TFG: Análisis Emocional para la Inclusión Digital

Gema Eugercios Suárez
Paloma Gutiérrez Merino
Elena Kaloyanova Popova

Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid

Mayo 2018

Documento maquetado con TEXIS v.1.0.

Este documento está preparado para ser imprimido a doble cara.

TFG: Análisis Emocional para la Inclusión Digital

Informe técnico del departamento
Ingeniería del Software e Inteligencia Artificial
IT/2009/3

Versión 1.0

Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid

Mayo 2018

ISBN 978-84-692-7109-4

Índice

Índice de figuras

Índice de Tablas

Capítulo 1

Introducción

Resumen: En este capítulo se va a realizar una introducción al trabajo que vamos a presentar. En primer lugar, en la sección 1.1 se explicará en qué consiste el trabajo. En la sección 1.2 se expondrán las principales dificultades a las que nos hemos enfrentado y las decisiones que se han tomado para afrontarlas. Por último, en la sección 1.3 se presentará la estructura del trabajo.

1.1. Definición

El tabajo se centra en la detección automática del contenido emocional de un texto. Esto nos va a ayudar a evitar ambigüedades emocionales a la hora de interpretar un texto facilitando su lectura a personas con algún déficit en la percepción de emociones, como aquellas con Trastornos del Espectro Autista. Se implementarán servicios web que permitan analizar emocionalmente un texto, identificando las emociones básicas que contiene y su intensidad, y hacer más explicito el significado emocional para hacer más fácil su comprensión a las personas con algún tipo de discapacidad emocional.

1.2. Dificultades

A lo largo de la realización del trabajo nos hemos ido encontrando una serie de problemas que se explican a continuación.

1.3. Estructura

El capítulo 2 presenta los aspectos más importantes de la computación afectiva e introduce a los servicios web, la metodología Scrum y la integración continua; metodología y tecnología que se va a utilizar. El capítulo 3 describe

las herramientas utilizadas a lo largo del trabajo. El capítulo 4 explica cada uno de los servicios web desarrollados durante el trabajo.

Capítulo 2

Estado del arte

Resumen: En este capítulo se van a tratar los aspectos más importantes tanto de la computación afectiva como de las diferentes tecnologías y metodologías que se van a utilizar. En primer lugar, en la sección 2.1, se define la computación afectiva y sus posibles aplicaciones, también se explican los distintos diccionarios afectivos ya existentes que permiten la marcación emocional de textos. En la sección 2.2 se introduce la tecnología que se va a utilizar para implementar el trabajo, los servicios web. En la sección 2.3 se fijan los conceptos relacionados con la metodología Scrum, la cual hemos seguido durante todo el TFG. En la sección 2.4 se explican las bases de la integración continua aplicada al desarrollo de software y cómo se va a aplicar en este trabajo.

2.1. Emociones

Las emociones son reacciones afectivas que surgen súbitamente ante un estímulo, duran un corto espacio de tiempo y comprenden una serie de repercusiones psicocorporales (Francisco, 2008) Todo el mundo sabe lo que son las emociones hasta que llega el momento de definirlas. Podríamos definir las emociones como reacciones automáticas que nuestro cuerpo experimenta ante un determinado estímlo. Sin embargo, todas ellas luego derivan en sentimientos más prolongados en el tiempo.

Existen 5 emociones básicas (enfado, miedo, alegría, sopresa y tristeza), las cuales podemos representar gracias a gestos y movimientos.

2.2. Computación Afectiva

La computación afectiva (Picard, 1997) es el estudio y el desarrollo de sistemas y dispositivos capaces de percibir, medir e interpretar las emociones humanas.

Esta rama de la computación permite un avance notable en la inteligencia artificial, hasta tal punto que los ordenadores lleguen a adaptarse a los humanos, sus necesidades y estados de ánimo. Los seres humanos están rodeados de emociones, en cualquier ámbito de su vida, tanto de las suyas propias como las de las personas con las que se comunican. Tanta importancia tienen para nosotros que influyen no sólo en nuestra comunicación, sino también en nuestro aprendizaje y toma de decisiones. Por ello, resulta artificial y en ocasiones incluso frustrante intentar comunicarse con una máquina que no es capaz de expresar sentimientos. La computación afectiva pretende mejorar la interacción hombre-máquina haciéndola más natural y asequible.

El funcionamiento de este tipo de sistemas se basa en identificar el estado emocional del sujeto a través de diferentes fuentes (voz, expresiones, señales fisiológicas, palabras...) y procesar la información para clasificarla y aprender de ella. Clasificar la información de entrada puede resultar complicado ya que se suelen recibir varias señales diferentes a la vez, lo que hace necesario utilizar técnicas de priorización para determinar cuáles son las que más aportan a la hora de analizar y gestionar la información. Una vez identificada la emoción predominante, el sistema responde adecuandose a ella. La salida dependerá del tipo de sistema y las herramientas de las que dispone este para expresar su respuesta (colores, sonidos, emoticonos...) En casos más complejos aplicados a robótica o modelado, la salida producida es una simulación de la respuesta que produciría un ser humano ante los estímulos recibidos imitando su expresión corporal, voz o gestos faciales. Por ejemplo los robots Geminoid (IEEE Sprectrum, 08/04/2010).

La computación afectiva tiene multitud de aplicaciones, ya que como se ha mencionado antes, las emociones están presentes en todos los ámbitos de la vida de una persona. Por lo tanto, puede aplicarse a áreas muy diferentes entre sí como:

- **Seguridad**: Poder analizar las emociones como el estrés, el aburrimiento o la distracción pueden ser muy interesantes de analizar para tareas repetitivas como conducir, trabajar en una fábrica o ser controlador (Baldasarri, 2016).

- **Marketing**: Actualmente una de sus aplicaciones más explotadas y área que más se preocupa por su desarrollo. Poder evaluar la reacción emocional de alguien ante un anuncio o producto es una estrategia comercial que ya está siendo utilizada cada vez por más empresas (Baldasarri, 2016) (**?**).

- **Salud**: Actualmente, se emplea principalmente para la detección del estrés y así minimizar sus efectos y aprender a controlarlo. Es posible inferir el nivel de estrés de una persona midiendo sus señales fisiológicas (ritmo cardiaco, respiración...) y si este nivel es demasiado alto se

reaccionará en consecuencia según el tipo de sistema. Se puede aplicar de forma similar a las fobias (**?**).

- **Entretenimiento**: La industria de los videojuegos ha crecido mucho en los últimos años e introducir este tipo de tecnología permite a las compañías crear juegos más adaptables y cercanos al jugador, lo que atrae a más público y mejora la experiencia de juego (Ng et al., 2012).

- **Robótica**: El mayor problema de los robots diseñados para interactuar con humanos es la carencia de emociones. Algunos de ellos llegan a producir una sensación de incomodidad. Dotar a este tipo de robots de cierta "humanidad"no sólo haría más cómodo el tratar con ellos sino que podrían realizar tareas como el acompañamiento de personas mayores (Riek et al., 2010).

- **Accesibilidad**: Existen numerosas aplicaciones terapéuticas para ayudar a personas con problemas emocionales. Se emplean herramientas para ayudar a personas con Trastornos de Espectro Autista. El paradigma actual obliga a cualquier usuario a adaptarse a las máquinas sin tener en cuenta las dificultades particulares que pueda tener. Es complicado crear una máquina capaz de adaptarse a todas las circunstancias. El uso de computación afectiva permite facilitar la inversión del paradigma para que las máquinas utilicen las emociones del usuario para adaptarse a él (Baldasarri, 2016).

En este trabajo vamos a centrarnos en el último área, la accesibilidad. En particular, en facilitar a personas que padecen Trastornos del Espectro Autista (TEA) el entendimiento de las emociones de textos, su carga emocional. Trataremos de analizar un texto de entrada para identificar las emociones predominantes y en que medida se presentan, evitando las posibles ambigüedades que pueda haber. Las emociones con las que trataremos son las básicas: alegría, tristeza, miedo, sorpresa y enfado. Una vez identificadas las emociones se etiquetarán para hacerlas más explícitas mediante emoticonos.

2.2.1. Métodos para el marcado de texto emocional

Los métodos existentes para el marcado de texto emocional podrían clasificarse en cinco categorías básicas: keyword spotting, afinidad léxica, procesamiento estadístico del lenguaje natural, métodos basados en el conocimiento del mundo real y métodos manuales. (Francisco, 2008)

- **Keyword spotting**: El marcado del texto se basa en la detección de palabras clave. Es decir, consiste en detectar la aparición de palabras emocionales como happy, sad... Las principales desventajas de este método son: que causa problemas cuando aparece una negación en

la frase y la dependencia del método en aspectos superficiales cuando en la práctica existen muchas frases cuyo contenido emocional no se encuentra en los adjetivos.

- **Afinidad léxica**: Este método no solo detecta palabras obviamente emocionales como los adjetivos, sino que asigna al resto de palabras una afinidad con las distintas emociones. Este método tiene dos problemas fundamentales: emplea tan solo las palabras sin tener en cuenta el contexto en el que se encuentran por lo que puede fallar con la aparición de las negaciones y además este tipo de métodos suelen entrenarse con un corpus lo que dificulta el desarrollo de un modelo reutilizable e independiente del dominio.

- **Procesamiento estadístico**: Consiste en alimentar a un algoritmo de aprendizaje con un varios textos marcados emocionalmente. Este método ha sido empleado en el proyecto de Webmind (Goertzel et al., 2000) entre otros.

- **Métodos manuales**: Estos métodos implican el modelado de distintos estado emocionales en términos de modelos afectivos basados en teorías psicológicas sobre las necesidades, los deseos y las metas de los seres humanos. Un ejemplo es el DAYDREAMER (Dyer, 1987)

- **Métodos basados en el conocimiento del mundo real**: Este método no solo mira los aspectos superficiales del texto, sino que va más allá, evalúa la calidad afectiva de la semántica subyacente que contiene el texto. Un ejemplo es el marcador llevado a cabo por Liu, Lieberman y Selker (Liu et al., 2002)

Las técnicas basadas en el procesamiento estadístico del lenguaje natural solo funcionan con un texto de entrada lo suficientemente extenso, los métodos manuales precisan un amplio análisis y entendimiento de los textos y hacen muy difícil la generalización y los métodos basados en el conocimiento del "mundo real"necesitan un amplio conocimiento denominado "mundo real"(Francisco, Gervás, 2006).

2.2.2. Diccionarios Afectivos

Un diccionario afectivo es un diccionario en el que las palabras se encuentran marcadas con etiquetas afectivas. Algunos ejemplos de etiquetas afectivas serían: alegría, tristeza, sorpresa, miedo...

En las siguientes secciones presentamos los diccionarios afectivos mas representativos.

2.2.2.1. Lasswell value dictionary y general inquirer

Fue a comienzos de los años 60 cuando Stone y Lasswell, ambos investigadores, comenzaron a construir diccionarios en lo que las palabras estaban marcadas con etiquetas afectivas. El diccionario Lasswell clasifica sus palabras de manera binaria y en ocho categorías básicas (riqueza, poder, respeto, rectitud, habilidad, iluminación, afecto y bienestar). Además de estas, el esquema de este diccionario distingue entre objetivos sustantivos y los elementos y atributos del proceso de distribución de valores, es decir, evaluación personal y asignación social. La clasificación social del contenido mantiene estas distinciones. Las preocupaciones con valores particulares se clasifican en objetivos sustantivos particulares. El esquema de clasificación también distingue varios tipos de transacciones de valor.

2.2.2.2. Diccionario de Hatzivassiloglou y Mckeown

Este diccionario fue creado en 1997, y tarta de encontrar etiquetas como positive o negative de manera automatizada a través del análisis de un corpus. Hatzuvassiloglou y McKeown tomaron una serie de adjetivos que aparecían de manera frecuente y decidieron darles una orientación, empleando un análisis estadístico en el que cuando dos adjetivos aparecían juntos, siguiendo el patrón adjetivo1 y adjetivo 2, devolvía una orientación positiva o negativa. Así, pudieron obtener un diccionario de adjetivos clasificados en positivos y negativos.

2.2.2.3. Wordnet affect

Se trata de una extensión de WordNet Domains, que incluye un subconjunto de synsets adecuados para representar conceptos afectivos correlacionados con palabras afectivas. De manera similar a nuestro método para etiquetas de dominio, asignamos a una serie de sintonias de WordNet una o mas etiquetas afectivas. En particular, los recursos afectivos que representan el estado emocional son individualizados por sintonizadores marcados por la emocion de la etiqueta A. Tambien existen otras etiquetas para aquellos conceptos que representan estados de animo, situaciones que provocan emociones o respuestas emocionales. El recurso se amplió con un conjutno de etiquetas A adicionales (llamadas categorias emocionales), organizadas jerarquicamente, con el fin de especializar sysnsets con una emocion de etiqueta. La estructura jerarquica de las nuevas etiquetas a se modelo en la relaciones de hiperonimos de wornet. En una segunda etapa, se definen cuatro etiquetas a-adicionales: positivo, negativo, ambiguo, neutral. Otra propiedad importante para el lexico afectico en lo que se se refiere principalmente a la interpretacion adejtivas es la dimension estacionaria/causativa.

2.2.2.4. Whissell's dictionary of affects in language (DAL)

Este diccionario es un recurso diseñado con el fin de medir el significado emocional de las palabras y textos. El objetivo del diccionario es comparar las palabras con una lista de 8.742 palabras que han sido marcadas por distints personas en terminos de las dimensiones activacion y evaluacion e imagenes.

2.2.2.5. Diccionario de Turney y Littman

Turney y Littman encontraron una manera mas eficiente para decidir cuando una palabra se podía considerar positiva o negativa. Dado un conjunto de palabras, que previamente ellos conocían como positivas o negativas, probaron como de frecuentemente podía aparecer una palabra en un contexto con un conjunto de palabras positivas o negaticas. Empleando la informacion de estudios cercanos y estadísticas clasificaron como positivas todas aquellas palabras que aparecían de forma más significativa con un conjunto de palabras positivas y como negativas aquellas palabras que aparecían con un conjunto de palabras negativas.

2.2.2.6. Clairvonyance affect lexicon

Este diccionario fue desarrollado a mano. Las entradas de este diccionario se dividen en cinco campos: el lema de la palabra, part-of-speech, clase afectiva, centralidad e intensidad.

2.2.2.7. ANEW

Se trata de un modelo espacial de varias palabras con carga emocional que contribuye a clasificar textos arbitrarios. Para obtener este diccionario, se pidió a una serie de personas que marcasen un conjunto de palabras con los valores para las dimensiones de una emoción (Bradley y Lang, 1999). Este diccionario será base del diccionario afectivo que vamos a emplear para marcar palabras en este trabajo. Concretamente, utilizaremos el diccionario ANEW traducido al español (Redondo et al., 2007). En la traducción de ANEW al castellano participaron 720 estudiantes de Psicología de entre 18 y 25 años, 560 mujeres y 160 hombres. Estos participantes evaluaron 1.034 palabras que contiene ANEW. Cada palabra debía ser marcada con las dimensiones emocionales: evaluación, activación y control, en una escala de 9 puntos. Cada entrada en el diccionario contiene un número que identifica a la palabra, de manera que, esta numeración coincide con el número que dicha palabra tiene en el ANEW; la palabra ingles (E-word), la palabra original en la base de datos ANEW; la palabra española (S-word) y las evaluaciones afectivas, los valores medios y la desviación estándar para cada dimensión emocional.

2.3. Servicios Web

Existen múltiples definiciones de servicios web por lo que resulta muy complejo dar una definición de este término. Esta complejidad nos permite hacernos una idea de la cantidad de servicios e implicaciones asociadas que se agrupan bajo este concepto.

En la actualidad, la definición más general de Servicio Web es la que lo define como un conjunto de aplicaciones o tecnologías capaces de interoperar en la Web. Estas aplicaciones o tecnologías intercambian datos con el fin de ofrecer unos servicios. Los proveedores ofrecen sus servicios como procedimientos remotos y los usuarios solicitan un servicio llamando a estos procedimientos a través de la Web (W3C).

2.3.1. Características de los Servicios Web

Un servicio que debe poder ser accesible a través de la web. Para ello debe utilizar procotolos de transporte estándares como HTTP, y condificar los mensajes en un lenguaje estándar que pueda conocer cualquier cliente que quiera utilizar el servicio.

Un servicio que debe contener una descripción de sí mismo. De esta forma, una aplicación podrá saber cuál es la función de un determinado servicio web.

Un servicio web debe poder ser localizado. Deberemos tener algún mecanismo que nos permita encontrar un servicio web que realice una determinada función. De esta forma tendremos la posibilidad de que una aplicación localice el servicio que necesite de forma automática, sin tener que conocerlo previamente el usuario. (Dto. de Ciencia de la Computación e Inteligencia Artificial)

2.3.2. Tipos de Servicios Web

A nivel técnico, los servicios pueden implementarse de varias formas. En este sentido, podemos distinguir dos tipos de servicios web (Dto. de Ciencia de la Computación e Inteligencia Artificial):

- SOAP: utilizan mensajes XML para intercomunicarse que siguen el estándar SOAP (Simple Object Access Protocol), un lenguaje XML que define la arquitectura y formato de los mensajes. Dichos sistemas normalmente contienen una descripción legible por la máquina de la descripción de las operaciones ofrecidas por el servicio, escrita en WSDL (lenguaje basado en XML para definir las interfaces sintácticamente).

- Servicios web RESTful: utilizan estándares conocidos como HTTP, SML, URI, MIME, y tienen una infraestructura "ligera"que permite que los servicios se construyan utilizando herramientas de forma mínima.

2.3.3. Arquitectura de los Servicios Web

Los servicios web se componen de varias capas entre las que destacan: servicios de transporte (constituidos por los protocolos de nivel más bajo, que codifican la información independientemente de su formato y que pueden ser comunes a otros servicios), servicios de mensajería, de descripción y de descubrimiento. En las siguiente subsecciones explicaremos más en detalle cada una de estas capas (Lázaro), (Garrido y Zunino, 2006)

- Servicios de transporte: Capa que se encarga de transportar los mensajes entre el servidor y el cliente. Normalmente se utiliza el protocolo HTTP para este transporte.
- Servicios de mensajería: Capa encargada de la codificación de los mensajes en formato XML estándar para que pueda ser interpretado en cualquiera de los nodos de la red. Puede implementar protocolos como XML-RPC o SOAP.
- Servicios de descripción: Capa encargada de definir la interfaz pública de un determinado servicio. Esta definición se realiza mediante WSDL (Web Service Description Language), tipo de documento XML que describe lo que hace un servicio web, dónde se encuentra y la forma de ser invocado.
- Servicios de descubrimiento: Capa encargada del registro centralizado de servicios, permitiendo que estos sean anunciados y localizados. Para ello se utiliza el protocolo UDDI, el cual define la especificación para construir un directorio distribuido de servicios web, donde los datos se almacenan en XML. Este registro también almacena información sobre las organizaciones que los proporcionan, la categoría en la que se encuentran, y sus instrucciones de uso.

2.3.4. Ventajas e inconvenientes

Las principales ventajas de los servicios web son:

- Aportan interoperabilidad entre aplicaciones de software independientemente de sus propiedades o de las plataformas sobre las que se instalan.
- Fomentan los estándares y protocolos basados en texto, de tal manera que facilitad su comprensión y accesibilidad.
- Permiten la combinación e integración de distintos servicios y software independientemente de su ubicación.
- Al apoyarse en HTTP, lo servicios Web se pueden aprovechar de los de seguridad firewall sin necesidad de cambiar las reglas de filtrado.

- Disminuyen el tiempo de desarrollo de las aplicaciones: Pues gracias a la filosofía de orientación a objetos utilizada, el desarrollo se convierte más bien en una labor de composición.

Las dos principales desventajas de los servicios web son:

- Bajo rendimiento si se compara con otros modelos de computación distribuida.
- Al apoyarse en HTTP, pueden esquivar medidas de seguridad basadas en firewall cuyas reglas tratan de bloquear la comunicación entre programas.

2.4. Scrum

Scrum es una metodología ágil para gestionar el desarrollo de software. Fue definido por *Ikujiro Nonaka* e *Hirotaka Takeuchi* a principios de los 80 (ScrumManager, 2016).

Scrum divide el trabajo en diferentes unidades llamadas *sprints*, que tienen una duración preestablecida de entre dos y cuatro semanas obteniendo siempre al final una versión del software con nuevas prestaciones listas para ser usadas. En cada *sprint* se ajusta la funcionalidad y se añaden nuevas prestaciones priorizando aquellas que aporten más valor al producto (ScrumManager, 2016), (Softeng).

2.4.1. Roles

Esta metodología, hace mucho énfasis en el «equipo de trabajo». Este equipo está formado por diferentes roles:

- **Product Owner**: Representa al cliente. Se encarga de definir los objetivos y de dar valor al producto.
- **Scrum Master**: Encargado de asegurar que se cumplen las buenas prácticas y valores descritos en el modelo scrum.
- **Scrum Team**: Equipo encargado de desarrollar y entregar el producto. Es autogestionado y multidisciplinar.

2.4.1.1. Artefactos

Los artefactos definidos en scrum son:

- **Product backlog**: Lista realizada por el Product Owner que contiene los requisitos del producto, ordenados por prioridad. A lo largo del

desarrollo crece y evoluciona. En scrum los requisitos se definen mediante historias de usuarios. Una historia de usuario es la descripción de una funcionalidad que debe incorporar un sistema de software, y cuya implementación aporta valor al cliente.

- **Sprint backlog**: Descomposición en tareas de las historias de usuario del product backlog seleccionadas para que el equipo las realice durante el sprint.

2.4.1.2. Eventos

Por último, scrum también define una serie de eventos:

- **Sprint**: Cada una de las iteraciones del desarrollo.
- **Sprint Planning**: Se trata de la reunión de planificación del sprint. Este evento consta de dos partes. En la primera parte el Product Owner presenta lo que quiere que se haga en el sprint y resuelve dudas acerca de la historia de usuario y explica la importancia de dicha tarea para que el grupo de trabajo tenga clara la prioridad. En la segunda parte el equipo estima el timpo que llevará a desarrollar cada una de las diferentes historias de usuario propuestas en el product owner y deciden cuantas van a implementar en el sprint, para posteriormente crear el Sprint Backlog.
- **Daily Scrum**: Reunión diaria como máximo de quince minutos, de pie, donde cada componente del equipo informa sobre cómo va en sus tareas, lo que hizo el día anterior, lo que hará ese día y los problemas que ha encontrado o los que cree que se va a encontrar.
- **Sprint Review**: Reunión que se realiza al concluir el sprint centrándose en el producto. Se presenta el producto creado en el sprint al Product Owner y este lo analiza y da su feedback.
- **Sprint Retrospective**: Reunión donde se habla de cómo ha funcionado el equipo en el sprint y qué cosas se pueden mejorar para el siguiente a nivel de proceso o metodología.

En la Figura 2.1 podemos ver un diagrama del flujo Scrum. Sería el siguiente: el Product Owner crea el Product Backlog con los requisitos y características por orden de prioridades. A continuación, en un Sprint Planning, se presenta el Product Backlog y se decide qué actividades van a desarrollar en el sprint. En esta reunión se elabora el Sprint Backlog con todas las historias de usuario que van a realizar divididas en tareas. A continuación, se comienza el sprint con el tiempo establecido, cada componente del grupo se asigna una tarea y en cuanto acabe seguirá con la siguiente actividad que

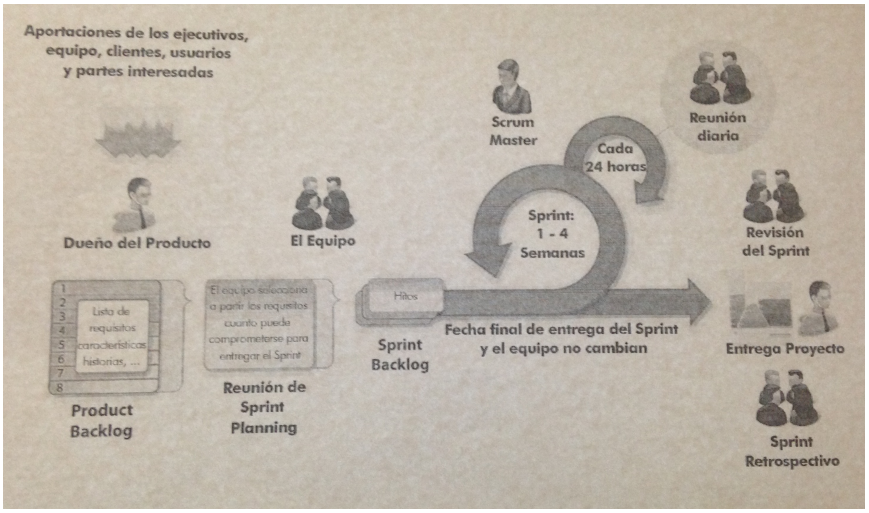

Figura 2.1: Flujo Scrum

no esté hecha. Es muy importante el orden de las tareas, ya que el cliente ha ordenado en el Product Backlog las actividades por prioridad y el equipo desarrollará estas siguiendo dicho parámetro. Cada día, el equipo se reunirá en una Daily Scrum. Al final del sprint se realiza la Sprint Review donde se entrega el incremento del producto al cliente. Por último, todo el equipo se vuelve a reunir para hacer el Sprint Retrospective.

2.5. Integracion Continua

La integración continua (Fowler, 2006), se basa en que los desarrolladores combinen todos los cambios que realicen en el código en un repositorio común de forma periódica (cada pocas horas, al menos una vez al día), de tal forma que una vez subidos estos cambios, se ejecutan una serie de pruebas automáticas sobre estos con el fin de validarlos y detectar errores de integración tan pronto como sea posible.

En general, las metodologías ágiles y en particular la metodología Scrum enfoca su objetivo en tener versiones del producto al finalizar cada etapa; es por esto por lo que la integración es vital en proyectos que empleen estas metodologías. Emplenado la integración continua, aseguras de encontrar los errores a tiempo para que el producto esté listo para la entrega al cliente o para ponerlo en producción si este así lo desea.

Las principales ventajas de la integración continua:

- **Detección de errores:** Cada vez que el código cambia se compila y se somete a pruebas de forma inmediata para garantizar que no hay

errores. Este proceso aumenta la calidad del software y minimiza los riesgos del proceso ya que se tiene control sobre las versiones en todo momento.

- **Visibilidad del proceso:** Todos los pasos que se realizan en el desarrollo son visibles a todo el equipo, que tiene una estrategia común muy bien definida.
- **Mejora del equipo:** Los desarrolladores no solo tienen una visión muy clara y estructurada del proceso sino que también aprenden a realizar todo tipo de pruebas, lo que les hace mejorar a nivel profesional.

Lo primero para poder utilizar integración continua es tener definido un pipeline, es decir, un conjunto de fases por las que tiene que pasar el software y que están automatizadas. Se establecen criterios para que el código pase de una fase a otra y estrategias para gestionar errores que puedan surgir en las diferentes fases (control de versiones). Es importante tener bien definidas las pruebas que se van a realizar sobre cada fase y que estas puedan garantizar la máxima corrección posible sin tardar mucho, ya que se necesita un feedback rápido para poder seguir avanzando en el proceso. Cada fase es un grupo de pruebas y cada subida de código es un pipeline distinto que avanza de forma independiente por las fases. Por lo tanto se sabe en todo momento en qué punto se encuentra una versión específica. Esto permite tener una visión general de todo el proceso facilitando notablemente la detección de errores en fases y pipelines concretos.

Para el correcto funcionamiento de esta práctica tiene que haber pequeñas integraciones de forma frecuente, una vez al día por ejemplo. Cuantos menos cambios haya más fácil es la integración en el producto general y solucionar los posibles errores que esta pueda generar. Cabe destacar que aunque una parte de código funcione de forma independiente no implica que vaya a funcionar al integrarlo en un programa más grande, por ello cuanto más frecuentes sean las integraciones mejor.

En nuestro caso la integración continua se aplicará de la siguiente manera:

- **Repositorio:** Se utilizará un repositorio común de *GitHub* en el que se subirán todos los cambios realizados en el código. Al ser un equipo de desarrollo pequeño y estar utilizando la metodología Scrum en principio todos los miembros del equipo estarán trabajando en la misma rama. Esto puede llegar a bloquear el proceso mientras una versión acabe de pasar las pruebas.
- **Jenkins:** Tendremos Jenkins corriendo en un servidor para realizar las pruebas automáticas. Cada vez que se detecten cambios en el repositorio este avisará al servidor que procederá a hacer las pruebas de compilación y funcionamiento sobre el nuevo código.

- **Pruebas manuales:** Algún miembro del equipo realizará las tareas de «tester». Después de las pruebas automáticas se realizará otra serie de pruebas planificadas de antemano para probar mejor la funcionalidad del código. El «tester» no será siempre la misma persona.
- **Pruebas automáticas:** Usaremos un software especial con el fin de controlar la ejecución de pruebas y la comparación entre los resultados obtenidos y los resultados esperados. El uso de estas pruebas, nos permite incluir pruebas muy repetitivas y necesarias, dado que habrá pruebas que realizarlas de manera manual nos podrá ser muy costoso.

Capítulo 3

Herramientas

Resumen: En este capítulo se profundizará en las herramientas que utilizaremos a lo largo del trabajo. En la sección 3.1 se presenta el diccionario que vamos a utilizar para el marcado emocional. En la sección 3.2 se introduce el framework que vamos a utilizar para el desarrollo de los servicios web, Django. En la sección 3.3 se explica cómo vamos a utilizar Trello para seguir la metodología Scrum. En la sección 3.4 se expone la forma de realizar las pruebas utilizando Jenkins y Doctest. En la sección 3.5 se presenta SpaCy, el framework que se utilizará para poder etiquetar las palabras que forman una frase para poder realizar el análisis emocional sobre ella.

3.1. Diccionario

Este diccionario, es una adaptación en el que se han elegido las 5 emociones básicas (tristeza, miedo, alegría, enfado y sorpresa) y la neutralidad para no expresar ninguna emoción. Para crearlo participaron 10 personas; cada una de ellas marcó cada una de las 1034 palabras con una emoción básica o la neutralidad. Cada palabra puede representar varias emociones por lo que se ha creado un número de confianza c, entre e 0 y el 1, que determina con qué certeza la palabra corresponde a esa emoción; es decir indica el tanto por uno de evaluadores que asignaron dicha emoción a la palabra. La suma de estos números debe sumar 1.

En la Tabla 3.1, podemos ver los valores obtenidos para las palabras «abandonado», «aborto» y «abeja». En este ejemplo se puede ver que tristeza fue la emoción asignada a la palabra «abandonado» por todos los evaluadores, mientras que con «abejas» y «aborto» no hubo acuerdo. En el caso de «aborto», un 67 % de los anotadores le asignaron tristeza, mientras que un 17 % consideró que la palabra no tenía emoción asociada y un 17 % le asignaron miedo.

S-Word	Tristeza	Miedo	Alegría	Enfado	Sorpresa	Neutral
abandonado	1,00	0,00	0,00	0,00	0,00	0,00
abejas	0,00	0,50	0,17	0,00	0,00	0,33
aborto	0,67	0,17	0,00	0,00	0,00	0,17

Figura 3.1: Fragmento de la adaptación del diccionario ANEW traducido.

3.2. Django

Toda la implementación del trabajo se hará en Django, un framework para aplicaciones web gratuito y open source escrito en Python.

Django incluye un servidor web que almacena la base de datos que contendrá las palabras del diccionario y permitirá hacer las consultas necesarias. La base de datos estará formada por instancias del modelo creado, en nuestro caso cada palabra aparecerá modelada mediante su significante y los grados de certeza que tiene de cada emoción. Para realizar las diferentes consultas sobre las palabras disponibles se crearán una serie de clases que implementan los diferentes métodos de un servicio web REST típico: **GET, POST, DELETE**. Esencialmente **GET**, ya que es el que devuelve información sobre la palabra o alguno de sus campos. Cada una de las diferentes clases o «vistas» nos aportarán una forma diferente de acceder a la información: acceso a toda la lista, a una palabra concreta, a un campo de una palabra concreta, etc... Los resultados serán devueltos en formato JSON.

3.3. Trello

Trello es una aplicación web que permite organizar proyectos y actividades. Para representar las tareas y las historias de usuario se usan tarjetas virtuales. En la Figura 3.2 podemos ver el estado inicial del proyecto. Se observa el **Product Backlog**, del que product owner saca la cantidad de las historias de usuario que quiere que se realicen durante el sprint y estas pasan a la lista **To Do**. En la Figura 3.3 se puede ver un ejemplo más avanzado en el que se puede ver como las historias de usuario han sido divididas en tareas para formar el **Sprint Backlog**, del que van saliendo en orden para estar **En Progreso** y, una vez acabadas, **Done**.

3.4. Doctest y Jenkins

Como ya se comentó en el capítulo 2.4 utilizaremos Jenkins como parte de la integración continua del proyecto. Esto nos permitirá asegurarnos de que la unificación es correcta y realizar las pruebas automáticas. Esto último

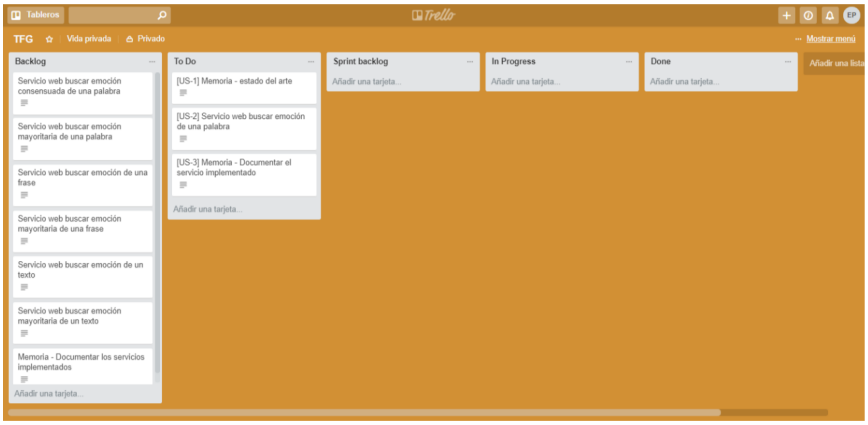

Figura 3.2: Sprint inicial que ilustra la estructura.

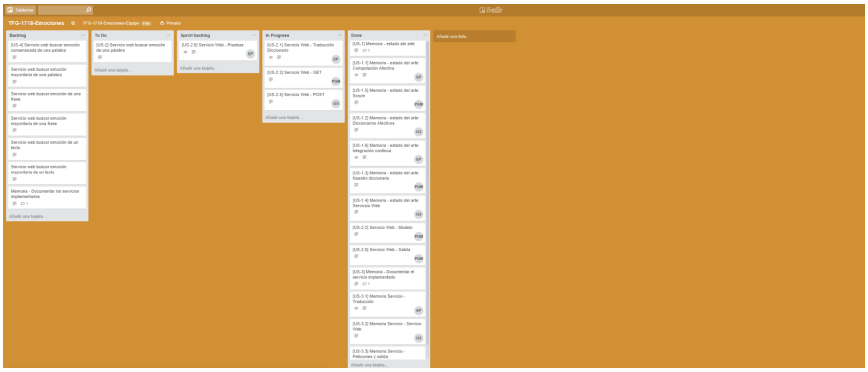

Figura 3.3: Final del sprint inicial.

se llevará a cabo mediante una orden shell que Jenkins ejecutará cada vez que se detecte un cambio en el repositorio. La orden únicamente se encarga de ejecutar el script de pruebas que contendrá las llamadas a los diferentes programas de pruebas que se desarrollen.

Los programas de pruebas utilizarán Doctest para hacer las pruebas. Doctest es un módulo incluido en la librería estandar de Python. Su funcionamiento se basa en definir la función que se quiera probar y, dentro de un comentario al inicio de esta, poner una serie de llamadas y el resultado que se espera obtener de ellas. Tiene una función «testmod» que realiza las pruebas y devuelve el número de fallos y el resultado de todas las pruebas. Si el número de fallos es mayor que cero provocamos una excepción que Jenkins detectará para notificar a todo el equipo que hay algún fallo. Los resultados de las pruebas se muestran por consola al acabar y Jenkins los guardará para ayudar a encontrar el problema.

3.5. SpaCy

El objetivo final es llegar a interpretar textos enteros, no sólo palabras. Para ello se necesita una herramienta que nos facilite trabajar con frases, etiquetando cada una de las palabras que las forman. **SpaCy** es una librería open source escrita en Python y dedicada al procesamiento de lenguajes naturales. Soporta el español y nos permite etiquetar las palabras para poder buscar sólo aquellas que puedan tener carga emocional. SpaCy recibirá el texto plano, en este caso una serie de frases, y devolverá un objeto de tipo «Doc», propio de la librería, que contendrá la frase con una serie de anotaciones sobre cada una de las palabras que la forman (lema, etiqueta, dependencias sintácticas, forma...).

Capítulo 4

Analis del Contenido Afectivo de un Texto

Resumen: En este capítulo se explicarán los servicios web que se han desarrollado con el fin de analizar el contenido afectivo de un texto. Primero, en la sección 4.1, se presentarán los servicios orientados a identificar la emoción predominante en una palabra concreta, tanto la consensuada como la mayoritaria. Después, en la sección 4.2, se explica como el análisis de la palabra se aplica a determinar la emoción de una frase entera para finalmente, en la sección 4.3, aplicar todo al anális de todo un texto.

4.1. Analis afectivo de una palabra

Como ya se comentó en la sección 3.1, para el desarrollo de este trabajo se utilizará un diccionario afectivo en español. Este diccionario, inicialmente en formato Excel, ha sido convertido a un fichero CSV para que resulte más sencillo obtener los datos.

Los datos estarán almacenados en el servidor Django.

- **Modelo:** Los campos que tiene el modelo para cada palabra, como ya se comentó en el capítulo anterior, son la propia palabra, el porcentaje de una determinada emoción que expresan (para las cinco emociones que consideramos en nuestro diccionario y la neutralidad). Cada una de las palabras que tenemos recogidas se encapsulan en este modelo para ser almacenadas en el servidor.
- **Consultas:** Se realizarán mediante declaración de clases «vista» según las funcionalidades que requiera cada uno de los servicios web que vamos a desarrollar.

```
GET /emocion/calor/percentages/

HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

"SADNESS:0% || FEAR:0% || JOY:0% || MADNESS:17% || SORPRISE:0% || NEUTRAL:83% "
```

Figura 4.1: Respuesta al buscar los porcentajes de la palabra «calor».

```
GET /emocion/te/percentages/

HTTP 404 Not Found
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "detail": "Not found."
}
```

Figura 4.2: Respuesta al buscar los porcentajes de una palabra que no está en el diccionario.

Una vez que se tiene el servidor y todo el código necesario para el funcionamiento de los servicios web se procede a subir las palabras que recoge nuestro diccionario. Para ello se ha desarrollado un programa en *Python* cuya función es leer el fichero CSV, interpretar cada una de sus líneas y subir la información que obtiene al servidor. Una vez que todas las palabras estén subidas ya se pueden realizar las consultas necesarias.

4.1.1. Servicio web porcentajes

Esta vista nos devuelve la información respectiva a los porcentajes de cada emoción que posee, mediante una petición **GET** al servidor, y los muestra al usuario.

Como entrada tendríamos la palabra de la cual queremos conocer sus emociones; existirán dos posibilidades, bien que la palabra esté en nuestro diccionario, por lo que la salida sería el porcentaje de cada emoción (Figura 4.1), o bien que la palabra no exista en nuestro diccionario, que devolverá un mensaje de «NOT FOUND» (Figura 4.2).

```
GET /emocion/bueno/agreed/

HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

"AGREED: JOY"
```

Figura 4.3: Respuesta al encontrar la emoción consensuada.

```
GET /emocion/calor/agreed/

HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

"NO AGREED EMOTION"
```

Figura 4.4: Respuesta si la palabra no tiene emoción consensuada.

4.1.2. Servicio web emoción consensuada

Esta vista nos devuelve, mediante una petición **GET** al servidor, si tiene o no emoción consensuada y en caso de tenerla cuál. Hablamos de emoción consensuada cuando la el grado de certeza que se tiene de una de sus emociones es 1.

Como entrada tendríamos la palabra de la cual queremos conocer su emoción consensuada, mientras que la salida será la emoción consensuada en caso de tenerla (Figura 4.3) o un mensaje de que no la tiene (Figura 4.4).

4.1.3. Servicio web emoción mayoritaria

Esta vista nos devuelve, mediante una petición **GET** al servidor, su emoción mayoritaria. Hablamos de emoción mayoritaria cuando el porcentaje de confianza de la una emoción es mayor al del resto de emociones.

Como entrada tendríamos la palabra de la cual queremos conocer su emoción mayoritaria, mientras que la salida será la emoción mayoritaria (Figura 4.5), o mayoritarias (en caso de que haya dos con el mismo porcentaje) (Figura 4.6).

Figura 4.5: Respuesta al encontrar la emoción mayoritaria.

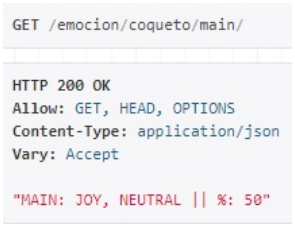

Figura 4.6: Respuesta si hay dos emociones mayoritarias.

Apéndice A

Así se hizo...

...

...

Resumen: ...

A.1. Introducción

...

Bibliografía

Y así, del mucho leer y del poco dormir,
se le secó el celebro de manera que vino
a perder el juicio.

Miguel de Cervantes Saavedra

Baldasarri, S. Computación afectiva: tecnología y emociones ara mejorar la expriencia de usuario. *Revista Institucional de la Facultad de Informática Universidad Nacional de La Plata (Argentina)*, 2016.

Bradley, M. M. y Lang, P. J. Affective norms for english words (anew): Instruction manual and affective ratings. 1999. Disponible en `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.306.3881&rep=rep1&type=pdf` (último acceso, Marzo, 2018).

Fowler, M. Continuous integration. 2006. Disponible en `https://www.martinfowler.com/articles/continuousIntegration.html` (último acceso, Marzo, 2018).

Francisco, V. *Identificación Automática del Contenido Afectivo de un Texto y su Papel en la Presentación de Información*. Phd, Universidad Complutense de Madrid, 2008.

Garrido, H. y Zunino, C. 2006. Disponible en `http://www-2.dc.uba.ar/materias/seginf/material/tp_1c2006/Seguridad_WebServices.pdf` (último acceso, Marzo, 2018).

Goertzel, B., Silverman, K., Hartley, C., Bugaj, S. y Ross, M. *The Baby Webmind Project*. 2000.

Dto. de Ciencia de la Computación e Inteligencia Artificial, U. d. A. Servicios web. ???? Disponible en `http://www.jtech.ua.es/j2ee/publico/servc-web-2012-13/sesion01-apuntes.html` (último acceso, Marzo, 2018).

Lázaro, D. Introducción a los web services. ???? Disponible en `https://diego.com.es/introduccion-a-los-web-services` (último acceso, Marzo, 2018).

LIU, H., LIEBERMAN, H. y SELKER, T. Adaptive linking between text and photos using common sense reasoning. *Proceedings of the 2n International Conference on Adaptive Hypermedia and Adaptive Web Based Systems*, 2002.

NG, Y., KHONG, C. y THWAITES, H. A review of affective design towards video games. *Procedia-Social and Behavioral Sciences*, vol. 51, 2012.

PICARD, R. W. *Affective Computing*. 1997.

REDONDO, J., FRAGA, I., PADRÓN, I. y COMESAÑA, M. The spanish adaptation of anew (affective norms for english words. *Behavior Research Methods*, 2007. Disponible en `https://link.springer.com/article/10.3758/BF03193031` (último acceso, Marzo, 2018).

RIEK, L. D., PAUL, P. C. y ROBINSON, P. When my robot similes at me: Enabling human-robot rapport via real-time head gesture mimicry. *Journal on Multimodal User Interfaces*, 2010.

SCRUMMANAGER. *Scrum Manager v. 2.6*. 2016. Disponible en `http://scrummanager.net/files/scrum_manager.pdf` (último acceso, Marzo, 2018).

SOFTENG. ???? Disponible en `https://www.softeng.es/es-es/empresa/metodologias-de-trabajo/metodologia-scrum/proceso-roles-de-scrum.html` (último acceso, Marzo, 2018).

W3C. Servicios web. ???? Disponible en `https://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb` (último acceso, Marzo, 2018).

–¿Qué te parece desto, Sancho? – Dijo Don Quijote –
Bien podrán los encantadores quitarme la ventura,
pero el esfuerzo y el ánimo, será imposible.

Segunda parte del Ingenioso Caballero
Don Quijote de la Mancha
Miguel de Cervantes

–Buena está – dijo Sancho –; fírmela vuestra merced.
–No es menester firmarla – dijo Don Quijote–,
sino solamente poner mi rúbrica.

Primera parte del Ingenioso Caballero
Don Quijote de la Mancha
Miguel de Cervantes