

INGENIERÍA DE SERVIDORES (2016-2017)
GRADO EN INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE GRANADA

IBM mainframes Machine Learning & Watson

Francisco Javier Morales Piquerasa
Rubén Morales Pérez

22 de mayo de 2017

Índice

1. Resumen	3
2. Memoria	3
2.1. Introducción	3
2.2. IBM mainframes	3
2.2.1. Arquitectura	4
2.3. Watson	6
2.3.1. Arquitectura	7
2.3.2. Entendiendo las preguntas	7
2.3.3. Evolución del sistema	9
2.4. IBM Bluemix	9
3. Conclusiones	9

Índice de figuras

2.2. Grid computing [7]	4
2.3. Watson en Jeopardy!	6
2.4. Arquitectura DeepQA	7
2.5. Evolución del sistema Watson [8]	9
2.6. Logo IBM Bluemix [3]	9

Índice de tablas

1. Resumen

2. Memoria

2.1. Introducción

Hay varios retos que tienen las compañías actualmente para poder mantenerse competitivas en un mundo cada vez más globalizado. El mundo de las tecnologías de la información y la comunicación toma un papel fundamental, cada vez hay más corporaciones con página web, aplicaciones u ofreciendo información actualizada a través de redes sociales.

Una vez que tenemos una base tecnológica y cierto volumen de negocio es recomendable pasar al siguiente nivel, tener datos suficientes y de calidad recopilados de forma que podamos obtener un beneficio competitivo extrayendo información. Entonces entra en juego el análisis de datos y mantener esa información segura, por temas de protección de datos. Aquí es donde interviene IBM mainframes [4], grandes ordenadores que nos ofrecen computación en la nube de forma que podremos almacenar los datos en dichos servidores con cierta garantía y a la vez dejar a estos ordenadores el procesamiento pesado.

El primer ordenador digital de propósito general de IBM fue ASCC (Automatic Sequence Controlled Calculator), se desarrolló junto con la Universidad de Harvard. Entre los sistemas más modernos hablaremos de z Systems.



(a) ASCC



(b) Z13

2.2. IBM mainframes

Usaremos la siguiente definición de mainframe

”Un mainframe es lo que las empresas usan para alojar bases de datos comerciales, servidores de transacciones y aplicaciones que requieren un mayor grado de seguridad y disponibilidad de lo que comúnmente se encuentra en máquinas de menor escala” [5]

A día de hoy, los mainframes juegan un papel esencial en las operaciones diarias de las grandes compañías. La era de la información reclama muchas innovaciones, y las grandes compañías son víctimas en la marcha implacable del progreso. IBM desarrolla continuamente sus mainframes al mismo tiempo que mantiene la compatibilidad con aplicaciones existentes.

El término mainframe ha evolucionado gradualmente de una descripción física de las mayores computadoras de IBM hasta entenderse como un estilo de computación en sí mismo. Un punto de

inflexión en la historia de los mainframes fue el *S/360*. El desarrollo del mainframe empezó en una serie de generaciones en los años cincuenta. En esa época eran las únicas computadoras y pocas empresas podían permitírselas, servían como repositorios de datos para las empresas. En los años sesenta los fabricantes de mainframes comenzaron a estandarizar el hardware y el software que ofrecían a los clientes.

El *S/360* fue el primero de estos ordenadores en usar microcódigo para implementar muchas instrucciones máquina, sin tener todas sus instrucciones de máquina cableadas en sus circuitos. El microcódigo (o firmware, como se le llama a veces) consiste en microinstrucciones (no disponibles para los usuarios) que proporcionan una capa funcional entre el hardware y el software. Nos ofrece flexibilidad ante cambios sin tener que reemplazar el hardware. Las primeras aplicaciones empresariales se escribieron en ensamblador, COBOL, FORTRAN o PL/1, y muchos de estos programas todavía están en uso hoy en día.

2.2.1. Arquitectura

Una arquitectura es un conjunto de términos y reglas que se utilizan para crear productos, son la estructura organizativa de un sistema. Una arquitectura puede descomponerse recursivamente en partes que interactúan a través de sus interfaces.

A principios de los 90 el modelo cliente/servidor, con nodos distribuidos de computadoras menos potentes, emergió para desafiar el dominio de las computadoras mainframe. La industria en estos momentos comenzó a llamar a los mainframes dinosaurios. Entonces los diseñadores crearon nuevos mainframe, llamados T-Rex, para satisfacer la demanda. Algunos de los servicios que ofrecía dicho mainframe eran el servicio web, mayor autonomía, recuperación ante desastres y grid-computing. Grid-computing hace referencia al uso simultáneo de diferentes recursos hardware y software de forma que cumpla varias características [6]:

- Coordinar recursos de forma descentralizada
- Usar protocolos e interfaces estándares, abiertos y de propósito general
- Ofrecer un servicio no trivial

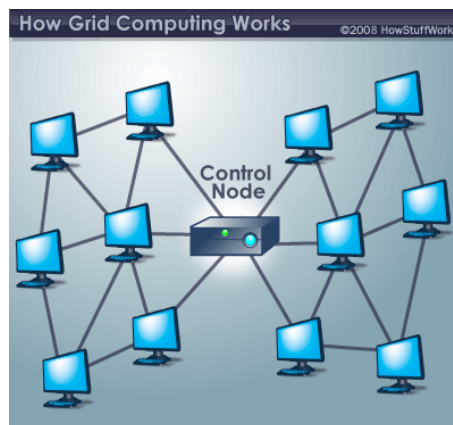


Figura 2.2: Grid computing [7]

Este tipo de computación forma los cimientos del Internet que conocemos hoy en día. Estos mainframes están diseñados para resistir los virus o comportamientos inadecuados.

Hoy en día no siempre se usa el término mainframe, se ha tendido a llamar cualquier computadora de uso comercial, sin importar su tamaño, servidor. El término mainframe se utiliza para los

servidores más grandes. Una empresa grande puede tener servidores web, de bases de datos, correo o usar un mainframe para hacer varias de estas tareas. Cuando se tiene un conjunto de servidores suele hacerse referencia a una granja de servidores. La presencia de un mainframe a menudo implica una forma centralizada de computación. Cuando usamos un mainframe tenemos mayor flexibilidad y evitamos la necesidad de actualizar varios computadores a la vez, sin embargo, el hecho de tener varios computadores distribuidos aumenta la potencia de cálculo en relación al precio. Los mainframes tienen la capacidad de configurarse dinámicamente, añadir procesadores, memoria y conexiones, mientras las aplicaciones continúan ejecutándose. Anteriormente se asociaba el término mainframe tanto con el hardware como con el software, pero dejó de tener sentido ya que el software de los grandes servidores ahora también puede ejecutarse en pequeñas computadoras. Eso permite desarrollar un programa delicado y probarlo en sistemas pequeños antes de moverlo al sistema de producción.

A menudo se utiliza el término plataforma para referirse al hardware y software como conjunto. El mainframe en sí suele cumplir alguna de las siguientes características [5]:

- Control centralizado de recursos.
- Compartir acceso disco con otros sistemas, con protección de los datos y control de privilegios.
- Procedimientos sistemáticos para copias de seguridad y recuperación desde otra ubicación alternativa.
- Trabajo rutinario con operaciones simultáneas de E/S.
- Permitir usar varias copias del sistema operativo como un único sistema, dicha tecnología se conoce como Parallel Sysplex, es análogo al concepto de cluster en UNIX. Esto permite controlar mejor las operaciones sobre las aplicaciones.
- Gran capacidad de compartir datos y recursos.

Muchas veces interactuamos como mainframes sin saberlo, por ejemplo cuando se utiliza un cajero automático.

Para identificar las fortalezas de un mainframe se utilizan las siglas RAS(**R**eliability, **A**vailability y **S**erviceability)

- Confiabilidad: El hardware tiene comprobación y recuperación de errores y se aplican mejoras software cuando existe un problema
- Disponibilidad: Capacidad de reemplazamiento del hardware y el software, aunque es deseable mantener un MTBF (Mean time between failure) lo más bajo posible
- Utilidad: Capacidad de entender el tipo de error que ha ocurrido, lo que permite conocer lo que hay que cambiar

Para mantener una buena tolerancia a fallos debe incluirse un mantenimiento constante con copias de seguridad redundantes y controlar los accesos la seguridad de los datos se define como la protección contra el acceso no autorizado, la transferencia, la modificación o la destrucción, ya sea accidental o intencional. mediante privilegios.

La seguridad de datos se define como protección contra acceso, transferencia, modificación o destrucción de datos sin autorización, ya sea accidental o intencional.

2.3. Watson

En 2007 IBM Research se propuso el reto de crear un sistema para competir con los grandes campeones en el juego Jeopardy!. Este sistema, en adelante Watson, entraría dentro de la categoría QA (Question answering), que engloba varias áreas de la ciencia de la computación y la inteligencia artificial. Algunas de estas áreas serían búsqueda y recuperación de información (IR), procesamiento de lenguaje natural (NLP), representación del conocimiento y razonamiento o el machine learning.



Figura 2.3: Watson en Jeopardy!

La comunicación humana es imperfecta, el lenguaje permite ambigüedades, polisemia, la misma información puede expresarse de varias formas, también tenemos ironías y la interpretación de una frase puede depender de conversaciones anteriores entre los interlocutores. Además, dicha comunicación no está estructurada (como en un lenguaje de programación o una base de datos), los datos no siempre están bien definidos y hay información implícita. A día de hoy la información no estructurada crece más rápido que la estructurada, por ello es buena idea desarrollar el análisis del lenguaje natural y hacer inferencia a partir de los datos de los que disponemos.

Desde 2001 hasta 2006 se construyó la base para el reconocimiento de información no estructurada, UIMA (Unstructured Information Management Architecture). UIMA proporciona una plataforma para integrar la información obtenida tras analizar textos, imágenes, etc. Su objetivo es integrar varios programas llamados *anotadores*, que asignan significado semántico a ciertas partes del texto o imagen.

Algunos desarrolladores habían participado anteriormente en Question answering dentro del proyecto PIQUANT [1]. Este sistema tenía un conjunto estático de tipos de respuestas (o clases de conceptos) que se solicitaban en una pregunta. El nuevo sistema no tendría conexión a Internet, y tendría que procesar la pregunta, buscar respuesta y estimar la probabilidad de que sea correcta. Tras analizar 2,000 partidas de *Jeopardy!* la media de los jugadores que ganaban era una precisión del 85 – 95 % de acierto sobre una media de 40 – 50 % de preguntas respondidas, o 85 % *Precision@40*. En 2007 se consiguió, usando PIQUANT, un rendimiento 16 % *Precision@70* en las preguntas de *Jeopardy!*.

2.3.1. Arquitectura

Sobre UIMA se desarrollaron dos técnicas esenciales en el desarrollo de *Watson*, la arquitectura DeepQA y un complemento para integrar distintas técnicas algorítmicas llamado AdaptWatson.

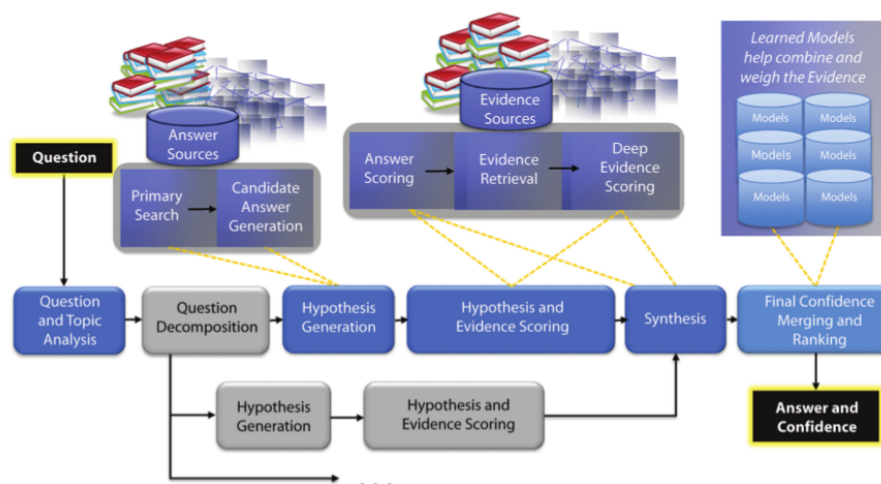


Figura 2.4: Arquitectura DeepQA

DeepQA define varias etapas del análisis, cada una tiene diferentes implementaciones que se ejecutan en paralelo y se combinan los resultados. No se presupone que se ha entendido una pregunta, de forma que pueda consultarse directamente la respuesta en nuestra base de conocimiento. Lo que se hace es generar varias preguntas candidatas, generar varias respuestas por pregunta candidata y estimar la probabilidad de que cada respuesta sea correcta.

Se le da valor al hecho de que una respuesta concuerde con el tipo que se espera, por ejemplo un monumento histórico, una persona, etc. También se tienen en cuenta fechas, geografía, la fiabilidad de las fuentes, etc. Este análisis produce cientos de valores o características, cada uno indicando el grado de evidencia dada una respuesta concreta. Con esas características se calcula la probabilidad de que una respuesta sea correcta dada una pregunta candidata. Para ponderar el valor de cada característica se entrenó un modelo estadístico usando machine learning a partir de preguntas y respuestas anteriores.

El rendimiento del sistema era bajo por lo que se integraron diferentes técnicas algorítmicas y se añadió una metodología llamada **AdaptWatson**. Estos componentes actúan sobre la arquitectura de DeepQA con el objetivo de entender las preguntas, buscar respuestas candidatas y valorar las respuestas. El equipo documentó más de 8,000 experimentos independientes, cada uno con 10 – 20GB de datos. A partir de estos datos se buscaban fallos y sus posibles causas, de esta forma se mejoró hasta un 85 % *Precision@70*.

2.3.2. Entendiendo las preguntas

Cada pregunta en *Jeopardy!* tiene una categoría que ayudará a ponderar las respuestas. Para entender las preguntas formuladas se usa un parser ESG (English Slot Grammar) y un generador PAS (predicate-argument structure).

- ESG identifica partes de la frase y roles sintácticos como el sujeto y relaciones entre las diferentes partes de la frase.

- El generador PAS crea una representación abstracta, la interfaz con la parte analítica.

Para las **respuestas y fuentes de evidencia** se añadieron enciclopedias y libros de referencia. Después se desarrolló un proceso semiautomático para aumentar la base de conocimiento, añadir contenido sin parar afecta negativamente al rendimiento del sistema. Para extraer conocimiento de las fuentes seleccionadas se usa *PRISMATIC* [2].

Cada una de las consultas usa simultáneamente información estructurada y no estructurada . Una métrica que se usa para medir la bondad de una respuesta es el porcentaje de preguntas en las que dicha respuesta es generada como candidata. Esta medida ayuda sobre todo cuando tenemos varias interpretaciones diferentes de la pregunta. De las experiencias con *PIQUANT* se sabía que no era una estrategia adecuada intentar anticipar todos los tipos de respuesta y crear algoritmos que busquen solamente instancias de dichas clases. Todo el sistema se apoyaría en reconocer correctamente el tipo de respuesta y que la respuesta esté bien clasificada dentro del sistema. En la siguiente pregunta de la categoría *decoración* el tipo de respuesta buscada es una *dirección*:

Si usted está de pie, es la dirección que debe buscar para ver el revestimiento.

Respuesta: Abajo

Para clasificar las respuestas se usó un sistema dinámico que tenía en cuenta el contexto de la pregunta, su nombre es *coacción de tipo*. Otras técnicas usadas por Watson eran *YAGO* y *WordNet*. Otra fase del proceso es **recolectar y evaluar la evidencia** (de forma paralela) de las respuestas. Aquí se utiliza la deducción de relaciones semánticas a partir de datos de entrenamiento. Por ejemplo, esta pregunta de la categoría *Madres e hijos*:

Aunque solo les separa un año de vida, ella interpretó a la madre de Colin Farrell en Alexander.

Respuesta: Angelina Jolie

Relación: protagoniza(ella, Alexander)

En Jeopardy! hay preguntas que tienen referencias implícitas en la frase, y unas partes del enunciado pueden depender de otras, como en el siguiente ejemplo de la categoría *Antes y después*:

La estrella de Jerry Maguire que mantiene automáticamente la velocidad de tu vehículo.

Respuesta: Control Tom Cruise

Primero se detectan este tipo de preguntas y las referencias enlazadas. Para interpretar las preguntas se dividen en distintas partes lógicas que puedan ser exploradas independientemente y combinar los resultados. Por ejemplo, en esta pregunta con categoría *Animales ficticios*, el sistema puede buscar por un lado personajes introducidos en 1894 y por otro aquello relacionado con "Hindi for bear":

El nombre de este personaje, introducido en 1894, proviene de "Hindi for bear".

Respuesta: Baloo

El sistema debe mejorar las 2 horas de procesamiento que necesita para responder una pregunta (en 2008). El *UIMA-AS* permite a cualquier aplicación *UIMA* ser desplegada en procesos asíncronos que utilizan paso de mensajes. Cada pregunta genera varias preguntas candidatas que se ejecutan independientemente, cada respuesta potencial por pregunta se busca independientemente, y cada comprobación de evidencia también. Tras usar 2880 procesadores la latencia de respuesta bajó de dos horas a tres segundos.

Este sistema de pregunta-respuesta es estático y no aprovecha todo el potencial de la arquitectura DeepQA. Watson puede aplicarse a la salud modificando su base de conocimiento y enseñándole a hacer las preguntas adecuadas a los pacientes para obtener información sobre ellos e inferir una solución a partir de su historial médico.

2.3.3. Evolución del sistema

Watson en la actualidad no solamente se utiliza aplicado a la medicina, también se usa en finanzas e inteligencia de negocio en general.

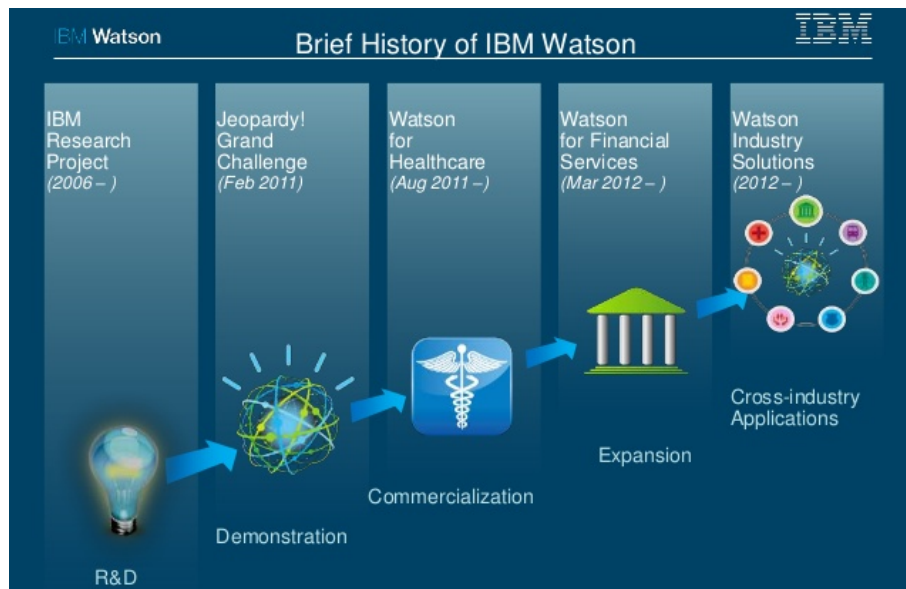


Figura 2.5: Evolución del sistema Watson [8]

2.4. IBM Bluemix

IBM Bluemix



Figura 2.6: Logo IBM Bluemix [3]

3. Conclusiones

Referencias

- [1] IBM T.J. Watson Research Center. Ibm's piquant in trec2003.
- [2] James Fan, David Ferrucci, David Gondek, and Aditya Kalyanpur IBM Watson Research Lab. Prismatic: Inducing knowledge from a large scale lexicalized relation resource. <http://www.aclweb.org/anthology/W10-0915>.
- [3] Inge Lorenzen <https://www.ibm.com/blogs/nordic-msp/author/inge-lorenzen/>. Free ibm blue-mix trial. <https://www.ibm.com/blogs/nordic-msp/free-ibm-bluemix-trial/>.
- [4] IBM. Ibm mainframes. https://www-03.ibm.com/ibm/history/exhibits/mainframe/mainframe_intro.html.
- [5] IBM. Mainframe concepts. <https://www.ibm.com/support/knowledgecenter/zosbasics/com.ibm.zos.zmainframe/toc.htm>.
- [6] Ian Foster. Argonne National Laboratory University of Chicago. What is the grid? a three point checklist. <http://dlib.cs.odu.edu/WhatIsTheGrid.pdf>.
- [7] JONATHAN STRICKLAND. How grid computing works. <http://computer.howstuffworks.com/grid-computing.htm>.
- [8] Salvador tapia <http://www.mixtrategy.com/author/admin/>. Watson de ibm. <http://www.mixtrategy.com/dr-watson/>.