UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

SISTEMA ADMINISTRADOR DE CITAS BIBLIOGRÁFICAS SOBRE LA PLATAFORMA PLONE

T E S I S

QUE PARA OPTAR POR EL GRADO DE:
MAESTRA EN CIENCIAS (COMPUTACIÓN)

PRESENTA:
ALEJANDRA MAQUEDA POLICARPO

DIRECTOR DE TESIS:
DR. SERGIO RAJSBAUM GORODEZKY
INSTITUTO DE MATEMÁTICAS

MÉXICO, D.F. MAYO 2014

Contenido

Agradecimientos

A mi madre quien ha sido mi gran ejemplo de vida. Agradezco su apoyo y motivación para el desarrollo de este proyecto.

Al Dr. Sergio Rajsbaum, director de la presente tesis, mi más amplio reconocimiento por su guía, apoyo y consejo.

A mis sinodales por sus valiosas recomendaciones para la mejora del presente documento y trabajo relacionado. En especial agradezco a la Dra. Adriana Ramírez Vigueras por gran su apoyo y asesoramiento en la construcción del sistema.

De igual forma agradezco el apoyo a Gildardo Bautista (técnico académico) del Instituto de Matemáticas por su asesoramiento y apoyo en cuestiones de programación y diseño de este sistema.

Finalmente agradezco a mi sobrina Guadalupe por su comprensión y paciencia.

Resumen

El presente documento exhibe la problemática de la falta de gestión de información ligada a las publicaciones científicas y citas relacionadas.

Ante estos inconvenientes se propone la construcción de un sistema que proporcione las vías y recursos necesarios que permitan gestionar de manera adecuada y centralizada la información referente a publicaciones científicas, que permita la adición de información extra concerniente a las citas y referencias, que proporcione los mecanismos de exploración efectiva entre conjuntos de publicaciones y que obtenga algunas métricas relacionadas al impacto y producción científica.

Dicho sistema plantea administrar las publicaciones de los investigadores del *Instituto de Matemáticas* e integrarse en un futuro al sistema *InfoMatem* de esta entidad. Motivo por el cual fue desarrollado bajo la misma plataforma que *InfoMatem*: Plone.

De esta manera el sistema sólo incluyó a las publicaciones de los investigadores del *Instituto de Matemáticas*.

La información utilizada por este sistema provino de la ejecución de *scripts* que realizaron la labor de búsqueda y tratamiento de las publicaciones y sus citas correspondientes.

Así una vez implementado el sistema, se pudo dar pie a la verificación de la información, a la correción de errores de origen, al cálculo de los indicadores de impacto y producción, a realizar consultas de mayor capacidad (llegar a citas y referencias) y a permitir ahorrar tiempo en el almacén de publicaciones mediante la importación rápida de los registros obtenidos.

De esta forma el sistema proporcionó las herramientas necesarias para alcanzar una buena/regular gestión de información referente a publicaciones de carácter científico y obtener además un mayor grado en el análisis de citas.

Se espera que manteniendo un control correcto de la información se pueda alcanzar una mayor consistencia de los indicadores de impacto y producción presentados.

Introducción

Antecedentes

En la actualidad la información que se obtiene de la gestión de publicaciones suele reflejar un índice de productividad o cierto nivel de impacto para un área de investigación en específico o en la carrera científica para algún investigador.

En particular el número de citas se utiliza para medir el impacto de los artículos, revistas, e investigadores y con frecuencia se incorporan en las decisiones de avance académico [22].

Las organizaciones que proporcionan y administran las publicaciones y la información asociada a ellas, son las llamadas bibliotecas digitales.

La Federación de Bibliotecas Digitales (1999), define *biblioteca digital* (Digital Library DL)de la siguiente manera:

Las bibliotecas digitales son organizaciones que proporcionan los recursos, incluyendo el personal especializado, para seleccionar, estructurar, ofrecer acceso intelectual, interpretar, distribuir, preservar la integridad y asegurar la persistencia en el tiempo de las colecciones de obras digitales de modo que sean fácilmente y económicamente disponibles para el uso de una comunidad definida o conjunto de comunidades [20].

Sin embargo, la gestión de estas bibliotecas sobre los recursos documentales no llega a ser del todo confiable, ya que la validez y los métodos detrás de la adquisición del número de citas han recibido una atención limitada [22].

El conocer cuántas y cuales publicaciones se relacionan con otras ha sido una problemática que requiere mayor atención y un estudio más detallado.

Particularmente el *Instituto de Matemáticas* (IM) no es la excepción en cuanto a esta problemática, dado que sus investigadores se han tenido que valer de diversas opciones no tan eficientes o con poco alcance para el manejo apropiado de sus publicaciones (*véase* sección 2.1). También de la información que puede resultar de un análisis más profundo del impacto de sus publicaciones.

Por lo que la presente tesis tratará de abordar esta problemática con ayuda de las herramientas necesarias que ayuden a la buena administración y presentación de publicaciones. Las publicaciones principales a consideración serán aquellas cuyos autores sean miembros de *Instituto de Matemáticas*. También serán consideradas aquellas publicaciones que tengan alguna relación con las publicaciones del *Instituto de Matemáticas*; las que hagan referencia a estos

artículos o sean referenciadas por éstos.

De aquí que se pretende obtener información un poco más precisa de las relaciones entre las publicaciones que ayuden al investigador conocer un poco la situación actual de su campo de estudio.

Se piensa que con una mayor gestión y atención en las publicaciones se pueden obtener información un poco más precisa y de utilidad al investigador.

Trabajo previo

Desde ya hace algunos años y en la actualidad, el *Instituto de Matemáticas* se ha preocupado por brindar una atención efectiva a los procesos de carácter administrativo y académico. Esto ha llevado a la búsqueda de soluciones ágiles que respondan a las necesidades de los usuarios (personal académico y administrativo) de acuerdo a los recursos y tecnologías disponibles.

Estas soluciones hacen referencia a diversas herramientas desarrolladas en el propio instituto a lo largo de varios años, implementadas bajo una misma plataforma: Plone.

Para el conocimiento del lector se listan las tesis desarrolladas dentro del *Instituto de Matemáticas*, en su mayoría por estudiantes del *Posgrado en Ciencia e Ingeniería de la Computación*:

- PloneVoteCryptoLib: Una biblioteca criptográfica para la implementación de elecciones en línea secretas y verificables por electores. Autor: Lázaro Clapp. Director de tesis: Dr. Sergio Rajsbaum. Programa: Licenciatura en Ciencias de la Computación, Facultad de Ciencias, UNAM. Nivel: Licenciatura. 2011 [6].

- Diseño e Implementación en Plone de un Sistema de Manejo de Solicitudes Mediante Flujos de Trabajo. Autor: Arturo Curiel. Director de tesis: Dr. Sergio Rajsbaum. Programa: Posgrado en Ciencia e Ingeniería de la Computación, UNAM. Nivel: Maestría. 2010 [9].

- Intercambio de Información y Web Semántico. Autor: David Méndez. Director de tesis: Dr. Sergio Rajsbaum. Programa: Licenciatura en Ciencias de la Computación, Facultad de Ciencias, UNAM. Nivel: Licenciatura. 2010 [27].

- Intercambio de información entre instituciones de la UNAM usando la Red Semántica. Autor: Hugo Rodríguez. Director de tesis: Dr. Sergio Rajsbaum. Programa: Licenciatura en Ciencias de la Computación, Facultad de Ciencias, UNAM. Nivel: Licenciatura. 2010 [35].

- Migración y nuevas características del sistema de votación electrónica del Instituto de Matemáticas de la UNAM. Autor: Iván Cervantes. Director de tesis: Dr. Sergio Rajsbaum. Programa: Posgrado en Ciencia e Ingeniería de la Computación, UNAM. Nivel: Maestría. 2009 [5].

- Desarrollo de un sistema de administración de procesos en Plone.Autor: Eduardo Espinosa. Director de tesis: Dr. Sergio Rajsbaum. Programa: Posgrado en Ciencia e Ingeniería de la Computación, UNAM. Nivel: Maestría. 2009 [11].
- Implementación de un Sistema de Votación electrónica como un producto sobre la plataforma Plone. Autor: Alexander Zapata. Director de tesis: Dr. Sergio Rajsbaum. Programa: Posgrado en Ciencia e Ingeniería de la Computación, UNAM. Nivel: Maestría. 2008 [45].
- Sistema sobre Plone para la captura y recolección de información curricular del Instituto de Matemáticas. Autor: Marco Antonio López Rabadán. Director de tesis: Dr. Sergio Rajsbaum. Programa: Posgrado en Ciencia e Ingeniería de la Computación, UNAM. Nivel: Maestría. Noviembre de 2007 [25].

En particular podemos decir que el sistema proporcionado por el *Instituto de Matemáticas* ofrece a sus usuarios la resolución de algunas solicitudes en menor tiempo y la presentación de información académica actualizada de los usuarios.

Provee de herramientas que ayudan a la gestión de ciertos procedimientos internos o externos del instituto. De los ejemplos en procedimientos internos destaca un sistema de votaciones para la elección de miembros de comisiones, funcionarios internos y representantes ante comisiones externas al instituto. De los ejemplos en procedimientos externos destaca la implementación de una red semántica que ayude en la búsqueda consistente de información entre diversas entidades de la UNAM.

Estos y algunos otros proyectos más conforman el actual sistema del *Instituto de Matemáticas*, llamado *InfoMatem*, un sistema que trata de resanar algunas problemáticas de organización y análisis de la información académica y administrativa. Esto mediante una plataforma robusta y flexible que permite la integración de nuevos componentes sin alterar los productos instalados y la plataforma misma.

De aquí que podemos desarrollar componentes independientes que conformen un sistema autónomo e integrable al actual sistema del *Instituto de Matemáticas*.

De esta manera el proyecto descrito en este documento pretenderá incorporarse al sistema actual del instituto y tratará de dar solución efectiva a los problemas del manejo de la información bibliográfica, obteniendo datos más precisos que sean de ayuda en el análisis de citas.

Planteamiento del problema

La problemática a abordar en términos generales se describe a continuación:

- Información insuficiente.

  Algunas bibliotecas digitales muestran la información descriptiva de sus artículos sin tomar en cuenta la información que se genera a partir de relaciones con otras publicaciones mediante referencias.

En este caso, es importante para el investigador y organizaciones académicas informarse acerca del impacto académico que poseen los artículos de determinado investigador.

El saber cuántas citas posee un artículo nos habla de la importancia de ese trabajo de investigación; un número considerable de artículos con un elevado número de citas nos habla de un alto grado de contribuciones de ese investigador, finalmente el número artículos publicados por un investigador nos habla del nivel de producción que tiene ese investigador considerando sus inicios.

En el caso de medir el impacto y la producción de publicaciones existen algunas métricas que nos ayudan a calcular estos dos aspectos, las cuales dependen ampliamente de los artículos generados y del número total de citas obtenidas de estos trabajos de investigación.

Es por ello que se considera importante al menos mostrar el número de citas por cada artículo y de manera total para cada investigador. De manera similar contemplar alguna métrica estándar de producción e impacto del investigador en cuestión.

- Inconsistencia numérica.

  Las publicaciones que se encuentran administradas por más de una biblioteca digital pueden presentar ciertas inconsistencias en el número de citas relacionadas. Esto en consecuencia puede afectar las métricas que miden el impacto y producción relacionadas al autor correspondiente.

  Las variaciones del conteo de citas pueden deberse a que posiblemente cada biblioteca digital gestiona individualmente sus recursos documentales sin compartir información con sus similares.

- Ausencia de relaciones entre publicaciones.

  Las bibliotecas digitales que tienen la capacidad de presentar información adicional respecto a citas y referencias, pueden no explotar este aspecto del todo. Esto puede afectar el panorama del usuario en cuanto a los recursos relacionados al artículo consultado.

  Es decir, las citas y referencias de los artículos pueden ayudar al lector a identificar de manera más clara cuales son los artículos más apropiados para abordar algún tema en específico, saber que artículo tiene más impacto e identificar a los investigadores se relacionan con esos artículos.

En general, podemos decir que estos problemas pueden derivarse de la gestión inadecuada de los recursos, ya que pese a que el sistema posea la capacidad de relacionar artículos, en ocasiones esto no se realiza completamente, los recursos pueden quedar relacionados parcialmente al conjunto de sus referencias.

Se necesita de una gestión adecuada y centralizada de la información de los recursos documentales.

Propuesta

Para tratar minimizar los problemas antes mencionados, se propone la construcción de un sistema que proporcione las vías y recursos necesarios que permitan gestionar de manera adecuada y centralizada la información referente a publicaciones científicas, que permita la adición de información extra concerniente a las citas y referencias, que proporcione los mecanismos de exploración efectiva entre conjuntos de publicaciones (los articulos y sus correspondientes conjuntos de citas y referencias) y que obtenga algunas métricas relacionadas al impacto y producción de artículos.

Para ello se prevé que se reúna información de diversas fuentes que complementen el listado de publicaciones de cada investigador. Esta recopilación de información se planea que se realice mediante la ejecución de *scripts* que además procesen y vinculen la información correspondiente a citas y referencias. Con ello el sistema obtiene la información necesaria para realizar las diversas acciones de consulta y calcular los índices de impacto y producción asociados a cada investigador.

Como plataforma de este sistema se contempla un sistema de gestión de contenidos o CMS (*Content Management System* por sus siglas en inglés).

En este caso se ha sugerido el CMS Plone, entre otras razones (descritas posteriormente), por una posible integración al sistema actual del *Instituto Matemáticas* (desarrollado en Plone).

Objetivos

Al implementar este sistema se busca obtener los siguientes beneficios:

- Concentrar la información en un mismo portal.
- El acceso al sistema no requiera de instalación de ningún tipo de software. El ingreso debe realizarse a través de un navegador Web.
- El investigador adquiera un espacio propio en el sistema para crear, editar o remover contenido. En este sentido el investigador podrá controlar la información referente a sus publicaciones.
- Realizar consultas a las publicaciones del los investigadores del IM y elementos relacionados a ellas.
- Evitar la búsqueda exhaustiva de las citas relacionadas a los trabajos de cada investigador.
- Alcanzar un mayor grado de veracidad en la información presentada (p.e. número de citas) al existir una mayor organización y control de las publicaciones.
- Que se pueda obtener información de interés a partir del número de citas y total de publicaciones: indicadores de impacto y producción científica.

- Que de las acciones de consulta a las publicaciones del investigador, sus citas y referencias, se pueda obtener información útil y de relevancia al investigador.

  Es decir que se pueda saber, en cuanto a las publicaciones del investigador y sus citas: cuáles de los artículos han sido citados el presente año, qué artículos (propios del investigador) cita determinado investigador (externo o interno), las citas recibidas en determinados años, etc.

  Y en cuanto a las publicaciones del investigador y a sus referencias: cuales han sido las referencias utilizadas en los últimos artículos, los artículos con referencias más longevas, las referencias más recientes de los últimos artículos, etc.

En general se pueda alcanzar un mayor grado de gestión de publicaciones, se puedan realizar busquedas más efectivas y profundas, que de ellas se pueda obtener informacion util y nueva, y que además se pueda obtener información estándar respecto al impacto y producción científica.

Estructura de la tesis

En esta sección se describe la estructura de la tesis correspondiente a la descripción de conceptos, herramientas y mecanismos en las etapas del desarrollo del sistema propuesto.

Parte I: Marco teórico

Capítulo 1. Definiciones y herramientas

En este capítulo se introducen los conceptos básicos, terminología y descripción de algunas herramientas utilizadas en el desarrollo del sistema.

Capítulo 2. Gestión, análisis y modos de acceso de publicaciones

Este capitulo describe los conceptos y herramientas relacionadas a la gestión de publicaciones, el análisis de citas y mecanismos de exploración. Describe algunos recursos auxiliares para la generación de firmas y define los tipos de publicaciones soportados de acuerdo al formato BibTex.

Parte II: Proceso de desarrollo del sistema

Capítulo 3. Análisis de requerimientos del sistema

Se contempla el levantamiento de requerimientos del sistema de acuerdo a los

recursos disponibles y necesidades del usuario.

Capítulo 4. Diseño del sistema

En este capítulo se describe el diseño del sistema a partir de los requerimientos del sistema. Se muestran modelos de interfaces y diagramas.

Capítulo 5. Desarrollo e integración

Este capítulo se detalla la implementación del sistema con base al diseño establecido. Se presentan las interfaces de usuario finales y su funcionamiento.

Capítulo 6. Resultados, conclusiones y trabajo futuro

Se presentan las características obtenidas tras la implementación del sistema. Los servicios y beneficios del sistema.

Se presentan las conclusiones con relación a la implementación, utilidad y limitaciones del sistema.

Finalmente se realiza un análisis de posibles mejoras al sistema. Esto denominado como trabajo futuro.

Apéndice A. Anexos

Esta sección se muestran algunos detalles importantes en el desarrollo del sistema, como son:

- El proceso de la instalación del CMS.
- Algunos detalles de la implementación de componentes.
- Descripción de utilerías o *scripts*.

Apéndice B. Acrónimos

En esta sección se muestran las siglas y acrónimos usados a lo largo del presente documento.

Parte I

Marco teórico

Capítulo 1

Definiciones y herramientas

1.1. Información y contenido

En la definición de Boiko Bob, información es lo que lo seres humanos transforman en conocimiento, cuando quieren comunicarse con otras personas. Es conocimiento hecho visible o audible, en palabras escritas, impresas o en el discurso [3].

Luego, para que la información se convierta en contenido, se le da una forma utilizable destinada a uno o más propósitos.

Entonces, la información se convierte en contenido después de que alguien la toma y trata de hacer algún uso de ella. Esto, mediante la adición de una capa de datos alrededor de ella.

La envoltura de datos típicamente describe la información y proporciona un contexto al hacerlo, hace la información tanto utilizable para los equipos y útiles para las personas.

Esta envoltura de datos es conocida como metadatos.

Los metadatos se describen perfectamente como datos sobre los datos, pero es mucho más que eso.

Los metadatos hacen el contexto y el significado de información suficientemente explícito para que una computadora puede manejarlo.

Por lo que cambiando de información a contenido, cambia el enfoque de la información a los metadatos que lo rodean.

1.2. Administración de contenidos

La administración de contenidos consiste en recopilar, gestionar y publicar contenido.

En este proceso se necesita saber cuál es el valor que se tiene para ofrecer, quién quiere qué partes de ese valor, y cómo éstos quieren que se entregue [3].

Por lo que un *sistema de gestión de contenidos* o CMS (*Content Management System* por sus siglas en inglés) sirve de ayuda para conseguir el material

adecuado a las personas indicadas en forma correcta.

De esta manera, el término *sistema de gestión de contenidos*, generalmente se refiere a una aplicación de software que se utiliza para crear, editar, gestionar y publicar contenido de forma consistentemente organizada [33].

Así las ventajas que se encuentran en el uso de un CMS son las siguientes:

- Separa el contenido de una página de su presentación [33].
- Permite a ciertos usuarios añadir y editar contenido [33].
- Aplica reglas sobre quién puede publicar qué y cuándo [33].
- Se puede aplicar reglas de negocio al contenido [33].
- Puede buscar e indexar información de manera "inteligente": puesto que el CMS puede llevar un registro de metadatos estructurados sobre el contenido, también se puede proporcionar capacidades de búsqueda mucho más inteligente y útil que simplemente una búsqueda de texto simple [33].

Así la clave de cualquier CMS es que proporciona una clara separación de los distintos elementos en él: seguridad, flujo de trabajo, plantillas, etc.

1.2.1. Componentes de un sistema de gestión de contenidos

Los elementos que componen cada una de las partes principales de un CMS son: el sistema de recolección, el sistema de gestión y el sistema de publicación.

El **sistema de recolección** es el responsable de los procesos que preceden a cualquier parte del contenido que este lista para su publicación. Convierte la información en bruto en un conjunto bien organizado de componentes de contenido.

Los procesos mencionados anteriormente incluyen lo siguiente:

- Autoría (creación): crear el contenido desde cero.
- Adquisición: recoger el contenido de una fuente existente.
- Conversión: convertir la información adquirida (o creada) conforme a los estándares aceptados por el sistema de contenido.
- Agregación: editar el contenido, dividirlo en componentes, y aumentarlo para que encaje dentro del sistema de metadatos deseado. En este proceso, los metadatos que se aplican al contenido permiten al sistema, un almacenamiento y recuperación (de contenidos) eficaz.
- Servicios de recolección: son programas y funciones del CMS que ayudan al proceso de recolección. El principal servicio que proporcionan, es la ayuda en la obtención de contenido en el repositorio. Por ejemplo, los servicios de recolección podrían producir los formularios web en el cual se especifica el contenido para los componentes.

El **sistema de administración** en un CMS es responsable del almacenamiento a largo plazo de los componentes de contenido y una variedad de otros recursos.

En un nivel más alto, permite saber lo que se ha coleccionado y su disposición.

Entonces, para cualquier pregunta razonable que se tenga sobre el contenido, publicaciones o sistema de recolección, se deben encontrar las respuestas disponibles en el sistema de gestión.

Para proporcionar esta capacidad, un sistema de gestión incluye:

- Repositorio: un lugar para almacenar el contenido. El repositorio es la pieza principal del sistema de gestión. El repositorio es el conjunto de bases de datos, directorios de archivos, y otras estructuras del sistema que almacenan el contenido del sistema, así como cualesquiera otros datos asociados al CMS.
- Administración: un sistema de administración para establecer y configurar el CMS.
- Flujo de trabajo: define conjuntos de pasos definidos para realizar operaciones sobre el contenido necesario, para que esté listo para ser publicado.
- Conexiones: un conjunto de conexiones (hardware o software) a otros sistemas dentro de la organización, que van desde redes y servidores en repositorios de datos.

El **sistema de publicación** es responsable de extraer los componentes de contenido y otros recursos fuera del repositorio y crear automáticamente publicaciones fuera de ellos.

Un sistema de publicación incluye:

- Plantillas de publicación: programas que crean publicaciones automáticamente.
- Servicios de publicaciones: un conjunto de herramientas para controlar lo que se publica y cómo se publica.
- Conexiones: herramientas y métodos utilizados para incluir los datos de otro sistema (no-CMS) en las publicaciones finales.
- Publicaciones web: la salida más común para la mayoría de los sistemas de gestión de contenidos.

1.2.2. Comparativas: sistemas de gestión de contenidos

En la actualidad existen diversas opciones en Sistemas de Gestión de Contenidos, tanto privativos como de fuente abierta.

En este caso se descartan los CMS de carácter privativo que involucran costo en la adquisición y que no permiten la libre modificación o distribución.

Además de ello, los sistemas de gestión de contenidos de fuente abierta también están diseñados para ser modulares. Al descargarlos uno obtiene el conjunto de características que fueron elegidas por un grupo de desarrolladores quienes decidieron qué debería contener el nivel básico de funcionalidad; este nivel básico es conocido como "núcleo" del sistema. Aunque existen además, una gran variedad de módulos que proporcionan funcionalidades adicionales para cada CMS y un rico entramado de desarrolladores que trabajan para crear más [31].

Así las comparativas quedan acotadas al conjunto de los sistemas de gestión de contenidos de fuente abierta.

Como primer punto de referencia se tiene a la organización Idealware[1] y su informe sobre sistemas de gestión de contenidos de código abierto para entidades no lucrativas, publicado en Diciembre del 2010, cubriendo WordPress, Joomla, Drupal y Plone (estos cuatro sistemas representan la mayor parte del mercado sin fines de lucro)

El informe clasifica a los cuatro sistemas de gestión contenidos en una variedad de criterios, como flexibilidad del sistema, facilidad de uso, disponibilidad de apoyo y algunos nuevos criterios, más notablemente, la accesibilidad Web y posicionamiento en buscadores [31].

En el cuadro 1.1 se muestran todos los criterios bajo los cuales los 4 CMSs fueron sometidos.

	WordPress	Joomla	Drupal	Plone
Facilidad de instalación y hosting.	●	●	●	◁
Facilidad de instalación: sitio simple	●	○	○	◁
Facilidad de instalación: sitio complejo.	●	●	○	○
Facilidad de uso: editores de contenido.	●	○	○	●
Facilidad de uso: administradoción del sitio.	●	○	○	○
Flexibilidad gráfica.	●	●	●	●
Accesibilidad y posicionamiento en buscadores o SEO.	○	●	◁	●
Flexibilidad estructural.	○	○	●	●
Funciones de usuario y flujo de trabajo.	◁	●	○	●
Comunidad/funcionalidad Web 2.0.	●	○	●	○
Extensión e Integración.	●	●	●	●
Seguridad.	◁	○	○	●
Soporte/resistencia comunitaria.	●	●	●	●

Cuadro 1.1: Comparativa entre CMS
● Excelente , ○ Sólido , ◁ Regular.

[1] Idealware es una organización sin fines de lucro, ofrece recursos bien investigados, imparciales y accesibles sobre software para ayudar a organizaciones no lucrativas a tomar decisiones inteligentes de software.

Donde cada uno de estos aspectos se refiera a:

- Facilidad de instalación y hosting. La facilidad de encontrar un proveedor de alojamiento Web (y potencialmente en el presupuesto que se tenga para ello) ya que no todos los CMS pueden ser alojados por cualquier empresa de hosting. Este aspecto, se refiere también a la facilidad en la instalación.
- Facilidad de instalación: sitio simple. Se refiere a la facilidad de configurar páginas con un esquema de navegación simple y la inclusión de elementos básicos en ellas.
- Facilidad de instalación: sitio complejo. La dificultad (o facilidad) de construir un sitio personalizado con mayores características de las que se proporcionan en el núcleo del sistema.
- Facilidad de uso: editores de contenido.
- Facilidad de uso: administración del sitio. La facilidad en la administración del sitio, esto incluye la edición y personalización de algunas partes del sitio, la administración del contenido y usuarios, respaldos y actualización del sitio, por nombrar algunas tareas.
- Flexibilidad gráfica. Se refiere al control adecuado del aspecto y diseño del contenido. Incluye la facilidad de encontrar, personalizar y actualizar temas principalmente.
- Accesibilidad y posicionamiento en buscadores u optimización de motores de búsqueda (SEO *Search Engine Optimization*). La accesibilidad se refiere a las características que hacen más fácil la utilización del sitio web especialmente para discapacitados visuales. Estas características tienen mucho en común con características que mejoran la probabilidad de que un sitio web aparezca en un lugar destacado en búsquedas de palabras clave en sitios como Google o Yahoo, esto es conocido generalmente como posicionamiento en buscadores o SEO.
- Flexibilidad estructural. Se refiere a la flexibilidad estructural que permite mostrar cierta información en diferentes maneras para diferentes listas en el sitio. Posibilita crear estructuras y tipos de contenido personalizados.
- Funciones de usuario y flujo de trabajo. Se refiere a la capacidad de asignar permisos de usuario para añadir, editar o publicar contenido mediante un criterio específico, también a la capacidad de controlar la visibilidad de cierto tipo de contenido a determinados usuarios.
- Comunidad/funcionalidad Web 2.0. Se refiere a las características que permiten la integración con alguna comunidad o red social.
- Extensión e Integración. Se refiere a la capacidad de creación de complementos para necesidades especialmente inusuales.

- Seguridad. Este aspecto define a un CMS más seguro en comparación a otro, si tiene un menor número de vulnerabilidades identificadas resueltas en menor tiempo.
- Soporte/resistencia comunitaria. Se refiere a la fuerza de la comunidad para dar soporte, resolver problemas y crear de complementos. Esto influye directamente en el uso más frecuente del sistema ya que mientras más soporte e información haya, más popular será el CMS utilizado.

De esta forma podemos conocer los aspectos más fuertes de cada CMS y su estado frente a otros CMS.

Los aspectos a considerar más importantes en el sistema a desarrollar son:

- Seguridad. Para brindar mayor protección y conservación de los datos.
- Extensión e integración. Para la integración de nuevas funcionalidades al sistema y la personalización de otros aspectos.
- Flexibilidad estructural. Para mostrar cierto tipo de información en diversas partes del sitio de manera entendible.
- Flexibilidad gráfica. Para controlar adecuadamente el aspecto y diseño del contenido.

De estos aspectos destacan dos CMS que cubren la mayoría de estas características de manera excelente: Drupal y Plone; de donde confirmamos la robustez y flexibilidad de la plataforma que sostiene al actual sistema del *Instituto de Matemáticas* desde hace ya varios años: Plone.

Recordamos que para este proyecto se opta por esta misma plataforma, debido, entre otras razones, por una posible integración al sistema del *Instituto de Matemáticas*, además de tener el mejor historial en seguridad hasta la fecha que cualquier otro CMS [30], aspecto que pudiera ser el más importante de todos los listados anteriormente.

1.3. Plone

Plone es un sistema de contenidos construido sobre Zope, un poderoso servidor de aplicaciones Web escrito en Python. Fue creado por Alexander Limi, Alan Runyan y Vidar Andersen, y la primera versión fue lanzada en 2001 [33].

Plone es un software de código abierto bajo la *Licencia Pública General* o GPL (*General Public License* por sus siglas en inglés), que garantiza la libertad de los usuarios [33].

Es fácilmente personalizable y extensible, ya que existen muchos productos adicionales se pueden integrar para añadir más características y cumplir con casi cualquier tipo de exigencia.

Específicamente, entre algunas características que distinguen a este CMS, se encuentran las siguientes:

- Empaquetado

  Plone mantiene instaladores para Windows, Linux y Mac. Productos de terceros y complementos, también vienen con instaladores. El mantenimiento de versiones de calidad de estos productos hacen que la instalación y manejo sea fácil. Además, cada nueva versión mantiene las rutas de migración y actualizaciones [33].

- Internacionalización

  Toda la interfaz de usuario de Plone se traduce en más de 35 idiomas, incluyendo el Chino, Japonés, Coreano e incluso los idiomas de derecha a izquierda como el Árabe y Hebreo, con facilidad [33].

- Usabilidad

  Plone ofrece altos niveles de usabilidad y accesibilidad. Plone proporciona una interfaz que es compatible con los estándares de la industria y el gobierno. Esto permite que los sitios construidos con Plone sean utilizados por personas con discapacidad visual. Además, esto proporciona el inesperado beneficio relacionado con el posicionamiento en buscadores o SEO que mejora la visibilidad de un sitio web en los resultados de diversos buscadores [33].

- Tematización fácil

  Plone separa el contenido de las plantillas reales o temas. Los temas se escriben en un excelente sistema de plantillas HTML, plantillas de página de Zope, y proporcionan una gran cantidad de CSS [33].

- Registro y personalización

  Plone cuenta con un sistema de registro de usuario completa. Los usuarios se registran en un sitio Plone, usando su propio nombre de usuario y contraseña. El administrador del sitio puede agregar y administrar los usuarios a través de la interfaz de usuario de Plone, y cada usuario puede crear, modificar su perfil personal y tablero de instrumentos, además de administrar las preferencias personales [33].

- Flujo de trabajo y seguridad

  El flujo de trabajo controla la lógica de procesamiento de contenido a través del sitio. Se puede configurar esta lógica a través de la Web utilizando herramientas gráficas. También se puede configurar listas de control de acceso para decidir quién tiene acceso a qué recurso y cómo los usuarios serán capaces de interactuar con él [33].

- Extensibilidad

  Dado que Plone es un software de código abierto, puede ser fácilmente alterado. Se puede cambiar y configurar casi cualquier aspecto de Plone para satisfacer casi cualquier necesidad. Innumerables paquetes y herramientas para Plone ofrecen una gran variedad de opciones para sitios muy pequeños o para empresas a gran escala [33].

- Personalización de contenido

  Los usuarios de un sitio Plone pueden añadir todo tipo de contenido. Los desarrolladores de Plone pueden crear sus propios tipos de contenido de modo que, casi cualquier tipo de contenido puede ser gestionado [33].

- Documentación

  El proyecto Plone mantiene una documentación, que se publica bajo la licencia de *Creative Commons*. Muchos equipos de usuarios y desarrolladores de todo el mundo han aportado documentación en otros idiomas aparte del idioma Inglés [33].

- Fiabilidad, crecimiento, y el futuro del CMS

  Cientos de desarrolladores están implicados de algún modo en el proyecto Plone en todo el mundo, trabajan cada día para mejorar las características del software y para resolver rápidamente cualquier error que pueda surgir. Plone ha existido por más de siete años, ha crecido, y lo ha hecho su comunidad y el número de usuarios en todo el mundo. De tal modo que se tiene la fuerte creencia que dicho comportamiento permanecerá en el futuro [33].

1.4. Servidor de aplicaciones Web

Un servidor de aplicaciones es un componente de software dedicado en una arquitectura de tres o múltiples capas que proporciona la lógica de aplicación (lógica de negocio[2]), permite la separación de la lógica de la aplicación de la funcionalidad de interfaz de usuario (capa cliente), la entrega de datos (servidor Web) y la gestión de datos (base de datos) [38].

En una arquitectura de tres niveles o multinivel, los servidores de aplicación típicamente hacen uso de varios servicios de *middleware*[3] que permiten la comunicación dentro y entre las capas.

Los servidores de aplicaciones generalmente proporcionan la base para la ejecución de aplicaciones distribuidas con garantías transaccionales sobre datos persistentes.

[2]Lógica de negocio. Se refiere a la entrada de procesamiento, creación, consultas y actualizaciones de datos [32], dice qué acciones a realizar bajo qué condiciones [39], también es responsable de la interacción con la memoria externa, almacenamiento y recuperación de datos [7].

[3]Middleware. Una capa de software presente en sistemas distribuidos, diseñados para simplificar el desarrollo de aplicaciones, proporcionar un conjunto común de servicios a ingenieros de software y proporcionar una abstracción de programación única para un conjunto de computadoras heterogéneas y plataformas de comunicación [2]. Facilita y gestiona la interacción entre aplicaciones a través de plataformas informáticas heterogéneas. Es la solución arquitectónica para el problema de la integración de una colección de servidores y aplicaciones bajo una interfaz de servicio común [1]. Proporciona soportes de software para la integración de componentes. Su propósito es conseguir que componentes independientes y distribuidos trabajen juntos [40].

En despliegues a gran escala, los sistemas pueden abarcar varias instancias de servidores de aplicaciones (clusters de servidores de aplicaciones).

Esto permite la distribución de las solicitudes del cliente a través de instancias de servidor de la aplicación con el propósito de equilibrio de carga.

Las aplicaciones de tres capas son más apropiadas para aplicaciones de gran tamaño y para aplicaciones que se ejecutan en la *World Wide Web*.

1.4.1. Zope y el CMF

Zope es un servidor de aplicaciones Web potente y flexible de código abierto desarrollado por *Zope Corporation* [33]. Es libremente distribuible bajo la licencia pública de Zope y puede ser modificado y mejorado por cualquier persona [44].

Zope, el producto de servidor de aplicaciones Web, es modular y sus piezas se pueden utilizarse independientemente de Zope.

Por eso necesitamos diferenciar entre Zope-el producto que es el servidor de aplicaciones Web y Zope-el-proyecto que produce una colección de componentes de software reutilizables escritos en Python.

Zope se ejecuta en todas las principales plataformas Unix, incluyendo Mac OS X, así como los sistemas operativos Microsoft Windows. Viene con su propio servidor Web, pero puede interoperar con un servidor Web existente, como Apache

Principales características de Zope-el producto:

- Base de datos de Objetos.

  Uno de los puntos fuertes de Zope es la base de datos de objetos Zope (ZODB) que almacena datos como objetos.

- Plantillas HTML/XML.

  En el desarrollo Web, es importante tener una manera fácil y productiva de producir lenguaje de marcado XML y HTML. Para ello Zope utiliza un sistema de plantillas XML conocido como ZPT (*Zope Page Template*).

  - Herramienta para generar HTML dinámico.
  - Pensado para permitir a diseñadores y desarrolladores trabajar en conjunto.
  - Utiliza etiquetas de HTML con atributos extra.
  - Permite utilización de macros[4] para reutilizar templates[5].

[4] Las macros definen una sección de una página que puede ser reutilizada en otras páginas. Una macro puede ser una página entera o sólo una parte de una página, como un encabezado o un pie de página. Después de definir una o más macros de una plantilla de página, se puede utilizar en otras plantillas de página [46].

[5] Una plantilla de página (*template*) es como un modelo de las páginas que se van a generar. En particular, es analizable por la mayoría de herramientas HTML [46].

Macros:

- Definen una parte de la página que puede ser usada en otros templates.
- Permiten insertar elementos dinámicos dentro de los macros.

Funcionamiento:

- Los templates utilizan el lenguaje TAL (*Template Attribute Language*).
- Las instrucciones de TAL se indican como atributos dentro de las etiquetas de HTML normales.

Para mayor información acerca del *Lenguaje de Atributo de Plantilla* o TAL visite http://docs.zope.org/zope2/zope2book/AppendixC.html

- Generación de formularios y validación.

  Zope ofrece una biblioteca de formularios (zope.formlib) que puede permitir tanto la construcción de formularios como la validación de los datos de entrada, según el esquema de datos del contenido (zope.schema).

- Internacionalización

  Zope brinda el medio para que los desarrolladores puedan crear aplicaciones multilingües (internacionalización i18n) y dependientes de la configuración regional (localización l10n) en zope.i18n (paquete que implementa varias APIs relacionadas con la internacionalización y localización). [6]

- Seguridad.

  Zope permite controles de seguridad muy finos. El mecanismo de publicación y el almacenamiento es centrada en el objeto y así es el sistema de seguridad. Tanto la política de seguridad y el sistema de autenticación son conectables y permiten el modelado de un sistema de seguridad flexible.

- Catalogación.

  Zope tiene un mecanismo de indización y catalogación que puede hacer que los datos almacenados en Zope sean fáciles de localizar.

- Pruebas.

  El código fuente de Zope contiene pruebas automatizadas que garantizan su estabilidad. Se trata de pruebas unitarias de bajo nivel, así como pruebas de integración y pruebas funcionales, basadas en HTTP.

[6] Internacionalización (i18n) se trata de hacer software (y sitios web) capaces de mostrar la interfaz de usuario y el contenido en varios idiomas, juegos de caracteres, etc.
La localización (l10n) se trata en realidad de la adaptación de una pieza de software internacionalizado a un lugar en particular (idioma y país).
El término "internacionalización" es usado algunas veces para referirse tanto a i18n como a l10n [47].

Se sabe que originalmente Zope fue desarrollado como un CMS independiente, pero con el tiempo no pudo satisfacer todas las necesidades de sus usuarios. Entonces *Zope Corporation* desarrolló el *Framework de Gestión de Contenido* o CMF (*Content Management Framework* por sus siglas en Inglés) como un proyecto de código abierto [33].

El CMF proporciona a los desarrolladores las herramientas necesarias para crear CMSs complejos [33]. Separa objetos de contenido de la presentación y lógica de la aplicación [44].

De esta manera, Plone se ubica en la parte superior del CMF, el cual se ejecuta en la parte superior de Zope.

Tanto Zope como el CMF son tecnologías clave que necesita Plone; sin ellos, no existiría Plone.

1.4.2. La base de datos de objetos de Zope

Por lo general, un objeto tiene dos componentes: estado (valor) y comportamiento (operaciones). Puede tener una estructura de datos complejos, así como operaciones específicas definidas por el programador.

Los objetos en un *Lenguaje de Programación Orientado a Objetos* OOPL (*Object-Oriented Programming Languages*) existen solamente durante la ejecución del programa; por lo tanto, se llaman objetos *transitorios*.

Una base de datos orientada a objetos puede extender la existencia de objetos para que permanentemente sean almacenados en una base de datos, y por lo tanto, los objetos se convierten en objetos *persistentes* que existen más allá de la terminación del programa y se pueden recuperar más adelante y ser compartidos por otros programas [10].

En otras palabras, las bases de datos orientadas a objetos guardan objetos persistentes permanentemente en almacenamiento secundario y permiten el intercambio de estos objetos entre múltiples programas y aplicaciones. Esto requiere la incorporación de otras características bien conocidas de los sistemas de gestión de bases de datos, tales como mecanismos de indización para localizar eficientemente los objetos, control de concurrencia para permitir que el objeto sea compartido entre los programas concurrentes y recuperación de fallas [10].

Así un sistema de bases de datos orientado a objetos típicamente se une con uno o más lenguajes programación orientados a objetos para proporcionar capacidades de objeto persistente y compartido.

En el caso de la *base de datos de objetos de Zope* o ZODB (*Zope Object Database*), se relaciona con el lenguaje de programación Python, así la ZODB es un sistema de persistencia para objetos de Python [21].

Los lenguajes de programación persistentes proporcionan servicios para escribir automáticamente objetos a disco y leerlos de nuevo cuando son requeridos por un programa en ejecución. Sin embargo una desventaja importante es que el programador debe administrar explícitamente objetos [21].

En el caso de la ZODB, ésta administra los objetos para mantenerlos en una caché, los escribe en el disco cuando son modificados y los quita de la memoria si no han sido utilizados desde hace tiempo [21].

Así entre otros aspectos que caracterizan a la ZODB se encuentran las siguientes:

- Atomicidad.

  No ocurren cambios parciales que podrían dejar a inconsistencias [44].

- Consistencia.

  Después de que una transacción se ha confirmado o anulado, la base de datos siempre se queda en un estado coherente [44].

- Aislamiento.

  Las múltiples transacciones realizadas al mismo tiempo no tienen impacto en los demás. La ZODB utiliza un mecanismo de control de concurrencia multi-versión o MVCC (*Multiversion Concurrency Control* por sus siglas en inglés). MVCC asegura que una operación de lectura utiliza una versión apropiada de los objetos que corresponden a un punto en el tiempo cuando la base de datos era definitivamente consistente [44].

- Durabilidad.

  Cuando los datos se escriben en la base de datos, persisten allí. Eso significa que los datos que sobrevivirán a un fallo del sistema [44].

- Transparencia.

  No hay métodos especiales que deban llamarse para almacenar o recuperar datos ni se tiene que implementar una interfaz especial para el cumplimiento de la persistencia [44].

- Deshacer.

  La ZODB no sólo admite una función de deshacer simple, realmente puede guardar cada revisión de todos los objetos almacenados, permitiendo control de revisión simple de datos almacenados [44].

- Almacenamiento extraíble.

  La ZODB puede almacenar datos persistentes de muchas maneras. El almacenamiento más común es *FileStorage* que almacena los datos en un único archivo llamado generalmente *Data.fs*. Otro almacenamiento popular es *DirectoryStorage* que almacena la revisión de cada objeto en un archivo separado.

  La ZODB también cuenta con un sistema de almacenamiento de red llamado *Zope Enterprise Objects* (ZEO) con la que una o muchas instancias de Zope (los clientes ZEO) delegan almacenamiento a un servidor de almacenamiento central (el servidor ZEO) sobre la red. Tal configuración se utiliza normalmente en entornos distribuidos o escala de alta disponibilidad [44].

De esta manera la base de datos de objetos de Zope resulta ser muy simple, transparente y robusta.

1.4.3. Python

Python es un lenguaje orientado a objetos interpretado y extensible, está disponible para múltiples plataformas (sistemas de hardware) y múltiples sistemas operativos (sistemas de software).

Python fue creado a principios de 1990 por Guido van Rossum en el *Stichting Centrum Mathematisch* como sucesor de un lenguaje llamado ABC.

En 2001, se formó la *Python Software Foundation* (PSF, ver http://www.python.org/psf/); una organización sin fines de lucro creada específicamente para poseer la propiedad intelectual relacionada con Python.

El interprete de Python es distribuido bajo una licencia de código abierto (*Python Software Foundation License*).

Es construido en torno a las clases y componentes reutilizables llamados *módulos*, esto permite mover fácilmente código Python de proyecto en proyecto.

Entre las ventajas y desventajas que se han encontrado en la utilización de este lenguaje, se listan las siguientes:

- **Desventajas**
  - Necesita hacer más cosas para el usuario final.

    Distribución todos los archivos que son necesarios para el intérprete.
  - Los archivos de origen son modificables por el usuario final.

    Los archivos de código fuente son archivos de texto simple. Esto significa que se encuentran en forma legible y por lo tanto modificable. La otra cara a esto que se puede detectar fácilmente qué archivos han cambiado mediante el uso de herramientas simples disponibles en casi cualquier entorno de sistema operativo.
  - No se encuentran errores hasta el tiempo de ejecución.

    Puesto que un archivo interpretado se procesa línea por línea, cualquier problema que exista en un módulo determinado no es encontrado hasta que ese módulo y la línea que contiene ese módulo, es cargado y procesado. Requiere un mayor grado de pruebas que el uso equivalente del código compilado.
- **Ventajas**
  - Es más fácil de depurar y mantener el código. Cuando se carga una pequeña sección de código, ésta se prueba libremente sin tener que compilar y vincular toda la aplicación, el proceso irá mucho más rápido. En adición se puede intentar probar pedazos de código sin tener totalmente terminado el programa.
  - Más fácil de actualizar el código de la aplicación rápidamente.

    El proceso de actualización de una aplicación se hace más eficiente. Una comparación de archivos es mucho más fácil que tratar de averiguar lo que ha cambiado en un archivo ejecutable binario.

- Los errores se pueden fijar rápidamente sin necesidad de redistribución completa.

  La capacidad de colocar simplemente un archivo de texto en un directorio y que la aplicación se comporte de una manera nueva o mejorada, es un gran beneficio no sólo al desarrollador, sino también para el personal de apoyo y mantenimiento del sistema del usuario.

- Puede ser embebido en otros lenguajes.

  La capacidad de ejecutar órdenes automáticamente, o para extender la funcionalidad de una aplicación mediante secuencias de comandos, es tan potente que se ha convertido en una parte estándar de aplicaciones más complejas. Python ofrece lo mejor de ambos mundos aquí. Permite mantener la seguridad y la velocidad de una aplicación compilada, mientras permite la posibilidad de personalización mediante scripts.

Capítulo 2

Gestión, análisis y modos de acceso a publicaciones

2.1. Software de gestión de referencias

En la actualidad en cuanto a software que ayude la "gestión" de publicaciones se tiene al llamado *software de gestión de referencias*.

El software de gestión de referencias (introducido por primera vez en la década de los 80's) realiza tareas independientes pero relacionadas.

En primer lugar, crea una base de datos de referencia, con cada artículo o recurso dado en un registro único. Los usuarios sólo tienen que introducir un artículo en la base de datos una vez y luego nunca tienen que escribir esa referencia de nuevo. Este enfoque minimiza los errores tipográficos, y ayuda a normalizar palabras clave de indización y formatos de referencia [14].

También se encarga de la búsqueda, permitiendo a los usuarios consultar bases locales y remotas, para artículos específicos.

Cabe señalar que en proceso de búsqueda en bases de datos remotas, interviene generalmente el protocolo Z39.50, que permite la búsqueda y recopilación de información cuyo resultado es dirigido a una base de datos existente en estos programas.

Este enfoque elimina la necesidad de aprender la sintaxis de varias bases de datos. Este tipo de software permite a los usuarios encontrar fácilmente referencias por palabra clave, autor o cualquier otro término de búsqueda. Los resultados, incluyen el extracto del artículo, por lo que pueden ser perfectamente importados en la base de datos local, ahorrar tiempo y evitar errores de transcripción [14].

Por otra parte, una variante de este tipo software es el basado en Web, el cual permite acceder, localizar, almacenar y recuperar referencias basadas en Web. Algunos de estos programas proporcionan una integración con navegadores Web junto con un servicio de envío de consultas para búsquedas en línea; mientras que otros utilizan un navegador preinstalado.

Una característica adicional general de este tipo de software es que esta enfocado en la recopilación específica de información (la que el usuario proporcione como entrada en una búsqueda).

Sin embargo este tipo software se limita a solo la adquisición de información referente a un artículo, no sus metadatos, como son el número de citas, índices, referencias; esa información sólo se muestra en las DLs, frecuentemente manera incompleta.

Por lo que este tipo de software resulta ser de capacidades limitadas en la gestión de información de elementos bibliográficos, aunque útil en el almacenamiento y búsqueda de publicaciones.

2.2. Analisis de citas

El área de investigación que utiliza la información contenida en publicaciones de investigación para obtener una mejor visión sobre la producción científica se denomina *bibliometría*.

Las unidades principales de medida en bibliometría son publicaciones científicas. Una publicación científica puede ser definida como cualquier tipo de material escrito, que contiene información respecto a actividades de investigación científica [37].

Así en bibliometría, el número de citas que recibe un artículo se considera un indicador de la importancia de la investigación original y se refleja en el factor de impacto de las revistas, en las cuales el artículo se publicó originalmente.

El factor de impacto o JIF (*Journal Impact Factor* por sus siglas en inglés) se calcula como el promedio de las citas recibidas en un año para todos los artículos publicados en una revista en los últimos 2 años [23].

Sin embargo, el análisis de citas se ha convertido en una forma polémica que los investigadores han tomado para demostrar el impacto de su trabajo y parece ser la principal forma de evaluación, comparación y clasificación de revistas científicas [15].

Para evaluar la calidad de la investigación de los científicos consideran una serie de indicadores bibliométricos.

La *productividad* y el *impacto* son las dos dimensiones principales de la calidad de la investigación. Algunos indicadores, como el número de artículos publicados y el número promedio anual de las publicaciones sólo reflejan la productividad de los científicos.

De esta manera, Hirsch propuso el índice h, que combina la productividad y el impacto.

El valor índice h de un científico se define como el máximo número natural h para el cual el científico tiene h artículos con al menos h citas. Esto da un límite inferior de h^2 citas al científico [26].

En el ejemplo mostrado en el cuadro 2.1 el índice h resulta ser 26.

Número de publicación	Número de citas
..	..
24	38
25	34
26	30
27	24
26	20

Cuadro 2.1: Cálculo del índice h=26.

En comparación con el número acumulado de citas, el índice h no se incrementa críticamente por un pequeño número de documentos altamente citados [26].

Además, Hirsch indica en el documento original que el índice h es concebido como un instrumento para evaluar a los investigadores en la misma etapa de sus carreras. No está pensada como una herramienta para comparaciones históricas [37].

De esta manera pueden surgir algunas otras situaciones en las que el indice h proporcione información que no describa la situación real de las publicaciones (p.e. importancia).

Ya que el índice h:

- No toma en cuenta el número de autores de un artículo. El índice h tienden a favorecer a grandes grupos de trabajo [37].
- Está limitado por el número total de publicaciones. Desventaja implícita a científicos con una corta carrera, independiente de la importancia de sus descubrimientos [37].
- Puede ser manipulado a través de auto-citas [37].
- Es un número natural que reduce su poder discriminatorio [37].

Con base al índice h, Hirsch define el índice de m o cociente m como valor de índice h de un científico dividido por el tiempo transcurrido (años) desde la primera publicación del científico focal [26]. Sin embargo, el índice m no es una medida de impacto sino de rendimiento, y no está exenta de problemas ya que es habitual que muchos investigadores muestren “espacios” de producción baja o nula, especialmente en sus inicios [16].

Así pues, para enfrentar estas situaciones, se muestran algunos otros índices enfatizando diferentes características.

Por ejemplo un indicador similar, que combina productividad e impacto es el índice de g. El índice de g de un científico se valora como el número más grande de artículos que recibe g^2 o más citas. Por definición para cada científico $g \geq h$ [26].

Asi, el índice g tiene un valor muy diferente que el índice h para quienes hayan publicado pocos artículos altamente citados [26].

Por otro lado, el índice *i10* es el número de publicaciones con al menos 10 citas [8]. Este simple pero útil indicador aparece en *Google Scholar* junto con el índice h desde Julio del 2011, donde cada indicador se calcula sobre todas las citas. También, aparece una versión más "actualizada" de ambas métricas de los últimos 5 años, donde el índice h más actual se obtiene como el máximo número h tal que h publicaciones tienen por lo menos h nuevas citas en los últimos cinco años, de manera similar, el índice *i10* corresponde al número de publicaciones que han recibido 10 nuevas citas en los últimos 5 años.

Ejemplos para el cálculo del índice h y el índice *i10* de las publicaciones del Dr. Sergio Rajsbaum G. según datos recopilados de *Google Scholar* son mostrados en el cuadro 2.2, cuadro 2.3 y figura 2.1:

Número de publicación	Número de citas
01	108
..	..
19	30
20	24
21	24
22	24
23	22
..	..
125	01

Cuadro 2.2: Índice $h = 22$.

Número de publicación	Número de citas
01	108
..	..
56	10
57	10
58	10
59	9
60	9
..	..
125	01

Cuadro 2.3: Índice *i10* $= 58$.

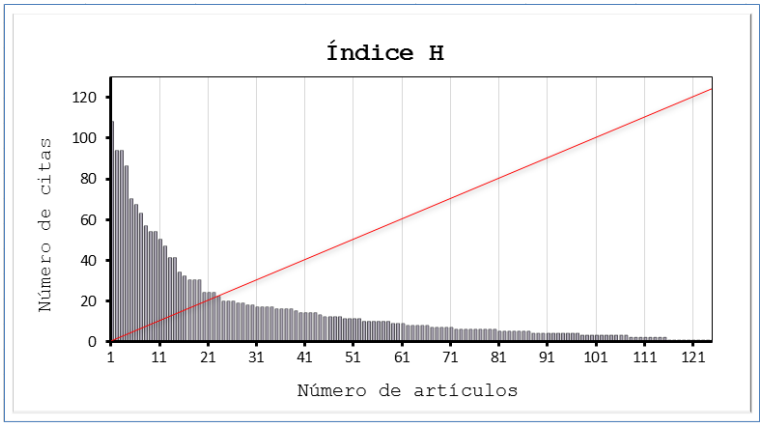

Figura 2.1: Gráfica de la distribución de citas ordenadas (de mayor a menor) correspondientes a las todas publicaciones del Dr. Sergio Rajsbaum. El valor del índice h toma lugar en la intersección con la función lineal, que representa el mínimo número de citas que se deben tener para un número de publicaciones dado. Así el valor del índice h será el máximo valor en el eje x (número de publicaciones) para el cual el número de citas esté por arriba o a la misma altura de la función lineal.

Un ejemplo más del cálculo del índice h y del índice *i10* para artículos publicados en los últimos cinco años (desde 2008) del Dr. Rajsbaum se muestra a continuación (cuadro 2.4):

Número de publicación	Número de citas	
01	30	
..	..	
07	14	
08	12	
09	12	
10	10	índice $h = 10$
11	10	índice *i10* $= 11$
12	08	
..	..	
31	01	

Cuadro 2.4: Índice h e índice *i10* de artículos publicados en los últimos cinco años.

Todos estos indicadores toman un significado cuando se intenta mostrar la importancia del trabajo científico, sin embargo, aunque alguna de estas medidas indique que un artículo sea importante, no significa necesariamente que lo sea, por lo que, en el estudio para evaluación de un trabajo de investigación se debe considerar más de un factor que corrobore el éxito de una publicación.

2.3. Normalización de publicaciones

Un aspecto importante que pudiera repercutir en el análisis de citas y productividad de un investigador es la información que se proporciona referente a sus trabajos de investigación.

En particular se ha encontrado, en algunas ocasiones, cierta irregularidad en los nombres de los autores y las instituciones donde éstos investigadores laboran.

En este sentido la normalización de la firma del autor tiene como fin elegir una única forma de firmar trabajos científicos que identifique claramente al investigador y le distinga de los demás para conocer su producción científica. Por su parte el nombre de la institución es muy importante como factor para identificar entre homónimos, y para que sean efectivos los indicadores bibliométricos por entidades y países.

En la actualidad existen diversas opciones que permiten resanar algunas consecuencias derivadas del uso de múltiples firmas, que van desde recomendaciones, estándares, uso de herramientas para la generación de firmas o la modificación de éstas en determinados portales.

Algunas de estas alternativas se describen a continuación:

- IRALIS (*International Registry of Authors-Links to Identify Scientists*). Es un sistema de estandarización de las firmas de los autores científicos [19].
- FECYT (*Fundación Española para la Ciencia y la Tecnología*). Recomendaciones para la correcta identificación de las publicaciones científicas. Normalización de autores e instituciones. [12].
- ORCID (*Open Researcher and Contributor ID*). Es un proyecto abierto, sin ánimo de lucro, comunitario, que ofrece un sistema para crear y mantener un registro único de investigadores y un método claro para vincular las actividades de investigación y los productos de estos identificadores [41].
- Researcher ID. Forma parte del conocido recurso *Web of Knowledge*. Permite generar un código identificador preciso de un autor para facilitar las búsquedas, identificar potenciales colaboradores, la creación de perfiles curriculares, y datos bibliométricos como el índice h y las citas recibidas. Además, la información ResearcherID se integra con la *Web of Science* y cumple ORCID, lo que le permite reclamar y mostrar sus publicaciones desde una sola cuenta [42].

En lo que respecta al sistema a desarrollar se tiene contemplado que el usuario tenga el control de sus propias publicaciones y con ello evitar la generación y/o uso de múltiples firmas. Esto por que la información almacenada en el sistema puede provenir de fuentes que manejen diversas firmas de un mismo autor.

Con estas correcciones el sistema puede proveer de mejores resultados en la realización de consultas. De lo contrario el usuario (investigador) debe al

menos tener en cuenta que posee más de una firma al momento de realizar sus consultas.

2.4. Navegación facetada

En la búsqueda de modos de acceso a la información se encuentran dos clasificaciones: búsqueda focalizada y búsqueda exploratoria.

En la búsqueda focalizada (específica), el usuario intenta localizar rápidamente elementos de información pertinentes sobre la base de su contenido.

En la búsqueda exploratoria (también llamada navegación) el usuario explora las relaciones entre elementos de una base de datos.

En particular (como se explica en la sección de Diseño 4.3) para el presente sistema se necesita el soporte de relaciones entre elementos de un conjunto de elementos dados, por lo que la mejor opción en este caso es el uso de búsquedas de tipo exploratorio.

En este modo de acceso seleccionado, las taxonomías estáticas (pertenecientes a este tipo), se basan en una jerarquía de conceptos que se pueden utilizar para seleccionar las zonas de interés y restringir la porción de la información que ha de ser recuperada [36].

También las taxonomías apoyan abstracción y son fáciles de entender por los usuarios finales.

Éstas las taxonomías no tratan con datos reales, sino con metadatos y, en el caso de documentos textuales, de los términos actuales encontrados en el documento.

Sin embargo, las taxonomías estáticas no son escalables para grandes bases de información, y el número promedio de documentos obtenido rápidamente llega a ser demasiado grande para la inspección manual.

Por lo que otras soluciones basadas en redes semánticas, ontologías generales y la Web Semántica parecen ser alternativas más poderosas que las taxonomías tradicionales.

No obstante los esquemas semánticos generales están diseñados para acceso mediante programación y se saben que son difíciles de entender y manipular. Además, la interacción con el usuario debe estar mediado por agentes especializados, y esto aumenta los costos, tiempo en el mercado y disminuye la transparencia y la flexibilidad de acceso de usuario.

Contrario a esto, las taxonomías dinámicas y sistemas de búsquedas facetadas se enfocan en el acceso exploratorio interactivo centrado en el usuario y proponen un enfoque integral en los modelos, la interfaz y cuestiones de interacción se consideran en conjunto.

Así, uno de los factores clave de este modelo es una búsqueda explícita de la simplicidad y minimalidad, en contraposición a las tendencias actuales de la investigación que tienden a soluciones de alta complejidad.

El esfuerzo en la reducción del modelo a sus componentes mínimos hace que sea fácilmente comprensible y utilizable por los usuarios finales sin necesidad de la mediación de cualquier agente.

Por lo que en la siguiente sección se definen los conceptos necesarios que ayuden a la construcción de una taxonomía dinámica que permita una navegación flexible e interactiva.

2.4.1. Taxonomías dinámicas

Las Taxonomías dinámicas o DT (*Dynamic Taxonomies*) también conocidas recientemente como sistemas de búsqueda facetada, son un modelo de gestión de conocimiento general sobre la base de una clasificación multidimensional de *objetos* de datos heterogéneos y se utilizan para explorar / buscar bases de información compleja de forma guiada pero sin restricciones a través de una interfaz visual. El modelo se basa principalmente en el acceso centrado en el usuario [36].

El esquema conceptual o *intensión* de una taxonomía dinámica es una jerarquía de *conceptos* (o términos) que va de lo más general a los conceptos más específicos y que no requieren ningún otro tipo de relación, además de inclusiones en conjuntos cada vez más grandes [36]. Ver figura 2.2.

Un **concepto** C es sólo una etiqueta abstracta que identifica todos los objetos clasificados en C [36].

Un concepto A está subsumido (incluido bajo) por un concepto B (A $\leq$ B) si el conjunto de instancias clasificadas en A está limitada intensionalmente de ser igual o un subconjunto del conjunto de instancias clasificadas en B: A $\subseteq$ B [36].

Los conceptos se definen por su *extensión* en lugar de por las propiedades específicas que presentan. Se definen dos tipos diferentes de extensión de un concepto C.

La extensión superficial de C (denotado por *shallowExtension*(C)) se define como el conjunto de objetos directamente clasificados en C [36].

La extensión profunda de C (denotado por *deepExtension*(C)) incluye todas las extensiones superficiales para el sub-árbol conceptual arraigado en C: o, de manera equivalente:

deepExtension(C) = {d | d $\in$ shallowExtension(C')$\wedge$ (C'$\leq$ C)} [36].

La extensión profunda del concepto raíz de la taxonomía incluye todo el universo U de objetos.

En una taxonomía dinámica, las relaciones entre conceptos que no sean subsunciones, se infieren a través de sólo la extensión, de acuerdo con la siguiente *regla de inferencia de extensional de base*:

Dos conceptos A y B están relacionados (denotado por A $\leftrightarrows$ B) si y sólo si hay al menos un objeto *d* en la extensión cual está clasificado al mismo tiempo en A o en uno de los descendientes de A y en B o en uno de los descendientes de B [36].

Es decir:

A$\leftrightarrows$B sii objects(A) $\cap$ objects(B) $\neq \emptyset$

Luego la regla de inferencia extensional base puede ampliarse para cubrir la relación entre un concepto determinado C y un concepto expresado por un subconjunto arbitrario S del universo: C se relaciona con S si sólo si hay al

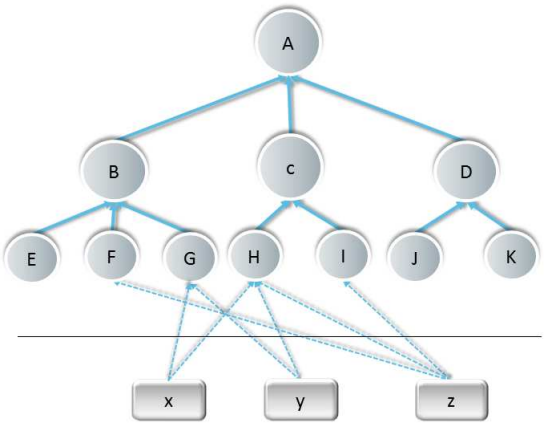

Figura 2.2: Taxonomía dinámica: *la intensión* se encuentra por encima de la línea divisora y la *extensión* por debajo de ella. Los círculos representan los conceptos y los objetos se representan por rectángulos. Arcos sólidos representan subsunciones, y los arcos de puntos representan clasificaciones.

menos un objeto o en S, que está también en objects(C), o equivalente, si y sólo si objects(C) $\cap$ S $\neq \emptyset$ [36].

Por lo tanto, la regla de inferencia de extensional puede inferir las relaciones no sólo para los conceptos de base, sino también para cualquier combinación lógica de los conceptos.

También puede utilizarse para producir un resumen conceptual de S según la taxonomía original mediante simplemente poda de la taxonomía todos esos conceptos C que no están relacionados con S, es decir, todos C $\notin$ RS (S).

Esto significa que la taxonomía original puede adaptarse y sintetizar cualquier subconjunto del universo (de ahí el término taxonomía dinámica).

Por otro lado, *el conjunto de conceptos relacionados a un conjunto S* se llama el conjunto de conceptos relacionados (RS(S)) y se define como:

RS(S) = {C | objects(C) $\cap$ S $\neq \emptyset$} [36].

Nos referimos a la cardinalidad asociada a *objects*(C) $\cap$ *S* como *contador relacionado*, rc (C | S): rc (C | S) = | *objects*(C) $\cap$ *S* | [36].

Por definición, rc (C | U) = | *objects* (C) |.

Ahora bien, en la interacción, el foco de interés inicial F_0 del usuario inicial es el universo U, es decir, todos los objetos en la base de información.

En el caso más simple, el usuario selecciona un concepto C en la taxonomía y hace un zoom sobre él.

La operación de zoom cambia el estado actual de dos maneras. En primer lugar, el foco actual F_i se convierte en F_{i-1} $\cap$ objects(C).

Luego, los objetos que no están en el foco se descartan. En segundo lugar, la representación del árbol de la taxonomía se modifica para resumir el nuevo

enfoque. Se conservan todos y sólo los conceptos relacionados con F_i y el contador de cada concepto se actualiza para reflejar el número de objetos en el foco de F_i.

La taxonomía reducida se deriva de la taxonomía inicial por poda todos los conceptos no relacionados con F_i, y es un resumen conceptual del conjunto de objetos identificados por F_i.

El proceso de recuperación puede ser visto como un proceso iterativo adelgazamiento de la base de información: el usuario selecciona un foco, lo que restringe la base de información descartando todos los objetos no en el enfoque actual.

Desde una perspectiva de la interacción persona-ordenador, el usuario es guiado efectivamente para alcanzar su objetivo por una lista clara y coherente de todas las alternativas posibles, y esta interacción es a menudo llamado adelgazamiento guiado o navegación guiada.

Las figuras 2.3, 2.4, 2.5 muestran como actúa la función zoom:

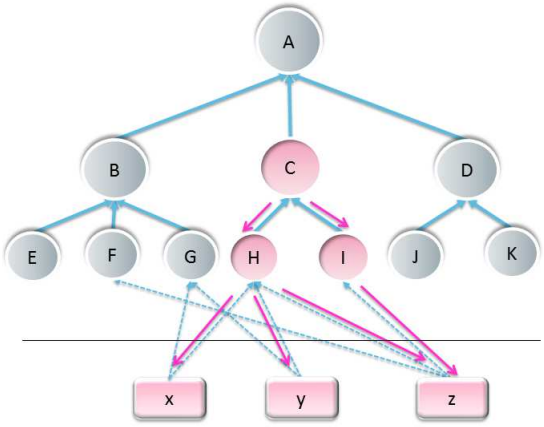

Figura 2.3: Cálculo de la extensión profunda del concepto C.

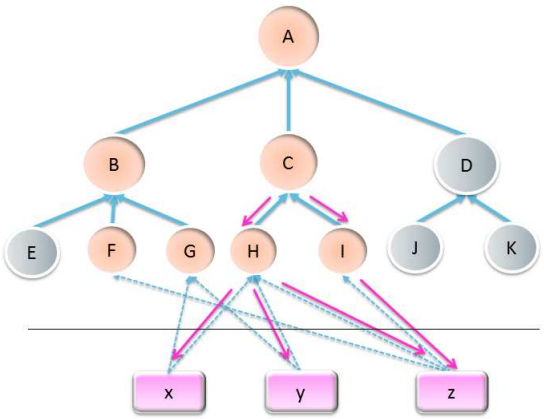

Figura 2.4: Cálculo del conjunto de objetos relacionados con el concepto C (en otras palabras RS(C)).

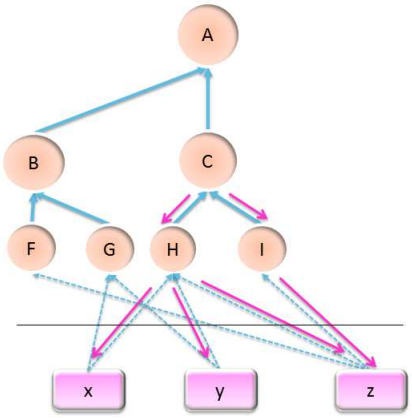

Figura 2.5: Taxonomía reducida para el foco/concepto C.

Para mayor información acerca de la teoría de taxonomías dinámicas y búsquedas facetadas consulte [36].

2.5. Formato BibTex

BibTex hace referencia a una herramienta y a un formato de archivo que se utiliza para describir y procesar las listas de referencias [13].

Una entrada de BibTeX consiste en un tipo, una clave y un número de

etiquetas que definen diversas características de la entrada BibTeX específica. Entre las etiquetas pueden ser por ejemplo: autor, título, año, etc. Algunas etiquetas son obligatorias para ciertos tipos de entradas BibTeX, algunos son opcionales [13].

Entre las etiquetas a usar se define un conjunto de etiquetas estándar, que pueden ser interpretados por BibTeX o herramientas de terceros. Aquellos que son desconocidos son ignorados por BibTeX.

Un ejemplo de este formato se muestra a continuación.

```
@inproceedings { 1835724,
author = {Herlihy, Maurice and Rajsbaum, Sergio},
title = {The topology of shared-memory adversaries},
booktitle = {Proceedings of the 29th ACM SIGACT-SIGOPS symposium on Principles of
distributed computing},
series = {PODC '10},
year = {2010},
isbn = {978-1-60558-888-9},
location = {Zurich, Switzerland},
pages = {105--113},
numpages = {9},
url = {http://doi.acm.org/10.1145/1835698.1835724},
doi = {10.1145/1835698.1835724},
acmid = {1835724},
publisher = {ACM},
address = {New York, NY, USA},
keywords = {combinatorial topology, fault-tolerance, set agreement, shared-memory},
}
```

Entre los tipos definidos para BibTex se tienen los siguientes:

- **@article**. Un artículo en un diario, revista o periódico que forma una unidad independiente con su propio título.

  Etiquetas requeridas: *author, title, journal, year.*
  Entradas opcionales: *volume, number, pages, month, note.*

- **@book**. Un libro de un solo volumen con uno o varios autores donde los autores comparten el crédito por el trabajo en su conjunto. Un libro con una editorial explícita.

  Etiquetas requeridas: *author, title, year.*
  Entradas opcionales: *volume, series, address, edition, month, note.*

- **@booklet**. Una obra similar a un libro pero sin una editorial formal o institución patrocinadora.

  Etiquetas requeridas: *author/editor, title,year.*
  Entradas opcionales: *howpublished, address, month, year, note.*

- **@conference**. Un artículo publicado en resúmenes de congresos. Esta entrada es idéntica a la entrada de *inproceedings* y se incluye compatibilidad con otros sistemas de formato de texto.

  Etiquetas requeridas: *author, editor, title, booktitle, year.*
  Entradas opcionales: *pages, organization, publisher, address, month, note.*

- **@inbook**. Una obra que forma parte de un libro, independiente con su propio título.

  Etiquetas requeridas: *author, title, booktitle, year.*
  Entradas opcionales: *volume, series, address, edition, month, note.*

- **@incollection**. Una contribución a una colección que forma una unidad autónoma con distinto autor y título.

  Etiquetas requeridas: *author, editor,title, booktitle, year.*
  Entradas opcionales: *editor, pages, organization, publisher, address, month, note.*

- **@inproceedings**. Un artículo publicado en resúmenes de congresos.

  Etiquetas requeridas: *author, editor, title, booktitle, year.*
  Entradas opcionales: *pages, organization, publisher, address, month, note.*

- **@manual**. Documentación técnica o de otro tipo, no necesariamente en forma impresa.

  Etiquetas requeridas: *author/editor, title, year*
  Entradas opcionales: *organization, location, month, year, note.*

- **@mastersthesis**. Tesis de maestría.

  Etiquetas requeridas: *author, title, institution, year.*
  Entradas opcionales: *address, month, note.*

- **@misc**. Un tipo de reserva para las entradas que no son de los tipos anteriores.

  Etiquetas requeridas: *author/editor, title, year.*
  Entradas opcionales: *howpublished, month, note.*

- **@phdthesis**. Tesis de doctorado.

Etiquetas requeridas: *author, title, institution, year.*
Entradas opcionales: *address, month, note.*

- **@proceedings**. Un volumen de resúmenes de congresos. Este tipo es muy similar a @collection. Soporta un campo opcional *organization* que sostiene la institución patrocinadora.

  Etiquetas requeridas: *editor,title, year.*
  Entradas opcionales: *publisher, organization, address, month, note.*

- **@report**. Un informe técnico, informe de investigación, o un libro blanco publicado por una universidad u otra institución. Se utiliza el campo *type* para especificar el tipo de informe. La institución patrocinadora va en el campo *institution.*

  Etiquetas requeridas: *author, title, type, institution, year.*
  Entradas opcionales: *number, address, month, note.*

- **@techreport**. Similar a @report excepto que el campo *type* es opcional.

  Etiquetas requeridas: *author, title, institution, year.*
  Entradas opcionales: *type, number, address, month, note.*

- **@unpublished**. Una obra con un autor y un título que no ha sido formalmente publicado. Se utilizan los campos *howpublished* y *note* para proporcionar información adicional en formato libre, si corresponde.

  Etiquetas requeridas: *author, title, note.*
  Entradas opcionales: *month, year.*

Para mayor información respecto a los tipos definidos BibTex y las entradas permitidas consulte [24] en la sección *Database Guide.*

2.6. Conclusiones

En este capítulo se describieron una serie de elementos y herramientas, que se ajustan y satisfacen algunas de las necesidades en el sistema a desarrollar.

También se introdujeron algunos conceptos fundamentales los cuales se manejarán a lo largo del desarrollo de este documento.

Entre los conceptos y herramientas que se han descrito se encuentra principalmente la definición y características de un sistema de gestion de contenidos. Ya que un sistema de gestión de contenidos es una plataforma flexible que permite la integración y el desarrollo de múltiples funcionalidades.

Parte II

Proceso de desarrollo del sistema

Capítulo 3

Especificación y análisis de requerimientos

En la literatura referente al desarrollo de software, se define un proceso de desarrollo de software como un conjunto de actividades que conducen a la creación de un producto de software [40].

En este caso se tomará el enfoque de la Ingeniería de Software Basada en Componentes CBSE (por sus siglas en inglés *Component-Based Software Engineering*). Cuyo modelo se presenta en la figura 3.1.

La ingeniería de software basada en componentes (CBSE) surgió a finales de los 90's como una aproximación basada en reutilización al desarrollo de sistemas de software [40].

De manera más formal la CBSE es el proceso de definir, implementar e integrar o componer en sistemas componentes [1] independientes débilmente acoplados [40].

Las infraestructuras de componentes proporcionan plataformas de alto nivel que reducen los costos del desarrollo de aplicaciones [40], en este caso esta plataforma hace referencia al CMS Plone, el cual ha de integrar algunos de los "productos" [2] existentes.

En el caso que no exista un componente que satisfaga algunas de las necesidades de los usuarios, se optará por el desarrollo e integración de ese componente al sistema.

[1] Un *componente de software* es un elemento de software que conforma a un modelo de componentes y que puede ser desarrollado independientemente y compuesto sin modificación de acuerdo a un estándar de composición [18].

Un *modelo de componentes* define la interacción específica y estándares de composición [18]. Estos estándares son utilizados por los desarrolladores de componentes para segurar que los componentes puedan interoperar [40].

Una *implementación de modelo de componentes* es el conjunto dedicado de elementos de software ejecutables requeridos para soportar la ejecución de componentes que conforman al modelo [40].

[2] Los productos son módulos que pueden dar a un sitio Plone funcionalidades adicionales y están hechos para la integración en la arquitectura de Plone [33].

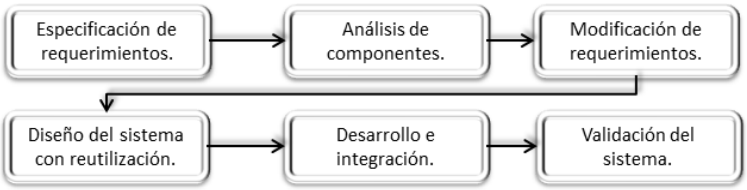

Figura 3.1: Modelo del proceso genérico para la Ingeniería de Software Basada en Componentes (CBSE) [40].

3.1. Requerimientos del sistema

Los requerimientos para un sistema son la descripción de los servicios proporcionados por el sistema y sus restricciones operativas. Estos requerimientos reflejan las necesidades de los clientes de un sistema que ayude a resolver algún problema [40].

3.1.1. Descripción de la problemática

Desde hace algún tiempo los investigadores del *Instituto de Matemáticas* han tenido la necesidad de organizar y administrar la información referente a sus publicaciones y citas relacionadas. Esta información, como se ha señalado al inicio de este documento, se sabe es de carácter importante ya que de ella se mide el impacto y la productividad de los trabajos de investigación.

También se ha dado a conocer que las principales fuentes de esta información (bibliotecas digitales) no son del todo fiables. Específicamente se presenta inconsistencia en el número de citas de un artículo y las referencias respectivas. También ocurre que pueden incluir nula información al respecto.

Otro problema ocurre cuando se obtiene un listado incompleto del total de publicaciones para un investigador, lo que conlleva a buscar en otras bibliotecas digitales (del área de investigación al que se pertenece) para obtener el completo listado de publicaciones.

Motivo por el cual se pretende desarrollar un sistema que permita la administración efectiva y centralizada de información bibliográfica que incorpore la asociación de citas, referencias y el conteo de las mismas.

En este sistema el investigador podrá almacenar su propio listado de publicaciones y tener en el mayor control de ellas, agregar citas y referencias y la posibilidad de hacer búsquedas flexibles de acuerdo a sus necesidades de información (búsquedas por año, colaborador, tipo de publicación, etc.).

Se piensa que teniendo una buena organización de la información se pueda alcanzar un mayor grado de fiabilidad en la información presentada y puedan realizarse analisis más completos de ella.

Ahora en cuestiones de referencia al sistema, éste será conocido como *Sistema Administrador de Citas Bibliográficas* o SACB.

3.1.2. Usuarios del sistema

Los usuarios participantes en el SACB:

- Usuario investigador. Este usuario es miembro del *Instituto de Matemáticas* y tiene una cuenta dentro del SACB. Puede añadir contenido referente a sus publicaciones y asociar las citas y referencias relacionadas con sus publicaciones. Puede consultar publicaciones propias o de otros usuarios investigadores.
- Usuario externo. Este usuario no tiene cuenta dentro del sistema y sólo puede ver contenido de carácter público. No puede editar ni agregar contenido.
- Usuario administrador. Este usuario tiene control sobre el sistema, puede otorgar permisos, agregar contenido, agregar usuarios y cambiar el estado del contenido.

3.1.3. Requerimientos del usuario y del sistema

Los requerimientos del usuario son declaraciones en lenguaje natural y en diagramas, de los servicios que se espera que el sistema proporcione y de las restricciones bajo las cuales debe funcionar.

Los requerimientos del sistema establecen con detalle las funciones, servicios y restricciones operativas del sistema.

En el proceso CBSE , los requerimientos del usuario se desarrollan inicialmente en forma de esquema en lugar de una manera detallada, debido a que los requerimientos muy específicos limitan el número de componentes que podrían satisfacer estos requerimientos. Sin embargo, se necesita un conjunto completo de requerimientos con el fin de que se puedan identificar para su reutilización tanto componentes como sea posible [40].

Lista 3.1.1. *Definición del requerimiento del usuario*

1. El sistema debe proporcionar cuentas de acceso a los usuarios investigadores.
2. El sistema debe permitir al usuario investigador agregar, modificar y eliminar de información blibliográfica.
3. El sistema debe permitir al usuario investigador añadir citas y referencias de cada artículo de su conjunto de publicaciones.

4. El usuario investigador podrá consultar las publicaciones de los investigadores pertenecientes al *Instituto de Matemáticas*. Esta consulta debe alcanzar sus correspondientes citas y referencias en caso de tenerlas.

5. Cualquier usuario (usuario externo) puede consultar las publicaciones que posean un estado de carácter público.

Lista 3.1.2. *Especificación de los requerimientos del sistema*

1. 1.1. El sistema debe proveer cuentas de acceso donde se requiera: nombre de usuario y contraseña. Previamente el administrador debe optar por agregar a todos los usuarios él mismo o permitir a los usuarios registrarse al sistema.

   1.2. El sistema debe proveer de un lugar de trabajo al usuario en el sistema, esto es una carpeta propia para gestionar sus recursos.

2. 2.1. El sistema debe proporcionar los servicios para permitir a los usuarios agregar, modificar o eliminar contenido referente a sus propias publicaciones.

   2.2. Para ello el sistema debe contar con la interfaz/interfaces apropiadas que ayuden a realizar cada acción.

   2.3. En cuanto a contenido se refiere, el sistema debe dar soporte a todo tipo de publicaciones (por ejemplo: artículos, libros, actas de conferencias, etc.)

3. 3.1. El sistema debe permitir a los usuarios investigadores la asociación de información extra (citas y referencias) a sus publicaciones.

4. 4.1. En el apartado de consultas, el sistema debe mostrar la lista publicaciones disponibles.

   4.2. El sistema debe permitir realizar consultas a través de un interfaz que se apoye de filtros o facetas.

   4.3. Los valores de las facetas se obtendrán de los valores de los atributos de las publicaciones (atributos como autor, año de publicación, etc.).

   4.4. Las publicaciones resultantes serán aquellas que satisfagan las restricciones impuestas por los valores de las facetas.

   4.5. La consulta de publicaciones debe alcanzar a sus respectivas citas y referencias.

      4.5.1. El sistema debe proveer apartados de consulta relacionados para estos 3 grupo de publicaciones.

      4.5.2. Un apartado dedicado a las publicaciones del investigador, las que llamaremos *publicaciones principales*.

      4.5.3. Un apartado dedicado a las citas de las publicaciones del investigador.

4.5.4. Un apartado dedicado a las referencias utilizadas por las publicaciones del investigador.

4.5.5. El sistema debe proveer resultados conforme a los valores de las facetas seleccionados.

4.5.6. El dominio de consulta de las citas y referencias deben depender de las publicaciones principales resultates y viceversa. Esto proporciona una mayor consistencia de los elementos mostrados en cada grupo de publicaciones; todos los elementos están en relación a los resultados obtenidos.

5. 5.1. El sistema debe proporcionar a los usuarios externos la posibilidad de consulta de las publicaciones si estas tienen un estado público.

3.1.4. Requerimientos funcionales

Son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que éste debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares. En algunos casos, los requerimientos funcionales de los sistemas también pueden declarar explícitamente lo que el sistema no debe hacer [40].

Lista 3.1.3. *Requerimientos funcionales*

1. El sistema no debe permitir que un usuario diferente al administrador añada y elimine usuarios.

2. El sistema debe proporcionar al usuario investigador agregar contenido de publicaciones: mediante el llenado de un formulario o de la importación de un archivo en un formato determinado (BibTex por ejemplo) al SACB.

3. El sistema deberá permitir la adición de contenido de una publicación si el formulario es llenado correctamente o si no encuentra errores en el archivo de formato especificado.

4. El sistema debe permitir al usuario investigador editar sus publicaciones, a fin de actualizarlas, corregirlas o añadirles información extra concerniente a citas o referencias.

5. El sistema no debe permitir que un usuario investigador edite contenido perteneciente a otro usuario investigador.

6. El sistema no debe permitir la edición y carga de contenido a usuarios externos al sistema.

7. El sistema debe permitir consultar las publicaciones a los usuarios, tanto internos como externos del sistema, mediante una interfaz que permita la realizar búsquedas exploratorias.

8. La interfaz de consulta debe proporcionar características que permitan una visualización práctica de las publicaciones, búsquedas flexibles mediante el uso de filtros o facetas (con base a atributos generales de las publicaciones) y alternativamente búsquedas por coincidencia de palabras.

9. El sistema debe encargarse de que los resultados arrojados por la consulta de publicaciones de un investigador tengan influencia sobre los conjuntos señalados como citas y referencias (de manera inversa, cuando se explora el conjunto de citas se repercute en los conjuntos restantes). Esto a fin de que se tengan resultados consistentes y relacionados entre sí.

10. El sistema debe calcular el índice h y el índice *i10* de cada autor. Estos datos deben ser proporcionados por la interfaz de consulta de publicaciones.

3.1.5. Requerimientos no funcionales

Son restricciones de los servicios o funciones ofrecidos por el sistema. Incluyen restricciones de tiempo, sobre el proceso de desarrollo y estándares. Los requerimientos no funcionales a menudo aplican al sistema en su totalidad. Normalmente apenas se aplican a características o servicios individuales del sistema [40].

Lista 3.1.4. *Requerimientos no funcionales.*

1. El acceso al sistema no debe tardar más de 3 minutos.

2. La consulta de las publicaciones debe entregar los resultados en un tiempo razonable conforme a la cantidad de datos a explorar. En conjuntos de cientos de elementos no debería tardar más allá de 20 segundos; en conjuntos menores a 100 elementos no más de 10 segundos.

3. Análogamente la importación de publicaciones al sistema no deberıá tomar más de un minuto para conjuntos menores a 100 elementos.

4. Por otro lado, las acciones sobre elementos (guardar, eliminar, cambiar estado) no deberían tomar más de 30 segundos en efectuarse.

Estas estimaciones de tiempo son las mínimas proyectadas para el sistema a desarrollar. Por lo que se espera tener respuestas en menor tiempo.

3.1.6. Requerimientos de dominio

Son requerimientos que provienen del dominio de aplicación del sistema y que reflejan las características y restricciones de ese dominio. Pueden ser funcionales y no funcionales.

Se derivan del dominio de aplicación del sistema más que de las necesidades específicas de los usuarios. Son importantes porque reflejan los fundamentos del dominio de la aplicación [40].

Lista 3.1.5. *Requerimientos de dominio*

1. El sistema debe contar con la información mínima de cada tipo de publicación. El sistema debe requerir los atributos marcados como obligatorios de cada tipo de publicación existente.

2. El dominio de consulta de las citas y referencias deben depender del resultado de las publicaciones principales y viceversa. Esto con el fin de que los resultados arrojados sean consistentes y relacionados entre sí.

3.1.7. Diagrama de casos de uso

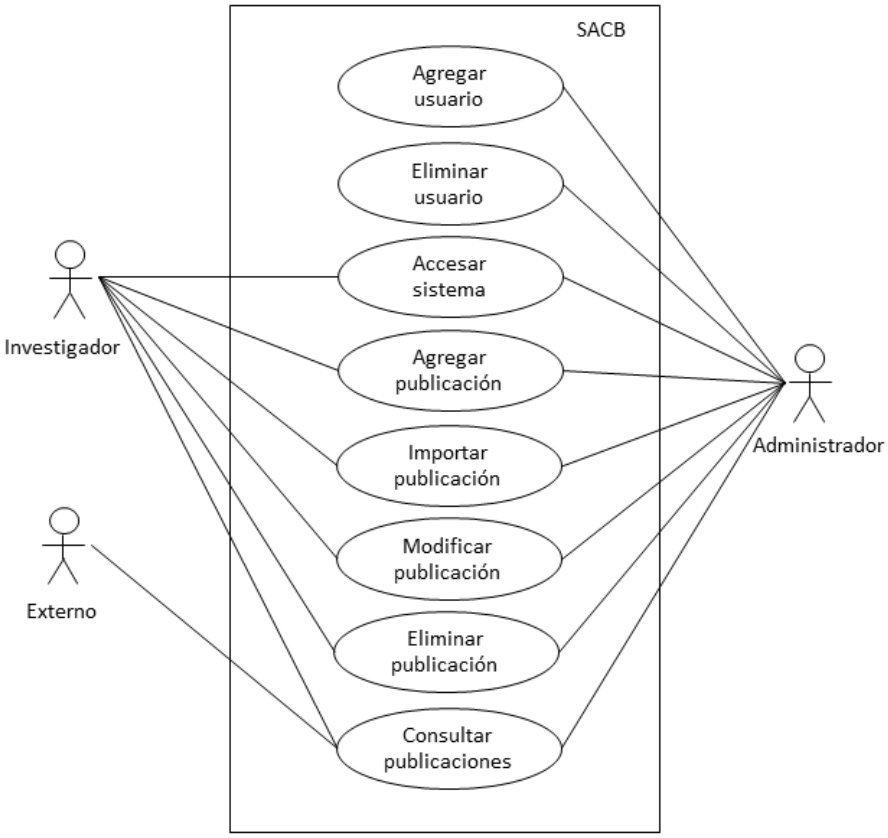

Figura 3.2: Diagrama de casos de uso del sistema SACB.

3.1.8. Casos de uso

Agregar publicación

Descripción breve: Esta transacción le permite a un usuario investigador o al administrador del sistema agregar una publicación al sistema vía un formulario.
Actor: Investigador, administrador.
Precondiciones: El usuario debe existir, su directorio y las carpetas de las publicaciones deben estar creadas. El usuario debe poseer permisos de administración del contenido.
Requerimientos que satisface: De la lista 3.1.1 satisface 2 y 3; de la lista 3.1.3 satisface a 2, 3 y 6; finalmente de la lista 3.1.5 satisface a 1.

Flujo de eventos: Agregar una publicación.

1. Dentro del directorio del investigador , el investigador o administrador selecciona una carpeta (a saber: *publicaciones, citas* y *referencias* según sea el caso) para almacenar las publicaciones correspondientes.
2. El sistema muestra el contenido de esa carpeta.
3. El investigador o administrador da clic en el menú *agregar publicación.*
4. El sistema muestra los diferentes tipos de publicaciones disponibles.
5. El investigador o administrador selecciona un tipo de publicación.
6. El sistema proporciona un formulario para llenar los campos obligatorios y opcionales del tipo solicitado.
7. El investigador o administrador llena el formulario y se da clic en guardar.
8. El sistema valida el formulario y guarda la publicación.

Postcondiciones: La publicación se ha almacenado en el sistema.

Importar publicación

Descripción breve: Esta transacción le permite a un usuario investigador o al administrador del sistema agregar una o varias publicaciones al sistema vía un archivo BibTex.
Actor: Investigador, administrador.
Precondiciones: El usuario debe existir, su directorio y las carpetas de las publicaciones deben estar creadas. El usuario debe poseer permisos de administración del contenido.
Requerimientos que satisface: De la lista 3.1.1 satisface a 2 y 3; de la lista 3.1.3 a 2 y 3.

Flujo de eventos: Importar publicaciones.

1. Dentro del directorio del investigador, el investigador o administrador selecciona una carpeta (a saber: *publicaciones, citas* y *referencias* según sea el caso) para almacenar las publicaciones correspondientes.
2. El sistema muestra el contenido de esa carpeta.
3. El investigador o administrador da clic en la opción *importar publicación*.
4. El sistema muestra una interfaz donde se permite la carga de un archivo en formato BibTex.
5. El investigador o administrador selecciona el archivo en el formato establecido y se da clic en aceptar.
6. El sistema valida el archivo y guarda las publicaciones definidas en el archivo.

Postcondiciones: La(s) publicación(es) se ha(n) almacenado en el sistema.

Modificar publicación

Descripción breve: Esta transacción le permite a un usuario investigador o al administrador del sistema modificar una publicación.
Actor: Investigador, administrador.
Precondiciones: Debe existir al menos una publicación en la carpeta seleccionada. El usuario debe existir, su directorio y las carpetas de las publicaciones deben estar creadas. El usuario debe poseer permisos de administración del contenido.
Requerimientos que satisface: De la lista 3.1.1 satisface 2 y 3; de la lista 3.1.3 satisface 4, 5 y 6.

Flujo de eventos: Modificar publicación.

1. Dentro del directorio del usuario, se selecciona la carpeta (a saber: *publicaciones, citas* y *referencias* según sea el caso) que contiene la publicación a modificar.
2. El sistema muestra el contenido de la carpeta.
3. El investigador o administrador selecciona la publicación a modificar.
4. El sistema muestra el formulario correspondiente a la publicación seleccionada.
5. El investigador o administrador edita la publicación mostrada y se da clic en aceptar.
6. El sistema valida y guarda los cambios de la publicación.

Postcondiciones: Los cambios en la publicación se han guardado.

Eliminar publicación

Descripción breve: Esta transacción le permite a un usuario investigador o al administrador del sistema eliminar una o varias publicaciones.
Actor: Investigador, administrador.
Precondiciones: Debe existir al menos una publicación en la carpeta seleccionada. El usuario debe existir, su directorio y las carpetas de las publicaciones deben estar creadas. El usuario debe poseer permisos de administración del contenido.
Requerimientos que satisface: De la lista 3.1.1 satisface a 2.

Flujo de eventos: Eliminar publicación.

1. Dentro del directorio del usuario, El investigador o administrador selecciona la carpeta (a saber: *publicaciones, citas* y *referencias* según sea el caso) que contiene la publicación a eliminar.
2. El sistema muestra el contenido de la carpeta.
3. El investigador o administrador selecciona la(s) publicación(es) a eliminar y se da clic en el botón *eliminar*.
4. El sistema elimina la(s) publicación(es) seleccionada(s).

Postcondiciones: El sistema elimino la publicación seleccionada.

Consultar publicaciones

Descripción breve: Esta transacción le permite a un usuario investigador, al administrador del sistema o a un usuario externo consultar las publicaciones pertenecientes y relacionadas a un investigador del *Instituto de Matemáticas*.
Actor: Investigador, administrador, usuario externo.
Precondiciones: Deben existir publicaciones del investigador a consultar.
Requerimientos que satisface: De la lista 3.1.1 satisface a 4, 5 (consecuentemente también satisface de la lista 3.1.2 a 4); de la lista 3.1.3 satisface a 7, 8, 10 y 9; de la lista 3.1.5 satisface a 2.

Flujo de eventos: Consultar publicaciones.

1. Dentro del directorio del usuario, el investigador o administrador selecciona la opción *Consultar publicaciones*.
2. El sistema muestra la interfaz de consulta de los grupos de publicaciones almacenados en directorio del usuario (a saber: *publicaciones, citas* y *referencias* según sea el caso). Donde cada grupo posee su propia navegación facetada.

   La interfaz inicial es la correspondiente a las publicaciones del investigador.

3. El investigador o administrador selecciona un valor de una faceta en particular.

4. El sistema regresa el compendio de publicaciones correspondiente al valor seleccionado y los elementos relacionados a estos resultados en las navegaciones restantes.

   ∗ Este paso puede repetirse hasta alcanzar el resultado deseado en la navegación presentada.

5. El investigador o administrador da clic en el link correspondiente al nombre del usuario para regresar al directorio del usuario.

6. El sistema regresa al directorio del usuario.

Flujos alternativos

A3.1. El investigador o administrador puede explorar las restantes navegaciones facetadas dando clic a las flechas *citedby* y *references*.

A3.2. Regresa al paso 2.

Postcondiciones: Entrega de resultados al momento de la consulta. Al salir de esta interfaz el sistema vuelve al mismo estado.

Agregar usuario

Descripción breve: Esta transacción le permite al administrador del sistema agregar un usuario.
Actor: Administrador.
Requerimientos que satisface: De la lista 3.1.1 satisface a 1; de la lista 1.2. satisface a 3.1.2 y de la lista 3.1.3 satisface a 1.

Flujo de eventos: Agregar usuario.

1. En el menú del administrador, el administrador selecciona *configuración del sistema*.

2. El sistema despliega la interfaz de configuración del sistema.

3. El administrador da clic en la opción *usuarios*.

4. El sistema despliega la interfaz de administración de los usuarios y la lista de usuarios del sistema.

5. El administrador da clic en la opción *Agregar usuario*.

6. El sistema despliega un formulario con los campos requeridos para agregar un usuario.

7. El administrador llena el formulario y se da clic en *Aceptar*.

8. El sistema valida, guarda al usuario, crea el directorio del usuario y otorga permisos de administración de ese directorio al usuario.

Postcondiciones: El sistema registra al usuario añadido. El sistema crea un directorio propio del usuario. El sistema otorga los permisos al usuario de gestión de los recursos contenidos en su carpeta.

Eliminar usuario

Descripción breve: Esta transacción le permite al administrador del sistema eliminar un usuario.
Actor: Administrador.
Requerimientos que satisface: De la lista 3.1.3 satisface a 1.

Flujo de eventos: Eliminar usuario.

1. En el menú del administrador, el administrador selecciona *configuración del sistema*.
2. El sistema despliega la interfaz de configuración del sistema.
3. El administrador da clic en la opción *usuarios*.
4. El sistema despliega la interfaz de administración de los usuarios y la lista de usuarios del sistema.
5. El administrador selecciona el usuario a eliminar y se da clic en eliminar.
6. El sistema elimina al usuario seleccionado.

Postcondiciones: El sistema elimina al usuario seleccionado.

Accesar al sistema

Descripción breve: Esta transacción le permite al administrador del sistema y a los investigadores del *Instituto de Matemáticas* accesar al sistema.
Actor: administrador, investigador.
Requerimientos que satisface: De la lista 3.1.1 satisface a 1

Flujo de eventos: Accesar al sistema.

1. El administrador o investigador selecciona la opción entrar en la página principal.
2. El sistema despliega la interfaz para acceder al sistema, donde se le pide al usuario que ingrese *nombre de usuario* y *contraseña*.
3. El administrador o investigador ingresa los datos al sistema.

4. El sistema valida y permite la entrada al usuario.

Postcondiciones: El sistema proporciona los recursos y contenidos del usuario validado. Cambia de estado de los recursos de acuerdo a los permisos que posea el usuario.

3.2. Análisis de componentes

En esta etapa, se buscan los principales componentes que satisfagan la especificación de requerimientos.

3.2.1. Búsqueda de componentes

En esta subactividad se proporcionan algunos componentes que puedan satisfacer los requerimientos especificados anteriormente. Cabe señalar que la actividad que es única para el proceso CBSE es la identificación de componentes.

Componente para agregar publicaciones

CMFBibliographyAT es la versión basada en Archetypes del producto CMFBibliograhy que permite el manejo de referencias a publicaciones (científicas) en Plone.

- Proporciona el tipo de contenido Bibliography Folder dedicado a sostener objetos de referencia de diversos tipos, como artículos, libros, borradores, techreports, etc.
- Soporta la importación/exportación de archivos con formato BibTeX por defecto y formatos adicionales dependiendo de la configuración.
- El diseño del esquema general sigue de cerca el enfoque de BibTeX.

Complementos para consultar publicaciones

Como se ha mencionado en la especificación de los requerimientos, las consultas deben satisfacer mucho más que sólo la visualización de la simple lista de publicaciones.

Para ello se necesita la incorporación de filtros o facetas que ayuden a las labor de consulta de publicaciones.

Este requerimiento puede satisfacerse con la implementación al sistema de los siguientes componentes.

- **Exhibit**. Es una aplicación de peso ligero que se ejecuta dentro de un navegador Web, que permite publicar colecciones de datos y navegar por ellos a través de normas comunes de la Web [28].

  - Exhibit hace fácil la agregación de clasificación, filtrado y facetas [3] a la pantalla de datos.
  - Permite la agregación de diferentes vistas: mosaico, tabular, línea de tiempo (timeline) y mapa.

- **EEA Faceted Navigation** (FacetedNav). El proyecto de Navegación Facetada EEA, proporciona una interfaz muy potente para mejorar la búsqueda dentro de grandes colecciones de elementos.
  - Permite gradualmente seleccionar y explorar diferentes facetas (propiedades de metadatos) de los contenidos del sitio.
  - Permite alternativamente, la implementación de una búsqueda avanzada para el sitio.
  - Las facetas pueden establecer valores predeterminados fijos y ocultos.
  - Cuenta automáticamente el número de elementos de contenido al lado de cada valor de faceta
  - Sincronización de la configuración en múltiples idiomas, soporte de i18n.

Componentes estructurales

Faculty/Staff Directory

Proporciona tipos de contenido para la creación y organización de directorios de personal dentro de las instituciones educativas. Se integra con la infraestructura de grupos y usuarios de Plone.

Mas concretamente:

- Proporciona tipos de contenido para crear y organizar los datos de las personas mediante una variedad de campos (correo electrónico, número telefónico, y así sucesivamente).
- Dispone de 3 clasificaciones que se pueden asignar a los objetos *Person*: profesores, personal y estudiantes de posgrado, aunque se pueden agregar clasificaciones propias.

[3] Las *Facetas* se refieren a categorías utilizadas para caracterizar los elementos de información de una colección. Esta categorización es definida mediante etiquetas, de esta manera, un conjunto de etiquetas está asociado con cada faceta [17].

Así, en la interfaz de navegación facetada, cuando una etiqueta es seleccionada por el usuario, todos los elementos que han sido asignados a esa etiqueta se recuperan [17].

Las Facetas son enlaces que son seguidas por los motores de búsqueda. Las facetas conducen generalmente a otras categorías donde la página contiene contenido relacionado con una palabra clave específica, clasificable.

Por otro lado, los *filtros* son enlaces que no son indexables por los motores de búsqueda. Un filtro se utiliza para ordenar o restringir el contenido de una página. Es decir si lo que se desea es ordenar su contenido de manera diferente, por característica específica y estas opciones no cambian el contenido, sólo lo reducen de alguna manera, se trata de un filtro.

- Proporciona varios tipos de contenido para agrupar a la gente: Departamentos, Especialidades y Comités (con posibilidad de renombrarlas fácilmente). Donde en cada asociación (entre una persona y una agrupación) puede darse una descripción.

3.2.2. Selección de componentes

Para comprobar si los componentes propuestos son adecuados o no, se necesita realizar alguna comprobación inicial que no requiera de pruebas detalladas.

3.2.3. Validación de componentes

En la validación de componentes se realiza una prueba de implementación de cada componente, donde se exponen los siguientes resultados:

- En una primera implementación del componente *CMFBibliographyAT* se encuentra que cubre con lo siguiente:

  - Adición de publicaciones en los tipos permitidos definidos por BibTex (article, book, booklet, inproceedings, etc). Referido en el caso de uso *Agregar publicación* en los pasos 3 y 4.
  - Permisos de edición para propietarios de los recursos y administrador. Referido en el caso de uso *Modificar publicación* en el paso 3.
  - Soporte para importación de archivos (BibTex, EndNote, RIS, Medline, etc). Referido en el caso de uso *Importar publicación* en el paso 3.

  Sin embargo queda un aspecto por cubrir:

  - El soporte del identificador ISSN para algunos tipos de publicaciones.

  De manera general podemos decir que este componente abarca los casos de usos *Agregar publicación, Modificar publicación* e*Importar publicación.* Ver figura 3.3.

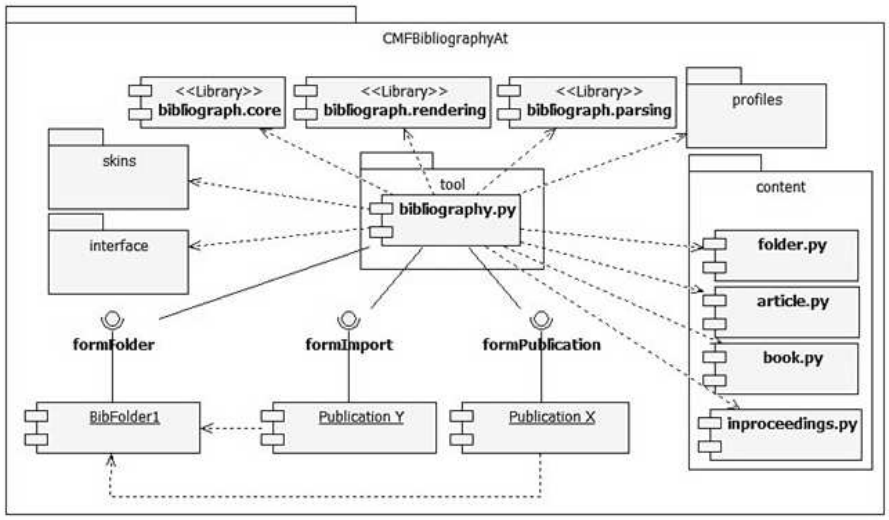

Figura 3.3: Diagrama de componentes del producto *CMFBibliographyAt*.

- Respecto al componente *Exhibit* usado para la visualización de datos se encontró que satisface lo siguiente:

  - Cumple con el uso de facetas para buscar información. Referido en el caso de uso *Consultar publicaciones* en el paso 2.

    Nota: Las publicaciones se definen en una lista en formato CSV (*Comma-Separated values*). Los valores de las facetas toman los valores definidos en este archivo.

  Sin embargo se obtuvieron algunos inconvenientes:

  - Las publicaciones mostradas son definidas en un archivo de valores separados por comas CSV (*Comma-Separated Values*), no son tomados por los existentes en el sistema.
  - No existe la posibilidad en enlazar los resultados de diferentes navegaciones facetadas para hacer consistente la exploración en su totalidad. Cada navegación es independiente.

    Por lo que esto no satisface los requerimientos solicitados por el usuario, dado que se necesita que todos los conjuntos esten en relación a los resultados arrojados por conceptos explorados en algún grupo.

    Es decir no cumple totalmente con el caso de uso *Consultar publicaciones* (en el paso 4).

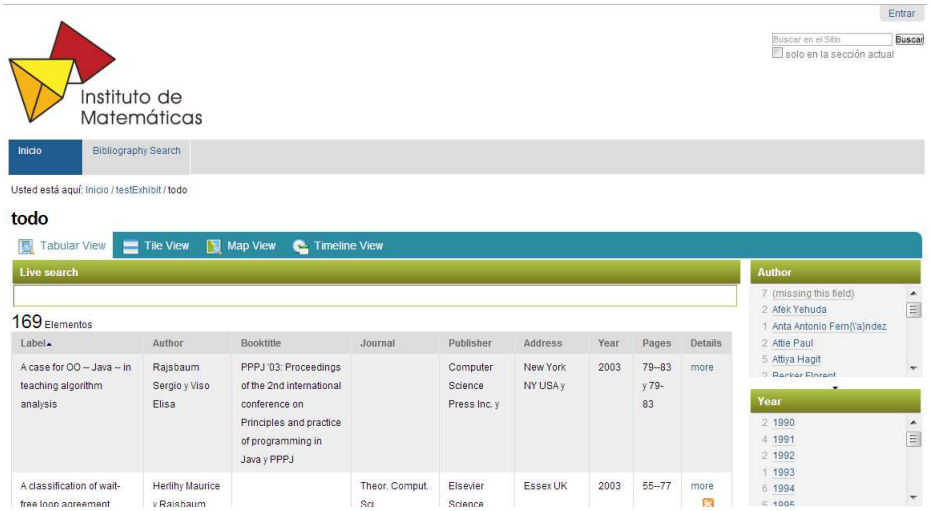

Figura 3.4: Búsqueda facetada con la herramienta Exhibit.

- Otro componente utilizado para la visualización de los datos es *EEA Faceted Navigation* cuyos requerimientos satisfechos fueron los siguientes:

  - La búsqueda mediante facetas. Referido en el caso de uso *Consultar publicaciones* en el paso 2.

  En comparación con el componente *Exhibit* se encuentran las siguientes ventajas:

  - Cada publicación mostrada en la búsqueda con *EEA Faceted Navigation* es tomado del sistema y pueden ser consultada, no es necesario volverla a definir en ningún formato (como en el caso de *Exhibit* mediante archivos CSV).
  - Proporciona diferentes formatos para las facetas (casillas de verificación, nubes de etiquetas, etc).

  Sin embargo:

  - Sólo se permite una búsqueda facetada de un conjunto de contenidos con sus respectivas facetas. De los diversos contenidos existentes tomamos sólo el conjunto de interés, las publicaciones (tipos definidos por el componente *CMFBibliographyAt).*

    En el caso particular de las publicaciones pertenecientes a algún investigador, éstas tienen asociadas elementos del mismo tipo que corresponden a sus respectivas citas y referencias. Entonces si se quisiera ir más allá e indagar sobre las citas y referencias de un conjunto de publicaciones dado, se ha verificado que este producto no ofrece

este soporte. No se puede enlazar los resultados de una navegación facetada con otra, cada una es independiente. Caso similar al de la herramienta *Exhibit* como ya se ha mencionado.

Es decir no cumple totalmente con el caso de uso *Consultar publicaciones* (en el paso 4).

Sin embargo este último aspecto resulta ser un tanto relevante en la especificación de requerimientos, ya que el usuario investigador busca ir más allá de lo presentado actualmente en las diferentes bibliotecas digitales e indagar sobre los elementos en relación a sus publicaciones.

Por lo cual esbozamos un diagrama de componentes de cómo debería ser nuestro componente de consultas de manera tal que se cumpla la mayoría de los requerimientos en el caso de uso *Consultar publicaciones*. Ver figura 3.5.

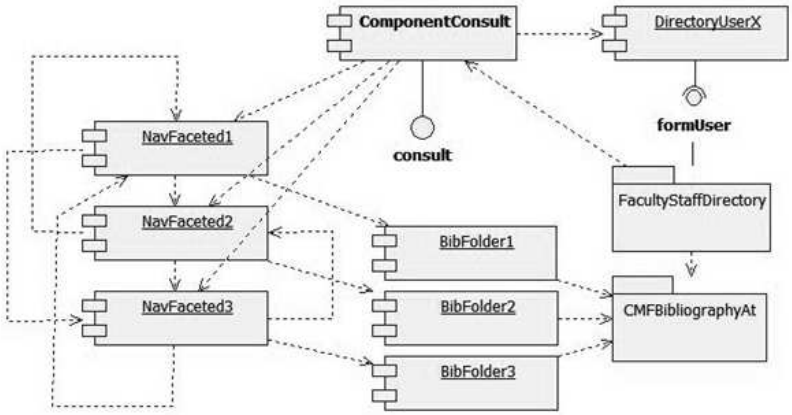

Figura 3.5: Diagrama de componentes del componente ideal nombrado temporalmente *ComponentConsult*.

- Por último el componente estructural *Faculty/Staff Directory* cubre los requerimientos siguientes:

  - Permite crear una estructura de directorios donde cada investigador tiene su propio directorio de almacén de publicaciones. Referido en el caso de uso *Agregar usuario* en el paso 8.
  - Gestión de recursos propios. Cada usuario investigador puede editar, agregar, modificar sus propias publicaciones. Referido en los casos de uso donde sólo el usuario propietario (o administrador) pueden hacer cambios o adiciones de contenido, a saber los casos de uso *Agregar publicación*, *Modificar publicación*, *Eliminar publicación* donde es una pre-condición poseer los permisos de administración de los recursos

propios y en el caso de uso donde se adquieren estos permisos *Agregar usuario* en el paso 8.

- Un investigador puede consultar pero no editar los recursos de otro investigador. Un requerimiento derivado del punto anterior, más estrechamente relacionado con el caso de uso *Modificar publicación*. También relacionado con el caso de uso *Consultar publicaciones* donde se permite que cualquier usuario únicamente consulte la información de los investigadores del *Instituto de Matemáticas*.

Y contiene además éstas características adicionales:

- Permite la clasificación de miembros del *Instituto de Matemáticas* dentro de algún grupo.
- Permite adicionar información de cursos que ofrecen los miembros del IM.
- Permite agregar información del investigador y descargarla en forma de vCard[4]
- Permite al investigador cambiar de estado (p.e publico o privado) los a recursos asociados a él (dentro de su carpeta en la estructura del Directorio).
- El usuario administrador puede configurar los tipos permitidos de contenido a cada investigador en el Directorio del *Instituto de Matemáticas*. Esto, mediante la *Interfaz de Administración de Zope* o ZMI (*Zope Management Interface*).

Sin embargo :

- Dentro de cada sección de los investigadores sólo se puede agregar publicaciones y cursos por default, no se pueden agregar algún otro tipo de contenido. Por lo que se necesita configurar y habilitar más opciones en este complemento.

Con base en los resultados obtenidos de la implementación de los componentes, se validan los siguientes componentes:

- *CMFBibliographyAT*
- *Faculty/Staff Directory*

Y se rechazan los componentes:

- *Exhibit*
- *EEA FacetedNavigation*.

[4]vCard. A veces llamada también tarjeta electrónica. Es una especificación que define el formato de una "tarjeta de presentación electrónica". Por lo general contiene el nombre de la empresa, dirección, número de teléfono, URL, logotipo, etc.

3.2.4. Modificación de los requerimientos

Esta sección identifica los requerimientos faltantes del sistema con base en los requerimientos satisfechos por los componentes validados y por el sistema Plone.

Los requerimientos satisfechos por el CMS Plone son aquellos relacionados con la administración del sitio, en particular con la administración de los usuarios, como agregar, eliminar o modificar permisos de usuarios.

Esto puede configurarse en el apartado *Configuración del sitio* (ubicado sobre el link *admin* que aparece en todas las interfaces) → *usuarios y grupos*.

Para el requerimiento relacionado con el auto-registro de los usuarios, este es satisfecho por el CMS Plone en el apartado *Configuración del sitio*→ *seguridad*.

En general, el administrador del sistema posee los privilegios para modificar cualquier contenido que el sistema tenga, su estado y los accesos a ellos. Por lo que en particular, tiene los permisos suficientes para agregar, modificar o eliminar las publicaciones de los usuarios investigadores.

Por otro lado en lo que respecta a los requerimientos no cubiertos por los componentes, éstos se listan a continuación:

- El sistema debe permitir al investigador añadir o editar o eliminar las citas y referencias relacionadas a cada artículo de su conjunto de publicaciones.
- La información de citas y referencias debe estar relacionada a la publicación a la cual se asocia y debe ser visualizada.
- El sistema debe permitir la exploración mediante facetas de un conjunto de publicaciones determinado, sus citas y referencias. Cada conjunto con su propia navegación facetada.
- El sistema debe proporcionar consistencia en la exploración de las publicaciones, no sólo del conjunto explorado, si no también los elementos de otros conjuntos en relación a los elementos resultantes. De aquí que es importante vincular a las publicaciones con otros elementos en relación.
- Por otro lado, el sistema debe soportar al identificador ISSN.
- Finalmente el sistema debe proveer alguna métrica estándar que haga referencia al impacto y productividad de los trabajos científicos de los investigadores, en este caso se ha elegido el índice h y el índice *i10*. Estos datos deben ser proporcionados por la interfaz de consulta de publicaciones de cada investigador.

De esta manera reducimos los casos de uso a desarrollar en nuestro sistema. Ver figura 3.6.

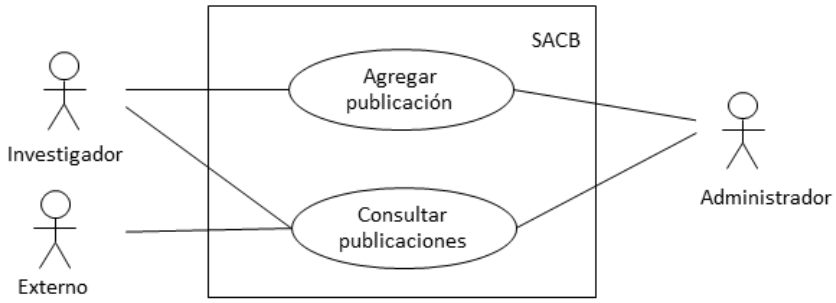

Figura 3.6: Caso de uso *Consultar publicaciones* por crear y caso de uso *Agregar publicaciones* por configurar/afinar.

3.3. Conclusiones

En el proceso de la CBSE algunas etapas dentro del proceso se llevan a cabo de misma forma que en otros procesos de software. Dichas etapas refieren a las etapas iniciales del proceso, como son: la etapa de especificación y la validación de requerimientos.

En lo que respeta a las actividades realizadas, se ha llegado a la validación de algunos componentes y a la identificación de requerimientos no satisfechos. Se pretende en etapas posteriores satisfacer en gran parte de la especificación de requerimientos.

Capítulo 4

Diseño y configuración de componentes

Este capítulo esta dedicado al diseño del sistema con reutilización de componentes y generación de otros nuevos. Todo con el objetivo de satisfacer la especificación de requerimientos.

En primer lugar se muestran las interfaces de los componentes disponibles, su manejo y descripción. Tambien se muestra la estructura general del sistema.

En segundo lugar se proporciona el diseño de un nuevo componente destinado a las consultas de publicaciones y elementos relacionados. Este componente es el llamado *ComponentConsult* propuesto en etapas anteriores (como el componente ideal que satisface las necesidades de los usuarios), en esta etapa lo renombramos como *matem.facetedbibliography*. De este componente mostramos su descripción, una propuesta de interfaz gráfica y los diagramas de clase correspondientes.

Posteriormente se incluyen las conclusiones del capítulo.

4.1. Interfaces de usuario.

En esta sección se muestran las interfaces de usuario de los componentes disponibles y del sitio en construcción.

Componente *Faculty Staff Directory*

Este componente provee parte de la estructura de nuestro sistema.

En nuestro sitio se ha de encontrar un portlet de navegación en la parte izquierda del sitio donde se encuentra el directorio *Instituto de Matemáticas* que contiene una serie de directorios destinados a los investigadores y recursos de la entidad.[1]

[1] El portlet para la navegación debe ser habilitado y configurado al igual que la creación y configuración del directorio *Instituto de Matemáticas* del tipo *Faculty/Staff Directory*; todo

Los investigadores en particular tienen un icono que hace distinción entre una carpeta y un usuario del sitio (miembro de la organización) en general. Ver figura 4.1.

Figura 4.1: Estructura del directorio del *Instituto de Matemáticas.*

Dentro del espacio reservado y/o destinado a un determinado investigador se encuentran las opciones de los recursos disponibles que el investigador puede añadir (siempre y cuando se encuentre en su propio directorio).

En otras palabras, dentro de la sección destinada a un usuario del sistema, se otorgan los permisos de administración al usuario de determinados tipos de recursos.

Esto es el poder cambiar o agregar contenido a conveniencia del usuario. Sin embargo estos permisos se limitan al directorio del investigador y no al de otro usuario (en esta situación se adquiere los permisos de *miembro* de la organización).

Uno de estos recursos es el directorio destinado a contener nuestros elementos bibliográficos conocido como *Bibliography Folder* proporcionado por el componente *CMFBibliographyAT*. De este componente se habla en la siguiente subsección.

esto mediante las opciones presentadas en la página default del sitio.

Componente *CMFBibliographyAT*: agregar publicaciones

Una vez que el usuario desee administrar la información de sus publicaciones se tiene que tener directorios tipo *Bibliography Folder* para contener dicha información.

Este folder permite sólo contenido especializado, que se refiere a los diferentes tipos de publicaciones existentes, como son *article*, *book*, *inproceedings*, etc.

Entonces una vez que el usuario desee agregar este tipo de contenido, el proceso es el siguiente:

1. En primer lugar nos ubicamos en la carpeta que se nos ha asignado dentro del directorio *Instituto de Matemáticas*. En este caso tomamos como ejemplo que tenemos asignado el usuario *Sergio Rajsbaum*

2. Creamos el directorio correspondiente a nuestras publicaciones. Ver figura 4.2.

Figura 4.2: Agregar un folder tipo *Bibliography Folder* como contenedor de las publicaciones.

3. Y lo guardamos con el nombre *publicaciones*.

4. Dentro de esta carpeta se presentan dos opciones para agregar publicaciones:

   - Agregar el tipo de publicación manualmente.
     - Se da click en la pestaña *Agregar nuevo*.
     - Se selecciona el tipo de publicación a añadir.
     - Se llenan los campos obligatorios marcados con un punto rojo (ver figura 4.3).
     - Alternativamente se pueden llenar los campos no obligatorios.
     - Una vez finalizado el llenado del formulario se procede a a guardarlo (click en el botón guardar).
   - Importar un archivo en formato BibTex, EndNote, RIS, Medline y XML.

- Al final de la visualización del folder creado (destinado a las publicaciones) se encuentra el enlace para importar archivos *import*.
- Este enlace conduce a una página donde se elige el formato deseado, el tipo de codificación, el alcance de las búsquedas de duplicados (local o global) y el archivo o texto en un formato especificado. Ver figura 4.4.
- Finalmente se guarda dando click sobre el botón *import*.

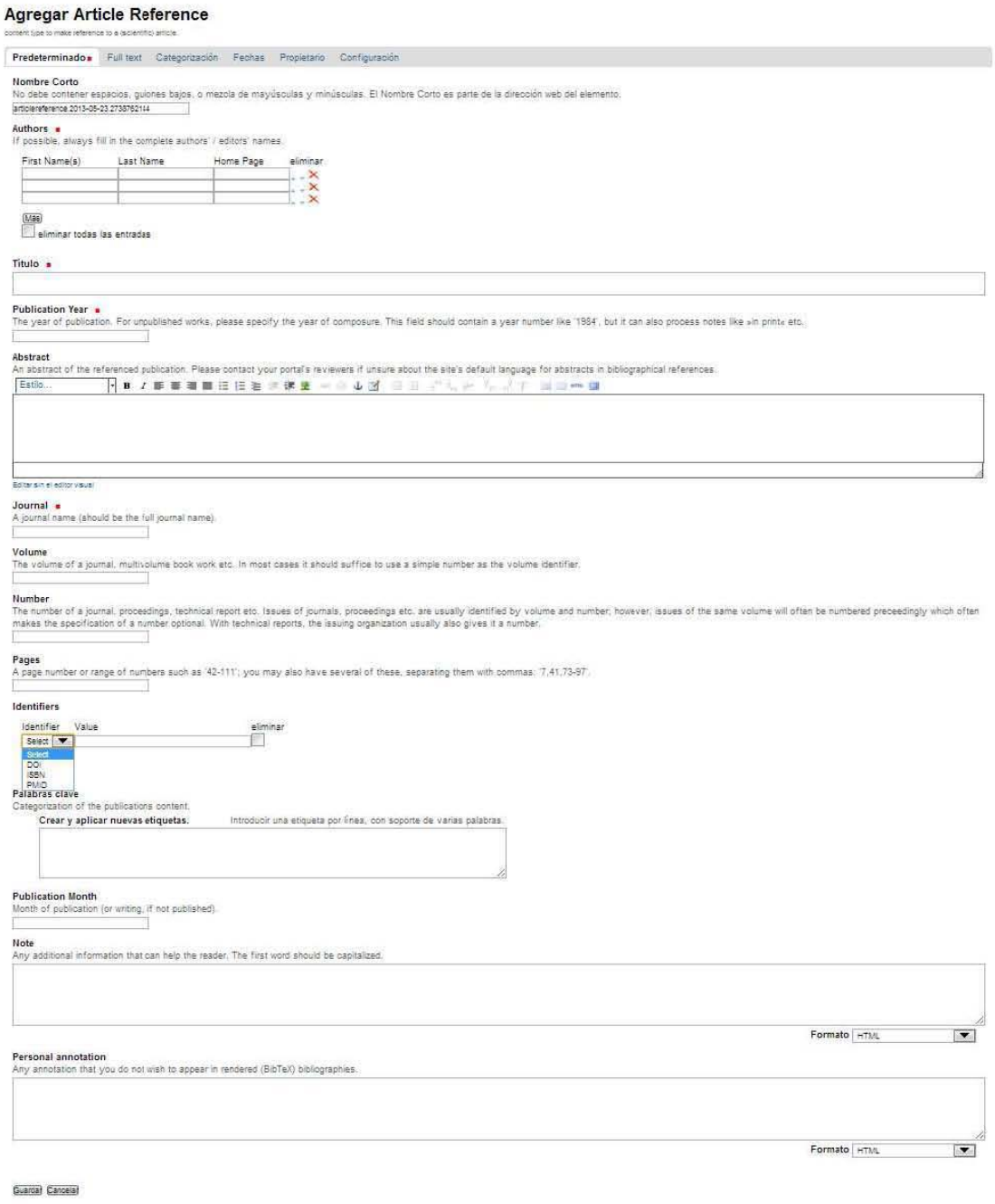

Figura 4.3: Formulario para una publicación del tipo *Article Reference*.

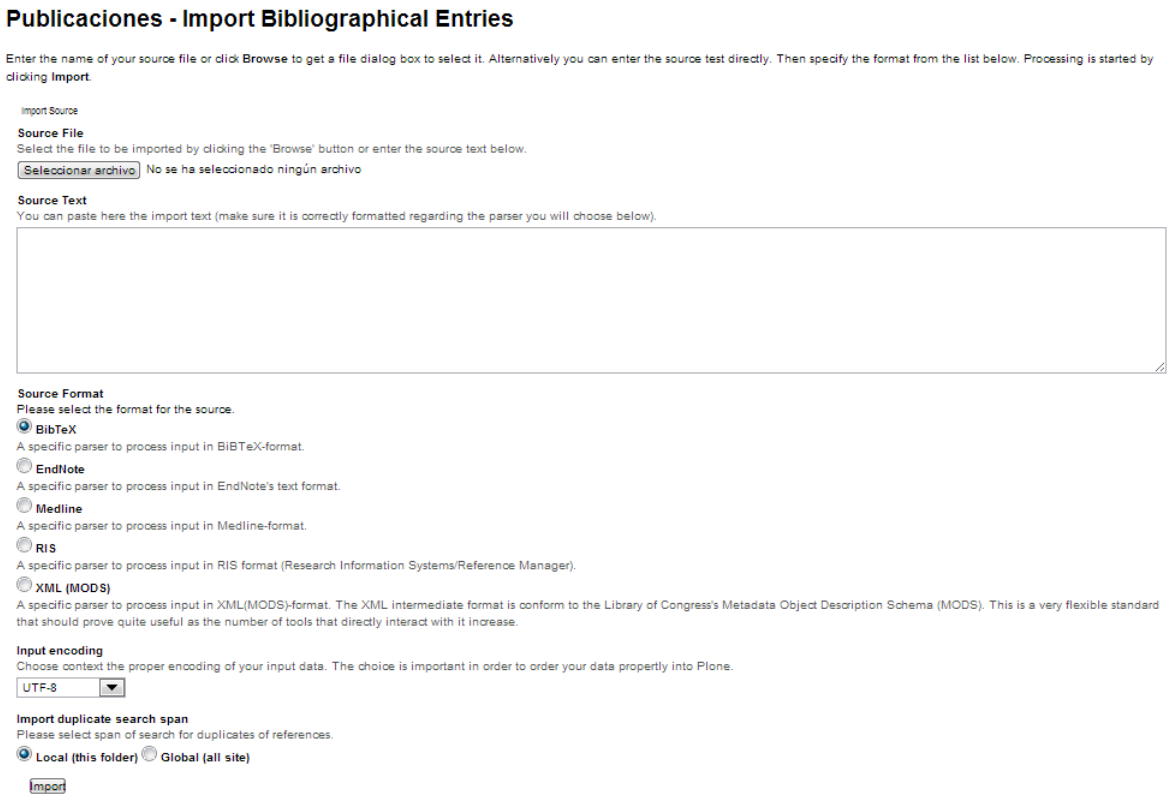

Figura 4.4: Importanción de una publicación en un formato establecido.

4.2. Modificación de componentes

En el proceso de ajuste de componentes se deben analizar las partes principales que conforman el producto.

Para ello se puede optar por analizar directamente el código. Analizar cada uno de los paquetes que integran al producto y modificar las partes necesarias que permitan satisfacer las necesidades de los usuarios.

Sin embargo Plone ofrece un tipo de desarrollo que permite al desarrollador del sitio personalizar partes específicas de sus componentes: el llamado desarrollo *a través de la Web*, que refiere simplemente al trabajo/desarrollo a través de un navegador.

El desarrollo de nuevas características para un sitio Plone es posible con la ayuda de Zope, añadiendo piezas de código directamente desde el navegador a través de la *Interface de Administración de Zope* o ZMI (*Zope Management*

Zope) [33]. Es una forma de desarrollo controlada que evita la exposición de código importante.

La interfaz es disponible mediante las opciones siguientes:

- Agregar */manage* al final de la URL del sitio en cuestión (por ejemplo: `http://localhost:8080/manage`)
- Al hacer clic en el enlace *admin* → *Configuración del sitio* → *Interfaz de Administración de Zope* en la interfaz del sitio Plone.

La ZMI es la interfaz básica que le permite acceder a la interfaz de Zope subyacente de Plone [33]. Ver figura 4.5.

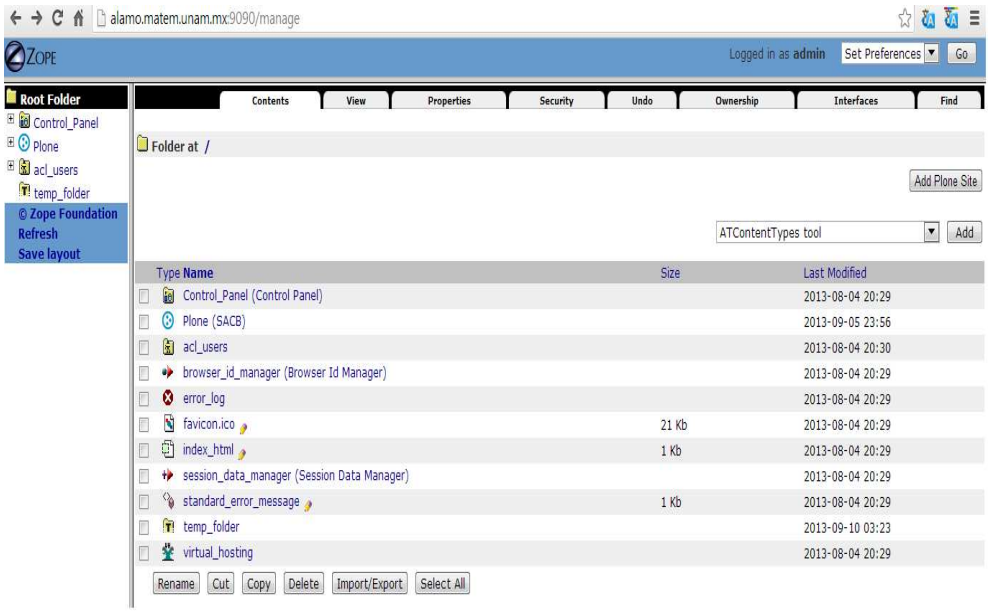

Figura 4.5: ZMI o *Interfaz de Administración de Zope.*

Una vez que el administrador se encuentra en el ZMI, se puede seleccionar un tipo determinado de contenido y explorar las pestañas disponibles que conducirán a diferentes formas de configuraciones del contenido.

Cada tipo de contenido puede tener diferentes pestañas dependiendo de cómo el contenido puede ser configurado. Las pestañas en común para la mayoría de los objetos son *View*, *Edit* y *Security*.

Sin embargo no todos los componentes pueden extenderse adecuadamente mediante la ZMI, por lo que se puede recurrir a la inspección y extension de código fuente directamente.

4.2.1. Configuración de componentes

Ahora bien, en el caso de la extensión/configuración de componentes, la ZMI puede ayudar en esta labor.

Para el caso del producto *CMFBibliographyAT* se busca incorporar el soporte para el identificador ISSN.

Por lo que se identifican las secciones que intervienen en el ajuste del producto a tratar.

Estas secciones son las siguientes:

- */Plone/portal_bibliography* cuyas principales pestañas de configuración son las siguientes: *Criteria* y *Properties*.[2] La pestaña *Criteria* establece el conjunto mínimo de entradas que deben cubrirse para cada tipo de publicación y la pestaña *Properties* administra algunas cuestiones relacionadas con la publicaciones, tales como mostrar o no el link isbn, ciertos permisos o diferentes configuraciones de identificación.

- */Plone/portal_properties/cmfbibat_properties* donde la pestaña principal es *Properties* donde se habilitan los formatos de archivos que sean requeridos (BibTex, EndNote, Medline, etc).

- */Plone/portal_properties/extension_properties* donde también la principal pestaña de configuración es *Properties*, donde se configuran las propiedades del producto *ATExtension*. Este producto actúa como dependencia del producto *CMFBibliographyAT*, por lo que su configuración de *ATExtension*[3] repercute igualmente en él.

- */Plone/portal_types* cuyas pestañas principales de configuración son *Content* y *Aliases*. Cuando se elige algún tipo de contenido en *Content* se tienen nuevamente opciones de configuración para cada elemento. En el caso del producto *CMFBibliographyAT* los tipos de interés son aquellos que hagan referencia al los tipos de publicaciones asociados a BibTex (tipos *article, inproceedings, book*, etc.). De estos tipos podemos modificar el tipo de vista, el tipo de contenido permitido, etc. En el caso de la opción *Aliases* éste hace referencia al conjunto de alias disponibles de las propiedades de cada contenido.

[2] La ruta puede variar dependiendo el nombre del sitio que hayamos puesto en la creación del sitio, en este caso se dejó el nombre default Plone.

[3] *ATExtension* Este paquete proporciona algunas extensiones a arquetipos. Hasta el momento, existen principalmente algunos campos personalizados, widgets y validadores, siendo el campo de registro/widget uno de los componentes más genéricos [30].

Archetypes (arquetipos) es un framework diseñado para facilitar la creación de aplicaciones para Plone y CMF. Su objetivo principal es proporcionar una forma estandarizada para construir objetos de contenido basados en definiciones de esquema. *Archetypes* se instala por defecto en todas las versiones modernas de Plone [30].

Widget Una pieza de software que se utiliza en una página de un sitio web para proporcionar al usuario información cambiante/dinámica de un determinado tipo en un área pequeña de la pantalla del ordenador [4].

En el caso del componente *CMFBibliographyAT* existen los opciones disponibles de configuración para modificar/agregar nuevos identificadores a las publicaciones. Esta opción se encuentra en */Plone/portal_properties/extension_properties*. Por lo que este requerimiento queda cubierto y soportado. Ver figura 4.6.

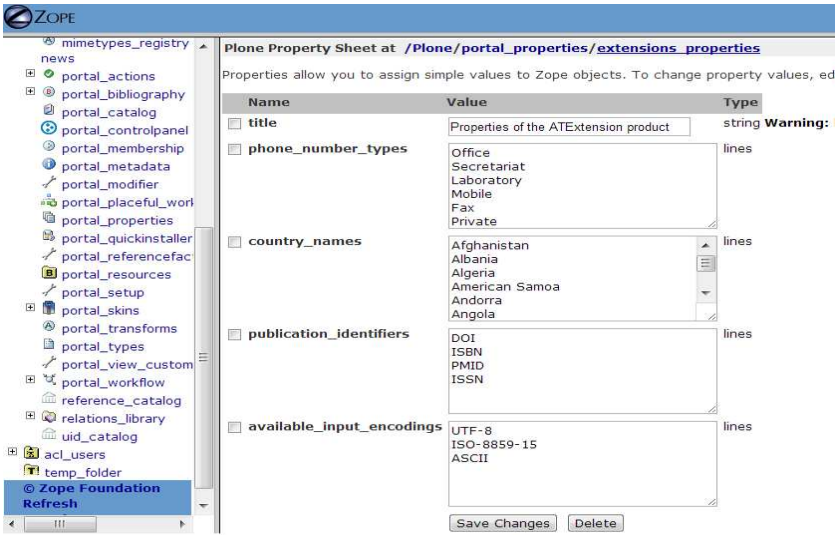

Figura 4.6: Propiedades del producto *ATExtension*: Adición del identificador ISSN para publicaciones.

Como nota adicional señalamos ciertos problemas de integración de este producto en la sección Apéndice A.2.

Por otro lado en cuanto a la configuración del producto *Faculty Staff Directory* pueden habilitarse los tipos de contenido que el usuario tiene permitido agregar en su sección dentro del directorio *Instituto de Matemáticas*. Ver figura 4.7.

En el caso de la navegación por facetas se opta por el desarrollo de un nuevo producto ya que los requerimientos no cubiertos son más revelantes y pueden reflejar mayor información al investigador acerca de sus publicaciones.

De esta manera, en la siguiente sección, el foco de atención se centrará en el diseño de una navegación facetada que contemple la adopción de citas y referencias en su proceso de exploración. Cabe señalar que la teoría detrás de éste diseño se encuentra en la sección 2.4 del presente documento.

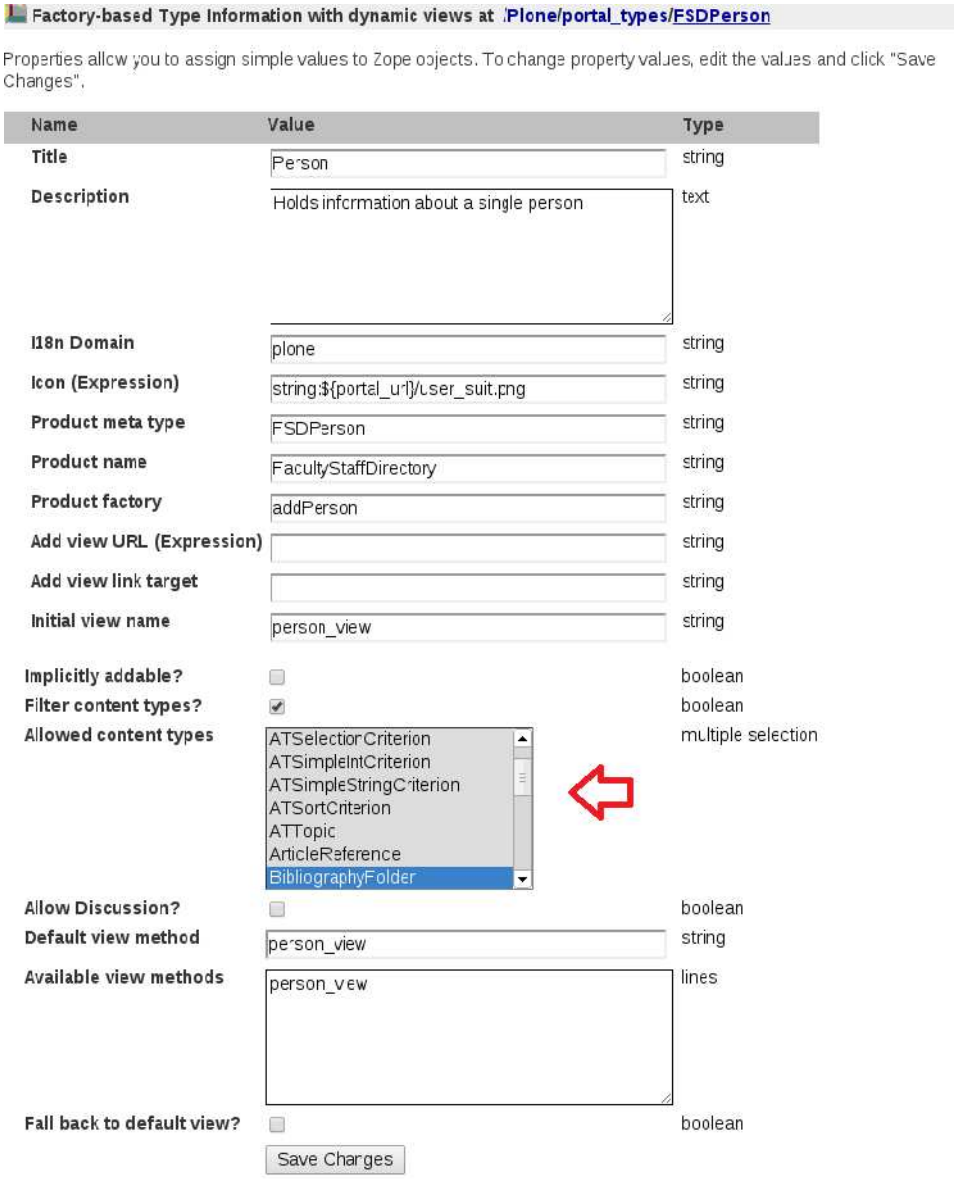

Figura 4.7: Configuración de los tipos de contenido para un usuario dentro de una estructura tipo *Faculty Staff Directory*.

4.3. Diseño del componente para consultas

En el proceso de realizar búsquedas de publicaciones se han efectuado hasta el momento acciones simples de exploración de publicaciones a través de los atributos más generales de las publicaciones. Esto con los productos presentados anteriormente.

Sin embargo la exploración de elementos adjuntos al contenido de publicaciones no pueden explorarse adecuadamente con estas herramientas, por lo cual, se ha decidido diseñar y producir un componente propio que provea un alcance mayor al explorar las citas y referencias, además de proporcionar consistencia en la exploración de estos elementos.

Como se señala en el marco teórico se hace uso de las llamadas taxonomías facetadas[4] para la generación de navegaciones través de facetas.

Estas taxonomías se basan en una jerarquía de *conceptos* que se pueden utilizar para seleccionar las zonas de interés y restringir la porción de elementos a ser recuperados.

Tales taxonomías se fundamentan estructuralmente en un grafo acíclico dirigido formado por conceptos y objetos.

En este caso, para una versión preliminar, diremos que nuestros conceptos de partida son *author, collaborator, type, year* y *journal*; entre los que diferenciamos a *author*, que se refiere al o los investigadores pertenecientes al IM y *collaborator* que se refiere a los autores externos, que no son parte del instituto. Luego en el siguiente nivel se definen las particularidades de cada concepto, donde estos últimos conceptos pueden relacionarse mejor con cada publicación y hacer una búsqueda más efectiva. Ver figura 4.8.

[4]Para mayor información consulte la sección 2.4

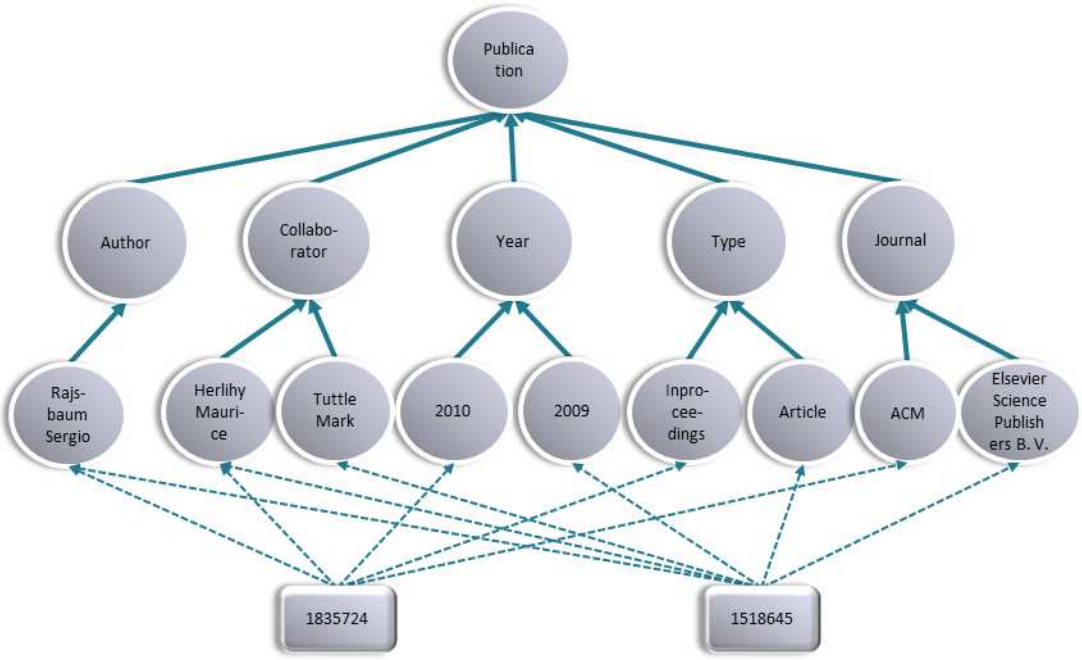

Figura 4.8: Taxonomía básica de publicaciones.

Hasta aquí el diseño de la taxonomía parece ser el más genérico ocupado por cualquier librería digital. Sin embargo se necesita de la visualización de las citas y referencias de los elementos presentados, de manera tal que también posean su propia navegación facetada.

Por lo que se propone emplear una navegación facetada que utilice esta es misma estructura genérica para las citas y referencias de las publicaciones presentadas.

El dominio de cada taxonomía adicional (citas y referencias) dependerá de las publicaciones presentes en la navegación facetada principal. Es decir, si los resultados arrojados en la exploración central cambian, incrementan o decrementan afectara el dominio de las otras navegaciones facetadas puesto que son citas y referencias de las publicaciones centrales activas. De manera similar el conjunto de publicaciones principales se verán afectadas por la exploración de las taxonomías secundarias de citas y referencias, de tal modo que los resultados arrojados sean consistentes con los atributos seleccionados en las taxonomías correspondientes.

De este modo se tendrá un mayor alcance en la exploración de publicaciones científicas y sus elementos relacionados.

4.3.1. Modo de operación: una taxonomía

Para construir una taxonomía similiar a la presentada en la sección anterior se debe proporcionar una fuente de información que permita construir el grafo que la represente. En este caso nos valemos de entradas con formato BibTex. El formato BibTex proporciona las etiquetas de cada campo con las cuales se puede construir una taxonomía.

Una vez construido el grafo, las opciones presentadas para realizar las consultas serán los conceptos clasificados bajo *author, type,year, journal* y *collaborator*. Es decir para el concepto *year* serán los años obtenidos tras el procesamiento de la entrada en formato BibTex, por ejemplo *2010, 2009*, etcétera y no el concepto *year* o *publication*.

Los objetos serán los registros de cada entrada BibTex encapsulados en un tipo de dato cuya etiqueta en la taxonomía corresponderá al identifcador que se tenga en la entrada BibTex.

Luego en la consulta de un concepto se aplica la operación *deepExtension* que obtiene los objetos asociados al concepto seleccionado.

Para realizar más de una consulta, se aplica el comportamiento observado en la interfaz de búsqueda facetada de DBLP[5], en los casos siguientes:

Cuando se desea consultar conceptos que tienen el mismo padre se toma la unión de los objetos obtenidos por cada concepto. Ver figura 4.9.

[5]La interfaz de búsqueda FacetedDBLP permite buscar publicaciones de ciencias de la computación en la colección DBLP (*Digital Bibliography & Library Project*) partir de algunas palabras clave y muestra el conjunto de resultados junto con un conjunto de facetas. Para conocer esta interfaz consulte http://dblp.l3s.de/.

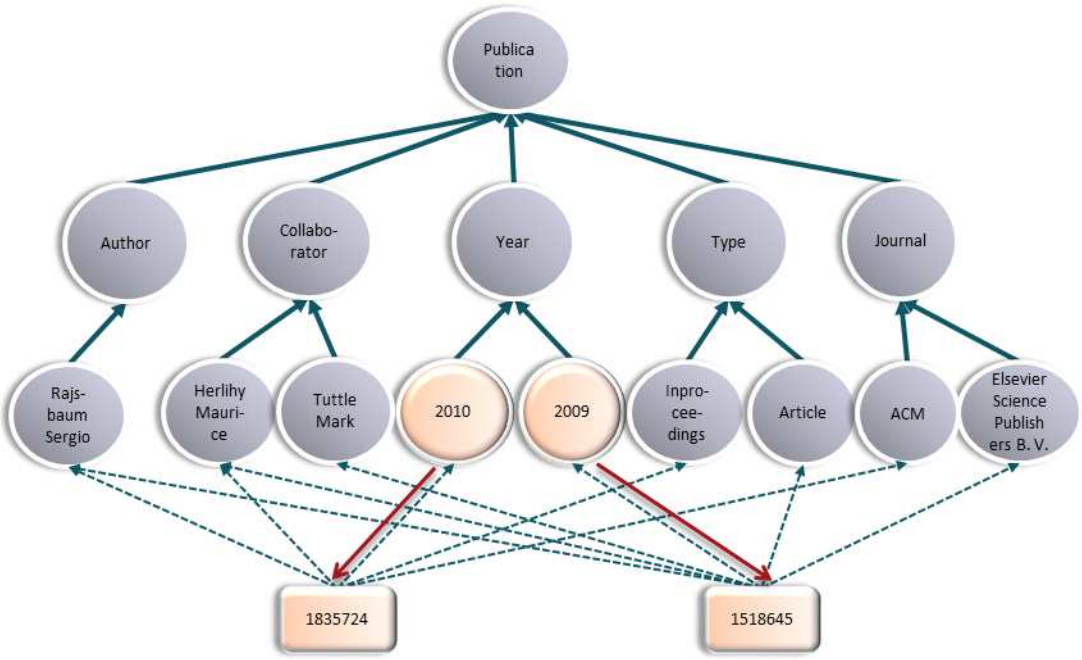

Figura 4.9: Consulta de conceptos (*2009, 2010*) con el mismo padre (*year*): unión de objetos resultantes.

Cuando se desea consultar conceptos que tienen diferente padre se toma la intersección de los objetos obtenidos por cada concepto con diferente padre. Ver figura 4.10.

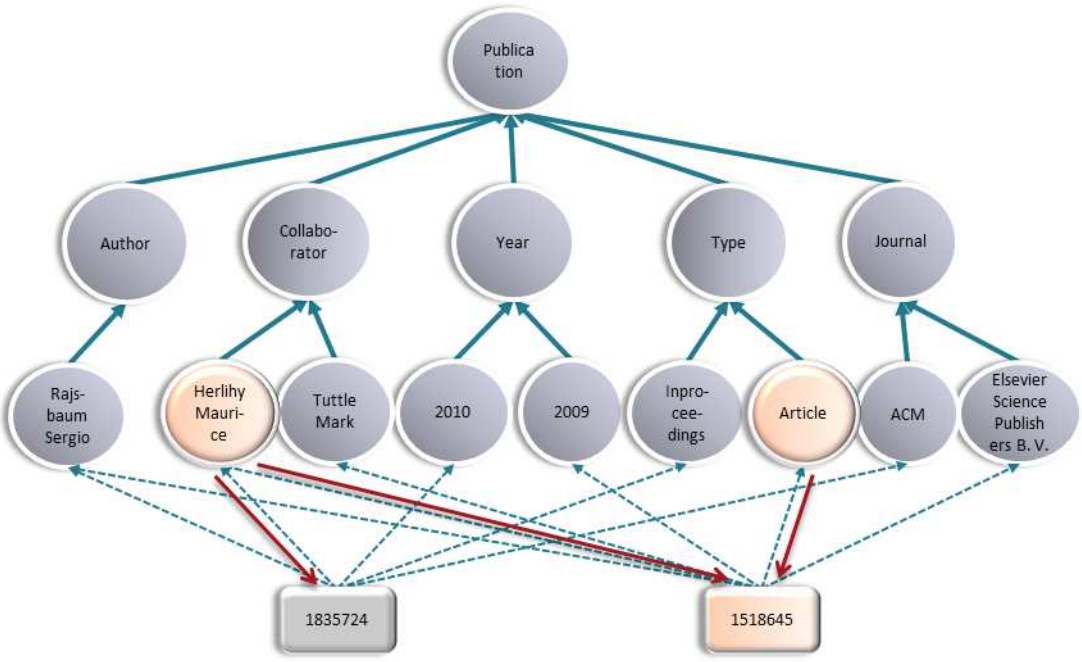

Figura 4.10: Consulta de conceptos (*Herlihy Maurice, article*) con diferente padre (*collaborator, type*): intersección de objetos resultantes.

En ambos casos se refiere al padre inmediato.

De esta manera cuando se tenga una consulta del tipo *2010, 2009* y *article* arrojará las publicaciones de los años 2010 y 2009 que sean estrictamente del tipo *article*.

4.3.2. Modo de operación global

En la construcción de una navegación facetada que contemple las citas y referencias de los elementos a explorar debemos tener en cuenta lo siguiente:

En cuanto a funcionamiento debemos tener en claro que si queremos explorar los elementos asociados a las publicaciones debemos crear taxonomías facetadas para estos conjuntos de elementos. Una taxonomía dedicada a las citas y otra dedicada a las referencias.

Evidentemente también debe proporcionarse la relación que guardan entre sí los elementos.

Es decir un elemento cualquiera debe contar con la información que le indique con cual(es) elemento(s) se relaciona y qué tipo de relación guarda con él (o ellos), es decir, si el elemento con el que se relaciona es un elemento citante (al artículo que representa) o una referencia.

Esta información se ha de adjuntar en el archivo BibTex en la etiqueta *abstract* del artículo. Se debe de indicar por separado los grupos de elementos asociados con una sintaxis específica que señale a los identificadores de los elementos de estos grupos. Es decir:

`abstract={citedby=(id_0, id_1, ... ,id_i) reference=(id_0, id_1, ... , id_j)},`

Entonces al momento de construcción, los objetos (que representan a las publicaciones) tendrán los identicadores de sus respectivas citas y referencias.

Esto con el fin de que al momento realizar alguna consulta en una determinada taxonomía se muestren únicamente los resultados y los elementos en relación a éstos en las taxonomías adyacentes.

Todo ello resulta en consistencia en la exploración global.

Consistencia de taxonomías.

La asociación de elementos de las diferentes taxonomías debe generar conjuntos de publicaciones relacionados entre sí tras una exploración.

Cada publicación debe tener en cuenta la información de los elementos con los que se relaciona independientemente de la taxonomía a la que pertenezca.

En la exploración de conceptos se pueden tener múltiples casos donde se pueden involucrar todas las taxonomías disponibles.

Para los diferentes casos mostramos un ejemplo de cómo obtener nuevos conjuntos de publicaciones relacionados entre sí tras la exploración de conceptos.

Un caso base sería la correspondiente exploración a una taxonomía y la actualización de las restantes a partir de los resultados obtenidos.

En el caso presentado a continuación se explora la taxonomía central. Los elementos resultantes están en asociación con los elementos de las taxonomías adyacentes, en este caso con las taxonomías 1 y 3. Ver figura 4.11.

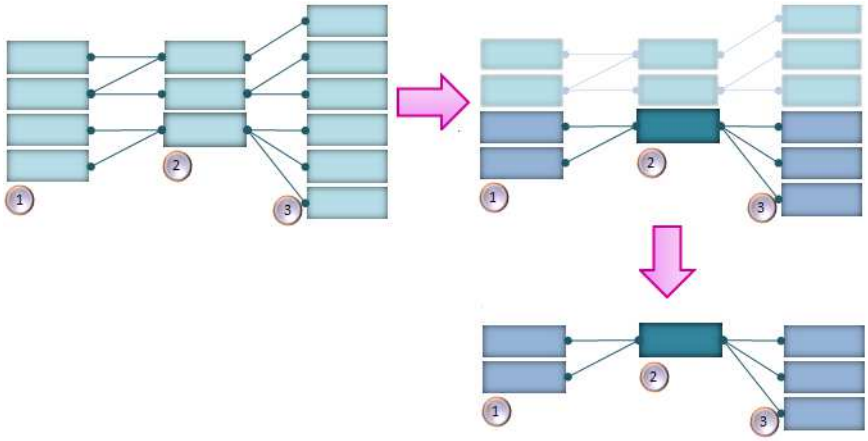

Figura 4.11: Estado inicial de las taxonomías. → Elemento resultante (azul obscuro) en la taxonomía central con elementos asociados resaltados. → Taxonomías finales.

Las asociaciones con elementos en otras taxonomías influyen en la actualización de las taxonomías adyacentes, quedando sólo elementos relacionados con los resultados obtenidos de la taxonomía central.

Este comportamiento es análogo en cualquier otra taxonomía (no necesariamente la central) alcanzando una influencia directa o indirecta en las demás taxonomías. Ejemplo en la figura 4.12

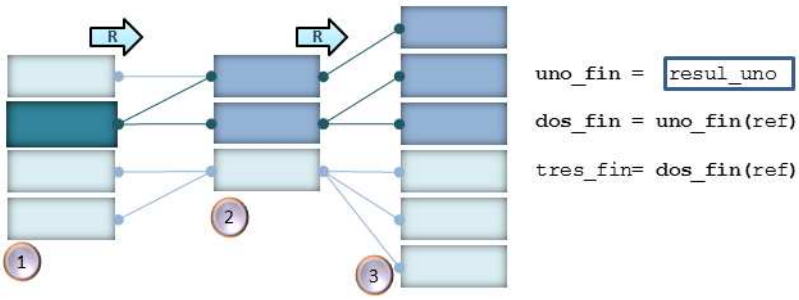

Figura 4.12: Los conjuntos de publicaciones finales se denotan como `uno_fin`, `dos_fin` y `tres_fin`. El cuadro marcado en azul oscuro denota el conjunto publicaciones resultantes (o `result_uno`) tras la exploración de conceptos; las demás taxonomías quedan influenciadas a partir de este conjunto. En este caso la influencia va de derecha a izquierda. La abreviatura *ref* señala al conjunto de referencias de un grupo de publicaciones dado (al igual que la etiqueta R en la ilustración).

En este otro caso se puede observar que la taxonomía consultada interviene en la actualización de publicaciones de la taxonomía central a través de las referencias obtenidas del conjunto resultante de publicaciones de la taxonomía 1, las referencias obtendias aluden a elementos en la taxonomía 2. De manera similar la influencia de la taxonomía 2 a la 3 (a través de las referencias obtenidas).

En el caso de consultas en dos taxonomías el comportamiento es similar, la diferencia radica en que ahora se toman en cuenta dos conjuntos de resultados que determinan a las restantes.

Consideramos que cuando una taxonomía queda influenciada por ambos conjuntos resultantes se toman sólo los elementos asociados en común (intersección de conjuntos). Ver figura 4.13.

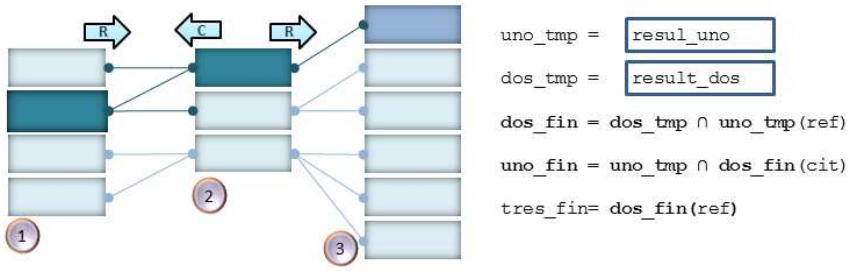

Figura 4.13: Los conjuntos resultantes tras la exploración de conceptos son `result_uno` y `result_dos` correspondientes a las taxonomías 1 y 2. Ambas taxonomías tienen influencia el una a la otra. La tercera taxonomía sólo se ve influenciada por los conjuntos finales obtenidos. La abreviatura *cit* señala al conjunto de citas de un grupo de publicaciones dado (al igual que la etiqueta C en la ilustración). Análogamente R y *ref* para referencias.

En el último caso de consulta a las tres taxonomías, se toman en cuenta los tres conjuntos resultantes y los alcances de cada una. En este caso, como en el caso anterior, consideramos únicamente los elementos asociados a los conjuntos resultantes en común (intersección de conjuntos). Ver figura 4.14.

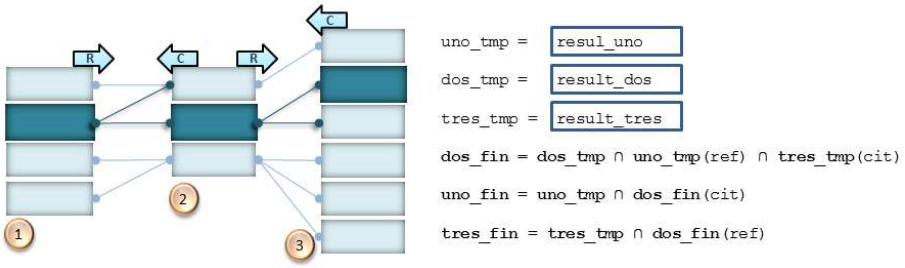

Figura 4.14: En este caso se exploran 3 taxonomías. Los conjuntos resultantes tras la exploración de conceptos son `result_uno, result_dos` y `result_tres` correspondientes a las taxonomías 1, 2 y 3 (resaltados en color azul oscuro). Cada taxonomía se ve influenciada por los resultados de sus correspondientes taxonomías adyacentes. De esta manera los conjuntos de publicaciones obtenidos son resultado de la intersección de diversos conjuntos que están en relación con los resultados obtenidos.

En todos los casos debemos tener en cuenta que en ocasiones pueden darse resultados nulos, debido a que no existen elementos en común relacionados a los resultados obtenidos

Sin embargo podemos concluir que en general el resultado es en sí un conjunto de taxonomías estrechamente relacionadas.

4.3.3. Diagramas de clases e interfaz gráfica

(Componente de consultas)

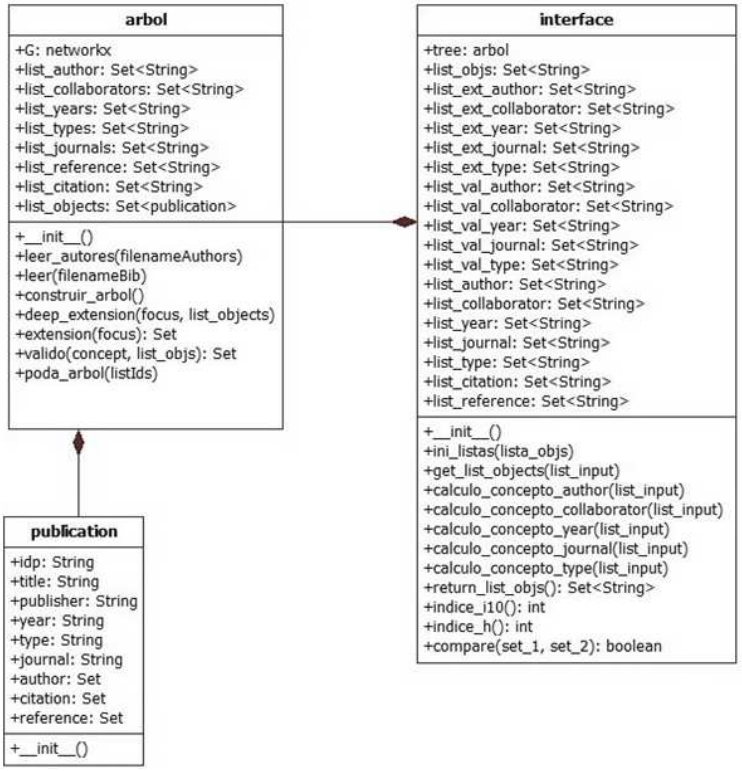

Figura 4.15: Diagrama de clases. Estas clases son las más representativas para la construcción de una taxonomía de acuerdo a la información suministrada al sistema. Se contemplan los métodos para poder realizar las exploraciones en la taxonomía. Se contemplan listas auxiliares para separar algunos resultados de cada grupo de facetas a explorar (p.e. *year, type, author*, etc).

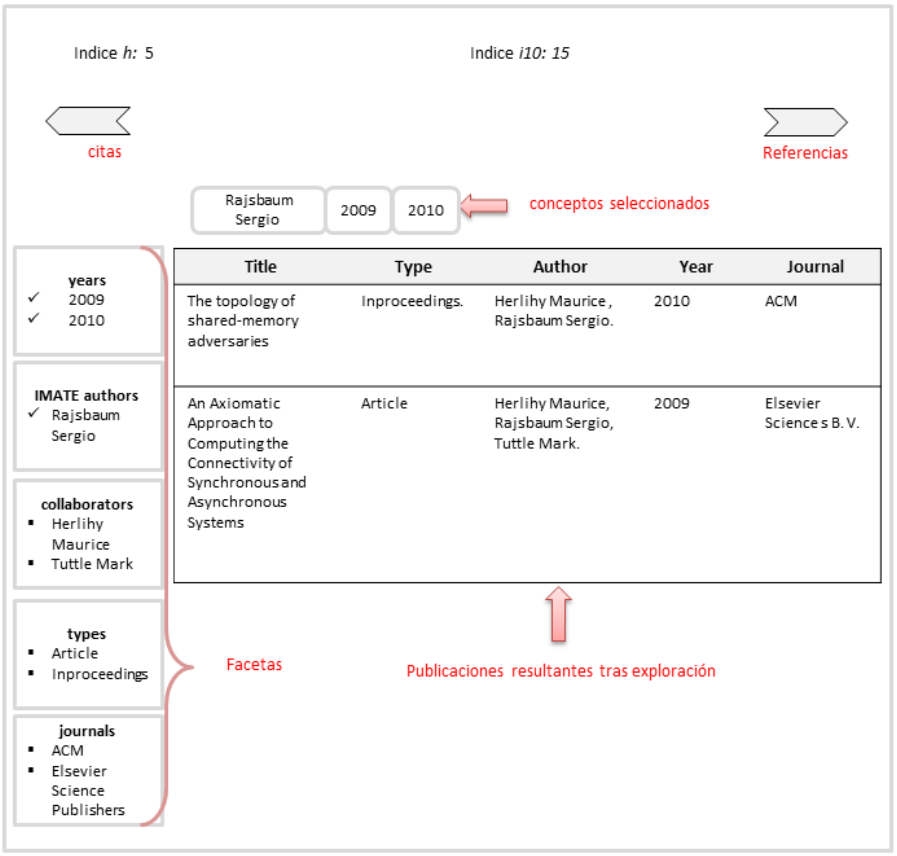

Figura 4.16: Diseño de la interfaz. Este diseño contempla 3 navegaciones facetadas que pueden visualizarse mediante las señalizaciones laterales de "citas" y "referencias", aunque por cuestiones de espacio sólo se muestra una por pantalla. Cada navegación facetada corresponde a cada grupo de publicaciones, las publicaciones principales a tratar, sus correspondientes citas y referencias.

4.4. Conclusiones

En este capítulo mostramos algunas interfaces de nuestro entorno de sistema. Estas interfaces son proporcionadas por los diferentes productos y cada uno de ellos cubre parte de nuestra especificación de requerimientos. Sin embargo como se ha visto en el capítulo anterior, no se ha encontrado un componente que resuelva efectivamente la exploración de publicaciones y elementos relacionados (citas y referencias) de manera consistente, por lo que este capítulo cubrió el diseño de un nuevo componente que satisfaga estas necesidades.

Capítulo 5

Desarrollo e integración

En esta sección se cubren las fases de *desarrollo* e *integración* de componentes al sistema.

Primero se describe los detalles del proceso de desarrollo del complemento para consultas y posteriormente algunos detalles de integración entre componentes.

5.1. Desarrollo del componente para consultas

Esta sección esta dedicada a la descripción del desarrollo del producto *matem.facetedbibliography*. Este producto crea navegaciones facetadas de un conjunto de publicaciones y elementos relacionados. Estos elementos relacionados son las correspondientes citas y referencias al conjunto de publicaciones dado.

Para el desarrollo de este producto se utilizan modelos y diseños en fases anteriores.

Recordamos que buena parte de la teoría detrás de este producto se encuentra en la sección 2.4 del marco teórico.

La estructura se muestra a continuación:

```
 matem.facetedbibliography
+ docs
+ src
    + matem
        + facetedbibliography
            - __init__.py
            - faceted.py
            - faceted_view.pt
            - configure.zcml
            + modules
              - __init__.py
              - publication.py
```

```
                - arbol.py
                - interface.py
              + js
                - bootstrap.js
              + css
                - bootstrap.css
                - cssfaceted.css
- setup.py
- README.txt
```

Donde las clases encargadas de la lógica de negocio se encuentran en el directorio `src/matem/facetedbibliography/modules`. Y las clases encargadas de la vista y administración de peticiones en `src/matem/facetedbibliography/`. Las definición de dependencias de nuestro producto se señalan en el archivo de configuración *setup.py* entre otros detalles (autor, versión, requerimientos extra, la ruta de algunos paquetes, etc.). Para mayor información en cuanto al desarrollo de productos consulte http://developer.plone.org/getstarted/paste.html#introduction.

5.1.1. Lógica de negocio

Clase *publication*

La primera clase a desarrollar es la clase *publication* que encapsula la información de las publicaciones en el sistema. Una instancia de esta clase representa a una determinada publicación en el sistema.

Instancias de la clase *publication* forman parte de la taxonomía creada en la siguiente clase.

Clase *arbol*

La clase *arbol* construye la taxonomía a partir de los datos almacenados en el sistema. Por ejemplo: figura 4.8.

Para distinguir a los investigadores del *Instituto de Matemáticas* del resto, la taxonomía se auxilia de un archivo de texto plano con las definiciones de las firmas de los miembros del IM, empezando por el apellido seguido del nombre o inicial (según sea el caso); se omiten las comas y se escribe una firma por linea. Este archivo es almacenado en el sistema con el nombre de *investigadores* en el directorio del usuario correspondiente.

Ahora bien, en el ejemplo de la taxonomá para publicaciones (figura 4.8) los nodos etiquetados por los identificadores de las publicaciones contienen la instancia de la publicación que representan. Los nodos que representan un *concepto* no necesitan más que la información que representan.

Para la construcción de la taxonomía pueden crearse desde cero las clases permitan gestionar y construir grafo apropiadamente.

Sin embargo se sabe que Python ofrece diversas opciones en forma de librerías que facilitan la creación y gestión de diferentes estructuras de datos, entre otras funcionalidades.

Es así como en la búsqueda de estas opciones, se ha encontrado la librería *networkx* [29] que permite la creación de grafos con cualquier tipo de contenido en sus nodos, grafos dirigidos o no dirigidos, creación de aristas con algún peso, funciones auxiliares (grado, número de nodos, vecinos, predecesores, sucesores etc.) entre otras características.

Así que después de la instalación de esta libreria, sólo queda añadirla a nuestra clase *arbol*:

```
import networkx as nx
```

De esta manera se facilita la creación de la taxonomía tras el procesamiento de datos de entrada y la atención se centra en los principales métodos para la gestión de la taxonomía, como son:

- La extensión profunda: para obtener los objetos relacionados a un concepto. La entrada es entonces un concepto y la salida es la lista de identificadores de los objetos resultantes. Los métodos encargados de esto en la clase *arbol* son *deep_extension* y *extension*. El método *deep_extension* es auxiliar (recursivo) del método *extension*, por lo cual el método principal es *extension*.

- El conjunto de conceptos relacionados a un conjunto S de objetos: toma como entrada una lista de identificadores de objetos (como conjunto S) y un concepto. Obtiene los conceptos incluidos o clasificados bajo el concepto de entrada que están relacionados con los objetos representados con la lista de identificadores. El método encargado de esto es *valido*.

- Poda taxonomía: remueve los conceptos que no vamos a utilizar. Este método hace referencia a la reducción de la taxonomía cuando ésta es explorada.

Clase *interface*

La clase *interface* gestiona a la taxonomía a modo que entregue resultados tras una determinada consulta.

Esta consulta puede estar conformada por varias entradas para perfeccionar la búsqueda de lo que se requiere.

El modo de operación de las consultas en una taxonomía se describe más detalladamente en la sección 4.3.1

La clase se auxilia de listas que almacenan información relevante del grafo y del estado actual de los conceptos tras la consulta. Por ejemplo los conceptos relacionados al conjunto de objetos resultantes (consulte la sección 2.4.1). Estos conceptos cambian a medida que se obtienen nuevos resultados en una nueva consulta. En la interfaz de consulta estos conceptos pueden resaltarse para indicar la relación de los conceptos con los resultados obtenidos.

Los métodos más relevantes para la gestión de esta clase son:

- Obtener la lista de objetos: que recibe como entrada una lista de conceptos a consultar y regresa la lista de los identificadores de los objetos resultantes. El método encargado de esto es *get_list_objects*.

- Cálculo de índices: una vez que se tiene el grafo construido tras el procesamiento de la información se procede a calcular los índices *h* e *i10* . Los métodos encargados de esto son *índice_h* y *índice_i10*.

5.1.2. Vista controladora

Plone tiene un mecanismo para generar páginas utilizando Python y opcionalmente un template de ZPT (consulte la sección 1.4.1), mediante *vistas* y *viewlets*.

Una *vista* es una clase de Python que usualmente se asocia con un template para producir una página web. Un *viewlet* es un fragmento de HTML generado por una clase de Python y un template, de manera similar a las vistas. La diferencia es que los *viewlets* se integran en una estructura de página definida por una serie de contenedores conocidos como *viewlet managers*, mientras que las vistas son totalmente independientes.

En este caso nos valemos de las *vistas* para generar la interfaz gráfica a nuestro modo y de manera independiente a la estructura de contenedores.

En este caso nuestra clase controladora debe ser una clase de Python que administre las peticiones generadas por los usuarios.

El template asociado ha de generar la interfaz gráfica del usuario, combinando código HTML, CSS, Javascript, y expresiones en *Lenguaje de Atributo de Plantilla* o TAL.

Este template envía peticiones a la clase controladora que se encarga de la administrar las peticiones de los usuarios. A su vez la clase controladora crea las instancias necesarias para responder a las peticiones.

Cabe señalar que la asociación entre la clase controladora y el template queda definida en el archivo de configuración *configure.zcml* basado en el *Lenguaje de Marcado de Configuración Zope* o ZCML (*Zope Configuration Mark-up Language*) de la siguiente manera:

```
<browser:page
name=‘‘faceted_bib’’
for=‘*’
class=‘‘.faceted.FacetedView’’
template=‘‘faceted_view.pt’’
permission=‘‘zope.Public’’
/>
```

Donde *class* hace referencia al nombre del archivo en Python y la clase dentro de ese archivo, *name* al nombre que se ha de mostrar en la navegación del sitio cuando se accede a esa vista y *template* para asociar el template correspondiente (*faceted_view.pt*).

Este archivo también contiene registrados los recursos que han de asociarse al producto, como son los archivos javascript y css.

faceted.py: class FacetedView

La clase mediadora entre la lógica de negocio y el template que genera la interfaz gráfica es FacetedView incluida en el archivo *faceted.py*.

Esta clase debe poseer al menos dos métodos importantes para la interacción con el template asociado: el método *__init__* y el método *__call__*.

El metodo *__init__* inicializa la(s) instancia(s) de clase. El metodo *__call__* permite que las instancias de la clase se comporten como funciones. Puede ser útil en clases cuyas instancias necesiten cambiar de estado.

La invocación al método *__call__* resulta ser una forma intuitiva y elegante para cambiar el estado de los objetos.

De este modo el método *__call__* se encarga de cambiar el estado de la instancia para responder a las solicitudes capturadas desde la interfaz gráfica.

Por lo tanto, la estructura mínima de una clase controladora es la siguiente:

```
from Products.Five import BrowserView
class FacetedView(BrowserView):
       template = ViewPageTemplateFile(‘‘faceted_view.pt’’)
       def __init__(self, context, request):
             self.context = context
             self.request = request
       def __call__(self):
             return ‘‘hello word’’
```

En particular la clase FacetedView incluye las instancias de lógica de negocio que ayuden a responder a las solicitudes de los usuarios.

En la creación de estas instancias se toman los datos almacenados en el sistema (obtenidos por el producto *CMFBibliographyAT*).

En la siguiente sección se detalla el proceso de integración entre el presente producto y *CMFBibliographyAT* para describir el proceso de adquisición de información almacenada en las carpetas tipo *Bibliography Folder*.

Sin embargo nos basta saber por el momento qué datos tomar.

Para eso necesario identificar qué datos corresponden a qué instancia. En nuestro caso debemos tener creadas 3 carpetas de tipo *Bibliography Folder* nombradas en el sistema de las siguiente manera:

- carpeta *publicaciones*: contiene las publicaciones de interés principal, los artículos de determinado miembro del IM. Con estas publicaciones se genera la llamada *taxonomía principal*.
- carpeta *citas*: contiene las citas de las publicaciones de interés principal. Genera la taxonomía relacionada a las citas.
- carpeta *referencias*: contiene las referencias de las publicaciones de interés principal. Genera la taxonomía relacionada a las referencias.

De cada carpeta deben crearse las correspondientes taxonomías. Éstas se inicializan y toman lugar en el método *__init__* de la clase FacetedView.

Con la información de cada taxonomía se construye la interfaz gráfica a partir de métodos definidos en la clase controladora que retornan algún tipo de información de estas taxonomías.

Por ejemplo:

```
def indice_h(self):
      return self.interface_principal.indice_h()
```

El método *indice_h* retorna el valor del índice h de la taxonomía principal. Éste valor se obtiene en el template mediante expresiones TAL. Esto se verá con mayor detalle en la siguiente subsección.

En cuanto a la percepción de solicitudes en la clase controladora éstas se manejan en el método *__call__* donde se verifica si algún evento en particular ha tomado lugar.

Por ejemplo al seleccionar una faceta/concepto se envía un *submit* del formulario que contiene los *checkboxes* seleccionados.

Luego en la clase controladora se verifica si este evento se ha suscitado.

De ser así, se reciben las entradas seleccionadas y se envían estos datos a las taxonomías correspondientes para obtener de ellas los resultados deseados.

Tras estas operaciones, las taxonomías obtienen nuevos datos y por lo tanto el template debe actualizar la información que despliega en la interfaz.

Esta actualización se indica en la clase controladora. Posteriormente se regresa el template asociado con los datos actualizados.

faceted_view.pt

El template asociado a la clase controladora FacetedView.

Este template utiliza diferentes recursos para construir la interfaz gráfica del usuario, tales como código o archivos javascript y css. También se vale del *Lenguaje de Atributo de Plantilla* o TAL para obtener los datos de la aplicación a través de la clase controladora.

Para construir la interfaz gráfica en su totalidad partimos del diseño de la figura 4.16 presentado en el Capítulo 4.

Los valores de las facetas *autor, collaborator, type* y *year* son presentadas como casillas de verificación, las cuales tras ser seleccionadas entregarán los resultados requeridos.

Sin embargo estas acciones llevan un proceso:

1. En el caso inicial se selecciona un valor de estas facetas en cualquiera de las navegaciones presentadas.

2. Esta accioń envía una petición con el dato seleccionado.

3. Se entrega esa petición a la clase controladora y se obtienen los resultados respectivos.

4. La interfaz se actualiza respecto a la casilla seleccionada. Es decir se muestran las publicaciones que cumplan con la petición requerida.

5. La casilla seleccionada se mantiene activa para realizar consultas más efectivas y de mayor alcance.

   ∗ Para deshabilitar una casilla basta con seleccionarla nuevamente.

Este proceso puede seguir hasta que hayamos alcanzado los resultados deseados.

Cabe señalar que las navegaciones facetadas restantes quedan influenciadas por los resultados obtenidos y también cambian sus datos tras la exploración.

Con ello obtenemos sólo elementos vinculados con nuestra exploración y nada más. Ver figuras 5.1, 5.2 y 5.3.

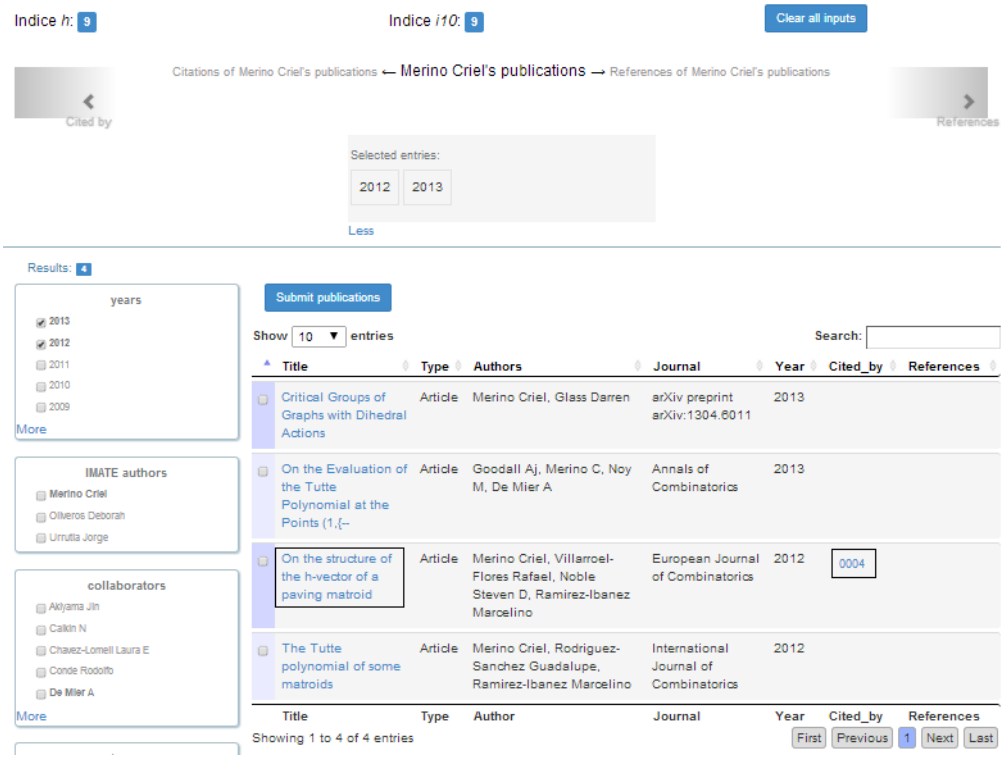

Figura 5.1: Navegación facetada de las publicaciones de un investigador con dos conceptos explorados: 2012 y 2013. En este caso sólo hay un elemento con citas, el cual es señalado en un recuadro.

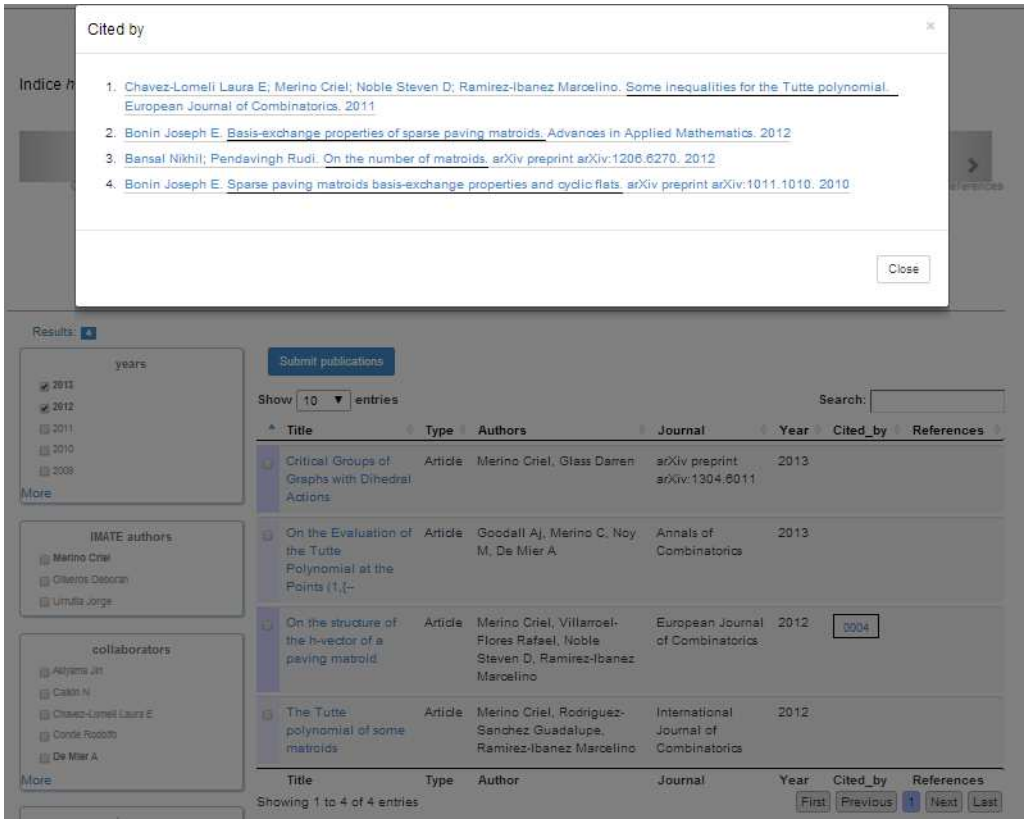

Figura 5.2: Consulta de citas de la publicación marcada en la figura anterior 5.1.

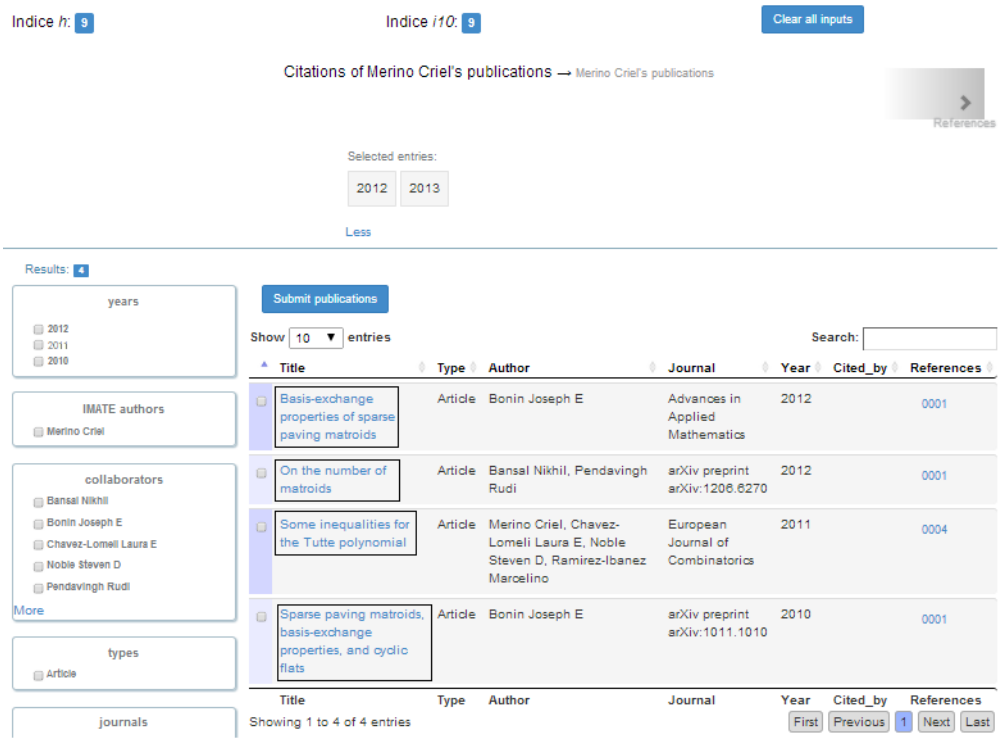

Figura 5.3: Navegación facetada de las citas de una publicación mostrada en las figuras anteriores 5.1 y 5.2. A esta pantalla se llega a través de las señalizaciones *Cited by* y *References* en la parte superior de los resultados y por debajo de los índices (en este caso se utilizó *Cited by*).

Por otra parte se han integrado algunos elementos a la interfaz propuesta en la fase de diseño, esto para hacer más fácil su manejo. También para aportar mayor información en la consulta de publicaciones. Algunos de estos elementos en la interfaz se describen a continuación:

- La visualización de citas y referencias de cada elemento en la interfaz de consulta.

  Estas opciones de citas y referencias deben estar disponibles para cada elemento en la interfaz de consulta. La información solicitada se entregará a través ventanas emergentes al momento de seleccionar la opción deseada. Esta característica pudo observarse en la figura 5.2.

- La posibilidad de selección de cualquier publicación para la exploración de los elementos asociados.

  Es decir, la posibilidad de seleccionar una o más publicaciones dadas. Esto para dar pie a la exploración de citas o referencias de determinados artículos.

Se propone implementar casillas de verificación para seleccionar cada publicación deseada. Sólo que a diferencia de las facetas, la selección de alguna publicación no provocará el envío automático de información, ya que si esto sucediera, el conjunto final de publicaciones se reducirá al primer elemento seleccionado.

De esta manera, el envío de información queda a cargo de otro elemento (el botón *Submit publications*). Así se pueden elegir tantas publicaciones como sean necesarias y se puede proceder posteriormente al envío de información cuando nuestra selección haya terminado.

- La posibilidad de limpiar todas las consultas.

  En la exploración conceptos, una vez que se tienen los resultados deseados, se llega a un punto donde el regreso al estado inicial implicaría la deselección de cada concepto consultado. Motivo por el cual es necesario tener una opción que nos evite este proceso; un elemento que deseleccione cada concepto y nos regrese al estado inicial de las taxonomías en un sólo paso. Esta opción en la interfaz se refiere en concreto al botón *Clear all inputs* ubicado en la parte superior derecha de la interfaz de consulta.

- Señalización de conceptos relacionados a las publicaciones resultantes.

  El saber qué conceptos tienen relación con el conjunto de publicaciones a explorar puede resultar una guía excelente en el refinamiento de la información. Esto, debido a que podemos conocer las posibles alternativas a explorar y evitar en algunos casos resultados nulos.

  En la interfaz de consulta, los conceptos relacionados quedan resaltados en "negritas". Es decir los valores de las facetas que están en negritas se relacionan directamente con las publicaciones presentadas en la consulta.

  El conjunto de conceptos relacionados y cómo obtenerlos puede entenderse con mayor claridad consultando el marco teórico en el apartado dedicado a las taxonomías dinámicas dentro de la sección 2.4.

Por otra parte, para hacer más claro el desarrollo de la interfaz, mostramos un ejemplo de cómo obtener información a partir de la clase controladora al template a través de expresiones TAL o *Lenguaje de Atributo de Plantilla*.

En este caso obtenemos los valores de los conceptos que denotan el tipo de publicación (en la taxonomía los conceptos clasificados bajo el concepto *type*). Estos conceptos se toman de la taxonomía y se define un método en la clase controladora que los devuelve al template. Este método es llamado *show_types*. El valor retornado es una lista de tuplas que contiene el valor de la faceta y un valor de tipo *booleano* que indica si ese concepto está relacionado a las publicaciones mostradas en la interfaz gráfica. De ser así se señala destacando el texto asociado.
De esta forma, mostramos una lista y el código en el template que marcan a cada concepto como relacionado o no relacionado.

```
type=[('article',True),('inproceedings',False),('book',True)]
```

Código en el template:

```
<div class=‘‘types’’ tal:define=‘‘local type view/show_types’’>
      <ul>
         <div tal:repeat=‘‘x python:type’’ tal:omit-tag=‘‘’’>
         <h5>
         <li class=‘‘list-unstyled’’>
         <input type=‘‘checkbox’’ name=‘‘faceta:list’’ tal:attributes = ‘‘value
python:x[0];’’ onclick=‘‘this.form.submit();’’>
         <span tal:condition=‘‘python:x[1]==True’’><strong>
         <span tal:replace=‘‘python:x[0]’’>this </span></strong></span>
         <span class=‘‘text-muted’’ tal:condition=‘‘python:x[1]==False’’>
         <span tal:replace=‘‘python:x[0]’’>this </span></span>
         </li>
         </h5>
         </div>
      </ul>
</div>
```

Los conceptos que se marcan en negritas son los que cumplen la condición `‘‘python:x[1]==True’’`. La sentencia `tal:repeat=‘‘x python:type’’` funciona de manera similar a un ciclo *for* en programación, recorriendo cada elemento de la lista.

Para mayor información acerca del *Lenguaje de Atributo de Plantilla* o TAL, visite http://docs.zope.org/zope2/zope2book/AppendixC.html

5.2. Integración al sistema

En la implementación del componente para consultas *matem.facetedbibliography* se ha de adquirir la información almacenada en el sistema a través del producto *CMFBibliographyAT*.

La interfaz de consulta se integra a la plataforma Plone como una vista que se ha de aplicar a las carpetas que contengan la información a procesar.

Esta información esta contenida en carpetas tipo *Bibliography Folder* del producto señalado.

Como bien se sabe, el producto *CMFBibliographyAT* tiene la capacidad de importar archivos BibTex desde la interfaz de usuario. De la misma manera este producto ofrece la funcionalidad de exportación de uno o todos los archivos contenidos en una misma carpeta, dependiendo si nos encontramos inspeccionando algún contenido en particular (publicación) o nos ubiquemos simplemente en el directorio de publicaciones.

Por lo tanto, podemos inspeccionar cómo se realiza esta operación e implementar algo similar en nuestra clase controladora.

En principio inspeccionamos cada una de las partes del producto, a saber la estructura del producto *CMFBibliographyAT* es la siguiente:

```
CMFBibliographyAT
    + adapters
    + bin
    + browser
        - __init__.py
        - configure.zcml
        - export.py
        - import.pt
        - import.py
        - listitemformatter.pt
        - listitemformatter.py
    + content
    + docs
    + exportimport
    + interface
    + skins
    + tool
    + transform
    + www
    - config.py
    - configure.zcml
    - ...
    - setup.py
    - README.txt
```

Donde encontramos que en el directorio *browser* archivo *export.py* se encuentra la clase encargada de manejar la exportación de publicaciones contenidas en carpetas *Bibliograhy Folder*.

En esta clase (*BibliographyExportView*) se define el método *export*, el cual es invocado al momento de dar click sobre el botón *export*.

Con ello, este método obtiene las publicaciones contenidas en una determinada carpeta tipo *Bibliograhy Folder*.

Por lo que implementamos esta misma utilidad en nuestra clase controladora. Es decir:
La definición del método:

```
    def _getRenderer(self, context):
      utils = component.getAllUtilitiesRegisteredFor(IBibliographyRenderer,
context)
      for renderer in utils:
      if renderer.available and renderer.enabled:
             return renderer
      return None
```

y el retorno de resultados en la variable *pub*

```
self.renderer = self._getRenderer(self.context)
pub = self.renderer.render(self.context['publicaciones'], output_encoding =
output_encoding, msdos_eol_style=eol_style)
```

Con ello se obtiene la información bibliográfica en formato BibTex de la carpeta *publicaciones*. Aunque se debe de obtener también la información de las carpetas *citas* y *referencias* para la generación de la interfaz.

También ha de adjuntarse el archivo que define las firmas de los investigadores del IM en la ruta donde se encuentran las carpetas de las publicaciones almacenadas (archivo nombrado en el sistema como *investigadores*). Esto para distinguir quienes son los investigadores del instituto y crear las facetas correspondientes en la interfaz de consulta.

La vista creada (o interfaz de consulta) debe aplicarse al directorio que contenga las carpetas de los grupos publicaciones, en este caso, al directorio del investigador en cuestión.

De esta manera nuestro producto desarrollado se integra satisfactoriamente con los componentes instalados en la plataforma y adquiere información específica a explorar.

El código fuente del producto *matem.facetedbibliography* queda a disposició pública en el repositorio de control de versiones *GitHub* en la cuenta del *Instituto de Matemáticas* cuyo enlace es el siguiente: https://github.com/imatem/matem.facetedbibliography.

5.3. Pruebas del sistema

5.3.1. Preparación

En un principio hemos probado la funcionalidad de nuestros componentes mediante un conjunto pequeño de publicaciones.

Ahora bien, se necesita probar que el sistema funciona para conjuntos mayores de publicaciones y usuarios con datos que se aproximen a la realidad.

Para ello es necesario obtener el listado de las publicaciones de los investigadores del instituto así como los correspondientes elementos relacionados. Estos elementos deben ser principalmente las citas del correspondiente grupo de publicaciones del investigador en cuestión. Todas estas publicaciones deben

presentarse en el formato ya establecido: BibTex.

Una primera manera de adquirir esta información, es obteniendo cada una de las partes involucradas (las publicaciones y sus citas). Luego de ello, debemos indicar las relaciones entre los elementos, clasificarlos en grupos y suministrarlos al usuario correspondiente.

Labor que resulta exhaustiva tanto para el investigador como para el administrador del sistema (dependiendo de quién se encargue de esta labor).

Por lo que debemos formular una solución que derive en un menor esfuerzo para cualquiera que tenga permisos de administración de información al sistema.

Así en la búsqueda de soluciones, se ha encontrado un par de herramientas que en su conjunto pueden ayudar a conseguir la información requerida.

Una de ellas es la librería ***Mechanize*** para Python. Esta librería se utiliza para la automatización de las interacciones con un sitio web. Puede seguir enlaces y enviar formularios.

La segunda herramienta es la librería ***Beautiful Soup*** de Python. *Beautiful Soup* es utilizada para la extracción de datos de archivos HTML y XML. Proporciona formas idiomáticas de navegar, buscar y modificar el árbol de análisis [34]. Entre sus características más importantes se encuentran:

- *Beautiful Soup* ofrece algunos métodos simples y expresiones Pythonicas para navegar, buscar y modificar un árbol de análisis: una herramienta para la disección de un documento y extraer lo que se necesita.
- *Beautiful Soup* convierte automáticamente documentos entrantes a Unicode y documentos salientes a UTF-8. En caso de que el documento no especifique que una codificación y *Beautiful Soup* no pueda detectarla entonces sólo tiene que especificar la codificación original.
- *Beautiful Soup* se sitúa encima de analizadores Python populares como lxml y html5lib, lo que permite probar diferentes estrategias de análisis o velocidad para una mayor flexibilidad.

De esta manera *Beautiful Soup* analiza todo lo que se le da, y hace el recorrido de árbol por el usuario. Proyectos que habría tomado horas ahora sólo toman minutos con *Beautiful Soup* [34].

Por lo que, combinado estas herramientas se pretende la creación de *scripts* que ayuden al proceso de búsqueda y recopilación de información en formato BibTex.

Se busca con *Mechanize* hacer el seguimiento de los links y con *Beautiful Soup* extraer la información necesaria.

Ambas herramientas estan dirigidas al lenguaje Python para que estos *scripts* puedan integrarse en un futuro a la plataforma Plone. Aunque *Beautiful Soup* esté dirigida exclusivamente al lenguaje Python, *Mechanize* no lo está, esta queda disponible para Ruby, Java, Perl y por supuesto Python.

Ahora bien, estos *scripts* deben aplicarse a ciertas fuentes de información, como lo son bibliotecas digitales y/o portales que proporcionen catálogos de publicaciones científicas.

Entonces tras un análisis de estas fuentes se ha encontrado lo siguiente:

- Algunas bibliotecas digitales presentan un listado de publicaciones insuficiente asociado a un investigador. Menos del 50 % del total encontrado en alguna otra biblioteca digital.
- Existen bibliotecas digitales que no presentan información acerca de las citas de sus publicaciones disponibles.
- Las bibliotecas digitales que presentan información referente a citas la proveen de manera escasa.
- Sin embargo se ha encontrado que el portal de *Google Scholar* presenta mucho más información que cualquiera de las bibliotecas y/o portales analizados (ACM, springerlink, ieeexplore, sciencedirect, dblp, etc.) ya que este portal recopila información de diversas fuentes de manera automática. Aunque a veces esta información no es del todo consistente debido a duplicados, falta de acceso a recursos, etc. [1].

Si bien estas características no suenan del todo alentadoras, la mejor opción que se ha encontrado hasta el momento ha sido *Google Scholar*, como fuente para obtener la información necesaria.

Luego de haber seleccionado la fuente a tratar, queda analizar la manera en la cual se obtendrá la información dada la estructura del portal.

En principio cada investigador debe poseer un perfil en el portal de *Google Scholar*.

En este perfil se encuentran todas las publicaciones pertenecientes al investigador asociado.

En cada publicación mostrada se proporcionan opciones de exportación en diferentes formatos, incluido BibTex.

También para cada publicación se muestra un link hacia el conjunto de citas asociadas a ella. Análogamente en cada cita se muestran opciones de exportación del elemento en cuestión.

Esta estructura resulta muy útil, ya que siempre se ha de proporcionar opciones de exportación y las correspondientes citas para cada publicación.

Ahora bien, el siguiente paso es construir una serie de *scripts* que obtengan la información de *Google Scholar* y realicen algún procesamiento de datos. Esto, con el fin de relacionar las publicaciones principales con sus respectivas citas. Este proceso queda descrito más detalladamente en la siguiente subsección.

5.3.2. Scripts

En las primeras pruebas para obtener la información de las citas de una publicación en formato BibTex de una sola página (que contiene 10 elementos) resultó exitosa.

Sin embargo tras implementar este proceso para todas las de citas de una publicación, este proceso no pudo completarse.

[1] Para mayor información acerca de la gestión de publicaciones de *Google Scholar* consulte http://www.google.com/intl/en/scholar/citations.html

Ya que ocurre lo siguiente:

1. *Google Scholar* detecta tráfico unusual en la red y restringe la entrega de información a la IP asociada al tráfico.

2. Tras la detección del tráfico, *Google Scholar* permite el acceso a sus recursos únicamente a través de la interfaz gráfica. En dicha interfaz se solicita la comprobación de uso humano a través de captura de información de una imagen presentada. Ver figura 5.4.

3. Nuevos intentos por obtener la información restante (nuevas ejecuciones del script) quedan restringidos; recursos no disponibles o prohibidos.
   Ejemplo:
   `mechanize._response.httperror_seek_wrapper: HTTP Error 403: Forbidden`

4. El número de páginas obtenidas antes de que *Google Scholar* detecte y restrinja la recopilación automática de información, es alrededor de 6 (aproximadamente 60 publicaciones).

Figura 5.4: Solicitud para comprobación de uso humano de la interfaz: *Google Scholar.*

Entonces una nueva solución debe incorporar las siguientes condiciones:

1. Obtener una nueva IP para hacer las peticiones.

2. El número de resultados a entregar sea limitado, se estima que sean unas 50 publicaciones por ejecución del script.

3. En caso de que la nueva IP sea restringida se debe cambiar nuevamente.

Así en la búsqueda de herramientas que propocionen el cambio efectivo de IP pública se ha encontrado la herramienta Tor.

Tor es una red de túneles virtuales que permite a las personas y grupos mejorar su privacidad y seguridad en Internet [43]. *Tor* protege contra una forma habitual de vigilancia en Internet conocida como "análisis de tráfico". El análisis de tráfico puede usarse para deducir quién esta hablando a quién sobre una red pública. Conocer el origen y destino de nuestro tráfico de Internet permite a otros seguir nuestro comportamiento e intereses.

Así que para evitar esto, en lugar de tomar una ruta directa desde el origen al destino, los paquetes de datos en la red *Tor* toman caminos aleatorios a través de varios repetidores que tapan el rastro para que ningún observador de un único punto se puede decir de dónde procedían los datos o hacia dónde se dirigen [43].

De esta forma muchos tipos de datos pueden ser intercambiados y varios tipos diferentes de aplicaciones de software se pueden implementar en la red *Tor* .

Por eficiencia, el software de *Tor* utiliza el mismo circuito para conexiones que se establecen dentro de los mismos diez minutos más o menos . A las peticiones posteriores se les proporciona un circuito nuevo para evitar que alguien pueda asociar las primeras acciones con las nuevas [43]. Lo que provoca que se proporcione una IP diferente cada cierto periodo de tiempo.

Sin embargo el cambio de IP simplemente no soluciona este problema. Se debe vincular explícitamente la IP de la red de *Tor* con nuestro script. Ya que de otra manera el script tomara la IP real y seguirá enviando peticiones con esa IP. Lo cual resulta en ejecuciones fallidas.

Para ello es necesario la creación de un *socket* que tenga asociada la IP creada por *Tor* y el puerto del servicio de *Tor*. En nuestro script es el correspondente a añadir las siguientes lineas:

```
import socks
import socket

def create_connection(address, timeout=None, source_address = None):
        sock = socks.socksocket()
        sock.connect(address)
        return sock

socks.setdefaultproxy(socks.PROXY_TYPE_SOCKS5, "127.0.0.1", 9050)

socket.socket = socks.socksocket
socket.create_connection = create_connection
```

Entonces nuestro(s) script(s) debe ajustarse para enviar peticiones de información pequeñas (alrededor de 50 publicaciones) por cada ejecución, esto para que se evite, en medida de lo posible, las restricciones de *Google Scholar*.

Claro está, cada ejecución tendrá un dominio diferente de publicaciones a obtener. Por lo que primero se deben obtener los links a procesar. En este sentido

no hay restricciones por parte de *Google Scholar*, el problema se presenta cuando se intenta acceder a la información de las publicaciones (en este caso los archivos BibTex correspondientes).

También recordamos que la actualización de la IP, *Tor* debe ser una constante, de manera tal, que cada vez que se desee obtener una parte de la información se tenga una IP diferente. Esto por mencionar una solución.

La versión final de los *scripts* generados se presentan en el apéndice del presente documento conteniendo las características aquí descritas.

5.3.3. Tratamiento de la información

La información obtenida a través de *scripts* debe tratarse de manera tal que cada elemento tenga la información que le corresponde de acuerdo a las relaciones que tenga con otros elementos.

Para ello primero describimos el formato en el que se nos ha entregado la información.

De los archivos generados sólo consideramos los de tipo BibTex de publicaciones del autor y sus respectivas citas.

El archivo de publicaciones del autor sólo contiene la URL de la publicación (correspondiente a *Google Scholar*) y la información en formato BibTex, esto por cada publicación.

El archivo de citas contiene la URL de la publicacíon que citan y por debajo los correspondientes elementos citantes. Así de manera repetitiva, cada link con sus respectivas citas por debajo de ésta.

Sin embargo se han encontrado algunos inconvenientes que se deben tratar adecuadamente a fin de evitar problemas futuros. Uno de estos problemas surge cuando observamos las llaves de los elementos obtenidos, éstas se componen principalmente por el apellido de un autor, el año y la primera palabra del título de la publicación. Sin embargo esto no garantiza que estas llaves sean únicas, pudiesen existir publicaciones diferentes con llaves iguales.

También existe la duplicidad de información al momento de la descarga, parcial o totalmente, lo que puede causar inconsistencias al momento de relacionar la información.

Así en la asociación entre los elementos de los diferentes grupos a tratar se debe considerar lo siguiente: los elementos citantes pueden relacionarse con más de una publicación, por lo que pueden aparecer más de una vez en el archivo BibTex de citas y debemos considerar que cada réplica contenga la información de todas las publicaciones a las que hace referencia.

De las publicaciones con la misma llave se deben identificar las publicaciones que son iguales entre sí y las que no.

De la duplicidad de información parcial nos aseguramos que una publicación duplicada no tenga información diferente a otra de sus citas.

Esto a partir de la identificación de la URL y sus citas, se considerará solo el conjunto de citas más grande para una misma URL.

De la duplicidad de información total se encargará el SACB a través del producto *CMFBibliographyAT*, que se encarga de separar publicaciones duplicadas.

Estos procedimientos, para dar forma a la información, deben realizar mediante *scripts* de Python y Bash que pueden visualizarse en el apedice dedicado a *scripts*.

5.3.4. Pruebas finales

Finalmente cuando se ha procesado toda la información se realiza a la carga de ésta al sistema

Se habrá de verificar el soporte del sistema en cuanto al número de elementos a almacenar, al número de elementos a importar en un mismo archivo y de las diferentes consultas a los usuarios registrados.

Nótese que el soporte en cuanto al número de elementos a almacenar e importar se refiere queda a cargo del producto *CMFBibliographyAT*, sin embargo esta información no se detalla en la descripción del producto (portal Plone), por lo que queda averiguar el soporte descrito.

En cuanto a las consultas, estas quedan a cargo del producto desarrollado *matem.facetedbibliography*, cuyos alcancesx se probarán en este apartado.

Ahora bien, el primer punto que se debe considerar es la información del sistema donde se realicen las pruebas mencionadas, ya que ello depende en gran medida el rendimiento del sistema. En el apartado dedicado a la instalación de la plataforma (apéndice) se mencionan algunos requerimientos mínimos con los que debe contar nuestro sitio.

Sin embargo dadas las características de los productos instalados y desarrollados se puede afirmar que el sistema a probar va más allá de un sitio simple.

De este modo nuestro sistema pudiera demandar muchos más recursos de los mínimos requeridos y así afectar el rendimiento del sistema.

En este caso el servidor donde se aloja nuestro sistema cuenta con el sitema operativo `Ubuntu 12.10` (`GNU/Linux 3.5.0-30-generic x86_64`) y las características de memoria y procesador se presentan en los cuadros 5.1 y 5.2.

cat /proc/cpuinfo		
processor	:	0
vendor_id	:	AuthenticAMD
cpu family	:	15
model	:	47
model name	:	AMD Athlon(tm) 64 Processor 3200+

Cuadro 5.1: Información del procesador.

cat /proc/meminfo		
MemTotal	:	2051328 kB
MemFree	:	73880 kB
Buffers	:	137828 kB
Cached	:	954496 kB
SwapCached	:	1336 kB
...		

Cuadro 5.2: Información de la memoria.

Luego, una vez conocida esta información, se puede dar pie la realización de las pruebas y obtener de ellas algunas observaciones:

- En una prueba de carga al sistema se probó importar hasta 2177 en una sola tanda, esto con el sistema en modo desarrollo.
- Y en las consultas notamos que:
  - El tiempo de inicialización de la interfaz de consulta depende en gran medida de la cantidad de datos a explorar de los 3 conjutos de publicaciones a tratar. Es decir que una gran cantidad de elementos resulta en un incremento de tiempo en la carga de las 3 navegaciones facetadas. Comportamiento que, parece predecible tras recibir mayor cantidad de datos a la interfaz. Estos incrementos pueden observarse en el orden de algunos miles de elementos.
  - El tiempo de consulta a un concepto disminuyó respecto al tiempo de inicialización, debido a la reducción de la taxonomía tras la exploración; la entrega de datos resulta ser menor a la interfaz que en un estado inicial.
  - Cuando se selecciona más de un concepto antes de que el sistema entregue los resultados del anterior seleccionado, se observa que del lado del servidor se toma en cuenta cada click realizado a los conceptos y por lo cual se genera una petición por cada concepto de manera secuencial en lugar de una sola como se pensaría.

Donde podemos hacemos notar que:

- El soporte de importación de miles publicaciones a partir de un mismo archivo queda soportado por el sistema.
- De la misma manera el almacenamiento soporta una gran cantidad de elementos resguardar. Sin embargo esto no es sorprendente puesto que tratamos con archivos de texto plano.
- También notamos que el formato del contador de publicaciones en la importación contiene 5 lugares a considerar, lo que pudiera indicar que los

elementos a importar en una sola tanda pueden de ir de algunas unidades a decenas de miles de elementos.

- En cuanto a las observaciones obtenidas de las consultas se concluye que:
  - Al seleccionar más de un concepto antes de obtener una respuesta genera las operaciones correspondientes a los clicks realizados. No se omite ninguna operación, la generación de resultados tomará el tiempo acumulado de las selecciones realizadas. Por lo que se recomienda seleccionar un concepto cada vez y esperar los resultados en menor tiempo (se evita la modificación de las taxonomías repetidas veces).
  - El tiempo de respuesta de los resultados obtenidos dependerá en gran medida de la cantidad de las publicaciones a entregar. Entre mayor sea el conjunto de publicaciones a entregar mayor será el tiempo de respuesta y viceversa. Es por ello que la interfaz de consulta en un estado inicial tarda un poco más en generarse, la cantidad de elementos son todos los almacenados en el directorio del usuario a consultar.
  - También debemos tener en cuenta que el rendimiento del sistema se ve afectado por las condiciones físicas de nuestro servidor. El tener un sólo procesador para gestionar todas las operaciones y además mantener el sitio en funcionamiento puede afectar el rendimiento del sistema. Esto, claro eśta, depende de la cantidad de datos a gestionar y a entregar.

Todas estas pruebas nos dan una idea de lo que ocurre con nuestro sistema. Una gran cantidad de información a bajo las condiciones fisicas actuales del servidor puede decrementar el rendimiento del sistema. Esto puede verse más claramente cuando consultamos diferentes conjuntos de publicaciones con diferente número de elementos.

Sin embargo pese al decremento del rendimento se puede observar que el sistema soporta y entrega la información que se le ha solicitado.

5.4. Conclusiones

En el proceso de desarrollo del producto *matem.facetedbibliography* se detallan las clases encargadas de la lógica de negocio, vista controladora e interfaz gráfica. Se describe la relación con otros productos y su función en el SACB.

En el proceso de integración se muestra la manera en que se obtiene la información de las publicaciones almacenadas en directorios del producto *CMF-BibliographyAt* e incorpora al producto desarrollado *matem.facetedbibliography* para consultas a diferentes grupos bibliográficos depediendo la ruta (directorios de los usuarios) donde se implemente la vista de consultas.

Una vez establecido nuestro sistema se realizan pruebas de soporte de información y usuarios. La información suministrada al sistema es obtenida a través de medios automáticos o semi-automáticos: *scripts*.

Los *scripts* ayudan a la recopilación de información bibliográfica desde el portal de *Google Scholar*. Cada ejecución obtiene un número limitado de publicaciones, debido a las restricciones de acceso impuestas por *Google Scholar*. Por lo que el número de ejecuciones a repetir depende en gran medida de la cantidad de publicaciones a obtener.

La información obtenida contiene ciertos detalles a pulir y manejar de acuerdo a la sintaxis establecida para la asociación de información. También se ha de manejar la duplicidad de información parcial y total, así como la duplicidad de llaves a elementos que no necesariamente son iguales.

Tras el procesamiento de información se procedió a la realización de las pruebas, obteniendo así los siguientes puntos:

- El soporte de importación de miles publicaciones queda soportado por el sistema.
- El soporte de almacenamiento de información también queda cubierto por el sistema.
- El tiempo de respuesta en la interfaz de consulta varía respecto a la cantidad de información entregada; comportamiento comúnmente presentado en el orden de algunos miles de elementos.

Con ello concluimos en general que nuestro sistema es capaz de soportar grandes cantidades de información. El almacenamiento de datos no llega a ser considerable, ya que la información almacenada es meramente texto. Que el tiempo de respuesta se ve influenciado por la cantidad de elementos a entregar y en gran medida también depende de las capacidades fisicas del servidor para gestionarlos.

En este caso nuestro servidor cuenta con recursos un tanto limitados que afectan el rendimiento del sistema.

Sin embargo podemos concluir que nuestro sistema puede soportar y gestionar miles de publicaciones por cada usuario.

Capítulo 6

Resultados, conclusiones y trabajo futuro

El sistema administrador de citas bibliográficas se propuso como una solución al problema de la escasa gestión de información bibliográfica encontrada en las diversas bibliotecas digitales. En este sentido se ha encontrado que muchas de las bibliotecas digitales no ofrecen información completa acerca de publicaciones dadas a un determinado investigador.

Tampoco existen muchas herramientas dedicadas a la gestión y presentación adecuada de esta información.

De esta manera el SACB pretende proveer las vías necesarias para el resguardo y presentación de información bibliográfica.

Por lo que este sistema ofrece al usuario las siguientes características que ayuden al usuario a alcanzar una buena administración:

1. En principio se ofrece cuentas de acceso al sistema vía Web a los usuarios al sistema.

2. Se proporciona un espacio reservado (directorio personal) al investigador dentro del directorio del *Instituto de Matemáticas*.

3. Se provee de directorios especiales que contendrán los conjuntos de publicaciones a almacenar.

4. En estos directorios se proporcionan las opciones para el almacén de información bibliográfica, a través de la importación de archivos o llenado de formularios para los diferentes tipos de publicaciones existentes.

5. Se genera una interfaz de consulta con base a la información almacenada. Es decir, las publicaciones del investigador y elementos relacionados. Esta interfaz de consulta se refiere más específicamente a una interfaz de navegación facetada del conjunto de publicaciones a consultar.

6. Se proporcionan además ciertos índices de impacto en la interfaz de consulta.

Estas carcaterísticas proporcionan al usuario las siguientes ventajas:

1. Una cuenta en un sistema que puede ser accesado vía Web, evita la instalación de cualquier tipo de software. Lo que proporciona un mismo sistema para todos los investigadores.

   Una cuenta propia al sistema garantiza que pueda acceder a recursos propios del instituto; recursos que pueden ser de caracter privado a usuarios externos.

2. Al poseer un directorio propio, el investigador adquiere los permisos de administración de los recursos contenidos. Por lo que puede agregar, actualizar o eliminar contenido. Ningún investigador puede modificar recursos de otro investigador.

3. Directorios del tipo *Bibliography Folder* separa el contenido bibliográfico del resto. Es más fácil de gestionar carpetas que contienen un mismo tipo de contenido (incluso para un desarrollador).

4. La importación de archivos en formato *BibTex* hace mucho más rápido el trabajo de ingreso de las publicaciones al sistema.

5. Una navegación facetada permite al usuario filtrar información específica y facilita las búsquedas de determinadas publicaciones.

6. Esta navegación, separa el conjunto de autores en general en dos conjuntos: uno con los investigadores propios del *Instituto de Matemáticas* y otro con autores fuera del mismo. De esta manera localizamos inmediatamente a los colaboradores dentro del instituto.

7. La exploración a través de facetas de los elementos relacionados (citas y referencias) proporciona información extra que no se provee de forma tan clara en las diferentes bibliotecas digitales. En ese sentido podemos conocer (como usuario investigador) por ejemplo, las publicaciones más recientes que nos han citado, las auto-citas o las primeras citas que obtuvimos.

   Las consultas en general estan pensadas para que adquieran un carácter público o al menos permisivo dentro de los miembros del *Instituto de Matemáticas*, así cualquier investigador puede consultar las publicaciones de sus colegas.

En la siguiente sección se presentan algunas conclusiones obtenidas tras la implementación del actual sistema. Conclusiones derivadas de emplear algunos enfoques y herramientas en este sistema. Conclusiones del estado actual del sistema (alcances) y de su posible mejora. Las propuestas de mejora del sistema se presentan como *trabajo futuro*.

6.1. Conclusiones

En cuanto al desarrollo e implementación de este sistema basado en CBSE se concluye u observa lo siguiente :

- El enfoque CBSE reduce la cantidad de software a desarrollarse lo cual influye en la reducción de costos y riesgos
- La implementación de componentes puede cambiarse sin afectar al resto del sistema. Debido a que estos componentes tienen como características debilidad de acoplamiento e independencia del sistema.
- Además las infraestructuras de componentes proporcionan plataformas de alto nivel que reducen los costos de aplicaciones. En este caso esta plataforma es un CMS.

De acuerdo a la utilización del CMS Plone como plataforma proporcionó principalmente:

- Organización y localización de contenido,
- Las reglas de contenido para otorgar permisos de administración de ciertos tipos de contenido a usuarios predeterminados.
- La búsqueda de información inteligente a través de metadatos adjuntos a los datos suministrados.
- La separación clara de elementos que conforman al CMS , como lo son la seguridad, lógica de negocio y plantillas por mencionar algunos.
- La posibilidad de integración de componentes a la plataforma de una forma sencilla al administrador.
- La posibilidad de crear nuevos componentes de manera independiente sin alterar la plataforma, esto, con base a una serie de reglas de configuración y organización que los componentes deben seguir.
- También ofrece el establecimiento en modo desarrollo de la plataforma para la creación de nuevos componentes de manera cómoda al desarrollador.
- Ofrece además la posibilidad de configuración/edición de los componentes y la plataforma misma a través de la interfaz de administración de zope ZMI.

Estas características permitieron crear un sistema seguro, estable y fácil de administrar. Esto una vez que se ha aprendido el uso, instalación/configuración de productos y la plataforma misma en sí.

Tambien se permitió el desarrollo de componentes de manera “rápida” una vez que se ha aprendido la manera de crear productos para esta plataforma (la estructura, configuración y lenguajes necesarios).

En cuanto a la incorporación de componentes, esta pudo realizarse de manera cómoda, obteniendo pocas fallas de integración. [1]

En cuanto a los alcances/limitaciones del sistema se debe tener en cuenta lo siguiente:

- La información presentada en el sistema debe proporcionarla el usuario correspondiente o administrador del sistema.
- Esta información es obtenida a través de *scripts* que consultan el portal de *Google Scholar* y consiguen el BibTex requerido.
- Para generar la interfaz de consulta deben proporcionarse los elementos a explorar: las publicaciones del investigador, las citas correspondientes y las referencias utilizadas en carpetas tipo *Bibliography Folder*, una por cada grupo de publicaciones. Cada carpeta con el nombre correspondiente: *publicaciones*, *citas* y *referencias*.
- El tiempo de respuesta en cada consulta se ve influenciado por la cantidad de elementos a entregar y de las capacidades físicas que el servidor tenga para gestionarlos. En este caso nuestro servidor cuenta con recursos un tanto limitados que afectan el rendimiento del sistema.

De donde podemos obtener propuestas de mejora al sistema implementado comunmente llamado *trabajo futuro*.

6.1.1. Conclusiones generales

En lo que respecta a las conclusiones generales obtenidas tras la implementación del sistema y acciones derivadas de ella, se obtiene lo siguiente:

- Una de las aportaciones más importantes en el desarrollo de este proyecto fue el desarrollo de *scripts* que obtuvieron los archivos BibTex correspondientes a las publicaciones de los miembros del IM y sus respectivas citas.

  Con ello, el usuario evitó la búsqueda exhaustiva de estas publicaciones y la vinculación manual de ellas con sus respectivas citas.

  Cabe señalar que la informacioń de una publicación respecto a sus citas no viene en el registro de exportación del artículo (en cualquier formato).
- Luego de la recolección e importación de información al sistema, se pudo examinar el contenido dando pie a la correción de alguno errores provenientes de la fuente de información.
- De la consideración de las citas, se obtuvieron los índices relacionados al impacto y producción científica. La fiabilidad de estos indicadores depende en medida de la cantidad de información suministrada al sistema (respecto a la producción real del investigador) y de la correcta vinculación de las publicaciones con sus respectivas citas.

[1] Estos detalles de implementación pueden consultarse en el apéndice del documento.

- De la presentación de navegaciones facetadas se permitió la exploración de las publicaciones mediante la consulta de los atributos más generales que la mayoría de las publicaciones posee, como son: el año, tipo, autor y la revista.

  También se pudieron realizar exploraciones más profundas de los elementos adjuntos a las publicaciones, es decir, sus citas y referencias. Esto con la finalidad de obtener mayor análisis de citas y/o referencias relacionadas.

  Por ejemplo a informarnos un poco más acerca de las rutas que han tomado los trabajos de investigación realizados, los investigadores relacionados a esos trabajos, a la contabilización de auto-citas, a la determinación de citas por año y su identificación, etc.

Con todos estos puntos podemos concluir que el sistema desarrollado provee de las herramientas necesarias para alcanzar una buena gestión de la información relacionada a publicaciones científicas y principalmente sus citas en relación, atrayendo consigo información útil y relevante al investigador.

6.2. Trabajo futuro

Como se sabe, en una primera versión del sistema, se obtuvo la información de las publicaciones del portal de *Google Scholar* haciendo uso de scripts.

Este proceso hizo el trabajo de búsqueda de publicaciones más fácil y en menor tiempo, tanto de las publicaciones para algún determinado investigador como sus respectivas citas.

Posteriormente una vez obtenida esta información, la carga al sistema se hizo a manera de importación. Sin embargo, este aspecto puede mejorarse en futuras implementaciones: la información pudiera cargarse al sitio sin necesidad de importación, es decir de manera automática al directorio correspondiente, según sea el caso.

Este proceso automático debería realizarse cada cierto periodo de tiempo con el fin de actualizar la información presentada. Ya que sabemos que las citas tienden a incrementarse al igual que la productividad.

Por otra parte, otro aspecto a considerar es la generación de nuevos *scripts* que puedan acceder a otros portales aún sin explorar. Esto, con el fin de obtener una mayor consistencia en la información de la que se tiene presentada actualmente.

En cuanto a rendimiento se refiere, podemos considerar en futuras implementaciones un entorno más potente para el resguardo de nuestro sistema, es decir un equipo con mejores capacidades fisicas, utilerías, herramientas u otras alternativas que ayuden a dar respuestas mucho más rápidas a partir de grandes conjuntos de publicaciones.

También consideramos la visualización de ciertos detalles encontrados en estos contenidos bibliográficos. Estos detalles pueden ser presentados en forma de gráficas u otros medios para dar una visión más general sobre aspectos a considerar importantes. Por ejemplo gráficas de citas por años, autores más

relacionados a nuestros campos de estudio y otras medidas de impacto (quizá de mayor complejidad), etc.

Todos estos aspectos harían de este sistema un poco más consistente, fácil y claro en algunos aspectos en el análisis de citas. Evidentemente con cada nueva versión del sistema se tendría el objetivo de alcanzar un mayor grado de veracidad en la información, y utilidad de ésta al usuario.

Apéndice A

Anexos

A.1. Instalación

Antes de iniciar la instalación de la plataforma Plone se debe asegurar de que cumple con los requisitos al menos minimos para su implementación.

Algunos de los más importantes requerimientos son:

- Hardware
  - Un mínimo de 256 MB en RAM y 512 MB de espacio swap por sitio Plone.
  - Un mínimo de 512 MB en disco duro.
- Software
  - Windows XP o posterior
  - Linux 2.6.x o posterior
  - OSX 10.4.x o posterior
  - Python 2.6 (optionalmente Python 2.7 para Plone 4.2 en adelante)

Para mayor información de los requerimientos consulte: http://plone.org/documentation/kb/plone-system-requirements.

Una vez que se ha verificado que se cumple con los requisitos iniciamos con las instrucciones propias para la instalación y configuración del CMS.

Así el proceso de instalación del CMS Plone consiste de los siguientes pasos:

1. Descargar del sitio *Plone.org* la distribución a instalar, en este caso seleccionamos la última versión (4.3).
2. Desempaquetamos el archivo con extensión tgz.

   ```
   tar -xvzf Plone-4.3-UnifiedInstaller.tgz
   ```

3. Dentro del directorio correspondiente ejecutamos el archivo *install.sh* para consultar las opciones disponibles en la instalación.

   En este caso creamos sólo una instancia de zope independiente indicando la ruta de Python.

   ```
   ./install standalone --with-python=/path/to/Python-2.7/bin/python2.7
   ```

4. Posteriormente nos cambiamos al directorio de nuestra instancia creada y ejecutamos lo siguiente:

   ```
   /path/to/Python-2.7/bin/python2.7 bootstrap.py
   ```

5. Editamos el archivo *buildout.cfg* para indicar el puerto

   ```
   http-address = 9090
   ```

   y agregar algunos productos:

   ```
   eggs=
         Plone
         Pillow
         plone.reload
         collective.js.jqueryui
         Products.CMFBibliographyAT
         Products.ATExtensions
         bibliograph.core
         bibliograph.parsing
         bibliograph.rendering
         pyisbn
         Products.FacultyStaffDirectory
         medialog.subskins
         collective.sortmyfolder
         matem.facetedbibliography
         collective.js.datatables
         plone.api
   ```

6. Cuando el sitio esté listo, procedemos a abrir un navegador con la dirección y puerto correspondiente, en este caso es:

   alamo.matem.unam.mx:9090

7. La interfaz nos solicitará un usuario y constraseña para poder acceder al sitio.

   La constraseña y usuario se encuentran en el archivo *adminPassword.txt* y se crea automáticamente al crear la instancia del sitio.

   La contraseña se puede definir al momento de crear la instancia añadiendo la opción `--password` y enseguida contraseña por ejemplo:

```
./install standalone --with-python=/path/to/Python-2.7/bin/python2.7
--password=mypassword
```

8. Una vez que hemos iniciado sesión se muestra un formulario para ingresar los datos del nuevo sitio y los paquetes a ser activados.

   Un ejemplo de los principales paquetes a instalar se muestra a continuación: Ver figura A.1.

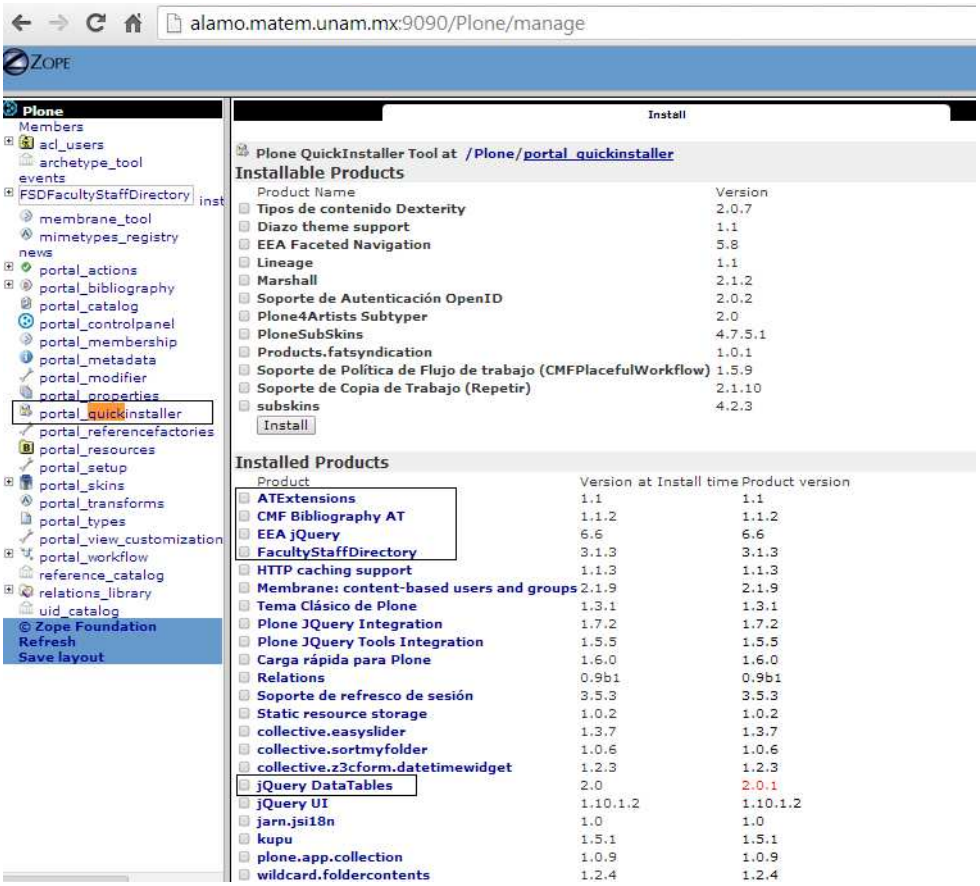

Figura A.1: Interfaz de instalación de productos en la ZMI. Esta interfaz es otra opción para instalar productos a la plataforma. Se presentan los productos activos e inactivos de la plataforma. Estos productos son definidos en el archivo *buildout.cfg* y descargados en etapas anteriores. Los productos encerrados en un recuadro son los más importantes a activar en nuestro sistema.

9. Finalmente podemos acceder a una nueva página de bienvenida del sitio y empezar a añadir recursos y configurar el aspecto del sitio.

Para detalles de configuración del archivo *buildout.cfg* consulte:
http://developer.plone.org/reference_manuals/old/buildout/buildoutcfg.html
y para agregar productos consulte:
https://plone-spanish-docs.readthedocs.org/es/latest/plone/instalar_productos/index.html

A.2. Algunos detalles de integración

CMFBibliographyAT

En la integración a la plataforma Plone del componente *CMFBibliographyAT* se han encontrado algunas fallas de integración. Sin embargo se ha podido solucionar cambiando la vista activada por defecto.

```
  Module zope.tales.expressions, line 217, in __call__
  Module Products.PageTemplates.Expressions, line 147, in _eval
  Module zope.tales.expressions, line 124, in _eval
  Module Products.PageTemplates.Expressions, line 74, in boboAwareZopeTraverse
  Module OFS.Traversable, line 317, in restrictedTraverse
  Module OFS.Traversable, line 285, in unrestrictedTraverse
   - __traceback_info__: ([], 'document_relateditems')
AttributeError: document_relateditems
```

Figura A.2: Interfaz del error.

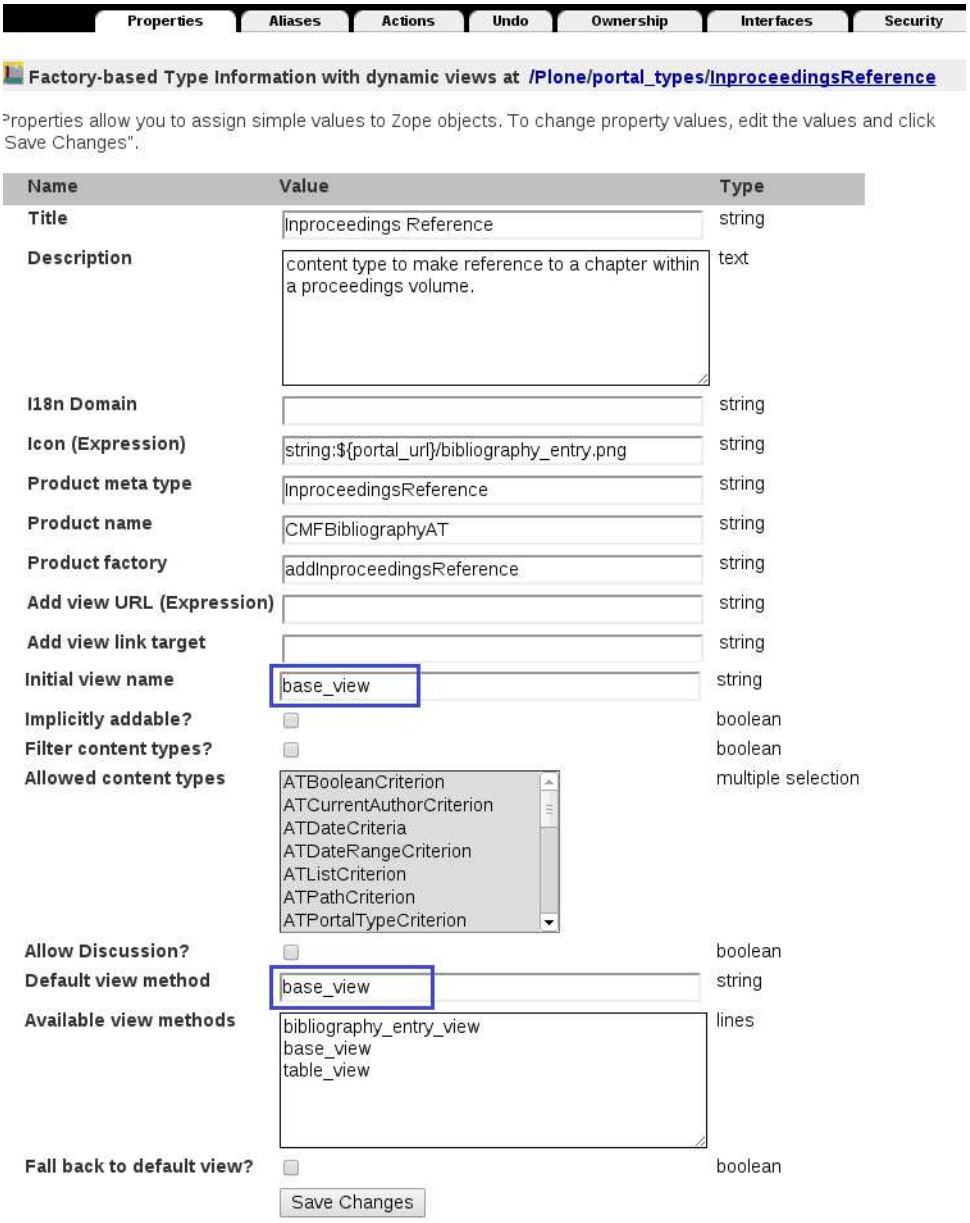

Figura A.3: Configuración de los tipos de publicaciones disponibles de *CMFBibliographyAT*. Cambio de la vista default *bibliography_entry_view* a *base_view* en el apartado de *portal_types* del sitio a través de la ZMI.

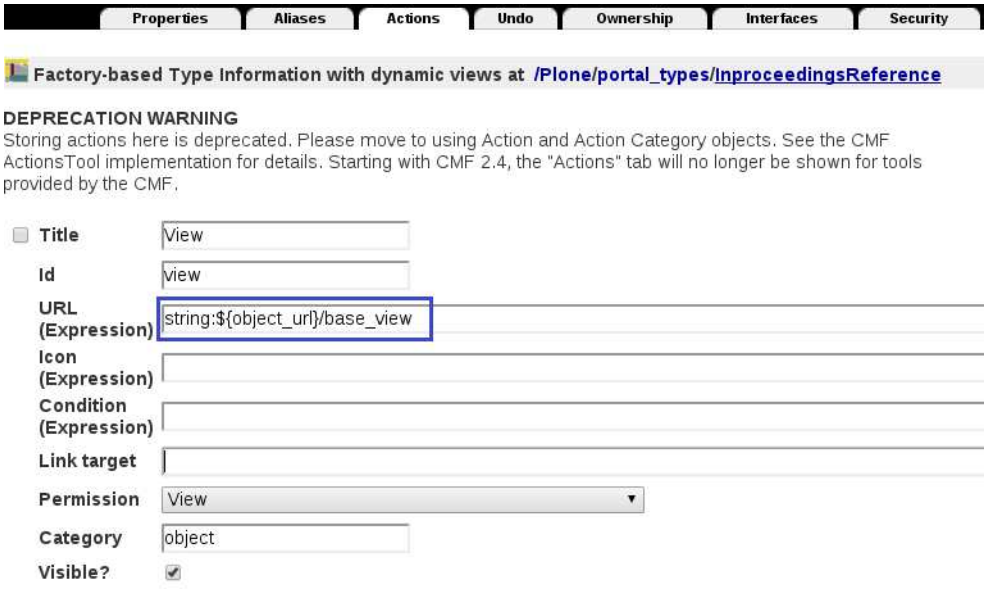

Figura A.4: Cambio de la vista default *bibliography_entry_view* a *base_view* a través de la ZMI.

A.3. Configuración Tor

En cuanto a la instalación y configuración de la herramienta *Tor*, esto se realiza de la siguiente manera:

Instalación

Sobre una interfaz de línea de comandos Linux descargar e instalar los paquetes:

```
wget https://jamielinux.com/pub/jamielinux-tor-release.noarch.rpm
yum install ./jamielinux-tor-release.noarch.rpm (en caso de usar fedora)
yum install tor privoxy (apt-get install tor privoxy)
systemctl enable tor.service
systemctl start tor.service
systemctl enable privoxy.service
systemctl start privoxy.service
```

Configuración

Del archivo de configuración *torrc* (ubicado en `/path/to/tor/`), descomentar

las líneas siguientes:

```
ControlPort 9051
```

y

```
HashedControlPassword 16:
```

Guardar los cambios.

Cambiar contraseña sobre línea de comandos:

```
tor -hash-password mypassword
```

Reiniciar *Tor*:

```
tor -f /path/to/tor/torrc
```

Finalmente podemos cambiar la IP mediante:

```
printf ‘‘AUTHENTICATE \‘‘mypassword\’’\r\nSIGNAL NEWNYM\r\n’’ | nc 127.0.0.1
9051
```

Para mayor información acerca de los parámetros y configuración más precisa, consulte https://www.torproject.org/.

A.4. Scripts: Obtención y tratamiento de la información

En el proceso de obtener la información bibliográfica desde el portal del *Google Scholar* se han generado *scripts* Python y Bash que ayuden en esta labor.

Estos *scripts* son los siguientes:

- Python
  - script01.py
    Encargado de generar los links necesarios a consultar para obtener la información deseada.
  - script03.py
    Encargado de obtener los archivos BibTex a partir de los archivos generados por el script *script01.py*.
  - script05.py

Encargado de eliminar la información incompleta y/o repetida. Se encarga también de verificar si las publicaciones que poseen llaves iguales son realmente equivalentes.

- script0x.py
  Encargado de asociar la información de las relaciones entre elementos bibliográficos.
- script0y.py
  Encargado de verificar la asociación de las citas con sus respectivas publicaciones.
- script0z.py
  Encargado de eliminar las duplicidades de artículos basándose en la verificación de llaves. Esto es posible debido a que en los análisis anteriores se verifica la equivalencia entre publicaciones.

- Bash
  - principal.sh
    Script encargado de ejecutar los *scripts script01.py* y *script03.py* para obtener la información bibliográfica de usuarios definidos en un archivo de entrada (*usersfile.txt* descrito más adelante).
  - duplicates-url.sh
    Script que detecta URLs duplicadas.
  - duplicates-key.sh
    Script que detecta llaves duplicadas.
  - information.sh
    Script encargado de ejecutar los *scripts* para el tratamiento de información, ya sea duplicados de llaves o información incompleta.
    Este último script genera los archivos finales a importar en el SACB.

De esta manera los pasos a seguir para obtener las publicaciones deseadas son:

- Generar el archivo de los usuarios a obtener información cuyo nombre será *usersfile.txt*

  El formato a seguir es el siguiente:
  ```
  nombreusuario01###URL-del-perfil-del-usuario01-en-GoogleScholar
  nombreusuario02###URL-del-perfil-del-usuario02-en-GoogleScholar
  ```

- Levantar el servicio *Tor*
  ```
  service tor star
  ```

- Inicializar `Tor`
  ```
  tor -f /path/to/tor/torrc
  ```

- Ejecutar el script *principal.sh*
  `sh principal.sh`
  Este script crea carpetas con los nombres de los usuarios definidos en el archivo *usersfile.txt*
  Dentro de cada carpeta se encuentran archivos auxiliares para obtener los archivos BibTex. El archivo con extensión `-pub.bib` hace referencia a las publicaciones del usuario definido y el archivo con extensión `-citations.bib` a las citas de las publicaciones en el archivo con extensión `-pub.bib`.

- El siguiente paso es ejecutar el script *information.sh*
  `sh information.sh`
  Que ejecuta los *scripts* dedicados al tratamiento y asociación de la información. Una vez que se hayan tratado los archivos Bibtex, los archivos finales serán los que tengan la extensión `-pub-url-key-abs-sindup.bib` para las publicaciones del usuario y `-citations-url-key-abs-match-sindup.bib` para las citas de las publicaciones del usuario.

  De esta manera estos dos archivos se importarán al sistema en una carpeta diferente, una con nombre *publicaciones* para el archivo con extensión `-pub-url-key-abs-sindup.bib`, otra con el nombre de *citas* para el archivo con extensión `-citations-url-key-abs-match-sindup.bib` y otra carpeta con el nombre *referencias* por lo pronto vacía.

  También se guarda el archivo que define las firmas de los investigadores del IM (una firma por linea y sin comas), en la misma ruta donde se encuentran las carpetas. Este archivo es guardado dentro del sistema con el nombre de *investigadores*.

  Con ello se lográ generar la interfaz de consulta de estas publicaciones. Es decir la vista que se aplica a la ruta donde están guardados estos recursos. La vista lleva el nombre de *faceted_bib*.

Las fuentes de estos *scripts* quedan a disposición del *Instituto de Matemáticas* para contemplar futuros cambios o como base para generar otros *scripts* que involucren otros portales.

Como nota adicional se observa que el progreso de obtención de información puede detenerse en algún momento por lo que se recomienda observar continuamente el progreso y en caso de haber demasiada demora se puede optar por reiniciar el servicio *Tor* (`tor -f /path/to/tor/torrc`) sin interrumpir el script *principal.sh*, con lo cual se reactivará el progreso en el punto donde se haya detenido.

Apéndice B

Acrónimos

API	Application Programming Interface
CBSE	Component-Based Software Engineering
CMF	Content Management Framework
CMS	Content Management System
CSS	Cascading Style Sheets
CSV	Comma-Separated Values
DBLP	Digital Bibliography & Library Project
DL	Digital Library
DT	Dinamic Taxonomies
EEA	European Environment Agency
GPL	General Public License
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IM	Instituto de Matemáticas
IP	Internet Protocol
ISSN	International Standard Serial Number
JIF	Journal Impact Factor
MVCC	Multiversion Concurrency Control
OOPL	Object-Oriented Programming Languages
PSF	Python Software Foundation
RIS	Research Information Systems
SACB	Sistema Administrador de Citas Bibliográficas
SEO	Search Engine Optimization
TAL	Template Attribute Language
URL	Uniform Resource Locator
UTF-8	8-bit Unicode Transformation Format
XML	Extensible Markup Language
ZCML	Zope Configuration Mark-up Language
ZEO	Zope Enterprise Objects
ZODB	Zope Objects Database
ZPT	Zope Page Template

Bibliografía

[1] Alonso, G., Casati, F., Kuno, H., and Machiraju, V. *Web Services: Concepts, Architectures and Applications*, 1st ed. Springer Publishing Company, Incorporated, 2010.

[2] Bidgoli, H. *The Internet Encyclopedia, G O*. The Internet Encyclopedia. John Wiley & Sons, 2004.

[3] Boiko, B. *Content Management Bible*. John Wiley & Sons, Inc., New York, NY, USA, 2004.

[4] Cambridge University Press. Cambridge dictionaries online. http://dictionary.cambridge.org/dictionary/british/, 2013.

[5] Cervantes, I. Migración y nuevas características del sistema de votación electrónica del instituto de matemáticas de la UNAM. Master's thesis, UNAM, 2009.

[6] Clapp, L. Plonevotecryptolib: Una biblioteca criptográfica para la implementación de elecciones en línea secretas y verificables por electores., 2011.

[7] Clark, D. *Beginning Object-Oriented Programming with VB 2005: From Novice to Professional*. Beginning: From Novice to Professional. Apress, 2005.

[8] Connor, J. Google scholar citations open to all. http://googlescholar.blogspot.mx/2011/11/google-scholar-citations-open-to-all.html, Noviembre 2011. The official source for information about Google Scholar.

[9] Curiel, A. Diseño e implementación en plone de un sistema de manejo de solicitudes mediante flujos de trabajo. Master's thesis, UNAM, 2010.

[10] Elmasri, R., and Navathe, S. *Fundamentals of Database Systems*, 6th ed. Addison-Wesley Publishing Company, USA, 2010.

[11] Espinosa, E. Desarrollo de un sistema de administración de procesos en plone. Master's thesis, UNAM, 2009.

[12] FECYT. Propuesta de manual de ayuda a los investigadores españoles para la normalización del nombre de autores e instituciones en las publicaciones científicas. https://www.accesowok.fecyt.es/wp-content/uploads/2009/06/normalizacion_nombre_autor.pdf, 01 2007.

[13] Feder, A. Bibtex.org. http://www.bibtex.org/, 2006.

[14] Fitzgerald, D. Managing references the easy way. http://www.the-scientist.com/?articles.view/articleNo/13769/title/Managing-References-the-Easy-Way/, 2002.

[15] Franceschini, F., and Maisano, D. Bibliometric positioning of scientific manufacturing journals: a comparative analysis. *Scientometrics 86*, 2 (Febrero 2011), 463–485.

[16] George-Nascimento, M. Una evaluación de los índices bibliométricos I e Is de Molina-Montenegro & Gianoli aplicada a investigadores en ciencias ecológicas en Chile. *Revista chilena de historia natural 83* (Junio 2010), 229–235.

[17] Hearst, M. A. Uis for faceted navigation: Recent advances and remaining open problems. In *HCIR 2008: Proceedings of the Second Workshop on Human-Computer Interaction and Information Retrieval* (2008), Citeseer, pp. 13–17.

[18] Heineman, G. T., and Councill, W. T., Eds. *Component-based software engineering: putting the pieces together*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001.

[19] IraLIS team. International registry of authors-links to identify scientists. http://www.iralis.org/.

[20] Jeng, J. What is usability in the context of the digital library and how can it be measured? *Information Technology and Libraries 24*, 2 (2005), 46–56.

[21] Kuchling, A. Zodb/zeo programming guide. http://www.zodb.org/, Enero 2006.

[22] Kulkarni, A., Aziz, B., Shams, I., and Busse, J. Comparisons of citations in web of science, scopus, and google scholar for articles published in general medical journals. *JAMA 302*, 10 (2009), 1092–1096.

[23] Kulkarni, A. V., Busse, J. W., and Shams, I. Characteristics associated with citation rate of the medical literature. *PLoS ONE 2*, 5 (Mayo 2007), e403.

[24] Lehman, P., Kime, P., Boruvka, A., and Wright, J. *The biblatex Package*, Octubre 2013.

[25] López Rabadán, M. A. Sistema sobre plone para la captura y recolección de información curricular del instituto de matemáticas. Master's thesis, UNAM, 2007.

[26] Mazloumian, A. Predicting scholars'scientific impact. *PLoS ONE 7*, 11 (Noviembre 2012), e49246.

[27] Méndez, D. Intercambio de información y web semántico., 2010.

[28] MIT and Contributors. The exhibit web site. http://www.simile-widgets.org/exhibit/, 2012.

[29] NetworkX developer team. Networkx. high-productivity software for complex networks. http://networkx.github.io/, 2013.

[30] Plone Foundation. Pagina de plone. http://www.plone.org, 2012.

[31] Quinn, L., and Gardner-Madras, H. Comparing open source content management systems: Wordpress, joomla, drupal and plone. http://idealware.org/reports/2010-os-cms, Diciembre 2010.

[32] Ramnath, S., and Dathan, B. *Object-Oriented Analysis and Design*, 1st ed. Springer Publishing Company, Incorporated, 2011.

[33] Redomino, and McKay, A. *The Definitive Guide to Plone, Second Edition*, 2nd ed. Apress, Berkely, CA, USA, 2009.

[34] Richardson, L. Beautiful soup. http://www.crummy.com/software/BeautifulSoup/, Noviembre 2013.

[35] Rodríguez, H. Intercambio de información entre instituciones de la UNAM. usando la red semántica, 2010.

[36] Sacco, G. M., and Tzitzikas, Y. *Dynamic Taxonomies and Faceted Search: Theory, Practice, and Experience*, 1st ed. Springer Publishing Company, Incorporated, 2009.

[37] Sanchez P., F. Citation analysis. En preparación, Octubre 2012.

[38] Schuldt, H. Application server. In *Encyclopedia of Database Systems*, L. LIU and M. T. ÖZSU, Eds. Springer US, 2009, pp. 104–104.

[39] Silberschatz, A., Korth, H., and Sudarshan, S. *Database Systems Concepts*, 5 ed. McGraw-Hill, Inc., New York, NY, USA, 2006.

[40] Sommerville, I. *Ingeniería del software.* Pearson Educación, 2005.

[41] The ORCID team. Open Researcher and Contributor ID. http://orcid.org/.

[42] Thomson Reuters Corporation. Researcher ID. http://www.researcherid.com/.

[43] Tor Project. Tor. https://www.torproject.org/, 2013.

[44] von Weitershausen, P. *Web Component Development with Zope 3*, 3rd ed. Springer Publishing Company, Incorporated, 2008.

[45] Zapata, A. Implementación de un sistema de votación electrónica como un producto sobre la plataforma plone. Master's thesis, UNAM, 2008.

[46] Zope Developers Community. The Zope 2 Book. http://docs.zope.org/zope2/zope2book/, 2010.

[47] Zope Foundation. zope.org. http://www.zope.org/, Noviembre 2011.