EDITOR PREDICTIVO DE MENSAJES EN PICTOGRAMAS

Curso 2013 / 2014

Trabajo de Fin de Grado en Ingeniería del Software

Realizado por:

Carlos Ruiz Martín

Paloma Galván Calleja

Dirigido por:

Raquel Hervás Ballesteros

Pablo Gervás Gómez-Navarro

Dpto. de Ingeniería del Software e Inteligencia Artificial

Facultad de Informática

Universidad Complutense de Madrid

Se autoriza a la Universidad Complutense de Madrid a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la memoria como el código, la documentación y/o el prototipo desarrollado.

Carlos Ruíz Martín Paloma Galván Calleja

Resumen

Actualmente existen numerosas enfermedades con las que el ser humano tiene que lidiar. Algunas de estas enfermedades generan una discapacidad seria en la persona en cuestión, a veces intelectual y otras veces física. Estas discapacidades suponen barreras constantes en el día a día de la persona afectada, ya no solo a nivel personal sino también al relacionarse con otras personas.

Para cubrir las necesidades comunicativas de las personas con discapacidad se inventaron los Sistemas Aumentativos y Alternativos de Comunicación (SAAC), que sin hacer uso de palabras articuladas son capaces de proporcionar los mecanismos para establecer una comunicación.

El presente proyecto hace frente a un problema de accesibilidad haciendo uso de uno de estos sistemas: los pictogramas. Los pictogramas son imágenes que representan conceptos de todo tipo independientemente del idioma. Existen varios conjuntos o sistemas de pictogramas que comprenden desde ilustraciones en color o en blanco y negro, hasta fotografías de objetos concretos o signos del lenguaje de signos.

Por otro lado, vivimos en una sociedad de la información y la comunicación en la que el uso dispositivos electrónicos como smartphones y tablets está cada día más extendido.

El propósito del presente proyecto es el de desarrollar una aplicación para dispositivos móviles mediante la cual el usuario sea capaz de construir frases usando dichos pictogramas, ayudándole en dicha tarea a través de un mecanismo de predicción.

Abstract

Nowadays, there are several diseases that humanity has to deal with. Some of these diseases produce important disabilities, intellectual or physical. These disabilities are recurrent barriers in the daily life of the affected person, not only in a personal level, but when communicating with people as well.

The Augmentative and Alternative Communication Systems were created to help these people to communicate. These systems do not use spoken words. Instead they provide different mechanisms to establish a communication.

This project faces this accessibility issue using one of these systems: pictograms. Pictograms are images that represent, regardless of the language, all kind of things such as objects, actions or ideas. There are several pictogram systems or sets, which have both color pictograms and black and white pictograms. These sets can also contain pictures of real objects and pictures of the sign language.

Besides, we live in a society of information and communication where the use of mobile devices such as smartphones and tablets is more extended every day.

The purpose of the present project is to develop a mobile application through which the user could create sentences using pictograms. This application would help the user to create these sentences using a predictive mechanism.

Palabras claves

- Sistemas alternativos y aumentativos de comunicación (SAACs)
- Pictograma
- Android
- Edición predictiva
- Servicio web
- Accesibilidad

Keywords

- Augmentative and alternative communication (AAC)
- Pictogram
- Android
- Predictive edition
- Web service
- Accessibility

Índice General

Índice de ilustraciones

Capítulo I

Introducción

I. Introducción

1.1 **Motivación**

En la actualidad vivimos en una sociedad que bien se podría definir, entre otras cosas, como sociedad de la información y las comunicaciones. Cada vez se hace más indispensable una comunicación rápida como prueba la proliferación de aplicaciones móviles de mensajería instantánea, sistemas de correo electrónico y redes de telecomunicaciones. La comunicación fácil y rápida está a la orden del día en el ámbito empresarial pero también lo está cada vez más en la vida cotidiana de las personas.

La comunicación se convierte en algo indispensable en el día a día de cualquiera (contacto con la familia, comunicación con amigos y planificación de eventos, etc.), pero hay ciertos grupos de personas que sufren enfermedades que impiden o dificultan en gran medida esta comunicación con el mundo exterior en general.

Hablamos de enfermos de Trastornos del Espectro Autista, Parálisis Cerebral, Esclerosis Lateral Amiotrófica o Esclerosis Múltiple. Para estas personas, la comunicación puede suponer un reto, así como para el resto de personas que pudiesen necesitar comunicarse con ellas como por ejemplo médicos y profesores.

De esta necesidad, surgieron los Sistemas Aumentativos y Alternativos de Comunicación y en particular los pictogramas. Estos pictogramas son imágenes que representan conceptos de todo tipo independientemente del idioma. Existen varios conjuntos o sistemas de pictogramas que comprenden desde ilustraciones en color o en blanco y negro, hasta fotografías de objetos concretos o signos del lenguaje de signos.

El propósito del presente proyecto es el de desarrollar una aplicación mediante la cual el usuario sea capaz de construir frases usando dichos pictogramas, ayudándole en dicha tarea a través de un mecanismo de predicción.

La predicción es un punto importante ya que en este caso no se trata solo de una comodidad, sino que en muchas ocasiones el usuario puede no disponer de la capacidad motriz o intelectual para manejar la aplicación y encontrar el pictograma buscado como lo haría una persona que no padece ninguna dolencia de este tipo. Así, la predicción proporcionada por el sistema hace más fácil la búsqueda de los siguientes pictogramas a utilizar cuando se está escribiendo una frase.

1.2 Objetivos

En vista de las necesidades comunicativas de las personas que sufren estas enfermedades se han establecido los siguientes objetivos:

- Estudio de las plataformas disponibles y elección de la más adecuada para este caso.
- Elección de un sistema pictográfico lo más extendido posible.
- Obtención y reestructuración si fuese necesario de una base de datos de pictogramas del sistema elegido.
- Diseño de una arquitectura que combine las tecnologías elegidas con un buen rendimiento.
- Diseño de un algoritmo de predicción útil e implementación del editor que usa dicho algoritmo.
- Diseñar una interfaz especialmente cómoda teniendo en cuenta el modelo de usuario objetivo de la aplicación.
- Publicación de la aplicación en algún tipo de repositorio o tienda para su difusión.
- Elaboración de funciones que permitan el envío, guardado y compartición de los mensajes compuestos mediante la aplicación para así ayudar en la comunicación.

1.3 Estructura del documento

Este documento sigue una estructura similar a la seguida durante el desarrollo del proyecto.

En primer lugar, en el Capítulo 3 se hace un repaso de las tecnologías usadas para el desarrollo de la aplicación así como de la situación actual en el desarrollo de aplicaciones de este estilo. En este caso se hablará de los SAAC, pictogramas, Android y servicios web.

Una vez conocido el contexto de la problemática y las tecnologías, en el Capítulo 4 se hace una aproximación a la arquitectura de nuestro sistema, introduciendo cada uno de los componentes principales que se desarrollarán más en detalle en capítulos posteriores.

En el Capítulo 5 se hará un recorrido por el proceso de transformación y organización de la base de datos de pictogramas.

El Capítulo 6 expone el servicio web que permite la comunicación entre el cliente de la aplicación y la base de datos de pictogramas.

El cliente de la aplicación PictoEditor, su funcionalidad y los detalles de su implementación en Android se exponen en el Capítulo 7.

En el capítulo 8 se exponen las aportaciones de los miembros del grupo a lo largo del proceso de desarrollo.

Por último, en los Capítulos 9 y 10 se presentan las conclusiones a las que se han llegado así como las características o modificaciones que se podrían añadir en futuras iteraciones del proyecto.

Capítulo II

Introduction

II. Introduction

2.1 **Motivation**

Nowadays, we live in a society that could be defined as Information Society. Year after year it is more necessary to have access to a fast communication, and a proof of that is the growth of instant messaging applications, e-mail providers and telecommunication networks. Fast and easy communication is an important part of business, and day after day becomes more important in everybody's daily life.

Communication becomes indispensable in everyday life (staying in touch with family, communicating with friends and event planning, etc), but there are certain groups of people that suffer from diseases that make it harder and even prevent them from communicating with anybody else.

We are talking here about Autism, Cerebral Palsy, Amyotrophic Lateral Sclerosis or Multiple Sclerosis patients. Communication can be challenging for them and for the rest of people that might need to communicate with these patients, like doctors or teachers.

Augmentative and Alternative Communication Systems appeared to cover this communication need, in particular, pictograms. These pictograms are images that represent, regardless of the language, all kind of things such as objects, actions or ideas. There are several pictogram systems or sets, which have both color pictograms and black and white pictograms. These sets can also contain pictures of real objects or pictures of the sign language.

The purpose of the present project is to develop a mobile application through which a user could create sentences using pictograms. This application would help the user to create these sentences using a predictive mechanism.

Prediction is essential since we are not talking here about comfort but need, as most of the users might have some kind of mobility or intellectual problem. These problems might difficult the search of pictograms, so the predictions made by the system make the search of the next pictogram easier while writing a sentence.

2.2 Objectives

Regarding the communication needs of people who suffer from the previously mentioned diseases, the following objectives have been established:

- Study available platforms and choose the most suitable for this project.
- Choose a widely used pictogram system.
- Obtain and restructure the chosen pictogram system database if required.
- Design an architecture that combines the technologies used in an efficient way.
- Design of a predictive algorithm and the client that uses it.
- Design an extra comfortable user interface regarding the type of users this application will have.
- Publish the application in some kind of repository or virtual shop so it becomes available for as many people as possible.
- Implementation of functions that allow the user to save and share the sentences created.

2.3 Document structure

The present document follows a similar structure as the development followed:

First of all, Chapter 3 makes a quick review at both the technologies used and the current situation in the development of similar applications. In this case, we will talk about AACS, pictograms, Android and web services.

Once we know the context of the problem and the technologies, Chapter 4 makes an overview of the system architecture, introducing every main component, which will be further explained in the following chapters.

In Chapter 5 we will explain the transformation and structuring process performed on the pictogram database.

Chapter 6 exposes the web service that allows communication between the client and the pictogram database.

The actual client of the application, its functionality and its implementation details will be covered in Chapter 7.

Chapter 8 exposes the individual contributions to the project made by the group members during development.

Finally, in Chapter 9 and 10 we expose the conclusions we obtained as well as the features or modifications that could be added in future iterations.

Capítulo III

Estado del arte

III. Estado del arte

En este capítulo se hablará de la base con la que partía el proyecto. Se describirán las tecnologías usadas para su desarrollo así como el campo en el que se engloba.

3.1 Sistemas Aumentativos y Alternativos de Comunicación, SAAC

Los SAAC, Sistemas Aumentativos y Alternativos de Comunicación, son sistemas de comunicación que no utilizan palabras articuladas pero que tienen suficiente nivel de estructuración para transmitir información. Son Sistemas Alternativos pues van dirigidos hacia aquellas personas que no tienen lenguaje oral o que se considera que no compensa el esfuerzo necesario para que se dé el lenguaje y es preciso el uso de otro sistema para que la persona en cuestión pueda comunicarse. Son sistemas aumentativos dado que han sido ideados para aumentar las capacidades comunicativas de las personas que los usan. No eliminan el lenguaje oral pero se usan en casos en que éste no es suficiente para establecer una comunicación satisfactoria.

Dentro de estos sistemas podemos distinguir entre sistemas de símbolos gestuales y de símbolos gráficos. Los símbolos gestuales abarcan una gran variedad de gestos que van desde mímica hasta ciertos signos manuales. A pesar de esto, los lenguajes de signos usados pon personas sordomudas no se consideran SAAC, dado que son idiomas de por sí desarrollados para solventar un problema concreto y que se aprenden de forma natural del mismo modo que ocurre con el lenguaje hablado. Por otro lado, el uso de estos lenguajes requiere ciertas habilidades motrices que ciertos colectivos no posean.

Por otro lado, los símbolos gráficos pueden ser usados tanto por personas con discapacidad intelectual o TEA como por personas con discapacidades motoras (Parálisis Cerebral, Esclerosis Lateral Amiotrófica, Esclerosis Múltiple, etc.). Un ejemplo de sistema de símbolos gráficos son los pictogramas, de los cuales se hablará más en profundidad en el siguiente epígrafe.

Otra manera de agrupar estos sistemas SAAC sería, por un lado, los que requieren de soportes externos a la persona (SAAC con ayuda), y los que no hacen uso de ningún apoyo externo a la persona (SAAC sin ayuda), como es el lenguaje de señas.

Después de esta breve introducción, nosotros nos centraremos en el estudio de los pictogramas, dentro de los grupos de los SAAC con ayuda y de los sistemas de símbolos gráficos, pues son los que nos atañen para realizar el siguiente proyecto. Concretamente pertenecen a los sistemas de símbolos pictográficos, y en España los más usados son el sistema SPC (Ministerio de educación y ciencia s.f.) y el sistema ARASAAC (ARASAAC s.f.).

En los SAAC, una característica importante es cómo se va a manipular el sistema, es decir, de qué manera el sistema va a interactuar con el usuario y cómo el usuario va a poder manejar el mismo. Existen varios modos como se describe en (ARASAAC s.f.):

- *La selección directa*, consiste en que el emisor del mensaje selecciona o pulsa el pictograma que desea comunicar.
- *La selección con ratón*, funciona del mismo modo que la selección directa, pero en este caso el emisor se ayuda de algún tipo de dispositivo de señalización. Estos dispositivos van desde ratones convencionales a ratones que funcionan con el movimiento de la cabeza, pasando por los típicos joystick o trackballs.
- *La exploración o barrido dependiente*, precisa de un interlocutor para la comunicación. Este interlocutor o asistente va señalando uno a uno los grupos de pictogramas y los pictogramas dentro de cada grupo. Es el emisor del mensaje el que señala al interlocutor mediante algún gesto o sonido el grupo y el pictograma dentro de ese grupo que desea.
- *La exploración o barrido independiente*, es análoga al método anterior, pero en este caso es un programa el que hace de interlocutor y el emisor precisa de alguna clase de dispositivo o conmutador para indicar al interlocutor cuando parar.
- *La selección codificada*, se basa como su nombre indica en una codificación numérica o incluso con colores de los pictogramas. De este modo el emisor simplemente indica el código del pictograma mediante selección directa o por barrido. Así pues, tiene menos elementos con los que interactuar por lo que una vez el emisor conoce los códigos la comunicación es mucho más rápida.

Decir que para realizar nuestra aplicación nos hemos decidido por la selección directa, por tratarse de una herramienta táctil.

Los pictogramas que vamos a utilizar para nuestra aplicación han sido facilitados por ARASAAC y pertenecen a uno de sus catálogos, pictogramas en color. Posteriormente nosotros gestionaremos la base de datos añadiendo diversas categorías.

3.1.1 Pictogramas

Los pictogramas son signos escritos que representan elementos del mundo real, ideas, acciones, cualidades, etc. En general, representan cualquier cosa que una persona desee comunicar de una manera clara y esquemática, de forma que no es necesario el uso del lenguaje oral o escrito para establecer una conversación.

Los usuarios de este SAAC deben aprender a relacionar la realidad que viven y sus elementos con estos signos en vez de con palabras. Una vez hecho esto, los pictogramas son para ellos como el lenguaje oral y escrito para la mayoría de las personas.

A continuación, en la Figura 1 podemos ver un ejemplo de una frase con pictogramas.

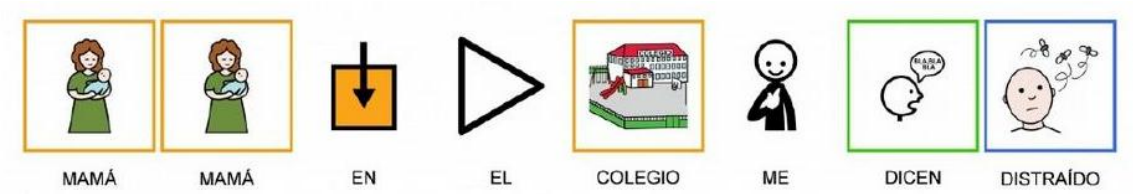


Figura 1: Ejemplo de frase con pictogramas

Existen varios sistemas de pictogramas para el español pero el de ARASAAC (CATEDU, Portal Aragonés de la Comunicación Aumentativa y Alternativa s.f.) es el más usado.

Figura 2: Pictogramas ARASAAC a Color

Como podemos ver en la Figura 2, para una misma palabra pueden existir varias representaciones en pictogramas.

ARASAAC también nos ofrece estos mismos pictogramas en blanco y negro.

Por ello, por la gran variedad y cantidad de pictogramas que nos ofrece y porque es el sistema más usado hemos decidido usar la base de datos de pictogramas en color de ARASAAC para este proyecto.

Figura 3: Pictogramas ARASAAC Blanco y Negro

Existen también distintas iniciativas que hacen uso de los sistemas de pictogramas existentes para crear documentos como por ejemplo cuentos (Difasia en Zaragoza s.f.), o documentos serios como la convención de los derechos de las personas con discapacidad (Ministerio de Sanidad y Política Social 2010).

3.1.2 ARASAAC

El portal ARASAAC ofrece recursos gráficos y materiales para facilitar la comunicación de aquellas personas con algún tipo de dificultad en esta área. Este proyecto ha sido financiado por el Departamento de Industria e Innovación del Gobierno de Aragón y forma parte del Plan de Actuaciones del Centro Aragonés de Tecnologías para la Educación (CATEDU), centro dependiente del Departamento de Educación, Universidad, Cultura y Deporte del Gobierno de Aragón.

El portal ARASAAC ofrece diferentes recursos, como son:

- Pictogramas a color: ARASAAC nos proporciona una base de datos de unos nueve mil pictogramas a color (Figura 4). En particular es la que se ha usado para el desarrollo de nuestra aplicación.

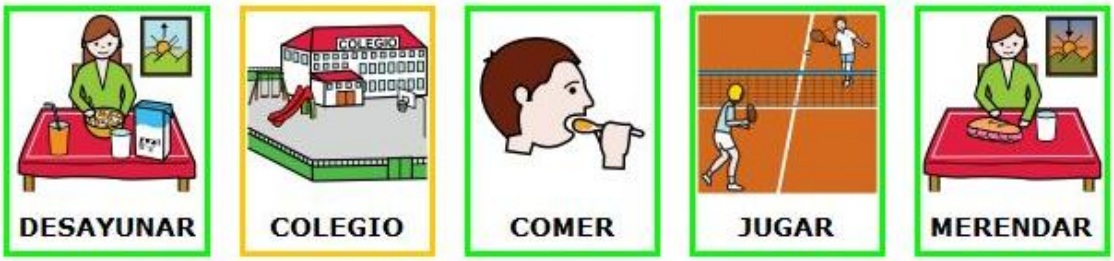


Figura 4: Pictogramas ARASAAC a Color

- Pictogramas en blanco y negro: El mismo catálogo está disponible en blanco y negro como se mostró en la Figura 3.
- Fotografías: Es un catálogo de fotografías simples de objetos más concretos como se puede ver en la Figura 5.

Figura 5: Fotografía a modo de pictograma

- Software: ARASAAC ofrece aplicaciones variadas basadas en sus pictogramas. Hablaremos de algunas de ellas más adelante.
- Fotografías de signos del lenguaje de signos español.

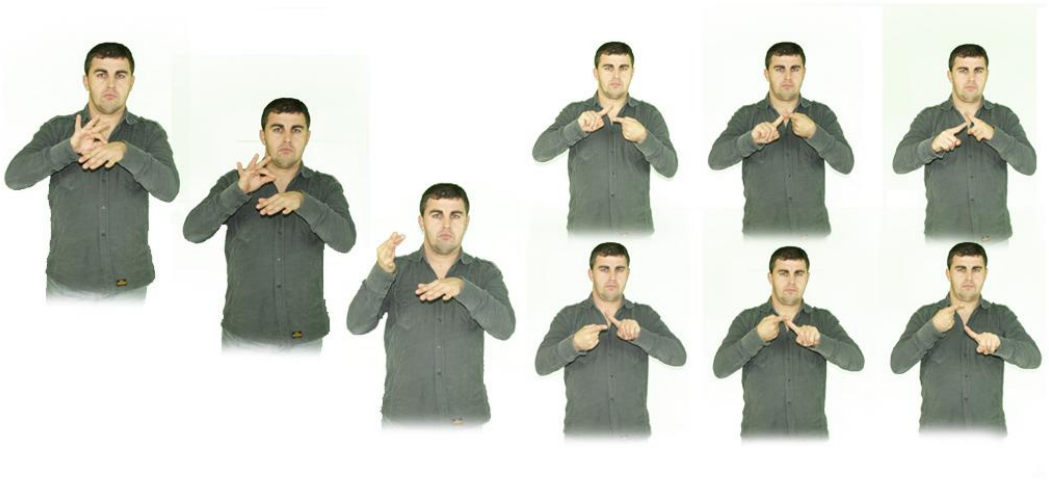
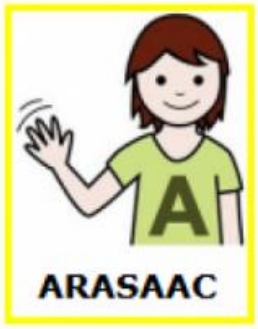


Figura 6: Fotografías de signos del lenguaje de signos

- Vídeos de signos del lenguaje de signos español.

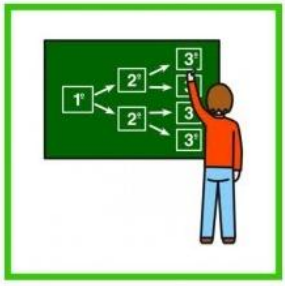
De todos los conjuntos que se han visto anteriormente, el más atractivo para este proyecto es el de pictogramas a color, ya que incluye un marco de color para cada pictograma que lo categoriza según la gramática española. La organización es la siguiente:

- Amarillo (Sujetos/Personas): Sustantivos, pronombres personales y nombres propios.



- Naranja: Sustantivos y nombres comunes

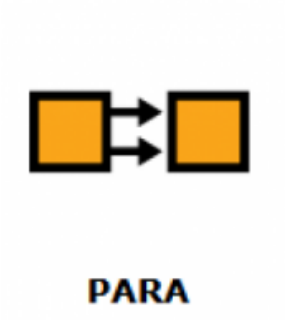
- Verde: Acciones/Verbos



- Azul: Descriptivos (adjetivos y adverbios)

- Rosa: Social, expresiones, fórmulas de cortesía y frases interrogativas.

- Blanco/sin marco: Miscelánea, suelen ser preposiciones, artículos, números, letras u otras palabras que no encajan bien en ninguna categoría anterior.



ARASAAC proporciona a desarrolladores una base de datos en Microsoft Excel con todos sus pictogramas a color y su información, como son los posibles significados o palabras que representa cada pictograma y la categoría para cada palabra que representa.

De este modo tenemos que un mismo pictograma puede ser usado como verbo o sustantivo dependiendo del color de su marco.

Además, para muchos de los pictogramas de la base de datos existe una columna con su correspondiente traducción a diferentes idiomas (alemán, inglés, italiano, catalán, valenciano, etc.)

Los pictogramas utilizados en la aplicación desarrollada facilitadas por la asociación ARASAAC, son propiedad del Gobierno de Aragón y han sido creados por Sergio Palao. Se publican bajo licencia Creative Commons (BY-NC-SA), autorizando de esta manera su uso para fines sin ánimo de lucro siempre que se cite la fuente, autor y se compartan bajo la misma licencia.

3.1.3 Ejemplos de aplicaciones basadas en pictogramas

AraWord (CATEDU, AraWord s.f.) es una aplicación libre que forma parte de un conjunto mayor de herramientas para la comunicación alternativa y aumentativa. AraWord es un procesador de textos que permite la escritura de texto y pictogramas a la vez, de modo que cada palabra va acompañada del pictograma correspondiente. Esto facilita en gran medida la creación de documentos y material adaptado a pictogramas.

Por otro lado, el hecho de que aparezca el pictograma asociado a la palabra que se escribe supone un refuerzo para personas que están aprendiendo a leer y escribir, ya que por ejemplo, si el pictograma se muestra, además del refuerzo visual que supone, indica que la palabra ha sido escrita correctamente.

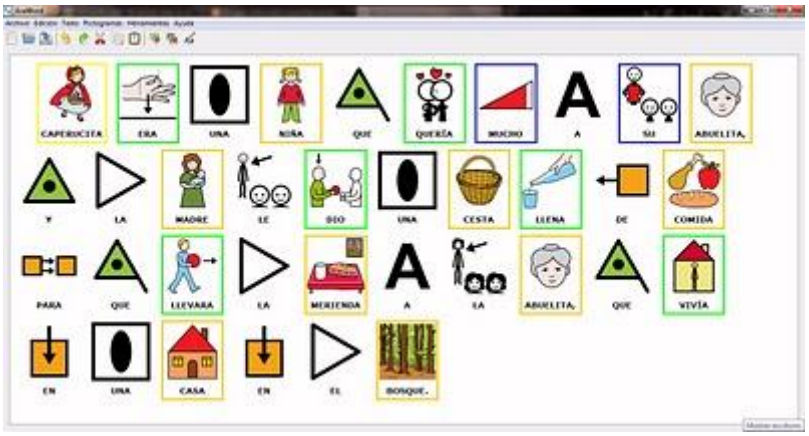


Figura 7: Interfaz de AraWord

El Proyecto TICO (Departamento de Informática e Ingeniería de Sistemas del CPS de Zaragoza s.f.) es una aplicación de escritorio que permite crear y usar tableros de pictogramas de un modo interactivo. La aplicación se compone a su vez de dos módulos independientes que se complementan entre sí: Editor e Intérprete.

Por un lado, el editor permite crear tableros de pictogramas, sonidos o elementos de control que el usuario necesite.

Por otro lado, el intérprete permite la selección de los elementos de dichos tableros para componer mensajes. Para ello proporciona un mecanismo de selección por barrido que recorre los elementos del tablero, facilitando su uso a personas con problemas motrices graves.

MICE - Ratón Virtual (CATEDU, MICE El ratón virtual s.f.), es una aplicación informática que emula las funciones del ratón en una ventana emergente. Está ideada para personas con un nivel de discapacidad motriz tal que no pueden manejar un ratón convencional.

MICE permite controlar el cursor del ordenador usando diversos dispositivos adaptados a las necesidades de cada persona. Entre estos dispositivos podemos encontrar teclados, micrófonos o conmutadores. MICE adapta las señales proporcionadas por estos dispositivos y permite acceder a todas las funcionalidades que nos ofrece un ratón tradicional, desde el movimiento hasta cada uno de los botones.

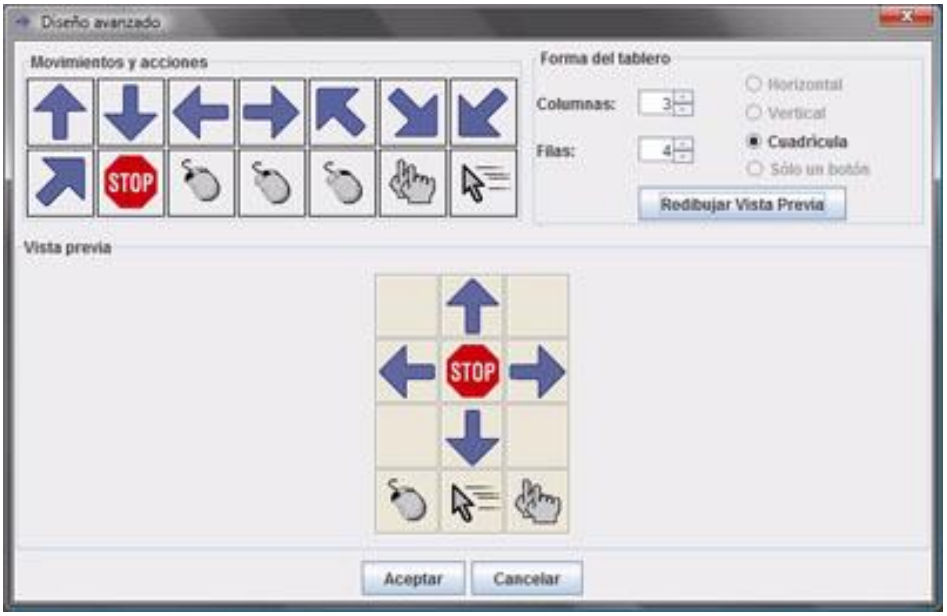


Figura 8: Interfaz de MICE

PictoAgenda (Pictoagenda s.f.) permite, a través de nuestro navegador, elaborar una agenda de actividades para un día determinado, anticipando al niño la secuencia de actividades a realizar y mejorando con ello la estructuración temporal, tan importante en niños con TEA.

PictoAgenda incorpora unos gráficos encima de los pictogramas para indicar el estado de la tarea descrita (terminada, no realizada, activa, etc.) y también permite añadir un color a estos gráficos. Con ello se pretende adaptar la aplicación a diferentes metodologías y códigos de color usados por padres y profesionales.

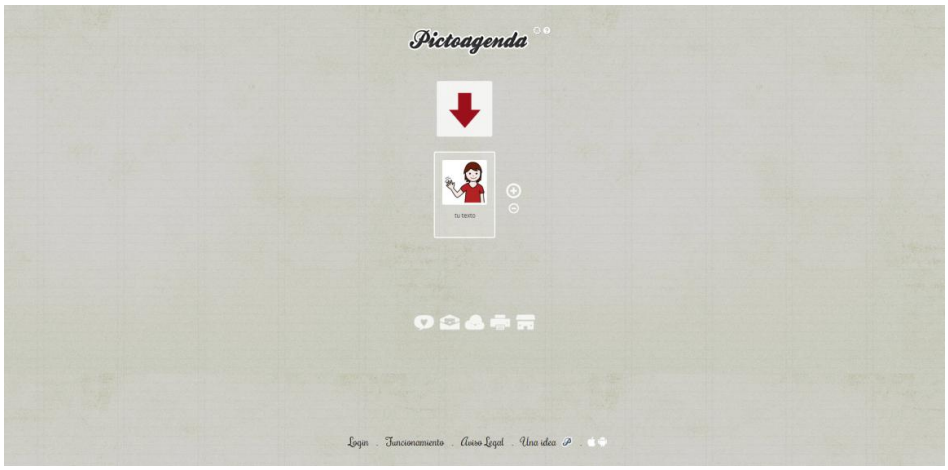


Figura 9: Interfaz de PictoAgenda (Web)

El Editor **Picto4me** (CATEDU, Picto4Me s.f.) para Google Drive permite crear, editar y compartir tableros de comunicación directamente en Google Drive.

Es una aplicación online que destaca por su sencillez a la hora de diseñar los tableros de comunicación, así como por la integración con Google Drive. Entre sus características están:

- Creación de nuevos tableros directamente desde el menú "Crear nuevo".
- Editar tableros con sólo hacer clic en ellos.
- Compartir los tableros al igual que se haría con otro documento de Google Drive.

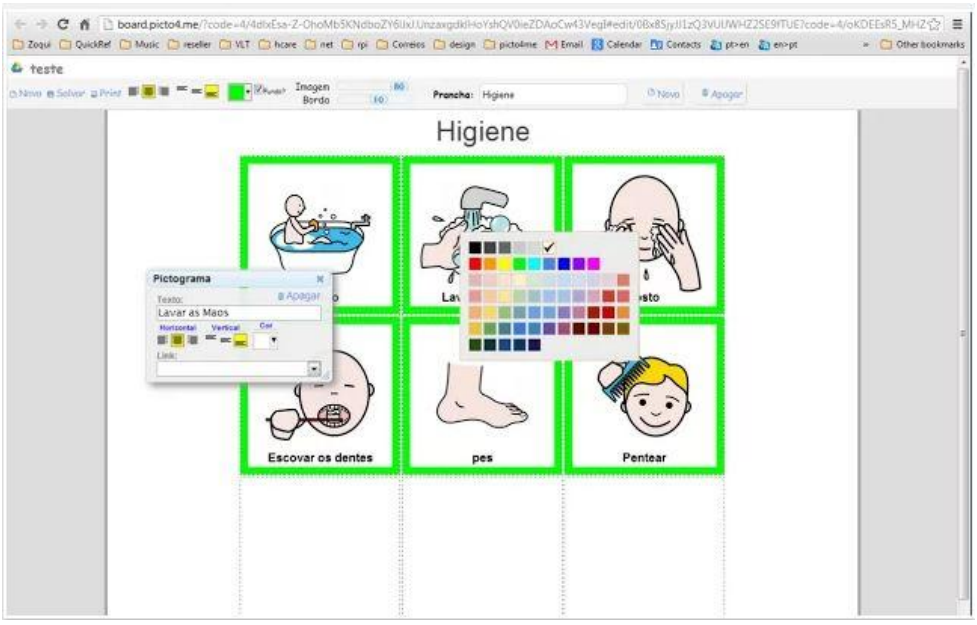


Figura 10: Interfaz de Picto4me (Web)

Pictosonidos (Pictosonidos s.f.) es una aplicación web de libre acceso pensada para el trabajo del vocabulario. Pictosonidos ofrece una serie de pictogramas que tienen asociado un sonido o locución verbal del estilo "la vaca hace muuu". Se pretende mejorar el proceso de aprendizaje de vocabulario añadiendo el refuerzo auditivo.

Los pictogramas se clasifican en diferentes categorías (Transportes, Alimentos, Cocina, Animales, etc.).

Dentro de cada categoría es posible navegar pictograma a pictograma o bien seleccionar uno concreto. Además incluye un navegador visual en la parte superior para facilitar esta labor.

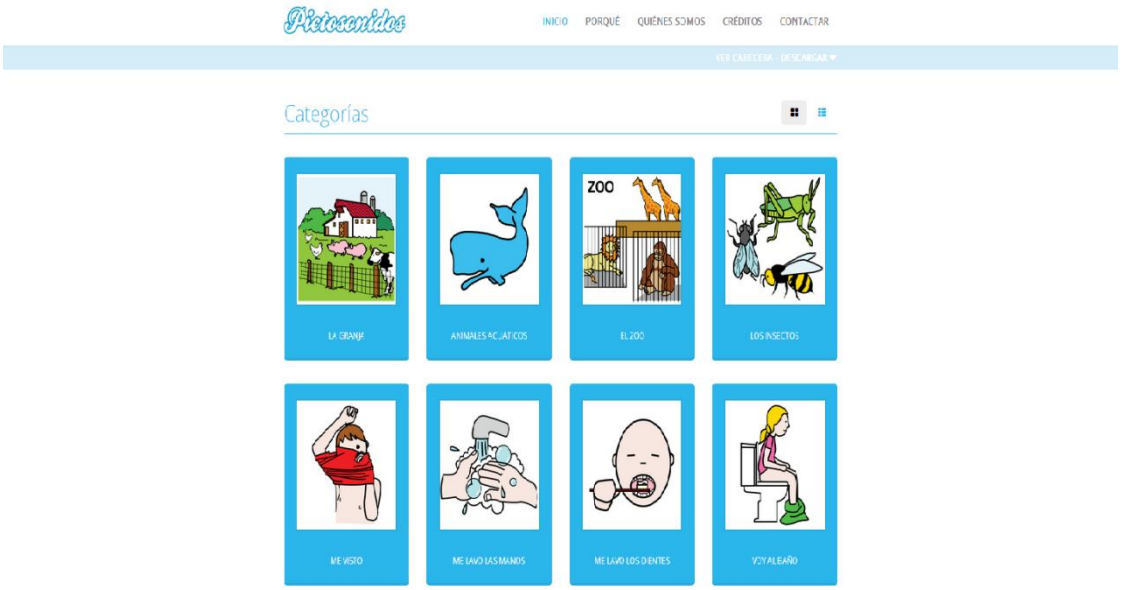


Figura 11: Interfaz Pictosonidos (Web)

El **Messenger Visual** (Fundació El Maresme, BJAdaptaciones s.f.) consiste en un servicio de mensajería instantánea que usa los pictogramas de ARASAAC como medio de comunicación. El Messenger Visual se presenta en una plataforma o interfaz con los mecanismos necesarios para que personas con limitaciones de lectura, escritura o movilidad puedan usarla con la mayor comodidad posible.

Pictotraductor (Grupo Promedia Soluciones Multimedia Para Empresa S.L. s.f.) es una aplicación web totalmente gratuita que nos permite traducir texto a pictogramas. Incorpora funcionalidades interesantes como la posibilidad de hacer el negativo de cualquier pictograma utilizando sufijos delante de la palabra (-no) y la creación de pictogramas de textos para aquellas palabras que no dispongamos de pictogramas de la base de datos de ARASAAC o dentro de los pictogramas que hayamos subidos a nuestro espacio de trabajo personal.

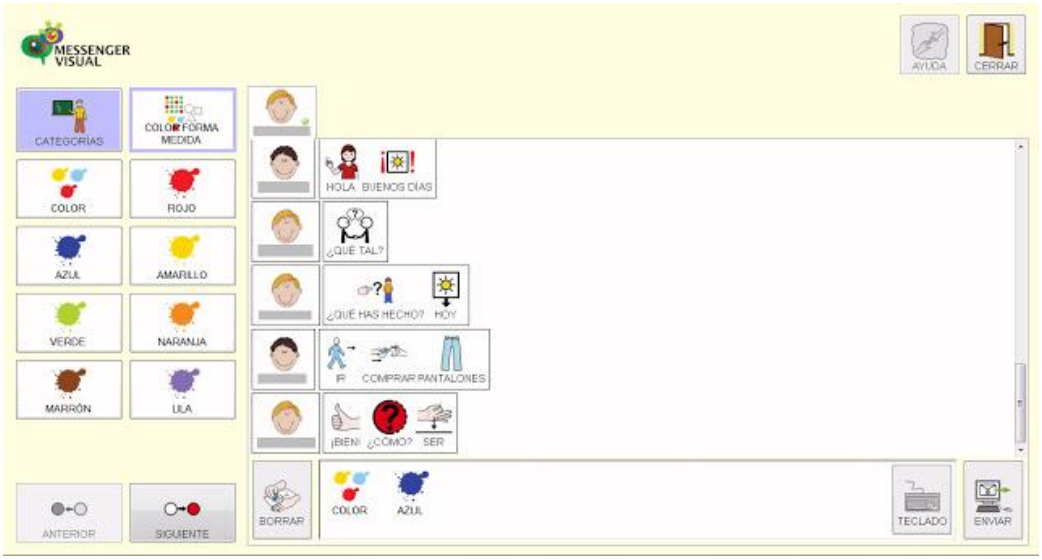


Figura 12: Interfaz de Messenger Visual

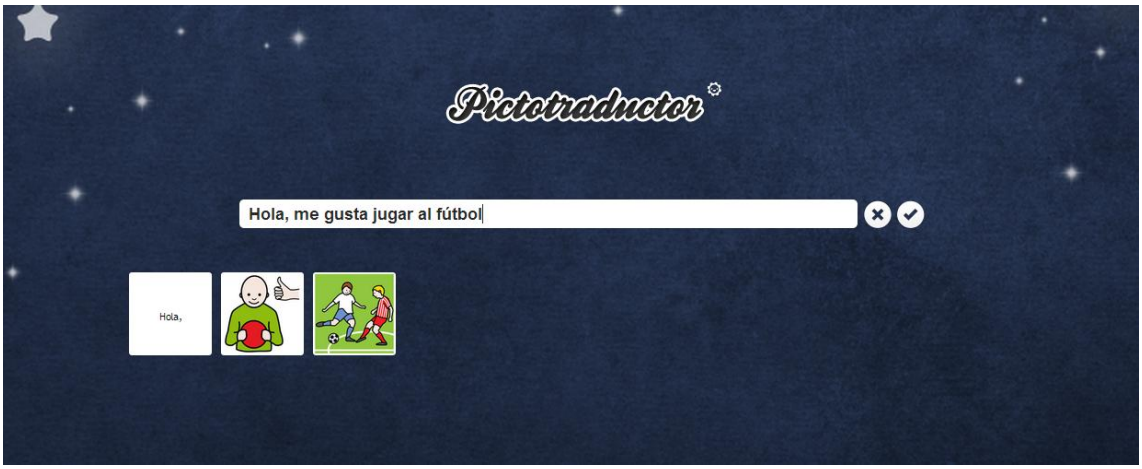


Figura 13: Captura Pictotraductor

Por otro lado, existen también aplicaciones para plataformas móviles que usan pictogramas. Éstas son algunas de ellas.

AraBoard (Marta García Azpiroz, Javier Marco Rubio s.f.) es un conjunto de herramientas que proveen una funcionalidad parecida a la del Proyecto TICO antes mencionado. AraBoard permite la creación y uso de tableros de pictogramas en distintos dispositivos ya que dispone de versión tanto para Windows como para Android. Esto permite por ejemplo su ejecución en smartphones y tablets siempre que tengan instalado Adobe Air.

Figura 14: Interfaz de AraBoard

Pictodroid Lite (Accegal s.f.), aplicación para dispositivos Android que permite a los usuarios comunicarse a través del uso de pictogramas. Esta aplicación permite la creación de frases simples del estilo "quiero beber...", "quiero jugar...", "vamos a...", "estoy...". La formación de la frase sigue una estructura básica de sujeto-verbo-predicado bastante lineal, dando al usuario en cada paso un conjunto de pictogramas limitado para elegir, aunque se pueden añadir más pictogramas a la aplicación si el usuario así lo desea.

Incorpora una categorización de pictogramas muy básica como se ve en la Figura 15, y no hace uso de ningún mecanismo de predicción, si no que como ya se ha dicho, la formación de la frase es bastante lineal.

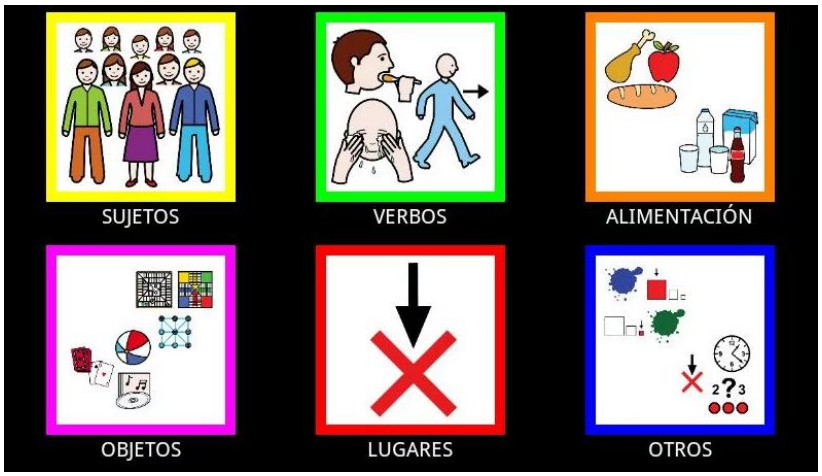


Figura 15: Selección de categoría de Pictodroid Lite

Al terminar la frase, la aplicación la reproduce mediante un sintetizador de voz, lo cual en un añadido que se agradece teniendo en cuenta el conjunto de posibles usuarios. En la siguiente figura se ve un paso del proceso de creación de una frase.

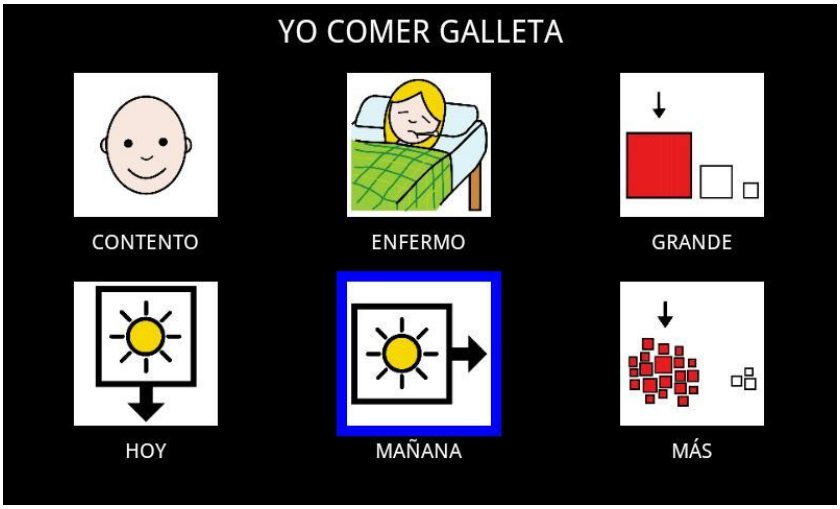


Figura 16: Paso en la creación de una frase en Pictodroid

PictogramAgenda (Lorenzo Moreno s.f.) es una aplicación móvil que permite generar tareas usando pictogramas, con un máximo de doce pictogramas por tarea. Está desarrollada en Android, por lo que está disponible para smartphones y tablets, lo cual es bastante útil debido a la naturaleza de la aplicación. PictogramAgenda permite ordenar los pictogramas para crear la tarea deseada, de modo que no es necesario eliminar ninguna tarea al añadir otra que debería ir antes.

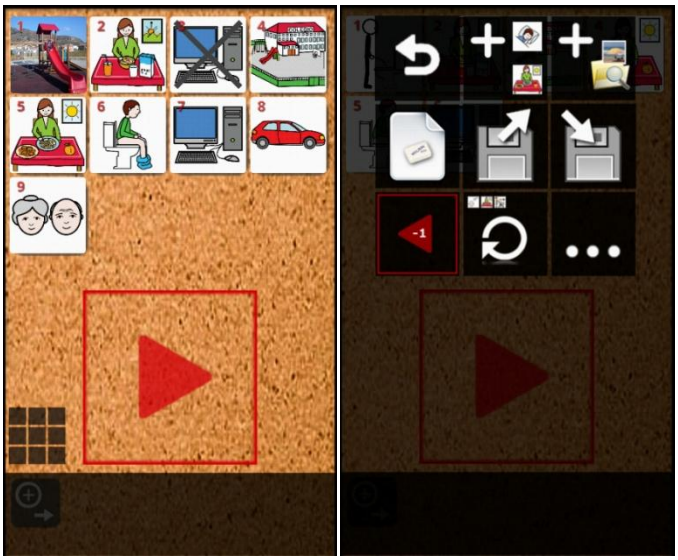


Figura 17: Interfaz de PictogramAgenda

3.2 Android

A continuación se presenta una pequeña descripción de Android, sus orígenes, características y arquitectura. Para más información consultar (Felipe Antonio García Ochando 2014), (Manuel López Michelone 2013), (Ángel J. Vico 2011), y por supuesto la correspondiente entrada en (Wikipedia s.f.) y sus respectivas fuentes.

Android es un sistema operativo basado en el kernel de Linux, diseñado para su uso en dispositivos móviles con pantalla táctil como tablets y smartphones. La interfaz de usuario está orientada a su directa manipulación mediante entrada táctil, esto es, pulsando la pantalla, arrastrando elementos o soltándolos. Los dispositivos que usan Android suelen estar provistos de giroscopios, acelerómetros y sensores de proximidad, los cuales son usados por algunas aplicaciones para responder ante ciertas acciones del usuario.

Android permite la personalización de la pantalla principal con fondos de pantalla, accesos directos a aplicaciones y widgets de las mismas. De este modo el usuario puede tener en la pantalla principal, por ejemplo, una lista de la bandeja de entrada de su correo electrónico o información del clima. El sistema permite también que las aplicaciones envíen notificaciones al usuario para informarle sobre eventos relevantes.

El código fuente de Android es liberado por Google bajo la licencia Apache, la cual permite la modificación del código con total libertad. Esto hace que tanto fabricantes como desarrolladores puedan modificar el código para adaptarlo a sus necesidades o gustos. A pesar de que los dispositivos son vendidos con software propietario adicional, la libertad que ofrece Android ha hecho posible la creación de una gran comunidad de desarrolladores que programan aplicaciones para ampliar la funcionalidad de los dispositivos.

Android es la plataforma más usada en dispositivos móviles, habiendo superado a Symbian en el cuarto trimestre de 2010. Android es popular entre empresas que necesitan un sistema operativo ligero, barato y personalizable para sus dispositivos. Esto ha hecho que, a pesar de estar diseñado para smartphones y tablets, también se ha hecho un hueco en el mercado de los televisores, videoconsolas, cámaras digitales y otros dispositivos electrónicos.

3.2.1 Orígenes

Android Inc. fue fundada en Octubre de 2003 por altos cargos o fundadores de diversas empresas, con el objetivo de crear "dispositivos móviles más inteligentes y más conscientes de la localización y preferencias de su propietario". La idea original

era desarrollar un sistema operativo para cámaras digitales, pero pronto se dieron cuenta de que era un mercado demasiado pequeño, por lo que redirigieron sus esfuerzos hacia la creación de un sistema operativo para smartphones que pudiese competir con Symbian y Windows Mobile (sustituido por Windows Phone en 2010). Durante bastante tiempo Android Inc. trabajó secretamente, revelando solo que estaba desarrollando software para teléfonos móviles.

Google compró Android Inc. en Agosto de 2005 y miembros importantes así como tres de los fundadores permanecieron en la empresa. Con este movimiento era fácilmente interpretable que Google quería entrar en el mercado de la telefonía móvil. Un equipo de Google, liderado por uno de los fundadores de Android Inc., desarrolló usando el kernel de Linux una plataforma para dispositivos móviles. Dicha plataforma fue presentada a fabricantes y tele operadoras bajo la promesa de un sistema flexible y actualizable.

En enero de 2007 el iPhone de Apple salió a la luz con una pantalla táctil. Esto supuso un contratiempo en el desarrollo de Android ya que, por aquella fecha, Google tenía un prototipo parecido a una BlackBerry, es decir, sin pantalla táctil y con teclado QWERTY. Inmediatamente los ingenieros de Google repensaron el sistema operativo y los prototipos para combinar las ideas detrás de sus diseños con unas capacidades diseñadas para competir con el iPhone.

El 5 de Noviembre de 2007, se formó la Open Handset Alliance, un conjunto de empresas incluida Google, fabricantes como Sony, HTC y Samsung, fabricantes de hardware como Qualcomm y otras empresas. Su objetivo era desarrollar estándares abiertos para los dispositivos móviles y ese día Android fue desvelado como su primer producto: una plataforma móvil construida sobre la versión 2.6 del kernel de Linux. El primer smartphone comercializado con Android fue el HTC Dream, lanzado al mercado en Octubre de 2008.

En 2010, Google lanzó la gama Nexus de dispositivos, una línea de smartphones y tablets con Android fabricados por un fabricante partner. El primero de ellos, el Nexus One, fue fabricado por HTC; y desde entonces la gama ha sido actualizada con nuevos dispositivos como el Nexus 4 (smartphone) o el Nexus 10 (tablet), fabricados por LG y Samsung respectivamente. Esta gama de dispositivos hace de buque insignia de la tecnología de Google en este campo, mostrando la última versión de Android y las últimas funcionalidades hardware implementadas en el momento del lanzamiento.

3.2.2 Arquitectura

Al igual que sucede en otras plataformas, la arquitectura de Android (Ángel J. Vico 2011) se divide en diferentes capas para ahorrar trabajo al programador. En el caso de Android, sobre el kernel se edifican una serie de librerías y frameworks gracias a los cuales los desarrolladores pueden acceder a la funcionalidad de más bajo nivel del sistema como gestión de memoria, uso de dispositivos (cámara, acelerómetro) sin tener que conocer las particularidades de cada componente. A continuación, la Figura 18 extraída del análisis de Android de (Patrick Brady 2008).

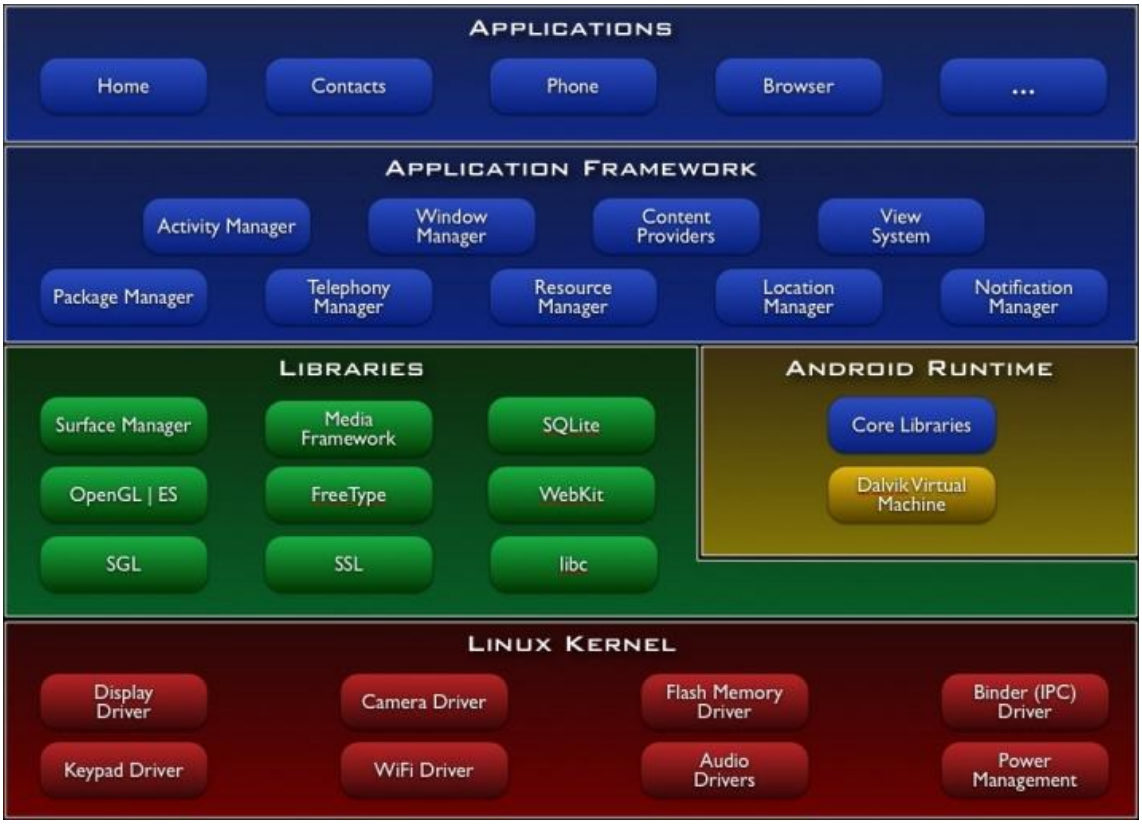


Figura 18: Arquitectura Android

Kernel de Linux: Como se mencionó con anterioridad en la introducción a Android, el kernel de Android no es más que una versión adaptada del kernel de Linux 2.6. Era necesaria una adaptación ya que las prestaciones y arquitectura de un ordenador no son las mismas que las de un dispositivo móvil como un smartphone o una tablet.

El cometido del kernel, entre otros, consiste en hacer de intermediario entre el hardware del dispositivo y el resto de capas, permitiendo al desarrollador olvidarse de las características concretas del dispositivo para el que está programando. El kernel contiene los drivers necesarios para manejar el hardware del dispositivo, pero no se

accede directamente a éstos, sino que se accede usando una serie de librerías que se montan sobre el kernel. Del mismo modo que el kernel contribuye a la comodidad del desarrollador, estas librerías hacen lo propio en relación a los detalles del kernel.

El kernel también se encarga de manejar tanto los recursos del dispositivo como los del propio sistema Android. Algunos ejemplos son la gestión de la memoria y de la energía, el control de los procesos o los elementos de comunicación en el caso de que los hubiese.

Librerías: También conocidas como bibliotecas nativas de Android, estos componentes se montan sobre el kernel y están programadas normalmente en C o C++. Estas librerías vienen incluidas cuando compramos un dispositivo Android ya que son desarrolladas por el fabricante del dispositivo, que es el que conoce los detalles del hardware.

La función de estas librerías consiste en proporcionar a las aplicaciones funcionalidad que suelen usar, para de este modo evitar que cada aplicación tenga que codificar de nuevo estas funcionalidades. Esto también permite que la tarea en cuestión pueda ser implementada de la manera más eficiente posible.

Estas librerías realizan diversas funciones que van desde proporcionar un sistema de base de datos (SQLite) hasta el motor gráfico (OpenGL), pasando por funcionalidades más rutinarias como la de tener un navegador web (Webkit).

Entorno de ejecución: Este componente de la arquitectura Android no se considera una capa como tal como se puede ver en la Figura 18 pero no por ello es menos importante. De hecho, es en el entorno de ejecución donde se sitúa la DVM (*Dalvik Virtual Machine*), a través de la cual se ejecutan todas las aplicaciones.

La máquina virtual Dalvik es una implementación desde cero de la máquina virtual de Java, lo cual permite evitar problemas de licencias con Sun y con Java ME. Técnicamente no opera con bytecode, por lo que en realidad no es una JVM (*Java Virtual Machine*), sino que se traduce de bytecode a dx, que es el formato de la Dalvik. Las aplicaciones para Android se programan en Java y son compiladas una única vez para la máquina Dalvik. De este modo se consigue el poder programar una aplicación para Android independientemente del dispositivo final en que se vaya a ejecutar, ya que todo dispositivo Android cuenta con la máquina virtual Dalvik.

Framework de aplicaciones: Contiene las clases y servicios que las aplicaciones Android usan directamente. Proporciona prácticamente toda la funcionalidad que una aplicación pueda necesitar, como por ejemplo acceso al sistema de notificaciones y alertas de sonido, ciclo de vida de la aplicación, localización del dispositivo o proveedores de contenido.

Estos componentes son en su mayoría librerías implementadas en Java, que usan los elementos inferiores de la arquitectura Android a través de la máquina virtual Dalvik.

Aplicaciones: Como su nombre indica, esta capa contiene todas las aplicaciones del dispositivo sean del tipo que sean, desde aplicaciones nativas con o sin vista hasta las instaladas por el usuario. El *launcher* también se encuentra en esta capa y es el que proporciona la vista de escritorios al usuario, muestra la lista de las aplicaciones que en usuario puede ejecutar y permite añadir accesos directos y widgets.

3.3 Servicios Web

Como se adelantó en la introducción del documento, la sociedad está cada vez más orientada hacia la comunicación, y ello también tiene su aspecto en las aplicaciones que se desarrollan. Con la acogida cada vez mayor de protocolos como HTTP ha sido posible el desarrollo de aplicaciones independientes que realizan una función muy concreta desde un servidor que puede ser accedido a través de la red por cualquiera que lo necesite. Eso es precisamente un servicio web.

Existen fundamentalmente dos tipos de servicios web, ligeros y pesados. A continuación se hará una introducción a ambos tipos y un recorrido más en profundidad de los pesados, ya que son los que se han usado en este proyecto.

3.3.1 Servicios web ligeros

Los servicios web ligeros surgen con alternativa al protocolo SOAP que se verá más adelante. SOAP usa para la comunicación mensajes XML, lo cual es una de las principales razones de su éxito, pero también puede ser un inconveniente. El uso de mensajes XML y del protocolo en sí supone una complejidad que en muchos casos es innecesaria, y lo que es peor, puede causar sobrecarga en la red.

Es por ello que surgieron los servicios web ligeros como alternativa a SOAP para situaciones en los que se dispone de un ancho de banda limitado o casos en los que no es necesaria la complejidad que SOAP conlleva.

Este tipo de servicios se caracterizan por no proporcionar una descripción del servicio y por usar mecanismos alternativos para el envío de mensajes. Algunos ejemplos son el protocolo Hessian (comunicación basada en binarios) y el protocolo Burlap (comunicación basada en SML [*Simple Markup Language*], subconjunto de XML).

Para más información: (Noé Fernández Iglesias 2006).

3.3.2 Servicios web pesados

Como se explica en (Carlos Andrés Morales Machuca s.f.), se pueden definir como *"un conjunto de tecnologías estándares de software para el intercambio de datos entre aplicaciones tales como SOAP, WDSL y UDDI"*. Estos pueden ser desarrollados en una gran variedad de lenguajes para ser implementados sobre muchos tipos de redes. El éxito de la interoperabilidad se consigue gracias a la adopción de protocolos y estándares abiertos. El conjunto de servicios y protocolos para los servicios web es conocido comúnmente como *"Web Services Protocol Stack"* y básicamente son utilizados para definir, localizar, implementar y hacer que un servicio web interactúe con otro.

Este conjunto de servicios está conformado esencialmente de cuatro subconjuntos como se puede ver en la Figura 19, extraída de (María Jesús Lamarca Lapuente s.f.):

- Servicio de transporte
- Mensajería XML
- Descripción del servicio

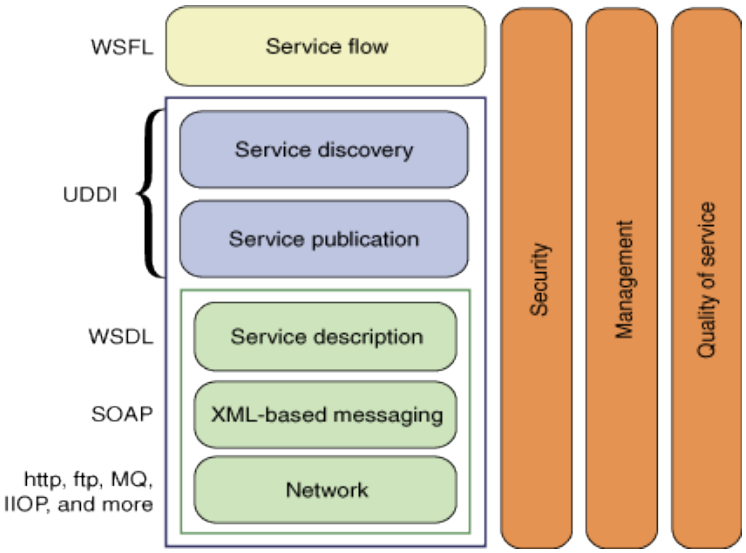


Figura 19: *Web Services Protocol Stack*

Servicio de transporte

Este servicio es la base de todo, ya que constituye el medio por el cual se van a enviar los mensajes. Comprende los propios protocolos de red como son HTTP (*HyperText Transfer Protocol*), FTP (*File Transfer Protocol*), SMTP (*Simple Mail Transfer Protocol*) o BEEP (*Block Extensible Exchange Protocol*). El servicio web implementado para este proyecto usa HTTP por lo que será el único del que se hable en este estado del arte.

HTTP es el protocolo del nivel de aplicación más utilizado en la red. Funciona como un protocolo de petición-respuesta basándose en un modelo cliente-servidor en el que el cliente sería por ejemplo un navegador web y el servidor la aplicación que gestiona el ordenador en el que se aloja la web.

Un cliente HTTP puede iniciar una petición estableciendo una conexión TCP con un servidor a través de un puerto que suele ser el 80. El servidor HTTP se mantiene escuchando a través de ese puerto hasta que llega alguna petición. Cuando el servidor recibe una petición, devuelve un mensaje de estado y a continuación un mensaje que puede variar. Si todo ha funcionado con normalidad, el segundo mensaje es el recurso solicitado por el cliente.

Mensajería XML

Esta capa se encarga de traducir los mensajes a XML para que puedan ser interpretados por cualquier nodo de la red. Algunos componentes más utilizados en este conjunto son los siguientes:

- XML (*eXtended Markup Language*) (W3C, W3CSchools XML Tutorial s.f.):

  XML es un lenguaje de marcado cuyo formato es legible tanto por una persona como por una máquina. Es por ello que es uno de los mecanismos más usados para la transmisión de información a través de la red. Aunque está enfocado a documentos de texto, con frecuencia se usa para la representación de estructuras de datos.

  Un documento XML se dice que está bien formado cuando sigue la DTD (*Document Type Definition*) que lo acompaña. La DTD contiene información de la estructura que debe seguir el documento XML para que cumpla la anterior. Dicha información incluye nodos, cantidad de nodos, atributos y nivel de anidamiento de los mismo entre otros.

- SOAP (*Simple Object Access Protocol*) (W3C, W3CSchools SOAP Tutorial s.f.):

  SOAP es un protocolo de la capa de aplicación que se basa en XML para la creación de mensajes que serán enviados a través de la red. Existen varios roles en el protocolo SOAP pero los principales son el "SOAP Sender" y el "SOAP Receiver" Ya que son los dos nodos entre los que se establece la comunicación. SOAP básicamente está constituido por:

  - Un sobre que describe qué es el mensaje y cómo procesarlo.
  - Un conjunto de reglas de codificación para representar los tipos de datos definidos por la aplicación.
  - Una convención para representar las peticiones y las respuestas.

Descripción Del Servicio

WSDL (*Web Services Description Languages*) (W3C, W3CSchools WSDL Tutorial s.f.) es un lenguaje para la definición de interfaces basado en XML que se usa para describir la funcionalidad que ofrece un servicio web.

Un documento WSDL asociado a cierto servicio web define lo que hace el servicio web, los métodos que posee, donde se encuentra y cuál es la forma de invocarlo. Es un formato que informa además de qué tipo de dato espera recibir y cuál será la respuesta, por lo que es de gran ayuda para los desarrolladores, ya que por ejemplo numerosos IDEs son capaces de parsear dicho documento WSDL, mostrar la API del servicio de una manera mucho más amigable y ofrecer autogeneración de código. Un documento WSDL contiene los siguientes elementos XML:

- Tipos <types>: Se definen los tipos de datos utilizados en los mensajes.
- Mensaje <message>: Normalmente un nodo <message> corresponde a una operación y dentro del nodo se encuentra la información necesaria para realizar dicha acción.
- Tipo de puerto <portType>: Este elemento define el servicio web, las operaciones que dicho servicio puede ejecutar y los mensajes necesarios para ejecutar cada operación.
- Etiquetas <binding>: Se definen el formato del mensaje y qué protocolos se van a usar. Aquí es donde se especifica, por ejemplo, que el protocolo de transporte va a ser HTTP.

3.3.3 Apache Axis2

Apache Axis es un framework de código abierto, basado en XML para servicios web. Consiste en una implementación en Java y otra en C++ del servidor SOAP, así como diversos utilitarios y APIs para generar y desplegar aplicaciones de servicios web. Axis se desarrolla bajo la supervisión de la Apache Software Foundation.

Apache Axis2 es un motor para servicios web surgido del rediseño y reimplementación de la pila SOAP de Apache Axis. Existen implementaciones de Axis2 tanto en Java como en C. Axis2 no solo provee la capacidad de agregar servicios web a las aplicaciones web, sino que además puede funcionar como servidor autónomo.

Para más información consultar (The Apache Software Foundation s.f.).

Capítulo IV

Arquitectura del sistema

IV. Arquitectura del sistema

En este capítulo se hará una introducción a los tres componentes principales de la aplicación: servidor de base de datos, servidor web y cliente.

La arquitectura que hemos utilizado podría describirse como una arquitectura cliente-servidor, en la que el cliente es la parte correspondiente a la aplicación Android y el proveedor de recurso es el servicio web montado sobre Axis2 de Apache.

Figura 20: Arquitectura del Sistema

4.1 Servidor de base de datos

Este componente es el que se encarga de la persistencia de la totalidad de los pictogramas y sus datos. Contiene tres tablas MySQL como se ve con más detalle en el Capítulo 5, las cuales almacenan las urls de los pictogramas, las categorías a las que pertenece un pictograma, su color gramatical y la palabra que representa en castellano.

La base de datos está alojada en un servidor de la Universidad Complutense de Madrid y no se puede acceder libremente a ella, sino que es necesario usar unas credenciales (usuario y contraseña) concretas.

4.2 Servidor web

El acceso a la base de datos se realiza mediante el servicio web *getMePictograms* implementado para este proyecto. Este servicio tiene las credenciales para acceder a la base de datos y ejecutar las consultas que se requieran. De este modo,

getMePictograms hace de puente entre el cliente y el servidor de la base de datos tal y como se ve en la Figura 20. Para acceder a este servicio el cliente debe disponer de conexión a internet y tener acceso al puerto 5223.

Para obtener información de la base de datos el servicio *getMePictograms* ofrece la siguiente API:

- **getMax:** sirve para conocer la cantidad máxima de pictogramas de una determinada categoría.
- **getId:** con este método conseguiremos una lista de todos los identificadores de pictogramas de nuestra base de datos y su color asociado.
- **getUrl:** devuelve una lista de urls dada una serie de identificadores.
- **getWord:** dado un identificador y un color, devuelve la palabra asociada.

Estos métodos son los que estarán disponibles para cualquier cliente que use *getMePictograms*. De esta manera se consigue una comunicación entre el cliente y el servidor de base de datos a la vez que evitamos operaciones indeseadas sobre la base de datos.

Es un servicio web SOAP cuyo WSDL se encuentra en la dirección web http://hypatia.fdi.ucm.es:5223/axis2/services/Pictos?wsdl

4.3 Cliente

La aplicación en Android realiza todo tipo de peticiones para conocer las cantidades máximas de objetos de una determinada categoría o la dirección url donde encontrar una determinada imagen, y el servicio web es el que da respuesta a través de consultas a la base de datos.

Para la parte del cliente, a la hora de comunicarse nuestra aplicación con el servicio web, utilizamos AsyncTask, una clase proporcionada por Android para ejecutar tareas en segundo plano. Y esto es necesario porque no se permite ejecutar en el hilo principal de una actividad una tarea como la que queremos llevar a cabo, conectar con un servidor.

Existe una clase para cada acción que requiera conectarse a la red, una para cada uno de los cuatro métodos del servicio web y otra para descargar una imagen dada una dirección url. Son acciones que no se pueden ejecutar en el hilo principal porque la aplicación no puede esperar hasta que terminen de realizarse. De esta manera cuando los datos se han recibido correctamente se pasan a una función del hilo principal para que los muestre.

Capítulo V

Base de datos

V. Base de datos

En este capítulo se abordará todo lo relacionado con el servidor de base de datos introducido en el capítulo anterior. Se hablará del estado inicial de la base de datos suministrada por ARASAAC, de las modificaciones que sufrió para ajustarse a las necesidades del proyecto y de la implementación final de la misma.

5.1 Base de datos original de ARASAAC

Inicialmente, la asociación ARASAAC nos facilitó un archivo de Excel con los datos de los pictogramas y una carpeta con los archivos en formato 'png' de las imágenes correspondientes a dichos pictogramas. Los campos que contiene el archivo proporcionado son: ID Imagen, Castellano, Inglés, Francés, Catalán, Italiano, Alemán, Portugués, Portugués Brasil, Gallego, Euskera, como se puede ver más claramente en la Figura 21.

ID imagen	Castellano	Ingles	Frances
2239	abeja=2;	bee=2;	abeille=2;
2242	vestido=2;abrigo=2;	dress=2;coat=2	vêtement=2;m
2243	abuela=2;yaya=2;abuelita=2;	grandmother=	grand-mère=2;
2244	abuelo=2;yayo=2;abuelito=2;	grandfather=2	grand-père=2;p
2245	aburrido=4;aburrido=4;aburrir=3;	boring=4;bore=	ennuyeux=4;er
2246	aceite=2;aderezo=2;aceitera=4;aceite de oliva=2;	oil=2;seasonir	huile=2;ornem

Catalan	Italiano	Aleman	Portugues	Portugues Brasi	Gallego	Euskera
abella=2;	ape=2;	Biene=2;	abelha=2;	abelha=2;	abella=2;	erle=2;
vestit=2;abric=	vestiario=2;ca	Kleid=2;Mante	vestido=2;sobı	vestido=2;casa	vestido=2;abri	jantzi=2;beroki
àvia=2;iaia=2;i	nonna=2;nonn	Großmutter=2;(	avó=2;vovó=2;a	avó=2;vovó=2;a	avoa=2;avoa=2	amona=2;amoı
avi=2;iaio=2;ia	nonno=2;nonn	Großvater=2;O	avô=2;vovô=2;a	avô=2;vovô=2;a	avó=2;avó=2;a	aitona=2;aiton
avorrit=4;avorr	noioso=4;noio	langweilig=4;l	aborrecido=4;(	aborrecido=4;t	aburrido=4;ab	aspergarri=4;a
oli=2;adorn=2;	olio=2;condim	Öl=2;Gewürz=2	azeite=2;adere	azeite=2;adorn	aceite=2;adere	olio=2;maneat

Figura 21: Base de datos original de ARASAAC

Debido a que nuestra aplicación no tiene como objetivo principal la traducción de los pictogramas a diferentes idiomas, nada más que utilizamos el campo de lenguaje '*Castellano*'. De hecho tampoco era un requisito indispensable inicialmente para nuestra aplicación pero finalmente decidimos que realmente resultaría una gran ayuda

y apoyo para los usuarios de nuestra aplicación el que indicáramos el término en castellano de los pictogramas en algún momento. Los campos que utilizamos pues de la base de datos fueron *'ID Imagen'* y '*Castellano'*.

- **ID Imagen:** Contiene un identificador propio y exclusivo que ARASAAC ha asignado a determinado pictograma y que a su vez da nombre al fichero que contiene la imagen que lo representa.
- **Castellano***:* Incluye los términos en castellano que dan significado al pictograma junto con los números correspondientes a la categoría semántica de cada término. Las categorías han sido establecidas por ARASAAC y sus valores se encuentran entre el 1 y el 6, identificándose cada una con un color diferente. El formato que siguen las celdas correspondientes a este campo es:

término = valor de su categoría;

Originalmente, la base de datos contaba con un total de 9301 pictogramas, de entre los cuales 8329 pertenecían a un solo color, 886 a dos colores y 86 a tres. Las cantidades de pictogramas por color son las siguientes:

- **Color 1**, Amarillo (Nombres propios, pronombres personales): 477
- **Color 2**, Naranja (Nombres comunes, sustantivos): 6789
- **Color 3**, Verde (Acciones, verbos): 1748
- **Color 4**, Azul (Descriptivos, adjetivos y adverbios): 808
- **Color 5**, Rosa (Social, expresiones, fórmulas de cortesía): 130
- **Color 6**, Blanco (Miscelánea, preposiciones, artículos, números, letras): 407

5.2 Transformación de la base de datos de ARASAAC

El proceso seguido para transformar estos datos en lo que necesitábamos para nuestro proyecto fue algo laborioso. Hubo que corregir toda la base de datos al completo pues muchas palabras no se correspondían con el color que tenían asignado. Los pictogramas que pertenecían a más de un color, se duplicaron o triplicaron según correspondiera, pues cada color significa una palabra distinta en la mayoría de los casos. Como ejemplo el caso del pictograma 4962 cuyas categorías son 2, 3 y 4 para 'hambre',' tener hambre' y 'hambriento' respectivamente.

Como resultado de estas correcciones la división de las categorías por color para la cantidad de 10270 pictogramas quedó de la siguiente manera:

- **Color 1**, Amarillo: 473
- **Color 2**, Naranja: 6807
- **Color 3**, Verde: 1737
- **Color 4**, Azul: 724
- **Color 5**, Rosa: 136
- **Color 6**, Blanco: 393

Lógicamente, el resultado de duplicar o triplicar los datos para un mismo identificador de pictograma ya nos estaba marcando el camino a la hora de desarrollar nuestras tablas para la base de datos relacional que realizaríamos más adelante, pues el identificador de pictograma ya no es único.

La siguiente tarea efectuada aún sobre el archivo Excel fue asignar a cada uno de los pictogramas resultantes de las anteriores transformaciones una categoría temática de tal manera que quedarán mejor distribuidos en grupos más equilibrados. Los grupos elegidos para este reparto son los siguientes:

-  **Categoría 1:** Personas, familia, sentimientos, personalidad, estados de ánimo, profesiones.
-  **Categoría 2:** Naturaleza, animales, plantas, árboles, frutas, vegetales, clima, tiempo, paisajes, lugares.
-  **Categoría 3:** Cosas y partes del hogar, comida, bebida, muebles, electrodomésticos, utensilios de cocina y de limpieza.
-  **Categoría 4:** Partes del cuerpo, aspecto, sentidos, aseo personal, medicamentos, salud, ropa, calzado, complementos, joyas.
-  **Categoría 5:** Deportes, cultura, música, libros, artes, cine y TV, personajes, juegos, juguetes, material escolar, manualidades.
-  **Categoría 6:** Tecnología, aparatos, herramientas, medios de transporte, partes de vehículos.
-  **Categoría 7:** Cosas de la ciudad, edificios, establecimientos, negocio, construcciones, maquinaria, trabajo, señales.

-  **Categoría 8:** Países, banderas, comunidades, nacionalidades, religiones, festividades, celebraciones, viajes, universo, fechas, horas.

-  **Categoría 9:** Verbos comunes, letras, números, colores, formas, dinero, medidas, signos, adverbios, frases/fórmulas de cortesía.

De esta forma los campos que finalmente tiene nuestra entidad pictograma son los siguientes:

- **Id**: Identificador que sigue siendo único para cada imagen y por tanto la dirección url donde ésta se encuentra.
- **Color**: Representa la categoría de color que puede repetirse para un mismo identificador.
- **Palabra**: Significado del pictograma en castellano. Este campo va asociado a un identificador y un color determinado.
- **Categoría**: El tema o los temas (1 como mínimo y 4 como máximo) correspondientes a un identificador y un color. Son las categorías mencionadas en la página anterior.
- **URL**: El dato que contiene la dirección url que contiene el fichero 'png' con la imagen del pictograma.

5.3 Implementación de la base de datos MySQL

A continuación, en la Figura 22, podemos ver las tablas de la base de datos y las relaciones entre ellas:

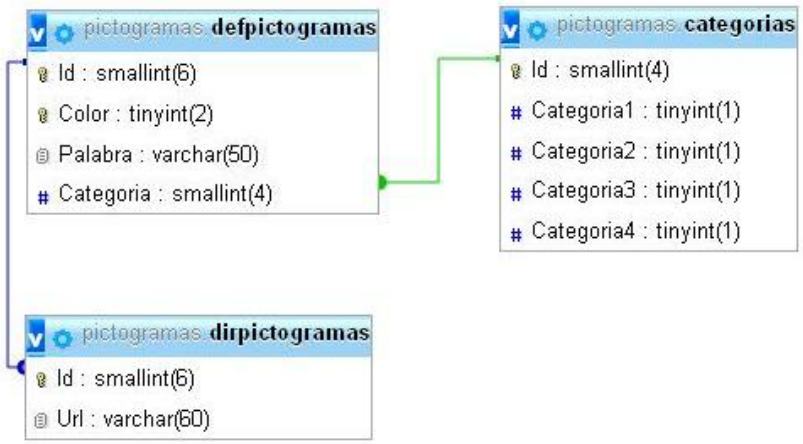


Figura 22: Diagrama entidad-relación de la base de datos

La otra parte fundamental de la base de datos corresponde a las imágenes facilitadas por ARASAAC con una dimensión de 500x500 píxeles. Pensamos que no era necesario que tuvieran un tamaño tan grande y con la aplicación 'Fotosizer' las redimensionamos a 200x200 píxeles, que creemos que es suficiente para el uso en tablets de hasta 10 pulgadas.

5.3.1 Tabla 'defPictogramas'

Posee una clave primaria doble (*Id, Color*) y a parte de estos dos campos, contiene *'Palabra'* y *'Categoria'*, identificador para la tabla de categorías temáticas.

Id	Color	Palabra	Categoría
2239	2	abeja	2
2242	2	abrigo	4
2243	2	abuela	1
2244	2	abuelo	1
2245	3	aburrir	1
2245	4	aburrido	1
2246	2	aceite	3
2247	2	acera	7

Figura 23: Extracto de la tabla 'defPictogramas'

```
-- Sentencia SQL de creación
-- Estructura de tabla para la tabla `defpictogramas`
--

CREATE TABLE IF NOT EXISTS `defpictogramas` (
`Id` smallint(6) NOT NULL,
`Color` tinyint(2) NOT NULL,
`Palabra` varchar(50) NOT NULL,
`Categoria` smallint(4) NOT NULL,
PRIMARY KEY (`Id`,`Color`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

5.3.2 Tabla 'categorias'

Tiene como clave primaria un identificador *'Id'* y los datos *'Categoria1', 'Categoria2', 'Categoria3' y 'Categoria4'*. Estas categorías corresponden a las descritas en el punto "5.2 Transformación de la base de datos de ARASAAC" de este documento. Curiosamente el único pictograma que hemos considerado que debía estar en cuatro categorías diferentes es el de 'comunicación aumentativa', que aparece en los grupos de personas, cuerpo, cultura y tecnología.

Id	Categoria1	Categoria2	Categoria3	Categoria4
89	8	9		
9	9			
256	2	5	6	
278	2	7	8	
378	3	7	8	
456	4	5	6	
1456	1	4	5	6

Figura 24: Extracto de la tabla "categorías"

```
-- Sentencia SQL de creación
-- Estructura de tabla para la tabla `categorias`
--

CREATE TABLE IF NOT EXISTS `categorias` (
  `Id` smallint(4) NOT NULL,
  `Categoria1` tinyint(1) NOT NULL,
  `Categoria2` tinyint(1) NOT NULL,
  `Categoria3` tinyint(1) NOT NULL,
  `Categoria4` tinyint(1) NOT NULL,
  PRIMARY KEY (`Id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

5.3.3 Tabla ‘dirPictogramas’

Corresponde a la tabla donde se almacenan las urls con dos campos únicamente, uno de ellos es el identificador del pictograma que ejerce de clave primaria, *‘Id’* y el otro campo es la dirección url que tiene asignada, *‘Url’*.

Id	Url
2239	http://hypatia.fdi.ucm.es/editor/pictos/2239.png
2242	http://hypatia.fdi.ucm.es/editor/pictos/2242.png
2243	http://hypatia.fdi.ucm.es/editor/pictos/2243.png
2244	http://hypatia.fdi.ucm.es/editor/pictos/2244.png
2245	http://hypatia.fdi.ucm.es/editor/pictos/2245.png
2246	http://hypatia.fdi.ucm.es/editor/pictos/2246.png
2247	http://hypatia.fdi.ucm.es/editor/pictos/2247.png

Figura 25: Extracto de la tabla "dirPictogramas"

```
-- Sentencia SQL de creación
-- Estructura de tabla para la tabla `dirpictogramas`
--

CREATE TABLE IF NOT EXISTS `dirpictogramas` (
  `Id` smallint(6) NOT NULL,
  `Url` varchar(60) NOT NULL,
  PRIMARY KEY (`Id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

5.3.4 Volcado de datos

A continuación hubo que convertir el archivo de Excel a formato ‘csv’. Con la aplicación MySQL se crearon las tablas de la Figura 22 y se importaron los datos del archivo ‘csv’. El problema de hacerlo de este modo fue que muchos caracteres no se reconocían y entre otras, las palabras con ‘ñ’ se guardaban incompletas. Hubo que crear una plantilla con Excel que contuviera los INSERT para de este modo guardarlo como fichero ‘sql’ directamente.

Una vez comprobado que los datos en MySQL eran correctos se exportaron cada una de las tablas para cargarlas en el servidor. De la siguiente manera comenzarían cada uno de los archivos sql 'defpictogramas', 'categorías' y 'dirpictogramas' respectivamente.

```
-- Volcado de datos para la tabla `defpictogramas`
--

INSERT INTO `defpictogramas` (`Id`, `Color`, `Palabra`, `Categoria`) VALUES
(2239, 2, 'abeja', 2),
(2242, 2, 'abrigo', 4),
(2243, 2, 'abuela', 1),
(2244, 2, 'abuelo', 1),
(2245, 3, 'aburrir', 1),
(2245, 4, 'aburrido', 1),
...


-- Volcado de datos para la tabla `categorias`
--

INSERT INTO `categorias` (`Id`, `Categoria1`, `Categoria2`, `Categoria3`, `Categoria4`) VALUES
(1, 1, 0, 0, 0),
(2, 2, 0, 0, 0),
(3, 3, 0, 0, 0),
...
```

```
-- Volcado de datos para la tabla `dirpictogramas`
--

INSERT INTO `dirpictogramas` (`Id`, `Url`) VALUES
(2239, 'http://hypatia.fdi.ucm.es/editor/pictos/2239.png'),
(2242, 'http://hypatia.fdi.ucm.es/editor/pictos/2242.png'),
(2243, 'http://hypatia.fdi.ucm.es/editor/pictos/2243.png'),
...
```

Capítulo VI

Servicio web getMePictograms

VI. Servicio web - getMePictograms

Ante la necesidad de comunicar el servidor de base de datos con nuestra aplicación nos decidimos por un servicio web con Axis2 de Apache. En este capítulo se describirá la API de dicho servicio y su implementación.

6.1 Implementación de los servicios

El servicio web se ha desarrollado como una clase en Java utilizando la librería de SQL mysql-connector-java-5.0.8. Para cada uno de los distintos métodos se establece conexión con el servidor a través del usuario y contraseña de que disponemos y se realizan consultas mediante sintaxis SQL, a través del método executeQuery() o executeUpdate() según convenga.

El siguiente paso es crear un fichero 'aar' con este servicio que hemos creado. El formato 'aar' es el tipo de archivo que utilizan los servicios web de Axis y mediante el plugin de Eclipse, 'Apache Axis Service Archiver', se puede convertir una clase de Java en un fichero de este tipo. Por último se coloca este fichero en nuestro servidor y ya se pueden solicitar los datos.

Este servicio web esta online y a disposición de cualquiera que quisiese utilizarlo. La dirección http://hypatia.fdi.ucm.es:5223/axis2/services/Pictos?wsdl es donde podemos encontrar su correspondiente archivo WSDL (Ver anexo A). Este archivo en formato XML describe los métodos del servicio web junto con los parámetros que necesita. Con este archivo cualquier desarrollador que desee dar uso a los métodos que hemos implementado podría hacerlo, pues es lo único estrictamente necesario, además de que el servicio web siga disponible y la base de datos correctamente organizada.

6.2 Descripción de la API

A continuación se describen los métodos que componen la API del servicio web. Estos métodos han sido desarrollados a medida que se veían necesarios para el desarrollo de la aplicación. Finalmente el servicio web consta de los cuatro métodos siguientes:

- **getMax:** sirve para conocer la cantidad máxima de pictogramas de una determinada categoría. Ver epígrafe "5.2 Transformación de la base de datos de ARASAAC".
- **getId:** con este método conseguiremos una lista de identificadores de pictogramas de nuestra base de datos y su color asociado.
- **getUrl:** devuelve una lista de urls dada una serie de identificadores.
- **getWord:** dado un identificador y un color, devuelve la palabra asociada.

Pasamos a detallar cada método y el modo en que pueden utilizarse.

6.2.1 Método getMax

Recibe como valor de entrada una categoría temática que debe estar comprendida entre 1 y 9. Devuelve la cantidad de pictogramas de esa categoría para que a la hora de pedir pictogramas al servidor, no se soliciten más una vez llegado al total de pictogramas de dicha categoría.

Por ejemplo, si efectuásemos esta llamada para conocer la cantidad de pictogramas de la categoría 'Naturaleza' **'getMax(2)'**:

El método del servicio web realizaría una consulta sql de (FROM) la tabla defPictogramas donde (WHERE) categoría es igual a 2 y la enviaría a la bases de datos. De los datos devueltos seleccionaríamos el total, COUNT(*) y este dato lo reenviaríamos a través del servicio web para que lo reciba la aplicación Android. La consulta resultante tendría la siguiente forma:

```
SELECT Id, Color, COUNT(*) AS Suma FROM defpictogramas
WHERE Categoria IN
    (SELECT Id FROM categorias WHERE Categoria1 =2 UNION
    SELECT Id FROM categorias WHERE Categoria2 =2 UNION
    SELECT Id FROM categorias WHERE Categoria3 =2 UNION
    SELECT Id FROM categorias WHERE Categoria4 =2)
```

El resultado obtenido a través de esta llamada sería **1123**.

6.2.2 Método getId

Dados una categoría temática y un valor que marca la posición a partir de la cual se comenzará, se obtiene una lista de hasta treinta pictogramas de la categoría solicitada y de esta manera se irá recorriendo la base de datos. Los valores de esta lista son identificadores y colores.

Una llamada válida sería algo así: **'getId(3, 90)'**

Nos devolvería una lista de los siguientes treinta pictogramas de la categoría 3, 'Hogar' o los que resten hasta el final de pictogramas disponibles para dicha categoría a partir de la posición 90. La consulta que se generará es la siguiente:

```
SELECT Id, Color FROM defpictogramas
WHERE Categoria IN
    (SELECT Id FROM categorias WHERE Categoria1 =3 UNION
    SELECT Id FROM categorias WHERE Categoria2 =3 UNION
    SELECT Id FROM categorias WHERE Categoria3 =3 UNION
    SELECT Id FROM categorias WHERE Categoria4 =3)
    ORDER BY Id, Color ASC LIMIT 90, 30
```

El valor 30 en este caso es una constante del servicio web, no se pasa como variable pues tenemos configurado de manera predeterminada para que sea más sencillo que siempre se pidan los pictogramas de 30 en 30 y de esta manera también será como se mostrarán en nuestra aplicación.

Esta llamada nos devolvería un objeto SOAP que es el siguiente array: **[2532, 2, 2534, 2, 2539, 2, 2541, 2, 2551, 2, 2552, 2, 2554, 2, 2555, 2, 2558, 2, 2560, 2, 2569, 2, 2570, 2, 2571, 2, 2573, 2, 2575, 2, 2577, 2, 2579, 2, 2581, 2, 2582, 2, 2583, 2, 2584, 2, 2585, 2, 2588, 2, 2592, 2, 2593, 2, 2598, 2, 2610, 2, 2611, 2, 2612, 2, 2614].**

6.2.3 Método getUrl

Toma como parámetro de entrada una lista de identificadores obtenidos de llamar al método anterior y devuelve la lista de urls para los determinados identificadores recibidos.

Para el siguiente ejemplo tomaríamos exactamente los identificadores del array anteriormente obtenido ('ids'), tal que 'int[] ids' es la lista de treinta identificadores

válidos de pictogramas. Realizamos entonces la llamada al método 'getUrl' de la siguiente manera': **'getUrl(ids)'**.

SELECT Url FROM dirpictogramas

WHERE Id IN (**2532, 2534, 2539, 2541, 2551, 2552, 2554, 2555, 2558, 2560, 2569, 2570, 2571, 2573, 2575, 2577, 2579, 2581, 2582, 2583, 2584, 2585, 2588, 2592, 2593, 2598, 2610, 2611, 2612, 2614**)

El resultado de ejecutar esta consulta es una lista de treinta direcciones url, que comenzaría de este modo: **[http://hypatia.fdi.ucm.es/editor/pictos/2532.png, http://hypatia.fdi.ucm.es/editor/pictos/2534.png, http://hypatia.fdi.ucm.es/editor/pictos/2539.png...]** y así hasta completar la lista de direcciones de los identificadores indicados como parámetros de entrada del método.

Hay que decir que el método 'getUrl' no devuelve obligatoriamente la misma cantidad de líneas que el método 'getIds', que devolvía la lista de identificadores, sino que puede que devuelva menos. Esto se debe a que del 'getIds' se pueden recibir identificadores repetidos cada uno correspondiente a un color, mientras que la dirección url es común y siempre es la misma para un único identificador.

6.2.4 Método getWord

Como valores de entrada toma un identificador y un color y el valor de salida es la palabra asociada.

Tomamos para este caso uno de los identificadores y colores obtenidos del método 'getIds' y la llamada quedaría de este modo: **'getWord(2534,2)'.**

Este método ejecuta la consulta más sencilla de todas y para el ejemplo que llevamos a cabo resultaría:

SELECT Palabra FROM defpictogramas WHERE Id =**2534** AND Color =**2**

La cadena de texto que se obtiene es **'muslo de pollo'**.

Capítulo VII

PictoEditor

VII. PictoEditor

En este capítulo se hará un recorrido por la parte de la aplicación que está en contacto directo con el usuario, esto es, desde las diferentes pantallas con las que el usuario puede interactuar hasta el algoritmo predictivo y el sistema de caché de pictogramas de la aplicación Android.

7.1 Funcionamiento

La aplicación dispone de tres pantallas para interactuar con el usuario. También cabe decir que sólo está disponible en modo apaisado pues era la única manera en que nos parecía que tenía sentido nuestra aplicación.

7.1.1 Pantalla de inicio

Ésta es bastante sencilla, pues solamente cuenta con los iconos de los grupos temáticos presentes en la siguiente pantalla. La pantalla principal que será el pilar fundamental de la aplicación. Debajo de los iconos de los grupos aparece un término, lo más descriptivo posible acerca del contenido de dicho grupo. También se puede observar un breve mensaje que da la bienvenida a los usuarios como se muestra en la Figura 26 a continuación.

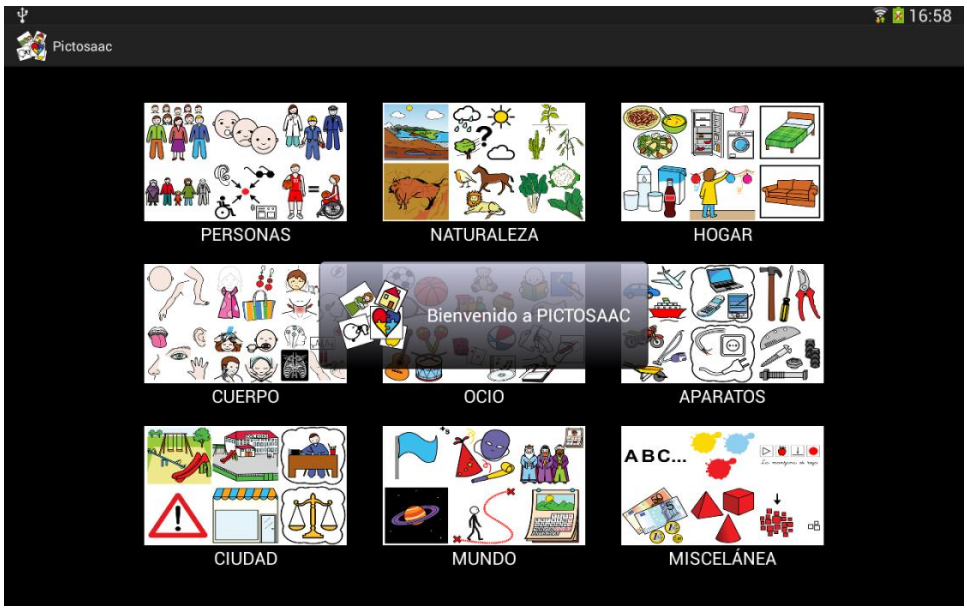


Figura 26: Pantalla de inicio de PictoEditor

7.1.2 Pantalla principal

Esta ventana será la más usada pues es a la que corresponde la introducción y carga de pictogramas y con la que más interactuará el usuario.

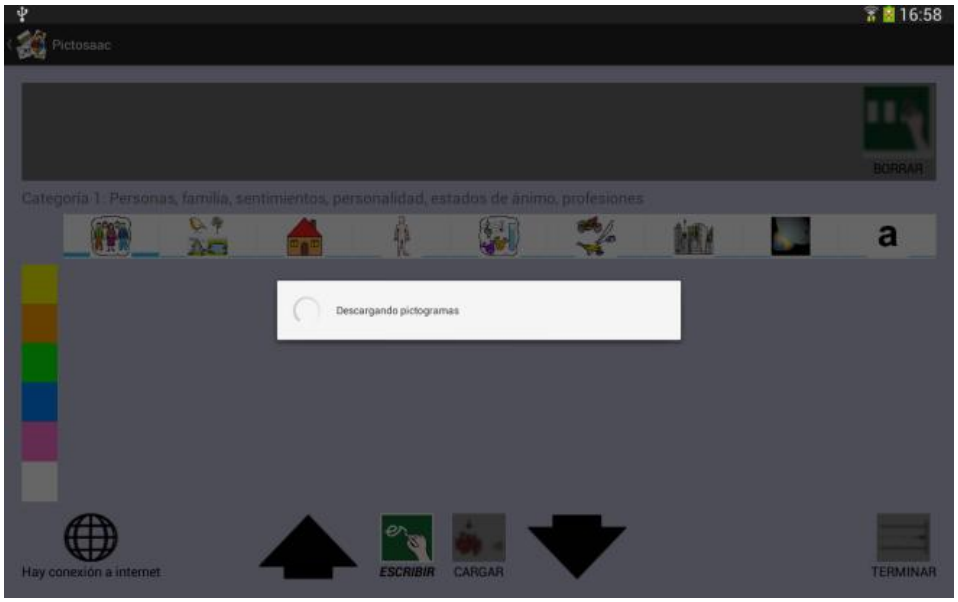


Figura 27: Diálogo de carga inicial

La primera vez que se abra la aplicación aparecerá una barra de progreso que indica que se están descargando pictogramas como se muestra en la Figura 27. Éstos son los pictogramas de inicio, que sirven para hacerse una idea de los pictogramas que el usuario encontrará en cada categoría. Si no se pudiese realizar la carga inicial, se volvería automáticamente a la pantalla de inicio. Si se realiza bien la carga, se muestran los pictogramas iniciales y desaparece la barra de progreso.

Una vez realizada la carga inicial el usuario podrá reconocer cuatro zonas en la interfaz. La primera zona (de arriba a abajo) resaltada en la Figura 28 corresponde a la pizarra donde el usuario irá componiendo la frase. Tiene la restricción de un máximo de 12 imágenes, que hemos considerado suficientes para una frase normal y dispone de una barra deslizante para los casos en que no se puedan visualizar a la vez. Según se vayan pulsando irán apareciendo la imagen del propio pictograma y debajo de ella, su significado en castellano. Se podrá borrar el último de la secuencia introducida pulsando el botón de la derecha, que es el pictograma de borrar. En la Figura 28 el botón de borrado aparece difuminado indicándonos que no existe ningún pictograma que borrar.

Figura 28: Zona superior de la pantalla principal

La segunda zona resaltada de la Figura 28 corresponde a la lista de términos que describen la categoría en la que nos encontremos. Esta categoría dependerá del botón que hayamos pulsado en la pantalla inicial o de la pestaña que hayamos pulsado una vez dentro de la pantalla principal como veremos a continuación.

En la Figura 29 se pueden ver dos zonas resaltadas principalmente. La central y más grande corresponde al cuadro de selección de pictogramas, la cual es con la que el usuario tendrá más interacción. Esta zona es un cuadro con nueve pestañas (una por cada categoría temática creada, ver epígrafe "5.2 Transformación de la base de datos de ARASAAC"). La zona bajo las pestañas carga los pictogramas contenidos en el dispositivo o en el servidor correspondiente a la categoría/pestaña seleccionada. Su comportamiento varía en función de si la aplicación está en modo escribir o modo cargar como se detallará más adelante. Este cuadro tiene una capacidad de treinta pictogramas.

Figura 29: Cuadro de selección de pictogramas y botones de colores

A la izquierda del cuadro de selección de pictogramas encontramos una sección con los colores gramaticales de ARASAAC. Cada recuadro de color es un botón que interacciona con el cuadro de selección de pictogramas, situando los pictogramas del color pulsado en las primeras posiciones.

Por último, en la parte inferior de la pantalla principal se encuentran una serie de botones e iconos tal y como se muestra en la Figura 30.

Figura 30: Zona inferior de la pantalla principal

El primero es un icono que informa al usuario si la aplicación dispone de conexión a internet o no. Este icono es meramente informativo.

A continuación encontramos cuatro botones de navegación. Dos flechas negras que cambian el contenido del cuadro de selección de pictogramas siguiendo un sistema de páginas, el botón para pasar a modo de escritura y el botón para pasar a modo de carga de pictogramas. Estos dos modos se describirán con más detalle más adelante. Se puede ver que el botón correspondiente al modo seleccionado no aparece sombreado, y el texto correspondiente (en este caso "ESCRIBIR" - Figura 30) esta resaltado en negrita y cursiva.

Por último, el botón Terminar redirige al usuario a la pantalla final, donde se muestra el mensaje compuesto y se le ofrece diferentes acciones.

7.1.2.1 Modo escritura

Este modo está activo por defecto la primera vez que se inicia la aplicación, aunque también se puede activar pulsando el botón escribir visto en la Figura 30.

Cuando la aplicación está en modo escritura, el cuadro de selección de pictogramas contiene los pictogramas que el usuario ha usado alguna vez. También contiene los pictogramas de la carga inicial si es la primera vez que se usa la aplicación o si el usuario no ha usado suficientes pictogramas como para rellenar los treinta huecos por página de los que dispone.

Este modo incorpora un sistema de predicción que ayuda al usuario a seleccionar el siguiente pictograma a escribir (ver epígrafe “7.2 Algoritmo predictivo”).

El usuario también puede borrar un pictograma almacenado en su dispositivo manteniéndolo pulsado. Se mostrará un diálogo como el de la Figura 31 y en caso de aceptar el fichero correspondiente se eliminará del almacenamiento del dispositivo. Si el usuario quisiese recuperar el pictograma tendría que hacer uso del modo carga de

pictogramas. Cuando se borra un pictograma se eliminan también sus estadísticas de uso

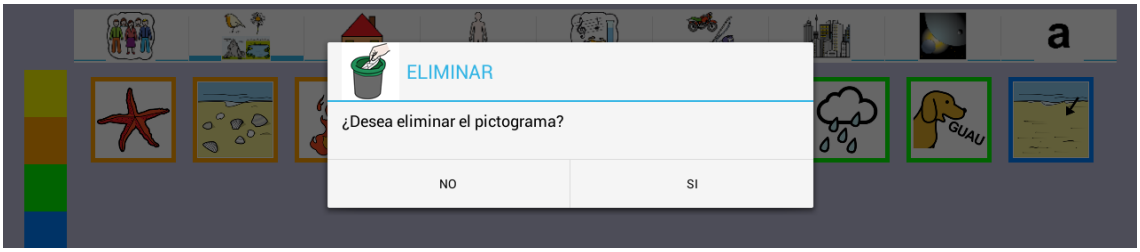


Figura 31: Diálogo borrar pictograma

Las flechas de navegación sirven para pedir a la caché de pictogramas otros treinta pictogramas o para que vuelva a mostrar los treinta anteriores. Para más detalles ver el epígrafe "7.3 Detalles de implementación - La caché".

7.1.2.2 Modo carga de pictogramas

El propósito de este modo es ofrecer al usuario pictogramas desde el servidor, que se muestran en el cuadro de selección de pictogramas, y que al pulsar en uno, se guarda en el almacenamiento del dispositivo para poder usarlo luego desde el modo escritura. Al entrar en este modo se cargan treinta pictogramas para cada categoría siempre que haya conexión a internet, que es necesaria para que funcione este modo.

Los botones de navegación por páginas (flechas negras) sirven para pedir al servidor otros treinta pictogramas indistintamente de la flecha que se pulse.

En este modo la predicción no tiene sentido, por lo que está desactivada, así como el botón de terminar ya que el usuario no está componiendo ninguna frase.

Los botones de reordenación por color y las pestañas de las categorías funcionan normalmente.

7.1.3 Pantalla final

Ésta es a la pantalla que llegamos cuando pulsamos el botón ‘TERMINAR’ de la pantalla principal. Se forma una imagen conjunta con los pictogramas que se habían introducido en la "pizarra" de la pantalla anterior. Como podemos visualizar en la Figura 32, cada pictograma aparece con su correspondiente significado en castellano bajo él, por lo que la frase es fácilmente legible por cualquier persona, indiferentemente de si conoce los pictogramas o no.

Figura 32: Pantalla final

Esta pantalla ofrece además dos acciones sobre la imagen mostrada: guardar y compartir. De los botones resaltados en la Figura 32, el de la izquierda nos permite guardar la frase generada. Se almacenará en la memoria externa del dispositivo, dentro de una carpeta denominada 'PictoEditor' que se puede abrir desde la Galería de Android o cualquier aplicación que proporcione un servicio similar. Esta imagen guardada contiene la secuencia de pictogramas así como las palabras correspondientes en castellano bajo ella. Si no ha ocurrido ningún error durante el proceso de almacenamiento se mostrará un mensaje como el que aparece en la Figura 33.

Figura 33: Mensaje guardado

El otro botón resaltado en la Figura 32 permite compartir la frase. Al pulsarlo aparece una ventana como la que se muestra en la Figura 34, que muestra todas las aplicaciones disponibles en el dispositivo a través de las cuáles se podría compartir una imagen. De este modo se puede establecer una comunicación a través de un cliente de mensajería instantánea de los que hay disponibles en Android.

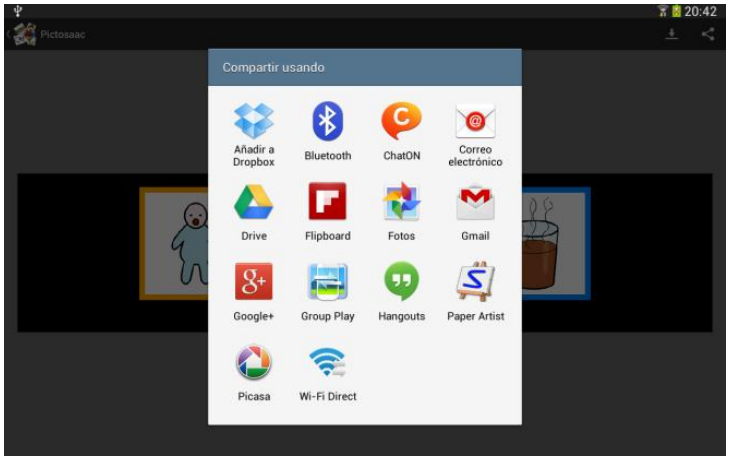


Figura 34: Diálogo compartir mensaje

7.2 Algoritmo predictivo

Como se mencionó en el epígrafe "7.1.2.1 Modo escritura", la aplicación consta de un algoritmo predictivo para facilitar al usuario la composición de frases. Este algoritmo se basa en los colores gramaticales de ARASAAC y en el número de usos de cada pictograma.

Cada vez que el usuario selecciona un pictograma en el modo escritura, internamente se aumenta el contador de número de usos para ese pictograma. Estos datos se guardan en una estructura de datos interna de la aplicación.

Por otro lado, cuando el usuario pulsa sobre un segundo pictograma, la aplicación registra el color del primer pictograma y el color del segundo, generando una relación de conectividad que se representa internamente como contadores. Al color del primer pictograma se le incrementa el contador de "color siguiente" del color que le ha sucedido. Veámoslo más claramente con un ejemplo:

Supongamos que tenemos un pictograma de color naranja (sustantivo). En la estructura de datos interna, el color naranja tiene un contador para cada uno de los otros cinco colores. Si el siguiente pictograma pulsado es de color azul (adjetivo), en la estructura de datos del color naranja se incrementará el contador correspondiente al color azul. Esto representa que la aplicación ha detectado que después de un sustantivo se ha escrito un adjetivo una vez.

El sistema pues, va aprendiendo del usuario, estableciendo relaciones entre los colores gramaticales de ARASAAC para facilitar la composición de frases. El algoritmo entra en acción cada vez que el usuario pulsa sobre un pictograma en el modo escritura. Cada vez que esto sucede, los pictogramas del cuadro de selección de pictogramas se

ordenan según su color y los datos sucesión de colores recolectados. Además, dentro de la ordenación por color se ordenan por número de usos de manera que se facilita la composición de frases rutinarias. Veamos un ejemplo:

En la Figura 35 vemos una frase en construcción. El último pictograma pulsado es de color naranja por lo que el algoritmo reordena el cuadro de selección de pictogramas para mostrar primero los adjetivos (azul), luego sustantivos (naranja) y por último los verbos (verde). Además, como se ha mencionado antes, pictogramas con mayor número de usos aparecen antes en la ordenación dentro de su color, por lo que el pictograma azul que se había escrito aparece el primero de los azules, y el primer pictograma de los naranjas aparece el primero. Por otro lado, en este ejemplo, el pictograma del sandwich sigue siendo el que tiene menos usos de los pictogramas naranjas mostrados, por lo que es el último dentro de su color.

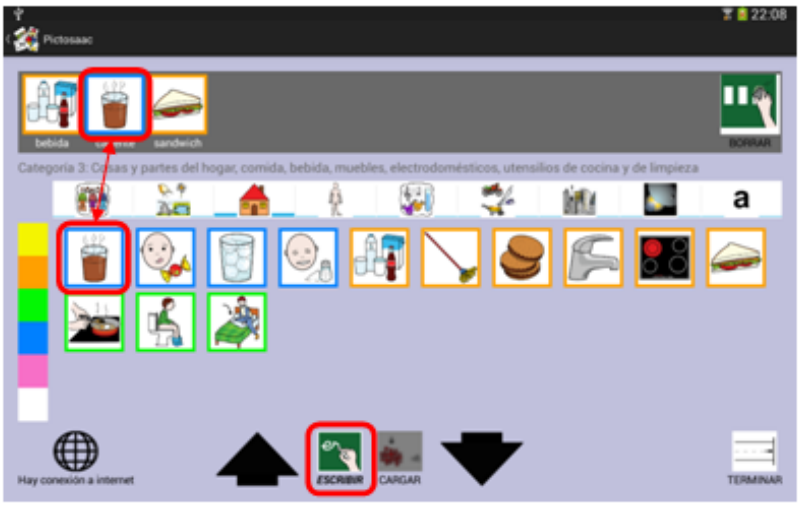


Figura 35: Ejemplo de predicción

Hay que tener en cuenta que estos datos dependen del usuario y sus costumbres, y que la escritura con pictogramas muchas veces no sigue la lógica a la que estamos acostumbrados. Por ejemplo, en castellano diríamos "El jardín es bonito" mientras que la correspondiente frase con pictogramas transcrita al castellano podría ser "El jardín es bonito", "Jardín es bonito" o "Jardín bonito" simplemente. Como se ve, en los dos primeros casos, un sustantivo va seguido de un verbo, pero en el último el sustantivo "jardín" va seguido de un adjetivo, omitiendo el verbo.

7.3 Detalles de implementación - La caché

Se ha hablado de toda la funcionalidad que ofrece la aplicación, y es por ello conveniente tratar en detalle la implementación que hace posible las labores de predicción y el almacenamiento local de pictogramas. Estamos hablando de la caché.

La caché de la aplicación tiene como funciones principales almacenar las estadísticas de uso de pictogramas y las estadísticas de colores para la predicción tal y como se explicó en el epígrafe anterior. Además, mantiene ordenados los pictogramas en función de su uso para que al mostrarlos aparezcan también ordenados. Para ello, usa y está formada por nueve tablas Hash (una por categoría) y nueve listas ordenadas de claves (una por categoría también).

Cada tabla Hash contiene un conjunto de pares clave-valor correspondiente al par [Id de pictograma - número de usos], que almacena el número de veces que se ha hecho uso de un pictograma en una categoría concreta. Estas tablas Hash están vacías inicialmente, y se van llenando con los pares clave-valor a medida que el usuario va usando pictogramas. Si el usuario borrase un pictograma, la correspondiente entrada en la tabla Hash se borraría también, perdiendo las estadísticas de uso del mismo, lo cual se ha creído conveniente para asegurar que la caché no crece de tamaño descontroladamente. Por otro lado, si el usuario ha borrado un pictograma se intuye que es porque o bien no lo usa, o bien ha añadido un pictograma que al no caber sustituye al menos usado, por lo que no tiene sentido conservar información de algo que no se usa.

Por otro lado y asociada a cada categoría, existe una lista ordenada que contiene las claves (Identificadores de pictogramas) ordenada por número de uso en la categoría en cuestión. Cada vez que el usuario pulsa sobre un pictograma se actualiza el valor de número de usos en su entrada en la tabla Hash correspondiente y se comprueba si supera el número de usos de su predecesor en la lista ordenada de claves. Si esto es así se reordena la lista de claves. Nótese, aunque pueda resultar obvio, que el valor de número de usos de un pictograma corresponde al uso de un pictograma en cierta categoría (ya que un pictograma puede pertenecer a varias categorías) y no a su uso global en la aplicación.

Además de las estructuras necesarias para la predicción, la cache tiene una lista de punteros al primer pictograma mostrado de cada categoría. Con estos punteros se controla el comportamiento de las flechas de navegación situadas en la parte inferior de la Pantalla principal. Cuando en el modo escritura se pulsa la flecha hacia abajo (siguiente página) el puntero correspondiente a la categoría en la que el usuario se

encuentra se incrementa en treinta (número de pictogramas por página), a menos que no haya suficientes pictogramas.

7.4 Evaluación del algoritmo de predicción

Aunque la intuición apunta a que el algoritmo de predicción ideado aporta bastante utilidad, se acordó necesaria la elaboración de unas pruebas sobre la labor predictiva que realiza, para poder asegurar que de verdad es un algoritmo útil.

Para ello se eligieron cuatro grupos de treinta pictogramas que se usarían para componer frases. Cada grupo presentaba unas proporciones en cuanto a colores acordes con el total de la base de datos de ARASAAC. Se decidió limitar el número de pictogramas por grupo dado que tener más de treinta no influía en las medidas que se iban a tomar, y por otro lado, la aplicación estaba preparada para ello por lo que no pareció conveniente cambiar el código a esa altura del desarrollo.

Las frases usadas fueron extraídas de unos documentos publicados por (Ana Heredia Sanz s.f.) en la web de ARASAAC. Para introducir los pictogramas en la aplicación y realizar la evaluación se realizó una carga inicial en la que cada grupo de pictogramas fue alojado en una categoría distinta sin tener en cuenta a categoría realmente pertenecían. Esto se decidió así dado que el algoritmo predictivo se basa en las estadísticas de colores, que son independientes de la categoría de los pictogramas. Por otro lado, como se verá más adelante, esto era necesario para poder realizar las labores de entrenamiento y evaluación.

Inicialmente, el proceso de evaluación consistió en comparar ciertas estadísticas al usar la aplicación con y sin predicción, pero se llegó a la conclusión de que al realizar las pruebas sin predicción, la disposición de los pictogramas dependería únicamente del orden de inserción en la carga inicial. Una vez cargados los pictogramas, se procedió al entrenamiento del algoritmo con tres de los cuatro grupos de pictogramas. Este entrenamiento consistió en la composición de diferentes frases con los pictogramas de cada grupo, para que las estadísticas de colores se fuesen rellenando. El total de frases usadas en el proceso de entrenamiento fue de veintiuna.

Una vez entrenado el algoritmo, procedimos a la escritura de otras seis frases para las cuales medimos el número de veces en que el color esperado aparece el primero y el número de veces en el que el pictograma buscado se encontraba en la primera fila con respecto al total de operaciones/pulsaciones sobre pictogramas. Las pruebas arrojaron los siguientes resultados:

Con respecto a un total de veintinueve pulsaciones, para la primera medida se obtuvieron veintiún casos en los que el color esperado resultaba ser el primero de la ordenación, lo cual constituye un 72,4% de aciertos. Muchos de los errores se debían a que las frases extraídas contenían bastantes preposiciones, algo no demasiado común en las frases creadas por los usuarios de este sistema. Tras una preposición normalmente iba un sustantivo, lo cual era predicho correctamente, pero según la estructura de las frases tomadas, tras un sustantivo podía haber tanto un verbo como una preposición con una proporción cercana al 50%/50%. Por ello, para un usuario que no suela usar preposiciones en sus mensajes, la tasa de acierto para este criterio sería superior.

Para la segunda medida, se obtuvo que en diecinueve de las veintinueve pulsaciones el pictograma buscado se encontraba en la primera fila, lo que supone un 65,5% de aciertos. Cabe considerar que en muchos de estos casos el pictograma que correspondía escribir era un sustantivo, los cuales son con diferencia los más comunes, por lo que normalmente ocupan más de una e incluso dos filas del cuadro de selección de pictogramas. En los demás casos, al haber menos pictogramas suelen estar situados en la primera fila.

En un principio este procedimiento se pretendía repetir para cada uno de los cuatro grupos, entrenando con los otros tres y evaluando con el elegido para esa iteración. Pero al analizar los conjuntos de frases se estimó que la variación en la estructura de las frases era mínima, por lo que las estadísticas finales realizadas con la media de las cuatro medidas para cada criterio no variarían significativamente. Por ello, y por motivos de calendario se decidió realizar solo una iteración de este proceso.

7.5 Casos de uso y diagramas de actividad

A continuación se hace un recorrido por los principales casos de uso de PictoEditor, aportando para cada uno una tabla con diferentes datos, una descripción en lenguaje natural del caso de uso en cuestión y el correspondiente diagrama de actividad.

En las descripciones, se hace especial hincapié en los diferentes mensajes de error y feedback en general que se da al usuario de la aplicación, ya que se ha considerado que, tratándose de este proyecto, era bastante importante.

7.5.1 CU01 - Iniciar pantalla principal

Este caso de uso corresponde a iniciar la pantalla principal y su diagrama de actividad se puede ver en la Figura 36.

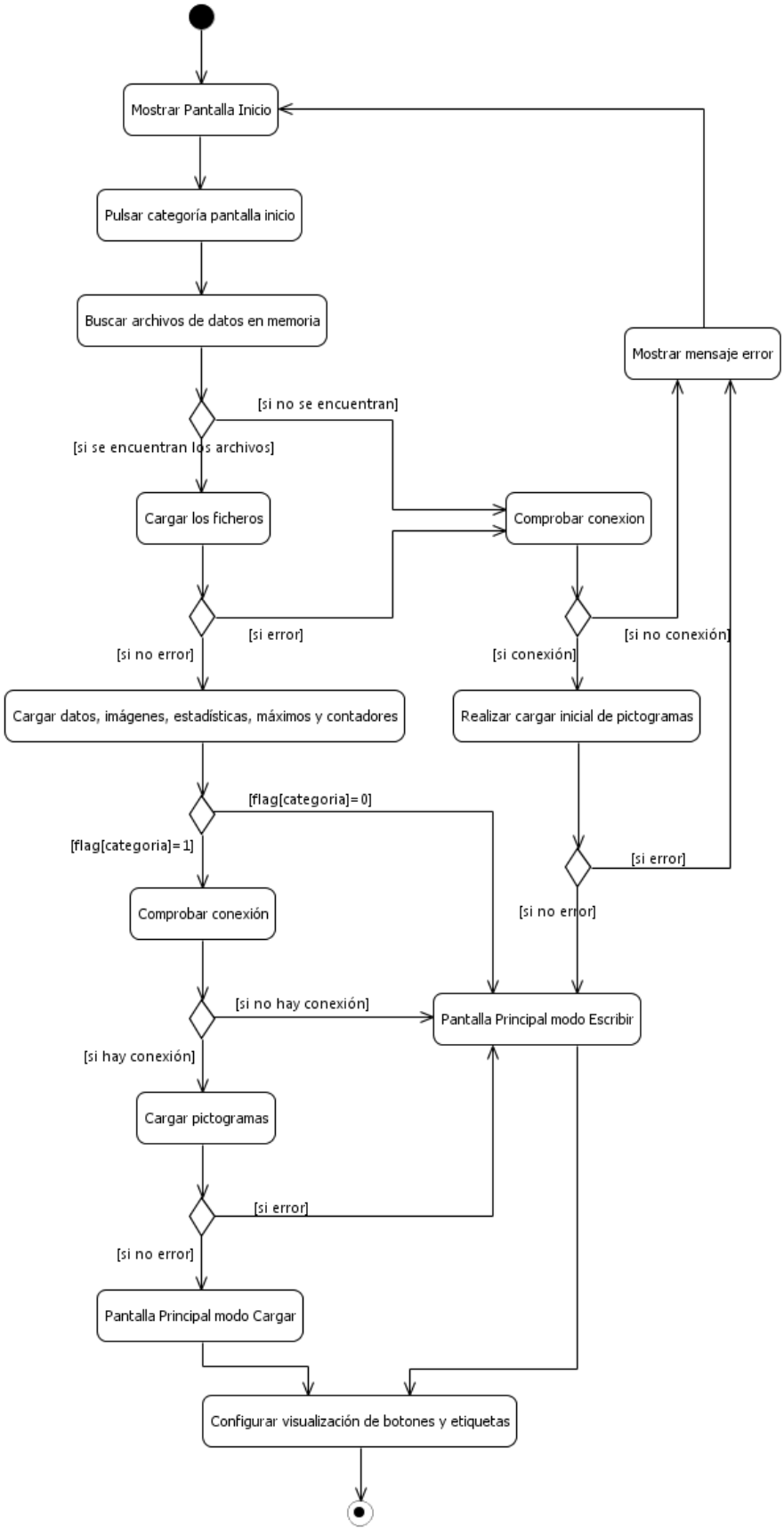


Figura 36. Diagrama de actividad 'Iniciar pantalla principal'

Nombre: Iniciar pantalla principal	**Prioridad:** Alta
Función: Iniciar	**Estabilidad:** Alta
Descripción: Iniciar la pantalla donde se realizar la mayor parte de operaciones de la aplicación	
Entrada: Categoría temática	**Salida:** Pantalla de la aplicación
Origen: Operador del sistema	**Destino:** Sistema
Necesita: Datos en memoria o conexión a internet	
Acción: Mostrar los pictogramas almacenados en la memoria del dispositivo	
Precondición: En el caso de no tener datos de la aplicación, necesita conexión a internet para acceder a la base de datos de pictogramas.	
Postcondición: Ninguna	
Efectos laterales: Ocupar memoria interna del dispositivo	

Existen varias situaciones cuando se inicia la actividad de la Pantalla Principal. El primer caso se da cuando el usuario no dispone de conexión a internet o cuando alguno de los ficheros que se cargan está corrupto. Si algún fichero está corrupto se comprueba si la aplicación dispone de conexión a internet, y en caso afirmativo se procede a realizar la carga inicial. En caso de no disponer de conexión a internet se muestra un mensaje de error y se redirige al usuario a la pantalla de inicio.

Si no se encuentra ningún fichero de pictograma en el dispositivo y conexión a internet, se descargarán los pictogramas de la carga inicial para que el usuario sepa cómo comenzar a utilizar la aplicación. Si sucede algún error durante la descarga, se le notificará al usuario y éste deberá volver a iniciar el proceso de descarga.

7.5.2 CU02 - Cambiar de pestaña

Este caso de uso corresponde a cambiar de pestaña y su diagrama de actividad se muestra en la Figura 37.

Nombre: Cambiar de pestaña	**Prioridad:** Alta
Función: Visualizar	**Estabilidad:** Alta
Descripción: Cambiar de pestaña, equivale a cambiar la categoría de los pictogramas que se visualizan.	
Entrada: Categoría temática	**Salida:** Pantalla de pictogramas
Origen: Operador del sistema	**Destino:** Sistema
Necesita: Datos en memoria o conexión a internet	
Acción: Mostrar los pictogramas de la categoría almacenados en la memoria del dispositivo o descargados de la base de datos.	
Precondición: Tener datos de la categoría elegida de pictogramas.	
Postcondición: Ninguna	
Efectos laterales: Ninguno	

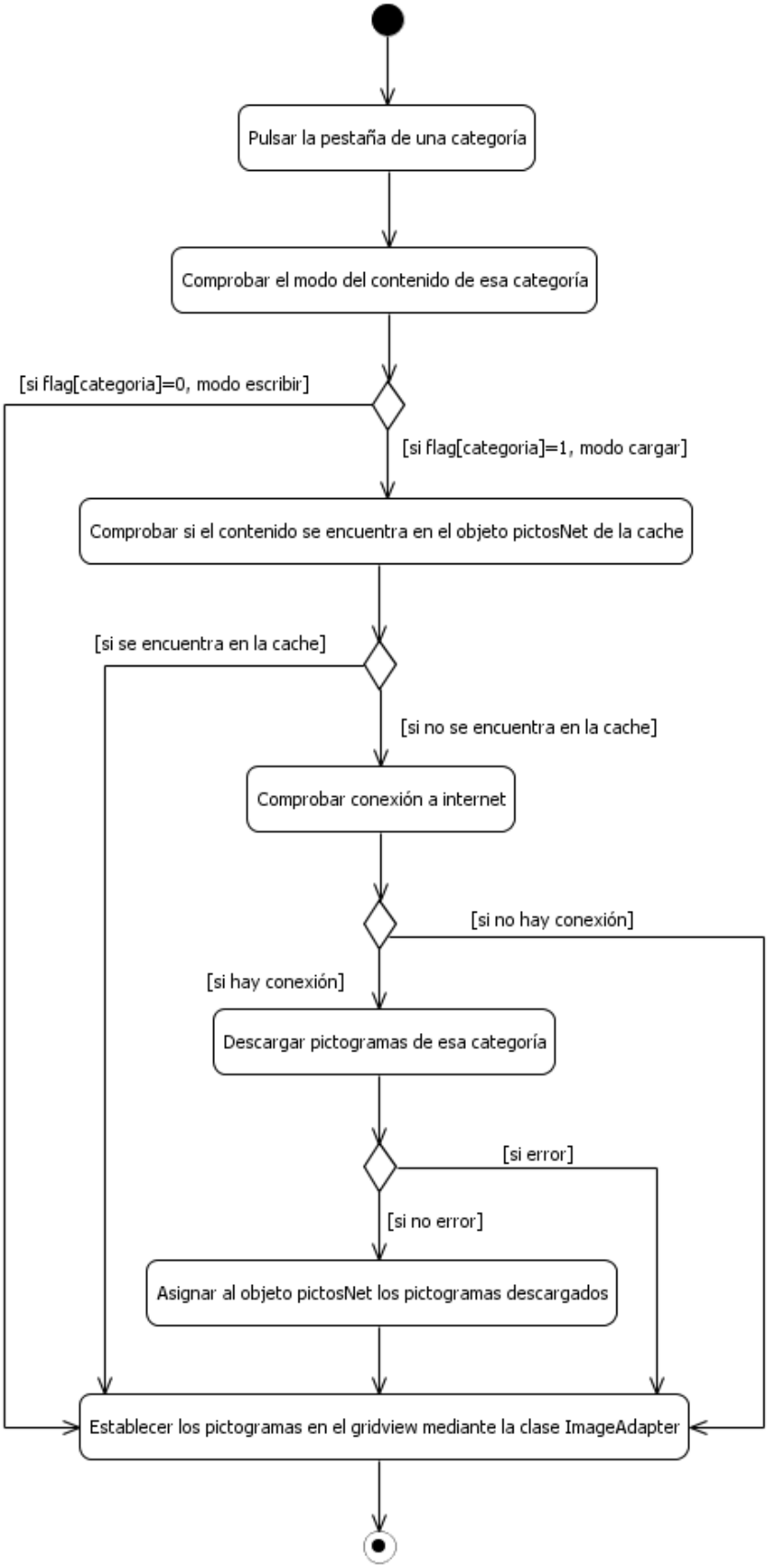


Figura 37. Diagrama de actividad 'Cambiar de pestaña'

Una vez en la Pantalla Principal, si el usuario quiere cambiar de categoría simplemente tiene que pulsar la pestaña correspondiente. Una vez hecho esto, se pueden dar dos casos, que el usuario estuviese en modo escritura o que estuviese en modo cargar.

El primer caso es trivial, ya que la aplicación cargaría los pictogramas desde la caché y se mostrarían al usuario con normalidad.

En el caso de que el usuario hubiese dejado seleccionado el modo cargar la aplicación comprobaría si dispone de conexión a internet. De ser así, descargaría los correspondientes pictogramas y los mostraría al usuario. De no disponer de conexión a internet, la aplicación cargaría desde la caché los pictogramas que había solicitado al servidor la anterior vez que el usuario pulsó el botón de modo escritura. Esto es así ya que al entrar en modo escritura con conexión a internet se cargan treinta pictogramas para cada categoría y se guardan en una estructura independiente de la caché. Se ha pensado que es preferible mostrar los pictogramas descargados anteriormente a mostrar un error.

Si no hubiese pictogramas almacenados en dicha estructura y no se dispusiese de conexión a internet, la aplicación cambiaría automáticamente al modo escritura para no mostrar un error. No obstante si el usuario pulsase el botón de modo cargar sin tener conexión a internet la aplicación mostraría un mensaje de error.

7.5.3 CU03 - Escribir un pictograma

Este caso de uso corresponde a escribir un pictograma y su diagrama de actividad se puede observar en la Figura 38.

Nombre: Escribir un pictograma	**Prioridad:** Alta
Función: Alta	**Estabilidad:** Alta
Descripción: Introducir un pictograma en la secuencia de edición.	
Entrada: Pictograma	**Salida:** Secuencia de pictogramas
Origen: Operador del sistema	**Destino:** Sistema
Necesita:	
Acción: Crear un elemento con formato de hijo de la secuencia de pictogramas y añadirlo como siguiente hijo.	
Precondición: Índice de secuencia menor que el máximo.	
Postcondición: Modificar botones de cargar y terminar según el índice.	
Efectos laterales: Ninguno	

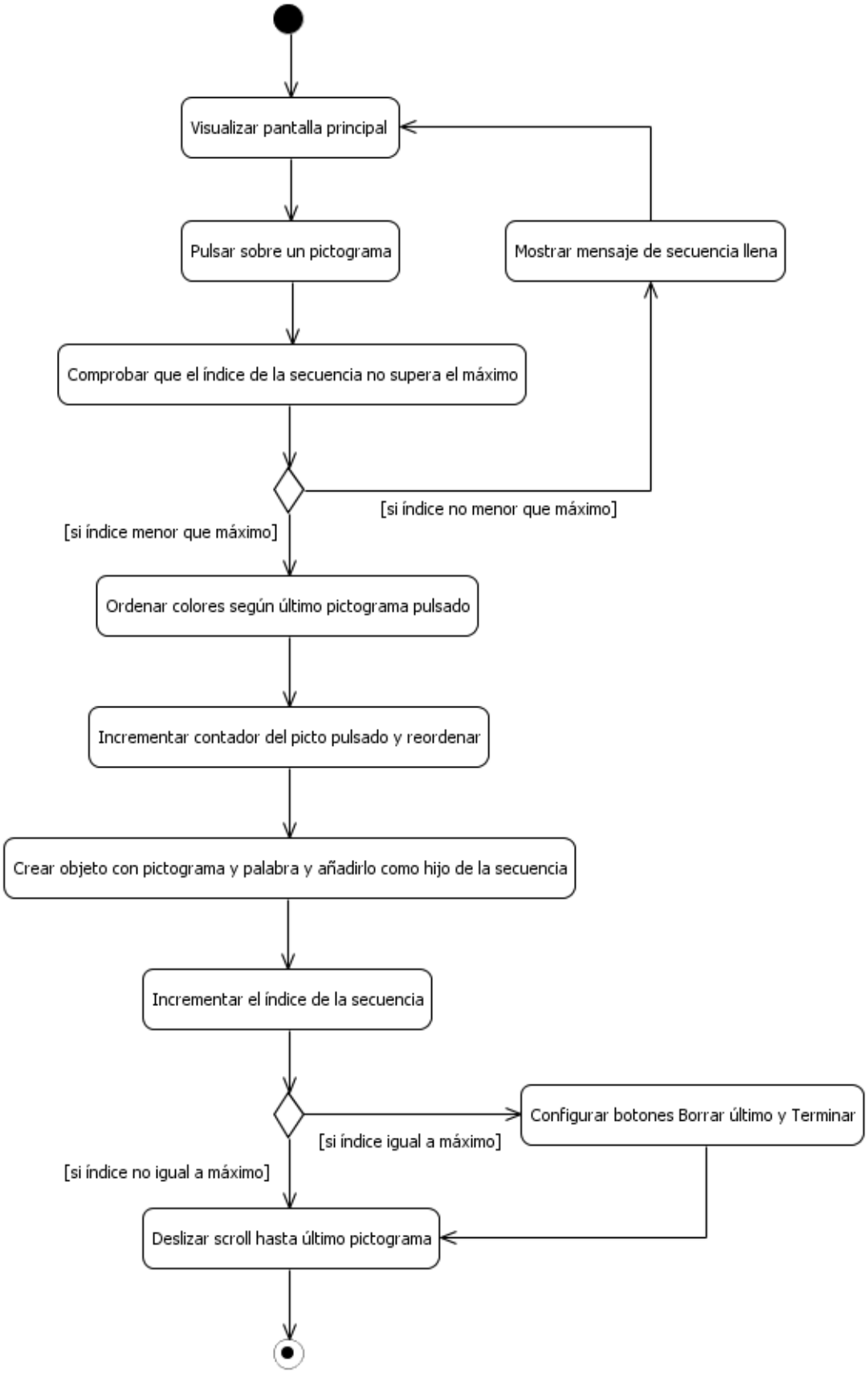


Figura 38. Diagrama de actividad 'Escribir un pictograma'

Para un editor de texto el caso de uso fundamental es la escritura de un carácter. En este caso se trata de pictogramas, pero la importancia del caso de uso es idéntica. Cuando el usuario pulsa en modo escriturо uno de los pictogramas del cuadro de selección de pictogramas, la aplicación lo añade a la pizarra y realiza una serie de operaciones ocultas relacionadas con las estadísticas de uso del pictograma y estadísticas de predicción por colores gramaticales.

En resumen, la aplicación incrementa el contador de número de usos que posee el pictograma y reordena el cuadro de selección de pictogramas basándose en las estadísticas de predicción por color, y dentro de cada color, por número de usos de cada pictograma.

7.5.4 CU04 - Cargar un pictograma

Este caso de uso corresponde a cargar un pictograma y su diagrama de actividad se puede ver en la Figura 39.

Nombre: Cargar un pictograma	**Prioridad:** Alta
Función: Alta	**Estabilidad:** Alta
Descripción: Descargar un pictograma de la base de datos	
Entrada: Pictograma	**Salida:** Mensaje de error o de confirmación
Origen: Operador del sistema	**Destino:** Sistema
Necesita: Conexión a internet	
Acción: Descargar los datos de un pictograma, descargar la palabra que le corresponde y almacenarlo en la cache de la aplicación.	
Precondición: Ninguna	
Postcondición: Ninguna	
Efectos laterales: Ninguno	

Primero se comprueba la conexión para que en caso de que no hubiese, indicar al usuario que es necesaria para llevar a cabo la operación.

La siguiente comprobación es acerca del estado de la caché, pues si se hubiese alcanzado el máximo de pictogramas establecidos para alguna categoría, se borraría un pictograma de cada categoría no vacía. Al realizar esto alguna categoría podría quedar vacía.

Si el pictograma ya se encontraba en la caché, se muestra el correspondiente mensaje y no se ejecutarían más operaciones. En otro caso, se almacenarían los datos obtenidos del pictograma para mostrarlo en pantalla. Realmente se guarda un objeto Pictograma con todos sus atributos excepto la ‘palabra’, que se descarga a continuación. El atributo ‘contador’ se establece a 0.

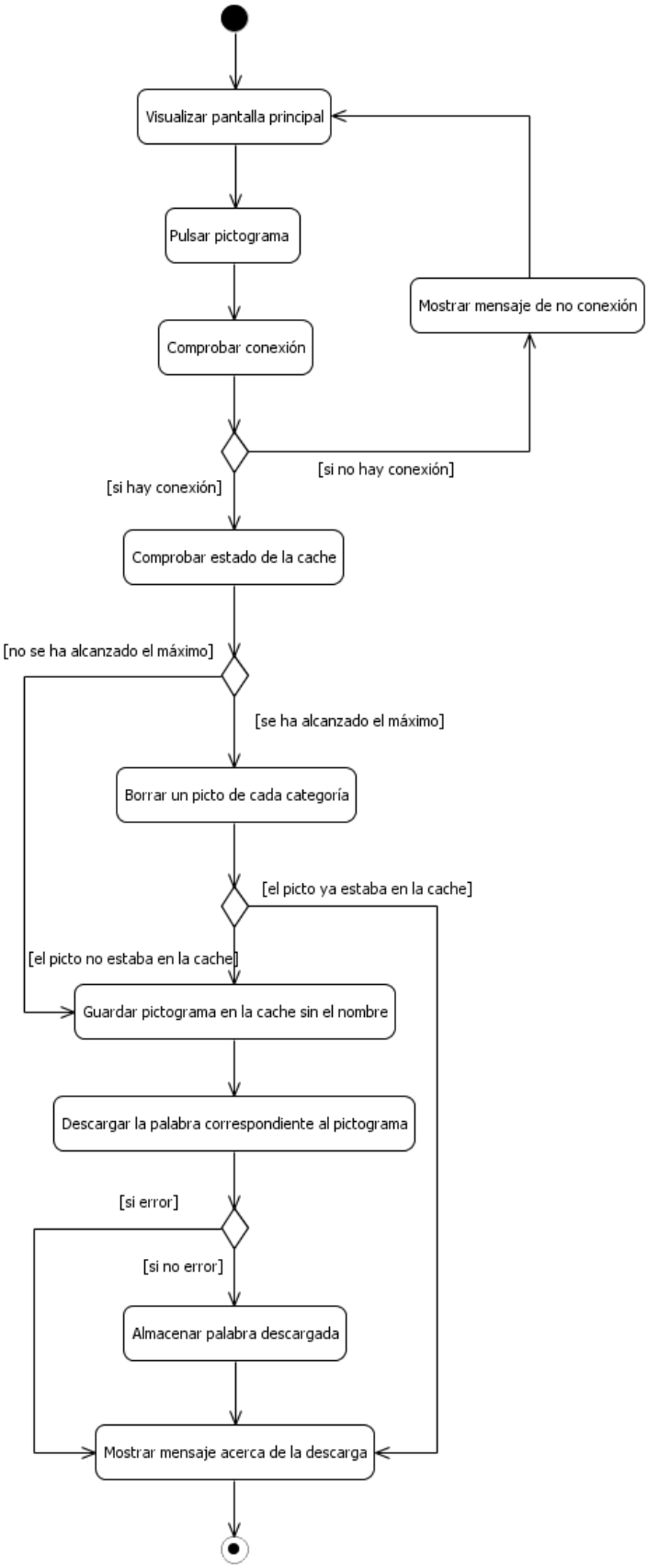


Figura 39. Diagrama de actividad 'Cargar un pictograma'

Si no ocurre ningún error durante la descarga de la palabra, se actualiza el pictograma almacenado con el valor del atributo que le faltaba y se muestra un mensaje de confirmación. Si por el contrario ocurriese un error, se borrarían los datos del pictograma de la cache y se notificaría al usuario.

7.5.5 CU05 - Retroceder la pantalla de pictogramas

Este caso de uso corresponde a retroceder la pantalla de pictogramas y su diagrama de actividad se muestra en la Figura 40.

Nombre: Retroceder la pantalla de pictogramas	**Prioridad:** Alta
Función: Visualizar	**Estabilidad:** Alta
Descripción: Visualizar una pantalla anterior de pictogramas	
Entrada: Flag[categoría], modo de la aplicación	**Salida:** Pantalla de pictogramas
Origen: Operador del sistema	**Destino:** Sistema
Necesita: Datos en memoria o conexión a internet, según el modo de la aplicación.	
Acción: Mostrar los anteriores pictogramas de la categoría almacenados en la memoria del dispositivo o descargados de la base de datos, cuando el usuario pulsa la flecha hacia arriba.	
Precondición: Ninguna	
Postcondición: Habilitar o deshabilitar botones de flecha según el contador.	
Efectos laterales: Ninguno	

Cuando el usuario pulsa en la Pantalla Principal la flecha que señala hacia arriba se pueden dar dos casos dependiendo del modo en el que esté la aplicación.

Si se encuentra en modo escritura se cargan desde la caché los treinta pictogramas anteriores y se comprueba si hay más páginas de pictogramas anteriores la actual. Si no es así, se deshabilita el botón para retroceder. En cualquier caso, se habilita el botón para avanzar.

7.5.6 CU06 - Avanzar la pantalla de pictogramas

Este caso de uso corresponde a avanzar la pantalla de pictogramas y su diagrama de actividad se puede observar en la Figura 41.

Este caso de uso es exactamente opuesto al anterior en cuanto a las comprobaciones de páginas restantes y la deshabilitación y habilitación de los botones de avanzar y retroceder.

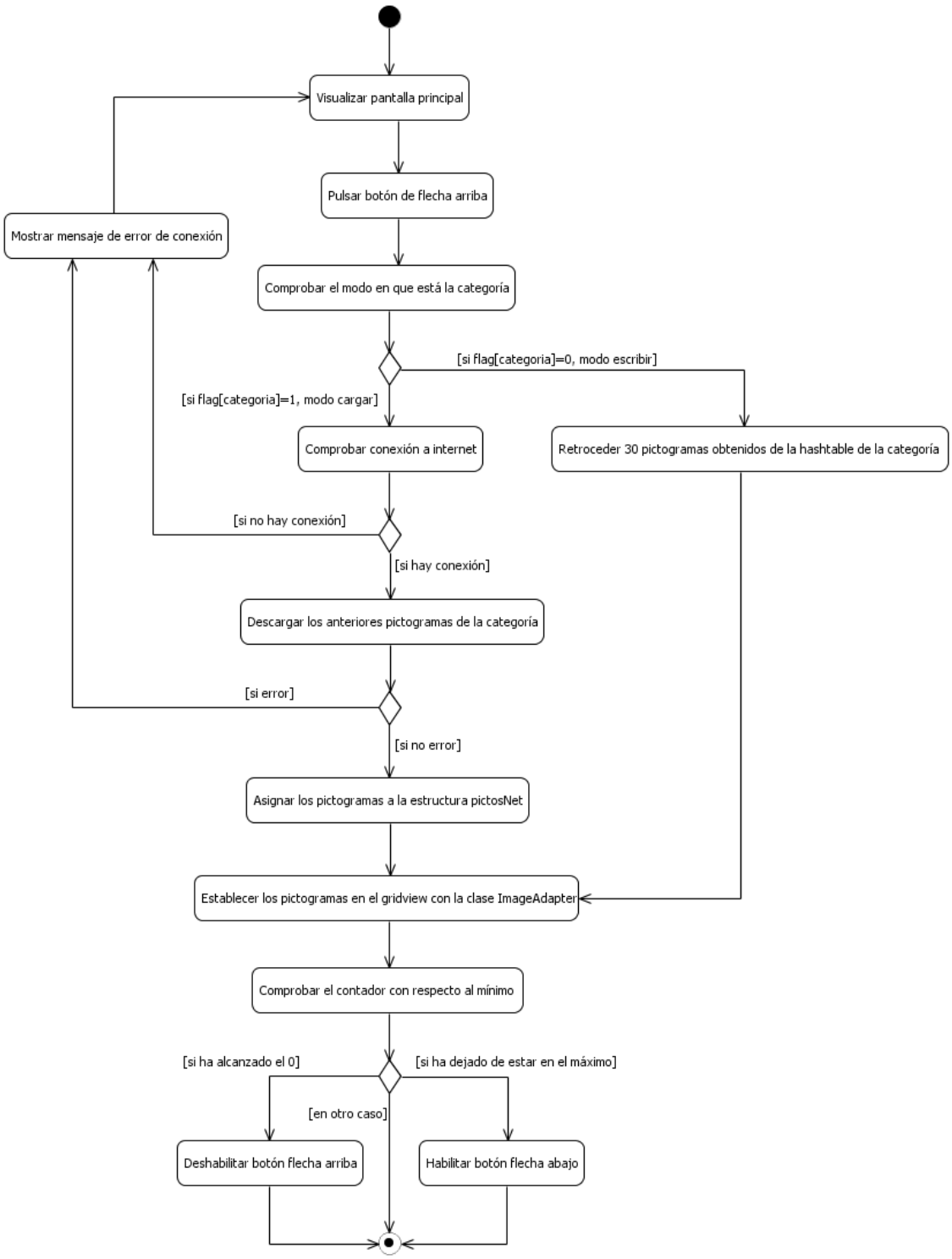


Figura 40. Diagrama de actividad 'Retroceder la pantalla de pictogramas'

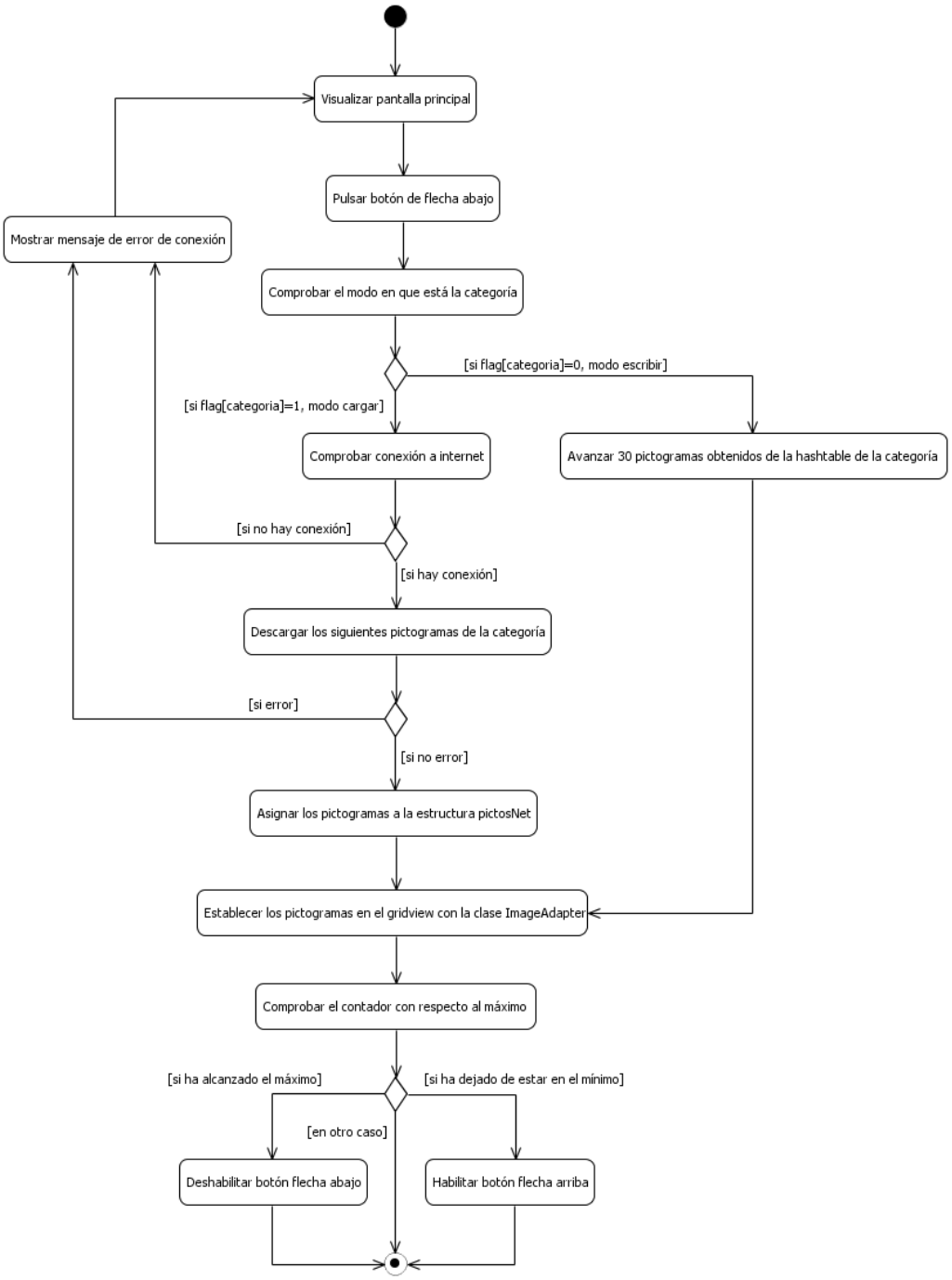


Figura 41. Diagrama de actividad 'Avanzar la pantalla de pictogramas'

Nombre: Avanzar la pantalla de pictogramas	**Prioridad:** Alta
Función: Visualizar	**Estabilidad:** Alta
Descripción: Visualizar una pantalla siguiente de pictogramas	
Entrada: Flag[categoría], modo de la aplicación	**Salida:** Pantalla de pictogramas
Origen: Operador del sistema	**Destino:** Sistema
Necesita: Datos en memoria o conexión a internet, según el modo de la aplicación.	
Acción: Mostrar los siguientes pictogramas de la categoría almacenados en la memoria del dispositivo o descargados de la base de datos, cuando el usuario pulsa la flecha hacia abajo.	
Precondición: Ninguna	
Postcondición: Habilitar o deshabilitar botones de flecha según el contador.	
Efectos laterales: Ninguno	

7.5.7 CU07 - Cambiar a modo cargar

Este caso de uso corresponde a cambiar a modo cargar y su diagrama de actividad queda representado en la Figura 42.

Nombre: Cambiar de modo	**Prioridad:** Alta
Función: Visualizar	**Estabilidad:** Alta
Descripción: Cambiar el modo en que se comporta la aplicación cuando pulsamos un pictograma.	
Entrada: Categoría temática	**Salida:** Pantalla de pictogramas
Origen: Operador del sistema	**Destino:** Sistema
Necesita: Conexión a internet	
Acción: Mostrar los pictogramas de la categoría descargados de la base de datos.	
Precondición: Tener conexión a internet si no tiene los datos en la cache.	
Postcondición: Ninguna	
Efectos laterales: Ninguno	

Ante el evento de que el usuario pulse al botón de cargar, existen cuatro posibles escenarios. Estas situaciones son las siguientes.

Si durante la ejecución de la aplicación, el usuario ya hubiese consultado el modo Cargar para la categoría actual, la aplicación tendría guardados en la caché los correspondientes pictogramas, por lo que los mostraría al usuario. Estos pictogramas permanecen en la caché hasta la detención de la aplicación o de la actividad Pantalla Principal.

El segundo escenario se da cuando la caché no tiene almacenado ningún pictograma para este cometido y además la aplicación no dispone de conexión a internet para la carga de nuevos pictogramas. En este caso la aplicación muestra un mensaje de error.

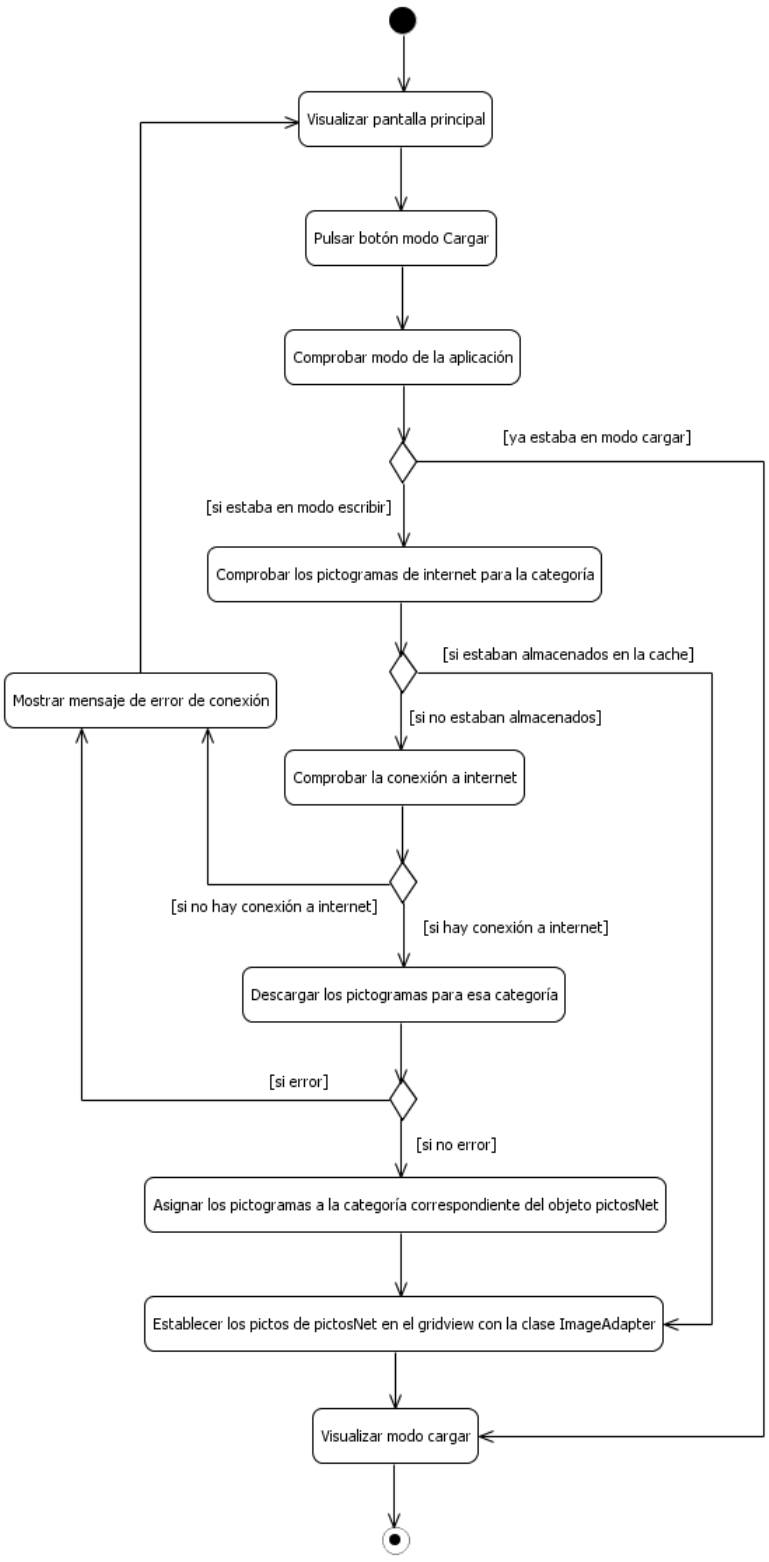


Figura 42. Diagrama de actividad 'Cambiar a modo cargar'

El siguiente caso se da cuando la conexión a internet funciona correctamente y la descarga de datos desde la base de datos se completa sin errores. Se muestran pues los pictogramas descargados para que el usuario los pueda añadir normalmente.

Por último, la aplicación ya estaba en modo cargar, simplemente se siguen mostrando los mismos pictogramas, pero no se realiza ninguna actualización de las estructuras de datos que soportan la vista.

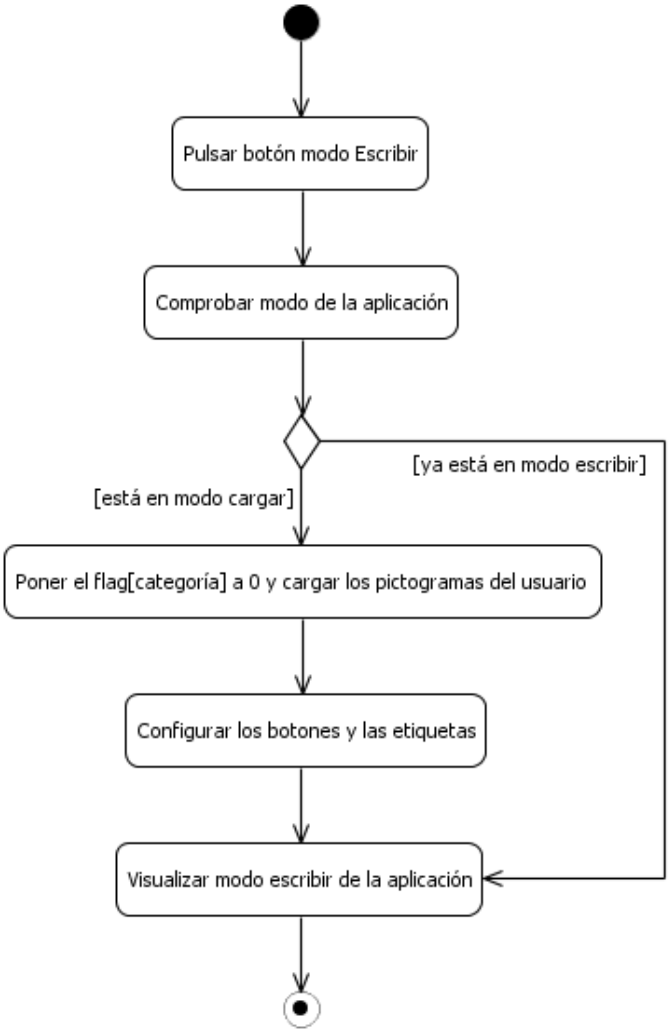


Figura 43. Diagrama de actividad 'Cambiar a modo escribir'

7.5.8 CU08 - Cambiar a modo escribir

Este caso de uso corresponde a cambiar la aplicación a modo escribir y su diagrama de actividad se puede ver en la Figura 43.

Nombre: Cambiar de modo	**Prioridad:** Alta
Función: Visualizar	**Estabilidad:** Alta
Descripción: Cambiar el modo en que se comporta la aplicación cuando pulsamos un pictograma.	
Entrada: Flag[categoría]=0	**Salida:** Pantalla de pictogramas
Origen: Operador del sistema	**Destino:** Sistema
Necesita: Acceso a memoria	
Acción: Mostrar los pictogramas de la categoría almacenados en la memoria del dispositivo.	
Precondición: Tener datos de la categoría elegida de pictogramas.	
Postcondición: Ninguna	
Efectos laterales: Ninguno	

Cuando el usuario pulsa el botón para cambiar al modo escritura la aplicación comprueba en qué modo se encuentra en ese momento. Si la aplicación se encontraba ya en modo escritura mantiene la vista sin realizar ninguna operación.

En caso contrario, carga desde la caché los pictogramas de la categoría en la que el usuario se encuentra y los muestra.

7.5.9 CU09 - Ordenar por un color

Este caso de uso corresponde a ordenar los pictogramas por un color y su diagrama de actividad se muestra en la Figura 44.

Nombre: Ordenar por un color	**Prioridad:** Media
Función: Visualizar	**Estabilidad:** Alta
Descripción: Cambiar el modo en que se presentan los pictogramas en la pantalla, y colocar primero los correspondientes a un determinado color.	
Entrada: Color	**Salida:** Pantalla de pictogramas
Origen: Operador del sistema	**Destino:** Sistema
Necesita:	
Acción: Mostrar los pictogramas de la categoría, ya sea en modo cargar o escribir, colocando en primer lugar los del color elegido.	
Precondición: Ninguna	
Postcondición: Ninguna	
Efectos laterales: Ninguno	

Al pulsar uno de los botones de ordenación por color, la aplicación comprueba el modo en el que está. Si se encuentra en modo carga simplemente agrupa los pictogramas por color, mostrando los pictogramas del color pulsado en primer lugar. Los demás colores no están ordenados por ningún criterio.

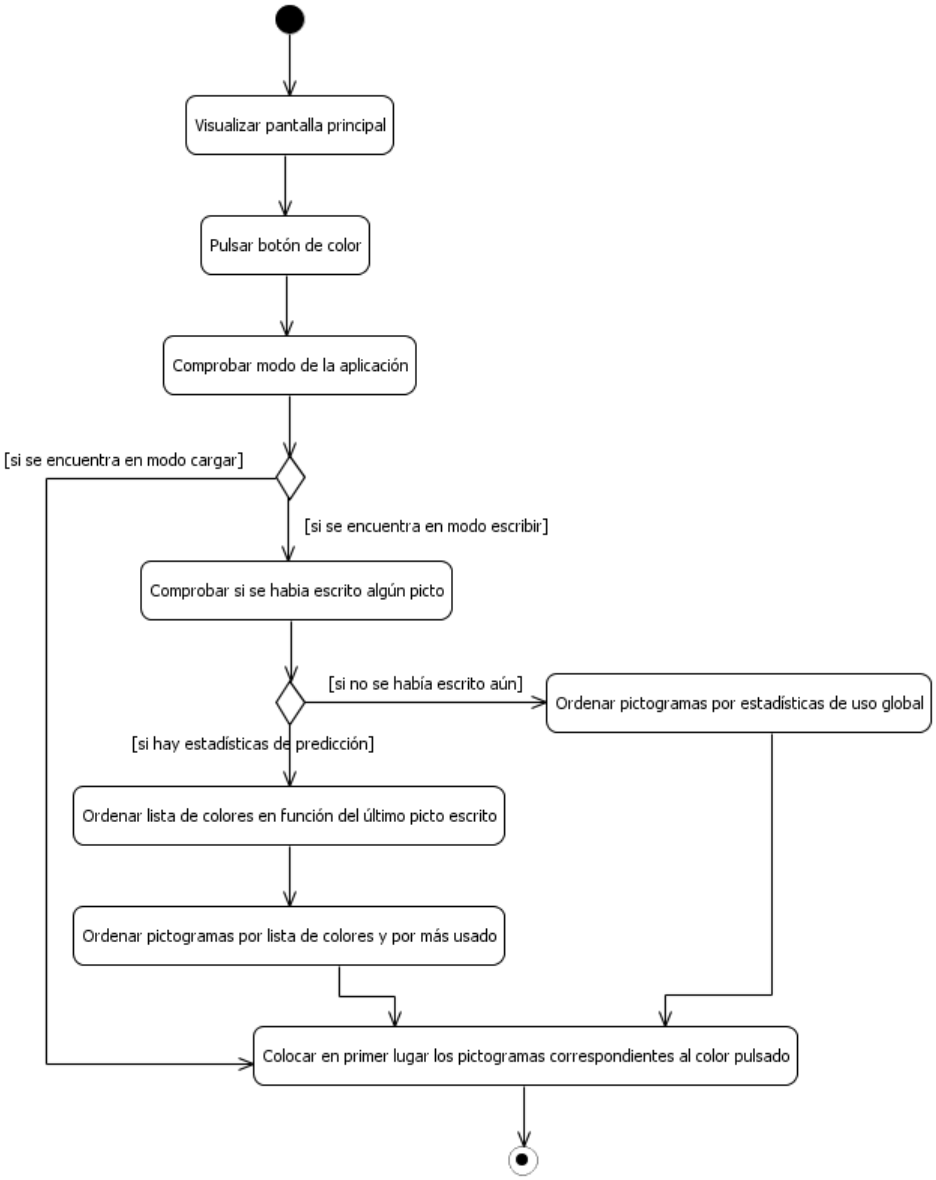


Figura 44. Diagrama de actividad 'Ordenar por un color'

Si el usuario se encuentra en modo escritura puede darse el caso de que hubiese algún pictograma escrito o de que la pizarra estuviese vacía.

Si hubiese escrito algún pictograma, se realiza el mismo proceso de agrupación por color y se sitúa el color pulsado en primer lugar, pero además, el resto de colores se ordenan en función de las estadísticas de predicción por color relativas al color del último pictograma escrito.

En cambio, si la pizarra estuviese vacía, los pictogramas del color seleccionado se situarían en primer lugar ordenados por frecuencia de uso. Los demás pictogramas se ordenarían a continuación por frecuencia de uso independientemente de su color.

7.5.10 **CU10 - Borrar último**

Este caso de uso corresponde a borra el último pictograma de la secuencia y su diagrama de actividad se puede observar en la Figura 45.

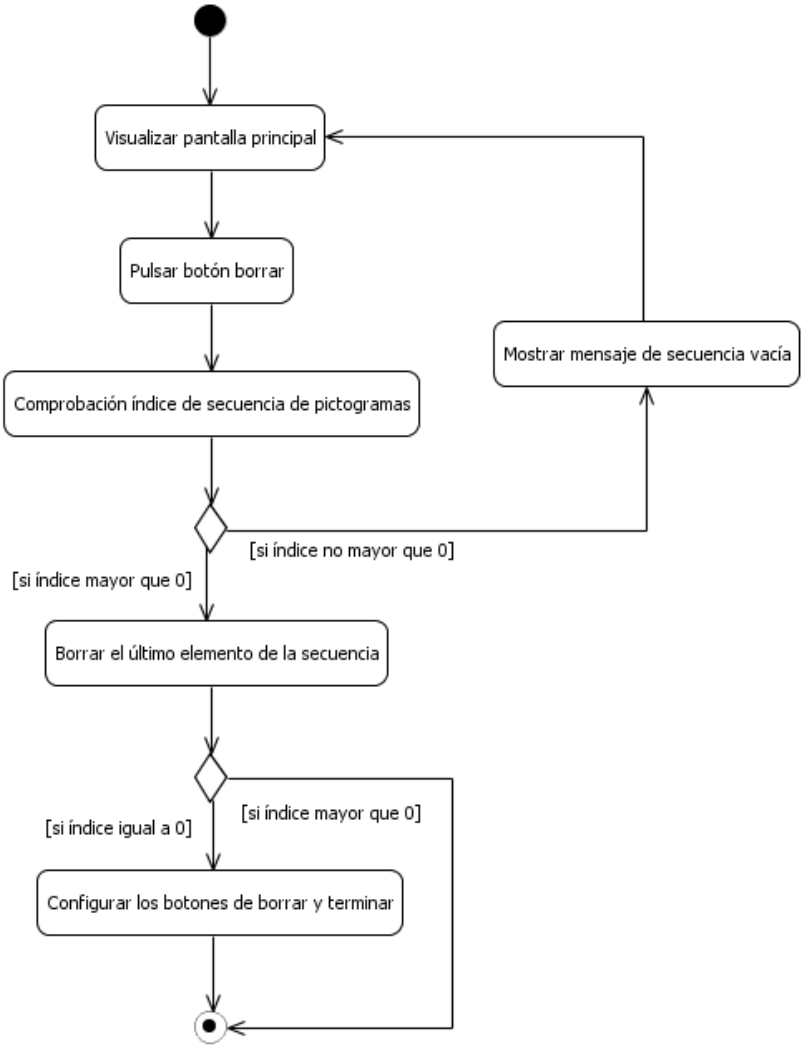


Figura 45. Diagrama de actividad 'Borrar último'

Nombre: Borrar último	**Prioridad:** Alta
Función: Baja	**Estabilidad:** Alta
Descripción: Borrar el último pictograma introducido en la secuencia de pictogramas.	
Entrada: Índice de la secuencia	**Salida:** Secuencia de pictogramas
Origen: Operador del sistema	**Destino:** Sistema
Necesita: Índice mayor que 0	
Acción: Eliminar el último hijo de la secuencia y restar uno al índice de la secuencia.	
Precondición: Haber introducido al menos un pictograma.	
Postcondición: Modificar botones de cargar y terminar según el índice.	
Efectos laterales: Ninguno	

El botón de borrar es el responsable de activar este caso de uso. Si hay al menos un pictograma en la pizarra, se elimina el último de la secuencia y se actualiza el índice correspondiente al número de pictogramas en la secuencia.

Si al borrar un pictograma la secuencia queda vacía, los botones de terminar y borrar se deshabilitan.

7.5.11 **CU11 - Terminar**

Este caso de uso corresponde a terminar de editar la secuencia en la pantalla principal y su diagrama de actividad queda representado en la Figura 46.

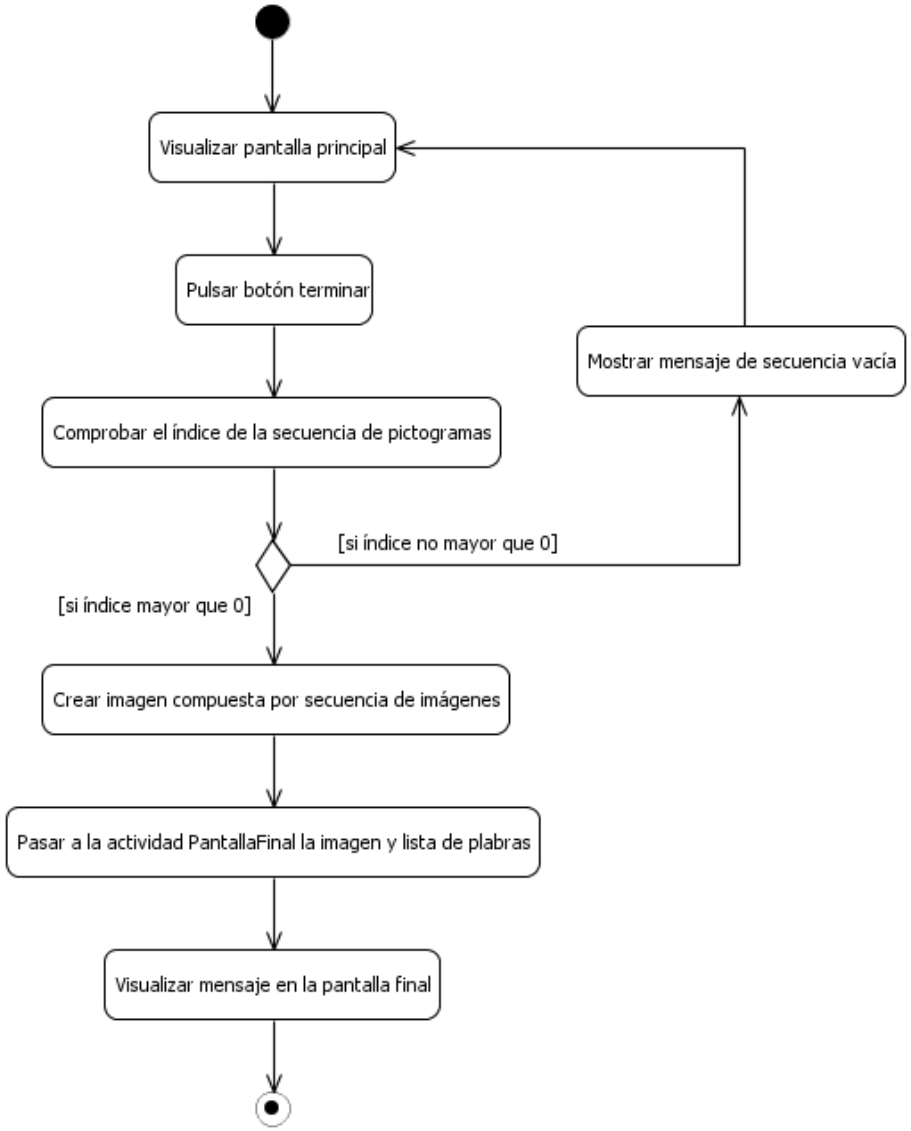


Figura 46. Diagrama de actividad 'Terminar'

Nombre: Terminar	**Prioridad:** Alta
Función: Visualizar	**Estabilidad:** Alta
Descripción: Termina de editar la secuencia de pictogramas.	
Entrada: Secuencia de pictogramas	**Salida:** Imagen de pictogramas y palabras
Origen: Operador del sistema	**Destino:** Sistema
Necesita: Índice de la secuencia mayor que 0	
Acción: Visualizar la secuencia de pictogramas en la pantalla final, ya no se edita, se podrá guardar o compartir.	
Precondición: Haber introducido al menos un pictograma.	
Postcondición: Ninguna	
Efectos laterales: Ninguno	

Al pulsar el botón de terminar la aplicación comprueba que hay al menos un pictograma en la pizarra. Si es así, compone una imagen a partir de los pictogramas añadidos a la secuencia y sus respectivos significados.

Una vez compuesta la imagen se envía a la actividad Mostrar Mensaje para que el usuario pueda guardarla y/o compartirla.

7.5.12 CU12 - Guardar

Este caso de uso corresponde a guardar la imagen de la secuencia de pictogramas y su diagrama de actividad se puede ver en la Figura 47.

Nombre: Guardar	**Prioridad:** Media
Función: Alta	**Estabilidad:** Alta
Descripción: Almacena la imagen creada a base de pictogramas y palabras.	
Entrada: Imagen de pictogramas y palabras	**Salida:** Archivo de la imagen
Origen: Operador del sistema	**Destino:** Sistema
Necesita: Acceso a memoria externa	
Acción: Guardar en memoria externa, en la carpeta de imágenes del dispositivo, la imagen compuesta por los pictogramas y palabras introducidos en la pantalla anterior.	
Precondición: Tener una secuencia de pictogramas	
Postcondición: Ninguna	
Efectos laterales: Ninguno	

En primer lugar la aplicación comprueba si tiene permisos para acceder a la memoria del dispositivo y en particular al directorio PictoEditor. Si es posible acceder al directorio solicitado, la aplicación intenta guardar la imagen y en caso de tener éxito, así lo notifica al usuario. Del mismo modo, cualquier error de acceso y/o guardado es notificado al usuario.

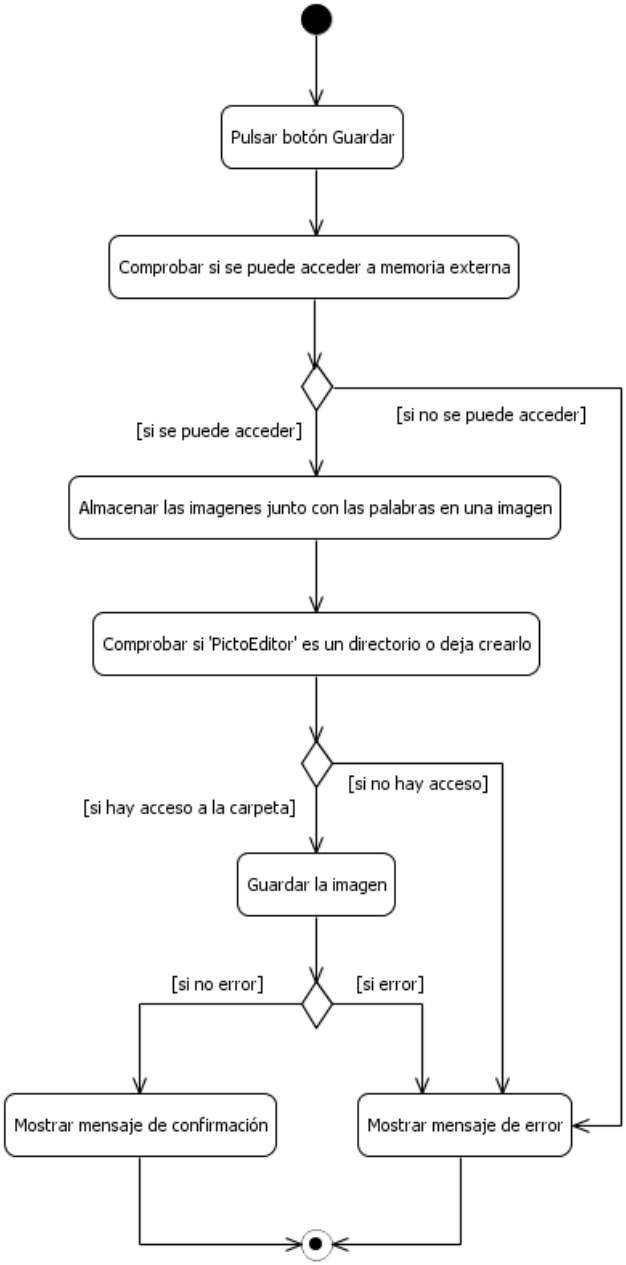


Figura 47. Diagrama de actividad 'Guardar'

7.5.13 **CU13 - Compartir**

Este caso de uso corresponde a compartir la imagen de la secuencia de pictogramas y su diagrama de actividad se muestra en la Figura 48.

Nombre: Compartir	**Prioridad:** Media
Función: Compartir	**Estabilidad:** Alta
Descripción: Comparte la imagen con la aplicación que seleccione el usuario.	
Entrada: Imagen de pictogramas y palabras	**Salida:** Archivo de la imagen
Origen: Operador del sistema	**Destino:** Sistema
Necesita: Acceso a memoria externa y conexión a internet	
Acción: Adjunta la imagen de pictogramas y la envía mediante la aplicación que seleccione el usuario dentro de las que tenga instaladas que le permitan tal operación.	
Precondición: Tener una secuencia de pictogramas	
Postcondición: Ninguna	
Efectos laterales: Ninguno	

Una vez el usuario pulsa el botón de compartir, la aplicación comprueba si tiene acceso a memoria para compartir la imagen, ya que en Android es necesario que la imagen esté previamente guardada. Dado que al pulsar el botón de terminar y pasar a la actividad Mostrar Mensaje la imagen ya se guardó, no debería producirse ningún error. A continuación la aplicación muestra un diálogo en el que el usuario puede elegir cualquiera de las aplicaciones instaladas en el dispositivo con las que puede compartir una imagen. Una vez elegido el modo de envío, el sistema redirige al usuario a dicha aplicación para que complete (si fuese necesario) los datos para el envío.

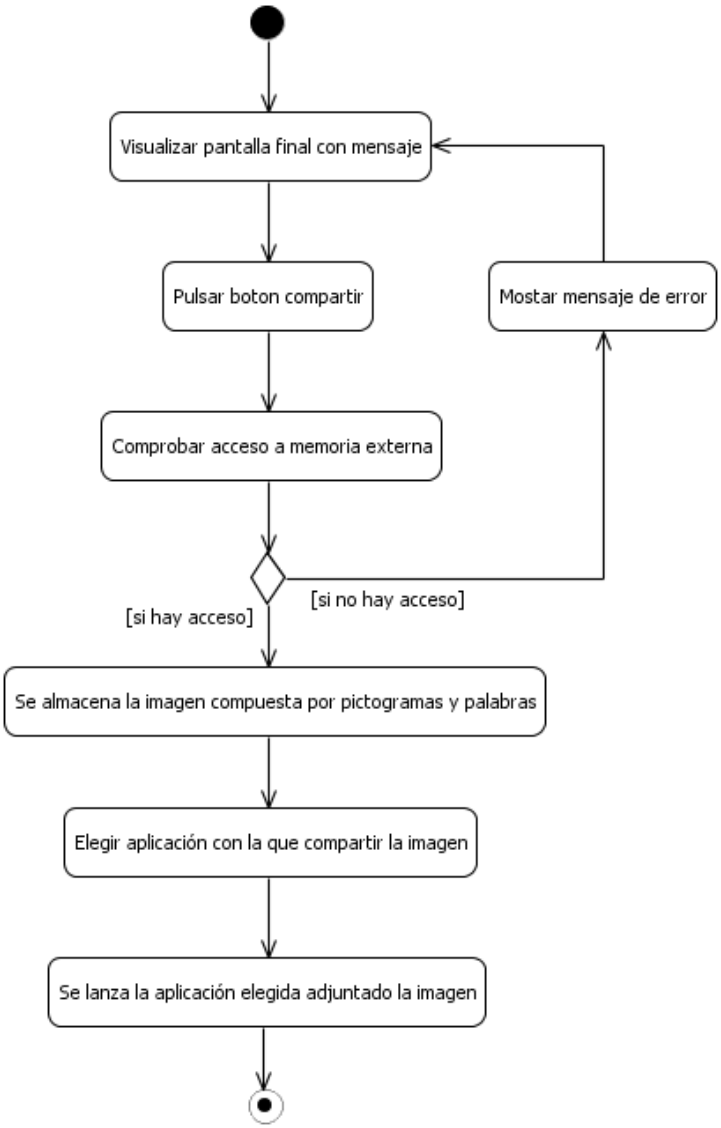


Figura 48. Diagrama de actividad 'Compartir'

Capítulo VIII

Aportación individual al proyecto

VIII. Aportación individual al proyecto

En el presente capítulo los componentes del equipo de desarrollo hacen un breve recorrido por las etapas del proyecto y las aportaciones realizadas al mismo por cada uno de ellos, contando también las impresiones que tuvieron.

8.1 **Carlos Ruíz Martín**

Inicialmente el trabajo se centró en la problemática que hay detrás del proyecto. Tuvimos que informarnos de qué recursos estaban a nuestra disposición y de las diferentes aplicaciones existentes que fuesen similares a la que pensábamos desarrollar. Decidimos en reuniones que un sistema de tableros como el que tenían la mayoría de aplicaciones no era lo más versátil, por lo que decidimos hacer algo más parecido al equivalente de un teclado para pictogramas.

También decidimos que lo mejor sería establecer un calendario de reuniones semanales para seguir el progreso y que los tutores nos pudiesen dar guías de hacia dónde deberíamos enfocar nuestro esfuerzo. También creamos una carpeta compartida en Google Drive para que los tutores pudiesen enviarnos la documentación que considerasen oportuna y para tener un repositorio gestionado por nosotros mismos.

En una de las primeras reuniones, los tutores nos propusieron hacer la aplicación en Android, y todos coincidimos en que era la mejor opción dado el caso. En esta primera etapa tuvimos que aprender Android y por tanto, Java. Por mi parte el aprendizaje se centró mucho más en Android ya que tenía conocimientos suficientes de Java.

La decisión de usar la base de datos de pictogramas de ARASAAC llevó a otra tarea de aprendizaje, ya que había que alojar dicha base de datos en el servidor que la Universidad Complutense de Madrid nos proporcionaba. Para ello tuvimos que aprender a usar un cliente FTP y realizar la inserción de la base de datos mediante una consola Linux, que es el sistema con el que nos podíamos comunicar con el servidor.

De la necesidad de comunicación entre el cliente en Android y la base de datos en el servidor surgió la necesidad de aprender a crear y usar servicios web. En nuestro caso

usamos servicios web SOAP que por suerte están bastante bien documentados por lo que el aprendizaje fue relativamente sencillo.

Una vez estuvimos familiarizados con las tecnologías, desarrollamos un pequeño prototipo usando todas ellas. El objetivo era comprobar que no teníamos problema en implementar la arquitectura cliente-servidor propuesta, y por ello la funcionalidad del prototipo era limitada.

En esta primera etapa, que corresponde a los primeros cuatro meses, también empezamos la memoria del proyecto como nos recomendaron. Dado que la mayor parte del trabajo se había centrado en documentación, y que no teníamos una versión suficientemente aproximada a la final, empezamos a escribir el estado del arte. En esta parte en concreto me dediqué a la documentación de Android y servicios web, tanto ligeros como pesados para que reflejase una idea lo más completa posible de lo que teníamos a disposición.

Durante el periodo correspondiente al segundo cuatrimestre del curso, mi disponibilidad bajó, por lo que gran parte de la implementación recayó sobre mi compañera. No obstante formé parte de la toma de decisiones de diseño que atañían a temas como los métodos del servicio web, el algoritmo de predicción y la caché de la aplicación la cual constituye una parte bastante importante ya que controla qué pictogramas están almacenados en el dispositivo así como las estadísticas de predicción y uso de pictogramas.

Una vez tuvimos desarrollado un prototipo con toda la funcionalidad de predicción que se esperaba de la aplicación, mi labor se centró en gran medida en el diseño e implementación de características que le diesen un valor adicional a la aplicación. Entre estas características se encuentran funcionalidades que podrían ser consideradas como comodidades y otras que aunque no eran realmente necesarias, aportaban acciones a la aplicación.

Las ampliaciones que se consideraron viables para una implementación a corto plazo fueron:

- Poder compartir una frase mediante cualquier método que permitiese el dispositivo.
- Guardar la frase en memoria como primer paso de un sistema de "frases favoritas".
- Eliminar un pictograma de los almacenados en el dispositivo.
- Cargar una frase guardada para poder modificarla o compartirla.

- Crear un botón de "leyenda de colores" que mostrase el significado de cada color.

De ellas, finalmente decidí implementar las dos primeras, ya que la fecha de entrega se aproximaba y todavía quedaba por hacer una parte considerable de la memoria, la cual sufriría revisiones y reescrituras.

En los últimos dos meses del desarrollo nos dedicamos a terminar la memoria lo mejor posible. Quedaban por redactar los detalles de la implementación, introducción y conclusiones, además de estas páginas de aportaciones personales. En cuanto a los detalles de implementación, mi compañera hizo una primera aproximación ya que había estado en contacto con el código mucho más que yo. Luego yo me encargué de reescribir el contenido para adaptarlo al formato narrativo del documento, además de añadir numerosos detalles que desde dentro podrían parecer obvios, pero que el lector hubiese echado en falta, por lo que en esta labor nos complementamos bastante bien.

Tras el desarrollo, nos encargamos de seleccionar las frases y pictogramas necesarios para la evaluación del algoritmo predictivo y realizamos las correspondientes pruebas. Luego me dediqué a poner por escrito los resultados y la toma de decisiones realizada durante las pruebas.

Por último, me encargué de reescribir prácticamente todo del estado del arte y añadir las correspondientes referencias. Además redacté el resumen, la introducción y conclusiones con sus pertinentes traducciones al inglés y las explicaciones de los diagramas de actividad.

8.2 **Paloma Galván Calleja**

Para empezar, con respecto a la base de datos facilitada por ARASAAC, me encargué de realizar una revisión exhaustiva, categorizando todos los pictogramas para adecuarlos a las necesidades de nuestra aplicación, seleccionando las palabras más significativas para cada pictograma. De la misma manera, realicé el diseño y la implementación de las tablas necesarias para la base de datos y las operaciones para importar dicha base de datos al servidor.

Me encargué de redimensionar con Fotosizer las imágenes correspondientes a los pictogramas de la anterior base de datos, pues el tamaño era demasiado grande para lo que exigía nuestra aplicación. Posteriormente, se colocaron en una carpeta del servidor.

Dediqué gran cantidad de tiempo a estudiar las tecnologías para desarrollar un servicio web, pues desconocía por completo ese tipo de funcionalidades. Diseñé e implementé el servicio web *getMePictograms* con base en Java y consultas sql, con los distintos atributos y métodos necesarios que aportasen las funcionalidades que requería nuestra aplicación. Posteriormente le di formato con librerías específicas y lo incluí en el servidor.

Antes de indicar mi aportación con respecto a la aplicación Android, debo añadir que tuve que dedicar previamente tiempo a estudiar tanto Java como Android, pues eran tecnologías completamente nuevas para mí. He llevado a cabo el diseño y la implementación de la primera versión completamente funcional de la aplicación PictoEditor. Esto incluye los siguientes archivos de la estructura de la aplicación Android:

- Los 3 *layouts* de las pantallas que contiene nuestra aplicación, la distribución de los elementos en cada una de ellas.
- Un *layout* que se utiliza para mandar mensajes al usuario de forma predeterminada indicando una imagen y una cadena de texto.
- Un último *layout* que sirve para dar formato a los pictogramas cuando se van añadiendo a la secuencia según se escriben.
- La clase de la caché utilizada para la gestión de los pictogramas con la memoria.
- La clase '*ImageAdapter*' que adapta los pictogramas para su correcta visualización en el *gridview* de la pantalla, junto con las operaciones que se realizan al pulsar un pictograma de este *gridview*. Se puede decir que actúa ante dos tipos de eventos, una pulsación normal y una pulsación prolongada. La pulsación normal se utiliza tanto en los modos cargar como escribir, para los cometidos que expresan sus propios nombres. La pulsación prolongada la introduje finalmente porque me pareció necesario que un usuario pueda borrar un elemento de su conjunto de pictogramas almacenados.
- La clase de las categorías temáticas para gestionar las cantidades de pictogramas de cada categoría y en ningún momento exceder dichas cantidades al solicitar información a la base de datos.
- La clase de los colore para registrar cómodamente las estadísticas de cada color y posteriormente consultarla durante la ejecución de la aplicación para poder controlar la predicción por colores.
- La clase *'Pictograma'* para crear objetos con todos los datos que manejamos acerca de los pictogramas, cuyos atributos son '*id', 'color', 'palabra', 'imagen'* y *'contador'*, que se corresponde al número de veces que se ha pulsado dicho pictograma para una categoría concreta.

- La pantalla inicial que da la bienvenida al usuario y recoge la primera selección de una categoría.
- La pantalla principal. Esta clase es sin duda la más extensa de todas, pues provee todo tipo de operaciones que gestionan la mayor parte de la aplicación. Cuenta con 5 tareas asíncronas que se encargan de llamar al servicio web para obtener datos de los pictogramas y una de ellas descarga las imágenes de los pictogramas. También se gestiona el almacenamiento de pictogramas, que no se supere un máximo para no colapsar la memoria del dispositivo. Establecí todo tipo de controles para así evitar fallos en la aplicación, como es el caso de índices, contadores, conexión a internet, acceso a datos nulos o acciones de los estados de vida de la aplicación (*onCreate, onStart, onStop, onDestroy*).

He incluido en el proyecto los diferentes archivos xml. Por un lado, los archivos xml con las medidas calculadas para 6 dimensiones diferentes de dispositivos. Esto conseguirá que la aplicación se adapte y se visualice mejor. Por otra parte, escribí los archivos xml con los diversos textos y colores que utilizamos para agilizar la gestión de los mismos. El hecho de que todos los mensajes se encuentren en un mismo archivo ayuda notablemente, pues se traduce en una mayor comodidad a la hora de introducir y modificar datos.

Por último, dentro de los archivos del proyecto, me he encargado de añadir todas las imágenes que dan dinamismo a la aplicación y sirven de botones para realizar diversas operaciones.

Tras el desarrollo, mi compañero y yo seleccionamos las frases y pictogramas necesarios para la evaluación del algoritmo predictivo y realizamos las correspondientes pruebas, que resultaron ser bastante positivas.

Dentro de la memoria, he realizado una primera versión de los apartados de “Sistemas Aumentativos y Alternativos de Comunicación” y “Uso de Pictogramas” del capítulo “Estado del arte”.

También he redactado los apartados del capítulo “Diseño e implementación” con capturas de la aplicación para su mejor comprensión. Por último he creado las tablas de los diferentes casos de uso con sus respectivos diagramas de actividad. Estos últimos los he diseñado mediante el componente Papyrus de Eclipse.

Capítulo IX

Conclusiones y trabajo futuro

IX. Conclusiones y trabajo futuro

9.1 Conclusiones

Tras familiarizarnos con los problemas que tenían los usuarios de los SAAC y estudiar las funcionalidades que ofrecían las demás aplicaciones que trabajan con pictogramas, dedujimos que era necesario crear una aplicación más versátil y que aportase mayores comodidades. Era necesario crear una aplicación que permitiese la selección de pictogramas como si se tratase de un tablero pero que al mismo tiempo predijese en cierto modo la palabra que el usuario estaba buscando. También era importante que de algún modo la aplicación pudiese mostrar la palabra asociada a cada pictograma dado que mucha gente que se tiene que comunicar con los usuarios podría no estar acostumbrada a los pictogramas.

El algoritmo predictivo basado en colores ha resultado ser una buena decisión, tal como se ve en los resultados obtenidos tras las pruebas. No obstante, pensamos que sería bastante positivo proseguir con su desarrollo para obtener mejores predicciones y predicciones no basadas en los colores adicionalmente.

La arquitectura cliente-servidor ha funcionado tal y como esperábamos, y gracias a ella ahora existe un servicio web disponible para aquel que lo desee. Creemos que esto puede ser útil para futuros desarrollos.

Con respecto a la interfaz, hemos logrado diseñar una serie de pantallas que creemos que son bastante intuitivas y cómodas para el usuario. Dadas las características de los usuarios, la interfaz no contiene ningún elemento desplegable sino que cada elemento se muestra tal y como es.

Además, decidimos que la mejor opción era desarrollar la aplicación para Android y publicarla en el PlayStore para que se pudiese beneficiar de ella el mayor número de personas posible.

9.2 Trabajo futuro

En vista de los resultados obtenidos al final del desarrollo del proyecto, es posible plantearse nuevas características para futuras iteraciones. Estas características pueden ser desde funcionalidades nuevas hasta de refuerzo o ayuda a la comprensión y manejo de la aplicación.

Uno de los principales problemas que se ha detectado es la dificultad para buscar nuevos pictogramas. A pesar de estar organizados por categorías y de poder usar los botones de ordenación por color para facilitar la búsqueda, este proceso puede ser tedioso. El pictograma buscado podría tardar en ser mostrado por el servicio web o podría estar en otra categoría. Por ello, una característica que se cree que debe estar en próximas iteraciones es la búsqueda de pictograma mediante la palabra que representa. Para ello, la aplicación usaría el teclado de Android.

Por otro lado, a pesar de que se ha supuesto que los usuarios de la aplicación serán principalmente aquellos que usen el sistema ARASAAC, podría darse el caso de un usuario que no conociese o recordase el significado de cada color gramatical de ARASAAC. Por ello, se cree que se debería incluir una especie de leyenda o botón de ayuda que mostrase el significado de cada color.

También se ha considerado el caso de que un usuario no encuentre o no le guste ninguno de los pictogramas que ARASAAC ofrece para representar algo concreto. Normalmente este caso se da al hablar de una persona concreta. Por ello creemos que una buena funcionalidad sería poder hacer una foto o cargar una existente y guardarla para que el usuario pudiese usarla como pictograma. Para ello, durante el proceso se tendría que elegir un color gramatical y una palabra que representase el pictograma. La aplicación podría tener también un mecanismo para editar estos pictogramas personalizados. Igualmente se podría considerar el modificar la palabra asociada a alguno de los pictogramas de ARASAAC si el usuario considera que otra es más representativa. En este último caso sería necesario contactar con ARASAAC para comprobar que no se vulnera ninguna licencia.

También creemos que podría ser útil poder cargar una frase guardada de modo que apareciese en la pantalla principal y se pudiese modificar como si de pictogramas individuales se tratara. Habría que implementar algún mecanismo que reconociese el texto bajo los pictogramas y pudiese convertirlo de imagen a cadena de texto.

Por último, nos gustaría que la aplicación estuviese disponible tanto en el PlayStore de Android como en la página de ARASAAC, ya que creemos que podría ser de utilidad a mucha gente.

Capítulo X

Conclusions and future work

X. Conclusions and future work

10.1 **Conclusions**

After getting familiar with the problems the AACS users had and studying the functionalities that the rest of applications that use pictograms were offering, we deduced that it was necessary to create a more versatile and comfortable application. It was necessary to create an application that would allow the user to select pictograms as if he was using a board, but at the same time it would predict in some way what word the user was looking for. It was also important that the application could show the word associated to a pictogram, as many people that have to communicate with pictogram users might not be used to them.

The color-based predictive algorithm has proven to be a good choice, as we can see in the results obtained after the tests. However, we think that it would be quite positive to continue with its development in order to obtain both better predictions and non color-based predictions.

The client-server architecture worked as we expected, and through it now exists a web service available for anyone who wishes. We think this could be helpful for future developments.

Regarding the user interface, we have succeeded in designing a set of layouts that we believe to be quite intuitive and comfortable for the user. Taking into account the characteristics of the user, the user interface does not contain any dropdown. Instead, every button is displayed as it is.

In addition, we decided that the better choice was to develop this application for Android and to publish it in the PlayStore, so that it could benefit as many people as possible.

10.2 **Future work**

Regarding the results that were obtained at the end of the development of this project, it is possible to consider new features to be included in future iterations. These features could be brand new functionalities or some kind of help for the comprehension and use of the application.

One of the main problems that has been detected is the difficulty to find new pictograms. Although of the pictograms are organized by categories, and it is possible to use the color-sorting buttons to make the search easier, this process can be tedious. The pictogram that the user is looking for could not be displayed in a reasonable time or it could be in a different category. For that reason, a feature that is desirable in future iterations is the search of pictograms by the word they represent. For that purpose, the application would use the Android keyboard.

We have been also considered the situation in which the user could not find a suitable pictogram in ARASAAC to represent something in particular. This often happens when talking about a specific person. For that reason, we think that a remarkable functionality would be to be able of making a picture or choosing one from the device to use it as a pictogram. During the process, the user would be prompted to select a color, a category and a word for that pictogram. In addition, the application could have a mechanism to edit these custom pictograms. It could be also considered the possibility of editing ARASSAC's pictograms, if the user considers that there is a better category for that pictogram, or there is another word more representative. In this last case it would be necessary to contact ARASAAC to check if it does not violate any license.

Despite we have supposed that the users of this application will be the ones who use the ARASSAC system, some user might not know or remember the meaning of every ARASAAC's grammatical color. Because of this, the application should include some kind of legend or help button that would show the meaning of each color.

We also believe that it could be useful to be able to load a saved sentence. This sentence should be shown in the proper place into the main screen, and the user should be able to edit it as if it were a sequence of individual pictograms. This would require implementing some mechanism that could recognize the text under the sentence and would be able to convert it from image to string.

Finally, we would like this application to be available both in the Android PlayStore and ARASAAC's official website, since we believe it could be helpful to many people.

Anexos

Anexo A: getMePictograms

El código correspondiente al WSDL del servicio web *getMePictograms* que se puede encontrar en la dirección:

http://hypatia.fdi.ucm.es:5223/axis2/services/getMePictograms?wsdl

```
<wsdl:definitions targetNamespace="http://ws">
 <wsdl:documentation>getMePictograms</wsdl:documentation>
 <wsdl:types>
  <xs:schema attributeFormDefault="qualified" elementFormDefault="qualified"
targetNamespace="http://ws">
   <xs:element name="getUrl">
    <xs:complexType>
     <xs:sequence>
      <xs:element maxOccurs="unbounded" minOccurs="0" name="id" type="xs:int"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="getUrlResponse">
    <xs:complexType>
     <xs:sequence>
      <xs:element maxOccurs="unbounded" minOccurs="0" name="return" nillable="true"
type="xs:string"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="getWord">
    <xs:complexType>
     <xs:sequence>
      <xs:element minOccurs="0" name="id" type="xs:int"/>
      <xs:element minOccurs="0" name="color" type="xs:int"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="getWordResponse">
    <xs:complexType>
     <xs:sequence>
      <xs:element minOccurs="0" name="return" nillable="true" type="xs:string"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="getMax">
    <xs:complexType>
```

```
      <xs:sequence>
        <xs:element minOccurs="0" name="categoria" type="xs:int"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="getMaxResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" name="return" type="xs:int"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="getId">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" name="categoria" type="xs:int"/>
        <xs:element minOccurs="0" name="posicion" type="xs:int"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="getIdResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" minOccurs="0" name="return" type="xs:int"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
 </xs:schema>
</wsdl:types>
<wsdl:message name="getMaxRequest">
 <wsdl:part name="parameters" element="ns:getMax"/>
</wsdl:message>
<wsdl:message name="getMaxResponse">
 <wsdl:part name="parameters" element="ns:getMaxResponse"/>
</wsdl:message>
<wsdl:message name="getWordRequest">
 <wsdl:part name="parameters" element="ns:getWord"/>
</wsdl:message>
<wsdl:message name="getWordResponse">
 <wsdl:part name="parameters" element="ns:getWordResponse"/>
</wsdl:message>
<wsdl:message name="getIdRequest">
 <wsdl:part name="parameters" element="ns:getId"/>
</wsdl:message>
<wsdl:message name="getIdResponse">
 <wsdl:part name="parameters" element="ns:getIdResponse"/>
```

```
  </wsdl:message>
  <wsdl:message name="getUrlRequest">
    <wsdl:part name="parameters" element="ns:getUrl"/>
  </wsdl:message>
  <wsdl:message name="getUrlResponse">
    <wsdl:part name="parameters" element="ns:getUrlResponse"/>
  </wsdl:message>
  <wsdl:portType name="getMePictogramsPortType">
    <wsdl:operation name="getMax">
      <wsdl:input message="ns:getMaxRequest" wsaw:Action="urn:getMax"/>
      <wsdl:output message="ns:getMaxResponse" wsaw:Action="urn:getMaxResponse"/>
    </wsdl:operation>
    <wsdl:operation name="getWord">
      <wsdl:input message="ns:getWordRequest" wsaw:Action="urn:getWord"/>
      <wsdl:output message="ns:getWordResponse" wsaw:Action="urn:getWordResponse"/>
    </wsdl:operation>
    <wsdl:operation name="getId">
      <wsdl:input message="ns:getIdRequest" wsaw:Action="urn:getId"/>
      <wsdl:output message="ns:getIdResponse" wsaw:Action="urn:getIdResponse"/>
    </wsdl:operation>
    <wsdl:operation name="getUrl">
      <wsdl:input message="ns:getUrlRequest" wsaw:Action="urn:getUrl"/>
      <wsdl:output message="ns:getUrlResponse" wsaw:Action="urn:getUrlResponse"/>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="getMePictogramsSoap11Binding"
type="ns:getMePictogramsPortType">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
    <wsdl:operation name="getMax">
      <soap:operation soapAction="urn:getMax" style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="getWord">
      <soap:operation soapAction="urn:getWord" style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
```

```
    <wsdl:operation name="getId">
      <soap:operation soapAction="urn:getId" style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="getUrl">
      <soap:operation soapAction="urn:getUrl" style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:binding name="getMePictogramsSoap12Binding"
type="ns:getMePictogramsPortType">
    <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
    <wsdl:operation name="getMax">
      <soap12:operation soapAction="urn:getMax" style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="getWord">
      <soap12:operation soapAction="urn:getWord" style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="getId">
      <soap12:operation soapAction="urn:getId" style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
```

```
      <soap12:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="getUrl">
    <soap12:operation soapAction="urn:getUrl" style="document"/>
    <wsdl:input>
      <soap12:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap12:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
</wsdl:binding>
<wsdl:binding name="getMePictogramsHttpBinding" type="ns:getMePictogramsPortType">
  <http:binding verb="POST"/>
  <wsdl:operation name="getMax">
    <http:operation location="getMax"/>
    <wsdl:input>
      <mime:content type="application/xml" part="parameters"/>
    </wsdl:input>
    <wsdl:output>
      <mime:content type="application/xml" part="parameters"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="getWord">
    <http:operation location="getWord"/>
    <wsdl:input>
      <mime:content type="application/xml" part="parameters"/>
    </wsdl:input>
    <wsdl:output>
      <mime:content type="application/xml" part="parameters"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="getId">
    <http:operation location="getId"/>
    <wsdl:input>
      <mime:content type="application/xml" part="parameters"/>
    </wsdl:input>
    <wsdl:output>
      <mime:content type="application/xml" part="parameters"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="getUrl">
    <http:operation location="getUrl"/>
    <wsdl:input>
      <mime:content type="application/xml" part="parameters"/>
```

```
      </wsdl:input>
      <wsdl:output>
        <mime:content type="application/xml" part="parameters"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="getMePictograms">
    <wsdl:port name="getMePictogramsHttpSoap11Endpoint"
binding="ns:getMePictogramsSoap11Binding">
      <soap:address
location="http://hypatia.fdi.ucm.es/axis2/services/getMePictograms.getMePictogramsHttpSoap
11Endpoint/"/>
    </wsdl:port>
    <wsdl:port name="getMePictogramsHttpSoap12Endpoint"
binding="ns:getMePictogramsSoap12Binding">
      <soap12:address
location="http://hypatia.fdi.ucm.es/axis2/services/getMePictograms.getMePictogramsHttpSoap
12Endpoint/"/>
    </wsdl:port>
    <wsdl:port name="getMePictogramsHttpEndpoint"
binding="ns:getMePictogramsHttpBinding">
      <http:address
location="http://hypatia.fdi.ucm.es/axis2/services/getMePictograms.getMePictogramsHttpEndp
oint/"/>
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

Bibliografía y referencias

Bibliografía y referencias

Accegal. *Accegal - Pictodroid Lite.* http://www.accegal.org/pictodroid-lite/ (último acceso: 22 de Junio de 2014).

Ana Heredia Sanz. *Materiales ARASAAC.* http://www.catedu.es/arasaac/materiales.php?id_material=96 (último acceso: 22 de Junio de 2014).

Ángel J. Vico. *Arquitectura Android.* 2011. http://columna80.wordpress.com/2011/02/17/arquitectura-de-android/ (último acceso: 22 de Junio de 2014).

ARASAAC. *¿Qué son los SAAC? Portal ARASAAC.* http://www.catedu.es/arasaac/aac.php (último acceso: 22 de Junio de 2014).

Carlos Andrés Morales Machuca. *Estado del arte: Servicios web, https://sites.google.com/site/camoralesma/articulo2.pdf?attredirects=0.* Universidad nacional de Colombia.

CATEDU. *AraWord.* http://www.catedu.es/arasaac/software.php?id_software=2 (último acceso: 22 de Junio de 2014).

—. *MICE El ratón virtual.* http://www.catedu.es/arasaac/software.php?id_software=9 (último acceso: 22 de Junio de 2014).

—. *Picto4Me.* http://catedu.es/arasaac/software.php?id_software=23 (último acceso: 22 de Junio de 2014).

—. *Portal Aragonés de la Comunicación Aumentativa y Alternativa.* http://www.catedu.es/arasaac/ (último acceso: 22 de Junio de 2014).

Departamento de Informática e Ingeniería de Sistemas del CPS de Zaragoza. *Tableros Interactivos de COmunicación.* http://www.catedu.es/arasaac/software.php?id_software=5 (último acceso: 22 de Junio de 2014).

Difasia en Zaragoza. *Cuentos con pictogramas.* http://disfasiaenzaragoza.com/cuentos/cuentos.html (último acceso: 22 de Junio de 2014).

Felipe Antonio García Ochando. *Descargar Android.* 2014. http://www.descargarandroid.com/la-historia-de-android/ (último acceso: 22 de Junio de 2014).

Fundació El Maresme, BJAdaptaciones. *El messenger virtual.* http://www.messengervisual.com/ (último acceso: 22 de Junio de 2014).

Grupo Promedia Soluciones Multimedia Para Empresa S.L. *Pictotraductor.* http://www.pictotraductor.com/ (último acceso: 22 de Junio de 2014).

Lorenzo Moreno. *PictogramAgenda.* http://www.lorenzomoreno.com/index.php/es/software/79-pictogramagenda (último acceso: 22 de Junio de 2014).

Manuel López Michelone. *UNOCERO.* 2013. http://www.unocero.com/2013/09/23/la-historia-de-android/ (último acceso: 22 de Junio de 2014).

María Jesús Lamarca Lapuente. *Hipertexto: El nuevo concepto de documento en la cultura de la imagen.* http://www.hipertexto.info/documentos/serv_web.htm (último acceso: 22 de Junio de 2014).

Marta García Azpiroz, Javier Marco Rubio. *Software que usa recursos gráficos de ARASAAC, AraBoard.* http://www.catedu.es/arasaac/software.php?id_software=8 (último acceso: 22 de Junio de 2014).

Ministerio de educación y ciencia. *Índice temático de los símbolos pictográficos para la comunicación (no vocal), http://disfasiaenzaragoza.com/pictogramas/ficheros/SPC%20Indice%20Tematico.pdf.* Toledo: Equipo específico de deficiencia auditiva.

Ministerio de Sanidad y Política Social. *Derechos de las personas con discapacidad, http://www.imserso.es/InterPresent1/groups/imserso/documents/binario/convencion_accesible2.pdf .* CEAPAT, 2010.

Noé Fernández Iglesias. «Servicios Web ligeros: alternativas al protocolo SOAP para la creación de servicios distribuídos, http://di002.edv.uniovi.es/~cueva/asignaturas/doctorado/2006/trabajos/SW_ligeros.pdf.» 2006.

Patrick Brady. *Anatomy and Physiology of an Android.* 2008. https://sites.google.com/site/io/anatomy--physiology-of-an-android (último acceso: 22 de Junio de 2014).

Pictoagenda. *Pictoagenda.* http://www.pictoagenda.com/ (último acceso: 22 de Junio de 2014).

Pictosonidos. *Pictosonidos.* http://www.pictosonidos.com/ (último acceso: 22 de Junio de 2014).

The Apache Software Foundation. *Apache Axis2 Java.* http://axis.apache.org/axis2/java/core/ (último acceso: 22 de Junio de 2014).

W3C. *W3CSchools SOAP Tutorial.* http://www.w3schools.com/webservices/ws_soap_intro.asp (último acceso: 22 de Junio de 2014).

—. *W3CSchools WSDL Tutorial.* http://www.w3schools.com/webservices/ws_wsdl_intro.asp (último acceso: 22 de Junio de 2014).

—. *W3CSchools XML Tutorial.* http://www.w3schools.com/xml/default.asp (último acceso: 22 de Junio de 2014).

Wikipedia. *Android-Wikipedia.* http://en.wikipedia.org/wiki/Android_(operating_system) (último acceso: 22 de Junio de 2014).