Universidad Autónoma Metropolitana Unidad Azcapotzalco
División de Ciencias Básicas e Ingeniería
Licenciatura en Ingeniería en Computación

Aplicación móvil para la gestión de noticias en español con traducción al náhuatl

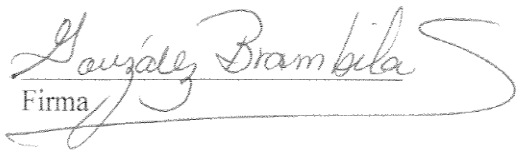
Modalidad: Proyecto tecnológico
Trimestre: 2016-Invierno

Alumno
Javier Solís Camilo
Matrícula: 210329945
al210329945@alumnos.azc.uam.mx

Asesora:
Silvia Beatriz González Brambila
Profesora Titular
Departamento de Sistemas
sgb@correo.azc.uam.mx

Co asesor:
José Alejandro Reyes Ortiz
Profesor Titular
Departamento de Sistemas
jaro@correo.azc.uam.mx

Yo, Silvia Beatriz González Brambila, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

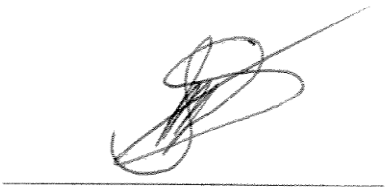


Firma

Yo, José Alejandro Reyes Ortiz, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Firma

Yo, Javier Solís Camilo, doy mi autorización a la Coordinación de Servicios de Información de la Universidad Autónoma Metropolitana, Unidad Azcapotzalco, para publicar el presente documento en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.



Firma

Resumen

La comunicación de los hechos de la actualidad siempre ha sido relevante para las personas, en un mundo donde es necesario estar al tanto de la información en todo momento. Es por eso que se desarrolló una aplicación informática que permite visualizar noticias en español y además su traducción al náhuatl.
Dicha aplicación tiene como objetivo principal hacer del conocimiento de la gente información relevante, así mismo, cumple como mínimo dos metas más: la primera consiste en posibilitar a las personas que sólo hablan náhuatl estar informados sobre los últimos acontecimientos que ocurren en nuestro país, la segunda consiste en otorgarle a aquellas personas que sólo hablan español conocer y además aprender acerca del náhuatl.

En la fase de diseño se lograron detectar los siguientes componentes: un diccionario, corpus paralelo donde se encuentran alineados palabras y frases español - náhuatl, un recolector de noticias encargadas de obtener información en la web, un método de traducción, un proveedor de servicios web y una aplicación móvil. Lo siguiente es el proceso de ejecución en el sistema cuando una persona desea consultar noticias de una fuente seleccionada, la aplicación móvil consume el servicio web con la intención de obtener noticias, el recolector obtiene las noticias de la fuente para posteriormente con el método de traducción por corpus paralelo realizar la transcripción al náhuatl, finalizando con el proveedor de servicios que es el encargado de regresar a la aplicación movil la colección de sucesos traducidos.

Se utilizó un corpus paralelo español - náhuatl de trece mil palabras y frases aproximadamente, luego se programó el recolector de noticias usando las bibliotecas del lenguaje de programación escogido para este proyecto, se codificó el algoritmo encargada de traducir la noticia usando el diccionario, se puso en marcha el servicio web para consumirlo en la aplicación móvil, se diseñó la aplicación móvil considerando los siguientes casos de uso: registrar usuario, inicio de sesión, seleccionar categoría, seleccionar fuente, seleccionar noticia, mostrar noticia y calificación de noticias para finalmente hacerlo funcionar en conjunto.

La traducción automática realizada en este proyecto basada en un corpus paralelo resulto medianamente satisfactorio debido a que no todas las palabras o frases fueron traducidas al náhuatl porque se necesita un corpus más extenso.

Los servicios web es una buena opción para proveer soluciones a varias plataformas utilizando el internet. La aplicación móvil cumplió la función de mostrar los resultados a las personas interesadas en la noticias español – náhuatl.

Índice general

Contenido

Índice de Figuras

1. Introducción

En México, actualmente, hay un 6% de población hablante de una lengua originaria, la lengua Náhuatl cuenta con el mayor porcentaje de hablantes [1] , al estar el español como lengua oficial hablado por el resto de los mexicanos, hay una necesidad de comunicación entre las personas que solo hablan el español o el náhuatl. Además, con el nacimiento de la web, la información generada de manera electrónica en México, se encuentra en su mayoría en español, esto ocasiona que las personas que hablan náhuatl no tengan un acceso oportuno a la información de noticias, tales como eventos relevantes, desastres, didáctica, entre otros.

Estas personas necesitan de un traductor para tener acceso a la información generada en periódicos nacionales, además de que sufren un desfase de tiempo en cuestión de actualización de las mismas.

Este proyecto trata de aportar una solución a esta problemática. Para ello, se desarrolló una herramienta computacional que realiza traducciones de noticias en español al náhuatl basada en la traducción por frases.

Como entrada para la herramienta, se tienen noticias extraídos de canales RSS[1] en español utilizando HTTP[2]. Posteriormente, se hace uso de un corpus paralelo[3] (diccionario),es decir, almacenes compuestos de textos en español alineados con su traducción en náhuatl, para realizar la transcripción de la noticia recibida buscando equivalencias de frases, finalmente usando REST[4] regresar el resultado al cliente, en este caso la aplicación móvil en Android[5], en formato JSON[6].

En la siguiente sección se mencionan los proyectos terminales, tesis y software relacionados con el proyecto desarrollado. En la tercera sección se expresa la justificación del proyecto. En la cuarta sección se mencionan los objetivos específicos que se alcanzaron. En la quinta sección se exponen las bases teóricas de los diferentes

[1] **RSS** (Really Simple Syndication), es un formato de marcas extensible para compartir contenido en la web.
[2] **HTTP** (Hypertext Transfer Protocol), es un protocolo de transferencia de información en la web.
[3] C**orpus paralelo**, es un recurso lingüístico consistente en textos de dos idiomas que están alineados a cierto nivel de granularidad; generalmente a nivel de párrafo, aunque también a nivel de sección, página o incluso a veces de palabra.
[4] **RESP** (Representational State Transfer), es un tipo de arquitectura de desarrollo web que se apoya en HTTP para proveer servicios.
[5] **Android,** es un sistema operativo diseñado para dispositivos móviles.
[6] **JSON** (JavaScript Object Notation), es un formato ligero de intercambio de datos.

conceptos y tecnologías usadas para este trabajo. En la sexta sección se describe el diseño, desarrollo e implementación de la aplicación móvil. En la séptima sección se muestran los resultados de las pruebas realizadas por cada caso de uso. En la octava sección se realiza un análisis y discusión de resultados. En la novena sección se exponen las conclusiones.

2. Antecedentes

2.1. Proyectos terminales

Sistema gestor de información para la difusión de noticias de la UAM Azcapotzalco utilizando un canal RSS[2].

La aplicación gestiona las noticias de la UAM Azcapotzalco y las difunde por un canal RSS llegando a un interfaz de usuario. Es un sistema para un dispositivo móvil. Este proyecto terminal tiene como objetivo difundir las noticias por un canal RSS, sin embargo, el trabajo que se propone pretende presentar las noticias en español y en náhuatl.

Aplicación Android para la práctica de verbos compuestos del idioma inglés [3].

Esta herramienta computacional permita hacer ejercicios en la formación de verbos compuesto en inglés, está dirigido a la plataforma móvil con sistema operativo Android. La relación que tiene éste proyecto con el propuesto es que ambos sistemas trabajan con un idioma, pero a diferencia del proyecto de integración, éste va hacer la traducción de una noticia en español a náhuatl.

SRCV: Sistema computacional interactivo basado en reconocimiento de comandos de voz para aplicaciones educativas [4].

Es una aplicación capaz de hacer un reconocimiento de voz y transformar un mensaje de audio a texto, en la cuál a diferencia del propuesto, éste va tener como entrada un canal en formato de marcas por donde se va recibir el texto (noticia en español), para posteriormente traducirlo al náhuatl.

2.2. Tesis

Vitalidad y desplazamiento en el náhuatl de Rafael Delgado, Veracruz [5].

Es una tesis que expresa la situación actual de la lengua náhuatl en un municipio llamado Rafael delgado del estado de Veracruz, la lengua está sufriendo alteraciones y está siendo desplazada. Esto se debe a varios aspectos; el contacto lingüístico del español, la influencia asimétrica, el servicio educativo en el pueblo y el fenómeno bilingüismo.

2.3. Software

Navegador “Mozilla Firefox” en Náhuatl [6]

Es un explorador web abierto, traducida a la lengua náhuatl. La única relación que existe entre este software y el proyecto es que ambos van enfocados para la población con la lengua originaria.

3. Justificación

En México, la información electrónica de noticias generada diariamente, en su mayoría se encuentra en el idioma español. Por ello, una herramienta que proporcione una traducción de una noticia del español al náhuatl ayuda a las personas que sólo hablan la lengua nativa para la comprensión de la información generada diariamente en México. Además, para los usuarios que hablan español y deseen aprender un poco de náhuatl, esta herramienta puede servir de plataforma para dicha actividad.

Este proyecto de integración aporta una herramienta de software implementada en Android. Además, se utilizó diccionarios de datos (base de datos), con ello se garantiza que este proyecto está fuertemente ligada a la computación.

4. Objetivos

Diseñar e implementar un sistema para la gestión de noticias traducidas del español al náhuatl obtenidas por medio de un canal RSS.

4.1. Objetivos específicos

- Diseñar e implementar una base de datos para la gestión de noticias basada en su categoría y su fuente, esto se logró gracias a que al momento de dar de alta una fuente RSS al sistema se tiene que ingresar o seleccionar la categoría a la que pertenece por lo que las noticias de la fuente pertenecen también a esa categoría y de esta forma permitirle al usuario llegar a la noticia seleccionando una categoría o fuente.
- Implementar un módulo encargado de gestionar usuarios y noticias a partir de un perfil deseado, en el registro de usuario existe la posibilidad de seleccionar categorías y fuentes RSS como preferidas de tal forma que se vuelven parte del perfil de usuario para posteriormente en la aplicación el usuario tener la oportunidad de consultar noticias a partir de esta información.
- Diseñar e implementar un método de traducción de noticias del español al náhuatl basado en un corpus paralelo, esto se logró gracias a un algoritmo que inicia con un texto en español, en el mejor de los casos todo el texto se traduce y en un caso promedio algunas frases o palabras que no se encuentran en el corpus quedan igual.
- Evaluar el método de traducción de noticias mediante la medida de precisión, esta evaluación se realiza con una estadística de palabras no encontradas en el corpus paralelo con la finalidad de agregarlo en el corpus.

5. Marco teórico

En esta sección se describen conceptos, tecnologías y teorías utilizadas para el desarrollo del proyecto. Android se utilizó para desarrollar una aplicación móvil con el fin de presentar las noticias y su traducción al usuario final, REST para ofrecer un REST API en el servidor con la finalidad de consumirlo en Android, RSS es el

formato de las noticias recolectadas en la web, traducción automática y corpus paralelo es la teoría utilizado para realizar la traducción automática.

5.1. ANDROID

Android es un sistema operativo fácil de usar que se puede personalizar. Se utiliza en más de mil millones de dispositivos de todo el mundo, desde teléfonos y tabletas hasta relojes, televisiones y automóviles [7], algunos dispositivos soportados se muestran en la Figura 1.

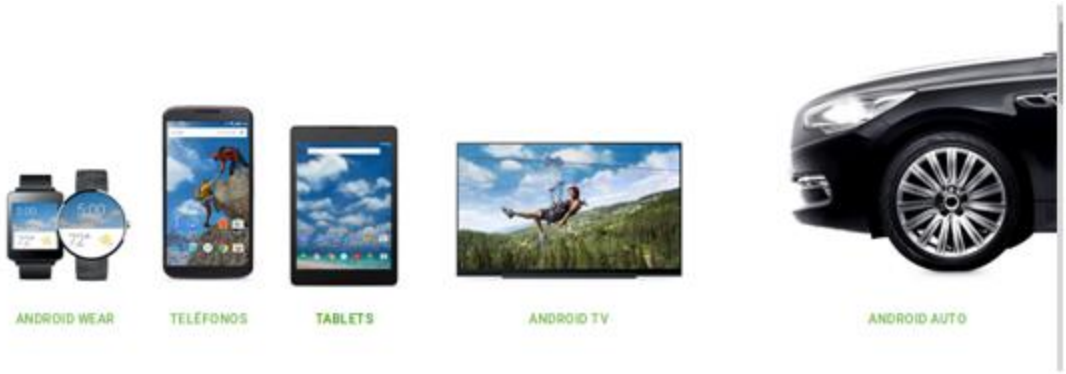


Figura 1: Dispositivos Android. Disponible en https://www.android.com/intl/es-419_mx/

Lo que lo hace diferente es que está basado en Linux, **un núcleo de sistema operativo libre, gratuito y multiplataforma**.

Este sistema operativo permite programar aplicaciones en una variación de Java llamada Dalvik [8], que proporciona todas las interfaces necesarias para desarrollar aplicaciones que accedan a las funciones de un teléfono (como el GPS, las llamadas, la agenda, etc.) de una forma muy sencilla.

5.1.1. Arquitectura Android

La arquitectura de Android está conformada por capas. Cada una de estas capas utiliza servicios ofrecidos por las anteriores, y ofrece a su vez los suyos propios a las capas de niveles superiores [9], tal como muestra en la Figura 2 ((c) Google).

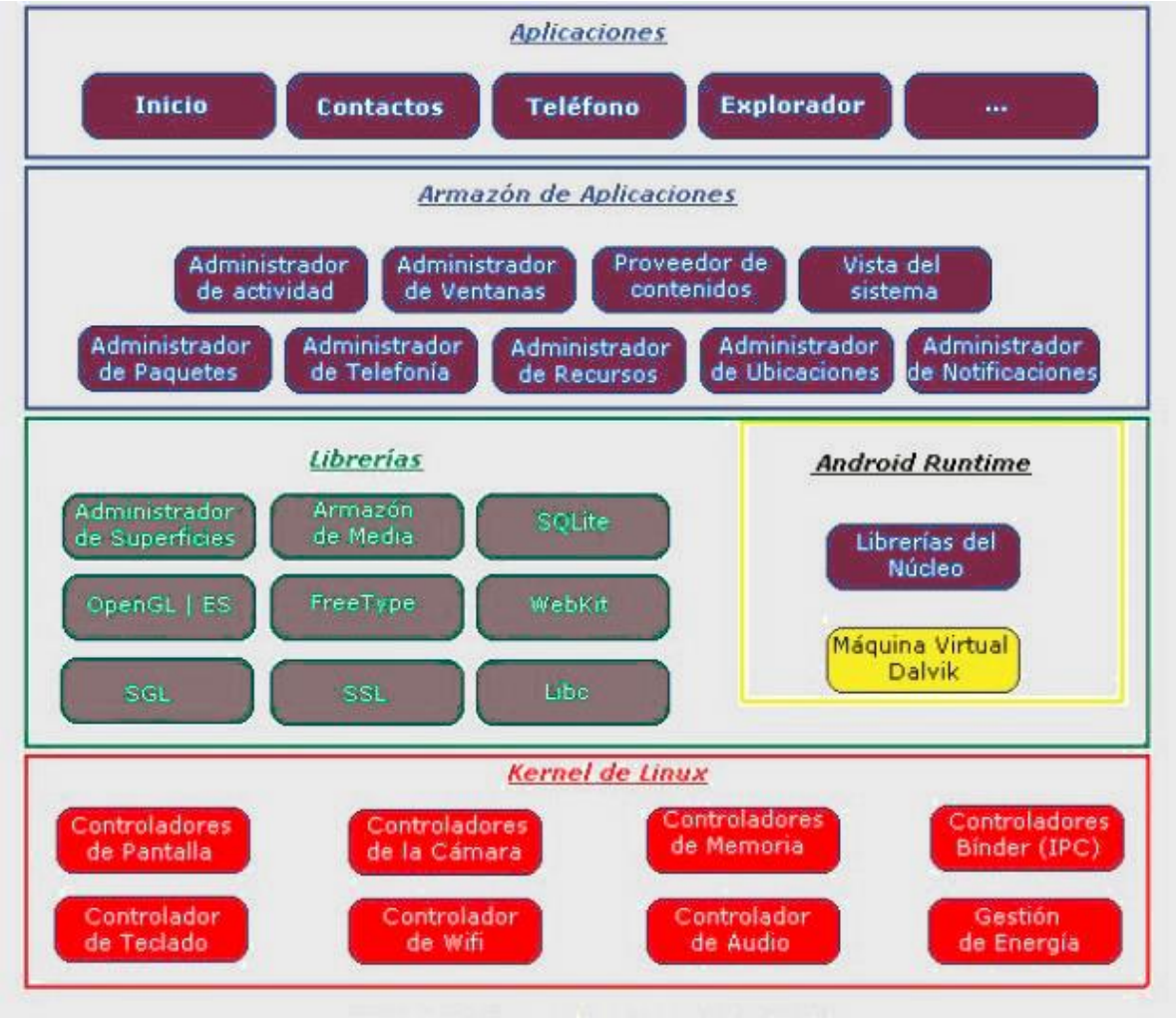


Figura 2: Arquitectura de ANDROID. Disponible en https://sites.google.com/site/swcuc3m/home/android/generalidades/2-2-arquitectura-de-android

Aplicaciones

Este nivel contiene, tanto las incluidas por defecto de Android como aquellas que el usuario vaya añadiendo posteriormente, ya sean de terceras empresas o de su propio desarrollo. Todas estas aplicaciones utilizan los servicios, las API y librerías de los niveles anteriores.

Armazón de Aplicaciones

Representa fundamentalmente el conjunto de herramientas de desarrollo de cualquier aplicación. Toda aplicación que se desarrolle para Android, ya sean las propias del dispositivo, las desarrolladas por Google o terceras compañías, o incluso las que el propio usuario cree, utilizan el mismo conjunto de API y el mismo armazón, representado por este nivel.

Entre las API más importantes ubicadas aquí, se pueden encontrar las siguientes:

- Administrador de actividad: Conjunto de API que gestiona el ciclo de vida de las aplicaciones en Android.

- Administrador de ventanas: Gestiona las ventanas de las aplicaciones y utiliza la librería del administrador de superficies.
- Administrador de telefonía: Incluye todas las API vinculadas a las funcionalidades propias del teléfono (llamadas, mensajes, etc.).
- Proveedor de contenidos: Permite a cualquier aplicación compartir sus datos con las demás aplicaciones de Android. Por ejemplo, gracias a esta API la información de contactos, agenda, mensajes, etc. será accesible para otras aplicaciones.
- Vista del sistema: Proporciona un gran número de elementos para poder construir interfaces de usuario (GUI), como listas, mosaicos, botones, "*check-box*", tamaño de ventanas, control de las interfaces mediante teclado, etc. Incluye también algunas vistas estándar para las funcionalidades más frecuentes.
- Administrador de ubicaciones: Posibilita a las aplicaciones la obtención de información de localización y posicionamiento.
- Administrador de notificaciones: Mediante el cual las aplicaciones, usando un mismo formato, comunican al usuario eventos que ocurran durante su ejecución: una llamada entrante, un mensaje recibido, conexión *Wi-Fi* disponible, ubicación en un punto determinado, etc. Si llevan asociada alguna acción, en Android denominada *Intent*, (por ejemplo, atender una llamada recibida) ésta se activa mediante un simple clic.

Librerías

La siguiente capa se corresponde con las librerías utilizadas por Android. Éstas han sido escritas utilizando C/C++ y proporcionan al sistemas operativo la mayor parte de sus capacidades más características. Junto al núcleo basado en Linux, estas librerías constituyen el corazón de Android.

Entre las librerías más importantes ubicadas aquí, se pueden encontrar las siguientes:

- Libc: Incluye todas las cabeceras y funciones según el estándar del lenguaje C. Todas las demás librerías se definen en este lenguaje.
- Administrador de superficies: Es la encargada de componer los diferentes elementos de navegación de pantalla. Gestiona también
- las ventanas pertenecientes a las distintas aplicaciones activas en cada momento.
- OpenGL y SGL: Representan las librerías gráficas y, por tanto, sustentan la capacidad gráfica de Android. OpenGL/SL maneja gráficos en 3D y permite utilizar, en caso de que esté disponible en el propio dispositivo móvil, el hardware encargado de proporcionar gráficos 3D. Por otro lado, SGL proporciona gráficos en 2D, por lo que será la librería más habitualmente utilizada por la mayoría de las aplicaciones. Una característica importante de la capacidad gráfica de Android es que es posible desarrollar aplicaciones que combinen gráficos en 3D y 2D.

- Armazón de media: Proporciona todos los códecs necesarios para el contenido multimedia soportado en Android (vídeo, audio, imágenes estáticas y animadas, etc.)
- FreeType: Permite trabajar de forma rápida y sencilla con distintos tipos de fuentes.
- Librería SSL: Posibilita la utilización de dicho protocolo para establecer comunicaciones seguras.
- Librería SQLite: Creación y gestión de bases de datos relacionales.
- Librería WebKit: Proporciona un motor para las aplicaciones de tipo navegador y forma el núcleo del actual navegador incluido por defecto en Android.

Android Runtime

Al mismo nivel que las librerías de Android se sitúa el entorno de ejecución. Éste lo constituyen las Librearías del núcleo , que son librerías con multitud de clases Java y la máquina virtual Dalvik.

Kernel de Linux

Android utiliza el núcleo de Linux 2.6 como una capa de abstracción para el hardware disponible en los dispositivos móviles. Esta capa contiene los drivers necesarios para que cualquier componente hardware pueda ser utilizado mediante las llamadas correspondientes. Siempre que un fabricante incluye un nuevo elemento de hardware, lo primero que se debe realizar para que pueda ser utilizado desde Android es crear las librerías de control o drivers necesarios dentro de este kernel de Linux embebido en el propio Android.

Esta sencillez, junto a la existencia de herramientas de programación gratuitas, hace que una de las cosas más importantes Android sea la cantidad de aplicaciones disponibles, que extienden casi sin límites la experiencia del usuario.

5.1.2. Historia del Sistema Operativo Android

Fue desarrollado por Android Inc. empresa que en 2005 fue comprada por Google, aunque no fue hasta 2008 cuando se popularizó, gracias a la unión al proyecto de *Open Handset Alliance*, un consorcio formado por 48 empresas de desarrollo hardware, software y telecomunicaciones, que decidieron promocionar el software libre. Pero ha sido Google quien ha publicado la mayor parte del código fuente del sistema operativo, gracias al software Apache, que es una fundación que da soporte a proyectos software de código abierto.

5.1.3. Libertad del sistema operativo Android

Una de las mejores características de este sistema operativo es que es completamente libre. Es decir, ni para programar en este sistema ni para incluirlo en un teléfono hay que pagar nada. Y esto lo hace muy popular entre fabricantes y desarrolladores, ya que los costos para lanzar un teléfono o una aplicación son muy bajos**.**

Cualquiera puede bajar el código fuente, inspeccionarlo, compilarlo e incluso cambiarlo. Esto da una seguridad a los usuarios, ya que algo que es abierto permite detectar fallos más rápidamente. Y también a los fabricantes, pues pueden adaptar mejor el sistema operativo a los terminales.

5.1.4. Multitarea en Android

Es un sistema multitarea que permite ejecutar aplicaciones en segundo plano, así que se puede realizar dos o más actividades tareas al mismo tiempo, por ejemplo escuchar música mientras se navega por internet o mientras se leen los correos electrónicos, así que la productividad puede aumentar en gran medida.

5.2. Transferencia de Estado Representacional (por sus siglas en inglés - REST)

REST (*Representational State Transfer)*, es un tipo de arquitectura de desarrollo web que se apoya totalmente en el estándar HTTP.

Los servicios web que siguen este estilo deben cumplir con las siguientes premisas:

Sin estado: Son servicios web que no mantienen estado asociado al cliente. Cada petición que se realiza a ellos es completamente independiente de la siguiente. Todas las llamadas al mismo servicio serán idénticas, como se observa en la Figura 3.

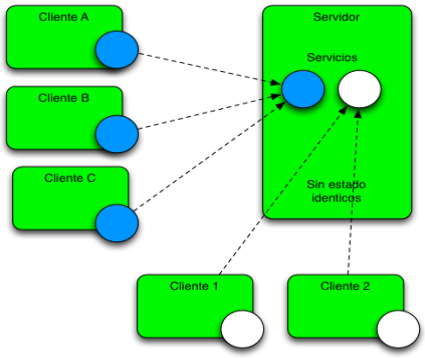


Figura 3: Propiedad sin estado, REST. Disponible en http://www.arquitecturajava.com/servicios-rest/

Cache: El contenido de los servicios web REST se puede cachear (ver Figura 4) de tal forma que una vez realizada la primera petición al servicio el resto puedan apoyarse en la cache si fuera necesario.

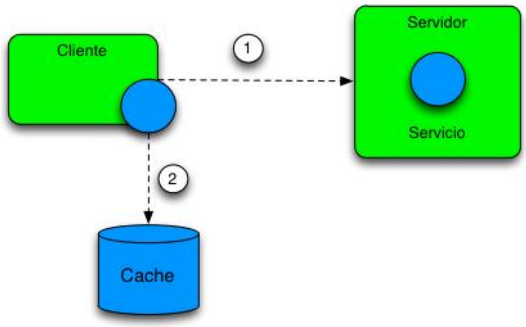


Figura 4:Propiedad cache, REST. Disponible en http://www.arquitecturajava.com/servicios-rest/

Servicios Uniformes: Todos los servicios REST compartirán una forma de invocación y métodos uniforme utilizando los métodos GET, POST, PUT, DELETE. Ver Figura 5.

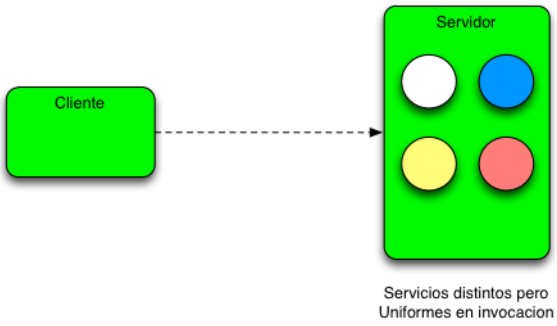


Figura 5: Propiedad servicios uniformes, REST. Disponible en http://www.arquitecturajava.com/servicios-rest/

Arquitectura en Capas: Todos los servicios REST están orientados hacia la escalabilidad y un cliente REST no será capaz de distinguir entre si está realizando una petición directamente al servidor, o se lo está devolviendo un sistema de caches intermedio o por ejemplo existe un balanceador que se encarga de redirigirlo a otro servidor. Figura 6.

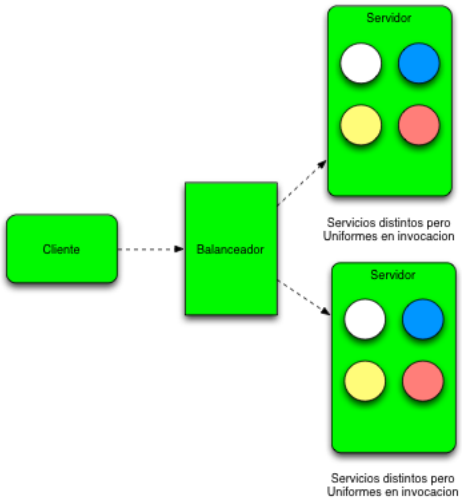


Figura 6: Propiedad arquitectura en capas, REST. Disponible en http://www.arquitecturajava.com/servicios-rest/

REST se definió en el 2000 por Roy Fielding, coautor principal también de la especificación HTTP. Se puede considerar a REST como un FRAMEWORK para construir aplicaciones web respetando HTTP.
Por lo tanto REST es el tipo de arquitectura más natural y estándar para crear APIs para servicios orientados a Internet.

5.2.1. SOAP Y REST

SOAP (Simple Object Access Protocol), es el protocolo que hace poco fue muy utilizado en los servicios web. Es un protocolo muy complejo pensado para dar soluciones a la mayoría de los problemas en las comunicaciones de computadoras como es la seguridad, transaccionalidad, mensajería y demás. Marcó una época gloriosa para los servicios web, las primeras implementaciones se llamaban WS1.0 y con el tiempo salieron especificaciones WS-* pensadas para dar soluciones cada vez más avanzadas por lo que también crecía

su complejidad y como consecuencia una tecnología con muchas capacidades que en la mayoría de los casos resultaban innecesarias.

Por su parte REST es simple. REST no pretende solucionar todo y por lo tanto no se paga con demasiada complejidad una potencia que quizá no se necesite.
Una diferencia fundamental entre un servicio web clásico (SOAP) y un servicio REST es que el primero está orientado a RPC (*Remote Procedure Call* – Llamada a método remoto), es decir, a invocar métodos sobre un servicio remoto, mientras que el segundo está orientado a recursos. Es decir, las operaciones son sobre recursos, no sobre servicios.

En una API REST la idea de "servicio" como tal desaparece. Lo que se tiene son recursos, accesibles por identificadores (URIs). Sobre esos recursos se puede realizar acciones, generalmente diferenciadas a través de verbos HTTP distintos (GET, POST, PUT, DELETE).

5.2.2. Ejemplo comparativo de SOAP y REST

En un servicio web clásico (SOAP) se tiene un servicio llamado *ServicioLibro* que tendría un método llamado *GetTodas* que devolvería todos los libros. La idea, independientemente de la tecnología usada para consumir el servicio web, es que se llama a un método (*GetTodas*) de un servicio remoto (*ServicioLibro*). Del mismo modo para obtener un libro en concreto se invocaría el método *GetPorId*() pasándole el id del libro. De ahí que se diga que están orientados a RPC. Por su parte en un servicio REST la propia idea de servicio se desvanece. En su lugar queda la idea de un "recurso", en este caso, "Colección de libros" que tiene una URI que lo identifica, p. ej. /Libros. Así, si se invoca dicha URI se debe de obtener una representación de dicho recurso, es decir, obtener el listado de todos los libros.
Para accesar a los datos de un libro, habrá otro recurso (libro) con una URI asociada. Además, entre los recursos hay relaciones: está claro que un libro forma parte de la "Colección de libros" así que parece lógico que a partir de la colección entera (/Libros) se pueda acceder a uno de sus elementos con una URI tipo /Libros/123, siendo "123" el ID del libro.
En el servicio SOAP: para dar de alta un libro seria llamando a un método *agregarLibro* de *ServicioLibro*, y enviando la información del libro a añadir. En cambio, en API REST añadir un libro consiste en usar la URI de dicho libro, (p. ej. /Libros/456 si se agrega el libro con ID 456) usando POST o PUT como verbo HTTP y colocando los datos del libro en el cuerpo de la petición. La URI para acceder al libro con ID 456 es siempre /Libros/456 y es el verbo HTTP (GET, POST, PUT, DELETE,...) el que indica cual es la operación que se desea hacer con ese libro.

5.3. Sindicación Realmente Simple (por sus siglas en inglés - RSS)

RSS (*Really Simple Syndication*), es un formato XML para sindicar o compartir contenido en la web. Es una forma muy sencilla para recibir información del internet, directamente en un dispositivo como es la computadora, Smartphone, Tablet o en una página web a través de un lector RSS información actualizada sobre las páginas web favoritas, sin necesidad de visitarlas una a una. Esta información se actualiza automáticamente, sin realizar una actividad extra. Para que esto suceda, se deben de cumplir dos requisitos, el primero es que los servicios RSS deben de estar disponibles en un servidor, segundo, el cliente debe de tener un lector RSS.

Para una persona, si existen varias páginas web de su interés con contenido que se actualiza cada cierto tiempo, un lector RSS le ahorrará mucho tiempo en esta tarea. Gracias al RSS, no habrá necesidad de visitar cada una de las páginas web para ver si han añadido o no algún artículo. Estas páginas informarán a través del lector de RSS automáticamente sobre todas las novedades que se han producido en la página en cuestión

5.4. Traducción automática

La traducción automática (TA), también llamada MT (del inglés *Machine Translation*), es un área de la lingüística computacional que investiga el uso de software para traducir texto o habla de un lenguaje natural a otro. En un nivel básico, la traducción por computadora realiza una sustitución simple de las palabras atómicas de un lenguaje natural por las de otro. Por medio del uso de corpora lingüísticos se pueden intentar traducciones más complejas, lo que permite un manejo más apropiado de las diferencias en la Tipología lingüística, el reconocimiento de frases, la traducción de expresiones idiomáticas y el aislamiento de anomalías [10].

Hoy en día, algunos programas de traducción automática permiten escoger un campo especializado o profesión en el cual se desea realizar la traducción, lo cual ayuda a delimitar el ámbito de términos a sustituir en el idioma destino. Esta técnica es efectiva en campos especializados los cuales ya tienen un modo de expresión establecido. Esto hace más fácil la traducción automática de textos jurídicos en comparación de textos más estandarizados.

Este proceso de traducción también puede ser mejorado gracias a la intervención humana, por ejemplo, algunos sistemas permiten al traductor escoger con anticipación los nombres propios, evitando que estos se traduzcan automáticamente. Con la ayuda de este tipo de técnicas, la traducción automática puede llegar a ser una herramienta de gran utilidad para los traductores, o incluso puede llegar a producir resultados que se puedan usar sin ningún tipo de modificación.

5.5. Corpus para la traducción automática

Un corpus lingüístico es un gran conjunto de ejemplos reales en el uso de una lengua. Estos ejemplos pueden presentarse escritos u orales, aunque lo más común es encontrarlos en formato de texto.

¿Cómo pueden ser útiles para la traducción?

A veces sabemos lo que una palabra significa, pero no sabemos cómo usarla, o sencillamente la definición del diccionario no nos convence y el significado no está totalmente claro. Los corpus nos presentan varias diferencias respecto a los diccionarios:

1. La palabra está rodeada de distintos contextos.
2. Más cercanía con el uso natural de la lengua.
3. Más pluralidad.

5.5.1. Tipos de corpus

Podemos encontrar varios tipos de corpus:

Corpus monolingües

Están formados por textos de una sola lengua. Se recopilan con el objetivo de dar cuenta de una lengua o variedad lingüística. Los corpus monolingües se pueden comparar a los diccionarios monolingües, pero a diferencia de los diccionarios, que ofrecen información sintáctica, los corpus ofrecen información analítica y un acercamiento más natural a la lengua. El BNC (*British National Corpus*) [11] es un ejemplo de corpus monolingüe en inglés. El CREA [12] o el CORDE [13] son ejemplos de corpus monolingües en español.

Corpus multilingües/bilingües

Este tipo de corpus trabajan con dos o más idiomas. Dentro de esta categoría podemos encontrar a su vez varios tipos de corpus.

Corpus comparables

Los Corpus comparables ("*paired texts*") consisten en compilaciones de textos en más de una lengua o variedad lingüística, que son similares al que nos interesa. Es lo que los traductores llaman "textos paralelos". Son muy útiles con el fin de adquirir jerga y vocabulario sobre el tema que se trata.

Corpus paralelos

Los Corpus paralelos ("*bi-texts*") son compilaciones de textos iguales en varios idiomas. Uno es la traducción del otro. El más sencillo se compone del original y su traducción. Son especialmente útiles en la traducción automática y en entornos bilingües o multilingües.

En una concordancia podemos encontrar varios tipos de información, como por ejemplo el significado de una palabra por el contexto, su comportamiento en una frase determinada, de qué otras palabras suele ir acompañada, la frecuencia de su uso, el estudio de su morfosintaxis o la estabilidad de sus construcciones.

Corpus alineados

Los corpus alineados son corpus paralelos en los que los textos están dispuestos unos al lado de otros en párrafos o frases. De esta forma, identificamos mejor las equivalencias de traducción. Estos pares de frases, llamados segmentos, están identificados mediante un programa específico de correspondencias de traducción. Los corpus también se pueden clasificar en robustos o virtuales.

Corpus robustos

Los corpus robustos almacenan una gran cantidad de información. Tienen un software de búsqueda, son estables y están aceptados por una comunidad y elaborados con criterios estrictos. Además, poseen una fiabilidad muy alta. Un ejemplo es el BNC (*British National Corpus*).

Corpus virtuales

Los corpus virtuales son elaborados por el mismo traductor para un proyecto concreto, son privados y con mucha información recogida de Internet. Son muy útiles para buscar información sobre términos y fraseología sobre un tema determinado. [14]

5.5.2. Utilidad de los corpus

Los corpus nos proporcionan información que no está disponible en cualquier parte. Son muy útiles para los traductores, para hablantes extranjeros, o simplemente para solventar cualquier duda que podamos tener sobre el uso de una palabra. En ciencias como la traducción o simplemente en el proceso de aprendizaje sobre una palabra, no debemos recurrir solo a los diccionarios. Los corpus son un complemento muy importante, ya que saber la definición de una palabra no nos ayuda necesariamente a usarla en una frase, no nos dice nada sobre si es habitual usarla o no ni tampoco sabemos su comportamiento general en otro idioma, ya que no todos se rigen por las mismas estructuras. Por eso, diccionarios y corpus deben ir acompañados.

6. Desarrollo del proyecto

El sistema desarrollado en este proyecto está compuesto de dos partes:

- Servidor, incluye receptor RSS, corpus paralelo, traductor y API REST.
- Cliente, aplicación Android que se encarga de proveer las interfaces gráficas para crear usuarios, iniciar sesión y mostrar las noticias.

Para la comunicación entre el servidor y el cliente se utilizó la API REST desarrollada en Java.
En general el sistema está conformado por cinco módulos, como se muestra en la Figura 7.

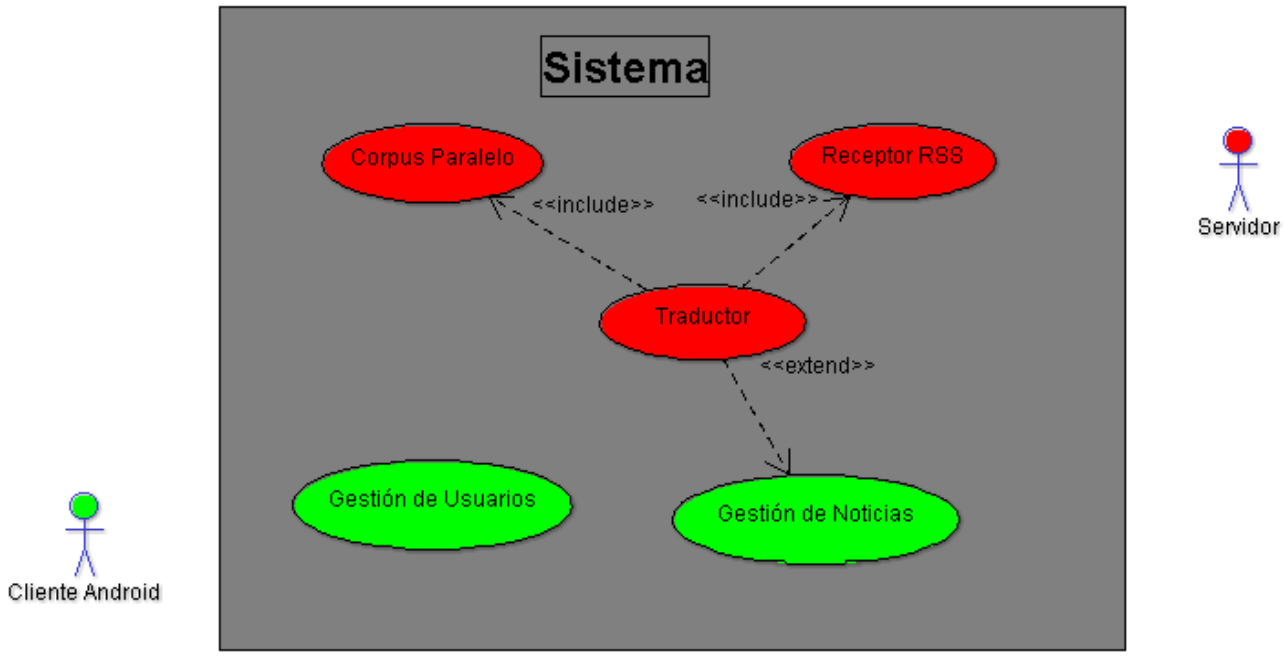


Figura 7: Casos de uso del sistema

6.1. Módulo Corpus paralelo.

El objetivo aquí es el manejo de la información para todo el sistema. En este módulo se utilizó el motor de base de datos MYSQL [15] y HIBERNATE [7] [16] para el mapeo de la base de datos.

El objetivo principal del sistema es la traducción de noticias, es por eso que este módulo es la base para ese fin porque el método pensado para realizarlo consiste en utilizar un corpus paralelo por lo que es fundamental tener un diccionario o corpus lo suficientemente grande para hacer la transcripción de la noticia lo mejor posible.

En el corpus paralelo se recopilo aproximadamente trece mil palabras y frases.

Aquí también se incluye el resto de la base de datos. Las tablas que se crearon son:

- Usuario
- Categoría
- Fuente
- Noticia
- Calificación

Para cada una de las tablas se mapeo con HIBERNATE, el código se puede ver en el apéndice A.1

6.1.1. Diagrama de la base de datos

El diagrama entidad relación completa se muestra en la Figura 8.

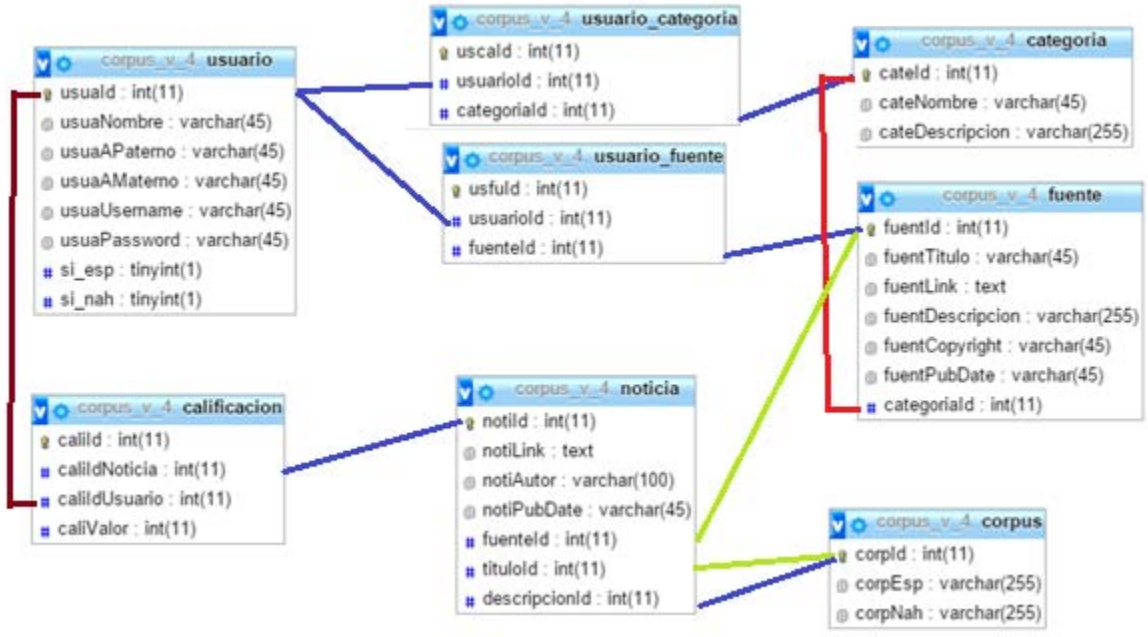


Figura 8: Diagrama entidad relación de la base de datos

[7] **HIBERNATE**, es una herramienta de Mapeo Objeto – Relacional para la plataforma Java que facilita el mapeo de atributos entre una base de datos.

6.1.2. Descripción técnica

El diagrama entidad relación se hizo en la aplicación PHPMYADMIN[8] [17], teniendo el diseño de la base de datos se procedió a crear el mapeo de datos en el IDE NETBEANS[9] [18].

Se siguieron los pasos siguientes para crear clases cuya finalidad fue mapear datos.

1. Crear el archivo para la configuración de HIBERNATE para conectarse a MYSQL.
2. Crear el archivo XML para la ingeniería inversa, para crear mapas y POJOS [10] de las tablas de la base de datos.
3. Ejecutar los dos archivos anteriores, en el IDE mencionado, y crear las mapas de HIBERNATE y los POJOS de las tablas.
4. Codificar las clases DAO [11] para cada uno de los POJOS.
5. Codificar una clase que reúne a todos los DAOs para tener un control universal sobre la información en la base de datos.

6.2. Módulo recolector.

El módulo se encarga principalmente de recolectar noticias en un canal RSS.
Para realizarlo se desarrolló un API con la intención de facilitar el acceso a las noticias junto con todos sus atributos a partir de una dirección URL que pertenece al canal (fuente RSS), en esta URL existe un archivo XML con la estructura que caracteriza un canal RSS para posteriormente parsearlo y finalmente obtener un objeto fuente con una colección de noticias.

La implementación se puede consultar en el apéndice A.2.

6.2.1. Diagrama del módulo recolector

El diagrama que muestra los casos de uso en este módulo está en la Figura 9.

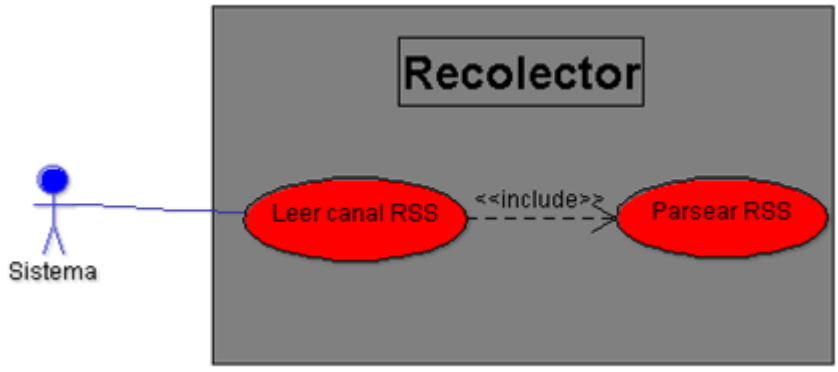


Figura 9: Casos de uso del módulo recolector

[8] **PHPMYADMIN**, es una herramienta libre para diseñar, implemtar y administrar bases de datos en la web.
[9] **NETBEANS**, es un entorno de desarrollo integrado libre para desarrollar en java.
[10] **POJOS** (*Plain Old Java Object*), son clases simples en java.
[11] **DAO** (*Data Access Object*), *es una clase java que suministra un irtefaz entre POJOS y base de datos.*

6.2.2. Descripción técnica

Como se explicó en el marco teórico, un canal RSS, es un archivo en formato XML que se pública en INTERNET y tiene la característica de cambiar su contenido cada cierto tiempo, lo que significa que se actualizan los ITEMS, que vienen siendo las noticias.

En el análisis se pudo observar que en la fuente RSS, hay un elemento padre llamado canal y esté contiene elementos hijos llamados ítems, tanto el canal como los ítems tienen atributos denominados título, descripción, url y fecha de publicación. Por lo que se creó una clase llamado ELEMENTO y de ahí se derivaron las clases CANAL e ITEM de esta forma los atributos comunes se colocaron en la clase padre.

Teniendo las clases mencionadas, se procedió a codificar un PARSER RSS, en esta clase se utilizó el paquete javax.xml.stream de la biblioteca de Java para procesar un archivo XML extraído de la URL recibida por el API.

Los pasos fueron:

1. Extraer la cadena de texto XML.
2. Leer la cadena con el paquete XML de la biblioteca de Java.
3. Parsear la fuente (canal) junto con las noticias(ítems).

Al final de este proceso se regresa en un objeto la fuente y su colección de objetos noticias.

6.3. Módulo traducción.

En esta parte se realiza la traducción utilizando un corpus paralelo que se encuentra en la base de datos. Es necesario el componente recolector y el diccionario para cumplir con la finalidad de este módulo. Para empezar se necesita la colección de noticias que regresa el API recolector, posteriormente se recorre la colección mencionada de tal forma que se van traduciendo las noticias al náhuatl con la ayuda del diccionario.

En el apéndice A.3 se muestra la codificación que pertenece a esta parte.

6.3.1. Diagrama del módulo traducción

El diagrama que muestra los casos de uso de este módulo se muestra en la Figura 10.

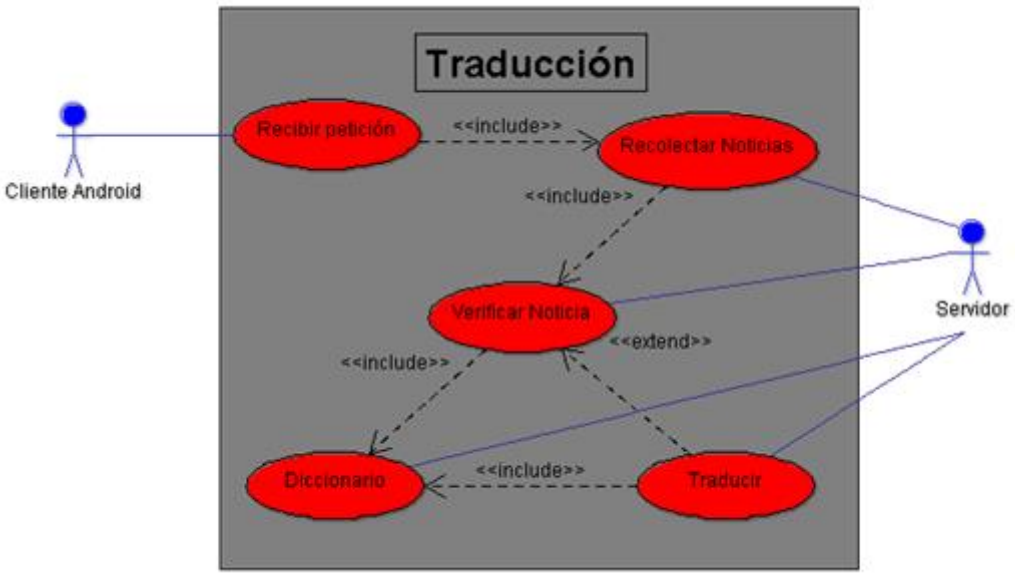


Figura 10: Casos de uso del módulo traducción

6.3.2. Descripción técnica.

Aquí se diseñó un algoritmo que tiene el propósito de hacer la traducción basándose en lo que existe en el corpus paralelo y en lo que se obtiene del API recolector RSS.

Este proceso se realiza cada vez que se recibe una petición desde el cliente cuando se escoge una fuente, entonces teniendo como base la fuente canal RSS, primero se hace un llamado al parseador RSS pasándole como argumento la URL de la fuente, el recolector regresa un objeto que contiene los atributos de la fuente y sus noticias, si no ocurre una excepción, se procede a recorrer la colección de noticias y en cada iteración se verifica el GUID, un identificador unívoco para la entrada, si ya existe en la base de datos, la traducción ya no se hace solo se lee lo que ya se encuentra almacenado, en caso contrario la traducción se realiza con el algoritmo que se describe a continuación:

Proceso segmentado de texto:

1. Leer la propiedad título o descripción del objeto noticia.
2. Dividir la cadena en segmentos, donde el delimitador es cualquier carácter especial o numérico, es decir cada segmento está conformado con caracteres alfabéticos y el espacio, los segmentos son frases que van a traducirse uno por uno.
3. Mandar secuencialmente los segmentos al "proceso traductor de segmentos" de esta forma se va ir formando el texto en náhuatl con los resultados regresados secuencialmente del "proceso traductor".
4. Fin

Proceso traducción de segmentos:

1. Recibir el segmento, llamémosle A, a traducir.

2. Iniciar una variable NAH igual a vacío donde se va guardar el texto en náhuatl.
3. Iniciar B igual a vacío.
4. Buscar A en el corpus.
5. Si A no se encuentra en el corpus y contiene n palabras, con n > 1, entonces A se le quita la última palabra, es decir A = palabra$_1$ palabra$_2$… palabra$_{n-1}$, y la última palabra, es decir la palabra$_n$ se concatena a la variable B. se regresa al paso 4.
6. Si A no se encuentra en el corpus y contiene 1 palabra, entonces A se concatena a NAH.
7. Si A se encuentra en el corpus, entonces A se concatena a NAH.
8. Si B contiene n palabras, con n>0, entonces A toma el valor de B y se regresa al paso 3.
9. Caso contrario, es decir B está vacío, entonces regresar NAH, que contiene la traducción del segmento.
10. Fin.

6.4. Módulo gestión de Usuarios

Este módulo se encarga de crear y autentificar usuarios en el sistema.

En la creación de usuario, los datos requeridos son, nombre completo, nombre de usuario, contraseña, indicar si habla español o náhuatl y por último, es opcional, seleccionar en una lista las fuentes y categorías de noticias de interés.

En la autentificación el usuario ingresa sus credenciales, en este caso es el nombre de usuario y la contraseña para ingresar al sistema

La implementación de este módulo se puede ver en el apéndice A.4.

6.4.1. Caso de uso: Registrarse

Si se desea leer las noticias en la aplicación es necesario que esa persona cuente con credenciales que lo identifiquen dentro del mismo.

Para obtener las credenciales, la persona debe ingresar los datos que se le piden en un formulario de registro, en donde se piden datos básicos para un perfil de usuario, estos son: nombre, sexo, si habla español o náhuatl , lista de categorías y fuentes de interés y necesariamente el nombre de usuario y contraseña, estos dos últimos datos son las que utilizará el usuario como credenciales para entrar al sistema y entonces dependiendo del perfil, el sistema tendrá funcionalidades variadas, por ejemplo, si el usuario habla las dos lenguas, el sistema le permitirá establecer una calificación a la noticia traducida o si escogió fuentes y categorías de noticias en el formulario, se le mostrará primero las que haya seleccionado.
A continuación, la ficha de caso de uso.

FICHA DE CASO DE USO	
ID	4.1
NOMBRE	Registrarse
DESCRIPCIÓN	Se carga un nuevo usuario a la base de datos.
FLUJO NORMAL	

ACTORES	Usuario final de la aplicación y la API REST
PRECONDICIONES	El usuario no debe existir previamente
ACTIVACIÓN	El usuario hace clic en el link “Registrarse”
DESCRIPCIÓN	1- El usuario ingresa a la interfaz gráfica que tiene dicho fin 2- Se ingresan los datos 3- Se envían los datos 4- Se regresa a la pantalla de acceso.
POSTCONDICIONES	Hay un nuevo usuario en la base de datos.
FLUJO ALTERNATIVO 1	
DESCRIPCIÓN	2A.- El usuario selecciona fuentes y categorías de noticias de su interés.
POSTCONDICIONES	En el perfil de usuario se registran las fuentes y categorías seleccionadas.
FLUJO ALTERNATIVO 2	
DESCRIPCIÓN	2B.-El usuario no ingresa uno o más campos
POSTCONDICIONES	Se informa con un mensaje de error el campo requerido que falta. Los datos previamente ingresados quedan en el formulario.
FLUJO ALTERNATIVO 3	
DESCRIPCIÓN	2C.- Se cancela el ingreso
POSTCONDICIONES	En la base datos hay la misma cantidad de usuarios. Se regresa a la pantalla de acceso.

6.4.2. Caso de uso: Inicio de sesión.

Se encarga de autentificar usuarios en el sistema, con el fin de proporcionar una interfaz gráfica personalizada de acuerdo al perfil del usuario.

FICHA DE CASO DE USO	
ID	4.2
NOMBRE	Inicio de sesión
DESCRIPCIÓN	El usuario desea ingresar al sistema.
FLUJO NORMAL	
ACTORES	Usuario final de la aplicación, API REST
PRECONDICIONES	El usuario debe estar en la base de datos.
ACTIVACIÓN	El usuario inicia la aplicación
DESCRIPCIÓN	1. El usuario ingresa su nombre de usuario. 2. El usuario ingresa su contraseña. 3. Se envían los datos al API REST 4. Se recibe información del API REST 5. Se verifica el status de las credenciales recibida del API REST.

	6. Si las credenciales son correctas se muestra la pantalla principal
POSTCONDICIONES	Se crea una variable de usuario en la aplicación.
FLUJO ALTERNATIVO 1	
DESCRIPCIÓN	A.- Registrarse
POSTCONDICIONES	Se abre la interfaz para crear usuarios
FLUJO ALTERNATIVO 2	
DESCRIPCIÓN	2A.-El usuario no ingresa algún campo o lo hace en un formato no permito y hace clic en enviar
POSTCONDICIONES	Se informa con un mensaje de error indicando la causa. Los datos previamente ingresados quedan en el formulario.
FLUJO ALTERNATIVO 3	
DESCRIPCIÓN	5A.- Las credenciales ingresadas son incorrectos
POSTCONDICIONES	Se mantiene en la misma interfaz. El formulario queda "limpio" para que se ingresen nuevos datos

6.4.3. Diagrama del módulo gestión de usuarios

El diagrama que muestra los casos de uso del módulo gestión de usuarios puede verse en la Figura 11.

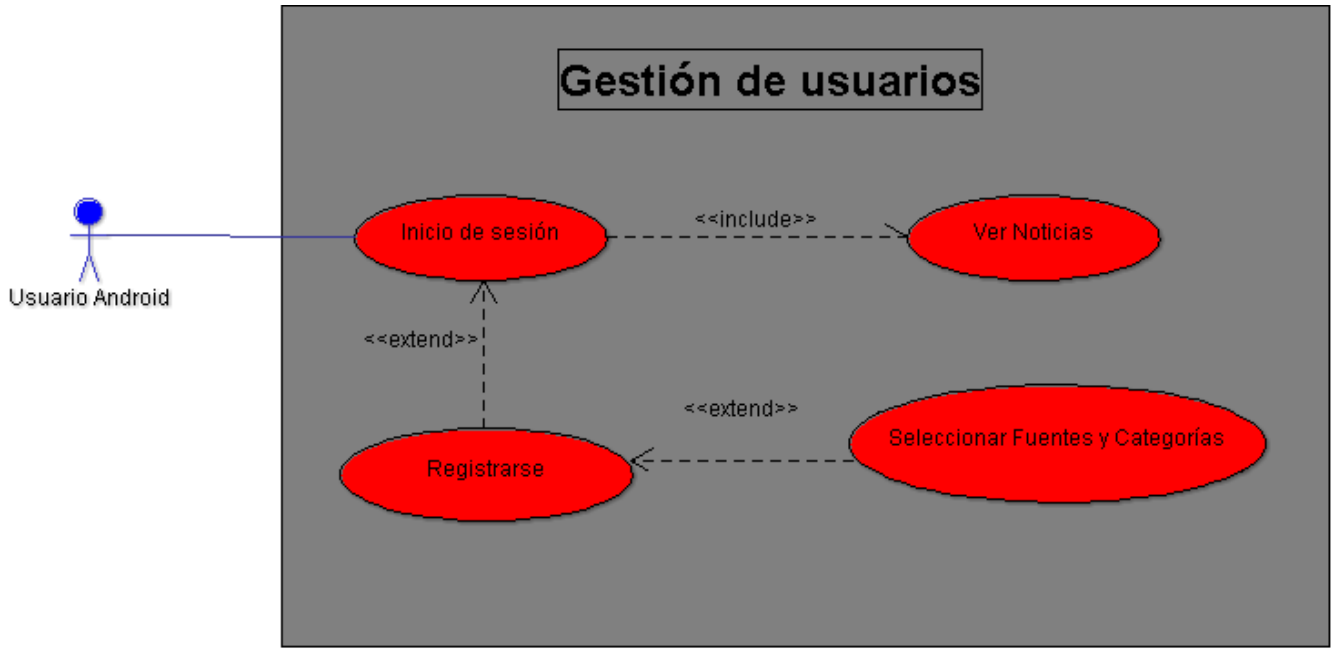


Figura 11: Casos de uso, registro e Inicio de sesión

6.4.4. Descripción técnica

En este módulo, como en todos, se utilizó el estilo de arquitectura REST (API REST) con el fin de que el cliente se comunique con el servidor, las invocaciones se realizan utilizando el FRAMEWORK RETROFIT[12] [19] en el cliente ANDROID al servidor con el contenedor de SERVLETS APACHE TOMCAT[13] [20].

Para el *caso de uso: Nuevo Usuario,* primero se crea un objeto usuario que es un POJO, luego se reciben los datos que se piden, haciéndoles una validación de formato, por ejemplo que el nombre y los apellidos sean de caracteres alfabéticos, que el nombre de usuario y la contraseña tengan una longitud mínima de 4 y máxima de 12 caracteres, que se establezca si el futuro usuario habla español o náhuatl en los CHECKBOX correspondientes, las dos primeras validaciones se realizan con expresiones regulares. Todo esto se hace inmediatamente después de hacer clic en el botón enviar de la interfaz. Si todas las validaciones son correctas entonces se agregan los datos al objeto en cuestión y después se verifica si el usuario ha hecho alguna selección de fuentes o categorías, en este caso también se agregan las listas al objeto, y por último se procede a hacer una petición de tipo POST con RETROFIT enviando como parámetro el objeto usuario. En el API REST se recibe el objeto y con HIBERNATE se agrega el usuario a la base de datos y las posibles respuestas al cliente pueden ser:

1. El usuario se agregó a la base de datos correctamente, se regresa un status válido y en el cliente ANDROID se muestra la interfaz de *autentificar*.
2. Ocurrió un error entre el servidor y MYSQL, se regresa un status inválido y un mensaje de la causa del error. En el cliente se muestra el mensaje y se mantiene en la misma interfaz.

Para el *caso de uso: Inicio de sesión*, antes se hacen las validaciones de formato para las credenciales y después se realiza una petición de tipo GET, utilizando el *FRAMEWORK RETROFIT* pasando como parámetros los dos datos que sirven como credenciales para el usuario, estos datos son, el nombre de usuario y la contraseña. En el servidor, el API REST recibe las credenciales para posteriormente iniciar un objeto que tiene como objetivo realizar operaciones en la base de datos utilizando el FRAMEWORK que tiene esta finalidad, esto es HIBERNATE, en este punto se realiza una consulta al motor de base de datos MYSQL con los datos recibidos, es decir, se ejecuta un QUERY como la que sigue: *"SELECT * FROM usuario WHERE nombreUsuario='nombreUsuarioRecibido' AND contraseña='contraseñaRecibido' "* con lo cual los resultados pueden ser:

1. No se encontró TUPLA alguna con el QUERY mencionado, esto significa que las credenciales son incorrectas, entonces se procede a responder al cliente con el status de usuario inválido. En Android se muestra el mensaje
2. Se encontró exactamente una TUPLA, esto significa que las credenciales son correctas, por lo cual se procede a responder con un status de usuario valido y además a regresar las listas preferidas del usuario, es decir el perfil del usuario. En el cliente se guarda el perfil en una variable de usuario y se muestra la pantalla principal de la aplicación.
3. Ocurrió un error entre el servidor y el motor MYSQL, se responde con un status de error y un mensaje con la causa del error. En el cliente se muestra el mensaje.

[12] **RETROFIT**, es un cliente REST para ANDROID y Java.
[13] **APACHE TOMCAT**, es un contenedor web.

6.5. Módulo gestión de noticias.

Aquí se realiza el almacenamiento de categorías, fuentes y noticias procedentes del módulo receptor a una base de datos para su posterior uso. La noticia está relacionada a una fuente RSS y la fuente a una categoría. Además, este módulo también se encarga de mostrar la lista de categorías, lista de fuentes, lista de noticias y, si aplica dependiendo del perfil del usuario, una interfaz de usuario para emitir una calificación a la noticia .La visualización de la noticia estará en función de la fuente a la que pertenezca y la visualización de la fuente dependerá de la categoría a la que pertenezca.

Los siguientes son casos de uso en este módulo: seleccionar categoría, seleccionar fuente, seleccionar noticia, mostrar noticia y, dependiendo el caso, calificar noticia.

En seleccionar categoría y seleccionar fuente hay dos modos de hacerlo, primero, mostrar las listas que el usuario haya seleccionado como preferidas, segundo, mostrar todas las categorías y fuentes que existen en el sistema. Con seleccionar categorías la finalidad es mostrar la lista de fuentes pertenecientes a la categoría seleccionada y en seleccionar fuente la finalidad es mostrar las noticias que pertenecen a la fuente seleccionada.

En seleccionar noticia, como se explicó en el párrafo anterior, primero se tiene que seleccionar una fuente para mostrar los ITEMS que pertenecen a la fuente seleccionada. Aquí la finalidad es mostrar la lista de las noticias y se hace presentando el título del suceso en ambas lenguas y el promedio de las calificaciones recibidas.

En mostrar noticia la finalidad es presentar el resultado de la traducción al usuario, y para este fin se hace una división de la pantalla, de tal modo que de un lado se visualiza el título y descripción de la noticia en náhuatl y del otro lado su equivalente en español.

En calificar noticia aplica solamente cuando el usuario sabe hablar español y náhuatl, esta información se indica al momento de registrarse en el sistema, la calificación se da en una escala de 1 al 5, el promedio de calificación les aparece a todos los usuarios en el caso de uso “seleccionar noticia”.

La implementación se puede ver en el apéndice A.5.

6.5.1. Caso de uso: Seleccionar categoría.

En la pantalla principal de la aplicación, en la parte posterior central, se encuentra una lista desplegable con cuatro opciones siguientes, todas las categorías, categorías preferidas, todas las fuentes y fuentes preferidas.

En la aplicación móvil, cuando un usuario ingresa al sistema con credenciales correctas, primero, el sistema verifica si en el perfil de este usuario hay al menos una fuente, si es el caso entonces se trata del caso de uso “seleccionar fuente”, caso contrario, se verifica si al menos hay una categoría en el perfil, en este caso, en la lista desplegable hay una opción llamado “Categorías preferidas” al seleccionarlo se muestran las categorías preferidas del usuario. En caso de que no haya categorías preferidas para el usuario, entonces hay otra opción en la misma lista desplegable llamado “Todas las categorías” consecuentemente el resultado es la lista de todas las categorías del sistema.

En cualquiera de los casos, categorías preferidas o todas las categorías, se tiene una lista y al seleccionar una la consecuencia es mostrar las fuentes que pertenezcan a la categoría seleccionada.

FICHA DE CASO DE USO	
ID	5.1
NOMBRE	Seleccionar categoría de las fuentes RSS.
DESCRIPCIÓN	Se escoge una categoría en una lista, puede ser de categorías preferidas o de todas las categorías
FLUJO NORMAL	
ACTORES	Usuario final de la aplicación y la API REST.
PRECONDICIONES	Existe la variable usuario donde esta guardada las categorías preferidas en caso de existir
ACTIVACIÓN	El usuario ingreso con credenciales válidas.
DESCRIPCIÓN	1- El usuario selecciona “categorías preferidas” o “todas las categorías” 2- Si no se han cargado previamente, se hace la petición al API REST para cargarlo. 3- Se carga o se recibe la lista. 4- Se muestra la lista. 5- Se selecciona una categoría.
POSTCONDICIONES	Se tiene una categoría seleccionada y consecuentemente se pasa al caso de uso “seleccionar fuente”.
FLUJO ALTERNATIVO 1	
DESCRIPCIÓN	A.- El usuario selecciona las dos opciones restantes de la lista desplegable que son, “fuentes preferidas” o “todas las fuentes” y estas dos opciones pertenecen al caso de uso “seleccionar fuente”.
POSTCONDICIONES	Se trata del caso de uso 5.2, lo que se describe a continuación.

6.5.2. Caso de uso: Seleccionar fuente.

Primero es importante aclarar que cada vez que se refiere “fuente” en este proyecto concretamente se hace mención a un canal RSS, es decir un archivo XML ubicado en un URL dentro de internet, el módulo recolector lo busca, extrae y parsea la información que vienen siendo las noticias. Cuando un usuario ingresa a la aplicación móvil con sus credenciales correctas, primero como se mencionó en el caso de uso anterior, el sistema verifica si hay al menos una fuente preferida en el perfil de este usuario, en este caso, se muestra esa lista y, en caso contrario, en la lista desplegable se encuentra la opción “Todas las fuentes” y consecuentemente se muestra la lista de todas las fuentes registradas en el sistema.

Aquí la consecuencia al seleccionar una fuente es mostrar la lista de noticias que pertenecen a lo escogido. En este caso de uso se cumple el objetivo principal del proyecto, es por eso que aquí se combinan todos los módulos. Primero se selecciona una fuente, se hace una petición al API REST y en el servidor el recolector busca las noticias en la URL de la fuente luego traduce cada uno con el apoyo del diccionario para finalmente regresar el resultado a la aplicación móvil.

FICHA DE CASO DE USO	
ID	5.2
NOMBRE	Seleccionar fuente RSS.
DESCRIPCIÓN	Se escoge una fuente en una lista.

FLUJO NORMAL	
ACTORES	Usuario final de la aplicación y la API REST.
PRECONDICIONES	Existe la variable usuario donde se encuentra almacenada las fuentes preferidas en caso de existir.
ACTIVACIÓN	El usuario ingreso con credenciales válidas.
DESCRIPCIÓN	1- El usuario selecciona “fuentes preferidas” o “todas las fuentes”. 2- Si no se han cargado previamente, se hace la petición al API REST 3- Se carga o se recibe la lista. 4- Se muestra la lista. 5- Se selecciona una fuente.
POSTCONDICIONES	Se tiene una colección de noticias que pertenecen a la fuente escogida.
FLUJO ALTERNATIVO 1	
DESCRIPCIÓN	A.- El usuario selecciona “categorías preferidas” o “todas las categorías” de la lista desplegable.
POSTCONDICIONES	Se trata del caso de uso 5.1

6.5.3. Caso de uso: Seleccionar noticia.

Este caso de uso se produce cuando se selecciona una fuente, caso de uso anterior, donde el API REST regresa una colección de noticias con su traducción al náhuatl pertenecientes al canal RSS seleccionada y en esta parte el usuario visualiza la lista en donde cada elemento se muestra el título en dos lenguas y el promedio de calificación recibida con el objetivo de que el usuario seleccione una de su interés para visualizarlo por completo.

FICHA DE CASO DE USO	
ID	5.3
NOMBRE	Seleccionar noticia de la lista regresada del caso de uso seleccionar fuente
DESCRIPCIÓN	Se escoge una noticia de la lista.
FLUJO NORMAL	
ACTORES	Usuario final de la aplicación y la API REST.
PRECONDICIONES	Existe la variable usuario.
ACTIVACIÓN	El usuario selecciona una fuente para cargar las noticias
DESCRIPCIÓN	1- Se carga una nueva interfaz. 2- Se hace una petición al API REST pasando la fuente seleccionada. 3- Se recibe la lista de noticias. 4- Se muestra la lista con el promedio de calificación recibida. 6- Se selecciona una noticia.
POSTCONDICIONES	Se tiene una noticia seleccionada.
FLUJO ALTERNATIVO 1	
DESCRIPCIÓN	3A.- El usuario regresa a la pantalla principal.

POSTCONDICIONES	Se muestra la lista de donde se seleccionó la fuente.

6.5.4. Caso de uso: Mostrar noticia.

Al momento de cargar la lista de noticias, en el caso de uso anterior, las noticias se cargan completamente, es decir, en la memoria se guarda la noticia en español y en náhuatl por lo que aquí es para mostrar la noticia en su totalidad, es decir, mostrarlo en español y en náhuatl, además de una interfaz para emitir una calificación si aplica, es decir si el usuario sabe hablar español y náhuatl.

FICHA DE CASO DE USO	
ID	5.4
NOMBRE	Mostrar noticia.
DESCRIPCIÓN	Se muestra la noticia en náhuatl y en español.
FLUJO NORMAL	
ACTORES	Usuario final de la aplicación.
PRECONDICIONES	Se tiene una noticia seleccionada.
ACTIVACIÓN	El usuario selecciona una noticia.
DESCRIPCIÓN	1- Se obtiene la noticia de la memoria. 2- Se muestra la noticia dividida en español y náhuatl.
POSTCONDICIONES	Se visualiza una noticia.
FLUJO ALTERNATIVO 1	
DESCRIPCIÓN	2A.- Si aplica, el usuario emite una calificación.
POSTCONDICIONES	Se hace una petición al API REST para guardar la calificación.

6.5.5. Caso de uso: Calificar noticia.

Este caso de uso solo aplica para los usuarios que hablan las dos lenguas, aquí se permite emitir una calificación en una escala de 1 al 5

FICHA DE CASO DE USO	
ID	5.5
NOMBRE	Calificar noticia.
DESCRIPCIÓN	Se guarda una calificación a la traducción de la noticia.
FLUJO NORMAL	
ACTORES	Usuario final de la aplicación y la API REST.
PRECONDICIONES	Se tiene una noticia seleccionada.
ACTIVACIÓN	El usuario selecciona la calificación que desea darle a la traducción de la noticia.

DESCRIPCIÓN	1. Se hace una petición al API REST pasando como argumento el valor de la calificación seleccionada. Se guarda en la base de datos el valor de la calificación
POSTCONDICIONES	En la base de datos se guarda el valor de la calificación emitida por el usuario.

6.5.6. Diagrama del módulo gestión de noticias

Diagrama que muestra los casos de uso en el módulo gestión de noticias, ver Figura 12.

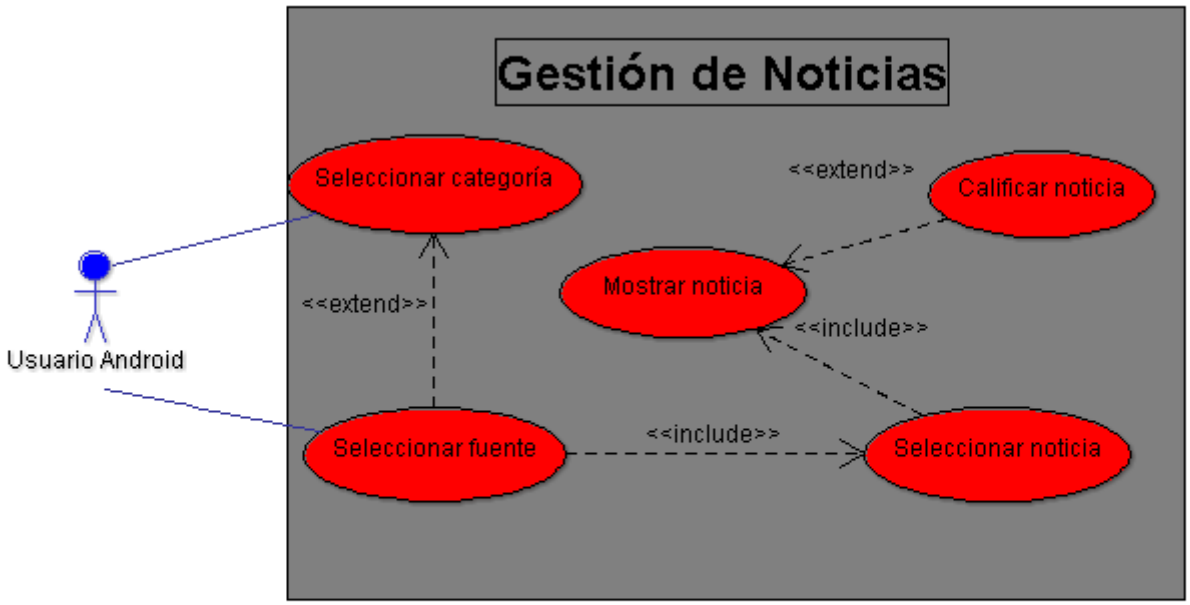


Figura 12: Casos de uso gestión de noticias

6.5.7. Descripción técnica

Esta aplicación fue desarrollada con el patrón de arquitectura de software Modelo-Vista-Controlador (MVC). Las vistas se crearon utilizando un LAYOUT[14] y dentro de esta, elementos WIDGETS[15]codificándolos en XML. Los controladores fueron creadas con clases en Java y en algunos casos se heredó de una clase controlador ya existente, por ejemplo en los casos de uso de selección se utilizó en la vista un WIDGET ListView [16] y en el controlador una clase heredada de "BaseAdapter". El modelo fueron los POJOS de las base de datos, es decir, todo lo que se recibe de la API REST.

[14] **LAYOUT**, es un contenedor de una o más vistas y controla su comportamiento.
[15] **WIDGETS**, son una subclase ya implementadas que dibujan los elementos en la pantalla.
[16] **LISTVIEW**, elemento que visualiza una lista vertical.

En la pantalla principal, la interfaz para la gestión de noticias, está conformado por un LAYOUT como contenedor y dentro se encuentra un ListView que tiene como fin visualizar la lista de forma vertical, donde cada componente es una categoría, fuente o noticia dependiendo lo que se deseé listar. Cada componente tiene también un LAYUOT con TextViews para la información a mostrar e ImageViews para imágenes. En la lógica del negocio se crearon tres clases derivadas de la clase "BaseAdapter" existente en la biblioteca estándar de Android , que sirve como controlador para el ListView, en cada una de estas clases derivadas lo que se personalizo fue la lista de componentes con su respectivo LAYOUT para mostrarlo correctamente.

En los casos de uso: Seleccionar categoría y Seleccionar fuente, la primera intención es mostrar la lista de categorías o fuentes de preferencia para el usuario, estas listas de componentes se cargan cuando el usuario se autentifica, por eso el primer proceso es instanciar un objeto a la clase controladora que le corresponde al componente en cuestión pasando como argumento la lista para luego relacionarla con ListView de la interfaz principal y el controlador padre se encarga del resto para mostrarla. Ahora en el caso de que el usuario seleccione ver todas las categorías o fuentes entonces antes del proceso anterior lo que se hace es verificar si anteriormente ya se han cargado del API REST todos los componentes solicitados, en caso contrario se procede a realizar la petición para cargarlos y después repetir el primer proceso.

En el API REST al momento de atender la petición lo que se hace es una consulta a la tabla categoría o fuente dependiendo el caso para finalmente regresar el resultado de la consulta. La diferencia entre estos dos casos de uso es lo siguiente, en seleccionar categoría la finalidad es cargar fuentes relacionados con la categoría, en cambio seleccionar fuente tiene como objetivo leer noticias de la fuente seleccionada y esto es el caso de uso siguiente.

En el caso de uso: Seleccionar noticia, cuando se selecciona una fuente en el controlador se obtiene el objeto fuente para después hacer una petición al API REST pasando como argumento la fuente. En este punto se relaciona este módulo con los módulos diccionario, recolector y traductor, que son el corazón del sistema, al trabajar en conjunto lo que se obtiene es una lista de noticias recientes traducidas, preparadas para ser leídas y precisamente esta colección es la que regresa el API REST para que finalmente en el cliente se muestren con el proceso descrito anteriormente en los demás casos de uso.

En el caso de uso: Mostrar noticia, es el único que se diferencia de los demás por el hecho de que no muestra una lista si no un solo componente, la noticia en español y en náhuatl, para este punto el LAYOUT contiene más LAYOUTs, TextViews y además un RatingBar que es el componente para la calificación con un rango de uno a cinco estrellas, sí aplica. Aquí sucede lo mismo como en los otros casos, teniendo el objeto con toda la información de la noticia cada atributo se relaciona con su vista correspondiente en el LAYOUT correspondiente a la noticia. Sí aplica, al momento de mostrar la noticia, se pide al API REST la calificación de la noticia por si anteriormente el mismo usuario lo haya establecido, en caso contrario se regresa un cero. En el momento de establecer una calificación se realiza una petición al API REST pasando como argumento la calificación, la noticia en cuestión, el usuario que emite esta calificación, recordemos que la información del usuario se estable en una variable cuando se autentifica, En el lado servidor al atender esta petición, se realiza una inserción en la base de datos en la tabla calificación con las llaves foráneas de la noticia y el usuario para tener una relación para su posterior uso en la petición de obtener calificación anterior.

7. Resultados

Los resultados obtenidos después de implementar todos los módulos se reflejan en la gestión de usuarios y en la gestión de noticias, por lo que se hicieron las pruebas siguientes por cada caso de uso.

7.1. Gestión de usuarios

7.1.1. Registrarse

Prueba 1.1: Registrarse

Se ingresó la información en cada uno de los campos: nombre, apellido paterno, apellido materno. Se definió un nombre de usuario, se creó una contraseña, se confirmó la contraseña, se marcaron los dos checks correspondientes al habla de náhuatl o español, se pulso el botón “CATEGORIAS” y apareció una lista de todas las categorías existentes en el sistema y se le dio check al segundo elemento de la lista, luego se pulso el botón “FUENTES” y apareció una lista de todas la fuentes existentes en el sistema y se escogieron los dos últimos elementos de la lista y para terminas se pulso el botón “ENVIAR”, ver Figura 13.

Figura 13: Prueba 1.1, registrarse

Prueba 1.2: Registrarse

El mismo caso de la prueba anterior, con la diferencia de que solo se le dio check al habla español y en la lista de categorías y fuentes se escogió elementos diferentes, ver Figura 14.

Figura 14: Prueba 1.2, registrarse

Prueba 1.3: Registrarse

Se ingresó la información requerida y sin pulsar ninguna de los dos botones correspondientes a las categorías y fuentes, se pulso el botón “ENVIAR” por lo que se muestra una mensaje de alerta informando que se tiene la posibilidad de escoger algunas fuentes como preferidas por lo que se procede a pulsar el botón “FUENTES” y seleccionar una fuente para finalmente volver a pulsar el botón “ENVIAR” y guardar la información capturada, ver Figura 15.

Figura 15: Prueba 1.3, registrarse

Prueba 1.4: Registrarse

Se escoge una fuente como preferida, ninguna categoría y se manda a guardar, ver Figura 16.

Figura 16: Prueba 1.4, registrarse

Prueba 1.5: Registrarse

Se escogen tres fuentes preferidas y una categoría, ver Figura 17.

Figura 17: Prueba 1.5, registrarse

Prueba 1.6: Registrarse

No se escoge ninguna categoría, solo fuentes, ver Figura 18.

Figura 18: Prueba 1.6, registrarse

Prueba 1.7: Registrarse

Se ingresa la información de una persona llamada Ana cristina Segura Carbajal con nombre de usuario igual a "anacristina", solo habla español y le llama la atención el canal "El universal Online – Minuto a minuto", ver Figura 19.

Figura 19: Prueba 1.7, registrarse

7.1.2. Inicio de sesión

Prueba 2.1: Inicio de sesión

Se intenta ingresar al sistema con el nombre de usuario de camilo, pero la contraseña ingresada es incorrecta por lo que se muestra un aviso indicando este error, ver Figura 20.

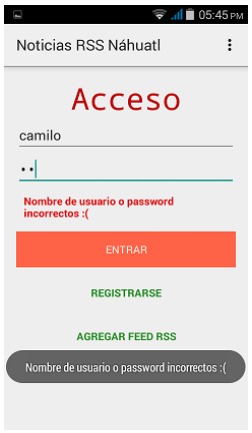


Figura 20: Prueba 2.1, Inicio de sesión

Prueba 2.2: Inicio de sesión

Se vuelve a intentar con el mismo nombre de usuario de la prueba anterior ingresando otra contraseña, las credenciales resultan correctas por lo que se muestra la pantalla principal de la aplicación indicando un inicio de sesión satisfactorio, ver Figura 21.

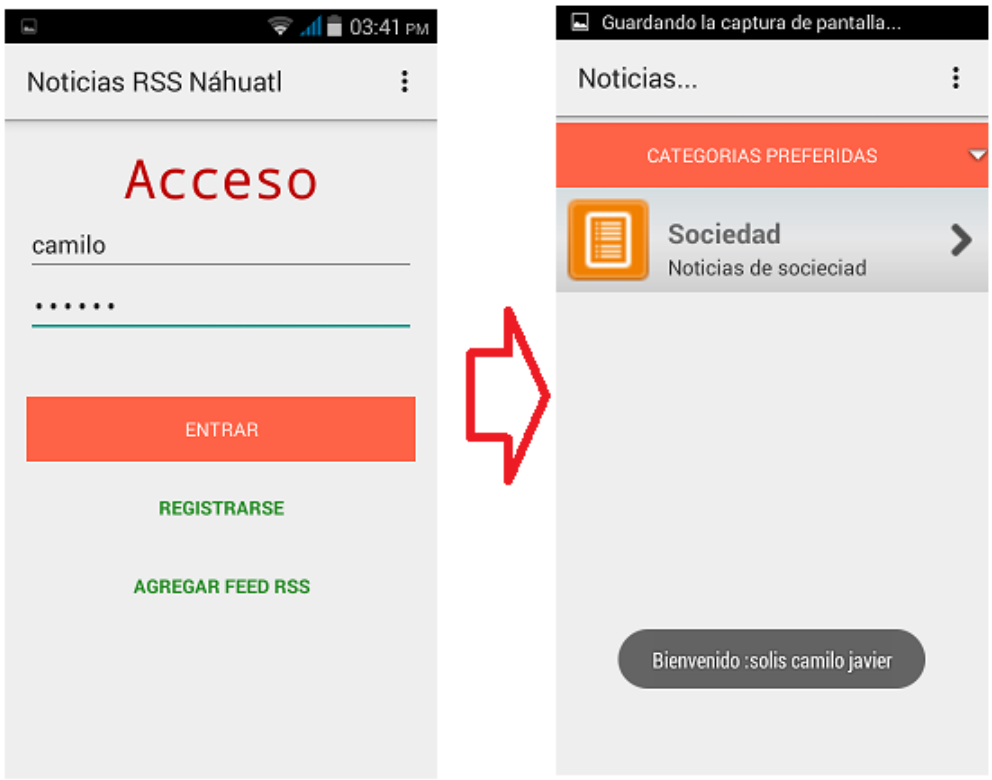


Figura 21: Prueba 2.2, Inicio de sesión

Prueba 2.3: Inicio de sesión

Se ingresa al sistema con el nombre de usuario de solano, en su perfil tiene dos fuentes y una categoría como preferidas, ver Figura 22.

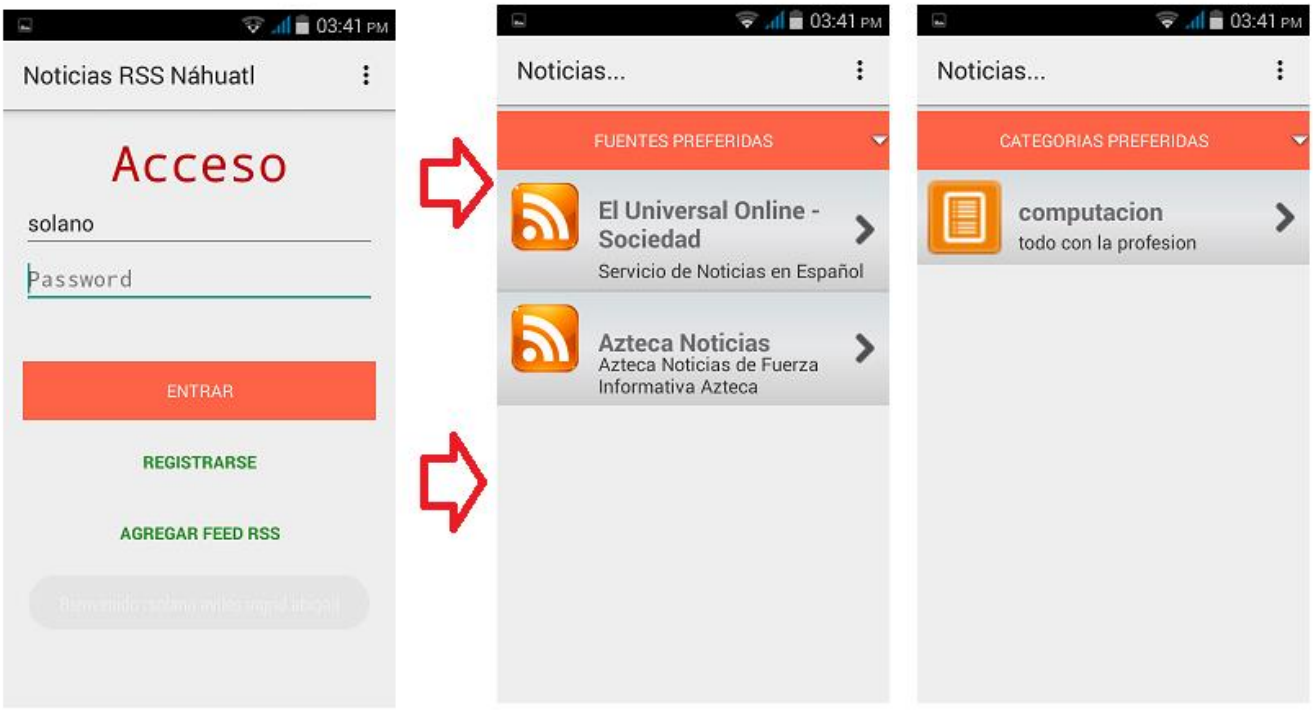


Figura 22: Prueba 2.3, Inicio de sesión

Prueba 2.4: Inicio de sesión

Se ingresa al sistema con el nombre de usuario isaura, este usuario solo tiene una fuente como preferida y una categoría también, ver Figura 23.

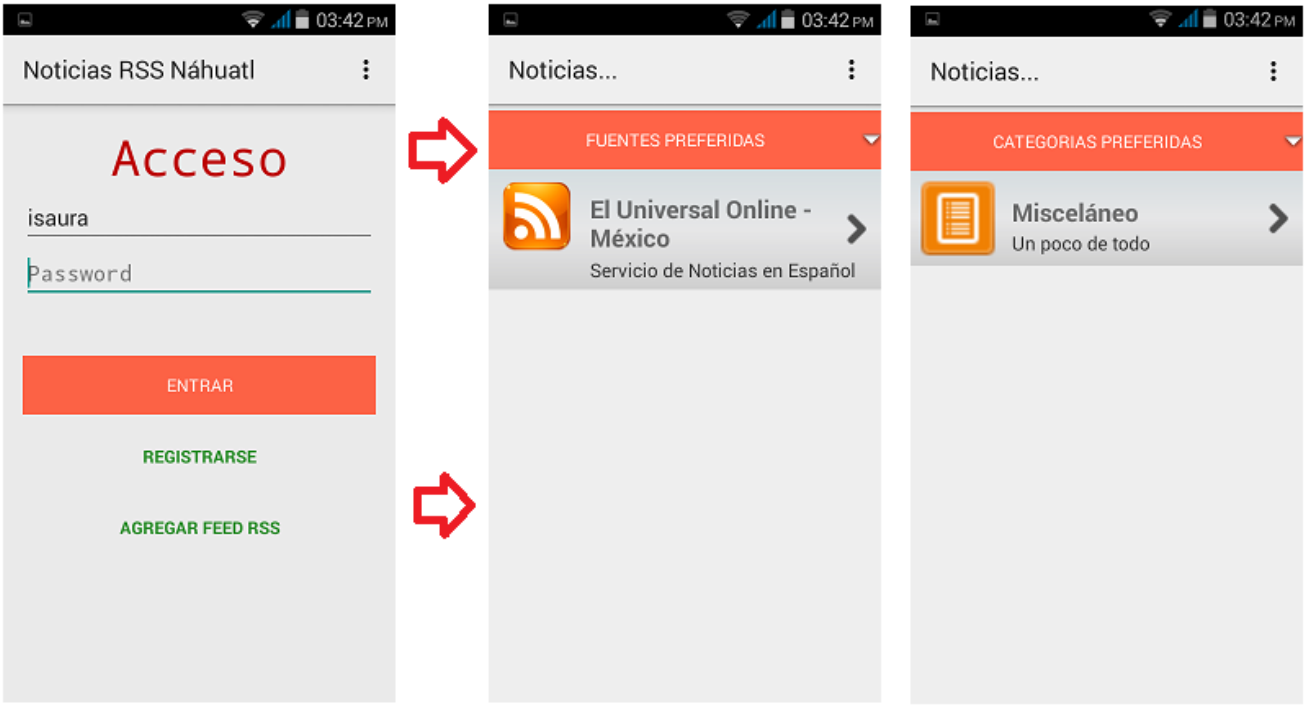


Figura 23: Prueba 2.4, Inicio de sesión

Prueba 2.5: Inicio de sesión

El usuario ”marco” tiene una fuente como preferida y ninguna categoría, ver Figura 24.

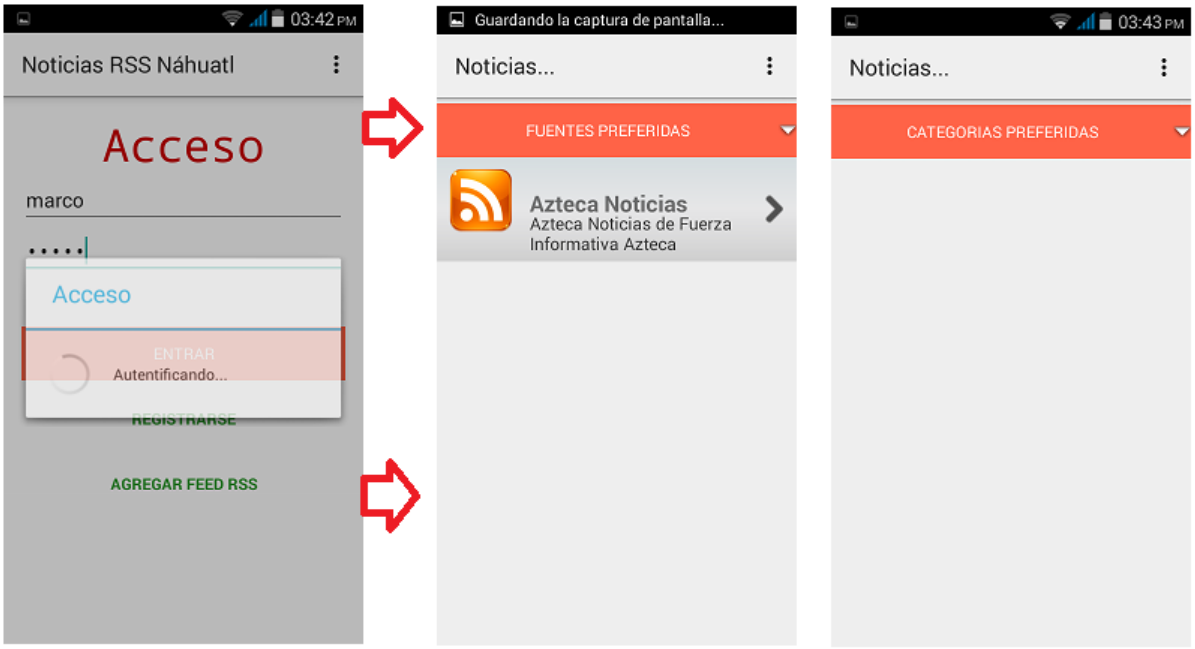


Figura 24: Prueba 2.5, Inicio de sesión

Prueba 2.6: Inicio de sesión

Con las credenciales de “rocio”, se ingresa al sistemas también y nos arroja que tiene tres fuentes como preferidas y una categoría, ver Figura 25.

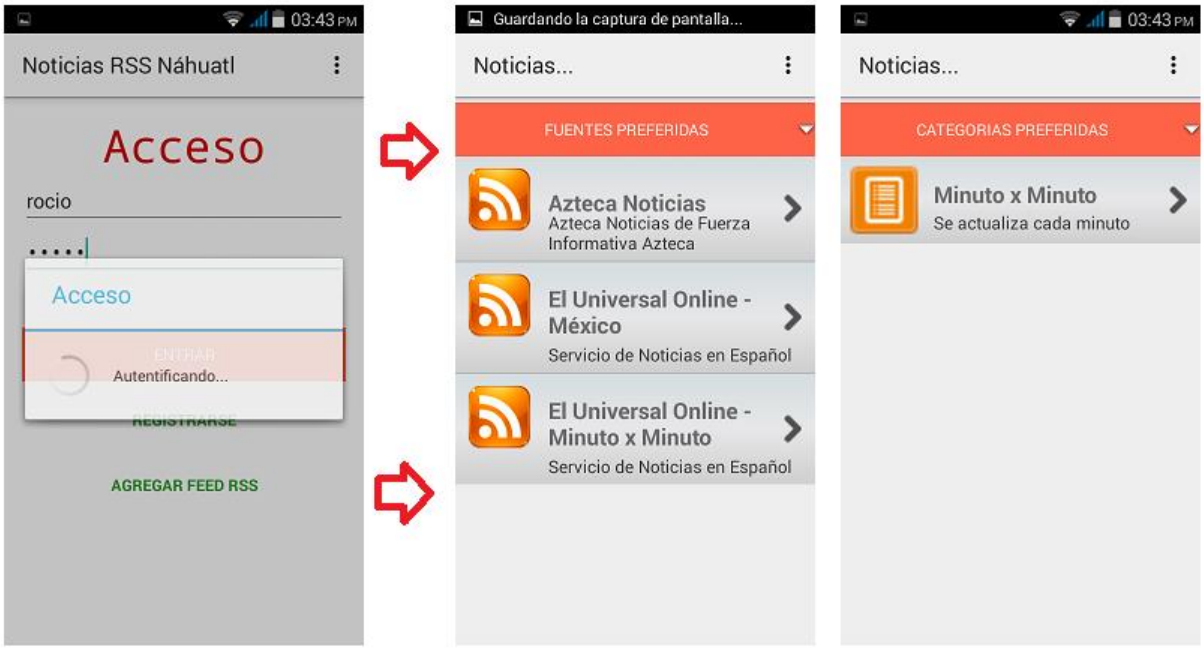


Figura 25: Prueba 2.6, Inicio de sesión

Prueba 2.7: Inicio de sesión

Para “bruno”, en su perfil está almacenado una fuente, ver Figura 26.

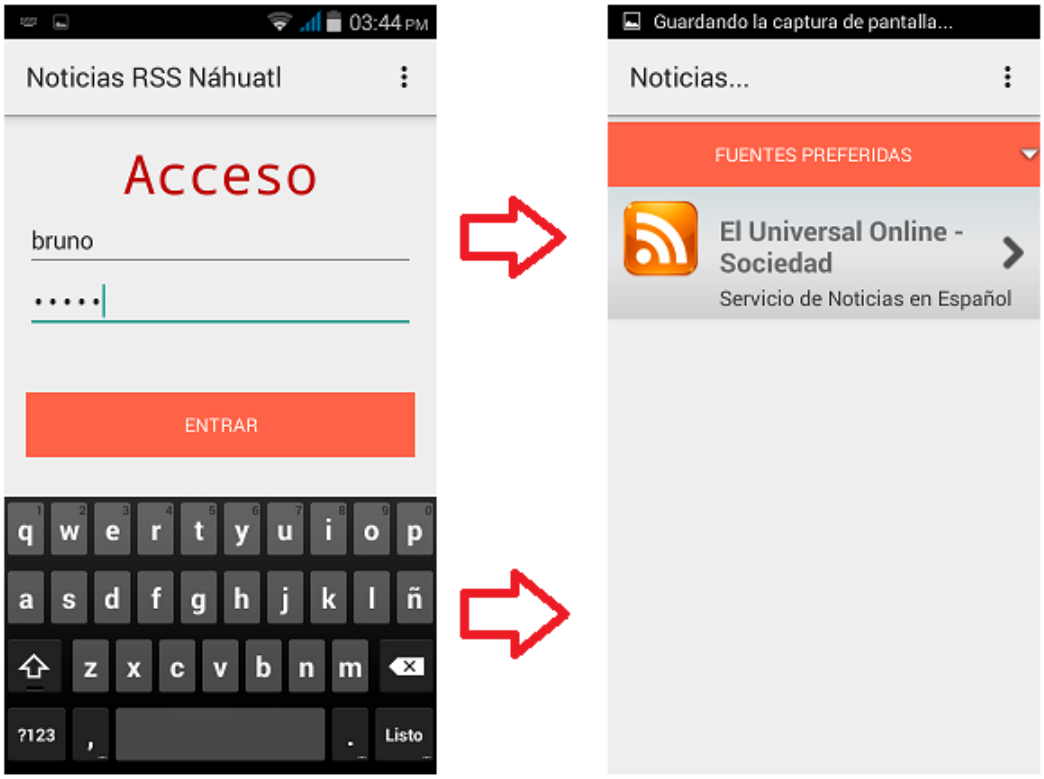


Figura 26: Prueba 2.7, Inicio de sesión

7.2. Gestión de noticias

7.2.1. Seleccionar categoría

Prueba 3.1: Seleccionar categoría

Se escoge en la lista desplegable la opción TODAS LAS CATEGORÍAS, al cargarse se selecciona la categoría “Sociedad” y se cargan las fuentes que pertenecen a la categoría, como se observa en la Figura 27.

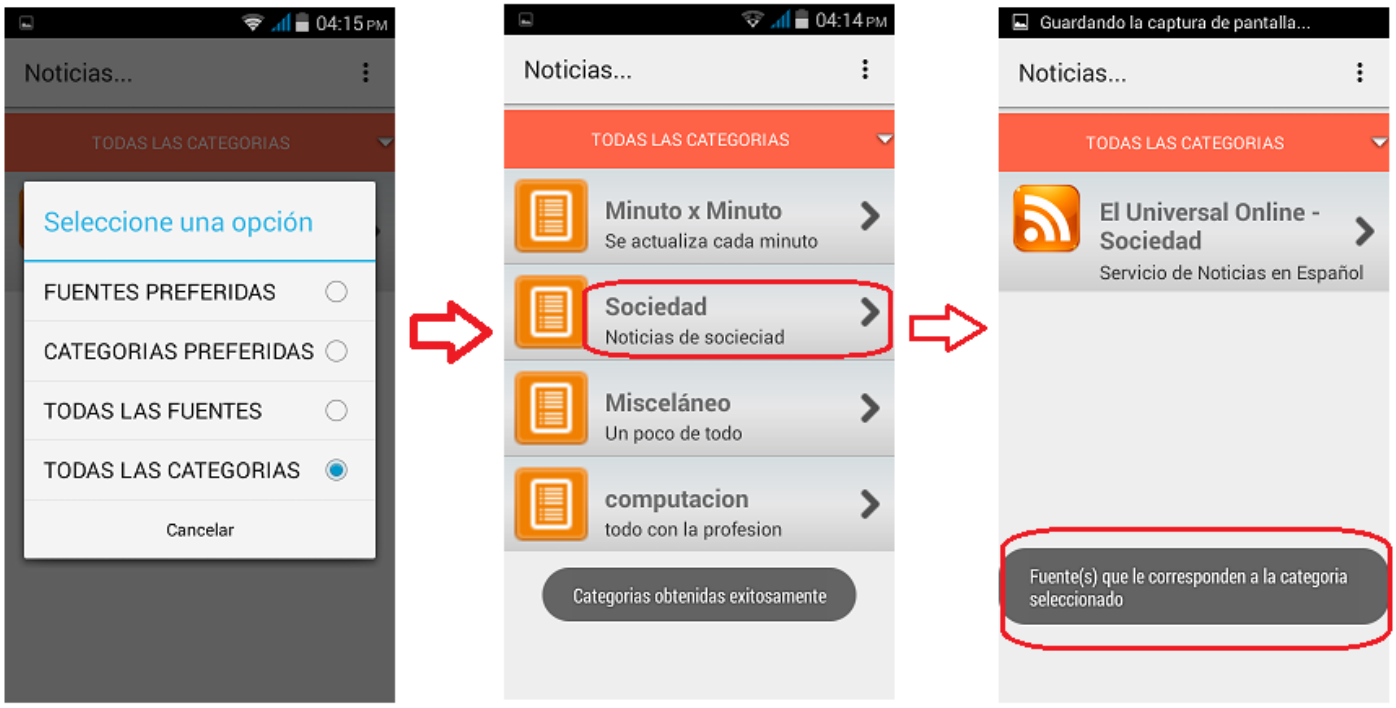


Figura 27: Prueba 3.1, seleccionar categoría

Prueba 3.2: Seleccionar categoría

Ahora se selecciona “Misceláneo” y se carga una fuente, la única que pertenece a esta categoría, ver Figuro 28.

Figura 28: Prueba 3.2, seleccionar categoría

Prueba 3.3: Seleccionar categoría

Se selecciona “Minuto x Minuto” y se carga la fuente “El Universal Online – Minuto x Minuto”, ver Figura 29.

Figura 29: Prueba 3.3, seleccionar categoría

Prueba 3.4: Seleccionar categoría

Se escoge “computación” y se carga la fuente “El Universal Online – México”, ver Figura 30.

Figura 30: Prueba 3.4, seleccionar categoría

7.2.2. Seleccionar fuente

Prueba 4.1: Seleccionar fuente

Se escoge la primera fuente, se cargan y se muestran las noticias que contiene el canal RSS seleccionado, ver Figura 31.

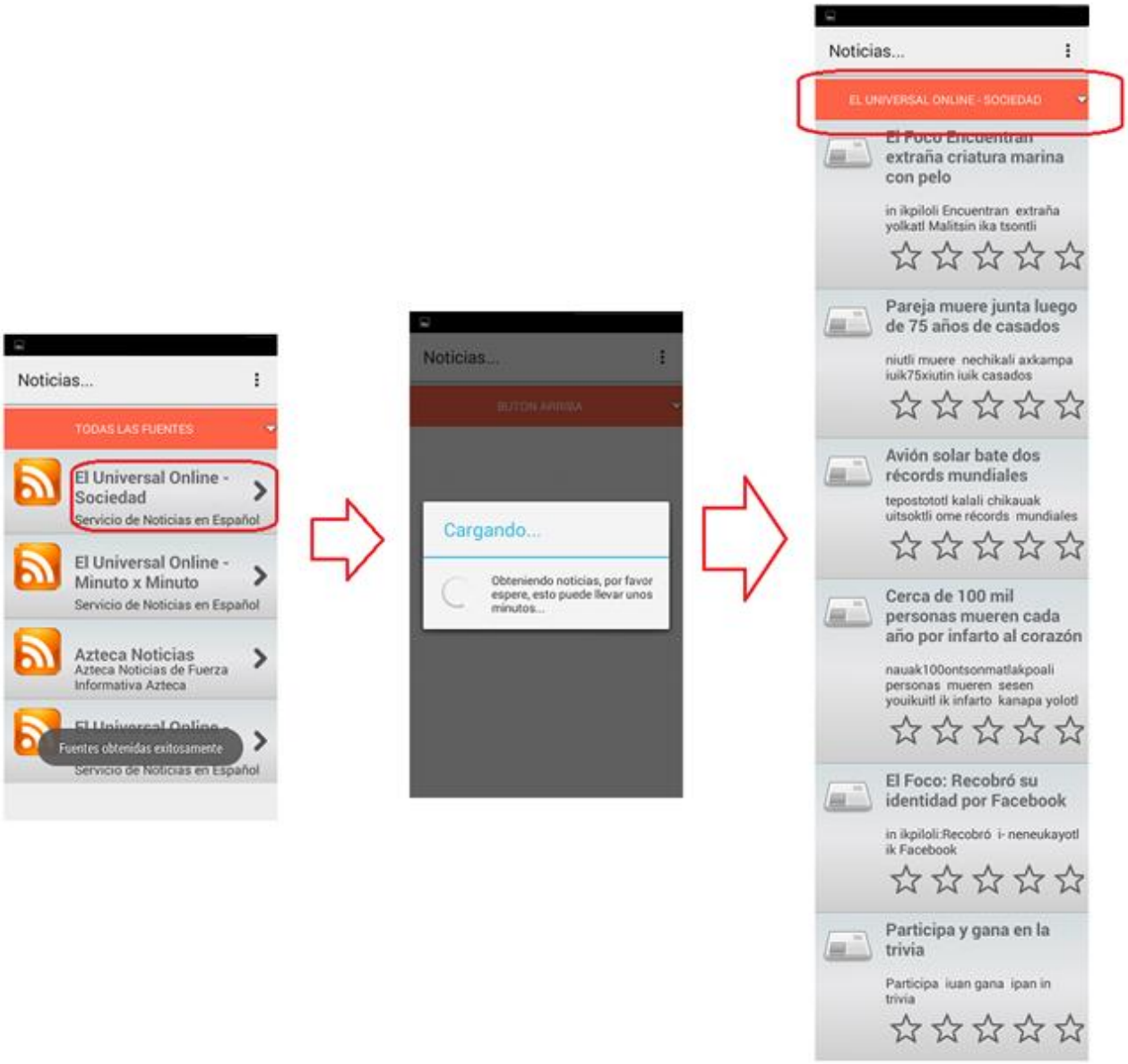


Figura 31: Prueba 4.1, seleccionar fuente

Prueba 4.2: Seleccionar fuente

Se escoge el segundo canal RSS, se cargan las noticias y se muestran, ver Figuro 32.

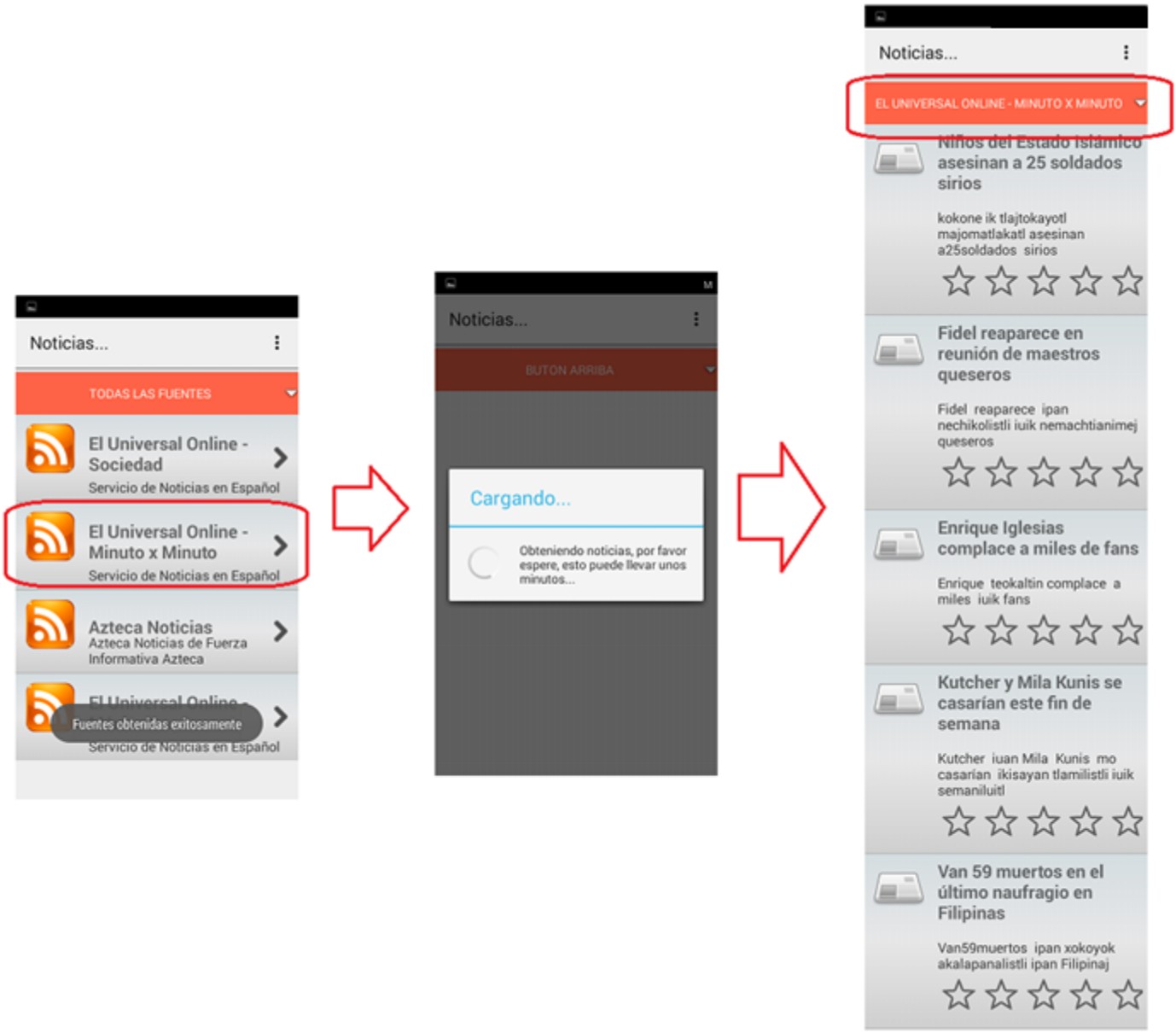


Figura 32: Prueba 4.2, seleccionar fuente

Prueba 4.3: Seleccionar fuente

Se selecciona “AZTECA NOTICIAS” y se reciben los ítems del canal RSS, ver Figura 33.

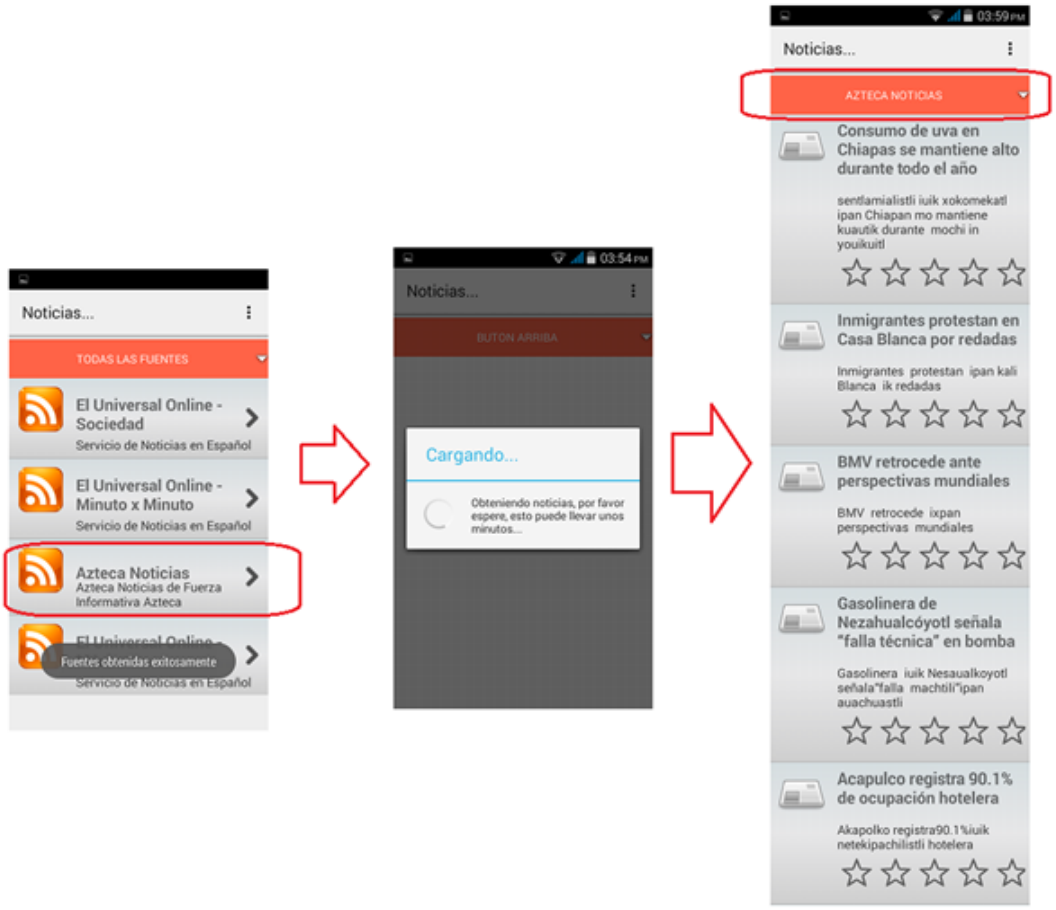


Figura 33: Prueba 4.3, seleccionar fuente

Prueba 4.4: Seleccionar fuente

Seleccionar la fuente “El Universal Online - México”, se carga una noticia, ver Figura 34.

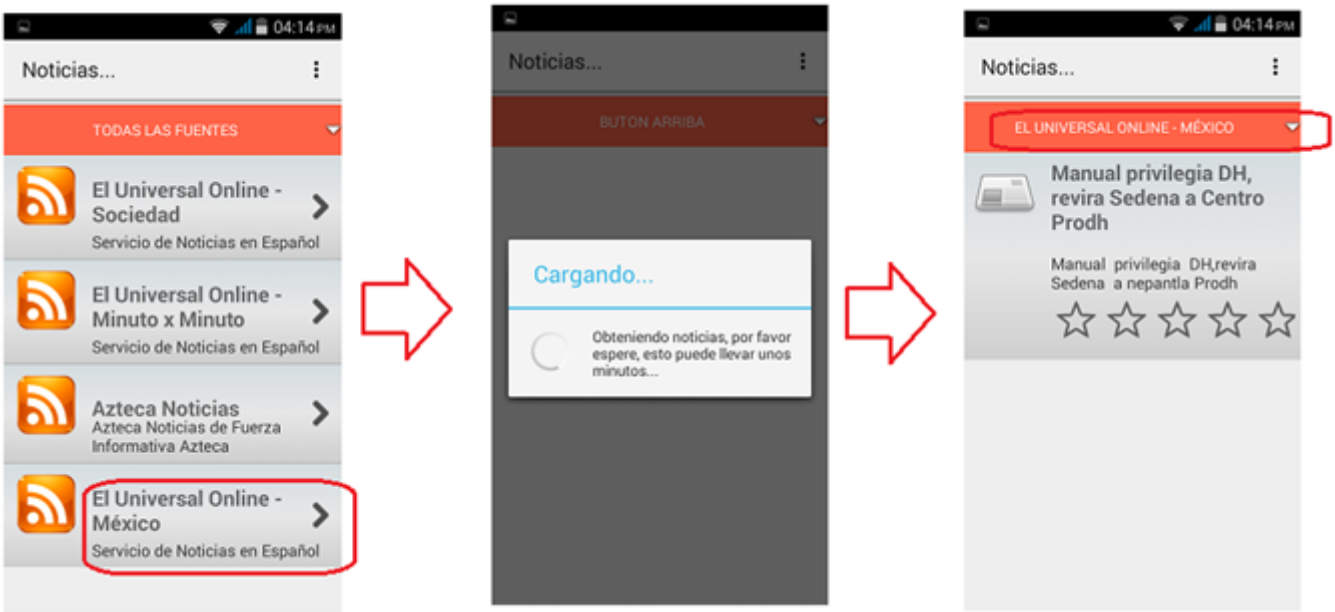


Figura 34: Prueba 4.4, seleccionar fuente

7.2.3. Seleccionar noticia

Prueba 5.1: Seleccionar Noticia

Se escoge una noticia y se muestra en una ventana la noticia completa, ver Figura 35.
Todas las pruebas siguientes se hicieron de la misma forma, lo único que cambia es la noticia seleccionada.

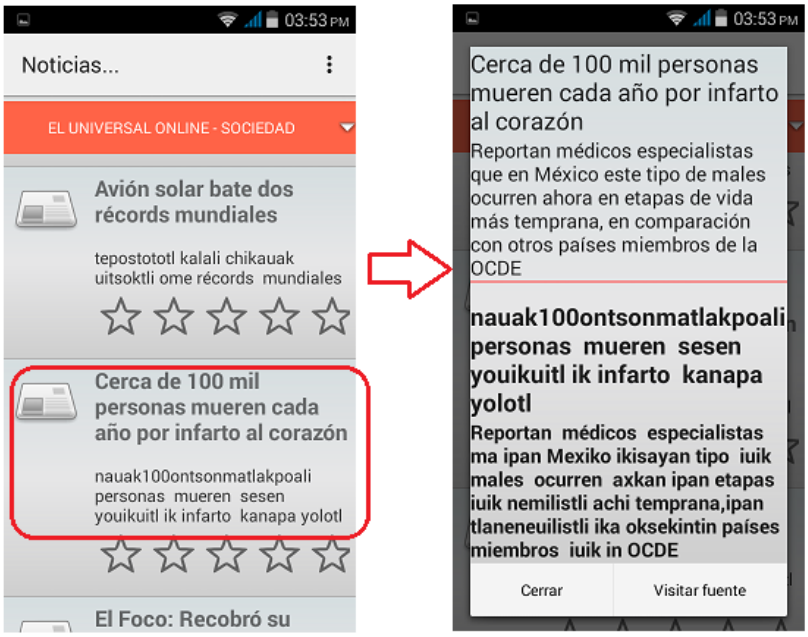


Figura 35: Prueba 5.1, seleccionar noticia

Prueba 5.2: Seleccionar Noticia

Se selecciona una noticia y se muestra completo, ver Figura 36.

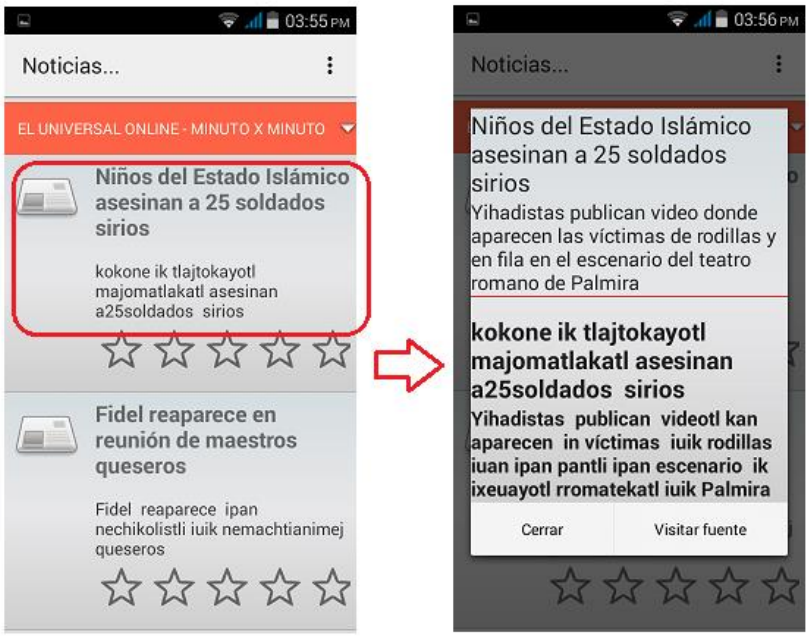


Figura 36: Prueba 5.2, seleccionar noticia

Prueba 5.3: Seleccionar Noticia

Se selecciona la noticia y se muestra el título de la noticia y la descripción seguido de la traducción del título y la descripción, ver Figura 37.

Figura 37: Prueba 5.3, seleccionar noticia

Prueba 5.4: Seleccionar Noticia
Se escoge la noticia y se muestra, ver Figura 38.

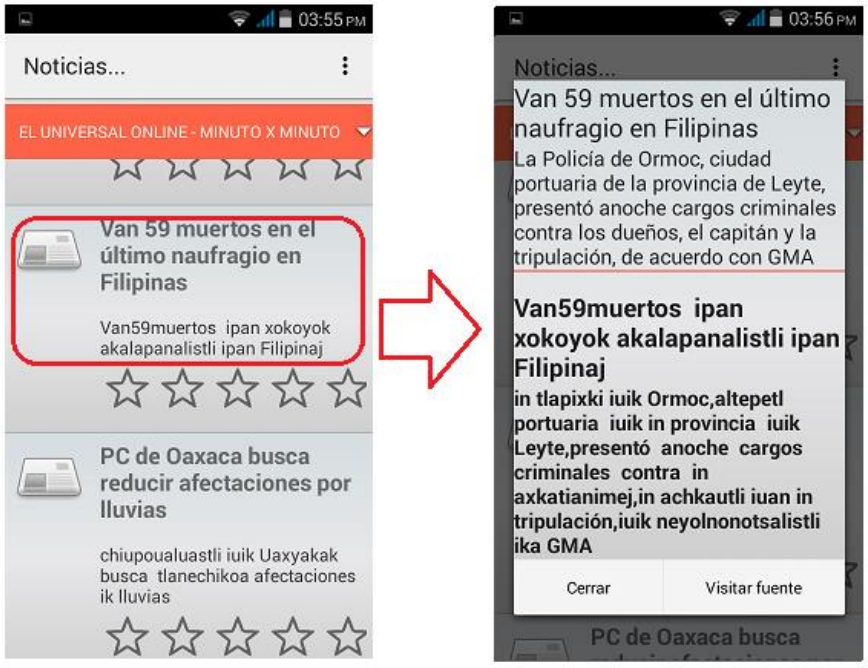


Figura 38: Prueba 5.4, seleccionar noticia

Prueba 5.5: Seleccionar Noticia

Se desea leer la noticia “PC de Oaxaca busca reducir afectaciones por lluvias” si pulsa con el dedo, a continuación se muestra el resultado en la Figura 39.

Figura 39: Prueba 5.5, seleccionar noticia

Prueba 5.6: Seleccionar Noticia

Se pulsa en la superficie de la noticia en el dispositivo móvil para visualizar la noticia, ver Figura 40.

Figura 40: Prueba 5.6, seleccionar noticia

Prueba 5.7: Seleccionar Noticia

Oprimir la noticia de interés en el listado vertical y la consecuencia es la visualización completa del ítem seleccionado.

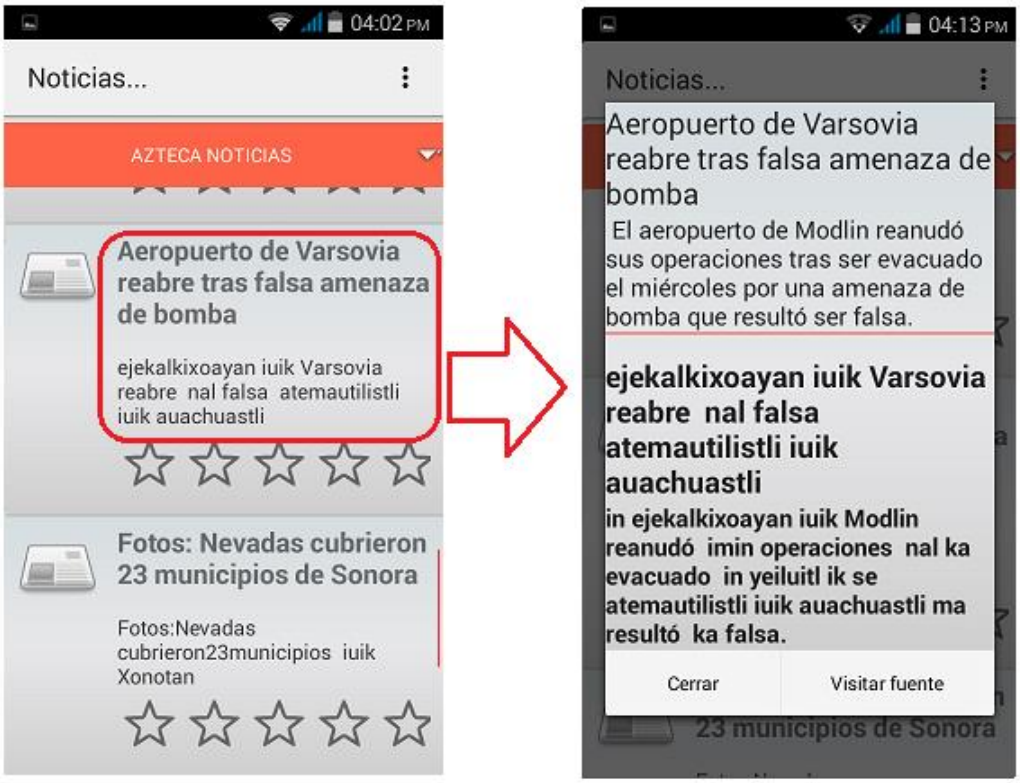


Figura 41: Prueba 5.7, seleccionar noticia

7.2.4. Calificar noticia

Prueba 6.1: Calificar noticia

Esta prueba fue posible gracias a que se ingresó a la aplicación con credenciales de un usuario hablante del español y náhuatl, al visualizar la noticia completa, en la parte inferior central se encuentran 5 estrellas y un poco más abajo la frase “Ayúdanos con una calificación”, al oprimir en la superficie de las estrellas se guarda una calificación y su valor depende a cual estrella se presionó por lo que el valor puede ser de 1 a 5 de izquierda a derecha. Las pruebas posteriores se hicieron del mismo modo, lo único diferente fue el valor de la calificación y con diferentes noticias, ver Figuras del 42 al 48.

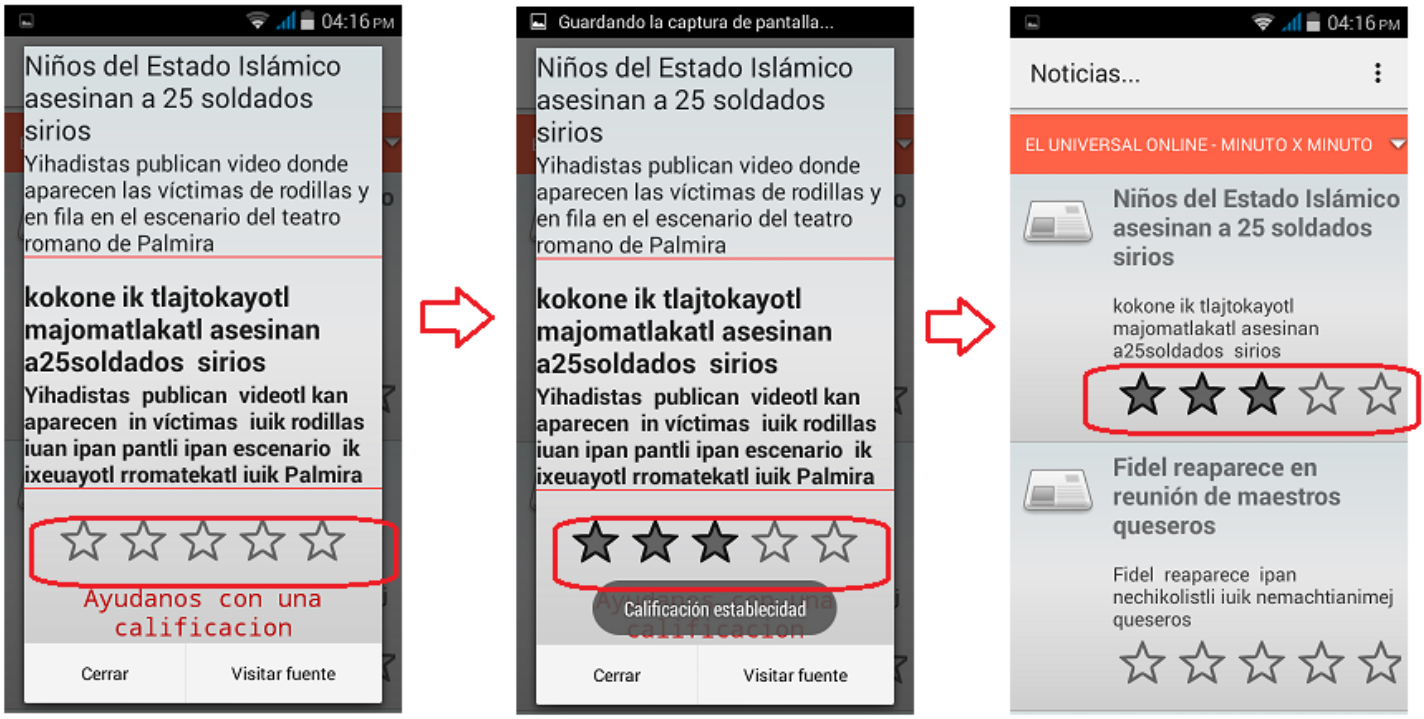


Figura 42: Prueba 6.1, calificar noticia

Prueba 6.2: Calificar noticia

Figura 43: Prueba 6.2, calificar noticia

Prueba 6.3: Calificar noticia

Figura 44: Prueba 6.3, calificar noticia

Prueba 6.4: Calificar noticia

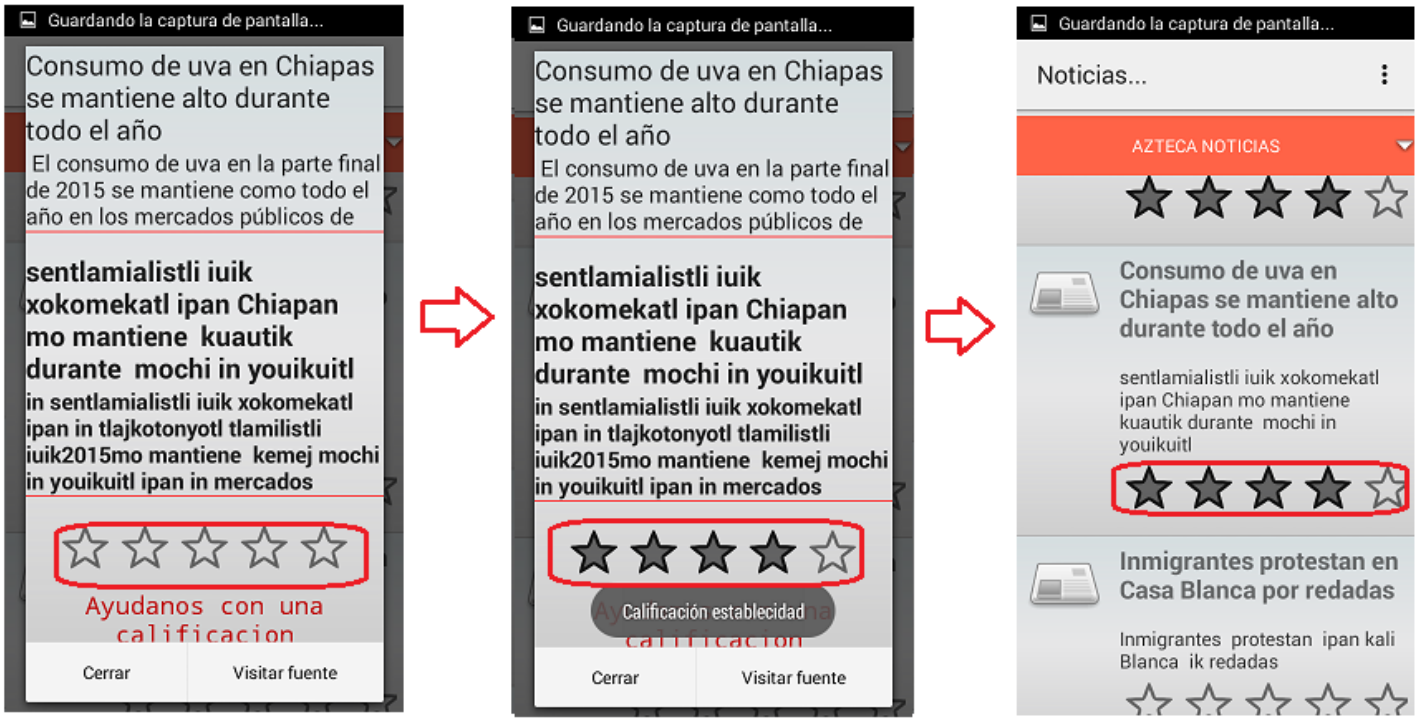


Figura 45: Prueba 6.4, calificar noticia

Prueba 6.5: Calificar noticia

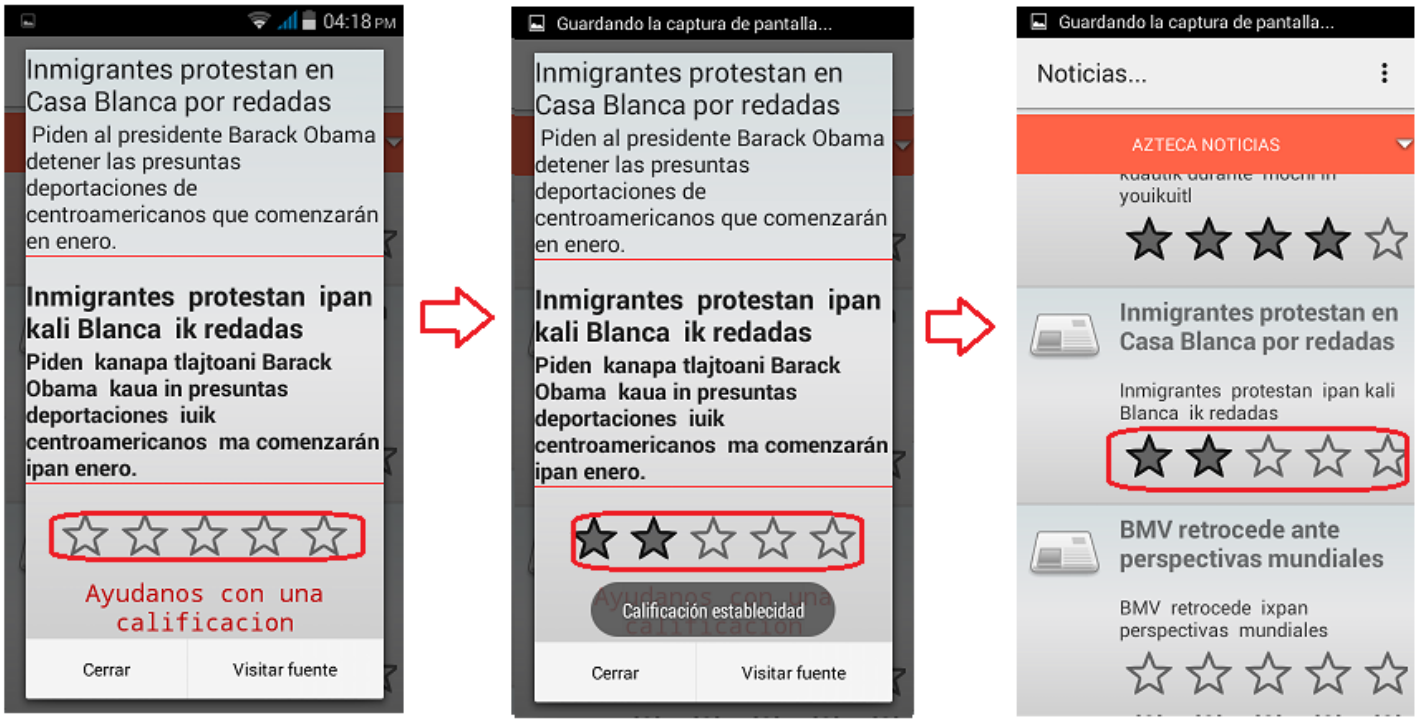


Figura 46: Prueba 6.5, calificar noticia

Prueba 6.6: Calificar noticia

Figura 47: Prueba 6.6, calificar noticia

Prueba 6.7: Calificar noticia

Figura 48: Prueba 6.7, calificar noticia

8. Análisis y discusión de resultados.

Las pruebas realizadas en el apartado anterior tuvieron la finalidad de mostrar las funcionalidades de cada caso de uso en la parte del cliente, lo que indirectamente significa también hacer pruebas de los casos de uso correspondientes al servidor porque la salida es un reflejo de lo que sucede en esa parte.

Comenzando con la gestión de usuarios que involucra dos casos de uso que son: Registrarse e Inicio de sesión, donde se hace uso de la base de datos que se encuentra almacenada en el servidor.

En el caso de uso registrarse, de las siete pruebas realizas, la información se almacenó en un 100%, ver Figura 49.

usuaId	usuaNombre	usuaAPaterno	usuaAMaterno	usuaUsername	usuaPassword	si_esp	si_nah
1	javier	solis	camilo	camilo	camilo	1	1
2	ingrid abigail	solano	aviles	solano	solano	1	0
3	isaura	mendez	santos	isaura	isaura	1	1
4	marco	petroni	carbajal	marco	marco	1	0
5	rocio	solis	camilo	rocio	rocio	0	1
6	bruno	diaz	vasquez	bruno	bruno	1	0
7	Ana cristina	segura	carbajal	anacristina	anacristina	1	0

Figura 49: Resultados en la base de datos de las pruebas del caso de uso registrarse

En el caso de uso inicio de sesión, en la primera prueba se ingresaron credenciales incorrectas y efectivamente el resultado fue lo que se esperaba. Posteriormente el resto de las pruebas fue con credenciales correctas y también se obtuvo el resultado esperado. Por lo que se demuestra que de las siete pruebas realizadas para este caso de uso fueron 100% satisfactorios. La explicación más sencilla de este resultado es porque se utilizó el FRAMEWORK HIBERNATE, una librería bien desarrollada y probada.

Siguiendo con la gestión de noticias.
En los listados de categorías y fuentes todas las pruebas salieron 100% exitosos, debido a que la dinámica es prácticamente igual, lo único que cambia es la tabla que se consulta en la base de datos.

La selección de fuentes fue el proceso más complejo debido a que en esta parte primero se recolectan las noticias de la fuente seleccionada, después se lleva a cabo la traducción y por último se regresa el resultado en un objeto con todas las noticias, esto significa que si una fuente contiene muchas noticias, el tiempo de espera es mucho mayor en el cliente Android, sin embargo, las pruebas realizadas fueron 100% exitosos.

En la selección de noticias la funcionalidad más simple por el hecho de que al momento de requerirse, la información ya se encuentra almacenada en la lista de noticias y lo que único que se hace es relacionarla con la vista que le corresponde por lo que en todas las pruebas se obtuvieron resultados 100% satisfactorios.

Por último, en la calificación de noticia, aunque es una tarea simple, en esta funcionalidad necesita hacer una comunicación con el servidor por el hecho de que el valor de puntos emitidos en el cliente Android se guarda en una base de datos que está en el servidor, sin embargo de las siete pruebas realizadas 100% fueron eficaces.

9. Conclusiones

El trabajo realizado en este proyecto es una herramienta de traducción automática de español al náhuatl, además se desarrolló un corpus paralelo de las dos lenguas mencionadas.

La traducción automática realizada en este proyecto basada en un corpus paralelo resulto medianamente satisfactorio debido a que no todas las palabras o frases fueron traducidas al náhuatl porque se necesita un corpus más extenso.

Los servicios web es una buena opción para proveer soluciones a varias plataformas utilizando el internet.

La aplicación móvil cumplió la función de mostrar los resultados a las personas interesadas en la noticias español – náhuatl, es sencilla y amigable para el usuario.

Durante el desarrollo se descubrió que hay varios aspectos que podrían mejorarse para este sistema de información. Por ejemplo, extender el corpus, optimizar el algoritmo de traducción porque el tiempo de espera en prolongado en el cliente. Otro de las características que pueden agregarse es la salida en audio de la traducción, así para una persona que solo habla náhuatl y además no sabe leer, esta característica le sería útil.

10.Bibliografía

[1] INEGI, «Instituto Nacional de Estadística y Geografía,» [En línea]. Available: http://cuentame.inegi.org.mx/poblacion/lindigena.aspx. [Último acceso: 09 12 2014].

[2] H. G. F. Javier, «Sistema gestor de información para la difusión de noticias de la UAM Azcapotzalco utilizando un canal RSS,» UAM, México, D. F., 2012.

[3] J. C. Castillo García, «Aplicación Android para la práctica de verbos compuestos del idioma inglés,» México, D. F., 2013.

[4] A. D. Mendoza, «SRCV: Sistema computacional interactivo basado en reconocimiento de comandos de voz para aplicaciones educativas,» México, 2013.

[5] J. S. Eutiquio, «Vitalidad y desplazamiento en el náhuatl de Rafael Delgado Veracruz,» México, 2012.

[6] Mozilla Foundation, «mozilla México,» [En línea]. Available: www.mozilla-mexico.org/category/lenguas/nahuatl. [Último acceso: 12 12 2014].

[7] Android, «Android,» 10 11 2015. [En línea]. Available: https://www.android.com/intl/es-419_mx/.

[8] Google Android, «Documentación Android,» 12 11 2015. [En línea]. Available: https://source.android.com/devices/tech/dalvik/.

[9] L. C. G. y. S. Ostos, «software de comunicaciones,» 15 11 2015. [En línea]. Available: https://sites.google.com/site/swcuc3m/home/android/generalidades/2-2-arquitectura-de-android.

[10] Wikipedia, «Traducción Automática,» 20 11 2015. [En línea]. Available: https://es.wikipedia.org/wiki/Traducci%C3%B3n_autom%C3%A1tica.

[11] U. d. Oxford, «British National Corpus,» 20 11 2015. [En línea]. Available: http://www.natcorp.ox.ac.uk/.

[12] Real Academia Española, «CREA,» 20 11 2015. [En línea]. Available: http://www.rae.es/recursos/banco-de-datos/crea.

[13] Real Academia Española, «CORDE,» 21 11 2015. [En línea]. Available: http://www.rae.es/recursos/banco-de-datos/corde.

[14] wikipedia, «Corpus lingüísticos,» 16 08 2015. [En línea]. Available: https://es.wikipedia.org/wiki/Corpus_ling%C3%BC%C3%ADsticos.

[15] ORACLE, «MYSQL,» 10 01 2015. [En línea]. Available: https://www.mysql.com.

[16] Hibernate, «hibernate,» 15 12 2015. [En línea]. Available: http://hibernate.org/orm/.

[17] phpmyadmin, «phpmyadmin,» 20 08 2015. [En línea]. Available: https://www.phpmyadmin.net/.

[18] oracle, «netbeans,» 15 01 2015. [En línea]. Available: https://netbeans.org/.

[19] Square, «Retrofit,» 02 02 2015. [En línea]. Available: http://square.github.io/retrofit/.

[20] Apache Tomcat, «Tomcat,» 10 02 2015. [En línea]. Available: http://tomcat.apache.org/.

[21] U. C. I. d. Madrid, «Software de Comunicaciones,» 20 02 2015. [En línea]. Available: https://sites.google.com/site/swcuc3m/home/android/generalidades/dalvikvm-1.

11. Apéndices

A. Código fuente

A.1. Código fuente del módulo diccionario

Archivo de conexión de HIBERNATE a MYSQL

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
"-//Hibernate/Hibernate Configuration DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
  <session-factory>
    <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
    <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
    <property
name="hibernate.connection.url">jdbc:mysql://localhost:3306/corpus_v_4?zeroDateTimeBehavi
or=convertToNull</property>
    <property name="hibernate.connection.username">root</property>
    <property name="hibernate.connection.password">4dm1n..*</property>
    <property
name="hibernate.query.factory_class">org.hibernate.hql.internal.classic.ClassicQueryTrans
latorFactory</property>
    <mapping resource="hibernate/modelo/Noticia.hbm.xml"/>
    <mapping resource="hibernate/modelo/Calificacion.hbm.xml"/>
    <mapping resource="hibernate/modelo/Usuario.hbm.xml"/>
    <mapping resource="hibernate/modelo/Corpus.hbm.xml"/>
    <mapping resource="hibernate/modelo/UsuarioCategoria.hbm.xml"/>
    <mapping resource="hibernate/modelo/Fuente.hbm.xml"/>
    <mapping resource="hibernate/modelo/UsuarioFuente.hbm.xml"/>
    <mapping resource="hibernate/modelo/Categoria.hbm.xml"/>
    <mapping resource="hibernate/modelo/Estadistica.hbm.xml"/>
  </session-factory>
</hibernate-configuration>
```

Archivo de ingeniería inversa para generar POJOs (acrónimo de Plain Old Java Object) de las tablas de la base de datos

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-reverse-engineering PUBLIC "-//Hibernate/Hibernate Reverse
Engineering DTD 3.0//EN" "http://hibernate.sourceforge.net/hibernate-reverse-engineering-
3.0.dtd">
<hibernate-reverse-engineering>
  <schema-selection match-catalog="corpus_v_4"/>
  <table-filter match-name="fuente"/>
  <table-filter match-name="usuario"/>
  <table-filter match-name="corpus"/>
  <table-filter match-name="categoria"/>
  <table-filter match-name="noticia"/>
  <table-filter match-name="usuario_fuente"/>
  <table-filter match-name="estadistica"/>
  <table-filter match-name="calificacion"/>
  <table-filter match-name="usuario_categoria"/>
</hibernate-reverse-engineering>
```

Mapeo de la tabla usuario

XML

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
<hibernate-mapping>
```

```
    <class name="hibernate.modelo.Usuario" table="usuario" catalog="corpus_v_4"
optimistic-lock="version">
        <id name="usuaId" type="java.lang.Integer">
            <column name="usuaId" />
            <generator class="identity" />
        </id>
        <property name="usuaNombre" type="string">
            <column name="usuaNombre" length="45" not-null="true" />
        </property>
        <property name="usuaApaterno" type="string">
            <column name="usuaAPaterno" length="45" not-null="true" />
        </property>
        <property name="usuaAmaterno" type="string">
            <column name="usuaAMaterno" length="45" not-null="true" />
        </property>
        <property name="usuaUsername" type="string">
            <column name="usuaUsername" length="45" not-null="true" />
        </property>
        <property name="usuaPassword" type="string">
            <column name="usuaPassword" length="45" not-null="true" />
        </property>
        <property name="siEsp" type="boolean">
            <column name="si_esp" not-null="true" />
        </property>
        <property name="siNah" type="boolean">
            <column name="si_nah" not-null="true" />
        </property>
    </class>
</hibernate-mapping>
```

POJO Usuario

```
package hibernate.modelo;

public class Usuario  implements java.io.Serializable {
    private Integer usuaId;
     private String usuaNombre;
     private String usuaApaterno;
     private String usuaAmaterno;
     private String usuaUsername;
     private String usuaPassword;
     private boolean siEsp;
     private boolean siNah;

    public Usuario() {}
    public Usuario(String usuaNombre, String usuaApaterno, String usuaAmaterno, String
usuaUsername, String usuaPassword, boolean siEsp, boolean siNah) {
        this.usuaNombre = usuaNombre;
        this.usuaApaterno = usuaApaterno;
        this.usuaAmaterno = usuaAmaterno;
        this.usuaUsername = usuaUsername;
        this.usuaPassword = usuaPassword;
        this.siEsp = siEsp;
        this.siNah = siNah;
    }
    public Integer getUsuaId() {
        return this.usuaId;
    }
    public void setUsuaId(Integer usuaId) {
        this.usuaId = usuaId;
    }
    public String getUsuaNombre() {
```

```
        return this.usuaNombre;
    }
    public void setUsuaNombre(String usuaNombre) {
        this.usuaNombre = usuaNombre;
    }
    public String getUsuaApaterno() {
        return this.usuaApaterno;
    }
    public void setUsuaApaterno(String usuaApaterno) {
        this.usuaApaterno = usuaApaterno;
    }
    public String getUsuaAmaterno() {
        return this.usuaAmaterno;
    }
    public void setUsuaAmaterno(String usuaAmaterno) {
        this.usuaAmaterno = usuaAmaterno;
    }
    public String getUsuaUsername() {
        return this.usuaUsername;
    }
    public void setUsuaUsername(String usuaUsername) {
        this.usuaUsername = usuaUsername;
    }
    public String getUsuaPassword() {
        return this.usuaPassword;
    }
    public void setUsuaPassword(String usuaPassword) {
        this.usuaPassword = usuaPassword;
    }
    public boolean isSiEsp() {
        return this.siEsp;
    }
    public void setSiEsp(boolean siEsp) {
        this.siEsp = siEsp;
    }
    public boolean isSiNah() {
        return this.siNah;
    }
    public void setSiNah(boolean siNah) {
        this.siNah = siNah;
    }
}
```

Mapeo de la tabla categoría

XML

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
<hibernate-mapping>
    <class name="hibernate.modelo.Categoria" table="categoria" catalog="corpus_v_4"
optimistic-lock="version">
        <id name="cateId" type="java.lang.Integer">
            <column name="cateId" />
            <generator class="identity" />
        </id>
        <property name="cateNombre" type="string">
            <column name="cateNombre" length="45" />
        </property>
        <property name="cateDescripcion" type="string">
            <column name="cateDescripcion" />
        </property>
```

```
    </class>
</hibernate-mapping>
```

POJO Categoría

```
package hibernate.modelo;

public class Categoria  implements java.io.Serializable {
    private Integer cateId;
    private String cateNombre;
    private String cateDescripcion;

    public Categoria() { }
    public Categoria(String cateNombre, String cateDescripcion) {
       this.cateNombre = cateNombre;
       this.cateDescripcion = cateDescripcion;
    }
    public Integer getCateId() {
        return this.cateId;
    }
    public void setCateId(Integer cateId) {
        this.cateId = cateId;
    }
    public String getCateNombre() {
        return this.cateNombre;
    }
    public void setCateNombre(String cateNombre) {
        this.cateNombre = cateNombre;
    }
    public String getCateDescripcion() {
        return this.cateDescripcion;
    }
    public void setCateDescripcion(String cateDescripcion) {
        this.cateDescripcion = cateDescripcion;
    }
}
```

Mapeo de la tabla fuente

XML

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
<hibernate-mapping>
    <class name="hibernate.modelo.Fuente" table="fuente" catalog="corpus_v_4" optimistic-
lock="version">
        <id name="fuentId" type="java.lang.Integer">
            <column name="fuentId" />
            <generator class="identity" />
        </id>
        <property name="fuentTitulo" type="string">
            <column name="fuentTitulo" length="45" />
        </property>
        <property name="fuentLink" type="string">
            <column name="fuentLink" length="65535" />
        </property>
        <property name="fuentDescripcion" type="string">
            <column name="fuentDescripcion" />
        </property>
        <property name="fuentCopyright" type="string">
            <column name="fuentCopyright" length="45" />
        </property>
        <property name="fuentPubDate" type="string">
```

```
            <column name="fuentPubDate" length="45" />
        </property>
        <property name="categoriaId" type="int">
            <column name="categoriaId" not-null="true" />
        </property>
    </class>
</hibernate-mapping>
```

POJO Fuente

```
package hibernate.modelo;

public class Fuente  implements java.io.Serializable {
    private Integer fuentId;
    private String fuentTitulo;
    private String fuentLink;
    private String fuentDescripcion;
    private String fuentCopyright;
    private String fuentPubDate;
    private int categoriaId;

    public Fuente() {}
    public Fuente(int categoriaId) {
        this.categoriaId = categoriaId;
    }
    public Fuente(String fuentTitulo, String fuentLink, String fuentDescripcion, String
fuentCopyright, String fuentPubDate, int categoriaId) {
       this.fuentTitulo = fuentTitulo;
       this.fuentLink = fuentLink;
       this.fuentDescripcion = fuentDescripcion;
       this.fuentCopyright = fuentCopyright;
       this.fuentPubDate = fuentPubDate;
       this.categoriaId = categoriaId;
    }
    public Integer getFuentId() {
        return this.fuentId;
    }
    public void setFuentId(Integer fuentId) {
        this.fuentId = fuentId;
    }
    public String getFuentTitulo() {
        return this.fuentTitulo;
    }
    public void setFuentTitulo(String fuentTitulo) {
        this.fuentTitulo = fuentTitulo;
    }
    public String getFuentLink() {
        return this.fuentLink;
    }
    public void setFuentLink(String fuentLink) {
        this.fuentLink = fuentLink;
    }
    public String getFuentDescripcion() {
        return this.fuentDescripcion;
    }
    public void setFuentDescripcion(String fuentDescripcion) {
        this.fuentDescripcion = fuentDescripcion;
    }
    public String getFuentCopyright() {
        return this.fuentCopyright;
    }
    public void setFuentCopyright(String fuentCopyright) {
```

```
        this.fuentCopyright = fuentCopyright;
    }
    public String getFuentPubDate() {
        return this.fuentPubDate;
    }
    public void setFuentPubDate(String fuentPubDate) {
        this.fuentPubDate = fuentPubDate;
    }
    public int getCategoriaId() {
        return this.categoriaId;
    }
    public void setCategoriaId(int categoriaId) {
        this.categoriaId = categoriaId;
    }
}
```

Mapeo de la tabla usuario_categoría

XML

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
<hibernate-mapping>
    <class name="hibernate.modelo.UsuarioCategoria" table="usuario_categoria"
catalog="corpus_v_4" optimistic-lock="version">
        <id name="uscaId" type="java.lang.Integer">
            <column name="uscaId" />
            <generator class="identity" />
        </id>
        <property name="usuarioId" type="java.lang.Integer">
            <column name="usuarioId" />
        </property>
        <property name="categoriaId" type="java.lang.Integer">
            <column name="categoriaId" />
        </property>
    </class>
</hibernate-mapping>
```

POJO UsuarioCategoria

```
package hibernate.modelo;

public class UsuarioCategoria  implements java.io.Serializable {
    private Integer uscaId;
    private Integer usuarioId;
    private Integer categoriaId;

    public UsuarioCategoria() {}
    public UsuarioCategoria(Integer usuarioId, Integer categoriaId) {
       this.usuarioId = usuarioId;
       this.categoriaId = categoriaId;
    }
    public Integer getUscaId() {
        return this.uscaId;
    }
    public void setUscaId(Integer uscaId) {
        this.uscaId = uscaId;
    }
    public Integer getUsuarioId() {
        return this.usuarioId;
    }
    public void setUsuarioId(Integer usuarioId) {
        this.usuarioId = usuarioId;
```

```
    }
    public Integer getCategoriaId() {
        return this.categoriaId;
    }
    public void setCategoriaId(Integer categoriaId) {
        this.categoriaId = categoriaId;
    }
}
```

Mapeo de la tabla usuario_fuente

XML

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
<hibernate-mapping>
    <class name="hibernate.modelo.UsuarioFuente" table="usuario_fuente"
catalog="corpus_v_4" optimistic-lock="version">
        <id name="usfuId" type="java.lang.Integer">
            <column name="usfuId" />
            <generator class="identity" />
        </id>
        <property name="usuarioId" type="java.lang.Integer">
            <column name="usuarioId" />
        </property>
        <property name="fuenteId" type="java.lang.Integer">
            <column name="fuenteId" />
        </property>
    </class>
</hibernate-mapping>
```

POJO UsuarioCategoria

```
package hibernate.modelo;

public class UsuarioFuente  implements java.io.Serializable {
    private Integer usfuId;
    private Integer usuarioId;
    private Integer fuenteId;

    public UsuarioFuente() { }
    public UsuarioFuente(Integer usuarioId, Integer fuenteId) {
       this.usuarioId = usuarioId;
       this.fuenteId = fuenteId;
    }
    public Integer getUsfuId() {
        return this.usfuId;
    }
    public void setUsfuId(Integer usfuId) {
        this.usfuId = usfuId;
    }
    public Integer getUsuarioId() {
        return this.usuarioId;
    }
    public void setUsuarioId(Integer usuarioId) {
        this.usuarioId = usuarioId;
    }
    public Integer getFuenteId() {
        return this.fuenteId;
    }
    public void setFuenteId(Integer fuenteId) {
        this.fuenteId = fuenteId;
    }
```

```
}
```

Mapeo de la tabla corpus

XML

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
<hibernate-mapping>
    <class name="hibernate.modelo.Corpus" table="corpus" catalog="corpus_v_4" optimistic-
lock="version">
        <id name="corpId" type="java.lang.Integer">
            <column name="corpId" />
            <generator class="identity" />
        </id>
        <property name="corpEsp" type="string">
            <column name="corpEsp" not-null="true" />
        </property>
        <property name="corpNah" type="string">
            <column name="corpNah" not-null="true" />
        </property>
    </class>
</hibernate-mapping>
```

POJO Corpus

```
package hibernate.modelo;

public class Corpus  implements java.io.Serializable {
    private Integer corpId;
    private String corpEsp;
    private String corpNah;

    public Corpus() {}
    public Corpus(String corpEsp, String corpNah) {
       this.corpEsp = corpEsp;
       this.corpNah = corpNah;
    }
    public Integer getCorpId() {
        return this.corpId;
    }
    public void setCorpId(Integer corpId) {
        this.corpId = corpId;
    }
    public String getCorpEsp() {
        return this.corpEsp;
    }
    public void setCorpEsp(String corpEsp) {
        this.corpEsp = corpEsp;
    }
    public String getCorpNah() {
        return this.corpNah;
    }
    public void setCorpNah(String corpNah) {
        this.corpNah = corpNah;
    }
}
```

Mapeo de la tabla noticia

XML

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
<hibernate-mapping>
```

```
    <class name="hibernate.modelo.Noticia" table="noticia" catalog="corpus_v_4"
optimistic-lock="version">
        <id name="notiId" type="java.lang.Integer">
            <column name="notiId" />
            <generator class="identity" />
        </id>
        <property name="notiLink" type="string">
            <column name="notiLink" length="65535" />
        </property>
        <property name="notiAutor" type="string">
            <column name="notiAutor" length="100" />
        </property>
        <property name="notiPubDate" type="string">
            <column name="notiPubDate" length="45" />
        </property>
        <property name="fuenteId" type="java.lang.Integer">
            <column name="fuenteId" />
        </property>
        <property name="tituloId" type="java.lang.Integer">
            <column name="tituloId" />
        </property>
        <property name="descripcionId" type="java.lang.Integer">
            <column name="descripcionId" />
        </property>
    </class>
</hibernate-mapping>
```

POJO Noticia

```
package hibernate.modelo;

public class Noticia  implements java.io.Serializable {
    private Integer notiId;
    private String notiLink;
    private String notiAutor;
    private String notiPubDate;
    private Integer fuenteId;
    private Integer tituloId;
    private Integer descripcionId;

    public Noticia() {    }
    public Noticia(String notiLink, String notiAutor,  Integer fuenteId, Integer
tituloId, Integer descripcionId,String notiPubDate) {
       this.notiLink = notiLink;
       this.notiAutor = notiAutor;
       this.notiPubDate = notiPubDate;
       this.fuenteId = fuenteId;
       this.tituloId = tituloId;
       this.descripcionId = descripcionId;
    }
    public Integer getNotiId() {
        return this.notiId;
    }
    public void setNotiId(Integer notiId) {
        this.notiId = notiId;
    }
    public String getNotiLink() {
        return this.notiLink;
    }
    public void setNotiLink(String notiLink) {
        this.notiLink = notiLink;
    }
```

```
    public String getNotiAutor() {
        return this.notiAutor;
    }
    public void setNotiAutor(String notiAutor) {
        this.notiAutor = notiAutor;
    }
    public String getNotiPubDate() {
        return this.notiPubDate;
    }
    public void setNotiPubDate(String notiPubDate) {
        this.notiPubDate = notiPubDate;
    }
    public Integer getFuenteId() {
        return this.fuenteId;
    }
    public void setFuenteId(Integer fuenteId) {
        this.fuenteId = fuenteId;
    }
    public Integer getTituloId() {
        return this.tituloId;
    }
    public void setTituloId(Integer tituloId) {
        this.tituloId = tituloId;
    }
    public Integer getDescripcionId() {
        return this.descripcionId;
    }
    public void setDescripcionId(Integer descripcionId) {
        this.descripcionId = descripcionId;
    }
}
```

Mapeo de la tabla calificación

XML

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
<hibernate-mapping>
    <class name="hibernate.modelo.Calificacion" table="calificacion" catalog="corpus_v_4"
optimistic-lock="version">
        <id name="caliId" type="java.lang.Integer">
            <column name="caliId" />
            <generator class="identity" />
        </id>
        <property name="caliIdNoticia" type="int">
            <column name="caliIdNoticia" not-null="true" />
        </property>
        <property name="caliIdUsuario" type="int">
            <column name="caliIdUsuario" not-null="true" />
        </property>
        <property name="caliValor" type="int">
            <column name="caliValor" not-null="true" />
        </property>
    </class>
</hibernate-mapping>
```

POJO Calificacion

```
package hibernate.modelo;

public class Calificacion  implements java.io.Serializable {
    private Integer caliId;
```

```
    private int caliIdNoticia;
    private int caliIdUsuario;
    private int caliValor;

    public Calificacion() {    }
    public Calificacion(int caliIdNoticia, int caliIdUsuario, int caliValor) {
       this.caliIdNoticia = caliIdNoticia;
       this.caliIdUsuario = caliIdUsuario;
       this.caliValor = caliValor;
    }
    public Integer getCaliId() {
        return this.caliId;
    }
    public void setCaliId(Integer caliId) {
        this.caliId = caliId;
    }
    public int getCaliIdNoticia() {
        return this.caliIdNoticia;
    }
    public void setCaliIdNoticia(int caliIdNoticia) {
        this.caliIdNoticia = caliIdNoticia;
    }
    public int getCaliIdUsuario() {
        return this.caliIdUsuario;
    }
    public void setCaliIdUsuario(int caliIdUsuario) {
        this.caliIdUsuario = caliIdUsuario;
    }
    public int getCaliValor() {
        return this.caliValor;
    }
    public void setCaliValor(int caliValor) {
        this.caliValor = caliValor;
    }
    public String toString(){
        return "id: " +caliId.toString()+" idN: "+Integer.toString(caliIdNoticia)+" idU:
"+Integer.toString(caliIdUsuario)+" valor: "+Integer.toString(caliValor);
    }
}
```

Objetos de acceso a datos (DAOs)

Clase interfaz de DAO genérico

```
package hibernate.dao;

import java.io.Serializable;
import java.util.List;

public interface GenericDao <T, PK extends Serializable> {
    PK add(T newInstance);
    T get(PK id);
    void update(T transientObject);
    void delete(T persistentObject);
    List<T> findBy(String[] args);
}
```

Implementación del DAO genérico

```
package hibernate.dao;

import java.io.Serializable;
import java.lang.reflect.ParameterizedType;
import java.util.List;
```

```
import org.hibernate.Query;
import org.hibernate.Session;

public abstract class GenericDaoImpl <T, PK extends Serializable> implements GenericDao
<T, PK> {
    private final Class<T> type;
    protected final Session session;

    public Class<T> getType() {
        return type;
    }
    public GenericDaoImpl(Session session) {
        this.type = (Class<T>) ((ParameterizedType)
getClass().getGenericSuperclass()).getActualTypeArguments()[0];
        this.session = session;
    }
    @Override
    public PK add(T newInstance) {
        return (PK) session.save(newInstance);
    }
    @Override
    public T get(PK id) {
        return (T) session.get(getType(), id);
    }
    @Override
    public void update(T transientObject) {
        session.update(transientObject);
    }
    @Override
    public void delete(T persistentObject) {
        session.delete(persistentObject);
    }
    @Override
    public List<T> findBy(String[] args){
        Query query=session.createQuery(args[0]);
        for (int i=1,l=args.length-1;i<l;i=i+2) {
            query.setParameter(args[i],args[i+1]);
        }
        return query.list();
    }
}
```

Usuario DAO

```
package hibernate.dao;

import java.util.List;
import org.hibernate.Session;
import hibernate.modelo.Usuario;

public class UsuarioDao extends GenericDaoImpl<Usuario, Integer> {
    public UsuarioDao(Session session) {
        super(session);
    }
    public List<Usuario> getTodo() {
        String hql = "FROM Usuario";
        return session.createQuery(hql).list();
    }
    public Usuario loggin(String name,String pass) {
        Usuario i;
        String hql = "FROM Usuario WHERE usuaUsername = :name AND usuaPassword = :pass";
        List l=super.findBy( new String[]{hql,"name",name,"pass",pass});
```

```
        i = (l.size() > 0) ? (Usuario)l.get(0) : null;
        return i;
    }
}
```

Categoría DAO

```
package hibernate.dao;

import java.util.List;
import org.hibernate.Session;
import hibernate.modelo.*;

public class CategoriaDao extends GenericDaoImpl<Categoria, Integer> {
    public CategoriaDao(Session session) {
        super(session);
    }
    public List<Categoria> getTodo() {
        String hql = "FROM Categoria";
        return session.createQuery(hql).list();
    }
}
```

Fuente DAO

```
package hibernate.dao;

import java.util.List;
import org.hibernate.Session;
import hibernate.modelo.*;

public class FuenteDao extends GenericDaoImpl<Fuente, Integer> {
    public FuenteDao(Session session) {
        super(session);
    }
    public List<Fuente> getTodo() {
         String hql = "FROM Fuente";
        return session.createQuery(hql).list();
    }
    public boolean existeByLink(String link){
        String hql="From Fuente WHERE fuentLink= :link";
        String[] query=new String[]{hql,"link",link};
        List<Fuente> fuentes=findBy(query);
        return !fuentes.isEmpty();
    }
}
```

Usuario_categoria DAO

```
package hibernate.dao;

import hibernate.modelo.UsuarioCategoria;
import java.util.List;
import org.hibernate.Session;

public class UsuarioCategoriaDao extends GenericDaoImpl<UsuarioCategoria, Integer> {
    public UsuarioCategoriaDao(Session session) {
        super(session);
    }
    public boolean existe(int us,int cate){
        String hql="From UsuarioCategoria WHERE usuarioId="+us+" and categoriaId="+cate;
        String[] query=new String[]{hql};
        List<UsuarioCategoria> uscas=findBy(query);
        return !uscas.isEmpty();
    }
```

```
}
```

Usuario_fuente DAO

```
package hibernate.dao;

import hibernate.modelo.UsuarioFuente;
import org.hibernate.Session;

public class UsuarioFuenteDao extends GenericDaoImpl<UsuarioFuente, Integer> {
    public UsuarioFuenteDao(Session session) {
        super(session);
    }
}
```

Corpus DAO

```
package hibernate.dao;

import java.util.List;
import org.hibernate.Session;
import hibernate.modelo.Corpus;

public class CorpusDao extends GenericDaoImpl<Corpus, Integer> {
    public CorpusDao(Session session) {
        super(session);
    }
    public List<Corpus> getTodo() {
        String hql = "FROM Corpus";
        return session.createQuery(hql).list();
    }
    public List<Corpus> getByEsp(String esp) {
        esp=esp.trim();
        String[] query = new String[]{"FROM Corpus WHERE corpEsp = :esp", "esp",esp};
        return findBy(query);
    }
}
```

Noticia DAO

```
package hibernate.dao;

import java.util.List;
import org.hibernate.Session;
import hibernate.modelo.Noticia;

public class NoticiaDao extends GenericDaoImpl<Noticia, Integer> {
    public NoticiaDao(Session session) {
        super(session);
    }
    public List<Noticia> getTodo() {
        String hql = "FROM Noticia";
        return session.createQuery(hql).list();
    }
}
```

Calificación DAO

```
package hibernate.dao;

import hibernate.modelo.Calificacion;
import java.util.List;
import org.hibernate.Session;

public class CalificacionDao extends GenericDaoImpl<Calificacion, Integer> {
    public CalificacionDao(Session session) {
        super(session);
```

```
    }
    public List<Calificacion> getTodo() {
        String hql = "FROM Calificacion";
        return session.createQuery(hql).list();
    }
    public int getPromedio(int noti,int usua) {
        String hql="FROM Calificacion "
                + "WHERE caliIdNoticia ="
                +Integer.toString(noti)+" "
                + "AND caliIdUsuario="
                +Integer.toString(usua);
        String[] query=new String[]{hql};

        return getPromedio(findBy(query));
    }
    public int getPromedio(int noti) {
        String hql="FROM Calificacion "
                + "WHERE caliIdNoticia ="
                +Integer.toString(noti);
        String[] query=new String[]{hql};

        return getPromedio(findBy(query));
    }
    public int getPromedio(List<Calificacion> lcali){
        int prom=0,total=0;
        for(Calificacion cali: lcali){
            System.out.println("calificacion >>"+cali);
            prom+=cali.getCaliValor();
            total++;
        }
        if(total==0){
            prom=0;
        }else{
            prom=prom/total;
        }
        return prom;
    }
    public int getPromedio(Calificacion cali) {
        return getPromedio(cali.getCaliIdNoticia(),cali.getCaliIdUsuario());
    }
}
```

Para la conexión a mysql con HIBERNATE

```
package hibernate.ctr;

import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;
import org.hibernate.service.ServiceRegistryBuilder;

public class HibernateUtil {
    private static final SessionFactory sessionFactory;

    static {
        try {
            sessionFactory = new Configuration()
            .configure()
            .buildSessionFactory();
        } catch (Throwable ex) {
            System.err.println("Error HibernateUtil :(" + ex);
            throw new ExceptionInInitializerError(ex);
        }
```

```
    }
    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }
}
```

Clase intermedia entre el API REST y la base de datos

```
package hibernate.ctr;

import entidadRelacion.NoticiaER;
import entidadRelacion.FuenteER;
import entidadRelacion.CategoriaER;
import org.hibernate.HibernateException;
import org.hibernate.Session;
import hibernate.dao.*;
import hibernate.modelo.*;
import java.util.ArrayList;
import java.util.List;

public class DB {
    private final Session session;
    public DB(){
        session = HibernateUtil.getSessionFactory().openSession();
    }
    public CategoriaDao getCateDao(){
        return (new CategoriaDao(session));
    }
    public CorpusDao daoCorp(){
        return (new CorpusDao(session));
    }
    public FuenteDao daoFuen(){
        return (new FuenteDao(session));
    }
    public NoticiaDao daoNoti(){
        return (new NoticiaDao(session));
    }
    public UsuarioDao daoUsua(){
        return (new UsuarioDao(session));
    }
    public UsuarioCategoriaDao daoUsca(){
        return (new UsuarioCategoriaDao(session));
    }
    public UsuarioFuenteDao daoUsfu(){
        return (new UsuarioFuenteDao(session));
    }
    public EstadisticaDao daoEsta(){
        return (new EstadisticaDao(session));
    }
    public CalificacionDao daoCali(){
        return (new CalificacionDao(session));
    }
    public Categoria getCategoriaById(Integer id) throws Exception{
        try{
            return (new CategoriaDao(session)).get(id);
        } catch (HibernateException e) {
            System.out.println("ErrorHibernate getCategoriaById: " + e);
            throw new Exception("ErrorHibernate getCategoriaById: " + e);
        }
    }
    public CategoriaER getCategoriaERById(Integer id) throws Exception{
        Categoria cate =(new CategoriaDao(session)).get(id);
```

```
        String hql="Select From Fuente WHERE categoriaId=:id";
        String[] query=new String[]{hql,"id",cate.getCateId().toString()};
        List<Fuente> fuentes=getFuentesBy(query);
        return new CategoriaER(cate,fuentes);
    }
    public Corpus getCorpusById(Integer id)throws Exception{
        try{
            return (new CorpusDao(session)).get(id);
        } catch (HibernateException e) {
            System.out.println("ErrorHibernate getCorpusById: " + e);
            throw new Exception("ErrorHibernate getCorpusById: " + e);
        }
    }
    public Noticia getNoticiaById(Integer id)throws Exception{
        try{
            return (new NoticiaDao(session)).get(id);
        } catch (HibernateException e) {
            System.out.println("ErrorHibernate getNoticiaById: " + e);
            throw new Exception("ErrorHibernate getNoticiaById: " + e);
        }
    }
    public NoticiaER getNoticiaERById(Integer id)throws Exception{
        Noticia noticia=getNoticiaById(id);
        Corpus corpTitulo=getCorpusById(noticia.getTituloId());
        Corpus corpDescripcion=getCorpusById(noticia.getDescripcionId());

        return new
NoticiaER(noticia,corpTitulo,corpDescripcion,daoCali().getPromedio(id));
    }
    public Fuente getFuenteById(Integer id)throws Exception{
        try{
            return (new FuenteDao(session)).get(id);
        } catch (HibernateException e) {
            System.out.println("ErrorHibernate getFuenteById: " + e);
            throw new Exception("ErrorHibernate getFuenteById: " + e);
        }
    }
    public FuenteER getFuenteERById(Integer id)throws Exception{
        Fuente fuente=getFuenteById(id);
        Categoria cate=getCategoriaById(fuente.getCategoriaId());
        String hql="Select From Noticia WHERE fuenteId=:id";
        String[] query=new String[]{hql,"id",fuente.getFuentId().toString()};
        List<NoticiaER> lner=getNoticiaERsBy(query);
        return new FuenteER(fuente,cate,lner);
    }
    public Usuario getUsuarioById(Integer id)throws Exception{
        try{
            return (new UsuarioDao(session)).get(id);
        } catch (HibernateException e) {
            System.out.println("ErrorHibernate getUsuarioById: " + e);
            throw new Exception("ErrorHibernate getUsuarioById: " + e);
        }
    }
    public List<Categoria> getCategorias()throws Exception {
        try {
            return (new CategoriaDao(session)).getTodo();
        } catch (HibernateException e) {
            System.out.println("ErrorHibernate getCategoriaAll: " + e);
            throw new Exception("ErrorHibernate getCategoriaAll: " + e);
        }
```

```
    }
    public List<Categoria> getCategoriasBy(String[] query) throws Exception {
        try {
            return (new CategoriaDao(session)).findBy(query);
        } catch (HibernateException e) {
            System.out.println("ErrorHibernate getCategoriaBy: " + e);
            throw new Exception("ErrorHibernate getCategoriaBy: " + e);
        }
    }
    public List<Corpus> getCorpus()throws Exception {
        try {
            return (new CorpusDao(session)).getTodo();
        } catch (HibernateException e) {
            System.out.println("ErrorHibernate getCorpusAll: " + e);
            throw new Exception("ErrorHibernate getCorpusAll: " + e);
        }
    }
    public List<Corpus> getCorpusBy(String[] query) throws Exception {
        try {
            return (new CorpusDao(session)).findBy(query);
        } catch (HibernateException e) {
            System.out.println("ErrorHibernate getCorpusBy: " + e);
            throw new Exception("ErrorHibernate getCorpusBy: " + e);
        }
    }
    public List<Fuente> getFuentes()throws Exception {
        try {
            return (new FuenteDao(session)).getTodo();
        } catch (HibernateException e) {
            System.out.println("ErrorHibernate getDiccionarioAll: " + e);
            throw new Exception("ErrorHibernate getDiccionarioAll: " + e);
        }
    }
    public List<Fuente> getFuentesBy(String[] query) throws Exception {
        try {
            return (new FuenteDao(session)).findBy(query);
        } catch (HibernateException e) {
            System.out.println("ErrorHibernate getFuentes: " + e);
            throw new Exception("ErrorHibernate getFuentes: " + e);
        }
    }
    public List<FuenteER> getFuenteERsBy(String[] query) throws Exception {
        List<FuenteER> fuentesEr=new ArrayList<FuenteER>();
        List<Fuente> fuentes=(new FuenteDao(session)).findBy(query);
        for(Fuente f:fuentes){
            FuenteER fer=new FuenteER(f,getCategoriaById(f.getCategoriaId()),null);
            fuentesEr.add(fer);
        }
        return fuentesEr;
    }
    public List<Noticia> getNoticias() throws Exception {
        try {
            return (new NoticiaDao(session)).getTodo();
        } catch (HibernateException e) {
            System.out.println("Error interno al intentar obtener la lista de Noticia." +
e);
            throw new Exception("Error interno al intentar obtener la lista de Noticia."
+ e);
        }
    }
```

```
    public List<Noticia> getNoticiasBy(String[] query) throws Exception {
        try {
            return (new NoticiaDao(session)).findBy(query);
        } catch (HibernateException e) {
            System.out.println("Error interno al intentar obtener la lista de Noticia." +
e);
            throw new Exception("Error interno al intentar obtener la lista de Noticia."
+ e);
        }
    }
    public List<NoticiaER> getNoticiaERsBy(String[] query) throws Exception {
        List<NoticiaER> lner=new ArrayList<NoticiaER>();
        List<Noticia> noticias=(new NoticiaDao(session)).findBy(query);
        for(Noticia n: noticias){
            NoticiaER ner=new
NoticiaER(n,getCorpusById(n.getTituloId()),getCorpusById(n.getDescripcionId()),daoCali().
getPromedio(n.getNotiId()));
            lner.add(ner);
        }
        return lner;
    }
    public List<UsuarioCategoria> getUsuarioCategoriaBy(String[] query) throws Exception
{
        try {
            return (new UsuarioCategoriaDao(session)).findBy(query);
        } catch (HibernateException e) {
            System.out.println("HibernateException  getUsuarioCategoriaBy:" + e);
            throw new Exception("HibernateException  getUsuarioCategoriaBy:" + e);
        }
    }
    public List<UsuarioFuente> getUsuarioFuenteBy(String[] query) throws Exception {
        try {
            return (new UsuarioFuenteDao(session)).findBy(query);
        } catch (HibernateException e) {
            System.out.println("HibernateException  getUsuarioFuenteBy:" + e);
            throw new Exception("HibernateException  getUsuarioFuenteBy:" + e);
        }
    }
    public List<Categoria> getUsCa(Integer id) throws Exception{
        List<Categoria> lcate=new ArrayList<Categoria>();
        String hql="From UsuarioCategoria WHERE usuarioId="+id.toString();
        String[] query=new String[]{hql};
        List<UsuarioCategoria> lusca=getUsuarioCategoriaBy(query);
        System.out.println("getUsCa idUsuario="+id+" categorias="+lusca.size());
        for(UsuarioCategoria usca: lusca){
            lcate.add(getCategoriaById(usca.getCategoriaId()));
        }
        return lcate;
    }
    public List<Fuente> getUsFu(Integer id) throws Exception{
        System.out.println(".getUsFu. "+id);
        List<Fuente> lfuen=new ArrayList<Fuente>();
        String hql="From UsuarioFuente WHERE usuarioId="+id.toString();
        String[] query=new String[]{hql};
        List<UsuarioFuente> lusfu=getUsuarioFuenteBy(query);
        for(UsuarioFuente usfu: lusfu){
            lfuen.add(getFuenteById(usfu.getFuenteId()));
        }
        return lfuen;
    }
```

```
    public Usuario addUsuario(Usuario usuario) throws Exception {
        try {
            session.beginTransaction();
            (new UsuarioDao(session)).add(usuario);
            session.getTransaction().commit();
            return usuario;
        } catch (HibernateException e) {
            System.out.println("Error interno al intentar guardar un usuario."+e);
            if (session != null) {
                session.getTransaction().rollback();
            }
            throw new Exception("Error interno al intentar guardar un usuario."+e);
        }
    }
    public Fuente addFuente(Fuente fuente) throws Exception {
        try {
            session.beginTransaction();
            (new FuenteDao(session)).add(fuente);
            session.getTransaction().commit();
            return fuente;
        } catch (HibernateException e) {
            System.out.println("Error interno al intentar guardar una Fuente."+e);
            if (session != null) {
                session.getTransaction().rollback();
            }
            throw new Exception("Error interno al intentar guardar una Fuente."+ e);
        }
    }
    public Categoria addCategoria(Categoria categoria) throws Exception {
        try {
            session.beginTransaction();
            (new CategoriaDao(session)).add(categoria);
            session.getTransaction().commit();
            return categoria;
        } catch (HibernateException e) {
            System.out.println("Error interno al intentar guardar una Categoria."+e);
            if (session != null) {
                session.getTransaction().rollback();
            }
            throw new Exception("Error interno al intentar guardar una Categoria."+e);
        }
    }
    public Corpus addCorpus(Corpus corpus) throws Exception {
        try {
            session.beginTransaction();
            (new CorpusDao(session)).add(corpus);
            session.getTransaction().commit();
            return corpus;
        } catch (HibernateException e) {
            System.out.println("Error interno al intentar guardar un Corpus."+e);
            if (session != null) {
                session.getTransaction().rollback();
            }
            throw new Exception("Error interno al intentar guardar un Corpus."+e);
        }
    }
    public Corpus updateCorpus(Corpus corpus) throws Exception {
        try {
            session.beginTransaction();
            (new CorpusDao(session)).update(corpus);
```

```
            session.getTransaction().commit();
            return corpus;
        } catch (HibernateException e) {
            System.out.println("Error interno al intentar actualizar un Corpus."+e);
            if (session != null) {
                session.getTransaction().rollback();
            }
            throw new Exception("Error interno al intentar  actualizar un Corpus."+e);
        }
    }
    public Noticia addNoticia(Noticia notica) throws Exception {
        try {
            session.beginTransaction();
            (new NoticiaDao(session)).add(notica);
            session.getTransaction().commit();
            return notica;
        } catch (HibernateException e) {
            System.out.println("Error interno al intentar guardar una Notica."+e);

            if (session != null) {
                session.getTransaction().rollback();
            }
            throw new Exception("Error interno al intentar guardar una Notica."+e);
        }
    }
    public Noticia updateNoticia(Noticia notica) throws Exception {
        try {
            session.beginTransaction();
            (new NoticiaDao(session)).update(notica);
            session.getTransaction().commit();
            return notica;
        } catch (HibernateException e) {
            System.out.println("Error interno al intentar actualizar una Notica."+e);

            if (session != null) {
                session.getTransaction().rollback();
            }
            throw new Exception("Error interno al intentar actualizar una Notica."+e);
        }
    }
    public UsuarioCategoria addUsuarioCategoria(UsuarioCategoria usca) throws Exception {
        try {
            session.beginTransaction();
            (new UsuarioCategoriaDao(session)).add(usca);
            session.getTransaction().commit();
            return usca;
        } catch (HibernateException e) {
            System.out.println("HibernateException addUsuarioCategoria:"+e);
            if (session != null) {
                session.getTransaction().rollback();
            }
            throw new Exception("HibernateException addUsuarioCategoria:"+e);
        }
    }
    public UsuarioFuente addUsuarioFuente(UsuarioFuente usfu) throws Exception {
        try {
            session.beginTransaction();
            (new UsuarioFuenteDao(session)).add(usfu);
            session.getTransaction().commit();
            return usfu;
```

```
        } catch (HibernateException e) {
            System.out.println("HibernateException addUsuarioFuente:"+e);
            if (session != null) {
                session.getTransaction().rollback();
            }
            throw new Exception("HibernateException addUsuarioFuente:"+e);
        }
    }
    public Calificacion addCalificacion(Calificacion cali)  throws Exception {
        try {
            session.beginTransaction();
            (new CalificacionDao(session)).add(cali);
            session.getTransaction().commit();
            return cali;
        } catch (HibernateException e) {
            System.out.println("HibernateException addCalificacion:"+e);
            if (session != null) {
                session.getTransaction().rollback();
            }
            throw new Exception("HibernateException addCalificacion:"+e);
        }
    }
    public boolean addEstadistica(String frase) throws Exception{
        boolean res=false;
        String[] query = new String[]{"FROM Estadistica WHERE estaFrase = :cad",
"cad",frase};
        try{
            EstadisticaDao dao=daoEsta();
            List<Estadistica> lesta=dao.findBy(query);
            if(lesta.isEmpty()){
                dao.add(new Estadistica(frase,1));
            }else{
                Estadistica esta=lesta.get(0);
                int num=esta.getEstaRepeticion();
                esta.setEstaRepeticion(num+1);
                dao.update(esta);
            }
            res=true;
        } catch (HibernateException e) {
            System.out.println("Error interno en Estadistica.verificar."+e);
            if (session != null) {
                session.getTransaction().rollback();
            }
            throw new Exception("Error interno hacer la operarion verificar."+e);
        }
        return res;
    }
    public Usuario loggin(String un,String p){
        return (new UsuarioDao(session)).loggin(un, p);
    }
}
```

A.2. Código fuente del módulo recolector

Clase Elemento

```
package rss;

public abstract class Elemento {
    String imagenUrl;
    String titulo;
    String descripcion;
```

```
    String link;
    String pubDate;

    public void setImagenUrl(String ImagenUrl) {
        this.imagenUrl = ImagenUrl;
    }
    public void setTitulo(String titulo) {
        this.titulo = titulo;
    }
    public void setDescripcion(String descripcion) {
        this.descripcion = descripcion;
    }
    public void setLink(String link) {
        this.link = link;
    }
    public void setPubDate(String pubDate) {
        this.pubDate = pubDate;
    }
    public String getImagenUrl() {
        return imagenUrl;
    }
    public String getTitulo() {
        return titulo;
    }
    public String getDescripcion() {
        return descripcion;
    }
    public String getLink() {
        return link;
    }
    public String getPubDate() {
        return pubDate;
    }
}
```

Clase Canal

```
package rss;

public class Canal extends Elemento{
    String lenguaje;
    String copyright;

    public void setLenguaje(String lenguaje) {
        this.lenguaje = lenguaje;
    }
    public void setCopyright(String copyright) {
        this.copyright = copyright;
    }
    public String getLenguaje() {
        return lenguaje;
    }
    public String getCopyright() {
        return copyright;
    }
    @Override
    public String toString() {
        return "\n\timagen=" + imagenUrl +
               "\n\ttitulo=" + titulo +
               "\n\tdescripcion=" + descripcion+
               "\n\tlink=" + link +
               "\n\tpubdate=" + pubDate +
```

```
                "\n\tlenguaje=" + lenguaje +
                "\n\tcopyright=" + copyright
                 + "\n\n";
    }
}
```

Clase Item

```
package rss;

public class Item extends Elemento{
    String autor;
    String guid;

    public void setAutor(String autor) {
        this.autor = autor;
    }
    public void setGuid(String guid) {
        this.guid = guid;
    }
    public String getAutor() {
        return autor;
    }
    public String getGuid() {
        return guid;
    }
    @Override
    public String toString() {
        return "\n\t\timagen=" + imagenUrl +
                "\n\t\ttitulo=" + titulo +
                "\n\t\tdescripcion=" + descripcion+
                "\n\t\tlink=" + link +
                "\n\t\tpubdate=" + pubDate +
                "\n\t\tautor=" + autor +
                "\n\t\tguid=" + guid
                 + "\n";
    }
}
```

Clase Fuente

```
package rss;

import java.util.ArrayList;
import java.util.List;

public class FuenteRss {
    Canal canal;
    List<Item> lItems;

    public FuenteRss() {
        lItems=new ArrayList<Item>();
    }
    public FuenteRss(Canal canal, List<Item> lItems) {
        this.canal = canal;
        this.lItems = lItems;
    }
    public void setCanal(Canal canal) {
        this.canal = canal;
    }
    public void setlItems(List<Item> lItems) {
        this.lItems = lItems;
    }
    public Canal getCanal() {
```

```
        return canal;
    }
    public List<Item> getlItems() {
        return lItems;
    }
}
```

Parseador

```
package rss;

import hibernate.modelo.Fuente;
import java.io.IOException;
import java.io.InputStream;
import java.net.MalformedURLException;
import java.net.URL;
import javax.xml.stream.XMLEventReader;
import javax.xml.stream.XMLInputFactory;
import javax.xml.stream.XMLStreamException;
import javax.xml.stream.events.XMLEvent;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class ParseadorRSS {
    static final String CANAL = "channel";
    static final String ITEM = "item";
    static final String IMG = "image";
    static final String TITULO = "title";
    static final String DESCRIPCION = "description";
    static final String LENGUAJE = "language";
    static final String COPYRIGHT = "copyright";
    static final String LINK = "link";
    static final String AUTOR = "author";
    static final String PUB_DATE = "pubDate";
    static final String GUID = "guid";
    static final String URL = "url";
    final URL url;
    private FuenteRss feed;
    private boolean esCorrecto;
    private String msj;

    public ParseadorRSS(String feedUrl) {
        System.out.println(feedUrl);
        try {
            this.url = new URL(feedUrl);
            leerFeed();
            esCorrecto=true;
        } catch (MalformedURLException e) {
            esCorrecto=false;
            msj=e.getMessage();
            throw new RuntimeException(e);
        }
    }
    public ParseadorRSS(Fuente f) {
        try {
            System.out.println(f.getFuentLink());
            this.url = new URL(f.getFuentLink());
            leerFeed();
            Canal c=feed.getCanal();
            f.setFuentCopyright(c.getCopyright());
            f.setFuentDescripcion(c.getDescripcion());
            f.setFuentPubDate(c.getPubDate());
```

```
            f.setFuentTitulo(c.getTitulo());
            esCorrecto=true;
        } catch (MalformedURLException e) {
            esCorrecto=false;
            msj=e.getMessage();
            throw new RuntimeException(e);
        }
    }
    public FuenteRss leerFeed() {
        feed=new FuenteRss();
        Elemento activo=null;
        Canal canal = null;
        Item item = null;
        String cad;
        try {
            XMLInputFactory xmlif = XMLInputFactory.newInstance();
            XMLEventReader xmler = xmlif.createXMLEventReader(read());
            while (xmler.hasNext()) {
                XMLEvent xmle = xmler.nextEvent();
                //System.out.println(xmle);
                if (xmle.isStartElement()) {
                    cad=xmle.asStartElement().getName().getLocalPart();
                    //System.out.println(cad);
                    switch (cad) {
                        case CANAL:canal=new Canal();activo=canal;break;
                        case ITEM:item=new Item();activo=item;break;
                        case TITULO:activo.setTitulo(getTexto(xmle, xmler));break;
                        case DESCRIPCION:activo.setDescripcion(getTexto(xmle,
xmler));sacarImg(activo);break;
                        case LINK:activo.setLink(getTexto(xmle, xmler));break;
                        case PUB_DATE:activo.setPubDate(getTexto(xmle, xmler));break;
                        case URL:activo.setImagenUrl(getTexto(xmle, xmler));break;
                        case GUID:item.setGuid(getTexto(xmle, xmler));break;
                        case LENGUAJE:canal.setLenguaje(getTexto(xmle, xmler));break;
                        case AUTOR:item.setAutor(getTexto(xmle, xmler));break;
                        case COPYRIGHT:canal.setCopyright(getTexto(xmle, xmler));break;
                    }
                } else if (xmle.isEndElement()) {
                    cad=xmle.asEndElement().getName().getLocalPart();
                    //System.out.println(cad);
                    switch (cad) {
                        case ITEM: feed.getlItems().add(item);break;
                        case CANAL:feed.setCanal(canal);break;
                    }
                }

            }
        } catch (XMLStreamException e) {
            throw new RuntimeException(e);
        }
        return feed;
    }
    private void sacarImg(Elemento item) {
        if(item!=null && !item.getDescripcion().isEmpty()){
            String text =item.getDescripcion();
            Pattern erImg = Pattern.compile("(\\<img)(.*?)(\\>)");
            Matcher mImg = erImg.matcher(text);
            if(mImg.find()){
                String simg=mImg.group();
                text=text.replace(simg,"").trim();
```

```
                Pattern erSrc = Pattern.compile("(src=\")(.*?)(\")");
                Matcher mSrc = erSrc.matcher(simg);
                if(mSrc.find()){
                    item.setImagenUrl(mSrc.group(2));
                }
                item.setDescripcion(text);
            }
        }
    }
    private String getTexto(XMLEvent event, XMLEventReader eventReader) throws
XMLStreamException {
        String result = "";
        event = eventReader.nextEvent();
        while (event.isCharacters()) {
            result+= event.asCharacters().getData();
            event = eventReader.nextEvent();
        }
        return result;
    }
    private InputStream read() {
        try {
            return url.openStream();
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }
    public boolean isEsCorrecto() {
        return esCorrecto;
    }
    public String getMsj() {
        return msj;
    }
    public FuenteRss getFeed() {
        return feed;
    }
}
```

A.3. Código fuente del módulo traducción

Clase Core

```
package core.ctr;

import entidadRelacion.*;
import hibernate.ctr.DB;
import hibernate.modelo.*;
import java.util.ArrayList;
import java.util.List;
import java.util.regex.Matcher;
import java.util.regex.Pattern;
import rss.*;

public class Core {
    private DB ctr;

    public Core() {
        ctr = new DB();
    }
    public FuenteER getFeed(Fuente f) throws Exception {
        System.out.println("getFeed");
        String hql = "From Noticia WHERE notiLink=:link";
```

```
        String[] query = new String[]{hql, "link", "un link :)"};
        FuenteER fer = new FuenteER(f, ctr.getCategoriaById(f.getCategoriaId()), null);
        List<NoticiaER> getlner;
        try {
            ParseadorRSS parser = new ParseadorRSS(f.getFuentLink());
            List<NoticiaER> lner = new ArrayList<NoticiaER>();
            for (Item item : parser.getFeed().getlItems()) {
                System.out.println(item);
                query[2] = item.getLink();
                getlner = ctr.getNoticiaERsBy(query);
                if (getlner.isEmpty()) {
                    lner.add(traduccion(item, f));
                } else {
                    lner.add(getlner.get(0));
                }
            }
            fer.setM_lNoticiaERs(lner);
        } catch (Exception e) {
            throw new Exception("Core getFeed: " + e);
        }
        return fer;
    }
    public NoticiaER traduccion(Item item, Fuente f) {
        String stitu, sdesc;
        int idtitu=0, iddesc=0;
        Corpus titulo,descripcion;
        NoticiaER ner = new NoticiaER();
        try {
            if (item.getTitulo() != "") {
                stitu = item.getTitulo();
                titulo = new Corpus(stitu, traduccion(stitu));
                System.out.println("TITULO\nesp :" + titulo.getCorpEsp()+"\nnah:
"+titulo.getCorpNah());
                ctr.addCorpus(titulo);
                idtitu=titulo.getCorpId();
                ner.setM_oCorpusTitulo(titulo);
            }
            if (item.getDescripcion() != "") {
                sdesc = item.getDescripcion();
                descripcion = new Corpus(sdesc, traduccion(sdesc));
                System.out.println("DESCRIPCION\nESP :" +
descripcion.getCorpEsp()+"\nNAH: "+descripcion.getCorpNah());
                ctr.addCorpus(descripcion);
                iddesc=descripcion.getCorpId();
                ner.setM_oCorpusDescripcion(descripcion);
            }
            Noticia noticia = new Noticia(
                    item.getLink(),
                    item.getAutor(),
                    f.getFuentId(),
                    idtitu,
                    iddesc,
                    item.getPubDate()
            );
            ctr.addNoticia(noticia);
            ner.setM_oNoticia(noticia);

        } catch (Exception e) {
            System.out.println("Core traduccion(item,f): " + e);
        }
```

```
        return ner;
    }
    public String traduccion(String esp) {
        String nah = "";
        boolean bseg = false;
        String sseg = "";
        esp = esp.trim();
        Pattern erAlfa = Pattern.compile("^[a-zA-Z\\sñàèìòùáéíóúÑÀÈÌÒÙÁÉÍÓÚ]");
        int tam = esp.length();
        Matcher matcher;
        System.out.println("Traduciendo : "+esp);
        for (int i = 0; i < tam; i++) {
            matcher = erAlfa.matcher(Character.toString(esp.charAt(i)));
            if (matcher.matches()) {
                bseg = true;
                sseg += esp.charAt(i);
            } else {
                if (bseg) {
                    nah+= traduccionSegmento(sseg);
                    bseg = false;
                    sseg = "";
                }
                nah += esp.charAt(i);
            }
        }
        if(sseg!=""){
             nah += traduccionSegmento(sseg);
        }
        return nah;
    }
    public String traduccionSegmento(String esp) {
        String nah = "",fin="";
        if(esp.matches("^[\\s]")){
            nah=" ";
        }
        if(esp.matches("[\\s]$")){
            fin=" ";
        }
        esp = esp.trim();
        String[] segs = esp.split(" ");
        int tam = segs.length;
        int i = 0;
        System.out.println("Segmento formado: "+esp+"<> Con "+tam+" palabras");
        try{
            while (i < tam) {
                for (int j = (tam - 1); j >= i; j--) {
                    String frase = "";
                    for (int f = i; f <= j; f++) {
                        frase += segs[f] + " ";
                    }
                    System.out.println("Buscando...: "+frase);
                    List<Corpus> lcorp = ctr.daoCorp().getByEsp(frase);
                    if (!lcorp.isEmpty()) {
                        Corpus corp = lcorp.get(0);
                        String[] palabrasNahuatl = corp.getCorpNah().split(",");
                        nah += " " + palabrasNahuatl[0];
                        i = j+1;
                        System.out.println("Encontrado nah: "+palabrasNahuatl[0]);
                        continue;
                    } else if ((j - i) == 0) {
```

```
                        ctr.addEstadistica(frase);
                        nah += " " + frase;
                        i = j+1;
                        System.out.println("No encontrado: "+frase);
                        continue;
                    }
                }
            }
        }catch(Exception e){
            for (int f = i; f < tam; f++) {
               nah += segs[f] + " ";
            }
        }
        return nah.trim()+fin;
    }
}
```

API REST Usuario

```
package rest;

import entidadRelacion.UsuarioER;
import hibernate.modelo.*;
import hibernate.ctr.DB;
import javax.ws.rs.GET;
import javax.ws.rs.POST;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.core.MediaType;
import respuesta.rUsuarioER;

@Path("/usuarioER")
@Produces(MediaType.APPLICATION_JSON)
public class UsuarioERApi {
    private final DB ctr;
    public UsuarioERApi() {
        ctr = new DB();
    }
    @GET
    @Path("/login/{username}/{password}")
    public rUsuarioER logging(@PathParam("username") String username,
@PathParam("password") String pass) {
        System.out.println("usuarioER/"+username+"/"+pass);
        boolean status = false;
        String msj="usuarioER/"+username+"/"+pass;
        UsuarioER uer=new UsuarioER();
        try{
            Usuario user = ctr.loggin(username, pass);
            if (user != null) {
                msj = "Bienvenido :" + user.getUsuaApaterno() + " " +
user.getUsuaAmaterno() + " " + user.getUsuaNombre();
                uer.setM_oUsuario(user);
                uer.setM_lCategorias(ctr.getUsCa(user.getUsuaId()));
                uer.setM_lFuentes(ctr.getUsFu(user.getUsuaId()));
                status = true;
            } else  msj = "Nombre de usuario o password incorrectos :(";
        }catch(Exception e){
            msj = "Error: " + e;
        }
        finally{
            return new rUsuarioER(status, msj, uer);
```

```
        }
    }
    @POST
    public rUsuarioER add(UsuarioER uer) {
        System.out.println("usuarioER->add");
        boolean status = false;
        String msj ="usuarioER->add";
        try {
            Usuario usua=uer.getM_oUsuario();
            ctr.addUsuario(usua);
            for(Categoria cate : uer.getM_lCategorias()){
                UsuarioCategoria usca=new
UsuarioCategoria(usua.getUsuaId(),cate.getCateId());
                ctr.addUsuarioCategoria(usca);
            }
            for(Fuente cafu : uer.getM_lFuentes()){
                UsuarioFuente usfu=new UsuarioFuente(usua.getUsuaId(),cafu.getFuentId());
                ctr.addUsuarioFuente(usfu);
            }
            msj = "Usuario nuevo";
            status = true;
        } catch (Exception e) {
            msj = "Error: " + e;
        }
        finally{
            return new rUsuarioER(status, msj, uer);
        }
    }
}
```

API REST Categoría

```
package rest;

import respuesta.*;
import hibernate.ctr.DB;
import hibernate.modelo.*;
import java.util.List;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.Produces;
import javax.ws.rs.QueryParam;
import javax.ws.rs.core.MediaType;

@Path("/categoria")
@Produces(MediaType.APPLICATION_JSON)
public class CategoriaApi {
    private final DB ctr;

    public CategoriaApi() {
        ctr = new DB();
    }
    @GET
    public rCategoria getById(@QueryParam("id") Integer id) {
        System.out.println("categoria?id="+id);
        boolean status = false;
        String msj="categoria?id="+id;
        Categoria objeto = null;
        try {
            objeto = ctr.getCategoriaById(id);
            status= objeto!=null?true:false;
        } catch (Exception e) {
```

```
            msj = "Error: " + e.getMessage();
        }
        finally{
            return new rCategoria(status, msj, objeto);
        }
    }
    @GET
    @Path("/all")
    public rCategorias getAll() {
        System.out.println("categoria/all");
        boolean status = false;
        String msj="categoria/all";
        List<Categoria> lista = null;
        try {
            lista = ctr.getCategorias();
            msj = "Categorias";
            status = true;
        } catch (Exception e) {
            msj = "Error: " + e.getMessage();
        }
        finally{
            return new rCategorias(status, msj, lista);
        }
    }
}
```

API REST fuente

```
package rest;

import entidadRelacion.FuenteER;
import respuesta.*;
import hibernate.ctr.DB;
import core.ctr.*;
import entidadRelacion.UsuarioER;
import hibernate.modelo.*;
import java.util.List;
import javax.ws.rs.*;
import javax.ws.rs.core.MediaType;
import rss.ParseadorRSS;

@Path("/fuente")
@Produces(MediaType.APPLICATION_JSON)
public class FuenteApi {
    private final DB ctr;
    private final Core rss;

    public FuenteApi() {
        ctr = new DB();
        rss=new Core();
    }
    @GET
    public rFuenteER getById(@QueryParam("id") Integer id) {
        System.out.println("fuente?id="+id);
        boolean status = false;
        String msj="fuente?id="+id;
        FuenteER fer = null;
        try {
            Fuente f = ctr.getFuenteById(id);
            status= f!=null?true:false;
            if(status && f.getFuentLink() != ""){
                msj="últimas noticias";
```

```
                fer=rss.getFeed(f);
            }else msj="Link del fuente inexistente";
        } catch (Exception e) {
            msj = "Error: " + e.getMessage();
             status=false;
        }
        finally{
            return new rFuenteER(status,msj,fer);
        }
    }
    @GET
    @Path("/byIdCategoria/{id}")
    public rFuentes getByCate(@PathParam("id") Integer id) {
        System.out.println("fuente//byIdCategoria/id"+id);
        boolean status = false;
        String msj="fuente/byIdCategoria/id";
        List<Fuente> lista = null;
        String[] query=new String[]{"From Fuente WHERE categoriaId="+id.toString()};
        try {
            lista = ctr.getFuentesBy(query);
            if(lista.isEmpty()){
                msj = "No hay fuentes que le correspondan a la categoria";
            }else{
                msj = "Fuente(s) que le corresponden a la categoria seleccionado";
                status = true;
            }

        } catch (Exception e) {
            msj = "Error: " + e.getMessage();
        }
        finally{
            return new rFuentes(status, msj, lista);
        }
    }
    @GET
    @Path("/all")
    public rFuentes getAll() {
        System.out.println("fuente/all");
        boolean status = false;
        String msj="fuente/all";
        List<Fuente> lista = null;

        try {
            lista = ctr.getFuentes();
            msj = "Fuentes:";
            status = true;
        } catch (Exception e) {
            msj = "Error: " + e.getMessage();
        }
        finally{
            return new rFuentes(status, msj, lista);
        }
    }
    @POST
    @Path("/FuenteER")
    public rFuenteER add(FuenteER fer) {
        System.out.println("fuente/FuenteER");
        Fuente f=fer.getM_oFuente();
        boolean status = false;
        String msj="fuente/FuenteER";
```

```
        ///FIN DE LIMPIAR URL DE LA FUENTE
        try {
            String urlFeed=f.getFuentLink().replace(" ","");
            f.setFuentLink(urlFeed);
            //CONSULTAR SI EXISTE LA URL RECIBIDA
            String hql="From Fuente WHERE fuentLink= :link";
            String[] query=new String[]{hql,"link",f.getFuentLink()};
            List<Fuente> fuentes=ctr.getFuentesBy(query);
            if(fuentes.isEmpty()){
                // si NO existe la fuente, se hace el procesamiento.
                Categoria cate=fer.getM_oCategoria();
                if (cate.getCateId()==null || cate.getCateId()==0){
                    ctr.addCategoria(cate);
                    System.out.println("Categoria nueva id="+cate.getCateId());
                }
                f.setCategoriaId(cate.getCateId());
                ParseadorRSS prss=new ParseadorRSS(f);
                status = prss.isEsCorrecto();
                if(status){
                     ctr.addFuente(f);
                     System.out.println("Fuente nueva id="+f.getFuentId());
                     msj = "Agregación de Fuente exitosamente.";
                }else{
                    msj=prss.getMsj();
                }
            }else{
                //caso contrario, no se hace nada.
                msj="Fuente ya existente.";
            }
        } catch (Exception e) {
            msj+= "Error: " + e;
            System.out.println("Error: " + e);
        }
        finally{
            return new rFuenteER(status, msj, fer);
        }
    }
}
```

API REST Calificación

```
package rest;

import respuesta.rCalificacion;
import respuesta.Response;
import hibernate.ctr.DB;
import hibernate.modelo.*;
import java.util.List;
import javax.ws.rs.*;
import javax.ws.rs.core.MediaType;

@Path("/calificacion")
@Produces(MediaType.APPLICATION_JSON)
public class CalificacionApi {
    private final  DB ctr;
    public CalificacionApi(){
       ctr = new DB();
    }
    @GET
    public Integer hola(@QueryParam("idnoti") Integer noti,@QueryParam("idUsua") Integer
usua){
       return ctr.daoCali().getPromedio(noti,usua);
```

```
    }
    @POST
    @Path("/{noti}/{usua}/{valor}")
    public Integer Byget(@PathParam("noti") int n,@PathParam("usua") int
u,@PathParam("valor") int v) {
        Integer res = -1;
        try {
            Calificacion entidad=new Calificacion(n,u,v);
            ctr.addCalificacion(entidad);
            res=ctr.daoCali().getPromedio(n);

        } catch (Exception e) {
            System.out.println(e);
        }
        return res;
    }
    @POST
    public Integer create(Calificacion entidad) {
        Integer res = -1;
        try {
            ctr.addCalificacion(entidad);
            res=ctr.daoCali().getPromedio(entidad.getCaliIdNoticia());
        } catch (Exception e) {
            System.out.println(e);
        }
        return res;
    }
}
```

A.4. Código fuente del módulo gestión de usuarios

AndroidManifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.camilo.rssnahuatl" >
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".Logeo"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".ListaAct"
            android:label="@string/title_activity_lista_noticias" >
        </activity>
        <activity
            android:name=".RegistroAct"
            android:label="@string/title_activity_registro" >
        </activity>
        <activity
```

```
            android:name=".FuenteAct"
            android:label="@string/title_activity_fuente" >
        </activity>
    </application>
</manifest>
```

Caso de uso registrarse

LAYOUT para la interfaz de registro

```
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:paddingBottom="@dimen/activity_vertical_margin"
        android:paddingLeft="@dimen/activity_horizontal_margin"
        android:paddingRight="@dimen/activity_horizontal_margin"
        android:paddingTop="@dimen/activity_vertical_margin">
        <TextView
            android:id="@+id/tvNombre"
            style="@style/fontGR"
            android:text="Nombre:" />
        <EditText
            android:id="@+id/etNombre"
            style="@style/item"
            android:hint="Nombre(s) ..." />
        <TextView
            android:id="@+id/tvAPaterno"
            style="@style/fontGR"
            android:text="Apellido paterno:" />
        <EditText
            android:id="@+id/etAPaterno"
            style="@style/item"
            android:hint="Apellido paterno ..." />
        <TextView
            android:id="@+id/tvAMaterno"
            style="@style/fontGR"
            android:text="Apellido materno:" />
        <EditText
            android:id="@+id/etAMaterno"
            style="@style/item"
            android:hint="Apellido materno ..." />
        <TextView
            android:id="@+id/tvUser"
            style="@style/fontGR"
            android:text="Nombre de usuario:" />
        <EditText
            android:id="@+id/etUser"
            style="@style/item"
            android:hint="Nombre de usuario..." />
        <TextView
            android:id="@+id/tvPass"
            style="@style/fontGR"
            android:text="Contraseña:" />
        <EditText
            android:id="@+id/etPass"
            style="@style/item"
            android:hint="Contraseña..."
```

```
            android:inputType="textPassword" />
        <TextView
            android:id="@+id/tvPassC"
            style="@style/fontGR"
            android:text="Confirme cotraseña:" />
        <EditText
            android:id="@+id/etPassC"
            style="@style/item"
            android:hint="Confirme su contraseña..."
            android:inputType="textPassword" />
        <CheckBox
            android:id="@+id/cbNah"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Habla náhuatl?" />
        <CheckBox
            android:id="@+id/cbEsp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Habla español?" />
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal">
            <Button
                android:id="@+id/btnSelecFuens"
                style="@style/btn"
                android:layout_marginRight="10dp"
                android:layout_weight="1"
                android:text="Fuentes"
                android:drawableRight="@android:drawable/arrow_down_float"
                android:onClick="btnFuens"/>
            <Button
                android:id="@+id/btnSelecCates"
                style="@style/btn"
                android:layout_marginLeft="10dp"
                android:layout_weight="1"
                android:text="Categorias"
                android:drawableRight="@android:drawable/arrow_down_float"
                android:onClick="btnCates"/>
        </LinearLayout>
        <Button
            android:id="@+id/btnEnviarForm"
            style="@style/btn"
            android:onClick="enviarForm"
            android:text="Enviar" />
    </LinearLayout>
</ScrollView>
```

Clase padre para todos los controladores de los LAYOUTs

```
package com.camilo.rssnahuatl;

import android.support.v7.app.ActionBarActivity;
import android.view.inputmethod.InputMethodManager;
import android.app.*;
import android.content.*;
import android.widget.Toast;
import com.camilo.modelo.*;
import com.camilo.respuesta.*;
import com.camilo.rest.REST;
import com.camilo.utils.UTIL;
```

```
import java.util.List;
import retrofit.Callback;
import retrofit.RetrofitError;
import retrofit.client.*;
import retrofit.client.Response;

public abstract class  Core extends ActionBarActivity {
    public static final String ALMACEN = "rssNahualt" ;
    public static final int MODO = Context.MODE_MULTI_PROCESS ;

    InputMethodManager m_immInputManager;
    public ProgressDialog m_pdCargando;
    SharedPreferences sharedpreferences;
    public void iniciar(){
        m_pdCargando = new ProgressDialog(this);
        m_immInputManager = (InputMethodManager)
getSystemService(Context.INPUT_METHOD_SERVICE);
        renderizacion();
    }
    public void alertL(String msg){
        Toast.makeText(getApplicationContext(), msg, Toast.LENGTH_LONG).show();
    }
    public void alertS(String msg){
        Toast.makeText(getApplicationContext(), msg, Toast.LENGTH_SHORT).show();
    }
    public void alertD(String tit,String msg){
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setTitle(tit)
                .setIcon(getResources().getDrawable(android.R.drawable.ic_dialog_info))
                .setMessage(msg)
                .setPositiveButton(R.string.ok,null);

        builder.create().show();
    }
    public void alertP(String tit,String msg){
        m_pdCargando.setTitle(tit);
        m_pdCargando.setMessage(msg);
        m_pdCargando.setCancelable(false);
        m_pdCargando.show();
    }
    public abstract void renderizacion();

    public void set(String clave,String valor){
        sharedpreferences = getSharedPreferences(ALMACEN, MODO);
        SharedPreferences.Editor editor = sharedpreferences.edit();
        editor.putString(clave, valor);
        editor.commit();
    }
    public String get(String clave){
        sharedpreferences = getSharedPreferences(ALMACEN, MODO);
        return sharedpreferences.getString(clave,null);
    }
    public String getExtra(String tag){
        return this.getIntent().getExtras().getString(tag);
    }
    public void newFuente(Fuente f){
        String fuens=get(getString(R.string.set_fuens));
        if(fuens!=null){
            rFuentes ff=(rFuentes)UTIL.fromJson(fuens, "rFuentes");
            ff.getM_lFuentes().add(f);
```

```
            set(getString(R.string.set_fuens),UTIL.toJson(ff));
        }
    }
    public void newCategoria(Categoria c){
        String cates=get(getString(R.string.set_cates));
        if(cates!=null){
            rCategorias rc=(rCategorias)UTIL.fromJson(cates, "rCategorias");
            rc.getM_lCategorias().add(c);
            set(getString(R.string.set_cates),UTIL.toJson(rc));
        }
    }
}
```

Controlador para la interfaz registro

```
package com.camilo.rssnahuatl;

import android.app.AlertDialog;
import android.app.DialogFragment;
import android.content.DialogInterface;
import android.os.Bundle;
import android.view.*;
import android.widget.*;
import com.camilo.dialogos.variosCheck;
import com.camilo.entidadRelacion.*;
import com.camilo.modelo.*;
import com.camilo.respuesta.*;
import com.camilo.rest.REST;
import com.camilo.utils.UTIL;
import java.util.ArrayList;
import java.util.List;
import retrofit.Callback;
import retrofit.RetrofitError;
import retrofit.client.Response;

public class RegistroAct extends Core {
    private final static String ENT="RegistroAct";
    private EditText m_etNombre,m_etAPaterno,m_etAMaterno,m_etUser,m_etPass,m_etPassC;
    private CheckBox m_cbNah,m_cbEsp;
    private Button m_btnEnviar;
    private variosCheck  m_diaFuens;
    private variosCheck  m_diaCates;
    private UsuarioER m_ouer;
    private List<Fuente> m_lFuentes;
    private List<Categoria> m_lCategorias;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_registro);
        iniciar();
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.menu_registro, menu);
        return true;
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        int id = item.getItemId();
```

```
        if (id == R.id.action_settings) {
            return true;
        }

        return super.onOptionsItemSelected(item);
    }
    public void renderizacion(){
        m_etNombre=(EditText)findViewById(R.id.etNombre);
        m_etAPaterno=(EditText)findViewById(R.id.etAPaterno);
        m_etAMaterno=(EditText)findViewById(R.id.etAMaterno);
        m_etUser=(EditText)findViewById(R.id.etUser);
        m_etPass=(EditText)findViewById(R.id.etPass);
        m_etPassC=(EditText)findViewById(R.id.etPassC);
        m_cbNah=(CheckBox)findViewById(R.id.cbNah);
        m_cbEsp=(CheckBox)findViewById(R.id.cbEsp);
        m_ouer=new UsuarioER();
    }
    public void enviarForm(View view) {
        String nombre=m_etNombre.getText().toString();
        String apaterno=m_etAPaterno.getText().toString();
        String amaterno=m_etAMaterno.getText().toString();
        String username=m_etUser.getText().toString();
        String password=m_etPass.getText().toString();
        String passwordC=m_etPassC.getText().toString();
        boolean bnah=m_cbNah.isChecked();
        boolean besp=m_cbEsp.isChecked();
        String nah=bnah?"Si":"No";
        String esp=besp?"Si":"No";

        if(UTIL.isNull(nombre)){
            alertS(getString(R.string.no_input_usuaNombre));
        } else if(UTIL.isNull(apaterno)){
            alertS(getString(R.string.no_input_usuaAP));
        } else if(UTIL.isNull(amaterno)){
            alertS(getString(R.string.no_input_usuaAM));
        } else if(!UTIL.validarName(username)){
            alertL(getString(R.string.no_input_usuaUsername));
        } else if(!UTIL.validarPass(password)){
            alertL(getString(R.string.no_input_usuaPass));
        } else if(UTIL.isNull(passwordC) || !passwordC.contentEquals(password)){
            alertS(getString(R.string.no_input_usuaPassC));
        } else if(!(bnah || besp)){
            alertS(getString(R.string.no_input_usuaIdiomas));
        }else if(m_diaFuens==null){

alertD(getString(R.string.msj_danger),getString(R.string.no_input_fuens_cates));
        }else{
            m_ouer.setM_oUsuario(new Usuario(nombre, apaterno, amaterno, username,
password, besp, bnah));

            List idFuens=m_diaFuens.getM_lItemsSeleccionados();
            List idCates=m_diaCates.getM_lItemsSeleccionados();
            m_ouer.setM_lFuentes(new ArrayList<Fuente>());
            for(int i=0,tam=idFuens.size();i<tam;i++){
                m_ouer.getM_lFuentes().add(m_lFuentes.get((int)idFuens.get(i)));
            }
            m_ouer.setM_lCategorias(new ArrayList<Categoria>());
            for(Object id:m_diaCates.getM_lItemsSeleccionados()){
                m_ouer.getM_lCategorias().add(m_lCategorias.get((int)id));
            }
```

```
            alertP(getString(R.string.ui2), getString(R.string.pet_add_uer));
            REST.get().usua().add(m_ouer, new Callback<rUsuarioER>() {
                @Override
                public void success(rUsuarioER rUsuarioER, Response response) {
                    m_pdCargando.dismiss();
                    if(rUsuarioER.isM_bStatus()){
                        set("etUser",m_etUser.getText().toString());
                        finish();
                    }
                    alertS(rUsuarioER.getM_sMsj());
                }
                @Override
                public void failure(RetrofitError error) {
                    m_pdCargando.dismiss();
                    alertD(getString(R.string.msj_error), error.getMessage());
                }
            });
        }
    }
    public void btnFuens(View v){
        String lf=get(getString(R.string.set_fuens));
        if(lf != null){
            rFuentes rfs=(rFuentes) UTIL.fromJson(lf, "rFuentes");
            m_lFuentes=rfs.getM_lFuentes();
            dialogFuens();
        }else{
            alertP(getString(R.string.ui2), getString(R.string.pet_fuens));
            REST.get().fuen().getTodas(new Callback<rFuentes>() {
                @Override
                public void success(rFuentes rFuentes, Response response) {
                    m_pdCargando.dismiss();
                    alertS(rFuentes.getM_sMsj());
                    if(rFuentes.isM_bStatus()){
                        set(getString(R.string.set_fuens), UTIL.toJson(rFuentes));
                        m_lFuentes=rFuentes.getM_lFuentes();
                        dialogFuens();
                    }
                }
                @Override
                public void failure(RetrofitError error) {
                    m_pdCargando.dismiss();
                    alertD(getString(R.string.msj_error), error.getMessage());
                }
            });
        }

    }
    public void dialogFuens(){
        if(m_lFuentes!=null && !m_lFuentes.isEmpty()){
            if(m_diaFuens==null){
                m_diaFuens = new variosCheck();
                m_diaFuens.setM_lItems(UTIL.fuenToList(m_lFuentes));
                m_diaFuens.setM_sTitulo(getString(R.string.select_fuens));
            }
            m_diaFuens.show(getSupportFragmentManager(), ENT);
        }else{
            alertD(getString(R.string.msj_danger), getString(R.string.msj_no_fuens));
        }
    }
    public void btnCates(View view){
```

```
        if(m_lCategorias!=null && !m_lCategorias.isEmpty()) {
            dialogCates();
        }else{
            String slista = get(getString(R.string.set_cates));
            if (slista != null) {
                rCategorias rcs = null;
                rcs = (rCategorias) UTIL.fromJson(slista, "rCategorias");
                m_lCategorias=rcs.getM_lCategorias();
                dialogCates();
            } else {
                alertP(getString(R.string.ui2), getString(R.string.pet_cates));
                REST.get().cate().getAll(new Callback<rCategorias>() {
                    @Override
                    public void success(rCategorias rCategorias, Response response) {
                        m_pdCargando.dismiss();
                        alertS(getString(R.string.pet_cates_success));
                        set(getString(R.string.set_cates), UTIL.toJson(rCategorias));
                        m_lCategorias=rCategorias.getM_lCategorias();
                        dialogCates();
                    }
                    @Override
                    public void failure(RetrofitError error) {
                        m_pdCargando.dismiss();
                        alertD(getString(R.string.msj_error), error.getMessage());
                    }
                });
            }
        }
    }
    public void dialogCates(){
        if(m_lCategorias.isEmpty()){
            alertD(getString(R.string.msj_danger), getString(R.string.msj_no_cates));
        }else{
            if(m_diaCates==null) {
                m_diaCates = new variosCheck();
                m_diaCates.setM_lItems(UTIL.cateToList(m_lCategorias));
                m_diaCates.setM_sTitulo(getString(R.string.select_cates));
            }
            m_diaCates.show(getSupportFragmentManager(), ENT);
        }
    }
}
```

Caso de uso Inicio de sesión

LAYOUT para la vista de login

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:orientation="vertical"
    tools:context=".MainActivity">
    <TextView
        style="@style/fontGR"
        android:text="Acceso"
        android:textSize="40dp"
        android:id="@+id/tvAcceso" />
```

```
    <EditText
        style="@style/item"
        android:hint="Nombre usuario"
        android:id="@+id/etUser"   />
    <EditText
        style="@style/item"
        android:inputType="textPassword"
        android:id="@+id/etPass"
        android:hint="Password" />
    <TextView
        style="@style/error"
        android:id="@+id/tvError"/>
    <Button
        style="@style/btn"
        android:text="Entrar"
        android:onClick="logeo"
        android:id="@+id/btnEnviar" />
    <Button
        style="@style/link"
        android:onClick="formRegistro"
        android:text="Registrarse" />
    <Button
        style="@style/link"
        android:onClick="formFuente"
        android:text="Agregar feed RSS"/>
</LinearLayout>
```

Controlador para la vista de login

```
package com.camilo.rssnahuatl;

import android.content.Intent;
import android.os.Bundle;
import android.view.*;
import android.widget.*;
import com.camilo.entidadRelacion.UsuarioER;
import com.camilo.respuesta.rUsuarioER;
import com.camilo.rest.REST;
import com.camilo.utils.UTIL;
import retrofit.Callback;
import retrofit.RetrofitError;
import retrofit.client.Response;

public class Logeo extends Core {
    private TextView m_tvError;
    private EditText m_etUser,m_etPass;
    private Button m_btnEnviar;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
        iniciar();
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.menu_main, menu);
        return true;
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        int id = item.getItemId();
        if (id == R.id.action_settings) {
```

```
            return true;
        }

        return super.onOptionsItemSelected(item);
    }
    public void renderizacion(){
        m_etUser=(EditText)findViewById(R.id.etUser);
        m_etPass=(EditText)findViewById(R.id.etPass);
        m_tvError=(TextView)findViewById(R.id.tvError);
        m_btnEnviar=(Button)findViewById(R.id.btnEnviar);
        m_etUser.setText(get("etUser"));
    }
    public void logeo(View view){
        String username,password,msj;
        username=m_etUser.getText().toString();
        password=m_etPass.getText().toString();
        if(UTIL.isNotNull(username) && UTIL.isNotNull(password)){
            alertP("Acceso","Autentificando...");
            m_immInputManager.hideSoftInputFromWindow(view.getWindowToken(), 0);
            REST.get().usua().login(username, password, new Callback<rUsuarioER>() {
                @Override
                public void success(rUsuarioER rUsuarioER, Response response) {
                    setResponse(rUsuarioER);
                }
                @Override
                public void failure(RetrofitError error) {
                    setResponse(new rUsuarioER(false,error.getMessage(),null));
                }
            });
        } else{
            alertL("Ingresa la informacion necesaria");
        }
    }
    public void setResponse(rUsuarioER r){
        m_pdCargando.dismiss();
        if(r.isM_bStatus()){
            String uer=UTIL.toJson(r.getM_oUsuarioER());
            m_etPass.setText("");
            set("etUser", m_etUser.getText().toString());
            set("uer",uer);
            Intent i = new Intent(this, ListaAct.class );
            i.putExtra("action", "acceso");
            i.putExtra("uer", uer);
            startActivity(i);
        }else {
            m_tvError.setText(r.getM_sMsj());
        }
        alertS(r.getM_sMsj());
    }
    public void formRegistro(View view){
        set(getString(R.string.set_fuens), null);
        set(getString(R.string.set_cates), null);
        Intent i = new Intent(this, RegistroAct.class );
        startActivity(i);
    }
    public void formFuente(View view){
        Intent i = new Intent(this, FuenteAct.class );
        startActivity(i);
    }
}
```

A.5 Código fuente del módulo gestión de noticias

LAYOUT para la pantalla principal

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <Button
        android:id="@+id/btnList"
        style="@style/btn"
        android:text="Buton arriba"
        android:drawableRight="@android:drawable/arrow_down_float"
        android:onClick="btnClick"/>
    <ListView
        android:id="@+id/lvListaUI2"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:divider="#b5b5b5"
        android:dividerHeight="1dp"
        android:listSelector="@drawable/list_feed"></ListView>
</LinearLayout>
```

Controlador de la pantalla principal

```
package com.camilo.rssnahuatl;

import android.app.AlertDialog;
import android.app.Dialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.view.*;
import android.widget.*;
import com.camilo.adaptadores.*;
import com.camilo.entidadRelacion.*;
import com.camilo.modelo.*;
import com.camilo.respuesta.*;
import com.camilo.rest.REST;
import com.camilo.utils.UTIL;
import java.util.List;
import retrofit.Callback;
import retrofit.RetrofitError;
import retrofit.client.Response;

public class ListaAct extends Core {
    final static int DIALOG_BTN_SELECT_ACCESO=1;
    final static int DIALOG_BTN_SELECT_NEWS=2;
    private CharSequence[] opcionesAcceso;
    private String m_caso;
    private List m_lLista;
    private Button m_btnLista;
    private Dialog m_dSelect;
    private ListView m_lvLista;
    public UsuarioER m_uer;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_lista_noticias);
        iniciar();
    }
    @Override
```

```
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.menu_lista_noticias, menu);
        return true;
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        int id = item.getItemId();
        if (id == R.id.action_settings) {
            return true;
        }
        return super.onOptionsItemSelected(item);
    }
    public void renderizacion() {
        m_btnLista=(Button)findViewById(R.id.btnList);
        m_lvLista = (ListView) findViewById(R.id.lvListaUI2);
        m_caso=getExtra("action");
        m_uer=(UsuarioER)UTIL.fromJson(get("uer"),"UsuarioER");
        switch (m_caso){
            case "acceso":
                opcionesAcceso =
this.getResources().getStringArray(R.array.opciones_acceso);
                if(m_uer.getM_lFuentes()!=null && !m_uer.getM_lFuentes().isEmpty()){
                    setAccionUsuario(0);
                }else if(m_uer.getM_lCategorias()!=null &&
!m_uer.getM_lCategorias().isEmpty()){
                    setAccionUsuario(1);
                }else {
                    setAccionUsuario(2);
                }
                break;
            case "noticias":
                Integer idf=(Integer)this.getIntent().getExtras().get("idFuente");
                setNoticias(idf);
                break;
        }
    }
    public void btnClick(View v){
        switch(m_caso){
            case "acceso":
                if(m_dSelect==null){
                    m_dSelect=crearDialogo(DIALOG_BTN_SELECT_ACCESO);
                }
                m_dSelect.show();
                break;
        }
    }
    public void setFuentes(List<Fuente> lista){
        setListVista(new listaFuente(this, lista));
    }
    public void setCategorias(List<Categoria> lista) {
        setListVista(new listaCategoria(this,lista));
    }
    public void setNoticias(Integer idf) {
        alertP(getString(R.string.loading), getString(R.string.pet_noticias));
        REST.get().fuen().getById(idf, new Callback<rFuenteER>() {
            @Override
            public void success(rFuenteER rFuenteER, Response response) {
                m_pdCargando.dismiss();
                if (rFuenteER.isM_bStatus()) {
```

```
                    setListVista(new listaNoticia(ListaAct.this,
rFuenteER.getM_oFuenteER().getM_lNoticiaERs()));

m_btnLista.setText(rFuenteER.getM_oFuenteER().getM_oFuente().getFuentTitulo());
                }
                alertS(rFuenteER.getM_sMsj());
            }
            @Override
            public void failure(RetrofitError error) {
                m_pdCargando.dismiss();
                alertD(getString(R.string.msj_error),error.getMessage());
            }
        });
    }
    public void setListVista(BaseAdapter adaptador){
        m_lvLista.setAdapter(adaptador);
    }
    public void setAccionUsuario(int accion){
        if(accion >= 0 && accion < opcionesAcceso.length)
m_btnLista.setText(opcionesAcceso[accion]);
        switch(accion){
            case 0:
                setFuentes(m_uer.getM_lFuentes());
                break;
            case 1:
                setCategorias(m_uer.getM_lCategorias());
                break;
            case 2:
                alertP(getString(R.string.loading), getString(R.string.pet_fuens));
                REST.get().fuen().getTodas(new Callback<rFuentes>() {
                    @Override
                    public void success(rFuentes rFuentes, Response response) {
                        m_pdCargando.dismiss();
                        alertS(getString(R.string.pet_fuens_success));
                        setFuentes(rFuentes.getM_lFuentes());
                    }

                    @Override
                    public void failure(RetrofitError error) {
                        m_pdCargando.dismiss();
                        alertD(getString(R.string.msj_error), error.getMessage());
                    }
                });
                break;
            case 3:
                alertP(getString(R.string.loading), getString(R.string.pet_cates));
                REST.get().cate().getAll(new Callback<rCategorias>() {
                    @Override
                    public void success(rCategorias obj, Response response) {
                        m_pdCargando.dismiss();
                        alertS(getString(R.string.pet_cates_success));
//                            set(getString(R.string.set_cates), UTIL.toJson(obj));
                        setCategorias(obj.getM_lCategorias());
                    }

                    @Override
                    public void failure(RetrofitError error) {
                        m_pdCargando.dismiss();
                        alertD(getString(R.string.msj_error), error.getMessage());
                    }
```

```
                });
                break;
        }
    }
    protected Dialog crearDialogo(int id) {
        switch (id) {
            case DIALOG_BTN_SELECT_ACCESO:
                return new AlertDialog.Builder(this)
                        .setTitle(R.string.select_single_option)
                        .setSingleChoiceItems(opcionesAcceso, -1, new
DialogInterface.OnClickListener() {
                            @Override
                            public void onClick(DialogInterface dialog, int which) {
                                dialog.dismiss();
                                setAccionUsuario(which);
                            }
                        })
                        .setNegativeButton(android.R.string.cancel, new
DialogInterface.OnClickListener() {
                            @Override
                            public void onClick(DialogInterface dialog, int which) {
                                dialog.dismiss();

                            }
                        }).create();
            default:  return null;
        }
    }
}
```

Caso de uso seleccionar categoría

LAYOUT para cada ítem categoría

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:background="@drawable/list_category"
    android:orientation="horizontal"
    android:padding="5dip">
    <LinearLayout
        android:id="@+id/llThumbnailItem"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_marginRight="5dip"
        android:padding="3dip">
        <ImageView
            android:id="@+id/ivIconItem"
            android:layout_width="60dip"
            android:layout_height="60dip"
            android:contentDescription="@string/app_name"
            android:src="@drawable/icon_category" />
    </LinearLayout>
    <TextView
        android:id="@+id/tvGrandeItem"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerVertical="true"
        android:layout_marginLeft="5dip"
        android:layout_toRightOf="@+id/llThumbnailItem"
        android:paddingBottom="10dip"
```

```
        android:text="Grande"
        android:textSize="20dip"
        android:textStyle="bold"
        android:typeface="sans" />
    <TextView
        android:id="@+id/tvChicaItem"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/tvGrandeItem"
        android:layout_below="@+id/tvGrandeItem"
        android:layout_centerHorizontal="true"
        android:paddingTop="5dip"
        android:text="Descripción"
        android:textColor="#343434"
        android:textSize="15dip" />
    <ImageView
        android:id="@+id/ivArrowItem"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true"
        android:layout_centerVertical="true"
        android:contentDescription="@string/app_name"
        android:src="@drawable/arrow_02" />
</RelativeLayout>
```

Controlador del adaptador de lista categorías

```
package com.camilo.adaptadores;

import android.view.*;
import android.widget.*;
import com.camilo.modelo.*;
import com.camilo.respuesta.rFuentes;
import com.camilo.rest.REST;
import com.camilo.rssnahuatl.Core;
import com.camilo.rssnahuatl.ListaAct;
import com.camilo.rssnahuatl.R;
import java.util.List;
import retrofit.Callback;
import retrofit.RetrofitError;
import retrofit.client.Response;

public class listaCategoria extends BaseAdapter implements View.OnClickListener {
    private static final String ENT = "listaCategoria";
    private Core m_aPadre;
    private List<Categoria> m_lObjeto;

    public static class RenglonHolder {
        public TextView m_tvGrande,m_tvChico;
        public ImageView m_ivIcon,m_ivArrow;
        public Categoria  m_obj;
    }
    public listaCategoria(Core a, List<Categoria> data) {
        this.m_aPadre = a;
        this.m_lObjeto = data;
    }
    @Override
    public int getCount() {
        return this.m_lObjeto.size();
    }
    @Override
    public boolean areAllItemsEnabled() {
```

```
        return true;
    }
    @Override
    public Categoria getItem(int posicion) {
        return m_lObjeto.get(posicion);
    }
    @Override
    public long getItemId(int pos) {
        return pos;
    }
    @Override
    public View getView(int pos, View convertView, ViewGroup parent) {
        RenglonHolder holder;
        if (convertView == null) {
            convertView = LayoutInflater.from(m_aPadre).inflate(R.layout.item_category,
parent, false);
            holder = new RenglonHolder();
            holder.m_tvGrande = (TextView) convertView.findViewById(R.id.tvGrandeItem);
            holder.m_tvChico = (TextView) convertView.findViewById(R.id.tvChicaItem);
            holder.m_ivIcon = (ImageView) convertView.findViewById(R.id.ivIconItem);
            holder.m_ivArrow = (ImageView) convertView.findViewById(R.id.ivArrowItem);
            convertView.setTag(holder);
        } else {
            holder = (RenglonHolder) convertView.getTag();
        }
        convertView.setOnClickListener(this);
        Categoria obj = m_lObjeto.get(pos);
        holder.m_obj = obj;
        holder.m_tvGrande.setText(obj.getCateNombre());
        holder.m_tvChico.setText(obj.getCateDescripcion());
        return convertView;
    }
    @Override
    public void onClick(View v) {
        final RenglonHolder h = (RenglonHolder) v.getTag();
        m_aPadre.alertP(m_aPadre.getString(R.string.loading),
m_aPadre.getString(R.string.pet_fuens));
        REST.get().fuen().getByCategoria(h.m_obj.getCateId(), new Callback<rFuentes>() {
            @Override
            public void success(rFuentes rFuentes, Response response) {
                m_aPadre.m_pdCargando.dismiss();
                m_aPadre.alertS(rFuentes.getM_sMsj());
                if(rFuentes.isM_bStatus()){
                    ((ListaAct)m_aPadre).setFuentes(rFuentes.getM_lFuentes());
                }
            }
            @Override
            public void failure(RetrofitError error) {
                m_aPadre.m_pdCargando.dismiss();
                m_aPadre.alertD(m_aPadre.getString(R.string.msj_error),
error.getMessage());
            }
        });
    }
}
```

Caso de uso seleccionar fuente

LAYOUT para cada ítem fuente

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
```

```
    android:layout_height="wrap_content"
    android:background="@drawable/list_feed"
    android:orientation="horizontal"
    android:padding="5dip">
    <LinearLayout
        android:id="@+id/llThumbnailItem"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_marginRight="5dip"
        android:padding="3dip">
        <ImageView
            android:id="@+id/ivIconItem"
            android:layout_width="60dip"
            android:layout_height="60dip"
            android:contentDescription="@string/app_name"
            android:src="@drawable/icon_rss" />
    </LinearLayout>
    <TextView
        android:id="@+id/tvGrandeItem"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerVertical="true"
        android:layout_marginLeft="5dip"
        android:layout_toRightOf="@+id/llThumbnailItem"
        android:paddingBottom="10dip"
        android:text="Grande"
        android:textSize="20dip"
        android:textStyle="bold"
        android:typeface="sans" />
    <TextView
        android:id="@+id/tvChicaItem"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/tvGrandeItem"
        android:layout_below="@+id/tvGrandeItem"
        android:layout_centerHorizontal="true"
        android:paddingTop="5dip"
        android:text="Descripción"
        android:textColor="#343434"
        android:textSize="15dip" />
    <ImageView
        android:id="@+id/ivArrowItem"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true"
        android:layout_centerVertical="true"
        android:contentDescription="@string/app_name"
        android:src="@drawable/arrow_02" />
</RelativeLayout>
```

Controlador para el adaptador de la lista fuentes

```
package com.camilo.adaptadores;

import android.content.Intent;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.*;
import com.camilo.modelo.Fuente;
import com.camilo.rssnahuatl.Core;
```

```
import com.camilo.rssnahuatl.ListaAct;
import com.camilo.rssnahuatl.R;
import java.util.List;

public class listaFuente extends BaseAdapter implements View.OnClickListener {
    private static final String ENT = "listaFuente";
    private Core m_aPadre;
    private List<Fuente> m_lObjeto;

    public static class RenglonHolder {
        public TextView m_tvGrande,m_tvChico;
        public ImageView m_ivIcon,m_ivArrow;
        public Fuente m_oFuente;
    }
    public listaFuente(Core a, List<Fuente> data) {
        this.m_aPadre = a;
        this.m_lObjeto = data;
    }
    @Override
    public int getCount() {
        return this.m_lObjeto.size();
    }
    @Override
    public boolean areAllItemsEnabled() {
        return true;
    }
    @Override
    public Fuente getItem(int posicion) {
        return m_lObjeto.get(posicion);
    }
    @Override
    public long getItemId(int pos) {
        return pos;
    }
    @Override
    public View getView(int pos, View convertView, ViewGroup parent) {
        RenglonHolder holder;
        if (convertView == null) {
            convertView = LayoutInflater.from(m_aPadre).inflate(R.layout.item_feed,
parent, false);
            holder = new RenglonHolder();
            holder.m_tvGrande = (TextView) convertView.findViewById(R.id.tvGrandeItem);
            holder.m_tvChico = (TextView) convertView.findViewById(R.id.tvChicaItem);
            holder.m_ivIcon = (ImageView) convertView.findViewById(R.id.ivIconItem);
            holder.m_ivArrow = (ImageView) convertView.findViewById(R.id.ivArrowItem);
           /* holder.m_ivIcon.setOnClickListener(this);
            holder.m_ivArrow.setOnClickListener(this);*/
            convertView.setTag(holder);
        } else {
            holder = (RenglonHolder) convertView.getTag();
        }

        convertView.setOnClickListener(this);

        Fuente obj = m_lObjeto.get(pos);
        holder.m_oFuente = obj;

        holder.m_tvGrande.setText(obj.getFuentTitulo());
        holder.m_tvChico.setText(obj.getFuentDescripcion());
```

```
        return convertView;
    }
    @Override
    public void onClick(View v) {
        RenglonHolder holder = (RenglonHolder) v.getTag();
        Intent i = new Intent(m_aPadre, ListaAct.class );
        i.putExtra("action", "noticias");
        i.putExtra("idFuente", holder.m_oFuente.getFuentId());
        m_aPadre.startActivity(i);
    }
}
```

Caso de uso seleccionar noticia

LAYOUT para cada ítem noticia

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:background="@drawable/list_news"
    android:orientation="horizontal"
    android:padding="5dip">
    <LinearLayout
        android:id="@+id/llThumbnailItem"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_marginRight="5dip"
        android:padding="3dip">

        <ImageView
            android:id="@+id/ivIconItem"
            android:layout_width="60dip"
            android:layout_height="60dip"
            android:contentDescription="@string/app_name"
            android:src="@drawable/icon_news" />
    </LinearLayout>
    <TextView
        android:id="@+id/tvGrandeItem"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="5dip"
        android:layout_toRightOf="@+id/llThumbnailItem"
        android:paddingBottom="10dip"
        android:text="Grandeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee"
        android:textSize="20dip"
        android:textStyle="bold"
        android:typeface="sans" />
    <TextView
        android:id="@+id/tvChicaItem"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/tvGrandeItem"
        android:layout_below="@+id/tvGrandeItem"
        android:layout_centerHorizontal="true"
        android:paddingTop="5dip"
        android:text="Descripciónnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn"
        android:textColor="#343434"
        android:textSize="15dip" />
    <RatingBar
        android:id="@+id/rbListNews"
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/tvChicaItem"
        android:layout_alignLeft="@+id/tvChicaItem"
        android:isIndicator="true"/>
</RelativeLayout>
```

Controlador para el adaptador de la lista noticias

```
package com.camilo.adaptadores;

import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.net.Uri;
import android.view.*;
import android.widget.*;
import com.camilo.entidadRelacion.NoticiaER;
import com.camilo.modelo.*;
import com.camilo.rest.REST;
import com.camilo.rssnahuatl.Core;
import com.camilo.rssnahuatl.ListaAct;
import com.camilo.rssnahuatl.R;
import java.util.List;
import retrofit.Callback;
import retrofit.RetrofitError;
import retrofit.client.Response;

public class listaNoticia extends BaseAdapter implements View.OnClickListener {
    private static final String ENT = "listaNoticia";
    private Core m_aPadre;
    private List<NoticiaER> m_lObjeto;

    public static class RenglonHolder {
        public TextView m_tvGrande,m_tvChico;
        public ImageView m_ivIcon,m_ivArrow;
        public  RatingBar m_rbCalificacion;
        public NoticiaER  m_obj;
    }
    public listaNoticia(Core a, List<NoticiaER> data) {
        this.m_aPadre = a;
        this.m_lObjeto = data;
    }
    @Override
    public int getCount() {
        return this.m_lObjeto.size();
    }
    @Override
    public boolean areAllItemsEnabled() {
        return true;
    }
    @Override
    public NoticiaER getItem(int posicion) {
        return m_lObjeto.get(posicion);
    }
    @Override
    public long getItemId(int pos) {
        return pos;
    }
    @Override
    public View getView(int pos, View convertView, ViewGroup parent) {
        RenglonHolder holder;
```

```
        if (convertView == null) {
            convertView = LayoutInflater.from(m_aPadre).inflate(R.layout.item_news,
parent, false);
            holder = new RenglonHolder();
            holder.m_tvGrande = (TextView) convertView.findViewById(R.id.tvGrandeItem);
            holder.m_tvChico = (TextView) convertView.findViewById(R.id.tvChicaItem);
            holder.m_ivIcon = (ImageView) convertView.findViewById(R.id.ivIconItem);
            holder.m_ivArrow = (ImageView) convertView.findViewById(R.id.ivArrowItem);
            holder.m_rbCalificacion = (RatingBar)
convertView.findViewById(R.id.rbListNews);
            convertView.setTag(holder);
        } else {
            holder = (RenglonHolder) convertView.getTag();
        }

        convertView.setOnClickListener(this);
        NoticiaER obj = m_lObjeto.get(pos);
        holder.m_obj = obj;
        Corpus titu=obj.getM_oCorpusTitulo();
        holder.m_tvGrande.setText(titu.getCorpEsp());
        holder.m_tvChico.setText(titu.getCorpNah());
        holder.m_rbCalificacion.setRating(obj.getM_iCalificacion());
        return convertView;
    }
}
```

Caso de uso calificar noticia

LAYOUT para la vista de la noticia

```
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:background="@drawable/list_feed"
        android:orientation="vertical">
        <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:layout_weight="3">
        <TextView
            android:id="@+id/tvTituEsp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Push the button"
            style="@android:style/TextAppearance.Large"
            android:layout_gravity="center_horizontal" />
        <TextView
            android:id="@+id/tvDescEsp"
            style="@android:style/TextAppearance.Medium"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Push the button"  />
        </LinearLayout>
        <View style="@style/separador" />
        <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
```

```
        android:layout_weight="3">
        <TextView
            android:id="@+id/tvTituNah"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Push the button"
            style="@android:style/TextAppearance.Large"
            android:layout_gravity="center_horizontal"
            android:textStyle="bold"/>
        <TextView
            android:id="@+id/tvDescNah"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Push the button"
            style="@android:style/TextAppearance.Medium"
            android:textStyle="bold"/>
        </LinearLayout>
        <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:layout_weight="1">
        <View style="@style/separador" />
        <RatingBar
            android:id="@+id/rbCalificacion"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal"
            android:numStars="5"
            android:stepSize="1" />
        <TextView
            android:id="@+id/tvNombre"
            style="@style/fontGR"
            android:text="Ayudanos con una calificacion" />
        </LinearLayout>
    </LinearLayout>
</ScrollView>
```

Controlador del evento clic en la calificación

```
      @Override
      public void onClick(View v) {
        final Usuario user=((ListaAct)m_aPadre).m_uer.getM_oUsuario();
        final RenglonHolder h = (RenglonHolder) v.getTag();
        View vNoticia;
        if(user.isSiNah()){
            final RatingBar calificacion;
            vNoticia = (LayoutInflater.from(m_aPadre)).inflate(R.layout.noticia_nah,
null);
            calificacion=(RatingBar) vNoticia.findViewById(R.id.rbCalificacion);
            calificacion.setOnRatingBarChangeListener(new
RatingBar.OnRatingBarChangeListener() {
                @Override
                public void onRatingChanged(RatingBar ratingBar, float rating, boolean
fromUser) {
                    if(fromUser){
                        Calificacion c=new Calificacion();
                        c.setCaliIdNoticia(h.m_obj.getM_oNoticia().getNotiId());
                        c.setCaliIdUsuario(user.getUsuaId());
                        c.setCaliValor((int)rating);
                        REST.get().cali().set(c, new Callback<Integer>() {
                            @Override
```

```
                            public void success(Integer integer, Response response) {
                                if(integer!= -1){
                                    calificacion.setRating(integer);

m_aPadre.alertS(m_aPadre.getString(R.string.set_cali));
                                    h.m_obj.setM_iCalificacion(integer);
                                    h.m_rbCalificacion.setRating(integer);
                                }
                            }
                            @Override
                            public void failure(RetrofitError error) {
                                m_aPadre.alertS(error.toString());
                                calificacion.setRating(0);
                            }
                        });
                    }
                }
            });
        }else{
            vNoticia = (LayoutInflater.from(m_aPadre)).inflate(R.layout.noticia, null);
        }
        ((TextView)
vNoticia.findViewById(R.id.tvTituEsp)).setText(h.m_obj.getM_oCorpusTitulo().getCorpEsp())
;
        ((TextView)
vNoticia.findViewById(R.id.tvTituNah)).setText(h.m_obj.getM_oCorpusTitulo().getCorpNah())
;
        ((TextView)
vNoticia.findViewById(R.id.tvDescEsp)).setText(h.m_obj.getM_oCorpusDescripcion().getCorpE
sp());
        ((TextView)
vNoticia.findViewById(R.id.tvDescNah)).setText(h.m_obj.getM_oCorpusDescripcion().getCorpN
ah());
        AlertDialog.Builder builder = new AlertDialog.Builder(m_aPadre);
        builder.setView(vNoticia)
                .setPositiveButton(m_aPadre.getString(R.string.visit), new
DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int id) {
                        Intent intent = new Intent(android.content.Intent.ACTION_VIEW,
Uri.parse(h.m_obj.getM_oNoticia().getNotiLink()));
                        m_aPadre.startActivity(intent);

                    }
                })
                .setNegativeButton(m_aPadre.getString(R.string.close), null);
        builder.create().show();
    }
```

B. Manual de usuario

Índice de ilustraciones

Propósito del documento

El propósito de este apartado es para entregar las pautas de operación de la aplicación móvil del presente proyecto, esta aplicación permite la creación de usuarios, inicio de sesión y consulta de noticias traducidas.

La consulta de noticias para cualquier persona eventualmente ocurre ya sea para consultar un asunto en particular o simplemente para estar informado en todo momento, por lo que se requiere una herramienta flexible, accesible y amigable con el usuario. Existen muchas aplicaciones lectores RSS, una de ellas es Feedly, para dispositivos Android, Apple IOS y Web.

Feedly permite buscar canales RSS por medio del título, URL o tema. También sugiere varios temas cuando se ingresa por primera vez, otra de las posibilidades es seleccionar un canal de las listas creadas por la comunidad de usuarios. Si algún canal RSS es de interés entonces se puede agregar a la cuenta y la aplicación los va clasificando por tema a demás proporciona la posibilidad de crear una lista de canales personalizada.
Feedly es un lector RSS muy bien desarrollado pensado para aprovechar las características de la sindicación web.

Conceptos importantes

Aplicación móvil

La aplicación móvil para la gestión de noticias en español con traducción al náhuatl fue desarrollado en Android. Está dirigido para todas las personas y tiene la misma funcionalidad para todos, excepto para los usuarios que hablan español y náhuatl, usuarios de tipo dos, para los cuales se agrega una funcionalidad más, calificar la traducción de la noticia.

Cómo empezar

Para instalar la aplicación primero se tiene que habilitar la opción “Fuentes desconocidas”, esto se hace en “ajustes” en la opción “seguridad”, ver Ilustración 1.

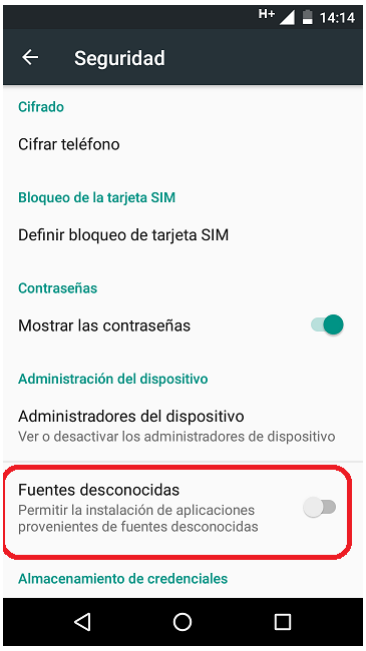


Ilustración 1: Activar fuentes desconocidas

Características de la aplicación

Funcionalidades

1. Autentificación de usuarios.
2. Registro de usuarios.
3. Listar categorías y seleccionar una categoría.
4. Listar fuentes y seleccionar una fuente.
5. Listar noticias y seleccionar una noticia
6. Calificar traducción de la noticia (disponible solo para usuarios de tipo dos).

Autentificación de usuarios.

Una persona quiere ingresar al sistema y ya tiene sus credenciales

Debe ingresar el nombre de usuario y la contraseña, posteriormente pulsar el botón “ENTRAR”, ver Ilustración 2.

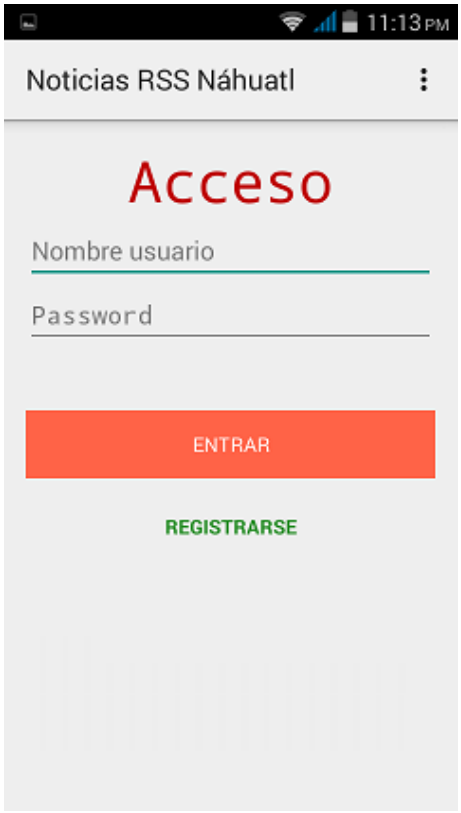


Ilustración 2: Pantalla de acceso

En el caso de que las credenciales fueron ingresadas, entonces se verifica la identificación del usuario y si resultan incorrectas, se muestra un mensaje de error, ver Ilustración 3.

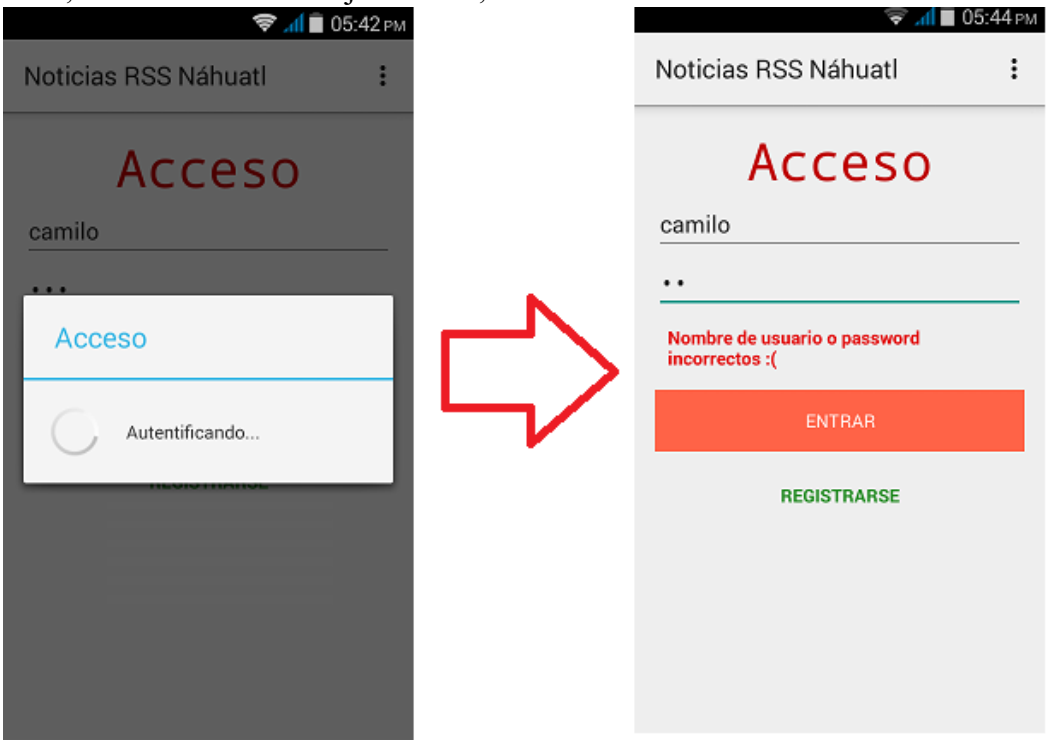


Ilustración 3: Inicio de sesión fallido

Una persona quiere ingresar al sistema y no tiene credenciales

En la pantalla de autentificación hay un link "REGISTRARSE" para llegar al formulario de registro, ver Ilustración 4.

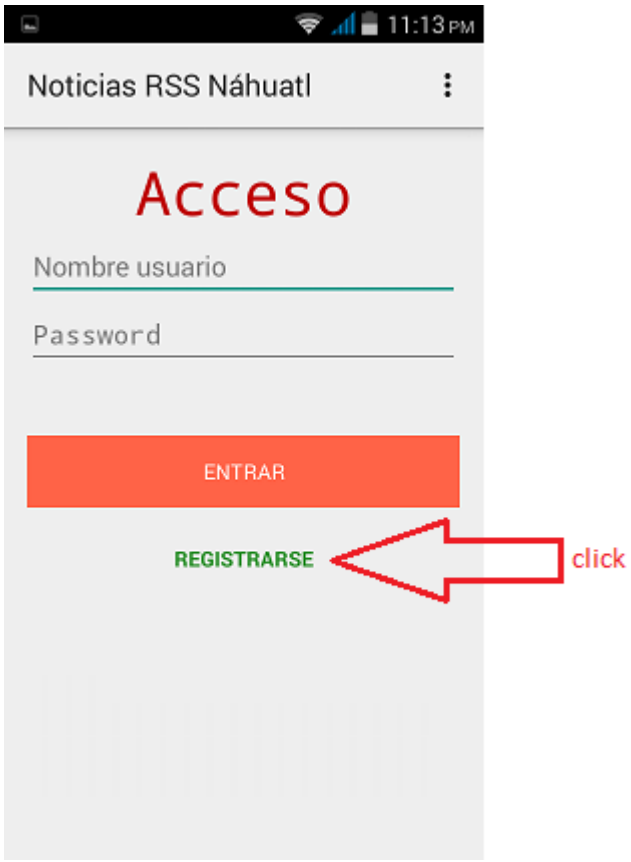


Ilustración 4: Link registrase

En el formulario se pide la información siguiente, ver Ilustración 5:

- Nombre completo de la persona.
- Nombre de usuario.
- Contraseña.
- Si habla español o náhuatl.
- Fuentes y categorías de preferencia.

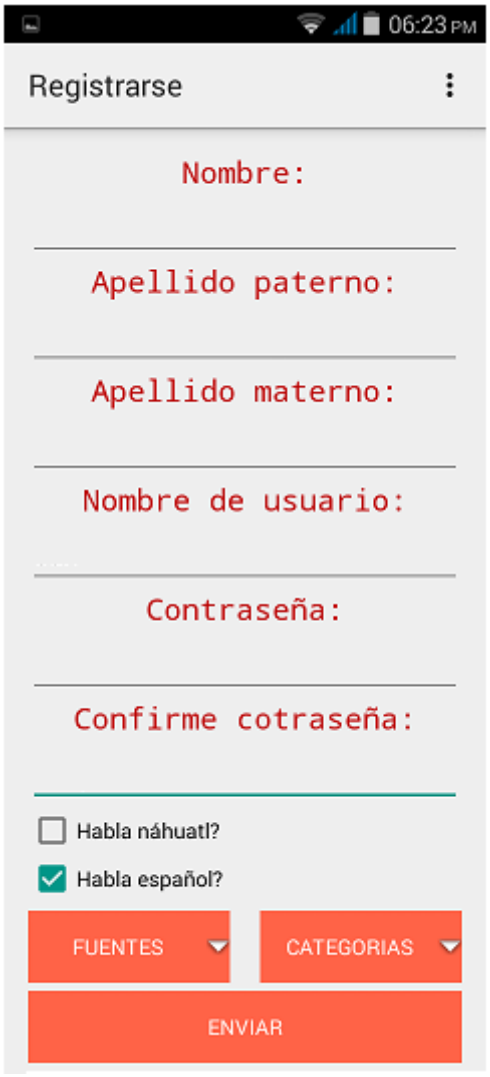


Ilustración 5: Datos requeridos en el registro.

Todos los datos son obligatorios y además algunos campos como el nombre de usuario y la contraseña deben ser ingresados con las siguientes características, solo caracteres alfanuméricas y con una longitud de cuatro a doce caracteres, ver Ilustración 6. Si no se cumple con lo mencionado, se notifica con un mensaje en la parte inferior del dispositivo, solo las categorías y fuentes no son obligatorias. Cuando toda la información se captura, se envía para guardarlo.

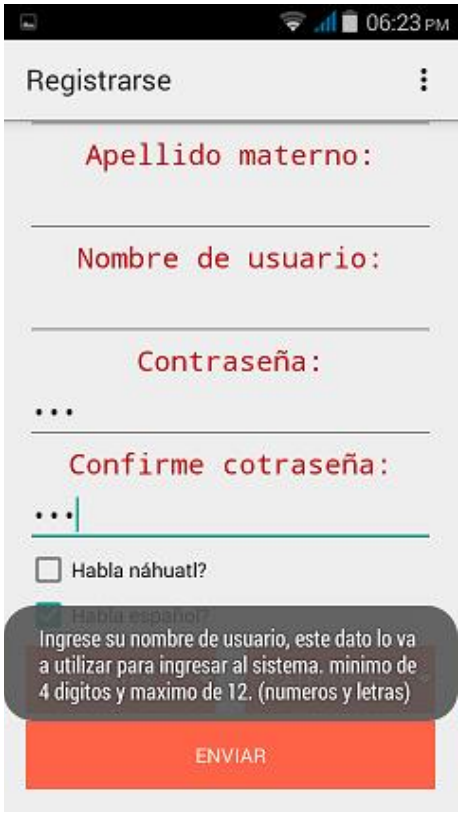


Ilustración 6: Formato de nombre usuario y contraseña

En el caso ideal, es decir si se almacenó la información sin ningún inconveniente, aparece “usuario nuevo” y se regresa a la pantalla de acceso de la aplicación, ver Ilustración 7.

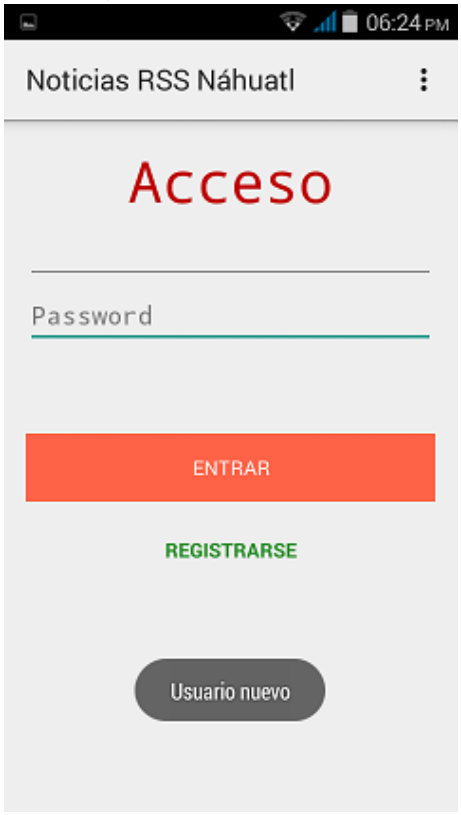


Ilustración 7: Fin exitoso de registro.

Si por alguna razón ocurre un error, entonces se mantiene en la misma pantalla otorgándole al usuario una oportunidad más para guardar sus datos.

Listar categorías y seleccionar una categoría

Esta funcionalidad permite buscar fuentes partiendo de la categoría, es decir cuando se selecciona una categoría se hace una búsqueda de fuentes que pertenecen a esa categoría para finalmente mostrarlos en la pantalla del dispositivo, ver Ilustración 8.

Ilustración 8: Fuentes pertenecientes a una categoría

Una persona escogió una lista de categorías como preferidas al momento de registrarse

Esta lista puede visualizarse cuando se selecciona categorías preferidas en la lista desplegable que se encuentra en la parte superior de la pantalla del dispositivo, inmediatamente se muestra la lista y se puede seleccionar una de las categorías, ver Ilustración 9.

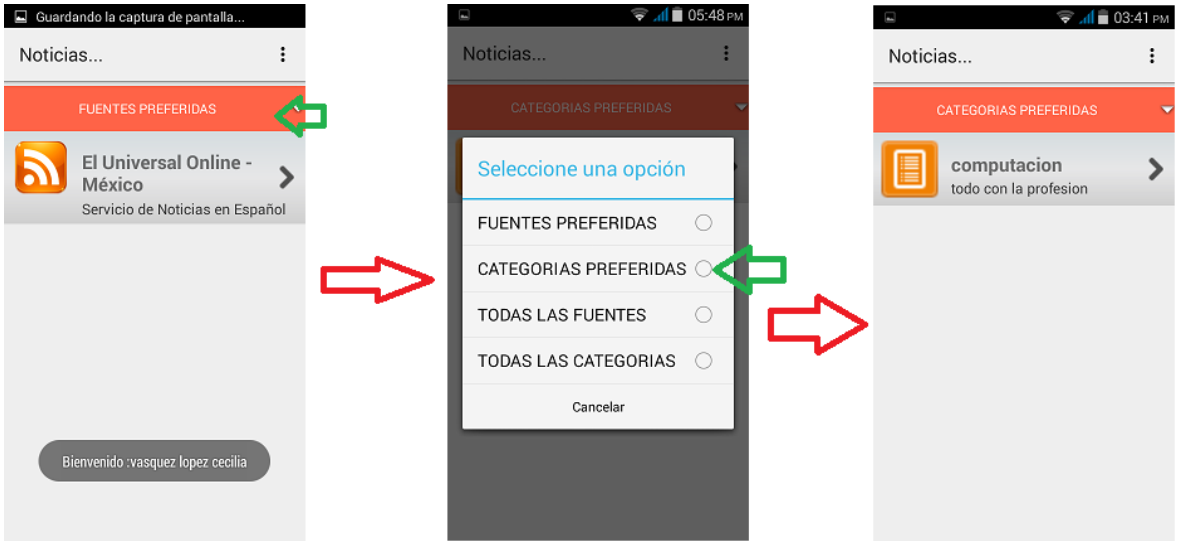


Ilustración 9: Categorías preferidas

Una persona no escogió ninguna categoría como preferida al momento de registrarse

En este caso al seleccionar "Categorías preferidas" no aparece ninguna opción sin embargo si la persona desea buscar noticias partiendo de la categoría tiene la posibilidad de escoger en la lista desplegable la opción "Todas las categorías" para visualizar todas las categorías registradas en el sistema para finalmente seleccionar una de ellas.

Ilustración 10: Todas las categorías

Listar fuentes y seleccionar una fuente

La finalidad de esta funcionalidad es mostrar noticias pertenecientes a una fuente, Ilustración 11.

Ilustración 11: Petición de noticias de una fuente

Una persona escogió una lista de fuentes como preferidas al momento de registrarse

Esta lista es la que se muestra cuando la persona inicia sesión, es la pantalla principal, también puede visualizarse cuando se selecciona “Fuentes preferidas” en la lista desplegable que se encuentra en la parte superior de la pantalla, inmediatamente se muestra la lista y se puede seleccionar una de las fuentes.

Ilustración 12: Fuentes preferidas

Una persona no escogió ninguna fuente como preferida al momento de registrarse

En este caso al seleccionar “Fuentes preferidas” no aparece ninguna opción sin embargo se tiene la posibilidad de escoger “Todas las fuentes” para visualizar todas las fuentes existentes en el sistema para finalmente seleccionar una de ellas, ver Ilustración 13.

Ilustración 13: Todas las fuentes

Listar noticias y seleccionar una noticia
Al momento de seleccionar una de las fuentes, se visualiza la lista de noticias, mostrando solo el título en español y en náhuatl dando la posibilidad de seleccionar una de ellas para verlo completo, Ilustración 14.

Ilustración 14: Lista de noticias

Una persona le interesa una noticia dentro de la lista

Para ver completa la noticia basta con seleccionarla e inmediatamente abre una ventana para mostrar la noticia en español y en náhuatl primero el título seguido por el contenido de arriba-abajo y en la parte inferior, dos botones, una para cerrar la ventana y la otra para abrir el navegador con el fin de consultar la noticia en el sitio web de la fuente, ver Ilustración 15.

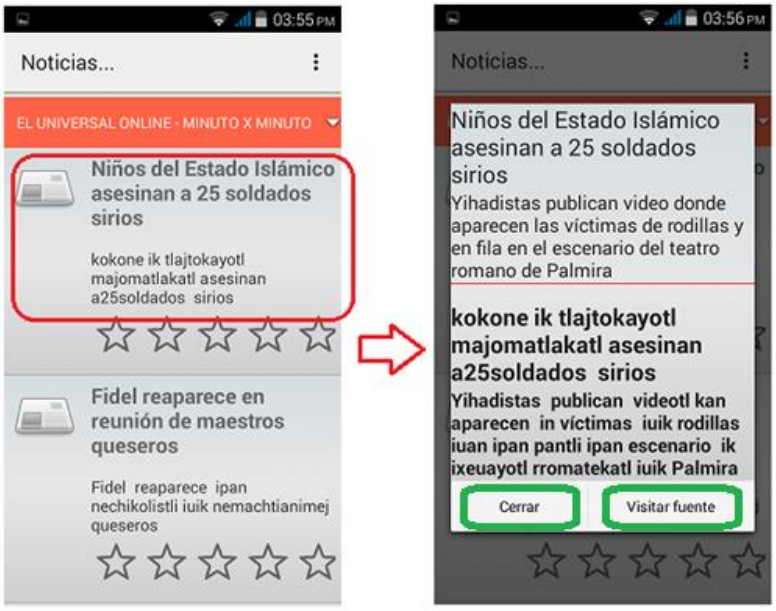


Ilustración 15: Noticia completa

Calificar noticia (solo para usuarios tipo dos)

Cuando ingresa un usuario que sabe español y náhuatl al visualizar una noticia completa, en la parte de abajo se muestra una sección para establecer una calificación presionando una de las estrellitas, ver Ilustración 16.

Ilustración 16: Calificación noticia