



“Búsquedas Textuales en el Cluster”

Ing. Marcelo Pesce

Ing. Horacio Vico

**División Tecnología Informática
Poder Judicial**

La jurisprudencia

- Ciencia del Derecho. Activo de información valioso en el ejercicio profesional
- 1996 primera base de datos informatizada en el Poder Judicial.
 - Cliente interno y convenios
 - Usuario especializado
- Nuevas necesidades -> una nueva base de jurisprudencia
- RENAJU

El Proyecto

- Fase 1: Migración desde BRS/Search a tecnologías “en uso”.
- Cronograma de 8 meses. Proyecto total: 7 años.
 - No hay retrasos
 - Retroalimentación Producto/Requerimientos
- 4 versiones como hitos tecnológicos
 - Oracle
 - Oracle Full Text
 - Lucene
 - Lucene/InfiniSpan

¿Por que esta evolucion tecnológica?

- Acompaña la evolución natural de la informática y sus tendencias
- Hay un cambio progresivo en los requerimientos
- “Evolución” tecnológica como respuesta a 5 DESAFIOS planteados al equipo de desarrollo en la ejecución de este proyecto.

1.- DISTANCIA ACADÉMICA

¿Yo habré faltado cuando dieron esto en clase?

1.- DISTANCIA ACADÉMICA

- El concepto de “recuperación de información”.
- Experiencia profesional y formación orientada a sistemas transaccionales. Atributo = valor y no recuperación de información
- Con el tiempo este aspecto demostró ser gravitante a la hora de evaluar el proyecto.
- Por lo tanto se requiere una maduración del equipo, ya no en aspectos funcionales como en cualquier proyecto, sino en la forma de pensar, analizar y diseñar (no sólo operaciones con strings).
- Cada versión refleja el intento de “madurar” el problema de la recuperación de la información.

2.- PLATAFORMAS TECNOLÓGICAS

- Plataformas tecnológicas imponen una visión de la realidad.
- Derivado de lo que analizamos anteriormente, si tengo que cambiar mi cabeza, mi forma de pensar, de analizar... seguramente esas plataformas que me resultan muy productivas y cómodas para el “sistema de facturación”, no sean las más idóneas para recuperar información.
- Primer error: aplicar la plataforma de desarrollo (GeneXus/Oracle) a un dominio de problema en el que no es idónea.
- Costo de las migraciones tecnológicas (formatos de datos).

3.- AUMENTAN PRESTACIONES - AUMENTAN REQUERIMIENTOS

- Al brindar más prestaciones y a un universo mayor de usuarios (cliente interno) se comienzan a recibir mayores requerimientos.
- Cambian las necesidades, se requieren mayores controles.
- Apertura del sistema de usuarios expertos (jurisprudencia) al usuario estándar requiere mayor atención a los aspectos de usabilidad.
- Interpretar las dificultades experimentadas a partir de los comentarios.

3.- AUMENTAN PRESTACIONES - AUMENTAN REQUERIMIENTOS

- Maduración del usuario y sus necesidades... del “me gustaría acceder” al “necesito”.
- Mayores prestaciones en el back end a fin de mejorar la calidad y disponibilidad de la información
- Mayores prestaciones en el front end a fin de mejorar la experiencia del usuario. Diferentes tipos de búsqueda Simple, Selectiva, Avanzada.
- Convenios, Interfaces específicas.

4.- “LO QUE QUEREMOS ES MUY SIMPLE... HAGAN ALGO COMO GOOGLE!!”

- Ser informático en estos tiempos es todo un reto... las expectativas del usuario son INFINITAS.
- Cómo se lo explicamos?
- Los usuarios no tienen por qué saber lo que hay detrás. Esto nos impone un altísimo nivel de expectativa.
- Desarrollar la búsqueda “google” fue sin dudas la más compleja y será la que de futuro requerirá más “conocimiento agregado”.
- La complejidad no está sólo en saber buscar, sino principalmente en saber “entender” o hasta incluso “intuir” lo que el usuario quiere encontrar.

5.- “QUEDÓ Lindo.. YA LO PUEDEN PONER EN INTERNET Y ABIERTO”

- Nunca subestimemos la capacidad de propuesta de nuestros usuarios!!
- Este sí es un reto tecnológico.
- Preparar una infraestructura capaz de soportar una demanda desconocida.
- Punto de inflexión claramente definido en esta etapa del proyecto.
 - Convencer a nuestro cliente interno que esto no era cuestión de abrir una canilla, concientizarlos del “nuevo público objetivo”
 - Diseñar una solución “responsable” en el uso de recursos, por ende que fuera “escalable” según la determinación de la demanda.
 - Mínimo impacto posible en un sistema que estaba ejecutando en forma estable para los clientes internos

¿IMPORTA EL PROCESO?



Plataforma de Desarrollo JBoss: SEAM 2

- Propuesta para integrar EJB3 y JSF. También POJOs como objetos de negocio.
- Framework full-stack.
- Introduce nuevos contextos: Conversation, Business proc.
- Inyección de dependencias: Bijection
- Generación automática de CRUDs: seam-gen
- JBoss Tools
- Integración con RichFaces y ICEFaces
- Generación de XLS, PDF, gráficos, etc.

Plataforma de Desarrollo JBoss: RichFaces

- Biblioteca para JSF de JBoss.
- Permite crear interfaces web full AJAX.
- Suite completa de componentes: <http://showcase.richfaces.org/>
- A4JSF integrada, añade interactividad AJAX a cualquier componente con facilidad.
- Soporte skins, themes, CSS, etc.

Plataforma de Desarrollo JBoss: Hibernate

- ORM de Jboss
- Potente lenguaje de consulta (HQL)
- Database reverse engineering
- Mapeo de entidades mediante anotaciones o XML

Búsquedas textuales: Apache Lucene

- Biblioteca / API de Apache, para indexación y búsquedas textuales de información.
- Maneja el “documento” como unidad, y los “fields” como sus atributos.
- Indices: Estructura de archivos invertidos, términos, punteros.
- Backends: Filesystem, RAM, JDBC, caches (Infinispan!)
- Dos formas de construir consultas:
 - API / Programa
 - QueryParser: BJN

Apache Lucene: QueryParser

- **Buscar por campos, frases y términos:**

titulo:"Quijote de la Mancha" AND
autor:Cervantes

- **Comodines:**

Gon?ale? => Gonzalez, Gonçales

test* => testamento, testimonio, testing

- **Búsquedas Fuzzy (parecidos):**

casa~

casa~0.8

- **Búsquedas de Proximidad:**

“accidente peatón”~10

- **fechas:**

fecha:[20120101 TO 20121231]

titulo:{Aida TO Carmen}

- **Boosting: Muy útil!**

responsabilidad OR “daños y perjuicios”
^5

- **Operadores Booleanos**

AND, OR (implícito), NOT, +, -

- **Agrupar cláusulas / subqueries**

(Montevideo OR Salto) AND Uruguay

titulo:(+”Quijote” +“El ingenioso hidalgo”)

- **Rangos alfanuméricos o de**

Hibernate Search

- Hibernate + Lucene
- Persistencia de objetos en B. de Datos, y actualización transparente de índices Lucene.
- Anotaciones para entities
- Búsqueda en profundidad sobre el modelo de entidades!
libro.autor.apellido:Cervantes AND libro.fecha_publicacion:1605
- Escalable, varias opciones

Hsearch – Escalabilidad: Cluster sincrónico

- Índice compartido por todos los nodos del cluster.
- Locks: Solo un nodo puede actualizar en un momento dado.
- Cada Entity maneja su lock
- Si se persisten los índices en filesystem, el lock es un archivo. Al compartir el directorio pueden aparecer problemas por caches intermedios que no muestran cambios de inmediato (NFS).
- Si se persisten en B. de Datos, se debe cuidar la performance y se deben superar otros problemas de concurrencia.

Hsearch – Escalabilidad: Cluster sincrónico

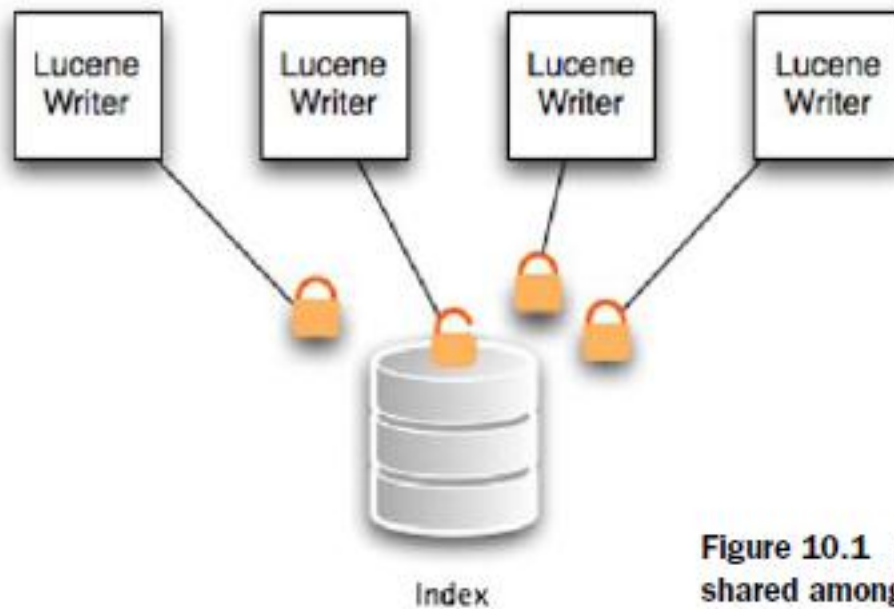


Figure 10.1 The index's pessimistic lock must be shared among all nodes.

Hsearch – Escalabilidad: Cluster asincrónico

- Concepto de nodo master y slaves
- Los slaves manejan localmente sus índices para las consultas. Las modificaciones se envían al nodo master.
- Cada cierto período de tiempo (configurable), los slaves actualizan su copia read-only local desde el master:

Problema?

- El master es el único que establece locks sobre índices.
- La comunicación se realiza por defecto mediante colas JMS.
- Master: Único punto de falla? JMS persistente / redundancia pasiva.

Hsearch – Escalabilidad: Cluster asincrónico

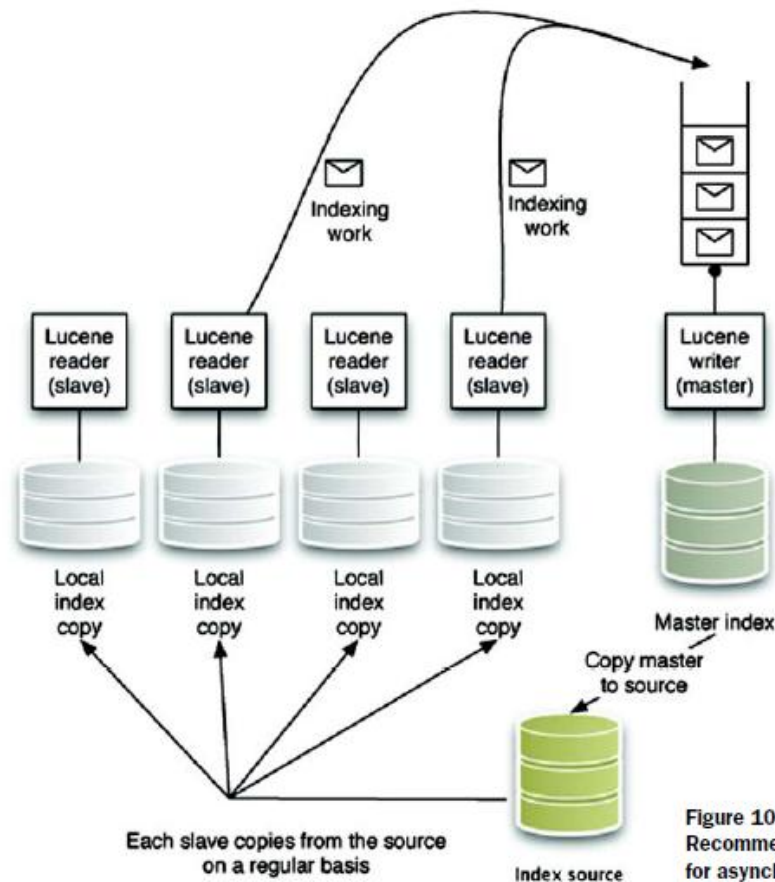


Figure 10.2
Recommended architecture
for asynchronous clustering

HSearch: Cluster “menos” asincrónico :=)

- Backend JGroups
 - Sistema de comunicaciones multicast escrito en JAVA
 - Múltiples protocolos de transporte, UDP (IP Multicast), TCP o incluso JMS.
 - Esquema master/slave, pero con distribución inmediata de cambios.
 - JGroups añade control de fallos / retransmisión (es UDP, hay que cuidarse!).
 - Soporta cifrado, y maneja protocolos para respetar el orden de los mensajes (FIFO, etc).

Hsearch: Almacenamiento de los índices

- Filesystem
- RAM
- Filesystem-master
- Filesystem-slave
- InfiniSpan

InfiniSpan

- Plataforma de data grid, licencia LGPL y 100% JAVA (pero utilizable desde otras plataformas).
- DataStore ultra escalable, basado en el concepto clave/valor
- Expone estructura Cache que extiende `java.util.Map`
- Descubrimiento “on the fly” de nuevos nodos, transferencia de estado. Se pueden sumar nodos bajo demanda rápidamente.
- Memoria HEAP masiva
 - Replicado: 100 servidores con 2 Gb = 2 Gb de HEAP replicado
 - Distribuido: 100 servidores con 2 Gb = 100 Gb de HEAP (margen para redundancia / tolerancia a fallos)

InfiniSpan

- Roadmap:
 - Computación distribuida Tirar un Runnable al grid! :=)
 - Management: Herramientas específicas en desarrollo. Ya se puede monitorear con JBoss RHQ.
- Transporte de red para replicación/distribución: JGroups!
- En la BJN hay dos canales JGroups activos, uno HSearch y el otro InfiniSpan.
- CacheStores / CacheLoader (Backend de persistencia)
 - Filesystem
 - JBDC: String o Binario (MySQL en BJN)
 - JClouds: Amazon S3
 - Otro cluster InfiniSpan
 - Y más opciones...

InfiniSpan y JGroups

- Configuración
 - Ventaja: Muy detallada, hilado muy fino.
 - Desventaja: Muy detallada, poca documentación: pero el foro oficial es activo
- Eviction y Expiration
 - Todo un problema al almacenar índices Lucene.
 - Se debe mantener toda la información en un CacheStore (JDBC en nuestro caso).
 - Al iniciar el cluster se cargan los índices en forma tardía (lazy loading)
 - Cuando se une un nuevo nodo, trae el estado de memoria desde otro cualquiera.
 - Desactivar Expiration, no tiene sentido en este uso
- Consideraciones en nuestra BJB:
 - Se aprovechó InfiniSpan para compartir información entre nodos.
 - Los JBoss AS no están configurados en cluster, son independientes 100% y no hay replicación de sesión.
 - La BJB “interna” es el master HSearch, y carga los índices InfiniSpan desde el JDBC al iniciar.
 - 3 Nodos BJB Pública, solo trabajan en RAM (400 Mbytes @ 50.000 documentos).

Balanceo: ha-proxy

- Balancedor TCP/HTTP
- Distintos algoritmos de balanceo: round robin con pesos, static-rr, leastconn, source (hashed), uri (hashed).
- Cookie para mantener sesión con el mismo nodo cuando sea posible.
- Chequeo de fallas flexible:
 - Httpcheck, smtpcheck, mysql-check, ssl-hello-check
- En la BJN, se hace un httpcheck contra una página que realiza una consulta trivial (de bajo costo) sobre el MySQL. Se chequea entonces tanto el app server como el serv. de B. de Datos.

Estadísticas BJNI Pública

- Promedio de unas 2000 búsquedas diarias
- 75% son búsquedas simples!
- 2500 sentencias judiciales vistas por día (5% del total!)
- Se realizó testing de carga con el framework Grinder
 - Siendo conservadores se pueden realizar 100 búsquedas concurrentes por nodo.
 - En caso de incremento de demanda: copy/paste de la VM y levantar un nodo InfiniSpan más en el blade.

Preguntas

?

Gracias

bjn.poderjudicial.gub.uy