones y servicios Web

Durante esta semana realicé tareas de prueba unitaria:

- **Certificados adjudicación:** Almacena copia de los certificados de adjudicación emitidos a los adjudicados. Mediante el uso de las librerías JUnit y PostgreSQL JDBC Driver realicé las correspondientes pruebas unitarias de los servicios Web de certificados de adjudicación, completé la planilla asociada, la cual fue observada por la encargada de calidad y luego pasé mediante Mantis la incidencia al integrante del equipo que realizó los servicios Web para su posterior corrección.
- **Irregularidades:** Almacena las denuncias en relación a las irregularidades de viviendas. Utilizando las librerías mencionadas anteriormente efectué el testing de irregularidades y completé la planilla asociada, el integrante del equipo realizó las correcciones necesarias, y nuevamente efectué las pruebas unitarias, y así sucesivamente hasta que el servicio Web quedara sin errores.

Documentación del sistema

Durante esta semana realicé la siguiente documentación del sistema:

- **Mapa de navegación del sistema de inscripciones (Inicio):** Comencé con el diagrama de navegación del sistema de inscripciones mediante el uso del software Power Designer.

[27-octubre-2014; 31-octubre-2014]

Tareas de Testings de funciones y servicios Web

Durante esta semana realicé tareas de prueba unitaria:

- **Operatorias de cambio titular entre fichas:** Almacena los pedidos de cambios de titularidad entre fichas. Es decir, los nuevos titulares corresponden a una ficha diferente de quien actualmente es titular. A través de mantis, el integrante del trabajo que desarrolló esta implementación me derivó la incidencia para que hiciera las pruebas correspondientes de la misma; utilizando el marco JUnit realicé las pruebas unitarias a través de un cliente de servicios Web RESTFul, las mismas se documentaron y fueron revisadas por la responsable de calidad; se derivaron al desarrollador para que hiciera las correcciones correspondientes y así sucesivamente hasta que el desarrollo de los servicios Web RESTFul correspondientes a Operatorias de cambio titular entre fichas, funcione de manera correcta .
- **Escrituras definitivas**: Es la escrituración definitiva de la vivienda en favor del adjudicatario. Realicé las pruebas correspondientes de escrituras definitivas para que el integrante que realizó la implementación del servicio Web pudiera corregir los errores encontrados en dichas pruebas.

Documentación del sistema

Durante esta semana realicé la siguiente documentación del sistema:

- **Mapa de navegación del sistema de inscripciones (continuación):** continué con el desarrollo del diagrama de navegación del sistema de inscripciones mediante el uso del software Power Designer. A medida que se van agregando pantallas al sistema, este diagrama se irá actualizando en posteriores semanas. A continuación se muestra un fragmento del diagrama correspondiente a Viviendas de Proyectos (ver Figura 16).

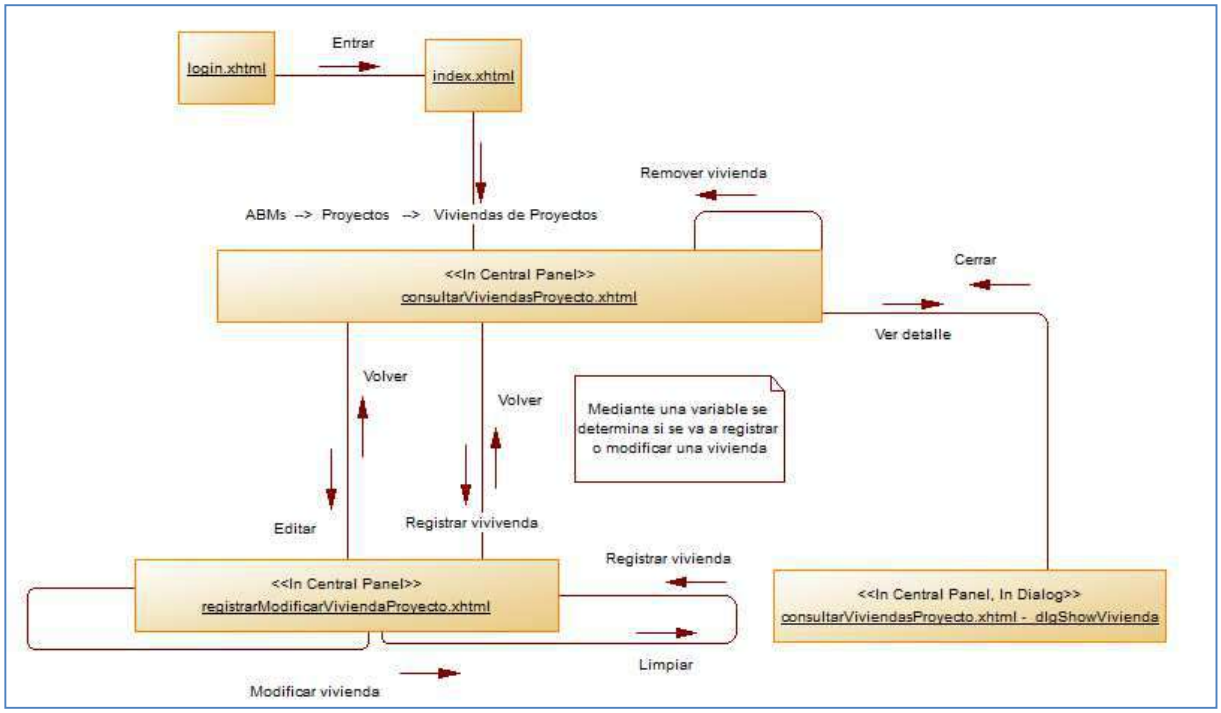


Figura 16: Fragmento del mapa de navegación de Inscripciones correspondiente a Viviendas de Proyectos.

[3-noviembre-2014; 7-noviembre-2014]

Tareas de Testings de funciones y servicios Web

Durante esta semana realicé tareas de prueba unitaria:

- **Irregularidades por nota:** Son notas que van incorporándose a la irregularidad para mantener actualizada la información, por si fuere requerida. Realicé las pruebas correspondientes usando el marco JUnit y completé la planilla asociada, repetí estos pasos hasta que el servicio Web RESTFul quedara sin errores.
- **Acta constatación de Irregularidades**: Corresponde a las diferentes actas de constatación asociadas con irregularidades. Efectué el testing correspondiente a acta constatación de Irregularidades y derivé la incidencia mediante Mantis al integrante que realizó la implementación para que éste pudiera corregir los errores detectados.

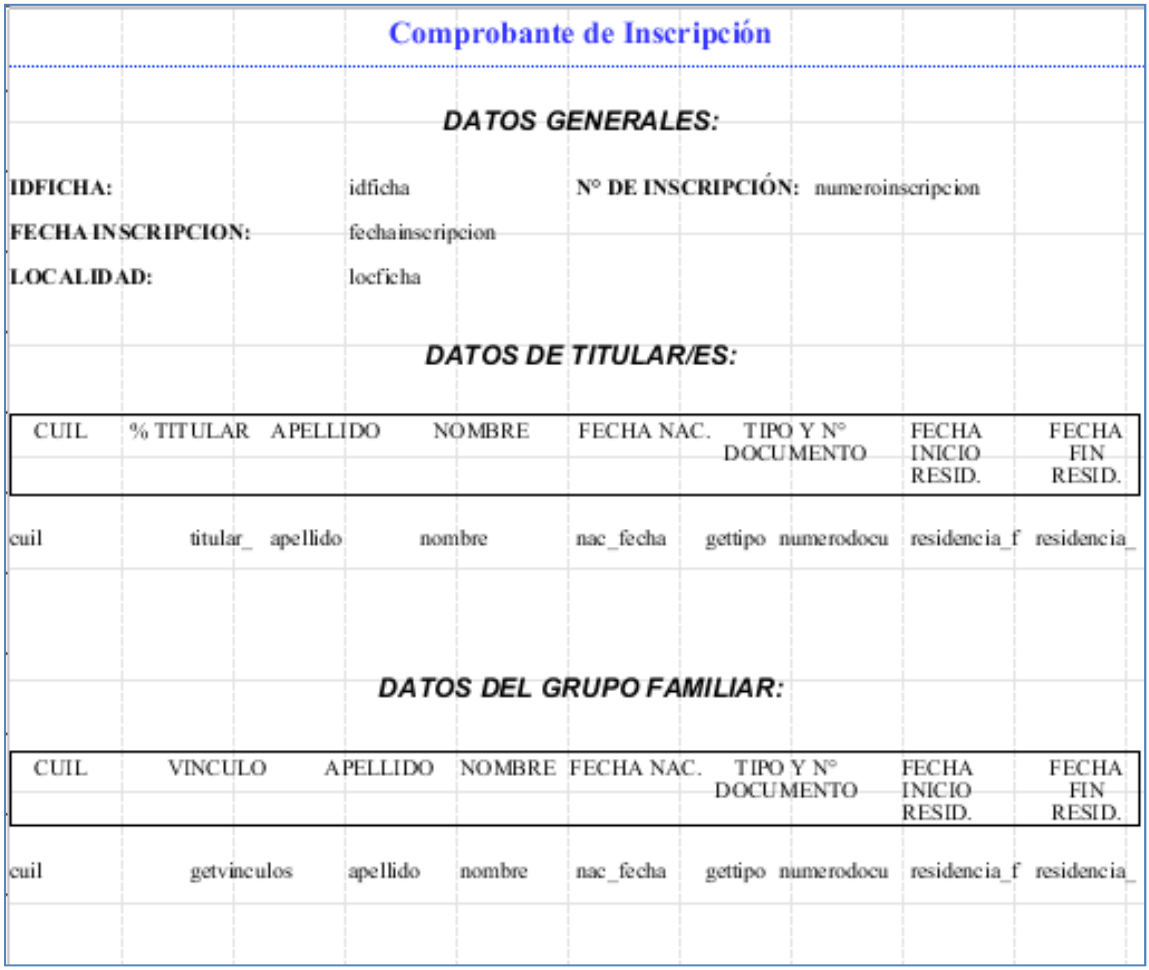
[10-noviembre-2014; 14-noviembre-2014]

Diseño y desarrollo de los diferentes reportes del sistema:

Durante esta semana diseñé e implementé los siguientes reportes a mostrar a los distintos usuarios del sistema, en las pantallas que se mencionan.

- **Comprobante de datos/inscripción de ficha (inicio):** En la pantalla de registrar fichas de inscripción, se cargan los datos de la ficha, luego se presiona el botón "Verificar datos de la

ficha" y se crea el documento de la ficha en formato PDF, para que la persona confirme que los datos a registrar sean correctos; si estos son exactos, en ese momento se registran en la base de datos. Luego del registro se remite el certificado de inscripción de la ficha. Mediante el uso del programa Report Designer y las librerías JFreeReport, Penthano e Itext desarrollé los reportes correspondientes (Ver Figura 17).



Comprobante de Inscripción

DATOS GENERALES:

IDFICHA: idficha N° DE INSCRIPCIÓN: numeroinscripcion

FECHA INSCRIPCION: fechainscripcion

LOCALIDAD: locficha

DATOS DE TITULAR/ES:

CUIL	% TITULAR	APELLIDO	NOMBRE	FECHA NAC.	TIPO Y N° DOCUMENTO		FECHA INICIO RESID.	FECHA FIN RESID.
cuil	titular_	apellido	nombre	nac_fecha	gettipo	numerodocu	residencia_f	residencia_

DATOS DEL GRUPO FAMILIAR:

CUIL	VINCULO	APELLIDO	NOMBRE	FECHA NAC.	TIPO Y N° DOCUMENTO		FECHA INICIO RESID.	FECHA FIN RESID.
cuil	getvinculos	apellido	nombre	nac_fecha	gettipo	numerodocu	residencia_f	residencia_

Figura 17: Fragmento del Certificado de inscripción de ficha desarrollado con el software Report Designer.

[17-noviembre-2014; 21-noviembre-2014]

Diseño y desarrollo de los diferentes reportes del sistema:

Durante esta semana diseñé e implementé los siguientes reportes:

- **Comprobante de datos/inscripción de ficha (continuación):** Continué con el desarrollo de los reportes de comprobación de datos de la ficha de inscripción y el comprobante de inscripción. Mediante el uso del programa Report Designer y las librerías ya mencionadas. Realicé la programación correspondiente en los JavaBean (JSF) y en las clases de java de "fichas de inscripción" y en las páginas Web de "registrar/consultar fichas de inscripción ",

para que muestre los certificados correspondiente cuando el usuario ejecute los comandos necesarios, el comprobante de datos antes de registrar la ficha y el comprobante de ficha luego del registro en la base de datos de la misma.

- **Certificado de adjudicación (Inicio):** Comencé con el desarrollo del reporte del certificado de adjudicación de una familia a una vivienda. Mediante el uso de del programa Report Designer y las librerías mencionadas diseñé y desarrollé el Certificado de adjudicación.

[24-noviembre-2014; 28-noviembre-2014]

Diseño y desarrollo de los diferentes reportes del sistema:

Durante esta semana diseñé e implementé los siguientes reportes:

- **Certificado de adjudicación (continuación):** continué con el desarrollo del reporte de certificado de adjudicación de una familia a una vivienda mediante el uso de del programa ya mencionado anteriormente y las librerías JFreeReport, Penthano y Itext. Realicé la programación necesaria, para que muestre el certificado de adjudicación de la vivienda a una familia según la ficha de inscripción de la misma.

[1-diciembre-2014; 5-diciembre-2014]

Diseño y desarrollo de los diferentes reportes del sistema:

Durante esta semana diseñé e implementé los siguientes reportes:

- **Comprobante de datos/inscripción de ficha (continuación):** corregí errores y realicé las observaciones detectadas por la encargada de calidad del reporte correspondiente al Comprobante de datos/inscripción de ficha
- **Certificado de adjudicación (Continuación):** Efectué los cambios correspondientes en el Certificado de adjudicación según lo detectado por la encargada de calidad.
- **Comprobante de turno:** Es el Comprobante que se le entrega a la persona cuando se le asigna un turno para la posterior inscripción de su familia, la cual aspira a una vivienda. Otro integrante del equipo diseñó e implementó dicho comprobante, yo realicé la programación en los JavaBean (JSF) y clases de java de "turnos" y en las páginas Web de "solicitar turno", para que muestre el comprobante luego de que se asigne un turno a la familia que aspira a una vivienda.

[9-diciembre-2014; 12-diciembre-2014]

Documentación del sistema

Durante esta semana realicé la siguiente documentación del sistema:

- **Mapa de navegación del sistema de inscripciones:** Continué con el progreso del diagrama de navegación del sistema de inscripciones mediante el uso del software Power Designer, de acuerdo a las pantallas que se fueron desarrollando desde la última actualización de dicho mapa.

[15-diciembre-2014; 17-diciembre-2014]

Documentación del sistema

Durante esta semana realicé la siguiente documentación del sistema:

- **Mapa de navegación del sistema de inscripciones:** Finalicé el mapa de navegación según las pantallas programadas hasta el momento, teniendo en cuenta la creación del último entregable del año a la Gerencia correspondiente.

Capítulo 5: Aplicaciones y Tecnologías empleadas durante la práctica

A lo largo de las diferentes semanas y durante toda la práctica, he utilizado los siguientes programas, Librerías y componentes:

1. **NetBeans v8.0.1 [19]:** Es un entorno de desarrollo, que permite la integración de diferentes lenguajes de programación, obteniendo así el sistema que se está programando.

   Este programa fue utilizado durante toda la práctica ya que el mismo permite ejecutar los diferentes sistemas que se están desarrollando, debido a que contiene un servidor Web integrado, posee documentación de los lenguajes más importantes y posibilita la identificación de los errores a medida que se va codificando el sistema, como también indica sugerencias sobre los comandos a utilizar, permite además realizar testings sobre el sistema, entre otras características. A continuación podemos ver el programa con los proyectos de seguridad e inscripciones y el conjunto de directorios que lo conforman (Ver Figura 18).

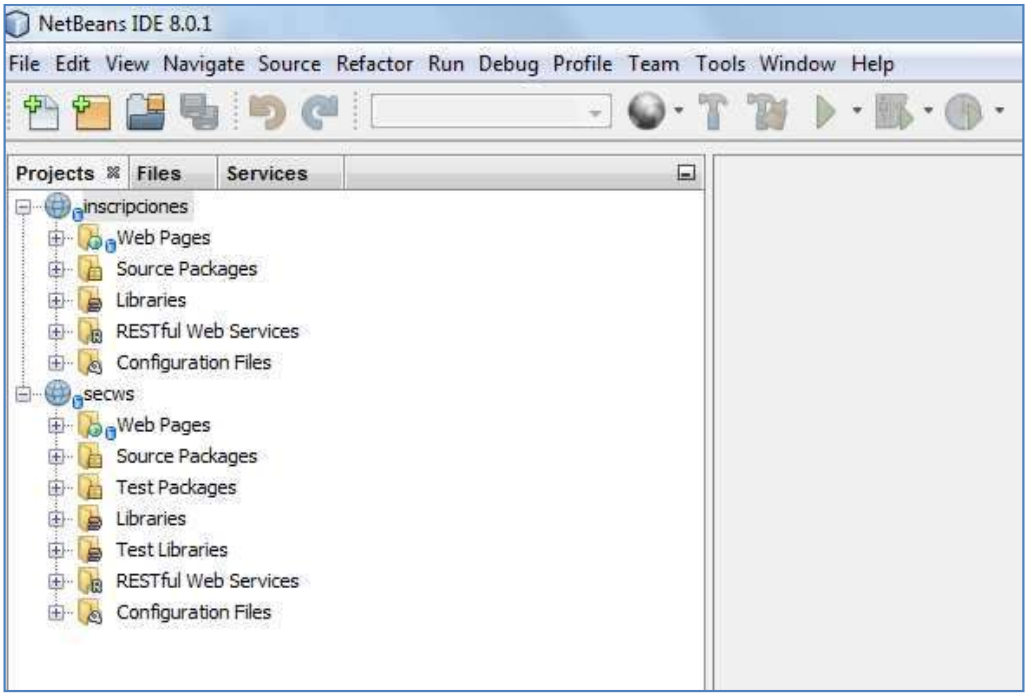


Figura 18: Programa NetBeans donde se muestra la implementación (sólo paquetes) de proyectos de seguridad e inscripciones.

2. **Report Designer v5.0.1 [20]:** Es un software que permite crear el diseño de diferentes reportes. Los mismos son incluidos en el sistema para que el usuario, mediante diferentes comandos, pueda visualizarlos. El software tiene la capacidad de generar reportes en distintos formatos PDF, HTML, Excel, etc.

En esta práctica se generaron diferentes reportes a visualizar por el usuario, como por ejemplo el comprobante de turno para inscripción de una grupo familiar para aspirar a una vivienda, el comprobante de la ficha de inscripción, el listado de personas inscriptas en el sistema, entre otros reportes.

A continuación se muestra el diseño de un reporte donde se muestra el listado de inscriptos en el IPAV realizado mediante el programa Report Designer (Ver Figura 19).

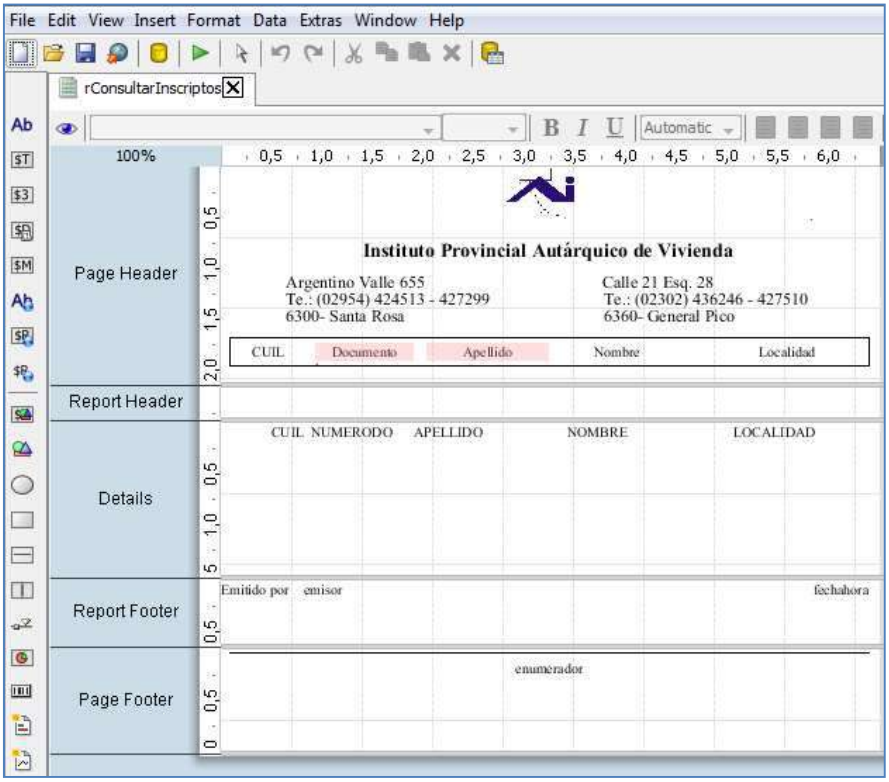


Figura 19: Programa Report Designer donde se muestra la implementación de un reporte de listado de inscriptos en el IPAV.

3. **PgAdmin III v1.18.1 [9]**: Es un administrador de la base de datos PostgreSQL. Permite definir los distintos componentes de la base de datos: tablas, vistas, índices, claves, entre otros, ingresar los resguardos que se tomarán en el momento de insertar, eliminar y/o modificar registros de la base de datos ya sea en forma intencional o involuntaria (verificación de datos entrantes, scripts, procedimientos, triggers, entre otros).

   En esta práctica el sistema se utilizó además, para ingresar datos de prueba en las distintas tablas de la base de datos para que, al momento de hacer los testings correspondientes de las funcionalidades del sistema, las respuestas de las pruebas unitarias sean las esperadas según los datos ingresados en las distintas tablas.

   El software es importante porque permite registrar datos y verificar que las distintas funciones y triggers definidos funcionen correctamente.

Cada integrante del equipo de trabajo posee un usuario y contraseña (Ver Figura 20) a las diferentes bases de datos del nuevo sistema que se está desarrollando para poder ejecutar las pruebas necesarias. Entre las pruebas que podemos realizar a nivel de base de datos está:

- Insertar un registro de una tabla.
- Modificar un registro de una tabla.
- Borrar un registro de una tabla.
- Modificar el estado de un informe. (Por ej. un informe social tiene diferentes estados: ABIERTO, CONSOLIDADO, CERRADO, ANULADO).
- Detectar el funcionamiento de las distintas funciones PL/pgSQL desarrolladas.
- Verificar el funcionamiento de las diferentes funciones y triggers desarrollados.
- Otras pruebas.

Figura 20: Programa PgAdmin III donde se muestra la conexión a las bases de datos correspondientes de seguridad e inscripciones.

4. **Power Designer v16.1.0 [8]:** Es un software que permite el modelado de datos, metadatos y procesos del sistema que se está desarrollando y la especificación de los diferentes diagramas que también formarán parte de la documentación del sistema.

   En ésta práctica se usó para desarrollar los diferentes diagramas de paquetes, diagramas de clases, mapas de navegación, etc. (Ver Figura 21).

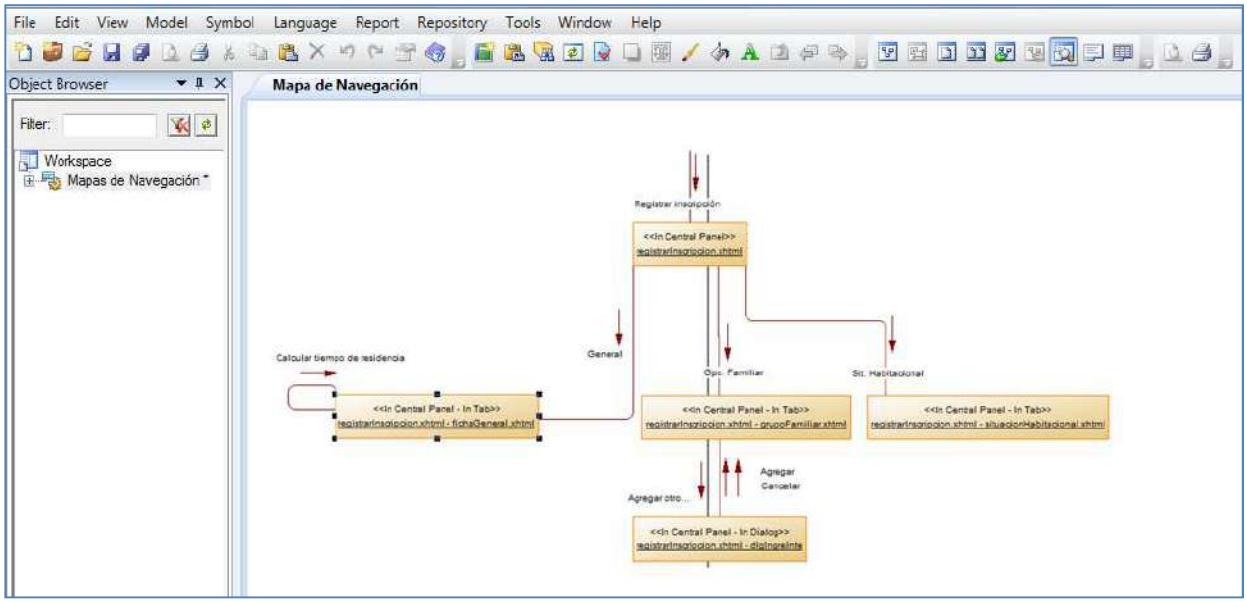


Figura 21: Programa Power Designer donde se visualiza un fragmento del mapa de navegación del sistema de inscripciones.

5. **Eclipse Process Framework (EPF) Composer v1.5.1 [2]:** Es un software que permite definir los procesos SPEM. Es decir delimitar los procesos de desarrollo de software y sistemas, y sus componentes.

   En esta práctica se utilizó para especificar los procesos SPEM de cada actividad del IPAV (ver Figura 22).

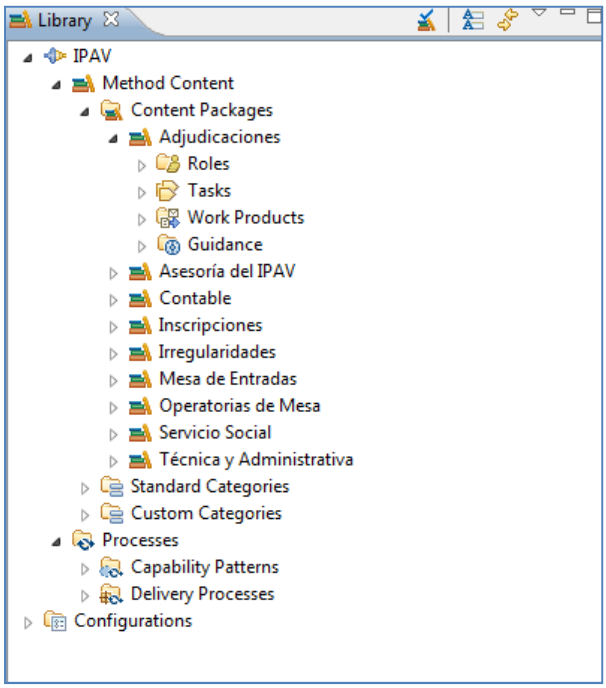
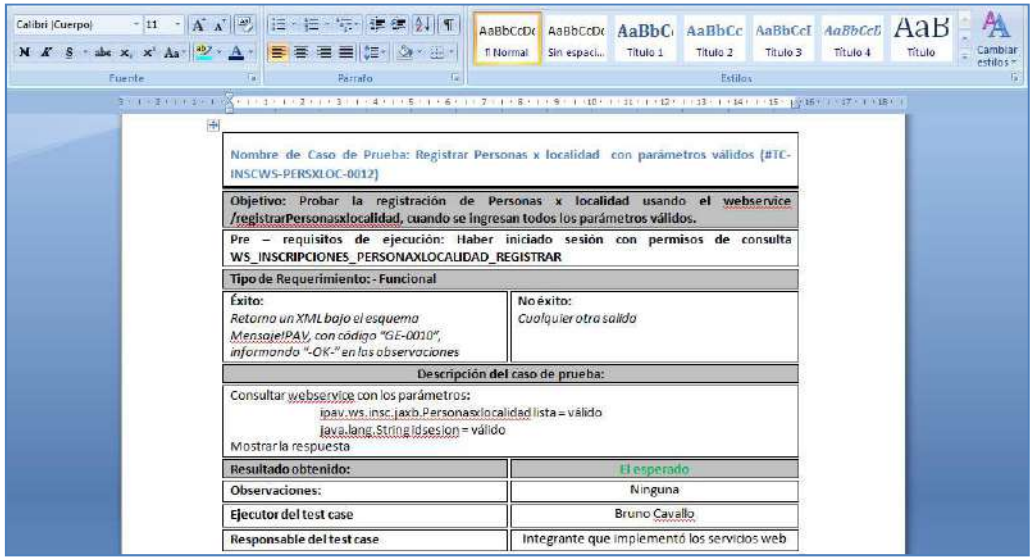


Figura 22: Programa Eclipse Process Framework (EPF) y los procesos SPEM definidos.

6. **Herramientas de Microsoft office v2007 [6, 8]:** son un conjunto de aplicaciones de escritorio que permiten definir diferentes documentos específicos. En esta práctica se usaron para crear los diferentes manuales para el usuario, plantillas de los diferentes casos de usos, casos de tests, y demás documentos que se integran al sistema.

   A continuación se puede observar la planilla de un caso de Personas por localidad donde se utilizó la herramienta Microsoft Word (Ver Figura 23).



Nombre de Caso de Prueba: Registrar Personas x localidad con parámetros válidos (#TC-INSCWS-PERSXLOC-0012)	
Objetivo: Probar la registración de Personas x localidad usando el webservice /registrarPersonasxlocalidad, cuando se ingresan todos los parámetros válidos.	
Pre – requisitos de ejecución: Haber iniciado sesión con permisos de consulta WS_INSCRIPCIONES_PERSONAXLOCALIDAD_REGISTRAR	
Tipo de Requerimiento: - Funcional	
Éxito: *Retorna un XML bajo el esquema MensajeIPAV, con código "GE-0010", informando "-OK-" en las observaciones*	**No éxito:** *Cualquier otra salida*
Descripción del caso de prueba:	
Consultar webservice con los parámetros: ipav.ws.insc.jaxb.Personasxlocalidad lista = válido; java.lang.String idsesion = válido; Mostrar la respuesta	
Resultado obtenido:	El esperado
Observaciones:	Ninguna
Ejecutor del test case	Bruno Cavallo
Responsable del test case	Integrante que implementó los servicios web

Figura 23: Ejemplo donde se documenta un caso de test en particular usando Microsoft Word.

7. **PostgreSQL v9.3.1 con PL/pgSQL [9]:** es un sistema de gestión de base de datos objeto-relacional, con código fuente disponible libremente y el lenguaje propio de PostgreSQL PL/pgSQL que permite ejecutar SQL mediante un conjunto de sentencias imperativas y de funciones.

   Este sistema de base de datos se utilizó para el sistema de inscripciones que se está desarrollando, además del lenguaje propio de PostgreSQL: PL/pgSQL para el desarrollo de las distintas funcionalidades a nivel de base de datos (funciones, triggers, y demás) (ver Figura 24).

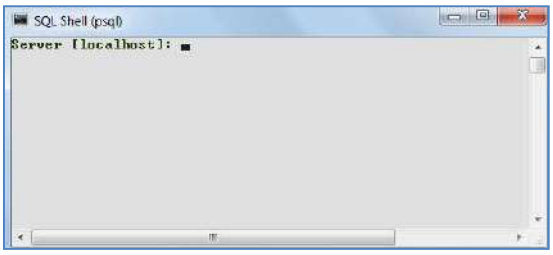


Figura 24: Componente de postgreSQL.

8. **Apache Tomcat 8.0.9.0 [21]**: Apache Tomcat es un servidor Web donde se alberga el sistema desarrollado para ser accedido desde las diferentes computadoras. Mediante servlets el servidor atiende cada petición al sistema.

   El IPAV proporcionó un servidor que nos permitió albergar los diferentes sistemas que se están desarrollando, en él se instaló Apache Tomcat para acceder a dichas aplicaciones.

   Además el NetBeans incluye el servidor Apache para ejecutar los sistemas desde nuestra computadora a medida que los vamos programando (Ver Figura 25).

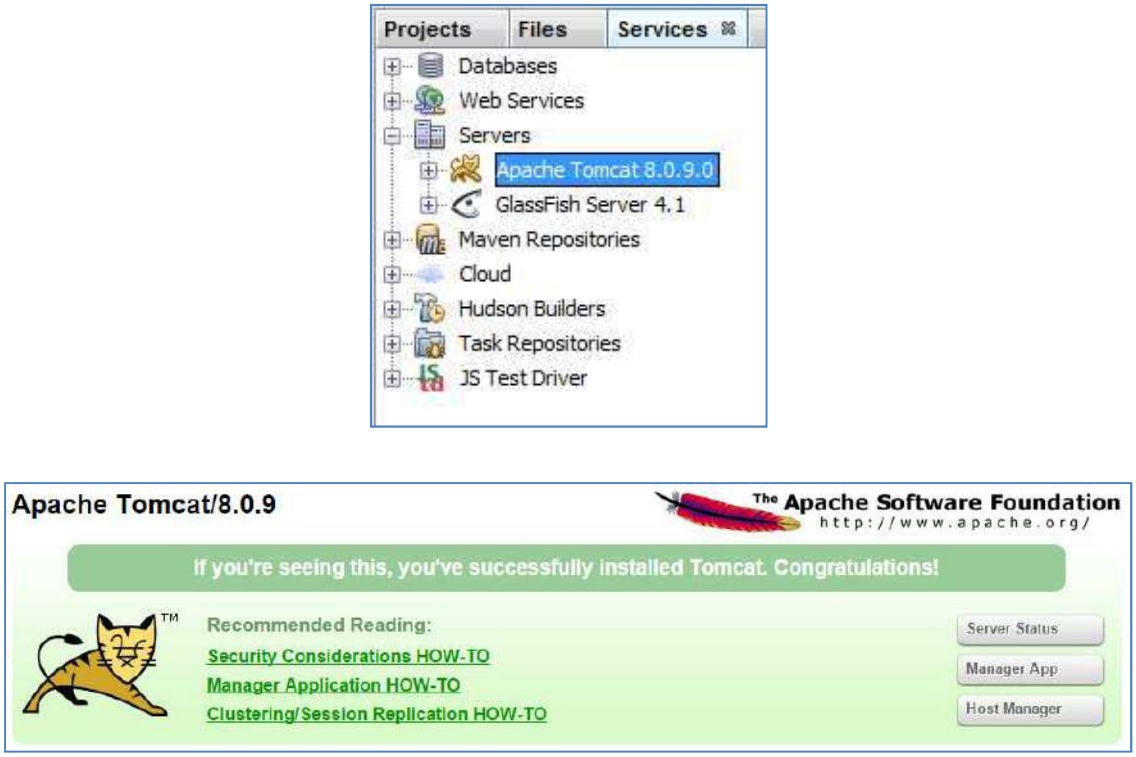


Figura 25: Apache Tomcat funcionando en NetBeans.

9. **Apache Httpd v2.2.15 [21]**: es el servidor de HyperText Transfer Protocol (HTTP) de Apache. Está diseñado para ser ejecutado como un proceso daemon standalone. Cuando se utiliza de esta manera se creará un grupo de procesos o tareas para gestionar las solicitudes (Ver Figura 26).

Figura 26: Proyecto Httpd de Apache Tomcat.

10. **Mantis v1.2.15 [12]**: Es una solución orientada a la Web para la gestión de las diferentes tareas entre los integrantes del equipo de trabajo. Con mantis se puede dividir el proyecto en varias categorías y definir los integrantes del equipo que trabajarán en cada una de ellas. Se registra cada tarea o incidente a realizar y quien debe atenderla. El sistema permite además la transición de las diferentes tareas entre los integrantes del equipo cuando ésta contiene acciones que deben ejecutarlas varios integrantes, generando así un flujo de las mismas, lo que permite hacer un seguimiento más exacto de éstas. Puede definirse quién puede abrir problemas, quién puede analizarlos y quién puede atenderlos. Es posible especificar un número indeterminado de estados para cada tarea (abierta, encaminada, testeada, devuelta, cerrada, reabierta…) y tantos perfiles como sea necesario (programador, tester, coordinador, entre otros).

    Durante el transcurso de la cédula de trabajo, se debate y organiza que tarea o funcionalidad del sistema debe realizar cada integrante del equipo, la misma se registra en mantis y es asignada a ese integrante (Ver Figura 27). Cuando la tarea está finalizada, de ser necesario se asigna a otro integrante para efectuar acciones sobre la misma, sobre todo el testing correspondiente; al culminar el testing el incidente vuelve al primer integrante para que éste corrija los errores o bugs encontrados, manteniendo así un control de avance de los trabajos realizados y cuales faltarían terminar.

    Si la tarea finaliza y el testing determina que no se encontraron más errores de funcionamiento la misma pasa a estado de cerrada.

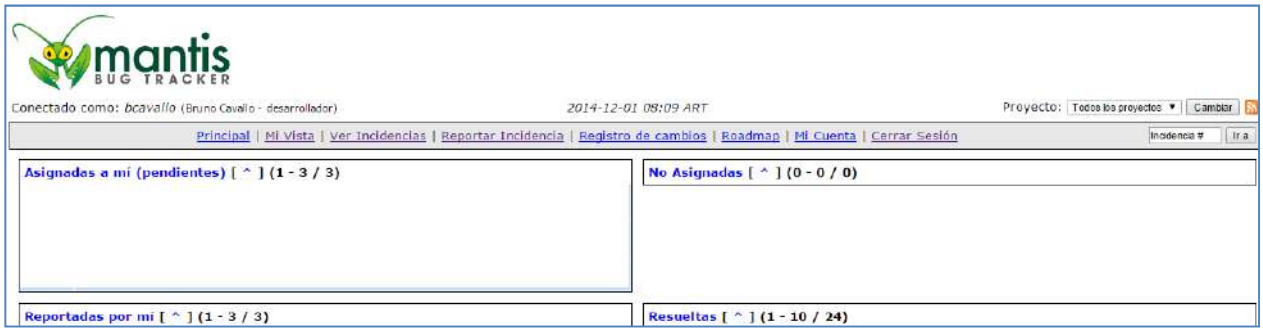


Figura 27: Fragmento de una pantalla del software Mantis (orientado a la Web).

11. **Git v1.7.1 [22]:** Es un software que permite gestionar las diferentes versiones de la aplicación en la que se está trabajando. Cuando hablamos de versiones nos referimos a cualquier cambio que se realice en sus diferentes archivos. Git es un sistema de versionado distribuido.

En los sistemas distribuidos, cada desarrollador tiene su propio repositorio. Estos se comunican con un repositorio central el cual permite compartir los cambios con otros desarrolladores.

Cuando un usuario desea actualizar su propio repositorio, luego de haber realizado cambios en los archivos del mismo, utiliza el comando "commit", cabe mencionar que dichos cambios se encuentran solamente en el repositorio local y no son visualizados por los demás desarrolladores. Cuando dichos cambios están comprobados, sin errores y mejoran la funcionalidad del sistema, el usuario los sube al repositorio central utilizando el comando "push" y a partir de allí se encuentran en el repositorio central. Igualmente aún no se hallan en los repositorios locales de los demás desarrolladores, para ello cada uno debe ejecutar el comando "pull" para descargar los cambios desde el repositorio central (Ver Figura 28).

También es posible la ejecución de los comandos", commit", "push" y "pull" entre los repositorios locales de los diferentes integrantes del equipo (Sistema distribuido) (Ver Figura 29). Además existen otros comandos de Git, de menor importancia que los nombrados anteriormente, como son "rebase", "diff", entre otros (Ver Figura 30).

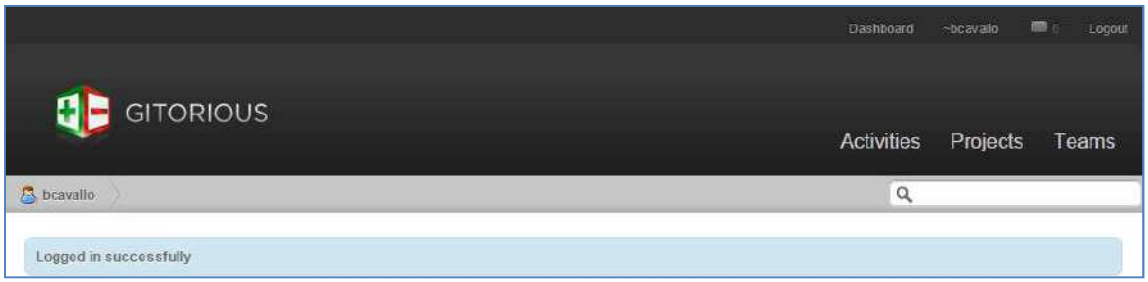


Figura 28: Fragmento de una pantalla del software Git (orientado a la Web).

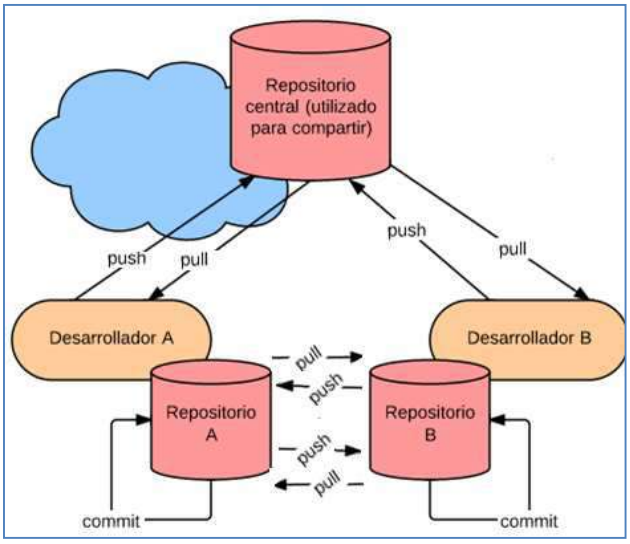


Figura 29: Mecanismo de funcionamiento del software Git.

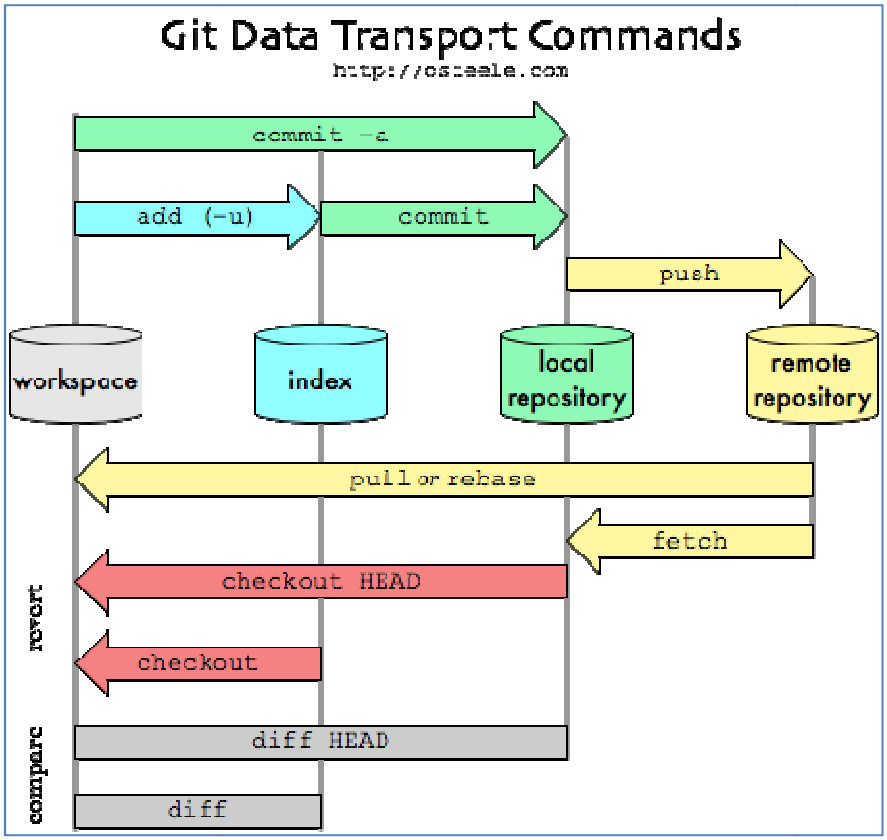


Figura 30: Comandos de transporte de datos de Git.

Servicios Web RESTFul con Jersey, JAXB, JodaTime y Google-GSON [5]:

REST (representational State Transfer), es un tipo de arquitectura de desarrollo Web que se apoya totalmente en el estándar HTTP. REST nos permite crear servicios y aplicaciones que pueden ser utilizadas por cualquier cliente que interprete HTTP.

12. **Jersey v2.4.1 [5]:** El servicio Web RESTful de desarrollo que soporta perfectamente exponer la información en una variedad de tipos de medios de representación y los detalles de bajo nivel de la comunicación cliente-servidor. El marco Jersey es de código abierto, para el desarrollo de servicios Web RESTful en Java que proporciona soporte para JAX-RS APIs. Jersey ofrece su propia API que se extienden al conjunto de herramientas JAX-RS con características y utilidades adicionales para simplificar el desarrollo de servicios Web REST y sus clientes en Java.

13. **Java Architecture for XML Binding (JAXB) v2.2.5 [15]:** permite asignar clases de Java a representaciones XML. JAXB proporciona dos características principales: la capacidad de

serializar las referencias de objetos Java a XML y a la inversa, es decir, deserializar XML en objetos Java.

14. **JodaTime v2.3 [14]:** Joda-Time ofrece una alternativa bajo licencia Apache 2.0 para el manejo de fechas y horas a la brindada por el JDK de Oracle.

15. **Google-GSON v2.2.4 [14]:** Gson es una librería Java que se puede utilizar para convertir objetos Java en su representación JSON. También se puede utilizar para convertir una cadena JSON a un objeto Java equivalente.
    JSON (JavaScript Object Notation): es un formato de intercambio de datos ligero. Es sencillo para las máquinas para analizar y generar. Se basa en un subconjunto del Lenguaje de Programación JavaScript. JSON es un formato de texto que es completamente independiente del lenguaje pero utiliza convenciones de C, C ++, C #, Java, JavaScript, Perl, Python, y muchos otros. Estas propiedades hacen que JSON sea un lenguaje importante para el intercambio de datos.

16. **Pentaho Reporting v5.1.0 [20]:** Define un conjunto de librerías programadas en Java para la generación de diferentes reportes.

17. **JFreeReport v0.8.5 [20, 23]:** Es una librería de Java de reportes gratuita. Permite usar datos obtenidos a través de la interfaz TableModel de Swing y la salida de archivos a la pantalla o impresora en varios formatos de exportación (PDF, HTML, CSV, Excel, texto plano). Además posee soporte para servlets.

18. **Itext v2.1.7 [23]:** es una biblioteca Open Source para crear y manipular archivos PDF, RTF, y HTML en Java. Con Itext un documento puede ser exportado en múltiples formatos, o múltiples instancias del mismo formato.

19. **Apache Poi v3.10 [23]:** La misión del Proyecto Apache POI es crear y mantener las API de Java para manipular varios formatos de archivos basados en los estándares Office Open XML (OOXML) y formato de Documento Compuesto de Microsoft OLE 2 (OLE2). Es decir, se pueden leer y escribir archivos de MS Excel, MS Word y MS PowerPoint usando Java.

20. **Java Server Faces (JSF) v2.2 [17]:** Es el framework para aplicaciones Web en Java. JSF es un framework orientado a la interfaz gráfica de usuario (GUI), facilitando el desarrollo de éstas, y que sin embargo, realiza una separación entre comportamiento y presentación, además de proporcionar su propio servlet como controlador, implementando así los principios del patrón de diseño Model-View-Controller (MVC), lo que da como resultado un desarrollo más simple y una aplicación mejor estructurada.

21. **JavaServer Pages Standard Tag Library (JSTL) v1.2.2 [24]:** Encapsula etiquetas simples para funcionalidades básicas comunes a muchas aplicaciones Web. JSTL tiene soporte para tareas comunes, estructurales como la iteración y los condicionales, etiquetas para manipular documentos XML y SQL. También proporciona un framework para la integración de etiquetas personalizadas existentes con las etiquetas JSTL.

22. **PrimeFaces v4.0 [18]:** PrimeFaces es una librería de componentes para JavaServer Faces (JSF) de código abierto que cuenta con un conjunto de componentes enriquecidos que facilitan la creación de las aplicaciones Web.

    Es utilizado durante el desarrollo de éste sistema para crear los diferentes componentes de la pantalla que visualiza el usuario, ya sea para entrada o salida de datos. Por ejemplo, en caso de que una pantalla contenga un formulario que el usuario deba completar, con Primefaces se pueden incluir diferentes componentes de entrada de datos, que van desde una caja de texto para escribir el "nombre de una persona" hasta un calendario para ingresar la "fecha de nacimiento" de la misma (Ver Figura 31).

Figura 31: Captura de Pantalla del sistema de inscripciones utilizando componentes de Primefaces.

En la pantalla se observa el ingreso de datos de una persona para registrar la misma en el sistema, luego se le asigna un turno para que se pueda inscribir y así aspirar al otorgamiento de una vivienda.

23. **JUnit v4.11 [13]:** Es un conjunto de bibliotecas creadas que son utilizadas en programación para hacer pruebas unitarias de aplicaciones Java.
    JUnit es un conjunto de clases (framework) que permite realizar la ejecución de clases Java de manera controlada, para poder evaluar si el funcionamiento de cada uno de los métodos de la clase tiene el comportamiento esperado. Es decir, en función de algún valor de entrada se evalúa el valor de retorno esperado, si la clase cumple con la especificación, entonces JUnit devolverá que el método de la clase pasó exitosamente la prueba; en caso de que el valor esperado sea diferente al que regresó el método durante la ejecución, JUnit devolverá un fallo en el método correspondiente.

Otras librerías y componentes usados

A continuación se enumeran otras librerías o componentes que se usaron a lo largo de la práctica:

1. **Java Development Kit o (JDK) v1.8.0.25 [14]:** Es un software que provee herramientas de desarrollo para la creación de programas en Java.
2. **Conector Apache Tomcat v8.0.9.0 [21]:** Permite integrar el servidor apache en NetBeans.
3. **Afternoon v1.0.10 [18]:** Tema de Primefaces que permite mejorar la visualización de las páginas Web.
4. **PostgreSQL JDBC Driver v9.3 [4]:** Permite conectar la aplicación Web con la base de datos PostgreSQL.

Otras utilidades usadas

A continuación se enumeran las diferentes utilidades que se usaron a lo largo de la práctica que se relacionan indirectamente con el desarrollo del sistema:

1. **Skype v6.22.60 y Gtalk v1.0.0.105:** Durante cada ciclo de trabajo el grupo se comunicaba mediante Skype, manteniendo una información global y así determinar las tareas que se habían completado, las que faltaban completar, los distintos problemas que se tuvieron y la manera en que se solucionaron o el debate de cómo solucionarlos. Igualmente durante toda

la práctica se usaron Skype y Gtalk para la comunicación entre los distintos integrantes del equipo, o como alternativa a mantis para que se pueda indicar a un integrante que tarea debía realizar.

2. **Dropbox v2.10.52:** Es un servicio de alojamiento de archivos multiplataforma en la nube. El servicio permite a los usuarios almacenar y sincronizar archivos en línea y entre ordenadores y otros equipos, y compartir archivos y carpetas con otros usuarios. Dropbox es un software que enlaza todas las computadoras mediante una sola carpeta en la nube (red de servidores), lo cual constituye una manera fácil de respaldar y sincronizar los archivos, esto permite que un archivo subido a Dropbox esté disponible al cabo de pocos minutos al resto de dispositivos sincronizados.

   En la práctica se utilizó para resguardar los archivos en una carpeta en común para todos los integrantes del equipo; también cuando existía la necesidad de pasarlos de un miembro del equipo a otro.

3. **Navegador Web (Generalmente Google Croome v39.0.2171 y Mozilla Firefox v34, entre otros para verificar compatibilidad del sistema con cada uno de los navegadores Web más usados):** Un navegador o navegador Web, o browser, es un software que permite el acceso a Internet, interpretando la información de archivos y sitios Web para que éstos puedan ser leídos.

   En la práctica se utilizó para visualizar las páginas Web del sistema que se está desarrollando como también los recursos multimedia y documentos escrutados en éste.

Capítulo 6: Nociones aprendidas durante la práctica

Durante esta práctica tuve la posibilidad de participar en la concreción del Nuevo Sistema de Seguridad y el de la Gerencia de Planificación y Adjudicación, interviniendo en cada una de las etapas del ciclo de vida.

Así, fue necesario realizar un trabajo en equipo donde cada integrante debía realizar su tarea y coordinarla con la de sus colaboradores, empleando una metodología ágil de gestión de desarrollo de software como SCRUM.

En la implementación del Nuevo Sistema fue necesario utilizar distintos programas, componentes y tecnologías. Algunos programas eran conocidos y se había tenido contacto durante el cursado de diferentes materias de la carrera de Ingeniería en Sistemas, mientras que otros eran totalmente nuevos, por lo que fue necesario la lectura y análisis de la documentación pertinente para poder avanzar en su aplicación. Entre éstos últimos, se encuentra la herramienta de Mantis utilizada para la gestión y seguimiento de las distintas tareas; el uso de Git como repositorio distribuido común, en donde ir subiendo/resincronizando todas las modificaciones que los distintos desarrolladores íbamos realizando, entre otras tantas.

Adicionalmente, fue de particular importancia el empleo de distintos programas de comunicación como Gtalk y Skype para resolver diferentes dudas entre los integrantes, mantener reuniones, indicar alguna sugerencia o efectuar un aviso relacionado al proyecto gestionado. Por otro lado, el empleo del software Dropbox permitió mantener un repositorio común entre los integrantes del equipo, resguardar e intercambiar distintos tipos de archivos.

Pude ampliar los conocimientos recibidos durante el cursado de distintas materias en la Facultad, a través de la lectura de bibliografía adecuada y el desarrollo de una práctica más intensiva sobre el uso de los servicios Web; en este proyecto en particular los servicios Web que se utilizaron fueron RESTFul.

Conclusión

Teniendo en cuenta que el IPAV tenía procesos de adjudicaciones de viviendas, irregularidades, permutas y demás actividades, pertenecientes al área de planificación y adjudicación, formalizados, pero aún restaba su automatización y que los sistemas actuales no eran interoperables ya que se manejaban como compartimientos estancos, lo que dificultaba la integración de los componentes y el intercambio de datos; se conforma un equipo de trabajo, del cual formo parte, para desarrollar un nuevo sistema integrado que cumpla con la posibilidad de administrar y monitorear la actividad del IPAV mediante un único sistema.

En este proyecto, y dada la misión del organismo, se prioriza inicialmente el sistema de Seguridad, para posteriormente avanzar sobre el módulo de Inscripciones de la Gerencia de Planificación y Adjudicaciones.

El sistema de inscripciones mencionado abarca todos los procesos de la Gerencia de Planificación y Adjudicación, es decir: la gestión y monitoreo de inscripciones, irregularidades, impugnaciones, sorteo, pre-adjudicaciones, cancelaciones, permutas y renuncias.

En los sistemas que se están desarrollando se usan servicios Web RESTFul. REST define un estilo de arquitectura para desarrollar servicios Web haciendo foco en los recursos del sistema, incluyendo como se accede al estado de dichos recursos.

El esquema de trabajo utilizado para el desarrollo del sistema informático está basado en el trabajo en célula, empleando SCRUM como esquema de gestión con ciclos cortos de dos semanas. Cada ciclo contiene los siguientes tipos de tareas dependiendo la etapa del ciclo de vida en la que nos encontremos. Por ejemplo, durante la implementación pueden apreciarse al menos las siguientes actividades:

- Programación de la capa lógica y de los servicios Web.
- Diseño y desarrollo de los diferentes reportes del sistema.
- Definición de los permisos necesarios por rol.
- Tareas de testing de funciones y servicios Web.
- Programación de los diferentes prototipos de pantallas.
- Desarrollo de las funcionalidades de las pantallas.
- Testing de las pantallas.
- Documentación del sistema.

A partir de todas las tareas realizadas por el equipo de trabajo y el desarrollo del nuevo sistema de Seguridad e Inscripciones, fue posible mejorar la operatoria de la Gerencia de Planificación y Adjudicación, y de ésta manera optimizar la calidad de los servicios brindados al ciudadano; obteniendo un mayor control de los datos y de las acciones a efectuar por los distintos empleados del IPAV.

En lo que respecta a mi desempeño dentro del equipo de trabajo conformado, considero que para el desarrollo del mismo fue fundamental el marco teórico y práctico adquirido durante las materias cursadas en la carrera de Ingeniería en Sistemas, ya que al estar las mismas, mayoritariamente orientadas al desarrollo de software, me permitieron obtener conocimientos sobre aspectos de análisis, documentación y desarrollo de sistemas, entre otros, los que fueron utilizados en el proyecto. Además de estos, intervinieron otros factores de la vida cotidiana que presentaron situaciones problemáticas ante las cuales tuve que buscar solución. Finalmente, todo ello contribuyó a consolidar mi formación e incrementar mi experiencia, ya que debía organizar mi tiempo de trabajo, realizar las modificaciones pertinentes luego de las diversas verificaciones, y trabajar en equipo, lo que implicó cumplir con la tarea en tiempo y forma en post del beneficio grupal y el objetivo común.

En relación a las herramientas utilizadas, las mismas fueron seleccionadas de acuerdo a las tareas que se debían realizar. La utilidad del software Mantis fue muy importante, ya que se usó para gestionar las tareas del proyecto y derivarlas de un integrante del equipo a otro. Por otro lado, el uso de Git sirvió para mantener un repositorio en común y permitir a los desarrolladores subir los diferentes cambios del sistema desarrollado.

Adicionalmente, fue significativo el uso de diferentes medios de comunicación entre los integrantes del equipo como Gtalk y Skype, cuando la comunicación directa no era posible, permitiendo desarrollar la experiencia real de una célula de desarrolla físicamente distribuida.

El sistema de base de datos seleccionado fue PostgreSQL. El mismo es responsable del sustento de la operatoria transaccional, tanto para las bases de datos de prueba como las de producción.

En lo que respecta al uso de las librerías Jfrereport, Itext, Poi y componentes de pentaho son tecnologías que no había utilizado y por lo tanto, tuve que aprenderlas desde cero para poder efectuar la emisión de diferentes reportes y listados en formatos PDF o Excel en las distintas pantallas desarrolladas. De esta manera, se hizo posible previsualizar e imprimir los datos de muchas

las operaciones que se efectúan en el IPAV, tales como los comprobantes de inscripción, comprobantes de adjudicación, entre otros.

Por otro lado, el uso de la librería JAXB implicó conocimientos nuevos, fue empleada en los servicios Web para el envío y recepción de mensajes en formato XML.

El uso de la librería Primefaces en conjunto con JSF y JSTL nos permitió obtener más componentes al momento de diseñar y desarrollar las pantallas del sistema, sumándose a los ya aportados por HTML.

El uso de NetBeans como un IDE para el desarrollo y programación del software y el servidor de aplicaciones integrado Tomcat para probarlas, son herramientas que ya había utilizado en prácticas realizadas en la Facultad, pero fui aprendiendo nuevos funciones de las mismas. Además, como el sistema es orientado a la Web, se usaron diferentes navegadores para verificar la accesibilidad y compatibilidad del mismo con estos.

La utilización de servicios Web RESTFul fue muy importante ya que permitió separar lo que visualiza el cliente con las tareas del servidor. El uso del marco JUnit para semi-automatizar los testings de las funciones de los distintos servicios Web, mediante diferentes entradas de datos, fue primordial para que al momento de prototipar las pantallas e implementar las funciones de la misma, no surgieran errores derivados de los servicios Web, lo que significa una mejora en el tiempo de implementación del sistema.

Otro aspecto importante es que el encargado de programar el servicio Web no era el mismo que el que lo testeaba, y así este último, además de indicarle los errores encontrados mediante las plantillas de testings, podía señalar alguna observación o mejora del servicio Web implementado. De este modo, se empleó la prueba de pares cumpliendo con el control por oposición.

La creación de los casos de usos y la utilización de los procesos SPEM fue importante para entender el funcionamiento de las actividades del organismo, y guiar así el análisis y diseño del sistema implementado. Por otro lado, el desarrollo de los mapas de navegación permitió aportar una perspectiva visual del dinamismo entre las distintas pantallas y su grado de acoplamiento.

Teniendo en cuenta lo expuesto, puedo mencionar que fui partícipe de las diferentes etapas del desarrollo del software, desde el relevamiento de requerimientos con las diferentes entrevistas a los empleados del IPAV, el análisis y diseño del sistema, hasta su implementación y prueba. Por lo tanto considero que el mismo, además de permitir una importante mejora en las distintas

operaciones de la Gerencia, me brindó la oportunidad de ampliar el campo de conocimientos adquiridos en la Facultad y poner en práctica la mayoría de ellos.

Bibliografía de consulta

1. IPAV (2014) "Instituto Provincial Autárquico de Vivienda" Fuente: http://IPAV.lapampa.gov.ar [Último Acceso: 2 de septiembre 9:09AM].

2. Erl, T (2007) SOA: Principles of Service Design. Prentice Hall.

3. Erl, T, Roy, S, Thomas, P & Tost, a (2014) SOA with Java: RealizingSerive-Orientation with Java Technologies. Prentice-Hall/Pearson PTR.

4. IBM (2014) "RESTFul Web Services: The basics". Fuente: https://www.ibm.com/developerworks/Webservices/library/ws-RESTful/ [Último Acceso: 26 de septiembre 15:20AM].

5. Richardson, L & Ruby, S. (2007) "RESTful Web Services. Web services for the real world". O'Reilly Media.

6. Lessard, C. & Lessard, J. (2007) "Project Management for Engineering Design". Morgan and Claypool Publishers.

7. Object Management Group (2008) "Software & Systems Process Engineering Meta-Model Specification".

8. Rumbaugh, J, Jacobson, I & Booch, G (2000) "El lenguaje unificado de Modelado. Manual de Referencia". Addison-Wesley.

9. PostgreSQL Global Development Group (2011) "PostgreSQL 9.0 Official Documentation - Volume II. Server Administration". Fultus Corporation.

10. Rubin, K (2012) "Essential SCRUM: A Practical Guide to the Most Popular Agile Process". Addison-Wesley

11. Sutherland, J & Schwaber, K (2013) "SCRUM Guide". SCRUM.org.

12. Kretsis, A., Kokkinos, P., Christodoulopoulos, K. & Varvarigos, E. (2013) "Mantis: Optical network planning and operation tool". In proceedings of 15th International Conference on Transparent Optical Networks (ICTON). IEEE. Pp. 1-4.

13. Cheng-hui, H. & Huo Yan, C. (2005) "A semi-automatic generator for unit testing code files based on JUnit ". In proceedings of IEEE International Conference on Systems, Man and Cybernetics. Vol.1 pp. 140-145.

14. Kurniawan, B (2014) "Java 7: A Comprehnsive Tutorial". BrainySoftware.

15. Hao, J., Lo, A., Ozyer, T. & Alhajj, R (2005) "XML views based approach for Web services". In proceeding of International Conference on Information Reuse and Integration (IRI). IEEE. Pp. 458-463.

16. Durán, F., Gutiérrez, F. & Pimentel, E. (2007) "Programación orientada a objetos con Java". Paraninfo.

17. Burns, E. & Schalk, C. (2010) "JavaServer Faces 2.0, The Complete Reference". McGraw Hill Professional.

18. Civici, C (2014) "Primefaces User's Guide 5.0". PrimeTek Informatics. Primera Edición.

19. Bai, Y (2011) "Practical Database Programming with Java". IEEE Press & Wiley

20. Gorman, W (2009) "Pentaho Reporting 3.5 for Java Developers". Packt Publishing.

21. Khare, T. (2012) "Apache Tomcat 7 Essentials". Packt Publishing.

22. OpenSuse (2011) "Git". Fuente: https://en.opensuse.org/Git [Último Acceso: 29 de octubre 18:20AM].

23. Lowagie, B. (2007) "IText in Action: Creating and Manipulating PDF". Manning.

24. Geary, D (2002) “Core JSTL: Mastering the JSP Standard Tag Library”. Prentice-Hall.