Universidad Politécnica de Cataluña

Facultad de Informática de Barcelona

Ingeniería de software

Repositorio de árboles genealógicos en BD NoSQL

Autor:
Daniel Albarral Nuñez

Supervisor:
Enric Mayol

Q1 - 2015-2016

Contents

1 Contextualización del proyecto

1.0.1 Los arboles genealógicos

Un arboles genealógico, también llamado genorama, es la representación gráfica de los antepasados y descendientes de un individuo.Para su representación se suelen usar tablas o arboles, siendo esta ultima la forma más común y la que se usara en el proyecto.

1.0.2 Uso y aplicación de los arboles genealógicos.

Los arboles genealógicos son una herramienta de la genealogía, que se encarga de estudiar y seguir la ascendencia y descendencia de una persona o familia. La genealogía es una ciencia auxiliar de la Historia y es trabajada por un genealogista. Uno de los objetivos del software a desatollar es dar soporte a los genealogistas. Por otro lado hay varias comunidades de aficionados que llevan sus propios arboles genealógicos, el software creado también les podrá dar servicio a esta tipología de usuarios.

1.1 Perspectiva general del software actual.

Todo software genealógico, como mínimo permite almacenar la siguiente información de un individuo: fecha y lugar de nacimiento, fecha de casamiento, muerte y relaciones familiares, contra más flexible es el programa más información te permite introducir acerca de un individuo. También proporcionan diferentes maneras de representar la información y permiten exportar a GEDCOM la información representada.

GEDCOM [1] (**G**enealogical **D**ata **COM**munication): Es un formato de archivo de datos, proporciona un formato flexible y uniforme para el intercambio de datos genealógicos computarizados.

La mayor parte del software genealógico actual esta basado en soluciones de escritorio, pero en los últimos años han proliferado diferentes soluciones web como myheritage o familysearch, que no solo sirven como plataforma de edición sino que también son grandes plataformas *cloud* en las que se almacenan los arboles genealógicos.

Las soluciones más avanzadas, aparte de la gestión de arboles también ofrecen herramientas más orientadas a la investigación, como podrían ser sistemas de búsqueda de individuos basados en sus relaciones o herramientas estadísticas.

1.2 Tecnología

Dada la naturaleza social del tipo de software que se busca desarrollar, nacen ciertas complicaciones tecnológicas que en los últimos tiempos se han sido considerablemente investigadas, debido a la gran repercusión de las redes sociales. Las tecnologías clásicas orientadas a la persistencia de datos como las bases de datos SQL, plantean la dificultad de tener un coste muy alto de consulta cuando se pregunta acerca de datos con un alto nivel de relación entre ellos. Una de las soluciones más usadas para solucionar este problema son las bases de datos basadas en grafos, uno de los casos de éxito es el caso de Twitter, que desarrollo su propia solución, FolckDB, una base de datos basada en grafos, tolerante a fallos, diseñada para tratar con grandes conjuntos de datos, con información no critica.

En el libro **Neo4j in Action** [3], Partner and Vukotic realizan el siguiente experimento:
El experimento consiste en comparar una base de datos relacional con neo4j *GraphDB*, en ambas se representa el mismo modelo donde diferentes personas tienen multiples amigos. Se realizan diferentes consultas donde se pretende aberiguar si existe un camino entre dos personas escojidas al azar que los una en una profundidad de como mucho cinco relaciones. Las bases de datos constan de 1000000 personas, cada una con 50 amigos aproximadamente. Los resultados fueron los siguientes:

Depth	RDBMS execution time(s)	Neo4j execution time(s)	Records returned
2	0.016	0.01	~2500
3	30.267	0.168	~110,000
4	1543.505	1.359	~600,000
5	Unfinished	2.132	~800,000

Figure 1: Comparativa entre neo4j y base de datos relacional.

Con este experimento podemos ver como en las consultas donde la profundidad es mayor, o dicho de otra forma, donde la conectividad juega un papel más importante, la base de datos basada en grafos es mucho más rápida. Esto es debido, en gran parte, a que las relaciones en las bases de datos basadas en grafos son *ciudadanos de primer orden* a diferencia de las bases de datos relacionales que tratan las relaciones mediante claves foráneas, lo que proboca que tengan que usar una gran cantidad de joins para resolver el problema, elevando mucho el coste computacional.

1.2.1 Bases de datos basadas en grafos.

La gran cantidad de proyectos que han nacido en los últimos años hace prácticamente imposible hacer una comparativa concreta de todas las tecnologías existentes. Pero podemos diferenciar el panorama actual haciendo dos generalizaciones:

Tecnologías usadas para propósitos transaccionales .

Orientadas ha dar un servicio online en tiempo real a una aplicación.

Estas tecnologías son llamadas bases de datos basadas en grafos. Estas son nuestro principal objeto de estudio. Son el equivalente a las OLTP en el modelo transaccional.

Tecnologías usadas principalmente para el análisis de grafos .

Llamados Motores de procesamiento de grafos, siguiendo el mismo símil que antes, podríamos pensarlos como herramientas de *data mining* y análisis de procesos(OLAP)

OLTP (**O**n**L**ine **T**ransaction **P**rocessing):
Es un tipo de procesamiento que facilita y administra aplicaciones transaccionales, usualmente para entrada de datos y recuperación y procesamiento de transacciones (gestor transaccional). Los paquetes de software para OLTP se basan en la arquitectura cliente-servidor ya que suelen ser utilizados por empresas con una red informática distribuida..

Las bases de datos basadas en grafos son sistemas de bases de datos que permiten operaciones CRUD (*Create, Read, Update y Delete*) sobre los objetos representados en ellas. Suelen estar orientadas a funcionar con sistemas transaccionales (OLTP), como aviamos mencionado anteriormente. Sus principales propiedades son, las relaciones son *ciudadanos de primer orden*, a diferencia de otros modelos, como el relacional que tienen que usar claves foráneas. En este tipo de base de datos para representar el dominio de nuestro problema simplemente nos basta con definir los nodos y las relaciones que lo componen.

En el libro Graph Databases [2], encontramos el siguiente gráfico (Figure 2), que nos da una idea de las principales tecnologías basadas en grafos y su orientación.

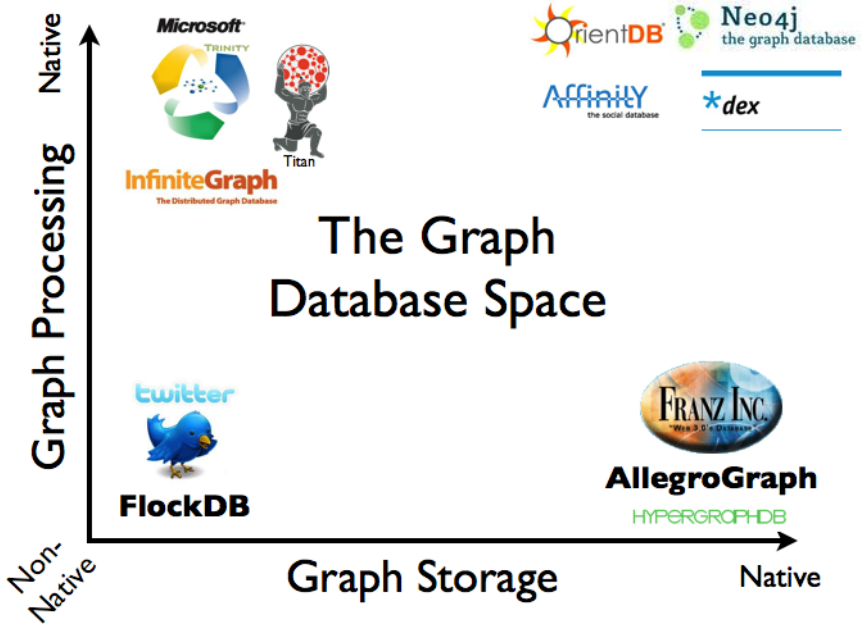


Figure 2: Visión del conjunto de tecnologías

2 Planificación

Para el desarrollo del proyecto se usara *Scrum*, por ello la planificación temporal estará estructurada para facilitar la consecución de esta metodología. El final del proyecto se ha establecido el 22 de abril del 2016.

2.1 Gantt

El gantt adjuntado a continuación tiene varias peculiaridades:

Dependencias de las tareas .

Dado que en el proyecto se desarrolla íntegramente por una sola persona, la secuencialidad del diagrama es absoluta, por ese motivo también se ha omitido el gráfico PERT.

Fase de desarrollo .

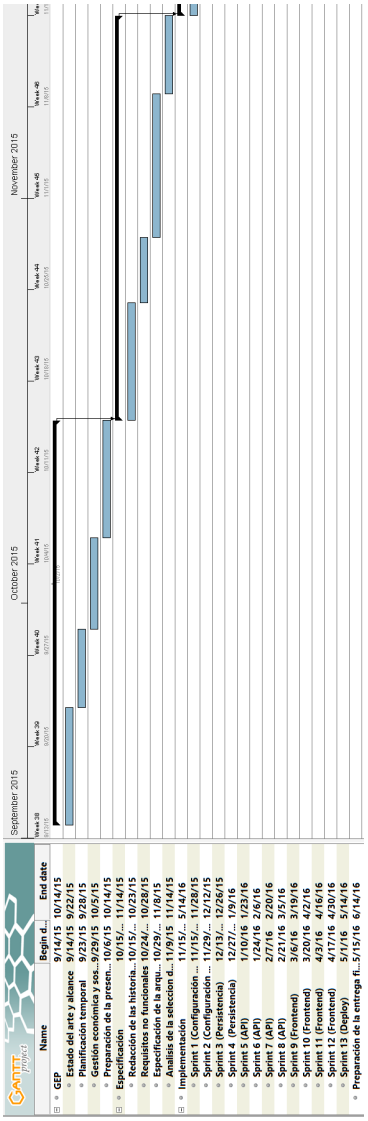
Durante la fase de desarrollo simplemente se especifican los *Sprints* que se llevaran a cabo hasta conseguir la consecución del proyecto, a medida que las historias de usuario se vayan redactando y priorizando se irán asignando a los *sprints*.

2.1.1 Tareas

Tasks

Name	Begin date	End date
GEP	9/14/15	10/14/15
Estado del arte y alcance	9/14/15	9/22/15
Planificación temporal	9/23/15	9/28/15
Gestión económica y sostenibilidad	9/29/15	10/5/15
Preparación de la presentación final de GEP	10/6/15	10/14/15
Especificación	10/15/15	11/14/15
Redacción de las historias de usuario	10/15/15	10/23/15
Requisitos no funcionales	10/24/15	10/28/15
Especificación de la arquitectura	10/29/15	11/8/15
Analisis de la seleccion de tecnologias	11/9/15	11/14/15
Implementación	11/15/15	5/14/16
Sprint 1 (Configuración del entorno de pre y pro)	11/15/15	11/28/15
Sprint 2 (Configuración del entorno de pre y pro)	11/29/15	12/12/15
Sprint 3 (Persistencia)	12/13/15	12/26/15
Sprint 4 (Persistencia)	12/27/15	1/9/16
Sprint 5 (API)	1/10/16	1/23/16
Sprint 6 (API)	1/24/16	2/6/16
Sprint 7 (API)	2/7/16	2/20/16
Sprint 8 (API)	2/21/16	3/5/16
Sprint 9 (Frontend)	3/6/16	3/19/16
Sprint 10 (Frontend)	3/20/16	4/2/16
Sprint 11 (Frontend)	4/3/16	4/16/16
Sprint 12 (Frontend)	4/17/16	4/30/16
Sprint 13 (Deploy)	5/1/16	5/14/16
Preparación de la entrega final	5/15/16	6/14/16

2.1.2 Gráfico gantt



2.2 Recursos

Las necesidades en recursos serán muy escuetas dado que todo el software que se usara es libre y las maquinas en las que se trabaja en preprducción serán virtuales. La única necesidad de tener unos recursos que se necesiten planificar, ya que requieren ser contratados, son los servidores en los que se hará el *deploy*, se tendrán que contratar durante las semanas que dures los *sprints* en los que se haga el *deploy*.

2.3 Alterativas y plan de acción

Dado que el proyecto se desarrollara usando *Scrum*, al final de cada *sprint*, durante el *sprint backlog* se determinara si se han cumplido las expectativas del *sprint*. Por otro lado en el *sprint planning* se tendrá en cuenta si han habido carencias o bugs en los anteriores *sprints* para incorporarlos como historias de usuario al siguiente *sprint*.

Como se ha comentado en el alcance, se diseñara un software abierto orientado al desarrollo continuado. Por lo que en el sprint 13 se dará por concluido el proyecto, y se documentara el estado en el que se encuentre y las historias de usuario pendientes.

3 Metodología y rigor

Las metodologías usadas son las siguientes:

3.1 SCRUM

Para el desarrollo del proyecto se ha escojido como metodologia de trabajo SCRUM, los sprints dado la naturaleza del proyecto se han adaptado esta metodologia agil para ser usada por una sola persona. La metodologia de trabajo se ha definido de la siguiente forma:

Product backlog :

Para crear el *product backlog* se han dfinido un seguido de historias de usuario, de acuerdo con el tutor del proyecto, que asume el rol de *product owner*. Estas historias de usuario se revisaran constantemente en los *sprint backlogs*.

Sprints :

Los sprints se desarrollaran a lo largo de dos semanas, en este tiempo se desarrollaran las historias de usuario escojidas.

Sprint backlog :

Se realizaran al final de cada sprint, dado que esta tarea estara limitada por la disponibilidad del tutor y el alumno, se dara flexibilidad pudiendo realizar el *sprint backlog* de varios sprints.

4 Análisis de alternativas

5 Integración de conocimientos

6 Integración de leyes y regulaciones

References

[1] Ryan Heaton. About gedcom, 2014.

[2] Ian Robinson, Jim Webber, and Emil Eifrem. *Graph Databases*. O'Reilly Media, Inc., 2013.

[3] Aleksa Vukotic, Nicki Watt, Tareq Abedrabbo, Dominic Fox, and Jonas Partner. *Neo4j in action*. Manning, 2015.