



PostgreSQL em Ambiente Financeiro de Alta Críticidade

Data: 24 de outubro de 2009

Flavio Henrique Araque Gurgel



<http://creativecommons.org/licenses/by-sa/3.0/deed.pt>

4Linux na Caixa

- ▶ Suporte nível 2, 24/7
- ▶ Configurações, *Tuning*
- ▶ Testes e documentação
- ▶ Transferência de conhecimento
- ▶ Compromisso: profissionais participantes das comunidades
- ▶ Correções e novas funcionalidades devem ser revertidas aos projetos

Caixa Econômica Federal

- ▶ 100% dos municípios brasileiros
- ▶ Benefícios sociais
- ▶ “Todo brasileiro já usou ou vai usar os serviços da Caixa”
- ▶ A Caixa também é um banco!

O ambiente Multicanal

O que é

- 1) Transações bancárias e sociais
 - ▶ Hoje: auto-atendimento
- 2) Monitoramento
- 3) Mensagens Curtas para Celulares

O ambiente Multicanal

Alguns números

- ▶ Mais de 20.000 ATMs
- ▶ Pico: mais de 6.000.000 de transações bancárias e sociais em um dia
- ▶ Pico: mais de 18.000.000 de transações de banco de dados em um dia
- ▶ Mais de R\$ 1 bilhão por mês

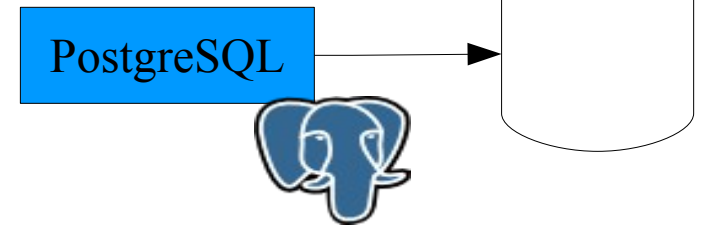
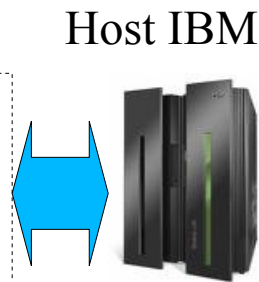
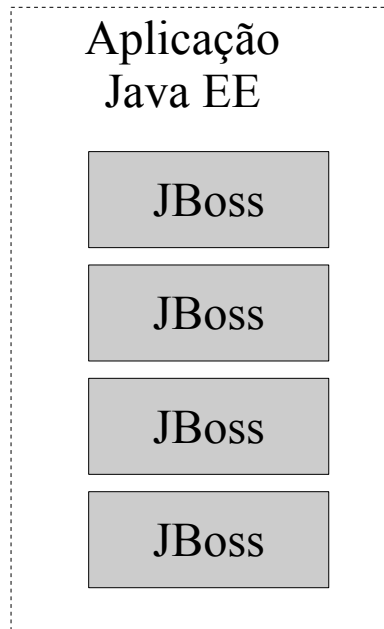
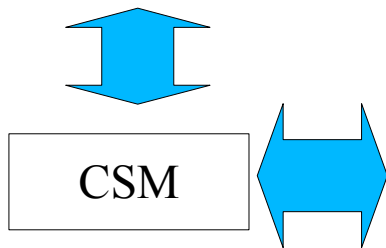
O ambiente Multicanal

Características

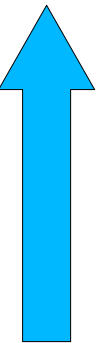
- ▶ Alta disponibilidade
- ▶ Alta performance
- ▶ Alta confiabilidade

MISSÃO CRÍTICA!!!

O ambiente Multicanal Transações (auto-atendimento)



Replicação via Hitachi USP-VM



O ambiente Multicanal

Os testes com o sistema

Mínimo de 750 TPS
Transações Financeiras por Segundo

zOS + WebSphere + DB2

Debian + JBoss + PostgreSQL

Solaris + Sun Java System + Oracle

O ambiente Multicanal

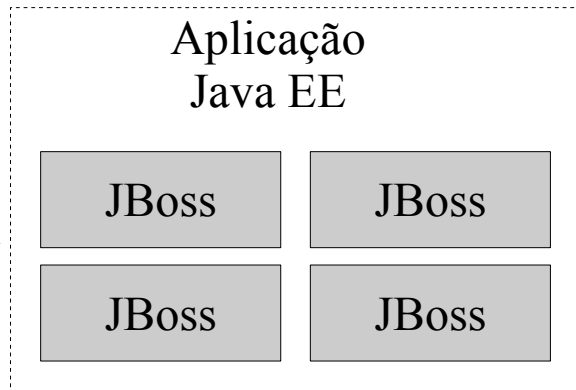
A escolha

- ▶ zOS + WebSphere + DB2
O maior TPS.
- ▶ Debian + JBoss + PostgreSQL
atingiu TPS mínimo
1% dos custos em relação ao zOS
certificado para a solução
- ▶ Solaris + JBoss + PostgreSQL
Solução com os menores custos totais
Solaris é largamente utilizado na Caixa
plataforma definitiva

O ambiente Multicanal Monitoramento



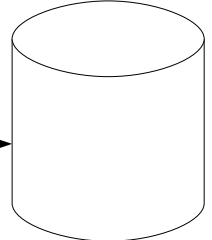
Aplicação Transações



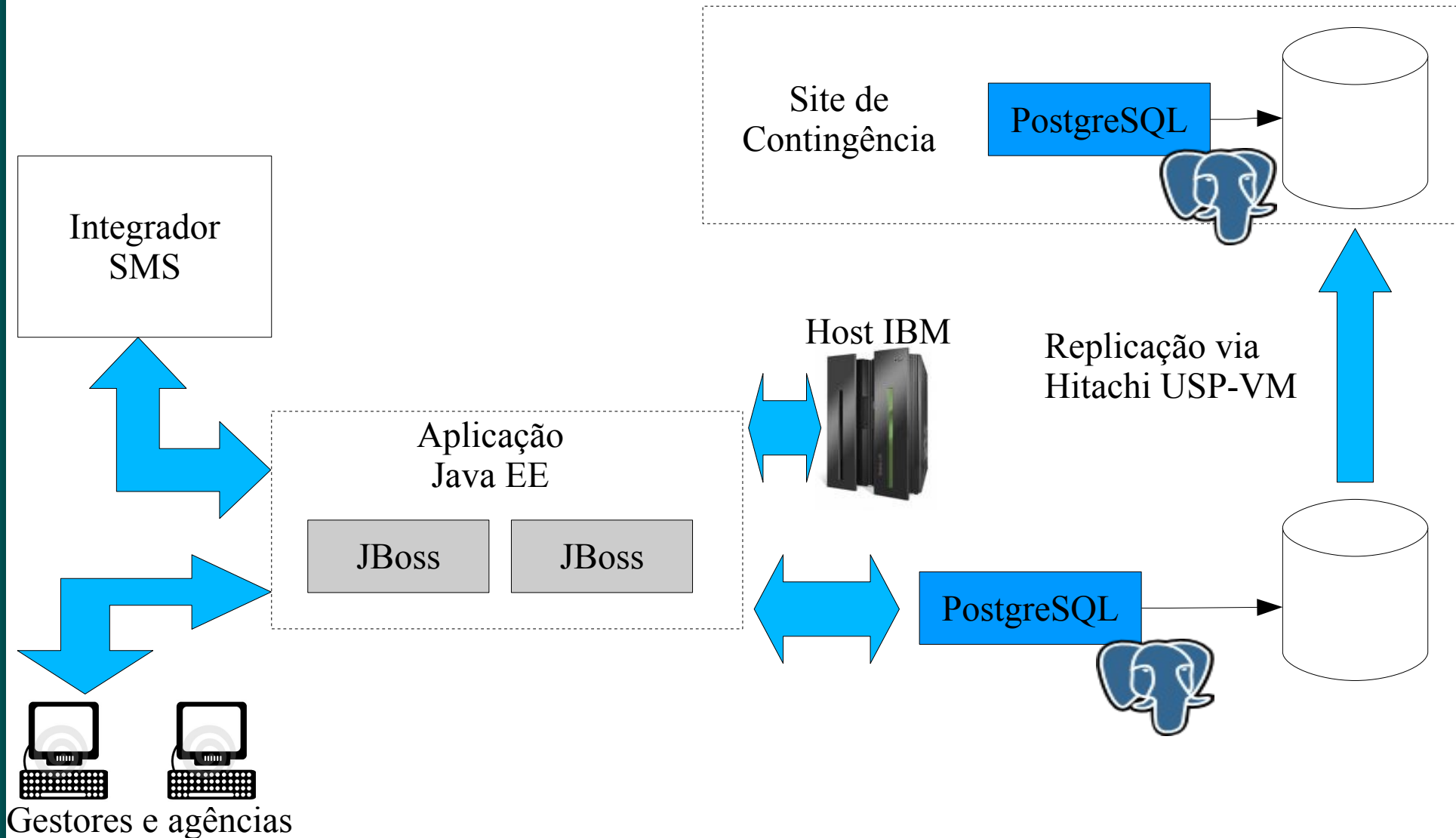
PostgreSQL



Replicação via Hitachi USP-VM



O ambiente Multicanal Mensagens para celular



O ambiente Multicanal Software Infraestrutura

- ▶ Sistema de mensagens curtas
 - ▶ Debian Etch kernel 2.6.24/64bit
 - ▶ PostgreSQL 8.3.6 – código oficial compilado 64bit
 - ▶ 20 GB de dados
- ▶ Sistemas de transações e monitoramento
 - ▶ Iniciados em Debian Etch/x86-64bit até escala final
 - ▶ Migrados para Solaris 10u6/Sparc 64bit
 - ▶ PostgreSQL 8.3.7 – Pacote Sun - 64bit
 - ▶ Transações – OLTP - 250 GB de dados
 - ▶ Monitoramento – DW - 1,5 TB de dados

O ambiente Multicanal Hardware

- ▶ Servidores Dell:
 - ▶ 2x Quad Xeon HT (2x4x2HT cores) @1.8GHz
 - ▶ 16 GB RAM
- ▶ Servidores HP:
 - ▶ 2x Quad Xeon (2x4 cores) @2GHz
 - ▶ 32 GB RAM
- ▶ Sun Fire E25K (por system board):
 - ▶ 2x Ultra Sparc IV+ (4x2 cores) @1.5GHz
 - ▶ 32 GB RAM

O ambiente Multicanal

Sub-sistema de armazenamento

- ▶ Características:
 - ▶ *Storage Area Network – Fiber Channel*
 - ▶ Virtualizador Hitachi – *cache* 16 GB
 - ▶ Switch McData
- ▶ Produção:
 - ▶ Hitachi USP-VM – *cache* 8 GB
 - ▶ SATA 15000 rpm – RAID5 – 6 e 7 discos
- ▶ *Backup:*
 - ▶ *On-line: Storage* EMC CLARiiON CX Series
 - ▶ Fita: IBM Tivoli



O ambiente Multicanal Replicação

- ▶ Virtualizador Hitachi
- ▶ Fibra óptica
- ▶ Comprimento aprox. 4km

Desafios diários

- ▶ Disponibilidade
- ▶ Estabilidade OLTP
- ▶ Performance DW
- ▶ Manutenção
- ▶ Paradigmas de outros SGBD

Desafios diários

Disponibilidade

- ▶ *Backup PITR – Point-In-Time Recovery*
- ▶ Uma cópia inicial por dia
- ▶ Dois dias em disco
- ▶ Cópia inicial e *Archiving* replicados

Desafios diários

Estabilidade

- ▶ Manter *Throughput!!!*
- ▶ *Tuning* PostgreSQL
 - ▶ shared buffers
 - ▶ work_mem
 - ▶ nº conexões
 - ▶ *checkpoints*
 - ▶ bgwriter
 - ▶ *sync method*
 - ▶ **autovacuum**

Desafios diários

Estabilidade

- ▶ Manter *Throughput*!!!
- ▶ *Tuning* Debian
 - ▶ escalonador de I/O
 - ▶ patch *kernel*, kswapd
- ▶ *Tuning* Solaris
 - ▶ `max_queue_msg`
 - ▶ `force_direct_io` (UFS)
 - ▶ ARC *cache* (ZFS)



Desafios diários

Estabilidade

- ▶ Manter *Throughput*!!!
- ▶ *Tuning* Discos
 - ▶ *tablespaces* tabelas mais acessadas
 - ▶ *tablespaces* índices mais acessados
 - ▶ *array* exclusivo *pg_xlog*
- ▶ *Tuning* Replicação síncrona
 - ▶ concorrência no *storage* remoto

Desafios diários

Performance DW

- ▶ Ajuste de consultas DW
- ▶ Tabelas > 10 milhões de tuplas vivas
 - ▶ Particionamento
 - ▶ Função automática de particionamento
 - ▶ Colaboração com desenvolvedores
- ▶ Tabelas com dados XML
 - ▶ Colaboração com desenvolvedores
 - ▶ Tabelas indexadoras

Desafios diários

Manutenção

- ▶ **O ambiente é 24 x 7 !!!**
- ▶ **autovacuum**
- ▶ agendamento REINDEX
- ▶ agendamento *dump/restore*
 - ▶ (*VACUUM FULL + REINDEX ???*)
- ▶ “máquina de estado” → particionada
 - ▶ Colaboração com desenvolvedores
- ▶ Expurgos (local e aplicação)

Desafios diários

Paradigmas

- ▶ Realidade nos *datacenters* **não** é PostgreSQL (ainda)
- ▶ Transferência de conhecimento
- ▶ Limitações relativas
- ▶ Vantagens relativas
- ▶ Nenhum SGBD é igual ao outro
- ▶ Como “pensar PostgreSQL”

Após os desafios, elogios

- ▶ *performance*, alto TPS
- ▶ nunca se perderam dados
- ▶ documentação farta
- ▶ modelo livre
- ▶ **a resposta aos questionamentos é muito rápida**
- ▶ custos relativos muito baixos

Necessidades

Performance e disponibilidade

- ▶ *cluster multi-mestre shared disk/buffers*
 - ▶ é uma necessidade e uma realidade dos *datacenters*
- ▶ escalabilidade horizontal
 - ▶ *Clusters* multi-mestre?
 - ▶ paralelização, memcached transparente
 - ▶ *threading*?
- ▶ melhor administração via SQL
- ▶ mais catálogos
- ▶ auditoria

Multi-mestre na atualidade

Testes da 4Linux e Caixa

- ▶ pgpool-II e Sequoia
- ▶ não se adequaram a OLTP no ambiente
- ▶ PostgreSQL *stand-alone* – 750 TPS*
- ▶ pgpool-II com 2 PostgreSQL – 170 TPS*
- ▶ Sequoia com 2 PostgreSQL – 50 TPS*

* Transações Financeiras por Segundo, mínimo exigido pela Caixa é de 700 TPS

Necessidade final (e mais importante) Evolução das Ferramentas

- ▶ Ferramentas de monitoramento e administração integradas e gráficas
- ▶ Administração de toda a infra e *clusters*, integração
- ▶ Alertas de prevenção de incidentes, relatórios automáticos

Conclusão

- ▶ O PostgreSQL está pronto para a missão crítica
- ▶ mas precisamos evoluir
- ▶ nós sabemos evoluir
- ▶ **nós podemos evoluir**

Agradecimentos Especiais

- ▶ **a Euler Taveira de Oliveira**
 - ▶ pela participação ativa no projeto Multicanal
 - ▶ pelas contribuições ao PostgreSQL
 - ▶ pelas dicas e ensinamentos
- ▶ **a toda a comunidade PostgreSQL**
 - ▶ pelo excelente PostgreSQL
 - ▶ pela documentação
 - ▶ pela resposta rápida aos questionamentos

Obrigado!

▶ Flavio Henrique Araque Gurgel

flavio@4linux.com.br