documento em construção

Proposta de Arquitetura da Evolução do Módulo Batch

Contextualização

Pretende-se com essa wiki, apresentar e receber contribuições dos interessados quanto a descrição da organização do módulo Batch, estabelecendo as principais decisões de projeto com o objetivo de realizar os requisitos não funcionais.

Motivação

Por motivos históricos o Gsan permanece utilizando praticamente a mesma versão do servidor de aplicação JBoss (4.x). O JBoss 4.0.0 teve sua data de lançamento em setembro de 2004, e o último release 4.2.3, foi lançado em julho de 2008. Essa versão é bastante antiga e apresenta uma série de vulnerabilidades de segurança conhecidas e não corrigidas. Portanto, faz-se necessário e urgente uma atualização da versão do servidor de aplicação do Gsan. A versão mais recente é o Wildfly 9.0.2 Final de outubro de 2015.

EJB 2 é uma parte requerida do Java EE 7, e como o WildFly é Java EE 7 compatível ele dá suporte a tecnologia EJB 2, com exceção de um determinado tipo de bean, chamado entity-bean, cuja persistência é gerenciada pelo container e que agora tornou-se opcional. Como o sistema Gsan só apresenta EJB 2 do tipo session bean e message driven bean, é tecnicamente possível realizar o deployment do Gsan no Wildfly.

Objetivo

Este documento compreende as informações pertinentes a Arquitetura Candidata para a atualização técnológica do Módulo Batch do GSAN. O documento está divido em três principais seções: Contexto do Sistema, Requisitos Arquiteturais e Representação da Arquitetura.

O Contexto do Sistema tem a finalidade de descrever o contexto e o ambiente que o sistema em desenvolvimento esta inserido, é apresentado uma breve explanação em relação ao negócio e enumerados os usuários, canais de comunicação e sistemas externos que o módulo em desenvolvimento possui interação.

Na seção Requisitos Arquiteturais são enumerados os requisitos não-funcionais e requisitos funcionais relevantes do ponto de vista arquitetural. A Representação da Arquitetural foi dividida em Visão Conceitual, Visão dos Módulos, Visão da Execução e Visão do Código.

Para fechamento do documento temos duas seções complementares: Definições, acrônimos e

abreviações e as Referências.

Estratégia

Migrar a aplicação para permitir a execução no Wildfly com o mínimo de interferência nas regras de negócio da aplicação. Será utilizado um migrador automático com o objetivo de diminuir o tempo necessário e permitir a reprodutibilidade das alterações de forma a minimizar possíveis bugs. Serão abordados os principais elementos arquiteturais e as bibliotecas utilizadas pelo Gsan de forma que seja possível o funcionamento do Gsan no Wildfly.

Benefícios

- Atualização tecnológica e de segurança
- Aumento da performance
- Possibilidade de utilização de cluster
- Alta disponibilidade
- Balanceamento de carga
- Administração de forma centralizada (domain)
- Melhores ferramentas de administração e monitoramento

Contexto do Sistema

Negócio

GSAN, sistema proposto para atender a demanda comercial de médias e grandes empresas de saneamento básico, tem como fim a gestão do faturamento, pendências, arrecadação, cadastral e operacional.

O Gsan é um sistema Java EE construído ao longo de mais de uma década e que contém cerca de 1.500.000 (um milhão e quinhentas mil) linhas de código e conta também com mais de 500 (quinhentos) relatórios diferentes. O sistema controla uma variedade de sistemas e módulos satélites, e possui um ciclo mensal de gerenciamento do faturamento e financeiro. O processamento do faturamento ocorre mensalmente e é formado por um conjunto de atividades e de procedimentos, que visam a obtenção do volume e do valor da água fornecida e do esgoto coletado, bem como a cobrança de cada serviço indireto, até a emissão das contas. A partir desse momento é desdobrado as demais ações comerciais relacionadas a arrecadação, com a baixa dos arquivos bancários através do EDI. Processos de garantia de receita, com as rotinas de cobranças e negativação que representam ações sobre as pendências comerciais.

Os processamentos de grupos de faturamento, bem como todos as demais rotinas críticas de processamento são realizadas em lote, utilizando tecnologia EJB 2.0, utilizando o projeto Open Source Quartz como agendador.

O agendamento é feito manualmente pelo usuário através de funcionalidade específica do sistema. A organização das execuções deve considerar aspectos de negócio, como o processamento de rotinas de negativação e cobrança sempre após o processo de arrecadação. Como considerar a carga máxima suportada pelo sistema e o histórico de execução de determinado processo de forma que sua

execução não conflite com a operação online. Naturalmente essas considerações variam de caso a caso, conforme infraestrutura e dimensão de cada operação.

Representação da Arquitetura base do sistema

Figura 1 – Representação da Arquitetura

Usuários e Responsabilidades

Foram identificados os seguintes usuários (papéis) no sistema:

- Administrador;
  - Consiste no usuário com privilégios elevados no sistema que atua na área de TI estabelecendo as configurações e ajustes críticos do sistema.
- Operador;
  - Trata-se do usuário responsável pelo agendamento e execução dos batches, bem como, o acionamento do plantão correspondente em caso de falhas.
- Usuário Externo;
  - Pode tanto ser uma PA de cobrança contratada, um terceirizado que atua na gestão de serviços em campo como qualquer outro usuário que faça relacionamento com o sistema além da fronteira da empresa responsável
- Usuário de área de negócio
  - Trata-se do usuário chave da área de negócio, como Faturamento, Arrecadação, Cobrança e qualquer outro setor relevante na Companhia de Saneamento.

Diagrama de Contexto do Sistema

Figura 2 – Diagrama de Contexto do Sistema

Requisitos Arquiteturais

Em virtude do projeto ainda estar em uma fase incipiente, a lista abaixo não pode ser considerada completa. Os requisitos arquiteturais devem ser revisados após o término dos documentos de Visão e da elaboração dos Casos de Uso. Os requisitos arquiteturais também podem sofrer alterações com a prova de conceito da arquitetura.

- 1. **Usabilidade**
  - 1.1. Interface com usuário na Web;

    - 1.2. Integração através de transferência de arquivos e web services.

- 2. **Suporte a processamento Batch e Workflow**
    - 2.1. ETL (Extract Transform Load), executar extração, transformação e carga.
    - 2.2. Suporte as diferentes fontes de dados: banco de dados, ORM, JMS e arquivos XML, Flat File e CSV.
    - 2.3. Controle transacional do processo;
    - 2.4. Configuração de intervalos para “commits” para melhora de desempenho;
    - 2.5. Fácil configuração do processo através de XML, não sendo necessário re-compilar a aplicação;
    - 2.6. Reutilização de classes e serviços (regras de negócios) de aplicações transacionais;
    - 2.7. Processamento em lotes com características de Workflow;
    - 2.8. Suporte a aprovações humanas ou automáticas para continuidade do processo;
    - 2.9. Processamento condicional e seqüencial com etapas dependentes;
    - 2.10. Atribuição de propriedade de execução única a uma etapa do processo;
    - 2.11. Recuperação automática após falha e re-execução de um passo de forma automática ou manual, sem a necessidade de executar todo o processo novamente;
    - 2.12. Possibilidade de “escapar” caso uma exceção previamente cadastrada ocorra, dando continuidade no processo;
    - 2.13. Suporte a particionamento do processamento através de inicialização de várias threads para melhora do desempenho;
    - 2.14. Suporte a clusterização para melhora do desempenho;
    - 2.15. Administração através de interface web;
    - 2.16. Suporte a pausa e finalização dos processos;
    - 2.17. Auditoria e acompanhamento dos processos, incluindo histórico de execuções;
    - 2.18. Suporte a agendamento de processos;
    - 2.19. Suporte a gatilhos de processos através de eventos, exemplo: recebimento de mensagem ou arquivo.

- 3. **Segurança**
    - 3.1. Possibilidade de Auditoria;
    - 3.2. Autenticação na Web;
    - 3.3. Autenticação na execução de Web Services;
    - 3.4. Filtro para acesso a URLs;
    - 3.5. Filtro para execução de métodos.

- 4. **Comunicação**
    - 4.1. Requisições pela Web, HTTP
    - 4.2. Requisições e envio por Web Services, REST.
    - 4.3. Requisições e envio por Arquivos, FTP.

- 5. **Persistência e controle transacional de Banco de Dados, Arquivos e Filas.**

- 6. **Integração com Sistemas Externos**
    - 6.1. Integração com sistemas externos a organização
    - 6.2. Integração através de troca de Arquivos
    - 6.3. Suporte aos formatos XML e Flat File
    - 6.4. Suporte a tipos pré-definidos de XML (XSD)
    - 6.5. Suporte a Web Services;

Diagrama Esquemático (Cluster)

Cluster de servidores de aplicação visam a aumentar a escalabilidade e disponibilidade. De forma a obter estes benefícios, faz-se necessário gerenciar a configuração de uma série de serviços e componentes como por exemplo a camada de transporte, a replicação/distribuição dos dados entre os nós do cluster, como também todo o esquema de balanceamento de cargas entre os nós.

O maior desafio na configuração de um cluster é achar as condições ótimas de escalabilidade e disponibilidade.

Disponibilidade se refere ao fato de caso um nó falhe, outro nó assumirá a função. Escalabilidade representa a fato de aumentar o poder computacional a medida que novos nós são adicionados ao cluster. Estas duas propriedades se bem equilibradas podem promover um aumento considerável da performance das rotinas batch e também do módulo online.

Figura 3 – Diagrama de Clusterização

Representação da Arquitetura

Referências

Java EE Technologies http://www.oracle.com/technetwork/java/javaee/tech/index-jsp-142185.html

Spring Integration http://www.springsource.org/spring-integration

Spring Batch http://www.springsource.org/spring-batch

Core J2EE Patterns http://www.corej2eepatterns.com

Patterns and Best Practices for Enterprise Integration http://www.eaipatterns.com

Domain-driven design http://en.wikipedia.org/wiki/Domain-driven_design

Documento em desenvolvimento...