GSIBot: Una Plataforma para el Desarrollo de Servicio de Bots

Miguel Coronado, Alejandro Marqués, Carlos A. Iglesias.
Depto Ingeniería de Sistemas Telemáticos
E.T.S.I.Telecomunicación Universidad Politécnica de Madrid
Ciudad Universitaria s/n 28040 Madrid.
miguelc@gsi.dit.upm.es, alejandrom@gsi.dit.upm.es, cif@gsi.dit.upm.es.

***Resumen*- El artículo presenta la plataforma GSI Bot para el desarrollo de servicios de bots. El objetivo de la plataforma es facilitar la asistencia en lenguaje natural a los usuarios que están usando un servicio, facilitando su integración multicanal (mensajería instantánea, Web, teléfono móvil) y el mantenimiento del servicio, con herramientas para la gestión, manejo y edición de la base de conocimiento de los bots. El sistema ha sido aplicado para un servicio de atención al cliente de una operadora, y está siendo aplicado a la gestión de dudas en e-learning.**

***Palabras Clave*-AIML, Bot, J2EE, Agente Inteligente.**

I. INTRODUCCIÓN

Prácticamente desde el comienzo de la andadura de los ordenadores, los programadores han tenido la ambición de crear programas que fueran capaces de mantener una conversación con un interlocutor humano. Posiblemente comenzaron como un juego o entretenimiento para el programador, pero hoy en día están muy extendidos y cada vez más perfeccionados. Los primeros bot conversacionales o chatterbots, como son conocidos, datan de la década de los 60, era necesario un ordenador dedicado a su funcionamiento y sólo una persona podía utilizarlos a la vez. En la actualidad, con el gran desarrollo de la Web es imposible imaginarlos desligados de Internet y son capaces de atender multitud de conversaciones simultáneamente. Ya no sólo aparecen como aplicaciones de demostración en la página del autor sino que se encuentran integrados, por ejemplo, en canales de IRC, como agentes en sistemas de mensajería instantánea, agentes expertos en FAQ, servicios de atención al cliente, sistemas domóticos...[1] Por supuesto forman parte de numerosas aplicaciones comerciales y su presencia es patente en áreas tan diversas como la enseñanza, el ocio o el marketing [2].

Sin embargo, aún queda mucho camino por recorrer: los bots conversacionales están lejos de ser totalmente inteligentes y sólo son capaces de hablar de aquello que se les ha enseñado, de lo que tienen conocimientos. Este comportamiento es semejante al de los seres humanos, con la diferencia de que la capacidad de aprendizaje del bot es limitada y que es totalmente pasiva. Enseñar a un bot conversacional (programar un bot conversacional) es un proceso largo, repetitivo, cuya complejidad aumenta conforme el conocimiento del bot se hace más basto y que requiere personal cualificado.

Este trabajo investiga el desarrollo de algoritmos y herramientas para facilitar el mantenimiento y creación de bases de conocimiento por parte de los usuarios finales. En particular, se ha trabajado en la definición de asistentes que permitan, mediante heurísticos, detectar conflictos cuando se amplía la base de conocimiento.

AIML [3], acrónimo de *Artificial Intelligence Markup Language* es la tecnología utilizada en el proyecto para programar las bases de conocimiento. AIML es un dialecto de XML específicamente diseñado para crear bots conversacionales. AIML es un lenguaje fácil de aprender que permite ampliar el conocimiento de un bot existente o crear uno desde cero en muy poco tiempo. Un conjunto de ficheros AIML forman, por tanto, la base de conocimiento de un bot conversacional. La especificación completa de AIML engloba un amplio conjunto de etiquetas [4]. La siguiente figura muestra un ejemplo de una regla AIML sencilla:

```
<category>
  <pattern>QUE TAL</pattern>
  <template>
    <think>
      <set name=”topic”>SALUDO</set>
    </think>
    Bien, ¿Y usted?
  </template>
<category>
```

Fig. 1. Ejemplo de código AIML.

AIML no necesita ser compilado sino que es interpretado por una plataforma para bots AIML. En el desarrollo del proyecto se ha utilizado ProgramD [5]. ProgramD es la plataforma de código abierto para bots AIML más utilizada en el mundo. Implementa todas las capacidades de AIML y soporta un número ilimitado de bots una simple instancia en el servidor [6].

AIML ofrece multitud de posibilidades a la hora de confeccionar patrones [7]. Ofrecen capacidades para mantener una conversación coherente (conversación causal) y para recordar datos relevantes del interlocutor tales como su nombre, edad e incluso aficiones (conversación personal). Los patrones en AIML son muy selectivos y potentes. Su escritura es sistemática y muy repetitiva, por lo que las bases de conocimiento escritas en AIML se prestan a su edición por parte de herramientas automatizadas. Sin embargo, es necesario hacer uso de algoritmos de análisis del código AIML para evitar que al añadir nuevos patrones AIML

interfieran en el conocimiento existente y se corrompan la base de conocimiento.

II. Gestión de Base de Conocimiento

Se pueden plantear muchas formas de automatizar la fase de enseñanza al bot. El sistema puede ser más o menos autónomo en función de la cantidad de decisiones que se toman sin consultar al usuario, decisiones de las que –por tanto- el sistema debe estar seguro –o bastante seguro. Un sistema totalmente autónomo sería capaz de, por ejemplo, leer un cierto número conversaciones guardadas hechas a través de clientes de mensajería instantánea e inducir a partir de ellas los patrones para generar AIML. Un sistema nada autónomo sería aquel que pidiera al usuario cada uno de los valores a introducir en cada uno de las etiquetas AIML. En tal caso el sistema se asemeja a un editor de XML.

A medida que la autonomía del sistema aumente también aumenta la probabilidad de cometer un error y por tanto, crear una entrada en la base de conocimiento que sea incorrecta: al aumentar la autonomía disminuye la fiabilidad. Si el sistema no es nada autónomo, y por tanto nada inteligente, todas las deducciones hechas se corroboran con el usuario y por lo tanto se le exige mayor nivel de cualificación en el uso de la herramienta.

Se pretende liberar de trabajo al usuario y desarrollar una interfaz intuitiva cuyo uso necesite la menor explicación posible. Para ello se ha alcanzado un compromiso entre autonomía y la fiabilidad desarrollando una interfaz semiautomática o asistida. El sistema toma aquellas decisiones sobre las que los algoritmos desarrollados poseen certeza absoluta. En el resto de casos presenta al usuario las decisiones tomadas y pide su corroboración. En estos casos se traducen las consideraciones propias de AIML a términos fácilmente entendibles por el usuario.

Interfaz de usuario

Se debe hacer un esfuerzo adicional en el diseño de la interfaz de usuario puesto que se pretende liberar totalmente al usuario de la necesidad de conocer el funcionamiento del sistema.

La aplicación desarrollada genera el código AIML necesario para responder lo que el usuario ha introducido como respuesta a un conjunto de frases que podríamos llamar sinónimas, que preguntan por lo mismo o hacen referencia a lo mismo.

La aplicación siempre solicita del usuario que escriba varias frases diferentes que signifiquen lo mismo (o parecido). Así se pretende inducir qué palabras deben aparecer en el patrón de código AIML. También se solicita al usuario que introduzca la respuesta que debe dar el bot cuando se recibe cada unas de estas frases como consulta.

El proceso de enseñanza está estructurado en 5 pasos. En el primero se solicita al usuario el conjunto de frases sinónimas, ya que es el punto de partida. En el segundo paso el sistema habrá analizado todas las frases para descubrir si hay palabras en plural o conjugaciones verbales y las sustituye por singulares e infinitivos. También se buscan sinónimos entre las frases introducidas por el usuario y se indica que se considerarán como la misma palabra. Ambas acciones se anuncian al usuario y se le da la opción de cambiarlas. En el tercer paso (cuya interfaz se muestra en la figura 2) las frases vuelven a ser analizadas para buscar las palabras clave, aquellas que deben incluirse en el patrón del código AIML. Los resultados se presentarán en forma de tabla coloreada en la que aparecen las frases introducidas por el usuario. Las palabras consideradas como palabras clave

Fig. 2: Extracto del paso 4 de la interfaz Web de Gestión de Preguntas.

marcadas en color. Una vez más se da la opción al usuario de modificar esta decisión y desestimar palabras que a priori se han considerado como palabras importantes u otorgar relevancia a otras. El cuarto paso pide al usuario las respuesta que el bot debe dar a cada una de las frases introducidas. También se solicita el valor de la URL a la que se hace referencia o que se quiere mostrar asociar a la respuesta en el caso de que exista alguna. El quinto paso es de comprobación. Entre el cuarto y quinto pasos el sistema genera el código AIML y lo incorpora a una copia de la base de conocimiento que denominaremos Base de conocimiento provisional. Se despliega un bot asociado a esta base de conocimiento para poder realizar un test corrección con el nuevo fichero de prueba para poder comprobar si existen interferencias de las nuevas reglas AIML introducidas con las existentes en la Base de conocimiento. En caso de que el resultado del test sea satisfactorio se dará la opción al usuario de conversar con el bot que incluye las reglas recién generadas. Si el bot ha aprendido la lección cumpliendo las expectativas del usuario se da la opción de guardar los cambios, lo que hará que se vuelquen a la base de conocimiento principal.

Arquitectura de gestión de Respuestas.

La arquitectura del sistema es más compleja que las descritas hasta el momento. Se centra en torno a la interfaz Web puesto que será el usuario quien irá autorizando las acciones a realizar. Esta formada por varios componentes.

El analizador morfológico, gestor de sustituciones y el analizador de palabras clave junto con la interfaz Web forman el componente gestor de preguntas. Este bloque es el que se encarga de generar el código AIML en función de las frases de entrada y las respuestas a las mismas introducidas por el usuario.

El Analizador Morfológico busca la raíz morfológica de cada una de las palabras presentes en las frases introducidas por el usuario en un diccionario morfológico. Sirve para reconocer singulares y plurales como la misma palabra así como tiempos verbales.

El Gestor de Sustituciones se encarga de analizar el AIML en busca de reglas que representen sustituciones. En el ámbito de este proyecto, se denominan sustituciones a las operaciones realizadas dentro del preprocesado que reciben las frases introducidas por el usuario y que transforman los plurales en singulares, los tiempos verbales en infinitivos y las palabras sinónimas entre sí. Este manejador guarda en memoria estas sustituciones para un fácil acceso y agrega las que el usuario haya incluido en la sesión de trabajo actual. El Gestor de sustituciones hace uso de un diccionario de sinónimos para proponer sustituciones de palabras por sus sinónimos favoreciendo así la comprensión del bot de mayor número de palabras.

El Analizador de Palabras Clave estudia en función de la posición en que aparecen las palabras clave en cada una de las frases introducidas por el usuario y de su frecuencia de aparición el código de AIML y el código para el Knowledge Test File que se debe generar. El diseño de este bloque es el más delicado puesto que debe tener en cuenta todas las características del lenguaje AIML y del idioma para el que se programa el bot a fin de que la base de conocimiento resultante sea escalable y a la vez funcione como desea el usuario.

Una vez generado el AIML y el código XML para el KTF el gestor de AIML y el gestor de KTF se encargan de modificar la base de conocimiento y el archivo KTF asociado a la misma respectivamente.

En la arquitectura se presenta una base de conocimiento provisional que será cargada por el servidor de bots en un Bot. Esta es la base de conocimiento que alberga los cambios hechos por el usuario y que permite al usuario y al módulo de prueba de Bots comprobar que los cambios introducidos en la misma funcionan como se desea y que no interfieren con el conocimiento previamente existente. Si ambas verificaciones concluyen satisfactoriamente la base de conocimiento provisional se copiará a la base de conocimiento principal.

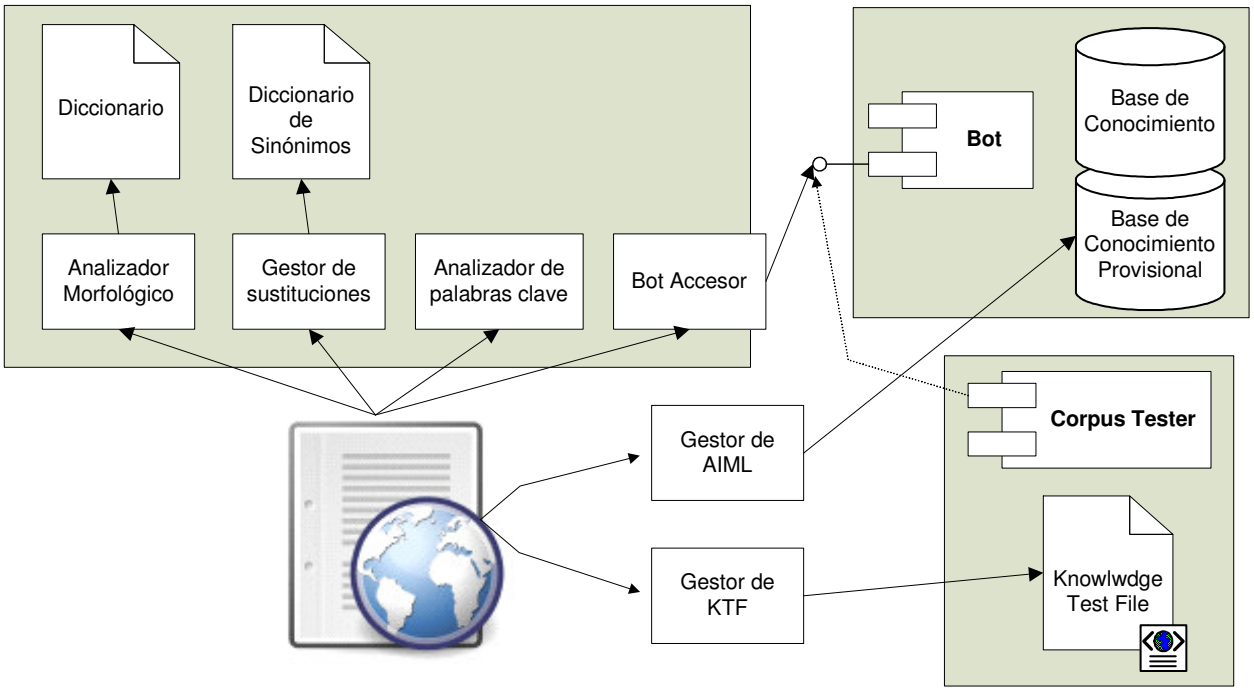

Fig. 3: Arquitectura de la herramienta Gestión de Preguntas.

Con esto se garantiza que, aunque en algún caso, el código AIML generado no cumpla con las expectativas, la base de conocimiento no se verá contaminada puesto que los cambios serán desechados.

III. Evaluación

Como conclusión se han desarrollado dos aplicaciones de edición de la base de conocimiento del bot: una para la edición de las respuestas ofrecidas por el bot y otra para añadir nuevo conocimiento de manera que el bot entienda un mayor número de frases de usuario descrita en este documento.

De forma paralela, para poder poner a prueba estos algoritmos y con el fin de generar estadísticas se ha desarrollado de un servidor de bots para el despliegue de bots. De tal manera que proporcionando los archivos que forman la base de conocimiento y los archivos de configuración se puede conversar con el bot por medio de peticiones HTTP con los parámetros adecuados. Esto facilita enormemente la creación de clientes de diversa índole.

Por último existe un módulo automatizado de pruebas de la base de conocimiento que a partir de unos ficheros de prueba comprueba si los cambios realizados en la base de conocimiento por los dos componentes anteriores interfieren con el conocimiento existente previamente.

El sistema desarrollado ha sido probado con éxito para la atención al cliente de una operadora, y actualmente se está investigando su integración para la teleeducación. En la figura siguiente se puede ver el interfaz del bot que se ofrece en la intranet de la operadora.

Fig. 4: Interfaz del bot desplegado en la intranet de una operadora.

IV. Conclusiones y trabajos futuros

En este trabajo se ha presentado una aplicación para gestionar, editar y mantener la base de conocimiento de un bot conversacional escrito en AIML de una manera eficiente, rápida, fácil y segura.

La línea de desarrollo futuro pasa por integrar todas estas aplicaciones en un framework así como proporcionar una interfaz Web conjunta para todas ellas.

Igualmente se planea desarrollar otras herramientas que permitan un control de versiones de la base de conocimiento de AIML, *merging* de bases de conocimiento entre las que existen interferencias, inclusión de los diccionarios de Open Office, módulo de estudio de logs de Bot para obtener estadísticas, creación de un servidor de bots mejorado (EnhancedProgramD)…

Agradecimientos

El proyecto ha sido realizado gracias a la colaboración con la Cátedra Orange a través de la financiación del proyecto "Aplicación de Técnicas Inteligentes al Desarrollo de Servicios de Valor Añadido basados en SIP".

Referencias

[1] Leonard, A. “Bots the origin of new species,” New York: Penguin Books, 1997, pp. 152-153.

[2] Christine Frey “Digital 'buddies' are elaborate marketing tools, but their lifelike responses in online instant messages can be misleading.” Los Angeles Times, Jul 18, 2001. Available: http://pqasb.pqarchiver.com/latimes/advancedsearch.html Los Angeles Times archive.

[3] A.L.I.C.E. AI Foundation Inc, “AIML – The Artificial Intelligence Markup Language”, official website of the A.L.I.C.E Project. [Online]. Available: http://www.alicebot.org/aiml.html

[4] Richard Wallace and Noel Bush, “AIML 1.0.1 Specifications” [Online] Available: http://www.alicebot.org/TR/2001/WD-aiml/

[5] Noel Bush, “The home of ProgramD and xAIML technologies“, official website of ProgramD Project. [Online]. Available: http://aitools.org/-Main_Page.

[6] Noel Bush and Kim Sullivan, “Getting Started with ProgramD” section :Configuration/Deployment. [Online] http://aitools.org/-Getting_Start-ed_with_Program_D# Configuration.2FDeployment

[7] Dr. Richard S. Wallace, “The Elements of AIML Style” 2004