

Universidad Empresarial Siglo 21



Sistema Web para la Administración Apícola

“BeeMore”

Trabajo Final de Graduación

Ingeniería en Sistemas de Información

Autor:

Carreño, Ignacio Luciano

SIS064

Noviembre 2012



## **Resumen**

En el presente trabajo se analizaron las reglas de negocio y los procesos involucrados al entorno apícola, basado en el análisis y observación del desarrollo de la actividad de dos apicultores.

Se estudiaron los procesos de negocio y se detectaron problemas relacionados con el manejo de la información. El diagnóstico concreto fue el faltante de una herramienta que permitiese llevar un registro ordenado de los datos generados en el proceso productivo de la mencionada actividad.

Utilizando una metodología de desarrollo de sistemas basada en RUP y UML, se diseñó y desarrolló un sistema web de administración, destinado a quienes necesiten administrar colmenas, apiarios, stock de herramientas, cosechas, niveles de producción, envasado de producto final, entre otros, para luego generar reportes estadísticos que ayuden a tomar mejores decisiones, no solo para estos apicultores, sino también para cualquier otro con necesidades similares.

## **Abstract**

In this paper we analyzed the business rules and processes involved in beekeeping environment, based on the analysis and observation of the activity development of two beekeepers.

We studied the business processes and related problems were detected with the management of information. The specific diagnosis was the inexistence of a tool that would allow orderly track data generated in the production process of beekeeping activity.

Using a systems development methodology based on RUP and UML, we designed and developed a web management system, for those who need to manage hives, apiary, stock tools, crops, production levels, final product packaging, among others, then generate statistical reports that helps taking better decisions, not only for these beekeepers, but also for those with similar needs.

## Índice

Índice de tabla de ilustraciones y gráficos .....	7
Capítulo 1, Sistema web para la administración apícola "BeeMore" .....	10
C1.1 Introducción.....	10
C1.2 Objetivos.....	11
C1.3 Límites y alcance .....	12
Capítulo 2, Marco teórico de referencia .....	14
C2.1 Disciplina de estudio .....	14
C2.2 Modelo de negocios.....	18
C2.3 Metodologías de desarrollo de software .....	19
C2.4 Tecnologías para el desarrollo web .....	32
Capítulo 3, Metodología de trabajo .....	38
C3.1 Procedimientos metodológicos para el desarrollo del proyecto .....	38
C3.2 Procedimientos metodológicos para el desarrollo del producto.....	39
Capítulo 4, Desarrollo del trabajo.....	41
C4.1 Relevamiento .....	41
C4.2 Diagnóstico.....	51
C4.3 Propuesta .....	53
Capítulo 5, Implementación.....	55
C5.1 Listado de requerimientos .....	55
C5.2 Diagramas de casos de uso y prototipos de interfaces.....	57

C5.3 Diagramas de secuencia.....	77
C5.4 Diagrama de clases .....	78
C5.5 Diagrama de entidad-relación.....	79
Conclusión .....	80
Bibliografía .....	82
Anexo 1 .....	84
A1.1 Ejemplos de gráficos estadísticos generados por BeeMore .....	84
A1.2 La apicultura.....	86
A1.3 Glosario apícola.....	87
A1.4 Productos derivados de la apicultura.....	88
A1.5 La colmena .....	90
A1.5 Población de una colmena .....	91
A1.7 Enfermedades de las abejas .....	92
A1.8 Patrones de desarrollo de software .....	95
A1.9 Historia de lenguaje PHP .....	105
A1.10 Herramientas utilizadas para el desarrollo del sistema: .....	107

## Índice de tabla de ilustraciones y gráficos

Diagrama de caso de uso 1 - Login.....	58
Diagrama de caso de uso 2 - Registrarse .....	59
Diagrama de caso de uso 3 - Añadir registro de ubicación.....	60
Diagrama de caso de uso 4 - Crear registro de apiario .....	61
Diagrama de caso de uso 5 - Añadir registro de colmena.....	62
Diagrama de caso de uso 6 - Añadir registro de reina .....	63
Diagrama de caso de uso 7 - Ver información de registro de reina.....	64
Diagrama de caso de uso 8 - Añadir registro de cosecha .....	65
Diagrama de caso de uso 9 - Añadir registro de empresa.....	66
Diagrama de caso de uso 10 - Añadir registro de herramienta .....	67
Diagrama de caso de uso 11 - Añadir registro de visita .....	68
Diagrama de caso de uso 12 - Ver estadísticas .....	69
Diagrama de caso de uso 13 - Administrar permisos .....	70
Diagrama de secuencia 1 - Registrarse .....	77
Imagen 1- Diagrama de conjuntos de elementos apícolas.....	15
Imagen 2 - Organigrama funcional de ambos clientes.....	42
Imagen 3 - Producción por ubicación .....	84
Imagen 4 - Rendimiento de apiario por ubicación.....	84
Imagen 5 - Relación producción-edad de reina p/colmena.....	85
Imagen 6 - Rendimiento por colmena.....	85
Imagen 7 - Esquema concreto de metapatrón mvc .....	99
Imagen 8 - Esquema abstracto de metapatrón mvc .....	100
Imagen 9 - Patrón dao .....	100
Imagen 10 - Patrón factory method .....	102

Imagen 11 - Aspecto estático de abstract factory .....	103
Imagen 12 - Esquema de patrón dao factory .....	105
Prototipo de interface 1 - Login .....	58
Prototipo de interface 2 - Registrarse.....	59
Prototipo de interface 3 - Añadir registro de ubicación.....	60
Prototipo de interface 4 - Crear registro de apiario.....	61
Prototipo de interface 5 - Añadir registro de colmena.....	62
Prototipo de interface 6 - Añadir registro de reina .....	63
Prototipo de interface 7 - Ver información de registro de reina .....	64
Prototipo de interface 8 - Añadir registro de cosecha.....	65
Prototipo de interface 9 - Añadir registro de empresa .....	66
Prototipo de interface 10 - Añadir registro de herramienta .....	67
Prototipo de interface 11 - Añadir registro de visita.....	68
Prototipo de interface 12 - Ver estadísticas .....	69
Prototipo de interface 13 - Administrar permisos.....	70
Prototipo de interface 14 - Añadir grupo .....	71
Prototipo de interface 15 - Añadir estado de apiario .....	71
Prototipo de interface 16 - Añadir tipo de visita.....	71
Prototipo de interface 17 - Añadir registro de enfermedad.....	72
Prototipo de interface 18 - Añadir tipo de cosecha.....	72
Prototipo de interface 19 - Añadir estado de colmena.....	73
Prototipo de interface 20 - Añadir tipo de colmena.....	73
Prototipo de interface 21 - Añadir registro de medicamento.....	74
Prototipo de interface 22 - Añadir registro de fuente de nectar.....	75
Prototipo de interface 23 - Añadir tipo de embalaje.....	75



Prototipo de interface 24 - Añadir taxonomía de salud .....	76
Prototipo de interface 25 - Añadir tratamiento .....	76
Tabla 1 - Información detallada de cliente (Carlos Cantarutti).....	41
Tabla 2 - Información detallada de cliente (Elio Cantarutti) .....	41

## **Capítulo 1, Sistema web para la administración apícola "BeeMore"**

### *C1.1 Introducción*

En el año 2007, con el objetivo de realizar el trabajo de seminario de práctica de la Universidad Empresarial Siglo 21, el autor propuso como idea desarrollar un sistema para la administración apícola, habiendo detectado lo difícil que es para el apicultor llevar un registro ordenado de todo su universo y la falencia de una herramienta que posibilitara realizar dicha tarea. El trabajo fue acotado a ciertas áreas de la cadena productiva y ajustado a las necesidades de un solo cliente, concretándose con éxito en el año 2010.

El proyecto actual será realizado en base a 2 usuarios finales, Carlos Cantarutti y Elio Cantarutti. También se recabará información de apicultores allegados a los clientes y se dispondrá de información brindada por profesionales de la Universidad Nacional de Río Cuarto, dedicados a la capacitación de apicultores.

Carlos Cantarutti hace más de 20 años que se dedica a esta actividad y considera la apicultura como una forma de vida, aparte de ser un ingreso importante de dinero. Actualmente cuenta con 6 apiarios y maneja un volumen de producción de aproximadamente 7000kg de miel por año.

Elio Cantarutti inicia su actividad como apicultor en el año 1978 y cuenta con el título de "Perito Apícola" desempeñando tareas de inspector sanitario para el SENASA<sup>1</sup> desde el año 2000. Posee 10 apiarios y un volumen de producción de 13000kg de miel por año. Para Elio la apicultura es su principal ingreso monetario y dedica todo su tiempo y esfuerzo a esta actividad. Ambos clientes concordaron en que el sistema anterior es poco práctico y no cubre todas las necesidades de administración que necesitan.

---

<sup>1</sup> Organismo responsable de garantizar y certificar la sanidad y calidad de la producción agropecuaria pesquera y forestal.

Dicho lo anterior, el siguiente trabajo propone el desarrollo de un sistema web para la administración apícola, tomando como referencia las necesidades de ambos clientes, pero pensado para el resto de los pequeños, medianos y grandes apicultores, para que los mismos puedan finalmente tomar mejores decisiones de su negocio.

## *C1.2 Objetivos*

### *Objetivo general*

Analizar y desarrollar un sistema web de administración apícola que ayude a la organización, planificación y control de los recursos propios de esta disciplina.

### *Objetivos específicos*

- Comprender las políticas, normas, operaciones, definiciones y restricciones propias de la apicultura
- Identificar todos los procesos de negocio en los cuales intervienen los clientes para luego diagramarlos y tener una visión documentada del negocio
- Determinar las problemáticas relacionadas al manejo de la información dentro de los procesos de negocio
- Recopilar, analizar y verificar las necesidades de los clientes, dentro de los límites del proyecto, para poder generar un listado completo y correcto de requisitos de software.
- Diseñar y modelar el sistema utilizando diagramas de UML, los mismos conformarán la base para el desarrollo del producto
- Construir y testear el producto, teniendo en cuenta que el mismo deberá satisfacer las necesidades detectadas y cumplir con los requerimientos recopilados anteriormente

### *C1.3 Límites y alcance*

#### *Límite del proyecto*

El límite inicial de este proyecto comienza con la detección de la necesidad de desarrollar el sistema, y finaliza con la entrega de la primera versión funcional del mismo.

#### *Alcance del proyecto*

- **Planificación:** Definir todos los trabajos necesarios para la realización del proyecto, para luego estimar tiempos y responsables de cada uno de ellos.
- **Análisis:** Determinar todas las necesidades a satisfacer del sistema, relevar la información actual y proponer los rasgos generales de la solución futura. En este punto también se hará hincapié en la tecnología que se utilizará para el desarrollo del sistema.
- **Diseño:** Formalizar toda la información procesada en el análisis en diagramas que definan la estructura del sistema, (casos de uso, diagramas de interfaz, entre otros) y si es necesario explicar su comportamiento dinámico (diagramas de secuencia).
- **Construcción:** Preparar el ambiente de desarrollo para luego codificar la primera versión funcional del sistema.
- **Pruebas:** Realizar un plan de testing<sup>2</sup> acotado a pruebas unitarias que permita encontrar errores que limiten el flujo normal de los casos de uso.

#### *Límites y alcance del sistema*

El sistema abarca desde los proceso de autenticación y validación de usuarios hasta los procesos de emisión de reportes e informes gráficos que den soporte a la toma de decisiones, utilizando toda la información que se genera en el circuito de producción de miel de los clientes, desde la generación de datos para identificar las colmenas hasta el envasado

---

<sup>2</sup> Procedimiento de analizar un programa en busca de problemas y errores (Brown, 1998)

del producto final y su trazabilidad, incluyendo aspectos básicos de gestión contable y financiera de la organización

## Capítulo 2, Marco teórico de referencia

### *C2.1 Disciplina de estudio*

#### *La apicultura*

Del lat. *apis*, abeja, y *-cultura*. Arte de criar abejas para aprovechar sus productos (RAE, 2010). La apicultura es una actividad desarrollada por el hombre con el fin de criar abejas, proporcionando un medio apto y controlado, para aprovechar la producción excedente de las mismas. Este medio apto y controlado acompañado de personas, herramientas y conocimiento será denominado Universo Apícola. Finalmente un apicultor es la persona que realiza esta actividad.

A continuación se explicarán sintéticamente los aspectos más importantes de la apicultura, que ayudarán al lector a comprender el desarrollo del trabajo. Para más información consultar el anexo 1, en la sección *La apicultura*.

El elemento inicial es la abeja europea, son insectos del orden de los himenópteros, pertenecientes al género *Apis* y especie *mellifera*. Viven en grandes sociedades llamadas colonias, en la apicultura moderna estas colonias son introducidas en cajones de madera construidas por el hombre llamadas colmenas, ello permite criar las abejas de manera organizada para el beneficio del mismo.

En la naturaleza las abejas construyen panales de cera<sup>3</sup> al resguardo de las inclemencias climáticas, generalmente debajo de árboles o cuevas. El apicultor, en cambio, construye colmenas artificiales y las agrupa estratégicamente cerca de fuentes de agua y néctar<sup>4</sup>. Este conjunto de colmenas es denominado apiario. (Enzenhofer, 2003)

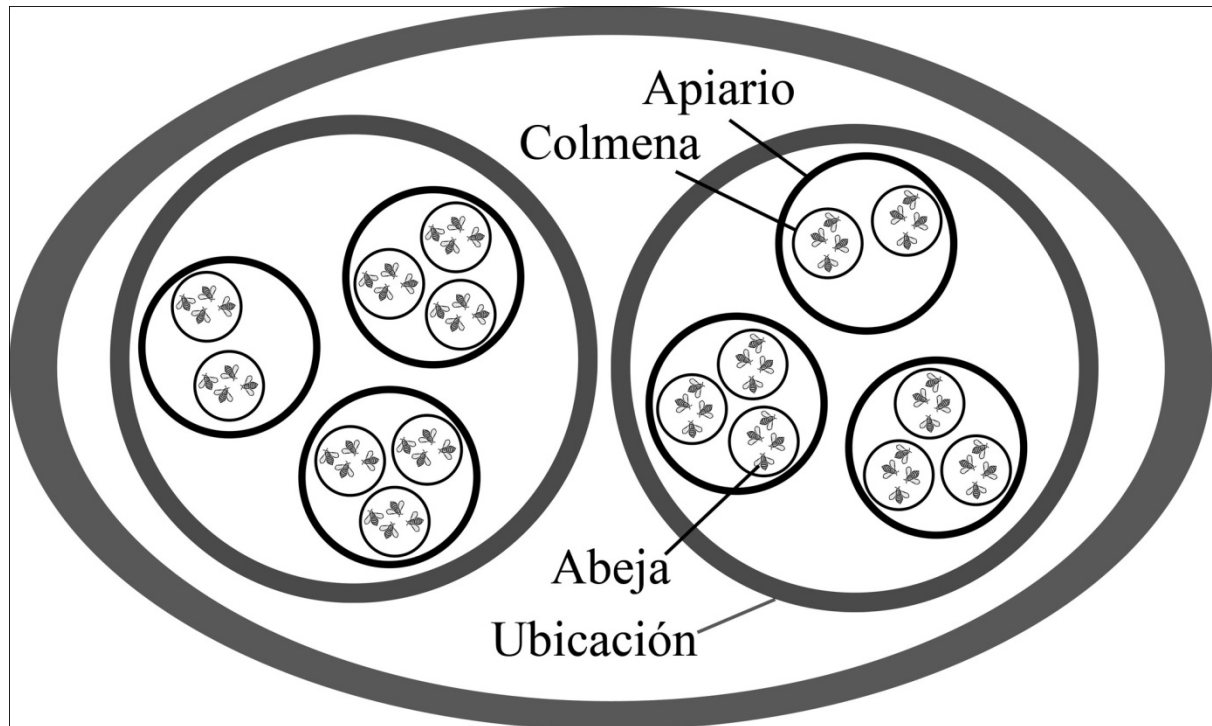
---

<sup>3</sup> Sustancia sólida, blanda, amarillenta y fundible que segregan las abejas para formar las celdillas de los panales y que se emplea principalmente para hacer velas. (RAE, 2010)

<sup>4</sup> Solución azucarada segregada por distintos vegetales (Root, 1982)

Finalmente, a un conjunto de apiarios se lo ubica geográficamente por país, provincia y ciudad el cual será denominado en el marco de este proyecto como ubicación.

La siguiente ilustración sintetiza la explicación anterior



*Imagen 1- Diagrama de conjuntos de elementos apícolas (Carreño I. 2012)*

Dentro de la colonia se observan tres categorías de individuos: reina, obreras y zánganos. Cada colonia de abejas debe tener por lo menos una reina para mantener la coherencia de sus individuos.

La abeja reina es una hembra que ha desarrollado todos sus órganos sexuales y su tarea más importante es poner huevos. Después de cinco días de vida, la reina virgen alcanza la madurez sexual y sale de la colmena para hacer su vuelo de fecundación. Al volar encuentra y se aparea con varios zánganos, o machos. Estos dejan su semen dentro de ella en un órgano denominado espermateca, en el cual puede almacenar suficientes espermatozoides para el resto de su vida. El apareamiento dura una semana, la reina puede salir dos o tres veces de la colmena para aparearse y luego permanecerá de por vida en la misma.

Después de hacer sus vuelos de fecundación y transcurrida una semana empieza a poner huevos todos los días del año. Si el huevo es fecundado por uno de los espermatozoides que guarda en su espermateca nacerá una obrera, de lo contrario un zángano. Los zánganos son los machos de la colonia, su tarea es fecundar a la abeja reina virgen y aquellos que lo logran mueren, asegurando no caer en una consanguinidad. Los zánganos están incapacitados para recoger néctar de las flores y carecen de aguijón.

La abeja obrera, al igual que la reina, es hembra, pero no se ha desarrollado para la reproducción. En casos muy especiales y cuando falta la reina, sus ovarios se desarrollan y consiguen poner huevos, pero al no ser fecundados, nacerán solamente zánganos. La abeja obrera, sin embargo, posee otros órganos que no se encuentran ni en la reina ni en los zánganos, que le permiten realizar las innumerables tareas relacionadas con la vida de la colonia (Enzenhofer, 2003).

Cuando la colonia tiene una abeja reina en buenas condiciones, las abejas obreras son laboriosas, pero si la reina tiene problemas físicos que la limitan o impide su postura o bien es de avanzada edad para transmitir los mensajeros químicos que mantienen a la colonia organizada, las abejas se alteran y, si es necesario, la eliminan para fecundar una nueva reina. (Root, 1982)

### *Productos derivados de la apicultura*

Todos los productos Originarios de las abejas tienen un beneficio económico. A continuación se detalla cada uno:

- Miel: Es una sustancia azucarada que las abejas producen a partir del néctar que recogen de las flores. Es el alimento básico de las abejas y a través de él adquieren energía necesaria para desarrollar todas las actividades de la colonia gracias su alto contenido en azúcares y calorías.



- Cera: Es un producto que producen las abejas a través de las glándulas cereras. La utilizan para construir los panales sobre los cuales la abeja reina depositará los huevos y las abejas almacenarán la miel y el polen. También la ocupan para sellar las celdillas que tienen larvas hasta el momento de nacer. La materia prima para producir cera es la miel, y las abejas necesitan consumir de 6 a 7 kg de miel para producir 1 kg de cera. El hombre utiliza la cera para hacer velas, aceites y artesanías en general.
- Jalea Real: Consiste en una sustancia que las abejas jóvenes segregan para alimentar a las larvas durante sus 3 primeros días y a la abeja reina durante toda su vida. Las materias primas necesarias para su elaboración son el polen, la miel y el agua, las cuales al ser consumidas por las abejas se transforman en jalea real por la acción de glándulas especiales.
- Propóleo: Es una resina que las abejas recogen de algunos árboles. El propóleo es un producto muy importante para la colmena, ya que a través de él mantienen el calor y la higiene. En algunos países se utilizan los extractos de propóleo en el campo de la medicina como cicatrizante, bactericida y fungicida.
- Polen: Es el elemento masculino de una flor por ende no es un producto elaborado por las abejas. El polen es de suma importancia para el crecimiento y la reproducción de la colonia siendo rico en proteínas, lípidos, vitaminas y minerales.
- Toxina: La toxina es producida por el propio cuerpo de la abeja obrera y lo utiliza exclusivamente como arma de defensa contra animales, insectos y todo aquello que amenaza el funcionamiento de la colonia. Se utiliza en la medicina alternativa para tratar el reuma y la artritis. (Enzenhofer, 2003).

## *La Colmena*

Una colmena es el lugar donde habita una colonia o familia de abejas. La apicultura moderna se vale de la colmena técnica para producir miel y otros productos. Las colmenas técnicas están compuestas por cuerpos rectangulares y un techo de acero galvanizado para proteger a las abejas de las inclemencias climáticas. Dentro de los cuerpos encontramos bastidores, son marcos de madera fina, con alambres para sujetar una lámina de cera, estas láminas forman la guía del panal que luego las abejas ocuparán con celdas en ambos lados. Un cuerpo rectangular completo por bastidores se denomina alza, y si el mismo posee miel, se denomina alza mielera. Esta es transportada desde un apiario hasta la sala de extracción donde los apicultores separarán la miel de los panales de cera para finalmente envasar. (Enzenhofer, 2003)

### *C.2.2 Modelo de negocios*

#### *El modelo freemium*

El modelo freemium que proporciona servicios básicos libre de costos y servicios premium por un monto de dinero, se ha vuelto cada vez más popular gracias a la digitalización creciente de bienes y servicios ofrecidos a través de la Web.

Representa modelos de negocio, principalmente Web, que mezcla servicios básicos gratuitos con servicios premium pagos. El modelo freemium se caracteriza por un gran número de usuarios free, beneficiado por una mínima cantidad de usuarios premium que paga por un bien o servicio extra.

La mayoría de estos usuarios free nunca se convertirán en clientes; sólo una pequeña porción, generalmente de menos de 10 por ciento, se suscribirá al servicio premium de pago.

Esta pequeña base de usuarios premium subsidia a los usuarios libres. Esto es posible debido al bajo costo marginal de servir a los usuarios free adicionales. En un modelo freemium, las métricas claves para ver son el coste medio de servicio a un usuario libre y el promedio de usuarios free que se convierten a premium.

Flickr, el popular sitio Web de uso compartido de fotos adquirido por Yahoo! en 2005, proporciona un buen ejemplo de modelo de negocio freemium. Los usuarios de Flickr pueden suscribirse gratuitamente a una cuenta básica que les permite subir y compartir imágenes. El servicio gratuito tiene ciertas limitaciones, como el espacio de almacenamiento limitado y un número máximo de envíos por mes. Por una pequeña cuota anual, los usuarios pueden comprar una cuenta "pro" y disfrutar de cargas de imágenes y espacio de almacenamiento ilimitado, además de funciones adicionales. (Osterwalder & Pigneur, 2010)

### *C2.3 Metodologías de desarrollo de software*

#### *UML*

UML por sus siglas en inglés significa Unified Modeling Language (Lenguaje Unificado de Modelado), actualmente es el lenguaje de modelado de sistemas de software más conocido y utilizado y está respaldado por el OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables.

Es importante resaltar que UML es un "lenguaje de modelado" para especificar o para describir métodos o procesos. Se utiliza para definir un sistema, para detallar los artefactos en

el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo. Se puede aplicar en el desarrollo de software entregando gran variedad de formas para dar soporte a una metodología de desarrollo de software (tal como el Proceso Unificado Racional o RUP), pero no especifica en sí mismo qué metodología o proceso usar.

### *Diagrama de clases*

Un diagrama de clases es un tipo de diagrama estático que describe la estructura de un sistema mostrando sus clases, atributos y las relaciones entre ellos. Los diagramas de clases son utilizados durante el proceso de análisis y diseño de los sistemas, donde se crea el diseño conceptual de la información que se manejará en el sistema, y los componentes que se encargaran del funcionamiento y la relación entre uno y otro.

### *Diagramas de casos de uso*

Esta notación gráfica define la naturaleza de un caso de uso; sin embargo una notación gráfica puede solo dar una vista general simple de un caso de uso o un conjunto de casos de uso. Los diagramas de casos de uso son a menudo confundidos con los casos de uso. Mientras los dos conceptos están relacionados, los casos de uso son mucho más detallados que los diagramas de casos de uso.

### *Planilla de casos de uso*

Es una planilla donde se describe de manera clara y concisa la funcionalidad de un caso de uso, indicando también el autor, fecha de creación, entre otras. Las mismas se utilizaran en casos donde sea necesario detallar un caso de uso muy complejo o importante

### *Diagrama de secuencia*

Este diagrama sirve para incluir una importante dimensión: el tiempo. Principalmente las interacciones entre los objetos se realizan en una secuencia establecida y esta secuencia tiene un principio y un fin.

El diagrama de secuencias consta de objetos que se representan del modo usual: rectángulos con nombre (subrayado), mensajes representados por líneas continuas con una punta de flecha y el tiempo representado como una progresión vertical.

Los objetos se colocan cerca de la parte superior del diagrama de izquierda a derecha y se acomodan de manera que simplifiquen al diagrama. La extensión que está debajo (y en forma descendente) de cada objeto será una línea discontinua conocida como la línea de vida de un objeto. Junto con la línea de vida de un objeto se encuentra un pequeño rectángulo conocido como activación, el cual representa la ejecución de una operación que realiza el objeto. La longitud del rectángulo se interpreta como la duración de la activación.

#### *Diagrama o modelo entidad-relación (DER)*

Diagrama o Modelo Entidad-Relación es un concepto de modelado para bases de datos, propuesto por Peter Chen en 1976, mediante el cual se pretende 'visualizar' los objetos que pertenecen a la Base de Datos

En el modelo entidad-relación, el universo de discurso o minimundo se representa por un conjunto de entidades que presentan ciertas propiedades llamadas atributos, definidos sobre un cierto dominio de datos. Las entidades se vinculan mediante relaciones que, en ciertas variantes de la notación, pueden también tener sus propios atributos. En principio, estas relaciones pueden ser n-arias, pero en la práctica se trabaja con relaciones binarias. Por ejemplo, una relación ternaria entre entidades A, B y C puede representarse por una nueva entidad D que tenga relaciones binarias con cada una de A, B y C.

Para cada entidad pueden existir en un momento dado cero, una o muchas instancias. Estas instancias toman valores para sus atributos de los dominios de datos definidos para aquellos. Las instancias de una relación son pares ordenados de instancias de las entidades que dicha relación vincula.

## *RUP*

El Proceso Unificado Racional (Rational Unified Process en inglés, habitualmente resumido como RUP) es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. El RUP no es un sistema con pasos firmemente establecidos, sino un conjunto de metodologías adaptables al contexto y necesidades de cada organización.

### Principios Básicos del desarrollo en RUP

- **Adaptar el proceso:** El proceso deberá adaptarse a las características propias del proyecto u organización. El tamaño del mismo, así como su tipo o las regulaciones que lo condicionen, influirán en su diseño específico. También se deberá tener en cuenta el alcance del proyecto.
- **Equilibrar prioridades:** Los requerimientos de los diversos participantes pueden ser diferentes, contradictorios o disputarse recursos limitados. Debe encontrarse un equilibrio que satisfaga los deseos de todos. Gracias a este equilibrio se podrán corregir desacuerdos que surjan en el futuro.
- **Demostrar valores iterativamente:** Los proyectos se entregan, aunque sea de un modo interno, en etapas iteradas. En cada iteración se analiza la opinión de los inversores, la estabilidad y calidad del producto, y se refina la dirección del proyecto así como también los riesgos involucrados.
- **Colaboración entre equipos:** El desarrollo de software no lo hace una única persona sino múltiples equipos. Debe haber una comunicación fluida para coordinar requerimientos, desarrollo, evaluaciones, planes y resultados.

- Elevar el nivel de abstracción: Este principio dominante motiva el uso de conceptos reutilizables tales como patrón del software, lenguajes 4GL o marcos de referencia (frameworks). Esto evita que los ingenieros de software vayan directamente de los requisitos a la codificación de software a la medida del cliente, sin saber con certeza qué codificar para satisfacer de la mejor manera los requerimientos y sin comenzar desde un principio pensando en la reutilización del código. Un alto nivel de abstracción también permite discusiones sobre diversos niveles y soluciones arquitectónicas. Éstas se pueden acompañar por las representaciones visuales de la arquitectura, por ejemplo con el lenguaje UML
- Ciclo de vida: El ciclo de vida RUP es una implementación del desarrollo en espiral. Fue creado ensamblando los elementos en secuencias semi-ordenadas. El ciclo de vida organiza las tareas en fases e iteraciones.

RUP divide el proceso en cuatro fases, dentro de las cuales se realizan varias iteraciones en número variable según el proyecto y en las que se hace un mayor o menor hincapié en las distintas actividades.

Las primeras iteraciones (en las fases de inicio y elaboración) se enfocan hacia la comprensión del problema y la tecnología, la delimitación del ámbito del proyecto, la eliminación de los riesgos críticos, y el establecimiento de una línea base de la arquitectura.

Durante la fase de inicio las iteraciones hacen mayor énfasis en actividades de modelado del negocio y de requerimientos.

En la fase de elaboración, las iteraciones se orientan al desarrollo de la línea base (baseline) de la arquitectura, abarcan más los flujos de trabajo de requerimientos, modelo de negocios (refinamiento), análisis, diseño y una parte de implementación orientado a la baseline de la arquitectura.

En la fase de construcción, se lleva a cabo la construcción del producto por medio de una serie de iteraciones.

Para cada iteración se selecciona algunos casos de uso, se refina su análisis y diseño y se procede a su implementación y pruebas. Se realiza una pequeña cascada para cada ciclo. Se realizan tantas iteraciones hasta que se termine la implementación de la nueva versión del producto.

En la fase de transición se pretende garantizar que se tiene un producto preparado para su entrega a la comunidad de usuarios.

Como se puede observar en cada fase participan todas las disciplinas.

Principales características:

- Forma disciplinada de asignar tareas y
- Pretende implementar las mejores prácticas en Ingeniería de Software
- Desarrollo iterativo
- Administración de requisitos
- Uso de arquitectura basada en componentes
- Control de cambios
- Modelado visual del software
- Verificación de la calidad del software

Artefactos:

RUP en cada una de sus fases (pertenecientes a la estructura estática) realiza una serie de artefactos que sirven para comprender mejor tanto el análisis como el diseño del sistema (entre otros). Estos artefactos son los siguientes.

Inicio:

- Documento Visión



- Especificación de Requerimientos

Elaboración:

- Diagramas de caso de uso

Construcción:

- Documento arquitectura que trabaja con las siguientes vistas:

Vista lógica:

- Diagrama de clases
- Modelo E-R (Si el sistema así lo requiere)

Vista de implementación:

- Diagrama de secuencia
- Diagrama de estados
- Diagrama de colaboración

Vista conceptual:

- Modelo de dominio

Vista física:

- Mapa de comportamiento a nivel de hardware.

### *Comentarios sobre alcance del RUP*

La metodología RUP es más apropiada para proyectos grandes, dado que requiere un equipo de trabajo capaz de administrar un proceso complejo en varias etapas. En proyectos pequeños, es posible que no se puedan cubrir los costos de dedicación del equipo de profesionales necesarios.

## *Notación de modelado de procesos de negocio*

El objetivo principal de la notación de modelado de procesos de negocio, BPMN por sus siglas en inglés (Business Process Modeling Notation), es proporcionar una notación entendible por todos los usuarios de negocio. Desde el analista de negocios que se encarga de crear los primeros borradores de los procesos, pasando por los desarrolladores a cargo de la implementación hasta los usuarios de negocio que utilizarán y monitorearán dichos procesos.

BPMN soporta un modelo interno que permite la generación de código BPEL4WS (Business Process Execution Language for Web Services) lo cual permite cerrar la brecha entre el diseño de los procesos de negocio y la implementación de los mismos. (White, 2004)

Para este proyecto se va a utilizar la notación BPMN para modelar los procesos de negocio de los clientes.

### Business Process Diagram (BPD)

Un BPD está compuesto por una serie de elementos gráficos. Estos elementos permiten la creación de diagramas simples que pretenden resultar familiares a la mayoría de los analistas de negocios ya que el mismo se encuentra basado en los diagramas de flujo (herramienta muy difundida en la actualidad) (White, 2004).

Cabe mencionar que una de las motivaciones que impulsó el desarrollo del BPMN fue la necesidad de crear una notación que otorgue un mecanismo simple para la creación de modelos de procesos, pero que a su vez pueda manejar las complejidades inherentes a los procesos de negocio. Para manejar esos objetivos tan dispares, se organizó la notación en categorías específicas. De esta manera se provee un conjunto de categorías acotados que permiten que la persona que lee dichos diagramas reconozca fácilmente los tipos de elementos y entienda el diagrama. (Object Management Group, 2008)

Las 4 categorías básicas son:

- Objetos de flujo: Eventos, Actividades, Rombos de control de flujo (Gateways)
- Objetos de conexión: Flujo de Secuencia, Flujo de Mensaje, Asociación
- Swimlanes (Carriles de piscina): Pool, Lane
- Artefactos: Objetos de Datos, Grupo, Anotación

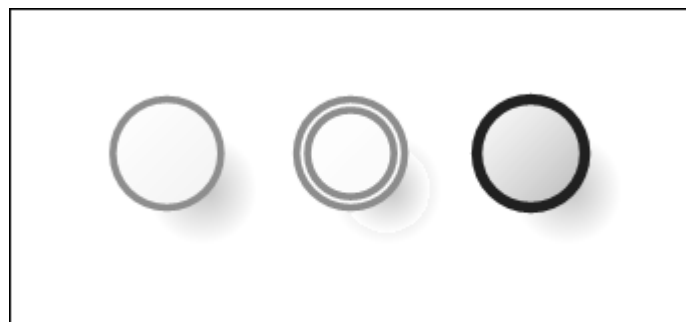
## Objetos de Flujo

Un BPD contiene 3 posibles elementos de flujo:

### Eventos

Un evento es representado por un círculo, es algo que “sucede” en el transcurso de un proceso. Estos eventos afectan el flujo del proceso y usualmente tienen una causa (disparador) o un impacto (resultado).

Los eventos son círculos con sus centros sin relleno, lo cual permite marcadores internos para poder diferenciar distintos disparadores o resultados. Existen 3 tipos de eventos, basados en el momento que afectan el flujo. Son, eventos de inicio, intermedios o de terminación.



*Gráfico 1: BPD - Elemento de flujo (Carreño I. 2012)*

### Actividad

Una actividad está representada por un rectángulo de esquinas redondeadas. Es un término genérico que se refiere al trabajo que se realiza. Una actividad puede ser atómica

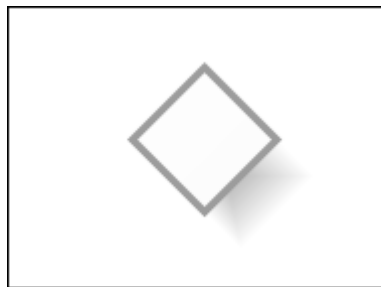
o no atómica (compuesta). Los tipos de actividad son: Tarea o sub-procesos. El sub-proceso se distingue con un pequeño signo más (+) en la parte inferior central de la forma.



*Gráfico 2: BPD - Elemento de flujo - Actividad (Carreño I. 2012)*

### Compuerta (Gateway)

Está representado por un rombo y es utilizada para controlar la divergencia y convergencia de los flujos de secuencia. Por ello, es el elemento a utilizar para identificar los puntos de decisión, así como los puntos de bifurcación y convergencia de caminos. Marcadores internos indican el tipo de comportamiento de control.

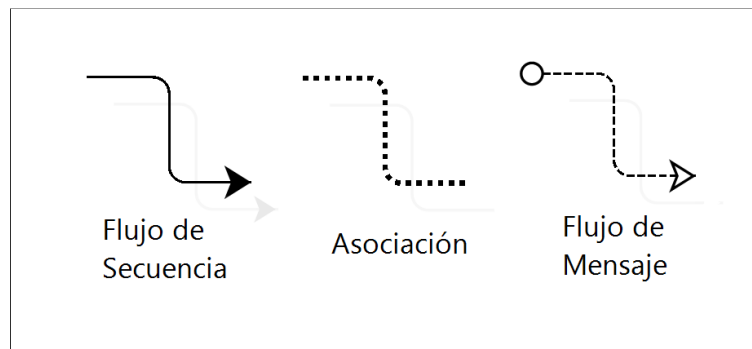


*Gráfico 3: BPD - Elemento de Flujo - Compuerta (Gateway) (Carreño I. 2012)*

### Objetos de conexión

Los objetos de flujo se conectan en el diagrama para crear una estructura básica del proceso de negocio. Existen 3 tipos de objetos de conexión:

- Flujos de secuencia: Está representado por una línea sólida terminada en una flecha. Es utilizada para mostrar el orden en que se realizarán las actividades del proceso.
- Flujos de mensajes: Representado por una línea punteada terminada en una flecha sin relleno, es utilizado para mostrar el flujo de mensajes entre 2 participantes de procesos (ya sean roles de negocio o entidades de negocio). En BPMN, diferentes pools en el diagrama representan a diferentes participantes.
- Asociaciones: Se encuentra representada por una línea de terminada en una flecha sin cerrar. Es utilizada para asociar datos, texto y otros artefactos con los objetos de flujo. Las asociaciones son utilizadas para mostrar las entradas y salidas de las actividades.



*Gráfico 4: BPD - Objetos de conexión (Carreño I. 2012)*

### Swimlanes (carriles de piscina)

Muchas metodologías de modelado de procesos utilizan el concepto de carriles de piscinas para organizar actividades en categorías separadas visualmente para poder ilustrar de manera más clara las diferentes competencias funcionales o responsabilidades. BPMN soporta este concepto con dos tipos de artefactos principales:

## Pool

Representa un participante en el proceso. También funciona como un contenedor gráfico que permite la de partición un conjunto de actividades con respecto a las de otros participantes. Se utilizan en el contexto de situaciones de negocio a negocio.

## Línea

Una línea es una sub-partición dentro de un pool. Se representan ya sea vertical u horizontalmente y permiten organizar y categorizar las actividades del participante.



*Gráfico 5: BPD - Swimlanes - Pool y Línea (Carreño I. 2012)*

Los pools se utilizan cuando el diagrama involucra entidades de negocio o participantes diferentes, y se encuentran físicamente separadas en el diagrama. Las actividades dentro de cada pool son consideradas como procesos auto-contenidos. Por lo tanto un flujo de secuencia no puede cruzar de un pool a otro. Para ello se utilizan los flujos de mensajes como mecanismo para mostrar la comunicación entre los participantes.

Las líneas son utilizadas usualmente para separar las actividades asociadas a una función específica dentro de una organización o de un rol. Los flujos de secuencia pueden

cruzar entre líneas dentro de un pool, pero no se pueden utilizar flujos de mensajes entre objetos en el mismo pool.

### Artefactos

BPMN está diseñado para permitir cierto grado de flexibilidad dentro del modelado, extendiendo la notación básica y proveyendo la habilidad de agregar adecuaciones de acuerdo al contexto. Se pueden agregar la cantidad necesaria de artefactos a un diagrama, para adecuarse de la mejor manera al proceso que se intenta modelar. La versión actual de la especificación de BPMN define 3 tipos de artefactos:

- **Objetos de datos:** Son mecanismos que muestran como ciertos datos son requeridos o producidos por actividades. Se conectan con las actividades a través de asociaciones.
- **Grupos:** Un grupo es representado por un rectángulo de esquinas redondeadas, dibujado con líneas de puntos. El agrupamiento es utilizado para documentación o propósitos de análisis, pero no afecta el flujo de secuencia.
- **Anotaciones:** Son un mecanismo que le permite al modelador proveer información extra para ayudar a la mejor comprensión del diagrama.

## *C2.4 Tecnologías para el desarrollo web*

### *Lenguajes*

#### PHP

PHP es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. Es usado principalmente en interpretación del lado del servidor (server-side scripting) pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica usando las bibliotecas Qt o GTK+.

PHP es un acrónimo recursivo que significa Hypertext Pre-processor (inicialmente PHP Tools, o, Personal Home Page Tools). Fue creado originalmente por Rasmus Lerdorf en 1994; sin embargo la implementación principal de PHP es producida ahora por The PHP Group y sirve como el estándar de facto para PHP al no haber una especificación formal. Publicado bajo la PHP License, la Free Software Foundation considera esta licencia como software libre.

#### JavaScript

JavaScript es un lenguaje de scripting<sup>5</sup> basado en objetos sin tipo y liviano, utilizado para acceder a objetos en aplicaciones. Principalmente, se utiliza integrado en un navegador web permitiendo el desarrollo de interfaces de usuario mejoradas y páginas web dinámicas. JavaScript es un dialecto de ECMAScript<sup>6</sup> y se caracteriza por ser un lenguaje basado en prototipos, con entrada dinámica y con funciones de primera clase.

Todos los navegadores modernos interpretan el código JavaScript integrado dentro de las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del Document Object Model (DOM).

---

<sup>5</sup> Un script (cuya traducción literal es guión) o archivo de órdenes o archivo de procesamiento por lotes es un programa usualmente simple, que por lo regular se almacena en un archivo de texto plano.

<sup>6</sup> ECMAScript es una especificación de lenguaje de programación publicada por ECMA International.



El lenguaje fue inventado por Brendan Eich en la empresa Netscape Communications, la que desarrolló los primeros navegadores web comerciales. Apareció por primera vez en el producto de Netscape llamado Netscape Navigator 2.0.

Tradicionalmente, se venía utilizando en páginas web HTML, para realizar operaciones y en el marco de la aplicación cliente, sin acceso a funciones del servidor. JavaScript se ejecuta en el agente de usuario, al mismo tiempo que las sentencias van descargándose junto con el código HTML.

Inicialmente los autores lo llamaron Mocha y más tarde LiveScript pero fue rebautizado como JavaScript en un anuncio conjunto entre Sun Microsystems y Netscape, el 4 de Diciembre de 1995.

En 1997 los autores propusieron JavaScript para que fuera adoptado como estándar de la European Computer Manufacturers Association ECMA, que a pesar de su nombre no es europeo sino internacional, con sede en Ginebra. En Junio de 1997 fue adoptado como un estándar ECMA, con el nombre de ECMAScript. Poco después también como un estándar ISO. (Vander Verr, 2008)

## AJAX

Si bien no es un lenguaje, se detalla a continuación debido a la correlación con lo explicado anteriormente.

Ajax, acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications). Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, lo que significa aumentar la interactividad, velocidad y usabilidad en las aplicaciones.

Ajax es una tecnología asíncrona, en el sentido de que los datos adicionales se requieren al servidor y se cargan en segundo plano sin interferir con la visualización ni el comportamiento de la página. JavaScript es el lenguaje interpretado (scripting language) en el que normalmente se efectúan las funciones de llamada de Ajax mientras que el acceso a los datos se realiza mediante XMLHttpRequest, objeto disponible en los navegadores actuales. En cualquier caso, no es necesario que el contenido asíncrono esté formateado en XML.

Ajax es una técnica válida para múltiples plataformas y utilizable en muchos sistemas operativos y navegadores, dado que está basado en estándares abiertos como JavaScript y Document Object Model (DOM). (Firtman, 2007)

## Historia

La verdad es que AJAX no es nuevo y antes se conocía con otros nombres pero no fue muy popular hasta que Google, gracias a sus excelentes servicios e interfaz, permitió que se hiciera conocido entre los usuarios y los desarrolladores. Desde internet Explorer 5.0, Microsoft incorporo un objeto conocido como XMLHttpRequest que permitía hacer lo que en ese momento se conoció como Remote Scripting, pero nadie vio el potencial hasta 6 años después. Solo Microsoft lo utilizaba en su Outlook web Access, su versión web incluida con Exchange.

Gracias a que Google implementó casi el 100% de sus servicios con AJAX, este se hizo tan conocido, que todos los navegadores encabezados por Mozilla, implementaron una versión del objeto XMLHttpRequest para que las aplicaciones desarrolladas con el mismo funcionen correctamente.

Hoy en día es AJAX es una técnica de programación, basada en estándares, que no posee dueño y la W3C, trabaja en la estandarización formal de la plataforma (Firtman, 2007)

## SQL

El lenguaje de consulta estructurado o SQL (por sus siglas en inglés structured query language) es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en éstas. Una de sus características es el manejo del álgebra y el cálculo relacional permitiendo efectuar consultas con el fin de recuperar -de una forma sencilla- información de interés de una base de datos, así como también hacer cambios sobre ella. Es un lenguaje informático de cuarta generación (4GL).

### *Frameworks para desarrollo web (PHP y JavaScript)*

#### jQuery

jQuery es un framework de JavaScript, creado inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con AJAX. Fue presentada el 14 de enero de 2006 en el BarCamp NYC.

jQuery es software libre y de código abierto, posee un doble licenciamiento bajo la Licencia MIT y la Licencia Pública General de GNU v2, permitiendo su uso en proyectos libres y privativos. jQuery, al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código, es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio.

(jQuery Foundation, 2012)

#### CakePHP

CakePHP es un framework para el desarrollo de aplicaciones web enteramente escrito en PHP, es un marco de trabajo compuesto por una librería de clases de alta cohesión escritas en PHP. Provee de arquitectura, componentes y herramientas para que los desarrolladores construyan aplicaciones web más rápidamente.

La velocidad de desarrollo, escalabilidad y mantenimiento hacen de este uno de los mejores frameworks del mercado, basado en buenas prácticas y patrones de desarrollo web con muchas librerías “third-party”. Es remarcable su facilidad de configuración, haciendo que su aplicación sea más flexible en entornos restringidos (web servers con limitada capacidad de configuración) (CAKEPHP, 2011)

### *Aplicaciones*

#### XAMPP

XAMPP es un servidor independiente de plataforma, Open Source, que consiste principalmente en la base de datos MySQL, el servidor Web Apache y los intérpretes para lenguajes de script: PHP y Perl. El nombre proviene del acrónimo de X (para cualquiera de los diferentes sistemas operativos), Apache, MySQL, PHP, Perl. El programa está liberado bajo la licencia GNU y actúa como un servidor web libre, fácil de usar y capaz de interpretar páginas dinámicas. Actualmente XAMPP está disponible para Microsoft Windows, GNU/Linux, Solaris, y MacOS X.

XAMPP solamente requiere descargar y ejecutar un archivo zip, tar, o exe, con unas pequeñas configuraciones en algunos de sus componentes que el servidor web necesitará. XAMPP se actualiza regularmente para incorporar las últimas versiones de Apache/MySQL/PHP y Perl. También incluye otros módulos como OpenSSL y phpMyAdmin. Para instalar XAMPP se requiere solamente una pequeña fracción del tiempo necesario para descargar y configurar los programas por separado.

Oficialmente, los diseñadores de XAMPP sólo pretendían su uso como una herramienta de desarrollo, para permitir a los diseñadores de sitios webs y programadores testear su trabajo en sus propios ordenadores sin ningún acceso a Internet. En la práctica, sin embargo, XAMPP es utilizado actualmente para servidor de sitios web y, con algunas

modificaciones, es generalmente lo suficientemente seguro para serlo. Con el paquete se incluye una herramienta especial para proteger fácilmente las partes más importantes.

## Capítulo 3, Metodología de trabajo

### *C3.1 Procedimientos metodológicos para el desarrollo del proyecto*

La primer parte del proceso metodológico consistió en comprender las reglas de negocio apícola. Para ello se seleccionaron importantes obras escritas sobre el tema como *The abc and xyz of bee culture*, *Herramientas de Trabajo para la Apicultura Moderna* y muchas otras de poco peso académico pero de excelente calidad informativa, para luego ser estudiadas y analizadas.

Más tarde se coordinaron reuniones con ambos clientes, primero con Carlos y luego con Elio para hacer una introducción formal de la propuesta y relevar la envergadura del negocio de cada uno.

El paso final y de mayor importancia para el desarrollo del trabajo, fue la observación directa de todas las actividades previas, in situ y posteriores de una visita a un apiario. Esto fue posible gracias a la colaboración de Carlos quien facilitó el equipamiento y transporte para que el autor de este trabajo pudiera asistir y observar las tareas de campo.

La información recabada, junto a la comprensión del entorno apícola de los clientes, fue utilizada para poder detectar todos los procesos de negocio y diagramar los más relevantes utilizando BPNM.

Con los procesos de negocios más importantes analizados y documentados más la observación directa del manejo de información que se efectuaban en los mismos, se elaboró un diagnóstico detallado de cada falencia o problemática de manejo de información, abarcando todas las áreas de las organizaciones de ambos clientes.

El procedimiento no fue tan complejo dado que ambos clientes presentaron problemáticas similares, todas asociadas al manejo de la información y al faltante de una herramienta que les ayude a tomar decisiones. Finalmente se analizaron los resultados de

ambos diagnósticos, uno para Carlos y otro para Elio, y se buscaron puntos en común para concluir en un único listado.

Para cerrar todo este proceso metodológico se elaboró una propuesta concreta, la cual concluyó en el desarrollo de un sistema web para la administración apícola.

### *C.3.2 Procedimientos metodológicos para el desarrollo del producto*

La metodología utilizada para documentar, analizar y diseñar el producto, fue una versión reducida de RUP, basada en la utilización de los diagramas de UML que mejor explicaron el comportamiento esperado del sistema.

En cuanto al ciclo de vida se optó por desarrollo en cascada, ajustándose perfectamente a las necesidades del sistema.

#### *Análisis de requerimientos*

Los mismos fueron el resultado de una extensa indagación a los clientes, con el apoyo de la información obtenida hasta ese momento (relevamiento, diagnóstico, entre otros)

#### *Diagrama de clases*

El mismo fue realizado como posible candidato, en caso de no utilizar ningún framework de desarrollo. Finalmente el sistema se desarrolló utilizando CakePHP, que como todo marco de trabajo nos brinda una estructura de clases predefinida, cuya documentación no tiene lugar en este trabajo.

#### *Diagramas de casos de uso*

Se realizó un extenso diagrama de casos de uso generales y luego se especificaron los casos de uso más importantes

#### *Planilla de casos de uso*

#### *Diagrama de secuencia*

*Diagrama o modelo entidad-relación (DER)*

Se realizó en base a estándares y convenciones de nombres proporcionados por el framework

*Codificación y construcción*

*Pruebas*

*Implementación*



## Capítulo 4, Desarrollo del trabajo

### C4.1 Relevamiento

#### *Relevamiento Estructural*

El siguiente relevamiento tiene como objetivo dar a conocer la estructura organizacional de los clientes y los procesos de negocio más importantes que se desarrollan en la misma.

- Información detallada de los clientes
  - Cliente 1:

Nombre completo	Carlos Cantarutti
Antigüedad como apicultor	26 años
Cantidad de apiarios	6
Cantidad de colmenas	300
Volumen máximo de producción	40 kg x colmena
Volumen mínimo de producción	5 kg x colmena
Cantidad de empleados	1
Dispone de PC/cantidad	Si/1
Dispone de Internet/conexión	Si/Cable módem

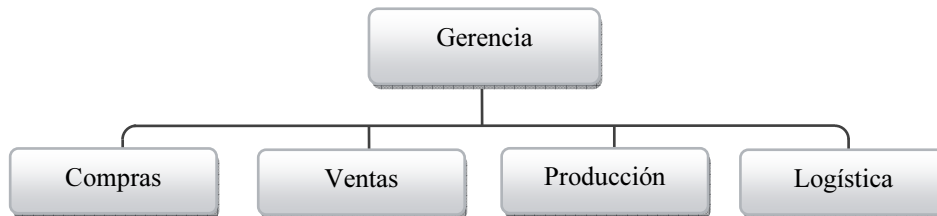
*Tabla 1 - Información detallada de cliente (Carlos Cantarutti)*

- Cliente 2:

Nombre completo	Elio Cantarutti
Antigüedad como apicultor	32 años
Cantidad de apiarios	10
Cantidad de colmenas	645
Volumen máximo de producción	40 kg x colmena
Volumen mínimo de producción	3 kg x colmena
Cantidad de empleados	1
Dispone de PC/cantidad	Si/1
Dispone de Internet/conexión	Si/Cable módem

*Tabla 2 - Información detallada de cliente (Elio Cantarutti)*

Es importante aclarar que la estructura organizacional de ambos clientes es similar y por consiguiente el organigrama de funciones que se presenta a continuación es el que mejor las describe.



*Imagen 2 - Organigrama funcional de ambos clientes (Carreño I. 2012)*

Tanto Carlos como Elio son los gerentes generales de su empresa, ambos cuentan con un empleado quien le sirve de ayuda principalmente en las actividades relacionadas a la producción. Elio recibe más ayuda por parte de algunos integrantes de su familia pero ambos desempeñan los roles de todas las tareas de la organización.

A continuación se listarán los procesos de negocio más significativos relacionados a las funciones del organigrama (Imagen 2)

- Logística
  - Planear visita
  - Planear viajes
  - Control de stock
  - Transportar alzas mieleras
- Compras y ventas
  - Registrar compras
  - Registrar ventas
  - Calcular ganancia neta

- Producción
  - Visitar apiario
  - Realizar tarea programada
    - Recolectar miel
    - Realizar cura de enfermedades
    - Realizar mantenimiento
  - Construir colmenas
  - Popular colmena (núcleo)
  - Dividir colonia
  - Criar abejas reinas
  - Preparar sala de extracción
  - Extraer miel de alzas mieleras para almacenar
  - Envasar miel

Se utilizará BPNM para diagramar y explicar los procesos de negocios más importantes.

## Procesos de negocio

- Visita

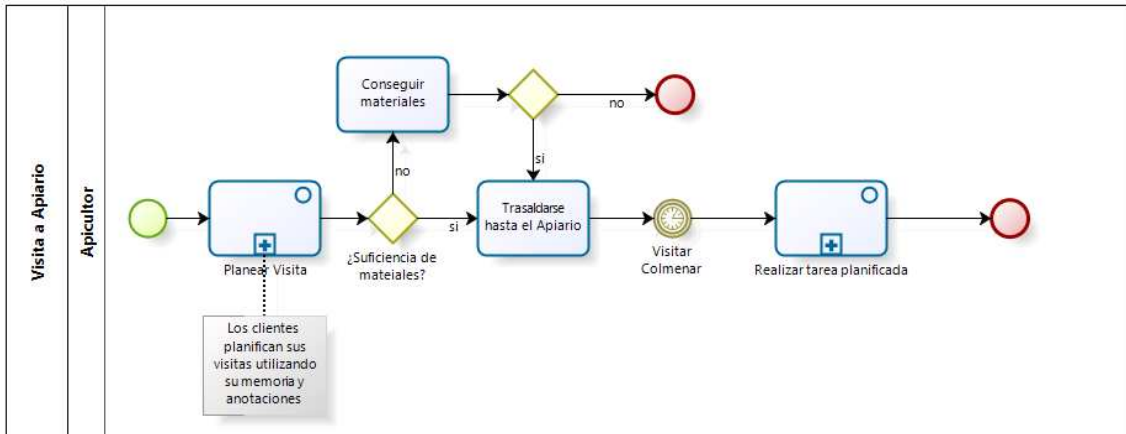


Diagrama 1 - Proceso de negocio visita (Carreño I. 2012)

### Sub procesos de visita

- Planear visita

Si bien este diagrama es sencillo, muestra que se está necesitando de alguna herramienta para planear la visita

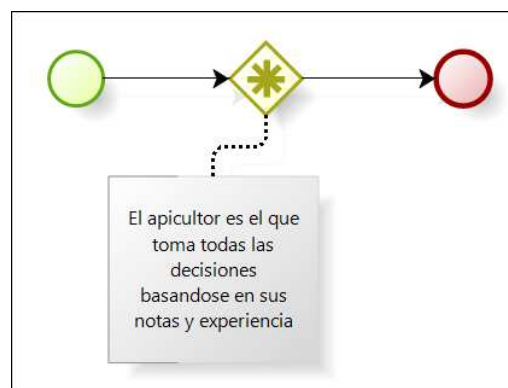


Diagrama 2 - Proceso de negocio planear visita (Carreño I. 2012)

- Realizar tarea planificada

El diagrama consta de un condicional múltiple en el cual se puede optar por los 3 subprocesos que detallaremos luego

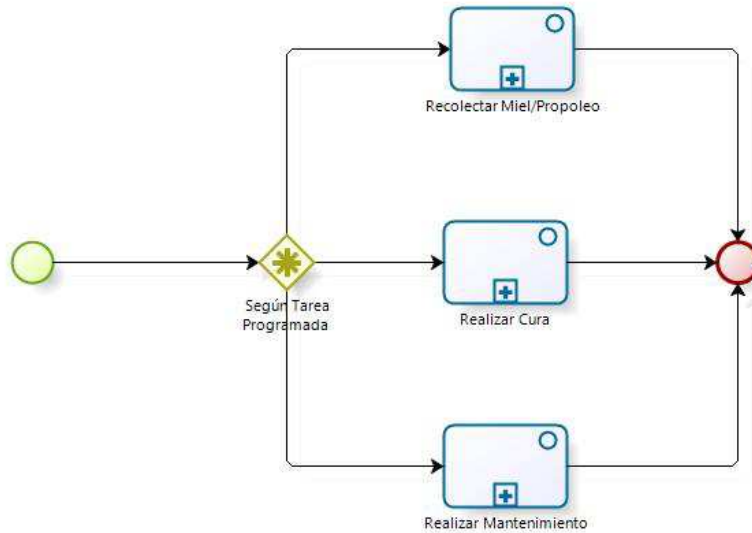


Diagrama 3 - Proceso de negocio realizar tareas programadas (Carreño I. 2012)

### Sub procesos de realizar tarea planificada

- Realizar Cura

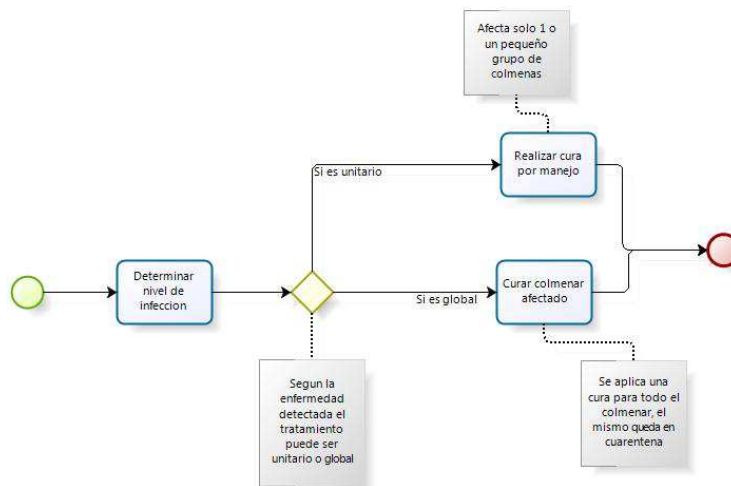


Diagrama 4 - Proceso de negocio realizar cura (Carreño I. 2012)

- Realizar Mantenimiento

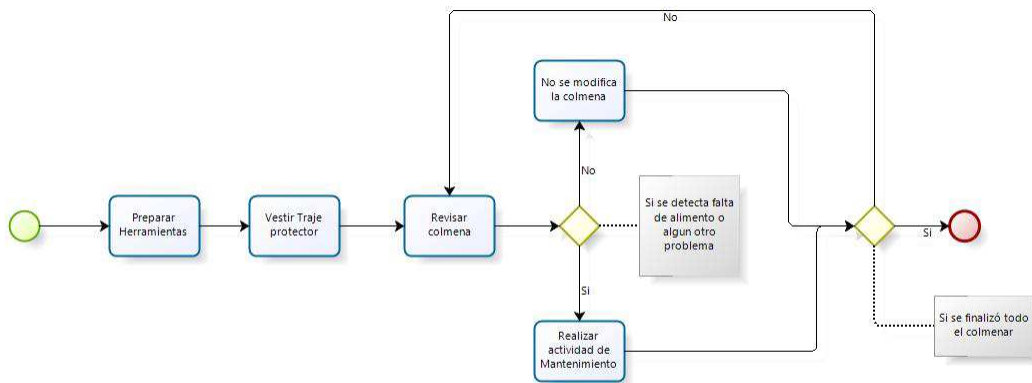


Diagrama 5 - Proceso de negocio realizar mantenimiento (Carreño I. 2012)

- Recolectar producto (miel/propóleo)

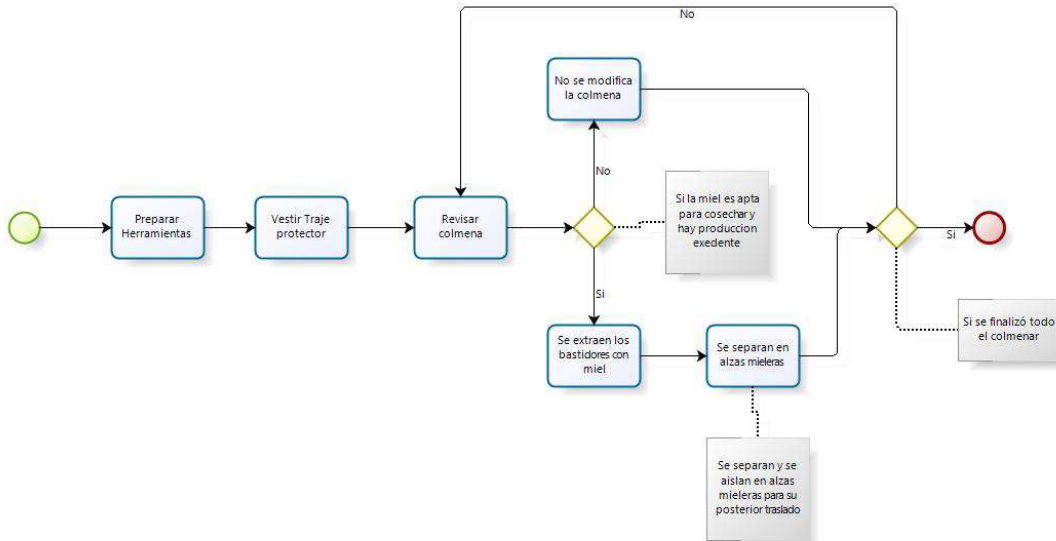


Diagrama 6 - Proceso de negocio recolectar producto (Carreño I. 2012)

### *Soluciones de Software similares*

El primer esquema racional de anotaciones sobre colmenas y apiarios se ofreció a los apicultores daneses en 1987. Desde entonces se ha ampliado gracias a la colaboración de la Sociedad de Apicultores de Dinamarca y de las aportaciones de los primeros usuarios.

Existen varios softwares apícolas disponibles en el mercado, a continuación se detallará el más importante y completo de todos.

#### *Apitrack*

Es uno de los más completos, se utilizó de ejemplo para implementar algunas ideas del sistema y está segmentado por módulos los cuales serán detallados a continuación:

##### Captor de datos con lector de código de barras o RFID

En dicho captor se pueden cargar todos los datos de los apiarios y/o las colmenas: Abejas reinas, agresividad, posturas, cámaras de cría, cuadros, cosecha, enfermedades, medicación, entre otros.

Este módulo permite:

- Identificar las colmenas leyendo la etiqueta de código de barras o los identificadores RFID.
- Acceder a la información de las últimas cuatro visitas realizadas a una colmena.
- Recolectar los datos por colmena o por apiario.
- Transferir los datos al computador central
- Programar tareas de campo.
- Controlar empleados.
- Controlar stock.
- Controlar el estado de las abejas reinas por apiario.

- Mover las colmenas o apiarios a nuevas locaciones.
- Auditar la cantidad, estado sanitario, ubicación, entre otros.

Programa para el ordenador

Módulo productor

Permite imprimir etiquetas de código de barras de colmenas, crear apiarios para trashumancia y polinización, realizar manejo de abejas reinas, enfermedades, colmenas, apiarios, estadísticas, control de visitas, historia de cada colmena. Si se registra esta información, se dispondrá de trazabilidad hasta la colmena. Permite emitir reporte de visitas para el control de empleados

Si no se dispone del captor de datos, permite cargar la información recogida en planillas y/o cuaderno de campo.

Este módulo permite:

- Identificar las colmenas de manera única.
- Programar sus tareas de campo.
- Control de personal.
- Control y producción de reportes diarios de visitas por colmena o por apicultor.
- Verificar el estado de las abejas reinas (vencidas - huérfanas) por colmena o apiario.
- Mover las colmenas a nuevas ubicaciones (por colmena o por apiario completo).
- Generar reportes de sanidad de colmenas, producción, polinización, legales entre otros.
- Generar Estadísticas



- Imprimir documentos exigidos por los organismos de control

#### Módulo sala de extracción

Este módulo permite:

- Envasar todos los productos de la colmena.
- Usar diferentes envases (vidrio, plástico, tamaños diversos).
- Identificar los envases con etiquetas de código de barras o identificadores RFID.
- Asegurar la trazabilidad de todos los productos envasados.
- Controlar stock.
- Controlar la recepción de alzas propias y en proceso para terceros.
- Generar estadísticas sobre rendimientos, producción, entre otros.
- Imprimir todos los documentos exigidos por los organismos de control.

#### Módulo de sala de fraccionamiento.

Permite envasar desde tolva y tambores a cualquier envase, generando el número de lote para asegurar trazabilidad hasta el envase de consumo, desde donde se tengan los datos cargados (colmena, apiario, productor).

#### Módulo de stock

Este módulo está preparado para agregar diferentes atributos a los productos ingresados (color, conductividad, origen botánico, entre otros.) permitiendo de esta manera, en caso que no se haya realizado la carga de la trazabilidad en las etapas anteriores, asegurar trazabilidad desde la sala de extracción, el acopiador o el exportador.

## Módulo de seguridad

Permite configurar acceso a diferentes usuarios.

## Panel de control

- Permite cargar los datos de trazabilidad de cada usuario.
- Hacer backups de la información.
- Administrar los apicultores usuarios.
- Actualizar el sistema y manuales (sin costo).
- Administrar los datos de trazabilidad

## Módulo de reportes de calidad y trazabilidad

Esta herramienta en línea permite a los compradores industriales y usuarios finales acceder a los reportes de calidad y trazabilidad vía Internet.

## C4.2 Diagnóstico

El diagnóstico será detallado siguiendo la estructura del organigrama funcional.

### *Logística*

- Dificultad para programar visitas: Este problema tiene su origen en la escasa información que los clientes pueden recabar de sus apiarios, provocando la planificación no estructurada de las visitas. Los clientes utilizan su experiencia como apicultores y su habilidad para recordar detalles especiales de cada visita pasada que hubiesen realizado, por ejemplo, si en la última visita al apiario "x" detectaron una colmena débil, en su próxima visita incluirán una porción de jarabe<sup>7</sup> para suministrarlo en dicha colmena.
- Sobrecarga de elementos a trasladar: Los clientes suelen trasladar desde su depósito hasta el apiario a visitar, muchos materiales y herramientas que quizás no sean utilizados, simplemente por precaución.
- Escaso control de stock: Dificultad para conocer en tiempo y forma la cantidad de miel disponible para comercializar y las herramientas que poseen.
- Colmenas sin identificación única: Falta de identificación única, estado y detalle de las colmenas; la causa de este problema es la ausencia de un método o sistema eficaz para documentar y analizar sus colmenas una vez concluida la visita al apiario.
- Abejas reinas sin identificación única: Al igual que la falta de identificación única para colmenas, las abejas reinas no están estrictamente identificadas y la edad de las mismas se estima según la edad de la colmena, tomando como punto de referencia el recambio de la colmena. Finalmente los clientes optan por observar periódicamente la abeja reina en busca de síntomas de vejez o disminución de aptitudes físicas para sustituirla por una nueva.

---

<sup>7</sup> Fluido compuesto de agua y azúcar que sirve de alimento para las abejas

### *Producción*

- Producto final imposible de trazar: Se desconoce el origen y el recorrido del producto final detalladamente. La documentación de la producción se realiza una vez finalizado el proceso de extracción de miel, concretamente en el envasado del producto. Los clientes se ven imposibilitados de rastrear la información a nivel de ubicaciones, apiarios y colmenas una vez envasado el producto.
- Falta de soporte para la toma de decisiones: A diferencia de la planificación de la visita, en el momento en que se está realizando la visita al apiario y desarrollando las tareas planificadas, los clientes manifiestan que no tienen una herramienta que les ayude a tomar decisiones.

### *Compras y ventas*

- Información mínima e inespecífica: Dificultad a la hora de generar un reporte gráfico o esquema de ingresos y egresos monetarios, que ayude a los clientes a ver la evolución de su producción, segmentado por ubicación y apiario y limitado por fecha.

### *C4.3 Propuesta*

#### *Propuesta del producto “BeeMore”*

Se propone desarrollar un sistema web para la administración apícola, construido en base a los requisitos de los clientes pero pensado para el resto de los pequeños, medianos y grandes apicultores con las mismas necesidades. Está destinado para todos aquellos que necesiten administrar sus colmenas, apiarios, visitas a apiarios, stock de herramientas, abejas reinas, cosechas, niveles de producción y envasado de producto final para luego poder generar los reportes estadísticos que ayudarán a tomar mejores decisiones de su negocio.

Con el registro sistemático de los datos que actualmente los clientes no pueden manejar, el sistema mitigará todas las falencias o problemas detectados en el diagnóstico, brindando información detallada en tiempo y forma para poder tomar decisiones.

El sistema se sustentará económicamente en base a un modelo de negocios freemium, limitando al usuario free del premium por la cantidad de colmenas registradas por el apicultor en el sistema. Inicialmente permitirá el registro de hasta 500 colmenas de forma gratuita. A partir de esta cantidad, el usuario deberá abonar una suscripción mensual de AR\$ 50.

El límite de 500 colmenas registradas surge por el análisis de los datos de los clientes. Carlos Cantarutti posee 300 colmenas y la apicultura es su segundo ingreso de dinero. Elio Cantarutti en cambio posee 645 colmenas y la apicultura es su principal fuente de ingresos. Aquí se puede ver que el promedio de ambas cantidades de colmenas es de 472.25 y a fines prácticos se ha redondeado a 500.

Debido a la incertidumbre que rodea a este tipo de negocios, tanto el límite entre usuarios free y premium como el monto de la suscripción mensual, pueden variar en el futuro. Cuando haya una mayor cantidad de usuarios registrados y por consiguiente más información para procesar, se podrán ajustar mejor estos límites

El sistema será desarrollado íntegramente en PHP utilizando el framework CakePHP para reducir los tiempos de codificación y pruebas. CakePHP es un framework estable, seguro y con un alto grado de madurez. El mismo es mantenido por una gran comunidad haciendo que el sistema sea escalable y entendible para otros desarrolladores.

## Capítulo 5, Implementación

### *C5.1 Listado de requerimientos*

#### *Funcionales*

- RF 1. Permitir registrarse al sistema
- RF 2. Validar el registro
- RF 3. Permitir iniciar y finalizar la sesión de usuario
- RF 4. Contemplar niveles de usuarios (roles) y permisos
- RF 5. Contemplar la existencia de un administrador (administrator)
- RF 6. Permitir registro como apicultor administrador (manager)
- RF 7. Permitir al manager agregar empleados (employee)
- RF 8. Permitir registro de employee asociando automáticamente a empresa de manager
- RF 9. Permitir crear, modificar, listar, leer y borrar ubicaciones para apiarios
- RF 10. Permitir crear, modificar, listar, leer y borrar apiarios
- RF 11. Permitir crear, modificar, listar, leer y borrar colmenas
- RF 12. Permitir crear, modificar, listar, leer y borrar abejas reinas
- RF 13. Permitir crear, modificar, listar, leer y borrar herramientas
- RF 14. Permitir crear, modificar, listar, leer y borrar visitas
- RF 15. Permitir crear, modificar y leer empresa
- RF 16. Permitir crear un perfil de usuario
- RF 17. Contemplar el envío de mensajes entre usuarios
- RF 18. Llevar el registro de cosechas por colmena
- RF 19. Contemplar el envío de notificaciones a usuarios
- RF 20. Generar reportes estadísticos de producción

RF 21. Contemplar ingresos y egresos monetarios de managers para elaborar un flujo de fondos

RF 22. Permitir trazar un elemento viendo su historial de movimientos

*No Funcionales*

RNF 1. Debe ser intuitivo

RNF 2. Debe ser compatible con tablets

RNF 3. Debe ser compatible con smartphones

*Candidatos*

RC 1. Implementar un asistente para que guie al usuario inexperto



C5.2 Diagramas de casos de uso y prototipos de interfaces

Casos de uso generales

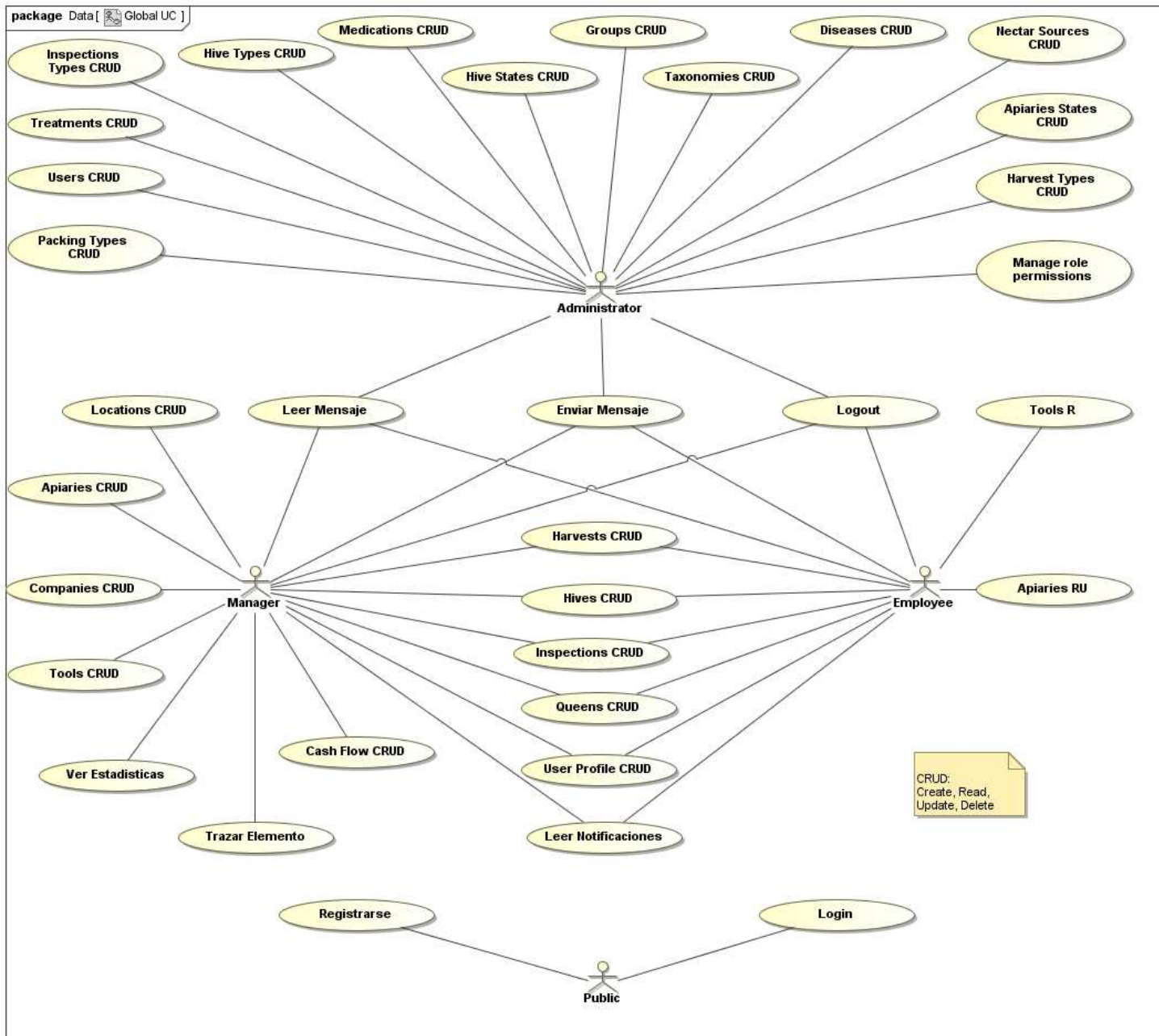


Diagrama 7 - Casos de uso generales (Carreño I. 2012)

Casos de uso específicos, prototipos de interface y ficha de caso de uso

- Actores involucrados: *Public*



Prototipo de interface 1 - Login (Actor: Public) (Carreño I. 2012)

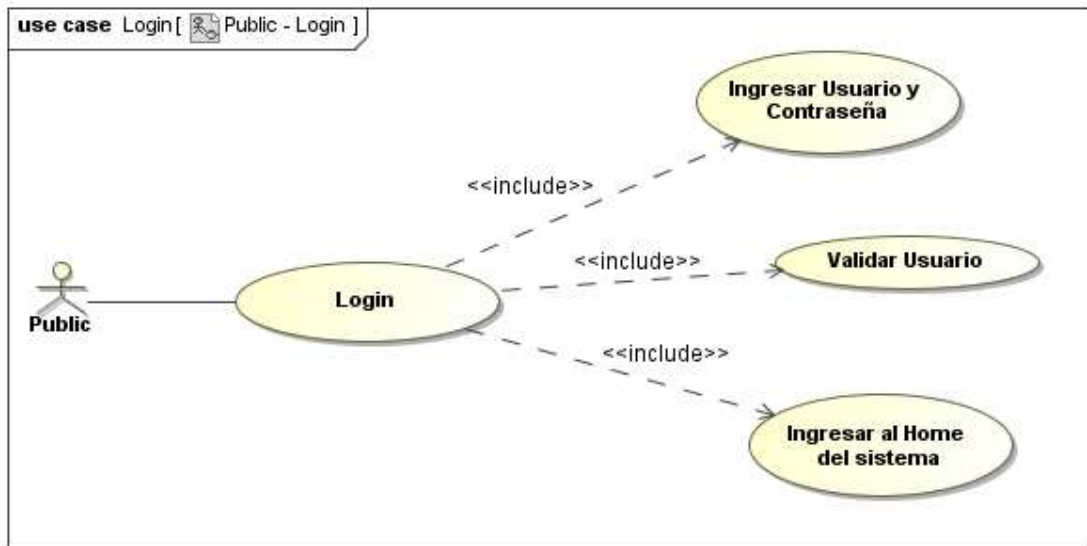


Diagrama de caso de uso 1 - Login (Actor: Public) (Carreño I. 2012)

## Registro

Nombre

Apellido

Contraseña

Correo electrónico

Prototipo de interface 2 - Registrarse (Actor: Public) (Carreño I. 2012)

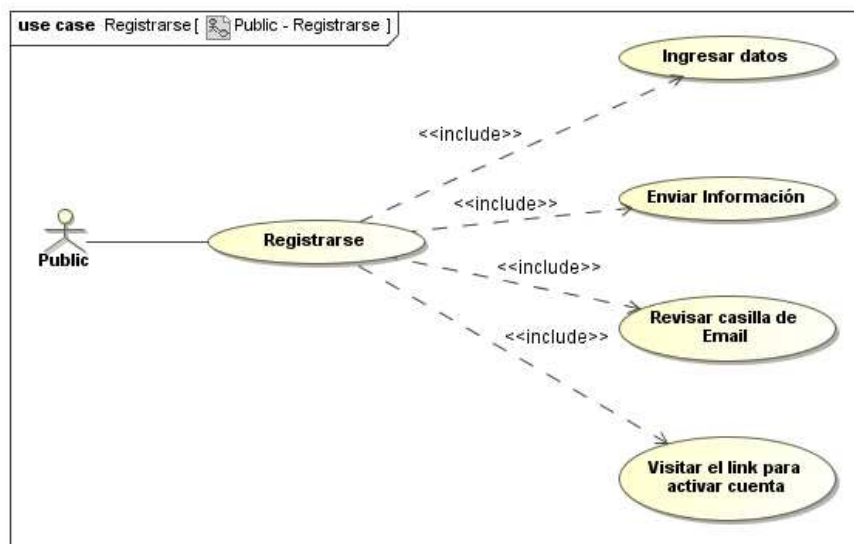


Diagrama de caso de uso 2 - Registrarse (Actor: Public) (Carreño I. 2012)

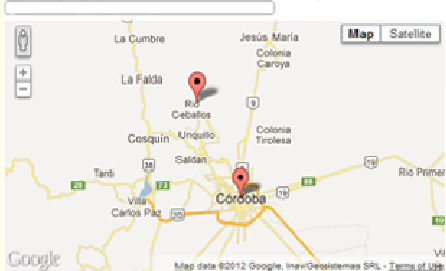
Para más detalle ver: Diagrama de secuencia 1 - Registrarse, pag:77

- Actores involucrados: *Manager - Employee*

## Añadir ubicación

**Nombre**

**Descripción**



A Google Map showing a region with several locations marked. A red pin is placed on 'Río Ceballos'. Other locations include La Cumbre, La Faldia, Cosquin, Saldoan, Villa Carlos Paz, and Corcooba. The map includes a search bar at the top and 'Map' and 'Satellite' buttons at the top right.

**Enviar**

Prototipo de interface 3 - Añadir registro de ubicación (Actor: Manager) (Carreño I. 2012)

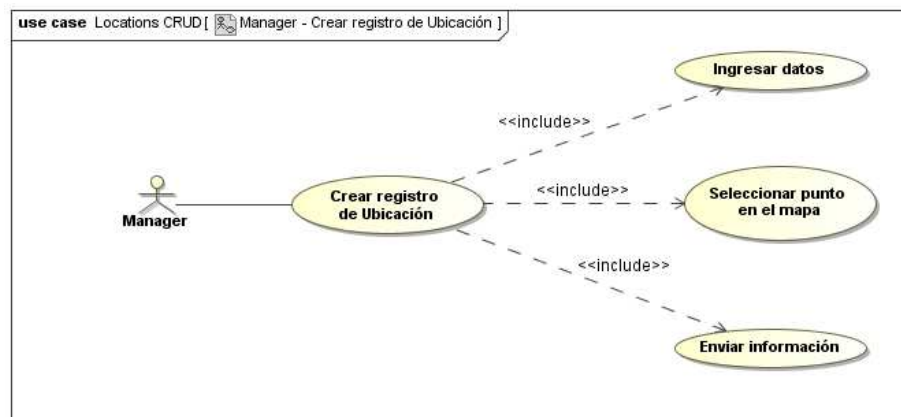


Diagrama de caso de uso 3 - Añadir registro de ubicación (Actor: Manager) (Carreño I. 2012)

## Crear Apiario

Estados de Apiario\*

Nuevo ▾

Ubicación

Justo Daract, San Luis, Argentina ▾

Nombre

Nota

Enviar

Prototipo de interface 4 - Crear registro de apiario (Actor: Manager) (Carreño I. 2012)

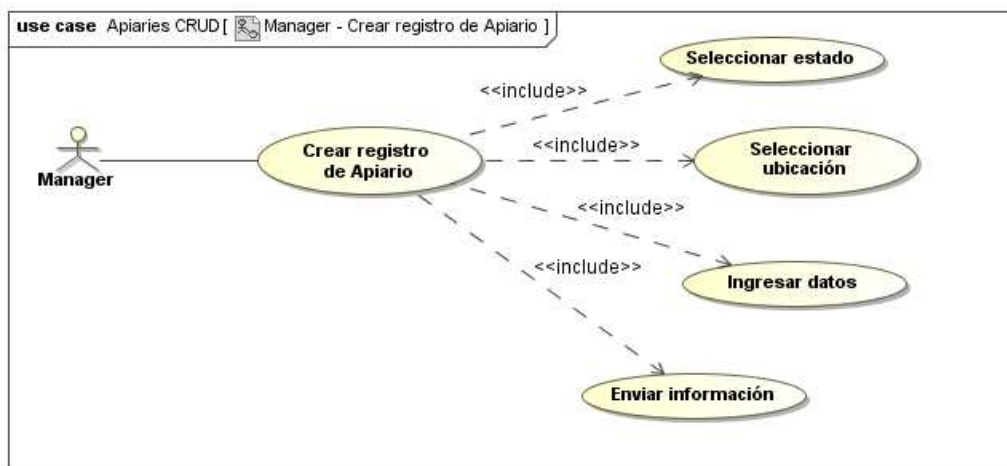


Diagrama de caso de uso 4 - Crear registro de apiario (Actor: Manager) (Carreño I. 2012)

### Añadir colmena

**Apiario\***

**Estado de colmena\***

**Tipo de colmena\***

**Nota**

Prototipo de interface 5 - Añadir registro de colmena (Actor: Manager, Employee) (Carreño I. 2012)

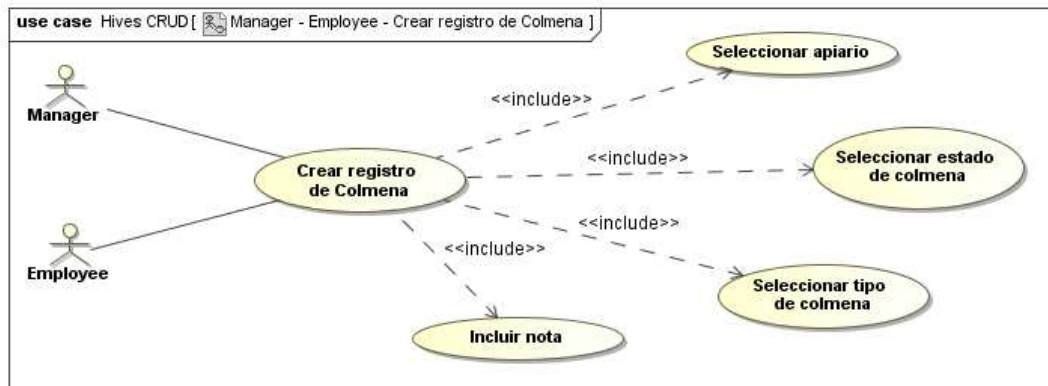


Diagrama de caso de uso 5 - Añadir registro de colmena (Actor: Manager, Employee) (Carreño I. 2012)



Prototipo de interface 6 - Añadir registro de reina (Actor: Manager) (Carreño I. 2012)

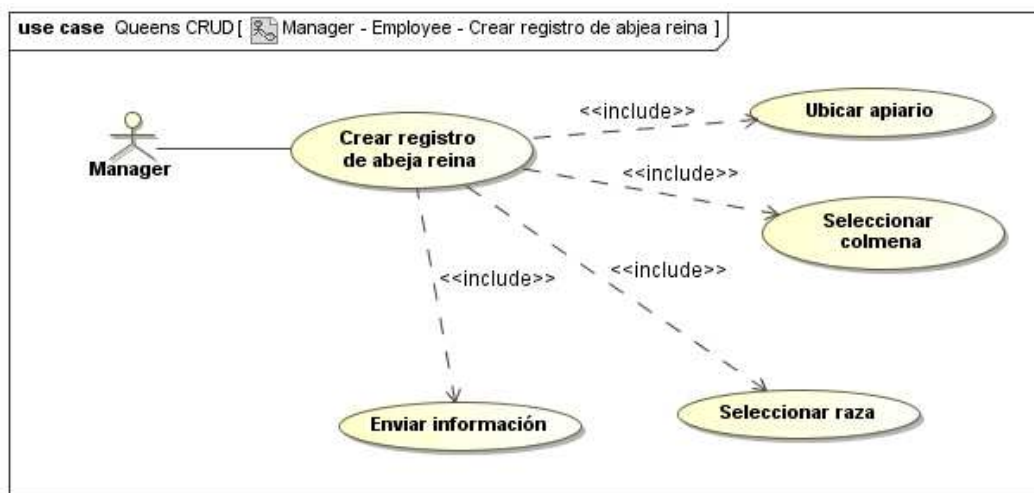


Diagrama de caso de uso 6 - Añadir registro de reina (Actor: Manager) (Carreño I. 2012)



Prototipo de interface 7 - Ver información de registro de reina (Actor: Manager, Public) (Carreño I. 2012)

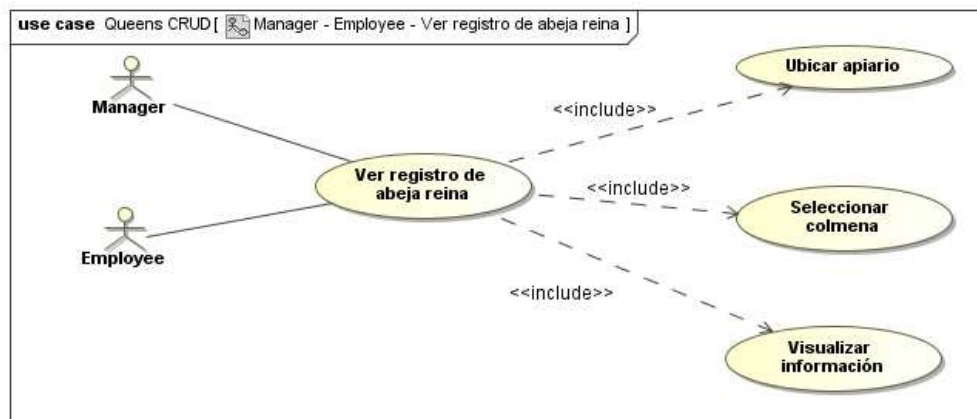


Diagrama de caso de uso 7 - Ver información de registro de reina (Actor: Manager, Public) (Carreño I. 2012)





Prototipo de interface 8 - Añadir registro de cosecha (Actor: Manager, Public) (Carreño I. 2012)

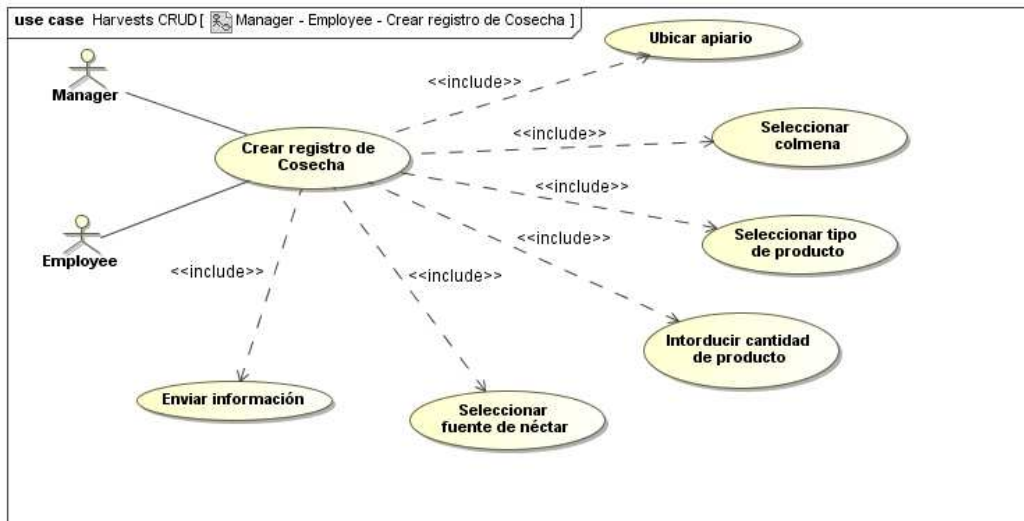


Diagrama de caso de uso 8 - Añadir registro de cosecha (Actor: Manager, Public) (Carreño I. 2012)

**Añadir empresa**

Nombre\*

Fecha de inicio

Tel

Cel

Descripción de la actividad

Agosto 2012

Lu	Ma	Mi	Ju	Vi	Sá	Do
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

Ahora

Prototipo de interface 9 - Añadir registro de empresa (Actor: Manager) (Carreño I. 2012)

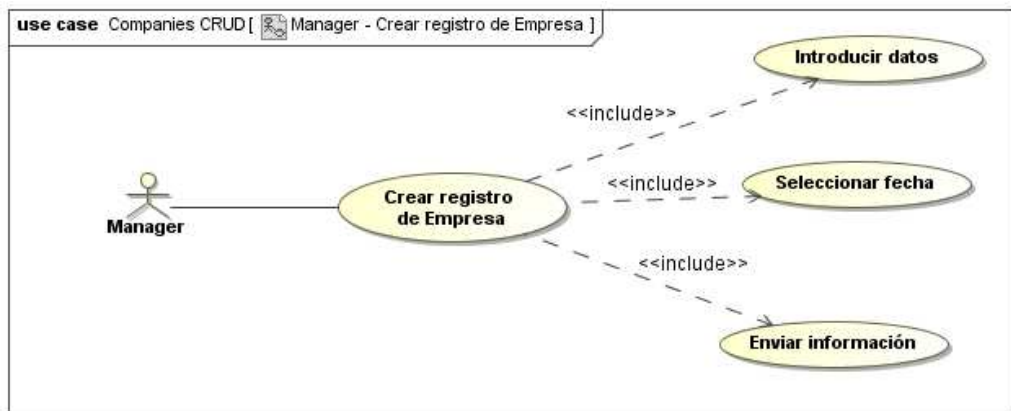


Diagrama de caso de uso 9 - Añadir registro de empresa (Actor: Manager) (Carreño I. 2012)

### Añadir herramienta

Nombre

Cantidad

Precio

Nota

Prototipo de interface 10 - Añadir registro de herramienta (Actor: Manager) (Carreño I. 2012)



Diagrama de caso de uso 10 - Añadir registro de herramienta (Actor: Manager) (Carreño I. 2012)

### Añadir visita

Tipo de visita\*  
Alimentacion

Descripción de tareas a realizar

Fecha

Integrantes

ccantarutti

Apriario

Herramientas

Enviar

Prototipo de interface 11 - Añadir registro de visita (Actor: Manager) (Carreño I. 2012)

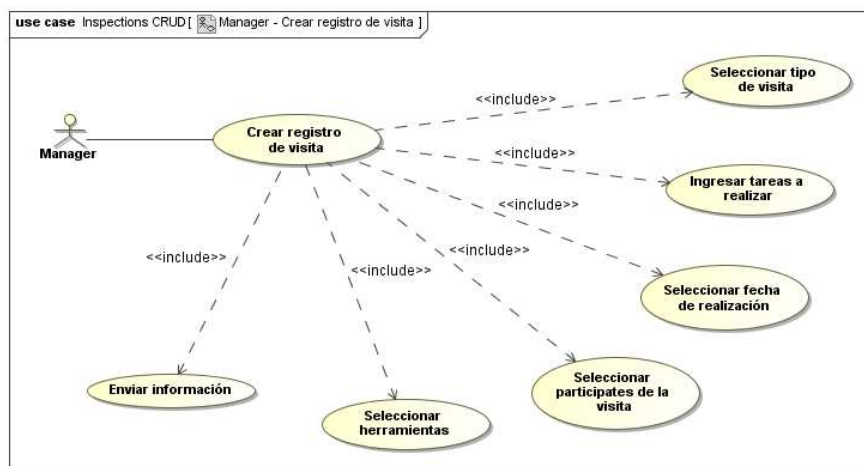
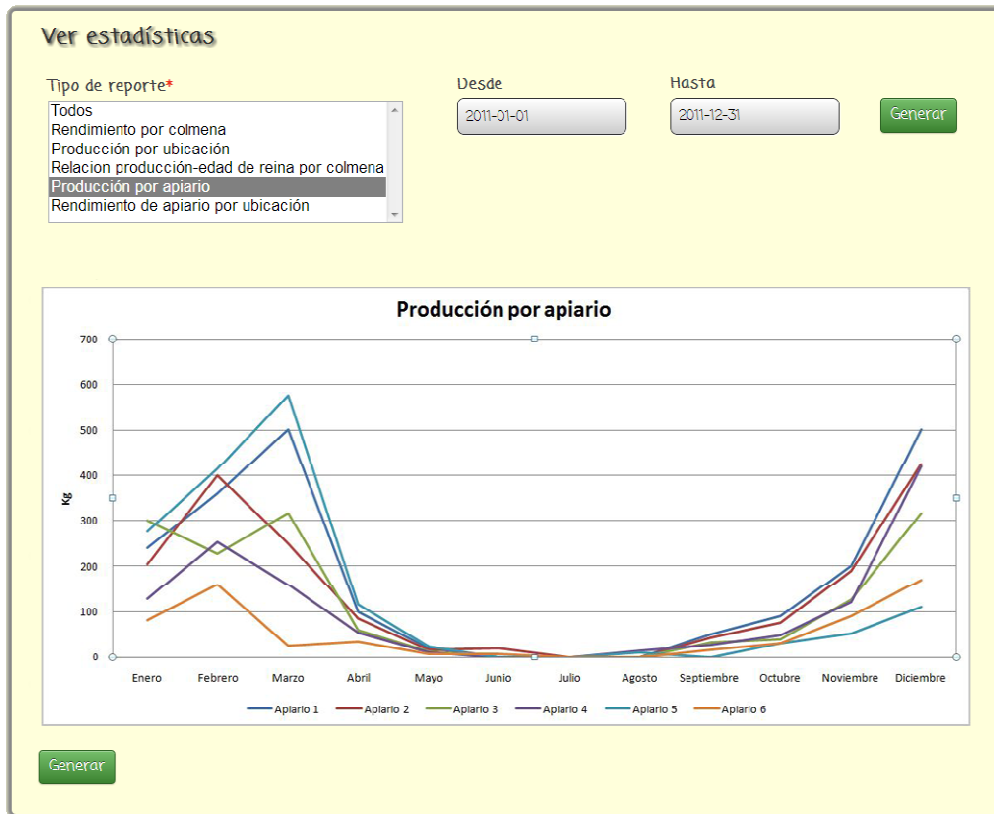


Diagrama de caso de uso 11 - Añadir registro de visita (Actor: Manager) (Carreño I. 2012)



Prototipo de interface 12 - Ver estadísticas (Actor: Manager) (Carreño I. 2012)

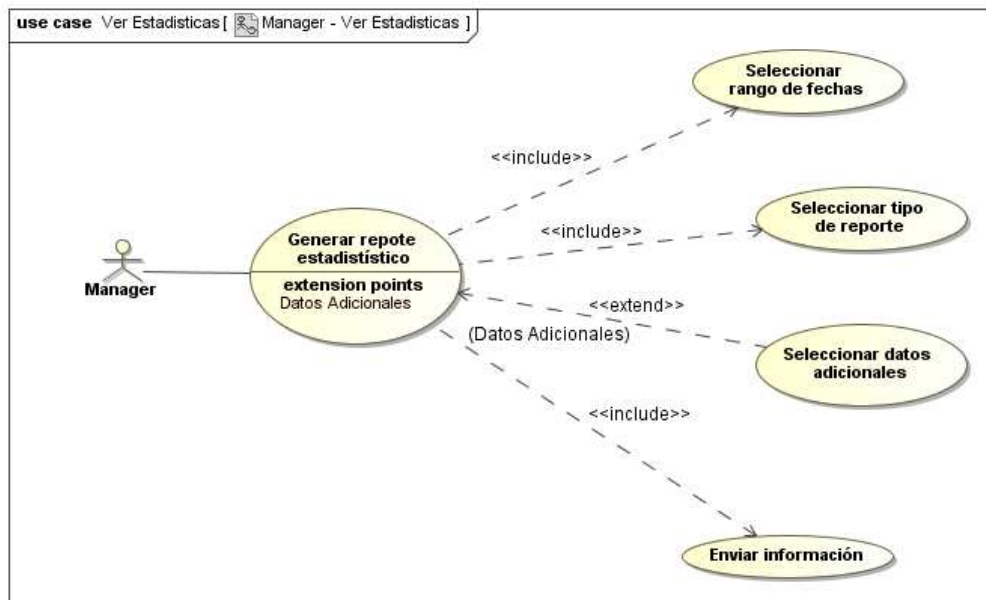


Diagrama de caso de uso 12 - Ver estadísticas (Actor: Manager) (Carreño I. 2012)

- Actor involucrado: *Administrator*

Identificador	Alias	Administrator	Manager	Employee	Public
2	<b>Apiaries</b>				
8	admin_index	✓	✓	✓	✗
9	admin_view	✓	✓	✓	✗
10	admin_add	✓	✓	✗	✗
11	admin_edit	✓	✓	✓	✗
12	admin_delete	✓	✓	✗	✗
13	<b>ApiaryInspections</b>				
24	<b>ApiaryStates</b>				
30	admin_index	✓	✗	✗	✗
31	admin_view	✓	✗	✗	✗
32	admin_add	✓	✗	✗	✗
33	admin_edit	✓	✗	✗	✗
	admin_delete	✓	✗	✗	✗
		•			
		•			
		•			
	<b>CostFlowConcepts</b>				
	<b>CostFlowPartings</b>				
Identificador	Alias	Administrator	Manager	Employee	Public

Prototipo de interface 13 - Administrar permisos (Actor: Administrator) (Carreño I. 2012)



Diagrama de caso de uso 13 - Administrar permisos (Actor: Administrator) (Carreño I. 2012)

**Añadir grupo**

Nombre

*Prototipo de interface 14 - Añadir grupo (Actor: Administrator) (Carreño I. 2012)*

**Añadir estado de Apiario**

Nombre

Descripción

*Prototipo de interface 15 - Añadir estado de apiario (Actor: Administrator) (Carreño I. 2012)*

**Añadir tipo de visita**

Nombre

Descripción

*Prototipo de interface 16 - Añadir tipo de visita (Actor: Administrator) (Carreño I. 2012)*

**Añadir enfermedad**

Nombre

Gravedad

Alcance

Taxonomía

Enfermedades
Ectoparásitos
Varroasis
Tropilaelapsosis

*Prototipo de interface 17 - Añadir registro de enfermedad (Actor: Administrator) (Carreño I. 2012)*

**Añadir tipo de cosecha**

Nombre

Descripción

Unidad

Densidad

*Prototipo de interface 18 - Añadir tipo de cosecha (Actor: Administrator) (Carreño I. 2012)*



**Añadir estado de colmena**

Nombre

Descripción

Enviar

*Prototipo de interface 19 - Añadir estado de colmena (Actor: Administrator) (Carreño I. 2012)*

**Añadir tipo de colmena**

Nombre

Descripción

Enviar

*Prototipo de interface 20 - Añadir tipo de colmena (Actor: Administrator) (Carreño I. 2012)*

### Añadir medicamento

Nombre comercial

Laboratorio

Precio

Composición

Presentación

Unidad

Alias

Taxonomía  
Medicamentos  
\_Quimicos  
\_Ejemplo1  
\_Ejemplo2  
\_Biologicos

Prototipo de interface 21 - Añadir registro de medicamento (Actor: Administrator) (Carreño I. 2012)

**Añadir fuente de néctar**

Nombre

Descripción

*Prototipo de interface 22 - Añadir registro de fuente de nectar (Actor: Administrator) (Carreño I. 2012)*

**Añadir tipo de embalaje**

Nombre

Descripción

Capacidad

*Prototipo de interface 23 - Añadir tipo de embalaje (Actor: Administrator) (Carreño I. 2012)*

### Añadir Taxonomía de Salud

Nombre

Descripción

Taxonomía padre

Prototipo de interface 24 - Añadir taxonomía de salud (Actor: Administrator) (Carreño I. 2012)

### Añadir tratamiento

Nombre

Descripción

Duración (días)

Taxonomía

Prototipo de interface 25 - Añadir tratamiento (Actor: Administrator) (Carreño I. 2012)

### C5.3 Diagramas de secuencia

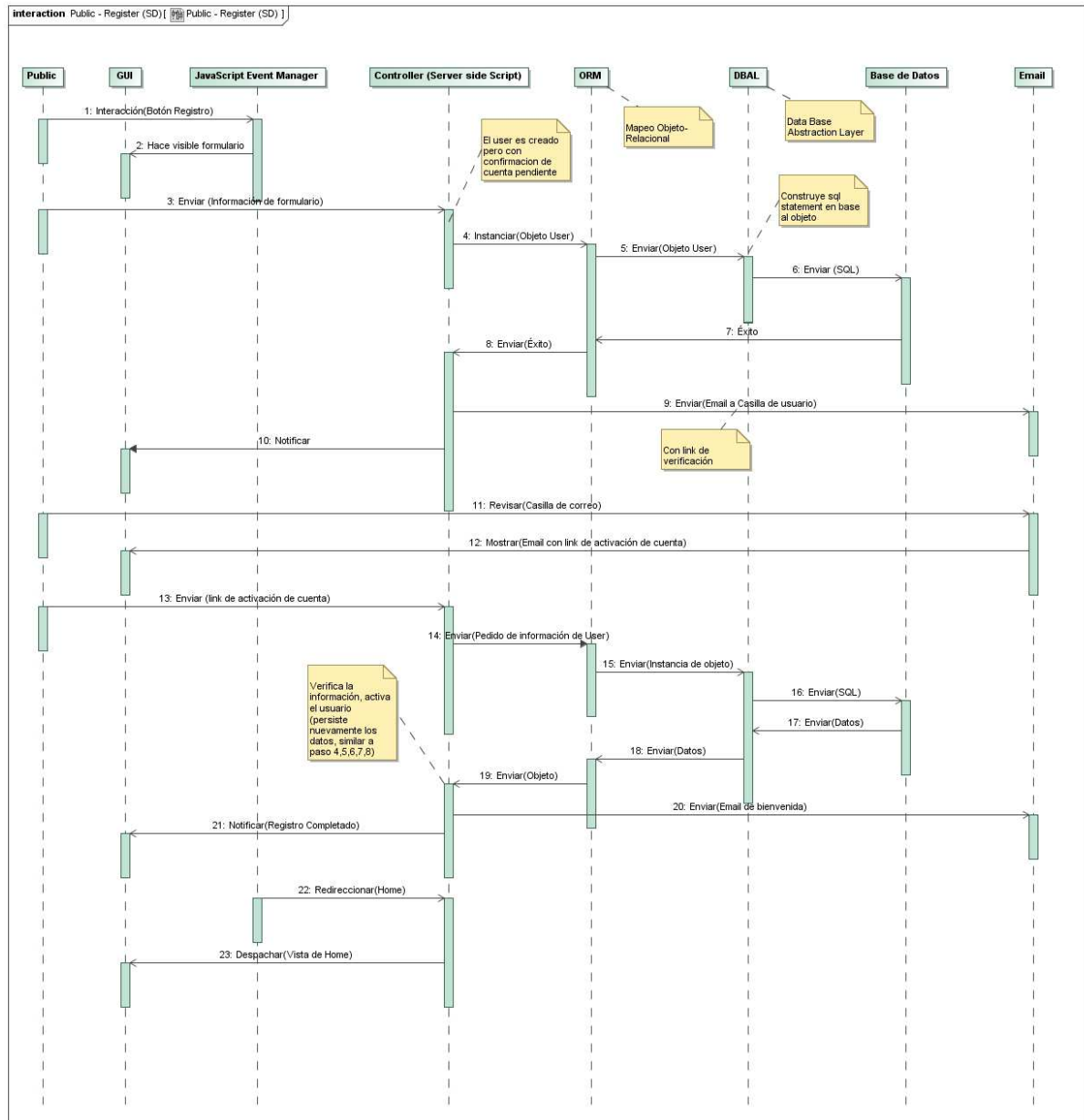
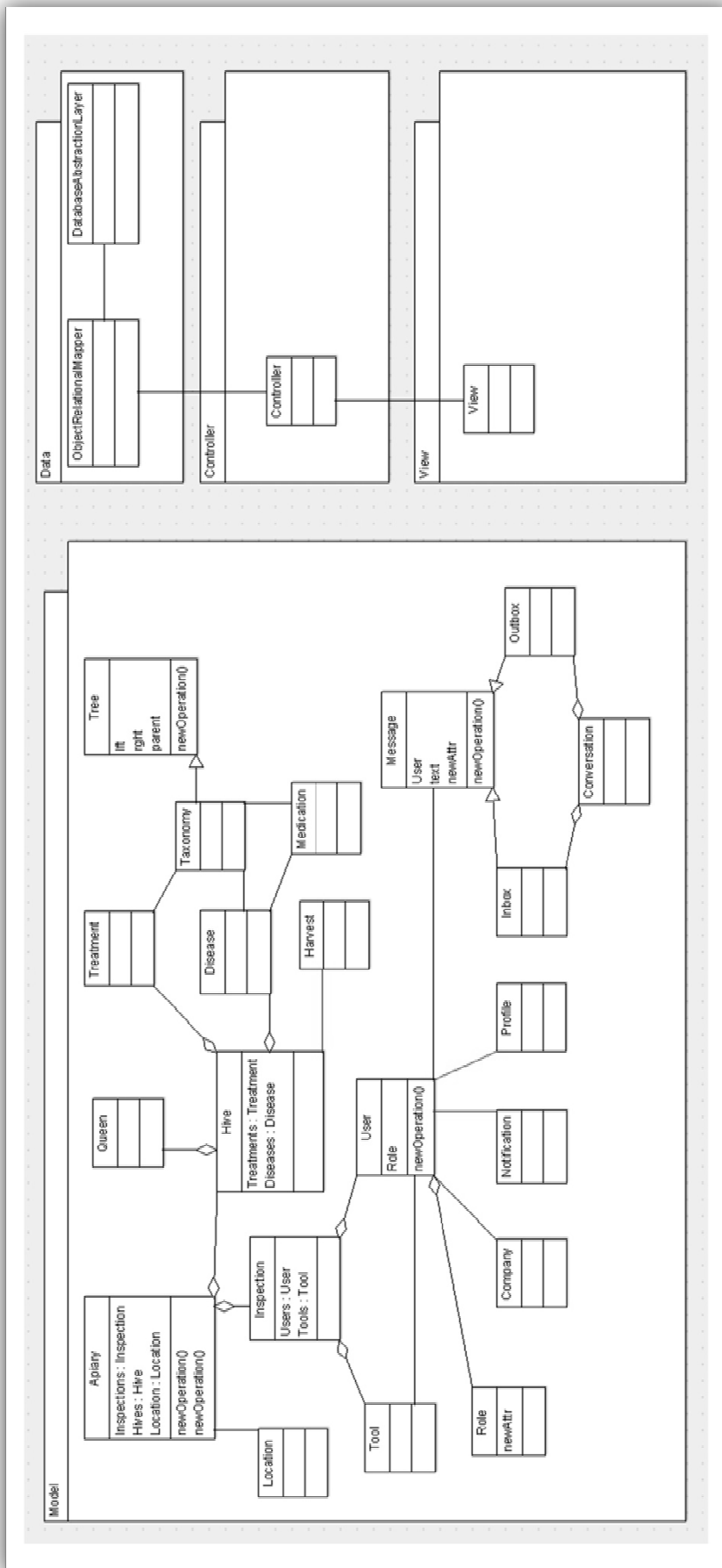
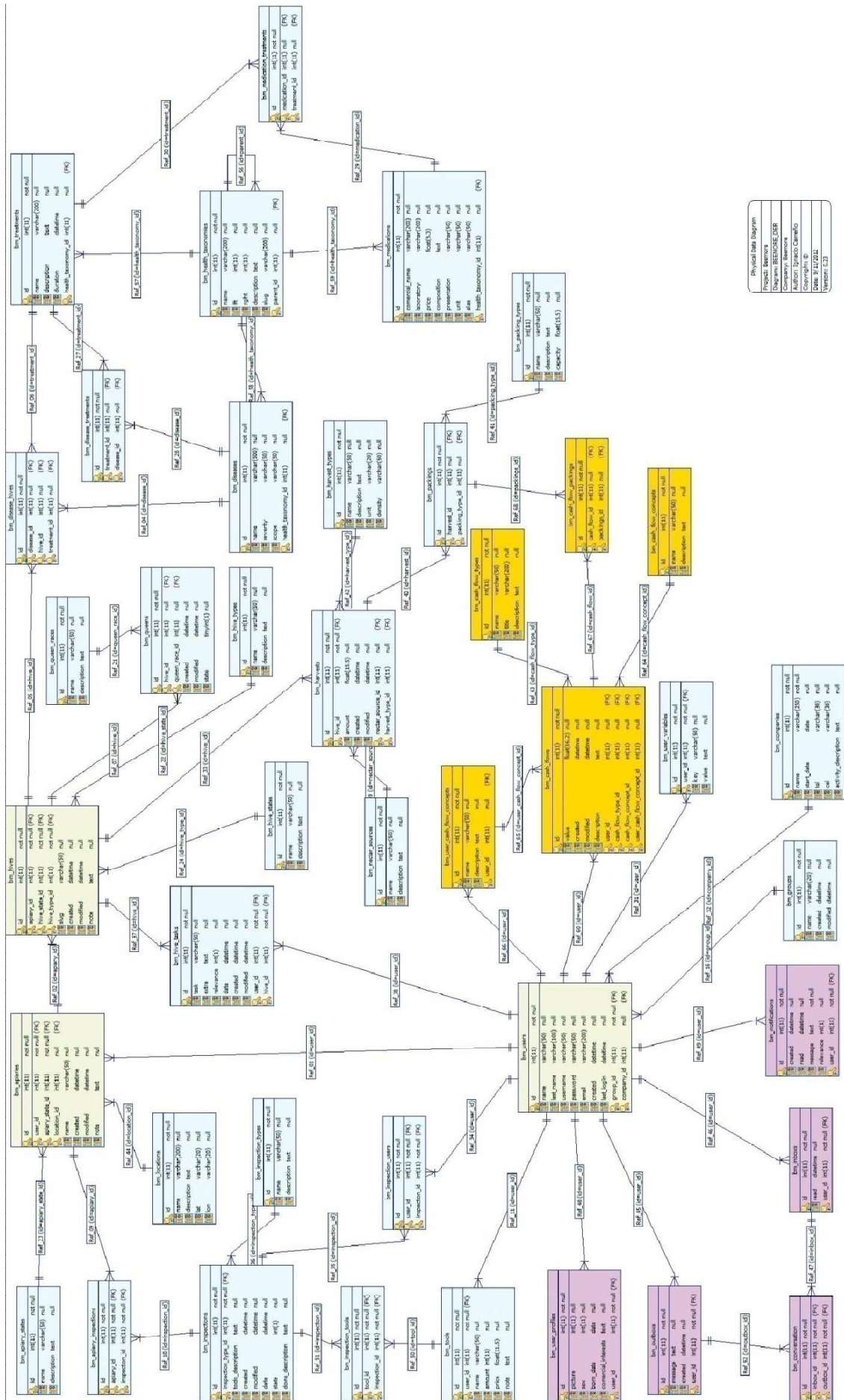


Diagrama de secuencia 1 - Registrarse (Carreño I. 2012)

### C5.4 Diagrama de clases



## C5.5 Diagrama de entidad-relación



## Conclusión

En el trabajo final de graduación, el autor pudo integrar conocimientos adquiridos a lo largo del cursado de la carrera *Ingeniería en Sistemas de Información*, para analizar y diseñar la vista abstracta de lo que luego se concretó en el desarrollo del sistema web de administración apícola “BeeMore”. Además pudo profundizar sus conocimientos en herramientas de programación y diferentes marcos de trabajo para el lenguaje PHP y escoger el más adecuado para la realización del mismo.

La motivación de hacer que el sistema sea apto para otros apicultores con necesidades similares a la de los clientes, hizo que el autor investigara modelos de negocios y escogiera el que más se adaptase a la comunidad apícola. Está pensado para que una gran cantidad de apicultores lo utilice, permitiendo la retroalimentación del sistema y la promoción del mismo a través de la recomendación de usuarios satisfechos a otros potenciales usuarios.

Este trabajo responde a una necesidad real de los apicultores y su finalidad va más allá del trabajo final de graduación ya que en un futuro estará disponible para su uso comercial.

Una vez concluido el desarrollo del sistema, se pudo observar cómo el buen uso de la herramienta brinda soporte a la toma de decisiones. Si bien este hecho no es comprobable plenamente hasta que el sistema esté implementado y se tenga feedback de los usuarios, se detallaran puntos que indican fuertemente que se han cumplido los objetivos planteados.

- Con el registro sistémico de colmenas y apiarios, y la ubicación exacta de los mismos, luego de recabar información de cosechas, obtenemos un gráfico estadístico de producción neta por ubicación, que luego se puede comparar con el rendimiento del apiario por ubicación. Esto puede revelar que el neto producido por los apiarios de determinada ubicación es el más alto pero no el más eficiente (Ver Imagen 3 e Imagen 4).



- Similar al punto anterior pero con más nivel de detalle, el sistema brinda gráficos estadísticos de porcentaje de producción por colmena. El apicultor puede observar si una colmena en particular presenta un porcentaje mayor de producción y con ello copiar sus características para aplicarlas en sus nuevas colmenas (Ver Imagen 6).
- Llevando un control de las edades de las abejas reinas de determinado apiario, el sistema genera un gráfico estadístico que interpola dicha información, con los niveles acumulados de producción de la colmena a lo largo de una temporada. Aquí el apicultor tiene una herramienta propia para determinar cuál es la franja de edades en la que sus abejas reinas hacen que su colmena sea más productiva (Ver Imagen 5).

Estos son algunos de los factores más relevantes que permiten al autor afirmar que el proyecto cumplió los objetivos generales y específicos planteados inicialmente, y el sistema, con los requerimientos extraídos de los clientes, pretendiendo brindar una herramienta capaz de organizar la información generada en la actividad apícola que ayude a la toma de decisiones.

## Bibliografía

- Brown, N. (1998). *Little Book of Testing* (Vol. I). Arlington: Computers & Concepts Associates.
- Cake Software Foundation, Inc. (2005-2012). *CakePHP*. Retrieved from <http://cakephp.org/>
- CAKEPHP. (2011). *The Cake PHP Book*.
- Chaves, M. (2005). La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. *Red de Revistas Científicas de America Latina y el Caribe, España y Portugal* , 13.
- Converse, T., Park, J., & Morgan, C. (2004). *PHP5 and Mysql Bible*. Wiley Publishing, Inc.
- Enzenhofer, L. (2003). *Herramientas de Trabajo par ala Apicultura Moderna*.
- Firtman, M. (2007). *AJAX, Web 2.0 para profesionales*. Alfaomega.
- jQuery Foundation. (2012). *jQuery*. Retrieved 09 10, 2012, from <http://jquery.org/about/>
- Object Management Group. (2008). *Business Process Model and Notation, V1.1*.
- OMG. (2011, 01 03). *BPMN, Business Process Model and Notation*. Retrieved 08 15, 2011, from <http://www.omg.org>: <http://www.omg.org/cgi-bin/doc?formal/11-01-03.pdf>
- Osterwalder, A., & Pigneur, Y. (2010). *Business Model Generation*. John Wiley & Sons, Inc.
- Potencier, F., & Zaninotto, F. (2010). *A Gentle Introduction to symfony*.
- RAE, R. A. (2010).

- Riehle, D. (2000). *Framework Design, A Role Modeling Approach*. Retrieved 6 4, 2011, from <http://www.riehle.org>: <http://www.riehle.org/computer-science/research/dissertation/diss-a4.pdf>
- Root, A. I. (1982). *The ABC and XYZ of bee culture* (14 ed.). (J. L. Ing. Argon, Trans.) Hachette S.A.
- Sanchez, V. A. (2008). *Patrones de Software*.
- The PHP Group. (2001-2012). Retrieved from [php.net](http://www.php.net): <http://www.php.net>
- Vander Verr, E. (2008). *Javascript for Dummies, Quick reference*.
- White, S. A. (2004). *Introduction to BPMN*.

## Anexo 1

### A1.1 Ejemplos de gráficos estadísticos generados por BeeMore

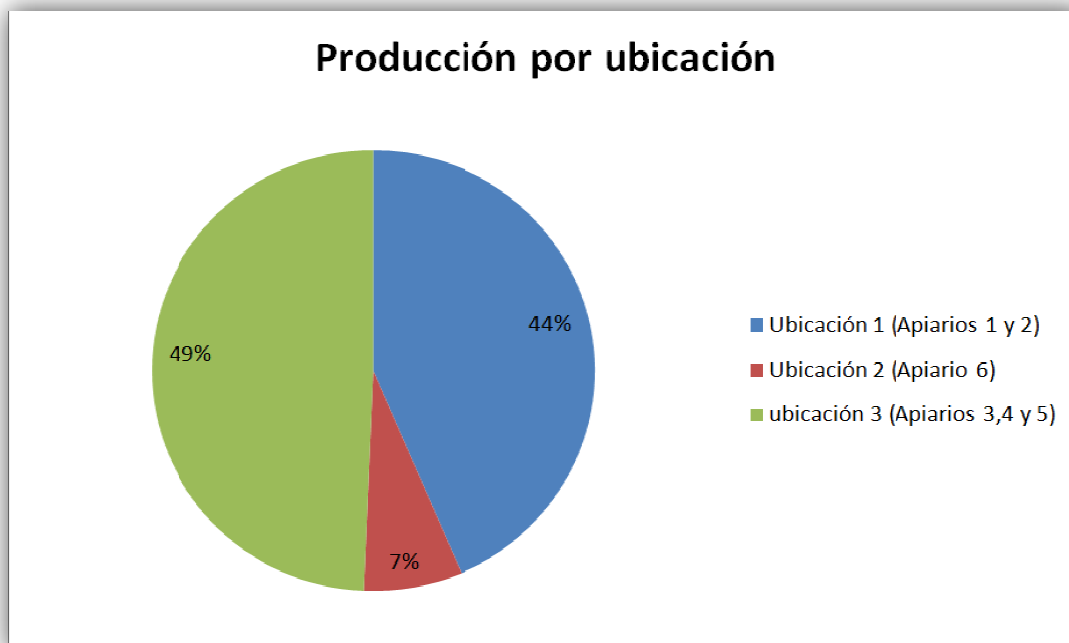


Imagen 3 - Producción por ubicación (Carreño I. 2012)

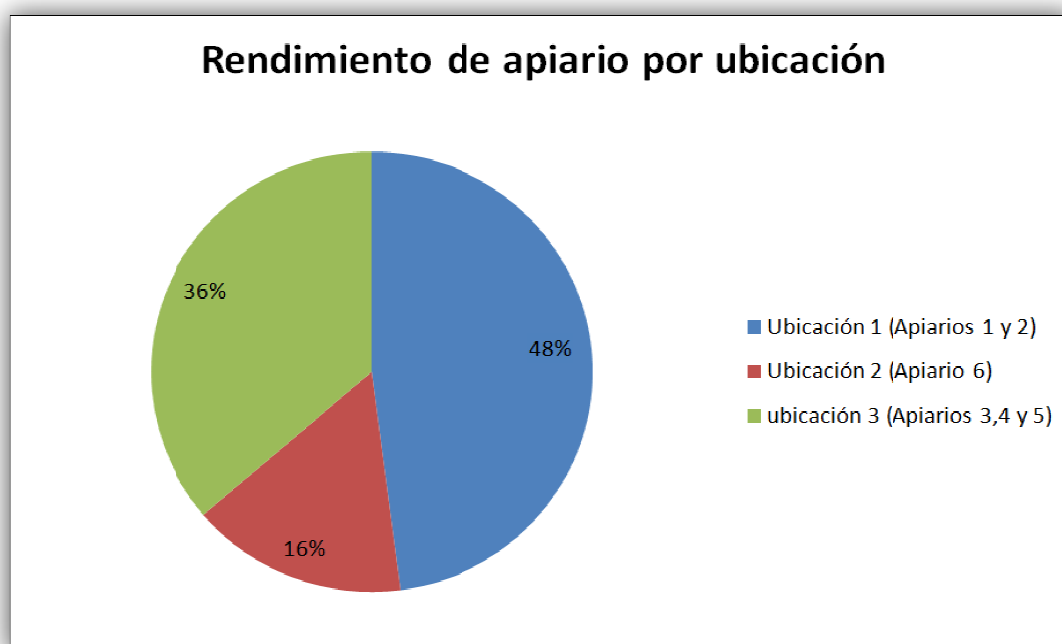


Imagen 4 - Rendimiento de apiario por ubicación (Carreño I. 2012)

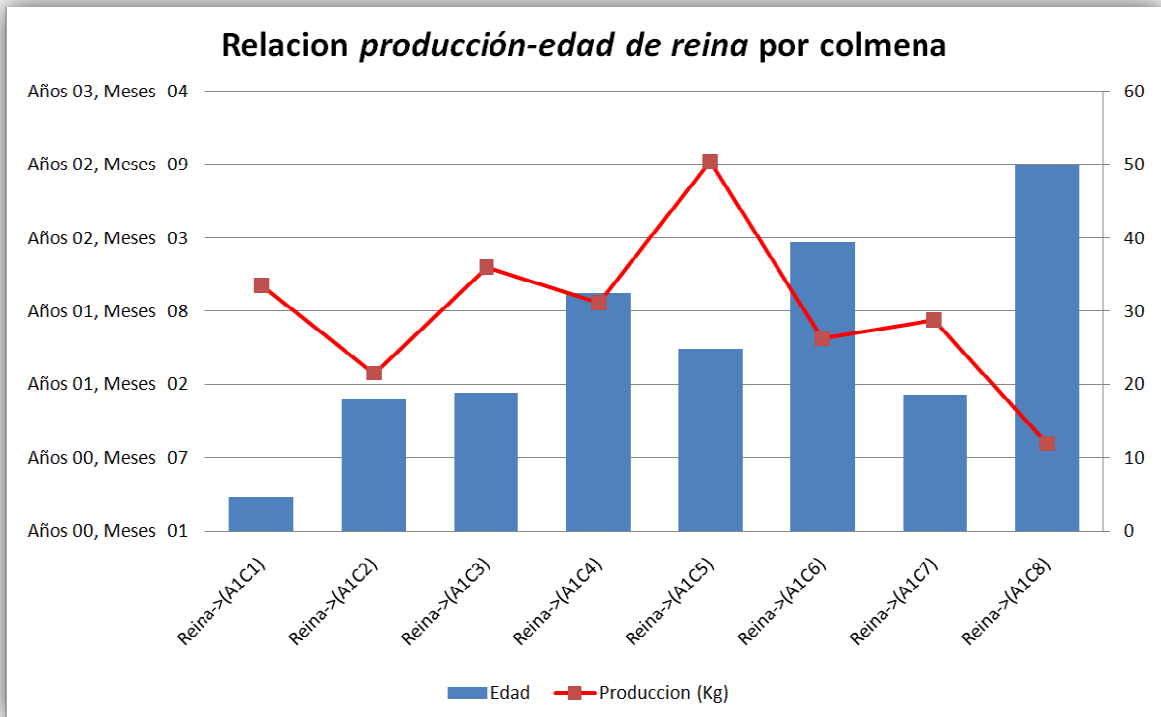


Imagen 5 - Relación producción-edad de reina p/colmena (Carreño I. 2012)

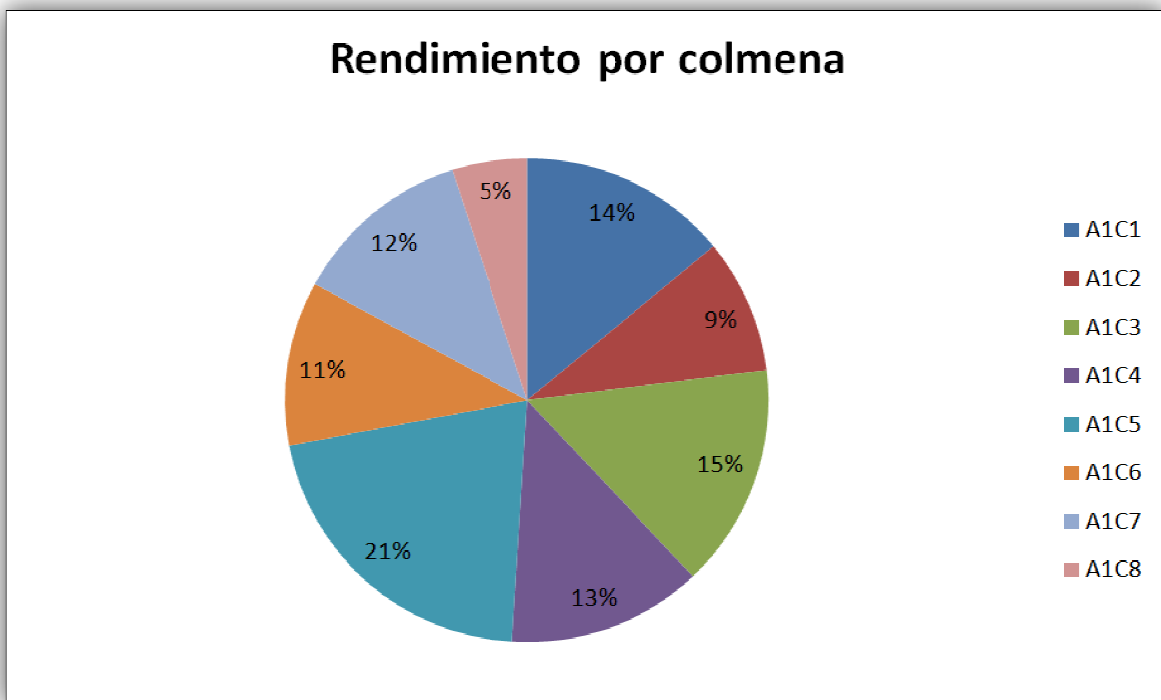


Imagen 6 - Rendimiento por colmena

## *A1.2 La apicultura*

### *Historia de la apicultura*

La evidencia más antigua de que el hombre realizara algún tipo de actividad apícola datan del mesolítico, presentes en las pinturas rupestres encontradas en la Cueva de la Araña, en Bicorp. Allí se pueden apreciar escenas de recolección de miel de panales silvestres y se calcula que estas podrían tener más de 7.000 años de antigüedad. Es recién en el Neolítico cuando el hombre aprende a controlar las abejas y enjambres.

Existen datos históricos que señalan la existencia de prácticas apícolas en el periodo predinástico de Egipto, gracias al descubrimiento de papiros que documentan el traslado de colmenas en embarcaciones a lo largo de río Nilo en el año 2400 a.C.

Los griegos, los años 480 a. C. en el Asia menor, veneraron la apicultura, dado que la Diosa Artemisa era representada en las monedas con el cuño de una abeja.

Los romanos, también practicaron la apicultura y en general heredaron las prácticas helénicas e hicieron de ellas un objeto de culto. Los poetas geórgicos dedicaron obras a la descripción de los instintos, costumbres, inteligencia de las abejas y a la explotación racional de estos animales que nunca dejaron de sorprenderlos.

La apicultura alcanzó su apogeo cuando el único elemento conocido para endulzar los alimentos era la miel. Esto cambió después del descubrimiento de la caña de azúcar, y la importancia de la apicultura decreció sensiblemente. Sin embargo su práctica no se interrumpió en ningún momento.

La apicultura moderna comienza con la creación de los panales artificiales, los cuadros móviles, las hojas de cera estampada y los extractores mecánicos, alcanzando su apogeo a fines del siglo XIX y a principios del siglo XX gracias a los trabajos de estudiosos como Karl R. von Frisch (Etólogo que describió la danza de la abeja), Harry Arthur Dade (Especialista en anatomía de las abejas), Arturo Wulfrath B (Especialista en apicultura

moderna y Autor de la Enciclopedia Apícola), Elton James Dyce (Desarrolló un procedimiento para la granulación controlada de la miel), Francesco De Hruschka (Inventor del extractor centrífugo), Jean Mehring (Inventor de la cera estampada), entre otros.

### *A1.3 Glosario apícola*

#### *Las Abejas*

Las abejas son insectos del orden de los himenópteros, pertenecientes al género APIS y especie mellifera. Viven en grandes sociedades llamadas colonias, en la apicultura moderna estas colonias son introducidas en “cajas” construidas por el hombre llamadas colmenas, ello permite criar las abejas de manera organizada para el beneficio del mismo.

Dentro de la colonia se observan tres categorías de individuos: Reina, Obreras y Zánganos.

#### *Abeja reina*

Cada colonia de abejas debe tener por lo menos una reina para mantener la coherencia de sus individuos. La reina es una hembra que ha desarrollado todos sus órganos sexuales y su tarea más importante es poner huevos. Después de cinco días de vida, la reina virgen alcanza la madurez sexual y sale de la colmena para hacer su vuelo de fecundación. Al volar encuentra y se aparea con varios zánganos, o machos. Estos dejan su semen dentro de la reina en una bolsa llamada espermateca, en la cual puede almacenar suficientes espermatozoides para el resto de su vida. El apareamiento dura una semana, la reina puede salir dos o tres veces de la colmena para aparearse y luego permanecerá de por vida en la misma. Después de hacer sus vuelos de fecundación y transcurrida una semana empieza a poner huevos todos los días del año. Durante el flujo principal de néctar pone hasta 1,500 huevos por día para aumentar la población de abejas.

Cuando la colonia tiene una buena reina, las abejas son laboriosas, pero si la reina tiene problemas físicos que la limitan o impide su postura o bien es demasiado vieja para transmitir los mensajeros químicos que mantienen a la colonia organizada, las abejas se alteran y, si es necesario, la matan para hacer una nueva reina.

### *Zánganos*

Los zánganos son los machos de la colonia. Durante los meses en que hay flores, existe mayor abundancia de zánganos en cada colonia, ya que son temporadas de reproducción. La tarea de los zánganos es fecundar a la reina virgen y aquellos que lo logran mueren, asegurando no caer en una consanguinidad.

Los zánganos están incapacitados para recoger néctar de las flores y carecen de aguijón.

### *Obreras*

La abeja obrera, al igual que la reina, es hembra, pero no se ha desarrollado para la reproducción. En casos muy especiales y cuando falta la reina, sus ovarios se desarrollan y consiguen poner huevos, pero al no ser fecundados, nacerán solamente zánganos. La abeja obrera, sin embargo, posee otros órganos que no se encuentran ni en la reina ni en los zánganos, que le permiten realizar las innumerables tareas relacionadas con la vida de la colonia.

## *A1.4 Productos derivados de la apicultura*

Todos los productos originarios de las abejas tienen un beneficio económico, alimenticio y medicinal para el hombre. A continuación se detalla cada uno:

### *Miel*



Es una sustancia azucarada que las abejas producen a partir del néctar que recogen de las flores. Es el alimento básico de las abejas y a través de él adquieren energía necesaria para desarrollar todas las actividades de la colonia gracias su alto contenido en azúcares y calorías.

### *Cera*

Es un producto que producen las abejas a través de las glándulas cereras entre su 13° y 18° día de edad. La utilizan para construir los panales sobre los cuales la reina depositará los huevos y las abejas almacenarán la miel y el polen. También la ocupan para sellar las celdillas que tienen larvas hasta el momento de nacer. La materia prima para producir cera es la miel, y las abejas necesitan consumir de 6 a 7 kg de miel para producir 1 kg de cera. El hombre utiliza la cera para hacer velas, aceites y artesanías en general.

### *Jalea Real*

Consiste en una sustancia que las abejas jóvenes segregan entre su 4° y 12° día de edad para alimentar a las larvas durante sus 3 primeros días y a la reina durante toda su vida. Las materias primas necesarias para su elaboración son el polen, la miel y el agua, las cuales al ser consumidas por las abejas se transforman en jalea real por la acción de las glándulas hipo faríngeas.

### *Propóleo*

Es una especie de resina que las abejas recogen de algunos árboles. El propóleo es un producto muy importante para la colmena, ya que a través de él mantienen el calor y la higiene. En algunos países se utilizan los extractos de propóleo en el campo de la medicina como cicatrizante, bactericida y fungicida.

### *Polen*

Es el elemento masculino de una flor por ende no es un producto elaborado por las abejas. El polen es de suma importancia para el crecimiento y la reproducción de la colonia siendo rico en proteínas, lípidos, vitaminas y minerales.

#### *Veneno*

El veneno es producido por el propio cuerpo de la abeja obrera y lo utiliza exclusivamente como arma de defensa contra animales, insectos, personas y todo aquello que amenaza el funcionamiento de la colonia. Se utiliza en la medicina alternativa para tratar el reuma y la artritis.

### *A1.5 La colmena*

Una colmena es el lugar donde habita una colonia o familia de abejas. El conocimiento que se tiene sobre las abejas ha sido posible gracias al desarrollo de una colmena técnica. Sus principales características son:

#### *Techo*

Sirve para cubrir la colmena y protegerla de la intemperie y la lluvia. El techo está cubierto con una lámina de chapa galvanizada.

#### *Tapa*

Sirve para cerrar la colmena. Debe ser resistente para facilitar su remoción en las revisiones que periódicamente se realizan.

#### *Bastidores o Panales*

Estos consisten en cuadros que se colocan dentro de la cámara de cría y las alzas. Dentro de los bastidores se le colocan alambres horizontales donde se incrustan láminas de cera. Estas láminas forma la guía del panal y las abejas construyen sus celdas en ambos lados.

### *Cámaras de cría*

Es el primer cuerpo de la colmena y contiene los panales centrales con cría y los laterales con miel y polen. La cámara de cría tiene diez bastidores.

### *Alza*

Son cajas con sus correspondientes panales, se colocan sobre la cámara de cría para que las abejas almacenen miel. Si la colmena es fuerte y la cámara de cría está llena, la reina subirá a la primera alza en busca de espacio donde depositar los Huevos. Esto ocurre principalmente en épocas de floración cuando la colonia está en su máximo apogeo.

### *Piso*

Llamado también fondo de la colmena, es donde se asienta la cámara de cría. En su parte libre denominada piquera es por donde las abejas entran y salen de la colmena. En épocas de poca floración esta abertura se debe reducir para evitar que otras abejas puedan entrar a robar la miel así como plagas y otros enemigos de las abejas.

## *A1.5 Población de una colmena*

La colmena está compuesta por la familia o colonia de abejas que la habite. Existen 2 tipos de colonias, el núcleo (colonia pequeña) y la colmena (colonia completa).

### *Núcleo*

El núcleo de abejas consta de:

- 3 panales con cría sellada, larvas y huevos
- 1 panal con miel y polen
- 1 reina fecundada

- Todos los bastidores deben quedar cubiertos de abejas, lo que equivaldría a unas 8,000 abejas, con un peso aproximado de 1 kg. de abejas.

### *Colmena*

La colmena completa consta de:

- 6 panales con cría sellada, larvas y huevos
- 4 panales de miel y polen
- 1 reina fecundada
- 2 Kg. de abejas

### *A1.7 Enfermedades de las abejas*

Son muchas las enfermedades que atacan a las abejas melíferas como resultado de la acción de diferentes organismos patógenos, afectando a abejas adultas obreras, zánganos, reina o a la cría en desarrollo, huevo, larva o pupa.

Cada agente patógeno tiene su propia estrategia de infección y desarrollo que serán detallados en breve.

El control de estas enfermedades suele ser más importante que los tratamientos, el manejo de las colmenas y su periódica revisión son vitales para evitar que se genere una situación que favorezca el ataque de los organismos patógenos.

#### *Acariosis (ataque por ácaros)*

- Ectoparásitos o Varroasis: Ataque por el ácaro Varroa.
  - Varroa jacobsoni. Ataca Apis cerana.
  - Varroa destructor. Ataca Apis mellifera.
  - Varroa rindereri. Ataca Apis koschevnikovi.
  - Varroa underwoodi. Ataca Apis nulensis, Apis nigrocincta, Apis cerana.
  - Euvarroa wongsirii. Ataca Apis andreniformis.
  - Euvarroa sinhai. Ataca Apis florea o Tropilaelapsosis
- Ataque por ácaro. Tropilaelaps clareae

- *Tropilaelaps clareae*: Ataca: *Apis cerana*, *Apis florea*, *Apis mellifera* y *Apis laboriosa*
- *Tropilaelaps koenigerum*: es parásito de *Apis laboriosa* y *Apis dorsata*
- *Acarapis woodi*: Ataque por ácaro Acaro traqueal.

*Piojos: Insectos*

- *Braula coeca*: Piojo de la abeja

*Endoparásitos*

- *Wolbachia*

*Bacterias*

- Loque americana: Ataque Bacteriano. En inglés American foulbrood (AFB).
- Loque europea: Ataque Bacteriano. En inglés European foulbrood (EFB).
- Espiroplasmosis: Ataque Bacteriano, producido por la bacteria *Spiroplasma*.

*Enfermedades de la cría*

- Bacterias
- Hongos
- Virus
- Insectos

*Enfermedades de adultos*

- Nosemosis: Ataque por *Nosema apis*.
- Nosemosis: Ataque por *Nosema ceranae*.
- Amebiasis de la abeja. Ataque por *Malpighamoeba mellificae*.
- Flagelosis: Ataque protozoos flagelados como *Leptomonas apis*, *Crithidia mellificae*.
- Gregarinosis: Ataque por protozoos. *Monoica apis*, *Apigregarina stammeri*, *Acuta rousseaui*, *Leidyana apis*.

*Enfermedades virales*

- Parálisis de la abeja. LLamado Síndrome de la abeja negra.
  - Virus de la parálisis crónica
  - Virus de la parálisis aguda
- Otras Virosis de la abeja.
- Rickettsiosis: Ataque por la bacteria *Rickettsia*.

*Enfermedades de la abeja reina*

- Celdas reales negras: Ataque de virus que mata la larva de reina.

- Melanosis. Enfermedad fúngica producida por *Mellanosella apis*.

#### *Trastornos no infecciosos*

- Enfriamiento de la cría
- Hambre
- Diarrea
- Defectos hereditarios
- Intoxicación por plaguicidas
- Cría pelada, desnuda o tubular.
- Cría sobrecalentada
- Abejas sobrecalentada
- Síndrome Half-moon (Síndrome media luna)

#### *Polillas y hongos*

- Atacan los cuadros de cera y polen

#### *Pesticidas*

- Gaucho (insecticida) En mayo de 2004 el gobierno de Francia prohíbe el uso de este insecticida nocivo para las abejas.
- Pesticidas restricciones en apicultura.
- *Bacillus thuringiensis*. Insecticida biológico.

#### Contaminantes de la miel

- Nitrofuranos
- Tilosina
- Oxitetraciclina

#### *Medicamentos*

- Tilosina. Antibiótico Loque europea, Loque americana.
- Oxitetraciclina. Antibiótico Loque europea, Loque americana.
- Fumagilina. Antibiótico Nosemosis.
- Ácido oxálico. Acaricida orgánico. Varroasis.
- Ácido fórmico. Acaricida orgánico. Varroasis.
- Ácido láctico. Acaricida orgánico. Varroasis.
- Timol. Acaricida orgánico. Varroasis.
- Rotenona. Acaricida orgánico. Varroasis.
- *Bacillus thuringiensis*. Insecticida biológico. Galleriosis.
- Aceites esenciales. Acaricidas orgánico con aceites de diferentes especies vegetales.

Metarhizium anisopliae. Hongo que controla Varroa.

### *A1.8 Patrones de desarrollo de software*

Los patrones de diseño (*design patterns*) son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces.

Un patrón de diseño es una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reusable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias.

Objetivos de los patrones de diseño

Los patrones de diseño pretenden:

Proporcionar catálogos de elementos reusables en el diseño de sistemas software.

Evitar la reiteración en la búsqueda de soluciones a problemas ya conocidos y solucionados anteriormente.

Formalizar un vocabulario común entre diseñadores.

Estandarizar el modo en que se realiza el diseño.

Facilitar el aprendizaje de las nuevas generaciones de diseñadores condensando conocimiento ya existente.

Asimismo, no pretenden:

Imponer ciertas alternativas de diseño frente a otras.

Eliminar la creatividad inherente al proceso de diseño.

No es obligatorio utilizar los patrones, solo es aconsejable en el caso de tener el mismo problema o similar que soluciona el patrón, siempre teniendo en cuenta que en un caso particular puede no ser aplicable. Abusar o forzar el uso de los patrones puede ser un error.

### Categorías de patrones

Según la escala o nivel de abstracción:

Patrones de arquitectura: Aquéllos que expresan un esquema organizativo estructural fundamental para sistemas software.

Patrones de diseño: Aquéllos que expresan esquemas para definir estructuras de diseño (o sus relaciones) con las que construir sistemas software.

Dialectos: Patrones de bajo nivel específicos para un lenguaje de programación o entorno concreto.

Además, también es importante reseñar el concepto de Antipatrón de Diseño, que con forma semejante a la de un patrón, intenta prevenir contra errores comunes de diseño en el software. La idea de los antipatrones es dar a conocer los problemas que acarrear ciertos diseños muy frecuentes, para intentar evitar que diferentes sistemas acaben una y otra vez en el mismo callejón sin salida por haber cometido los mismos errores.

Además de los patrones ya vistos actualmente existen otros patrones como el siguiente:

Interacción: Patrones que nos permiten el diseño de interfaces web.

### Estructuras o plantillas de patrones

Para describir un patrón se usan plantillas más o menos estandarizadas, de forma que se expresen uniformemente y puedan constituir efectivamente un medio de comunicación



uniforme entre diseñadores. Varios autores eminentes en esta área han propuesto plantillas ligeramente distintas. Si bien la mayoría definen los mismos conceptos básicos.

La plantilla más común es la utilizada precisamente por el GoF y consta de los siguientes apartados:

Nombre del patrón: nombre estándar del patrón por el cual será reconocido en la comunidad (normalmente se expresan en inglés).

Clasificación del patrón: creacional, estructural o de comportamiento.

Intención: ¿Qué problema pretende resolver el patrón?

También conocido como: Otros nombres de uso común para el patrón.

Motivación: Escenario de ejemplo para la aplicación del patrón.

Aplicabilidad: Usos comunes y criterios de aplicabilidad del patrón.

Estructura: Diagramas de clases oportunos para describir las clases que intervienen en el patrón.

Participantes: Enumeración y descripción de las entidades abstractas (y sus roles) que participan en el patrón.

Colaboraciones: Explicación de las interrelaciones que se dan entre los participantes.

Consecuencias: Consecuencias positivas y negativas en el diseño derivadas de la aplicación del patrón.

Implementación: Técnicas o comentarios oportunos de cara a la implementación del patrón.

Código de ejemplo: Código fuente ejemplo de implementación del patrón.

Usos conocidos: Ejemplos de sistemas reales que usan el patrón.

Patrones relacionados: Referencias cruzadas con otros patrones.

## MVC

Actualmente hay quienes discuten sobre el modelo MVC tratando de ubicarlo por fin entre los patrones de arquitectura, de análisis o de diseño, y a veces hasta se habla de implementación. Lo cierto es que en cada una de estas instancias, el modelo MVC parece tener una componente, por ejemplo: desde la óptica del arquitecto, debería ser un patrón de arquitectura ya que es una manera de estructurar un sistema, decisión que se debe tomar al comienzo, sobre todo en cuanto a la cantidad de capas, la definición de roles por capas, entre otros.

Para el analista, se da una situación parecida, ya que tomando como base los patrones GRASP, se definen los contenidos de cada capa bajo normas de “Bajo acoplamiento”, “Alta Cohesión”, “Experto”, “Controlador”, entre otros. Y así podemos seguir enumerando las opiniones de cada uno de los integrantes del equipo, sobre en donde situar el Modelo MVC.

Pero a decir verdad, ninguno tiene una visión errónea, solo están mirando desde su perspectiva, y tienen razón. El modelo MVC se lo puede encasillar como un Meta patrón, que si condice con la arquitectura de distribución de sistemas en capas, cada una bien definida en su objetivo e interfaces con las otras, pero dando total libertad de elección sobre cómo y cuantas capas agrupar en cada “Meta capa” definida por dicho modelo.

El modelo MVC, Model View Controller o Modelo Vista Controlador, es bastante simple e intuitivo en su nombre. Describe una forma de organización de subsistemas de un software en donde cada Meta capa tiene una funcionalidad específica y bien definida. En MVC, hablamos de 3 meta capas principales y que vienen dadas por su nombre, que son:

Modelo: hace referencia al modelo de resolución del dominio de la aplicación. Es en donde encontramos las entidades interactuantes y la lógica de negocio que llevan a cabo. Denominado “Back-End”.

Vista: hace referencia a todo lo que tenga que ver con la visualización del sistema por parte del usuario, como pantallas, reportes, gráficos, entre otros. Denominado también “Front-End”

Controlador: es la parte del sistema que se encarga de definir las reglas de comunicación entre la Vista y el Modelo, y viceversa.

En si se trata de un modelo conceptual, el cual, llevado a la práctica, nos favorece a largo plazo permitiéndonos mayor reutilización y mantenimientos flexibles y rápidos. Como podemos ver en la siguiente figura, se puede utilizar cualquier recurso para llevar a cabo la funcionalidad de las meta capas del modelo, e incluso, contener más de una capa o partes de otras por abstracción.

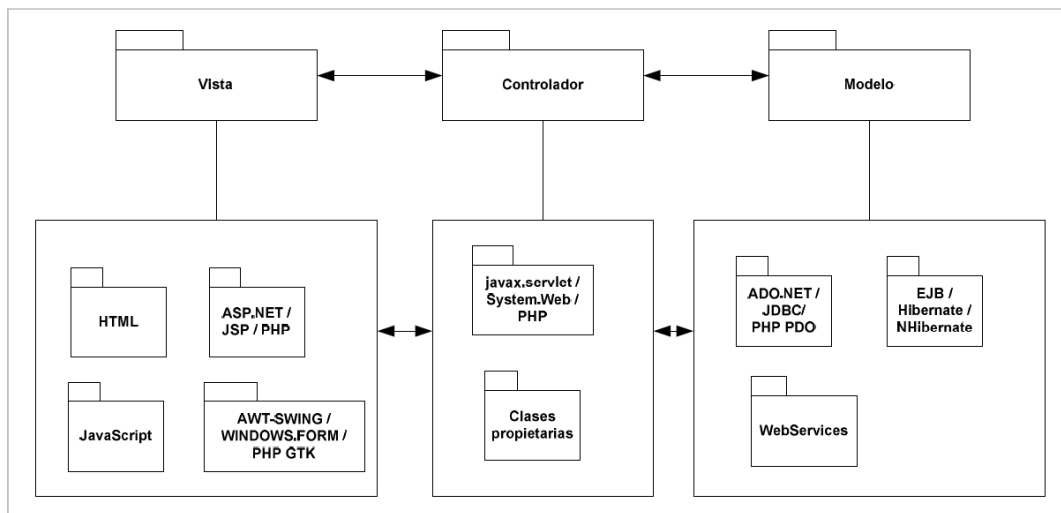


Imagen 7 - Esquema concreto de metapatrón mvc (Sanchez, 2008)

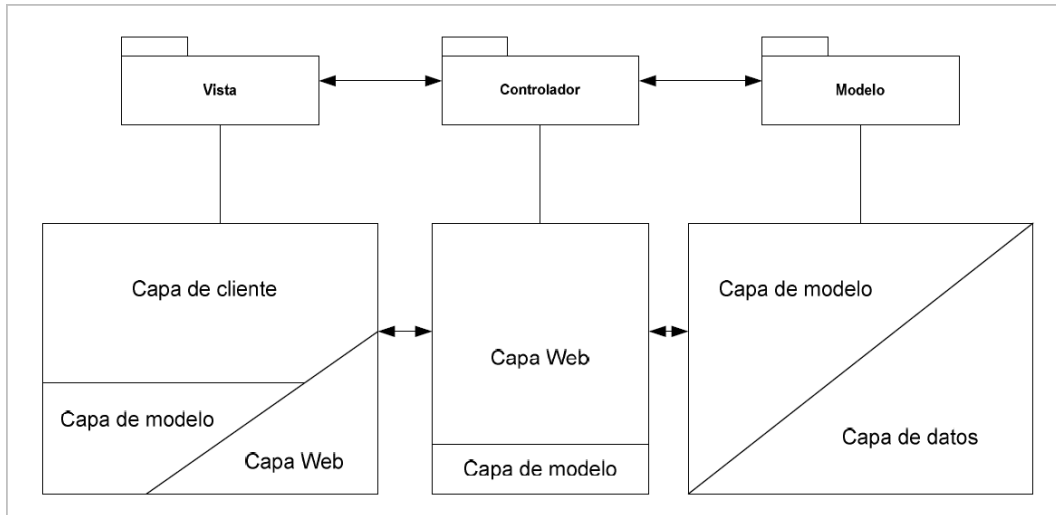


Imagen 8 - Esquema abstracto de metapatrón mvc (Sanchez, 2008)

## DAO

DAO o Data Access Object es un patrón de software que nos permite definir interfaces de comunes de acceso a distintos orígenes de datos, como bases de datos relacionales, bases de datos orientadas a objetos, archivos planos, archivos XML, entre otros. Se implementa como adaptador (o Adapter) entre el modelo de negocio de la aplicación y las distintas fuentes de datos que el DAO soporta o adapta. Para esto introduce una interfase que publica todos los métodos necesarios, y las distintas implementaciones de ésta interfase son las encargadas de llevar a cabo el acceso propiamente dicho a las distintas fuentes de datos.

## Estructura

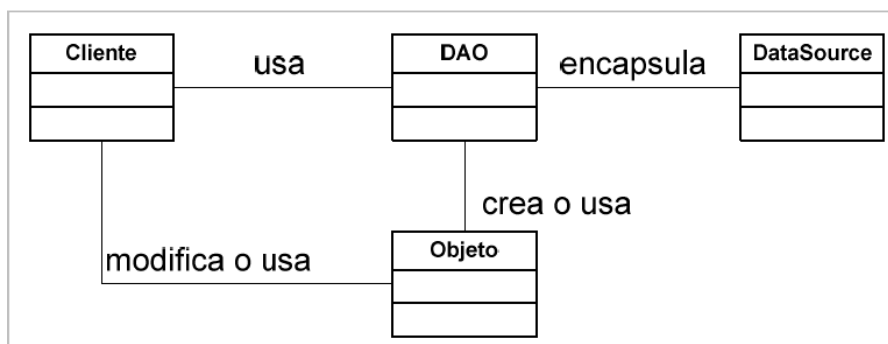


Imagen 9 - Patrón dao (Sanchez, 2008)

Cliente: hace referencia a la lógica de nuestra aplicación que necesita acceder a determinado Objeto de nuestro negocio, almacenado en el DataSource

DAO: clase principal del patrón. Se encarga de la totalidad de las funciones de actualización del Objeto involucrado.

Objeto: clase de nuestra lógica de negocio que necesita persistencia. Generalmente es un JavaBean. También se lo conoce como DTO o Data Transfer Object.

DataSource: es la fuente de datos que contiene la persistencia del Objeto. Puede ser una base de datos relacional, o cualquier otro contenedor de datos.

## Factory Method

### Contexto y problema

Debemos proporcionar una interfaz para crear objetos, pero dejar a las subclasses decidir qué clase instanciar

### Motivo

Los frameworks<sup>8</sup> utilizan clases abstractas para definir y mantener relaciones entre objetos, además de ser el responsable de crear estos objetos. Consideremos entonces un framework para aplicaciones que es capaz de presentar múltiples tipos de documentos a los usuarios. Dos abstracciones claves en este framework son las clases Aplicación y Documento. Ambas clases son abstractas y los clientes deben heredarlas para llevar a cabo la implementación específica. Para crear una aplicación de dibujo, por ejemplo, definimos las clases AplicacionDibujo y DocumentoDibujo. La clase Aplicacion es la responsable de manejar los objetos del tipo Documento, y los creará cuando que se necesario (opción Nuevo, Abrir, entre otros).

---

<sup>8</sup> Un framework, o marco de trabajo, es una plataforma de software reusable, diseñada para desarrollar aplicaciones, productos y soluciones de software (Riehle, 2000)

Debido a que la subclase DocumentoDibujo de Documento tiene una implementación específica, la clase Aplicacion no puede predecir que subclase de Documento instanciar (la clase Aplicacion solo sabe cuando crear un documento, pero no sabe qué tipo de Documento). Esto presenta un dilema: El framework debe crear objetos, pero solo conoce clases abstractas, que no se pueden instanciar.

El patrón Factory Method ofrece una solución. Encapsula el conocimiento de que subclase de Documento crear, y lo saca fuera del framework. Las subclases de Aplicacion redefinen un método abstracto CrearDocumento para retornar la subclase apropiada de Documento. Una vez que una subclase de Aplicacion es instanciada, puede crear objetos del tipo Documento específicos para la aplicación en cuestión. Llamamos a CrearDocumento un Factory Method porque es el responsable de “fabricar” un objeto

### Estructura

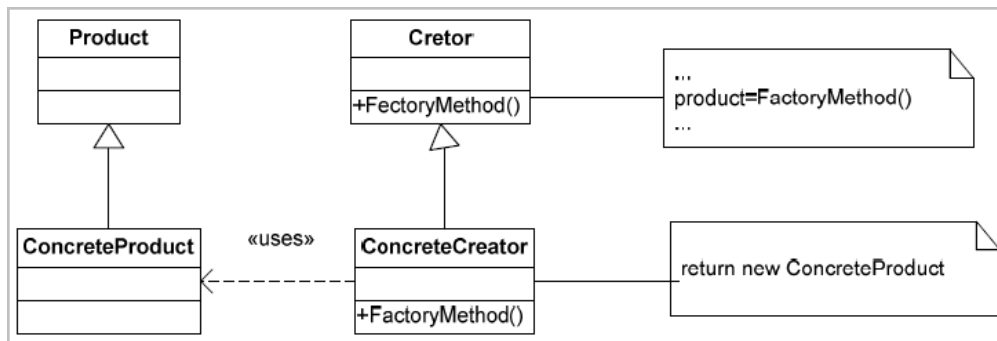


Imagen 10 - Patrón factory method (Sanchez, 2008)

### Abstract Factory

#### Contexto y problema

Debemos crear diferentes objetos, todos pertenecientes a la misma familia. Por ejemplo: las librerías para crear interfaces gráficas suelen utilizar este patrón y cada familia

sería un sistema operativo distinto. Así pues, el usuario declara un Botón, pero de forma más interna lo que está creando es un BotónWindows o un BotónLinux, por ejemplo.

El problema que intenta solucionar este patrón es el de crear diferentes familias de objetos.

El patrón Abstract Factory está aconsejado cuando se prevé la inclusión de nuevas familias de productos, pero puede resultar contraproducente cuando se añaden nuevos productos o cambian los existentes, puesto que afectaría a todas las familias creadas.

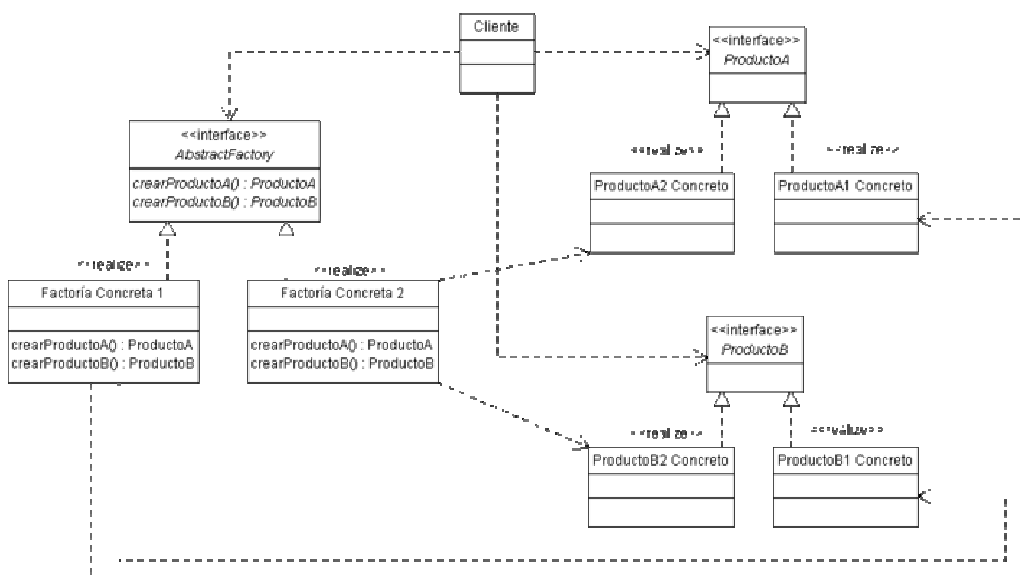


Imagen 11 - Aspecto estático de abstract factory (Sanchez, 2008)

### Estructura

Cliente: La clase que llamará a la factoría adecuada ya que necesita crear uno de los objetos que provee la factoría, es decir, Cliente lo que quiere es obtener una instancia de alguno de los productos (ProductoA, ProductoB).

AbstractFactory: Es de definición de la interfaces de las factorías. Debe de proveer un método para la obtención de cada objeto que pueda crear. ("crearProductoA()" y "crearProductoB()")

Factorías Concretas: Estas son las diferentes familias de productos. Provee de la instancia concreta de la que se encarga de crear. De esta forma podemos tener una factoría que cree los elementos gráficos para Windows y otra que los cree para Linux, pudiendo poner fácilmente (creando una nueva) otra que los cree para MacOS, por ejemplo.

Producto abstracto: Definición de las interfaces para la familia de productos genéricos. En el diagrama son "ProductoA" y "ProductoB". En un ejemplo de interfaces gráficas podrían ser todos los elementos: Botón, Ventana, Cuadro de Texto, Combo... El cliente trabajará directamente sobre esta interfaz, que será implementada por los diferentes productos concretos.

Producto concreto: Implementación de los diferentes productos. Podría ser por ejemplo "BotónWindows" y "BotónLinux". Como ambos implementan "Botón" el cliente no sabrá si está en Windows o Linux, puesto que trabajará directamente sobre la superclase o interfaz.

### Dao Factory

Cuando sabemos que la fuente de datos no es variable, podemos implementar la estrategia de obtener los objetos DAO a través de un DAO Factory, que combina el patrón DAO con el Factory Method.



## Estructura

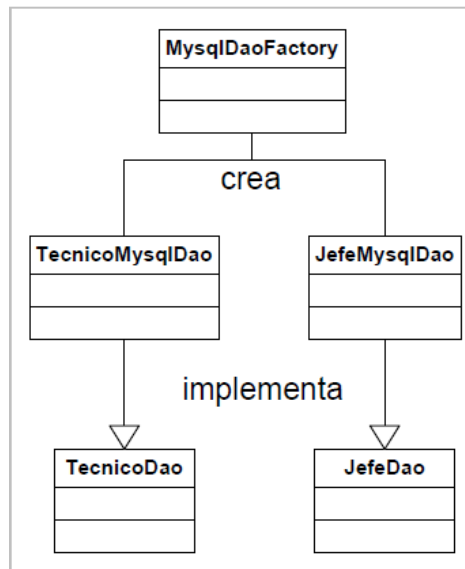


Imagen 12 - Esquema de patrón dao factory (Sanchez, 2008)

### A1.9 Historia de lenguaje PHP

PHP fue originalmente diseñado en Perl, con base en la escritura de un grupo de CGI binarios escritos en el lenguaje C por el programador danés-canadiense Rasmus Lerdorf en el año 1994 para mostrar su currículum vitae y guardar ciertos datos, como la cantidad de tráfico que su página web recibía. El 8 de junio de 1995 fue publicado "Personal Home Page Tools" después de que Lerdorf lo combinara con su propio Form Interpreter para crear PHP/FI.

Dos programadores israelíes del Technion, Zeev Suraski y Andi Gutmans, reescribieron el analizador sintáctico (parser en inglés) en el año 1997 y crearon la base del PHP3, cambiando el nombre del lenguaje a la forma actual. Inmediatamente comenzaron experimentaciones públicas de PHP3 y fue publicado oficialmente en junio del 1998.

Para 1999, Suraski y Gutmans reescribieron el código de PHP, produciendo lo que hoy se conoce como motor Zend. También fundaron Zend Technologies en Ramat Gan, Israel.

En mayo de 2000 PHP 4 fue lanzado bajo el poder del motor Zend Engine 1.0. El día 13 de julio de 2007 se anunció la suspensión del soporte y desarrollo de la versión 4 de PHP, a pesar de lo anunciado se ha liberado una nueva versión con mejoras de seguridad, la 4.4.8 publicada el 13 de enero del 2008 y posteriormente la versión 4.4.9 publicada el 7 de agosto de 2008. El 13 de julio de 2004, fue lanzado PHP 5, utilizando el motor Zend Engine 2.0 (o Zend Engine 2). La versión más reciente de PHP incluye todas las ventajas que provee el nuevo Zend Engine 2 como:

- Mejor soporte para la programación orientada a objetos, que en versiones anteriores era extremadamente rudimentario.
- Mejoras de rendimiento.
- Mejor soporte para MySQL con extensión completamente reescrita.
- Mejor soporte a XML ( XPath, DOM, entre otros. ).
- Soporte nativo para SQLite.
- Soporte integrado para SOAP.
- Iteradores de datos.
- Manejo de excepciones.
- Mejoras con la implementación con Oracle.

### *A1.10 Herramientas utilizadas para el desarrollo del sistema:*

#### IDE

Un entorno de desarrollo informático (en inglés integrated development environment) es un programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un sólo lenguaje de programación o bien, para varios.

Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI). Los IDEs pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes. El lenguaje Visual Basic, por ejemplo, puede ser usado dentro de las aplicaciones de Microsoft Office, lo que hace posible escribir sentencias Visual Basic en forma de macros para Microsoft Word.

Los IDE proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como PHP, C++, Python, Java, C#, Delphi, Visual Basic, entre otros.

#### Net Beans IDE

La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos. Un módulo es un archivo Java que contiene clases de Java escritas para interactuar con las APIs de NetBeans y un archivo especial (manifest file) que lo identifica como módulo. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que estos pueden ser desarrollados independientemente, las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de software.

NetBeans es un proyecto de código abierto de gran éxito con una gran base de usuarios, una comunidad en constante crecimiento, y con cerca de 100 socios en todo el

mundo. Sun Microsystems fundó el proyecto de código abierto NetBeans en junio de 2000 y continúa siendo el patrocinador principal de los proyectos.

### MS Project

Microsoft Project (o MSP) es un software de administración de proyectos diseñado, desarrollado y comercializado por Microsoft para asistir a administradores de proyectos en el desarrollo de planes, asignación de recursos a tareas, dar seguimiento al progreso, administrar presupuesto y analizar cargas de trabajo.

El software Microsoft Office Project en todas sus versiones (la versión 2010 es la más reciente) es útil para la gestión de proyectos, aplicando procedimientos descritos en el PMBoK (Management Body of Knowledge) del PMI (Project Management Institute).

## Formulario descriptivo del Trabajo Final de Graduación

### Identificación del Autor

Apellido y nombre del autor:	Carreño, Ignacio Luciano
E-mail:	ignacio.carre@gmail.com
Título de grado que obtiene:	Ingeniero en Sistemas de Información

### Identificación del Trabajo Final de Graduación

Título del TFG en español	Sistema Web para la Administración Apícola, BeeMore
Título del TFG en inglés	Web Based System for Beekeeping Management, BeeMore
Tipo de TFG (PAP, PIA, IDC)	PAP
Integrantes de la CAE	Frias, Fernando; Varas, Andrea
Fecha de último coloquio con la CAE	23 de Noviembre de 2012
Versión digital del TFG: contenido y tipo de archivo en el que fue guardado	1 archivo en formato PDF del TFG 49 archivos en 9 carpetas en formato PNG

### Autorización de publicación en formato electrónico

Autorizo por la presente, a la Biblioteca de la Universidad Empresarial Siglo 21 a publicar la versión electrónica de mi tesis. (marcar con una cruz lo que corresponda)

- Si, inmediatamente
- Si, después de ..... mes(es)
- No autorizo

Firma del alumno