

# Um Estudo sobre Implantação de Sistemas Distribuídos Baseados em Componentes de Software

Amadeu Andrade Barbosa Júnior <ajunior@inf.puc-rio.br>

**INF2556 – Seminários de Sistemas Distribuídos**

Departamento de Informática  
Pontifícia Universidade Católica do Rio de Janeiro

# Agenda

- ◆ Objetivos
- ◆ Critérios de avaliação
- ◆ Trabalhos estudados
- ◆ Comparação das abordagens
- ◆ Conclusões

# Objetivos

- ◆ Identificar **desafios e ferramentas** para implantação de sistemas distribuídos baseados em componentes
- ◆ Definir um critério para avaliação das diferentes ferramentas
- ◆ Buscar uma estratégia que possa:
  - ◆ Implantar componentes distribuídos gerenciando dependências externas ao componente numa abordagem multi-plataforma e multi-linguagem
  - ◆ Permitir maior dinamicidade na implantação, como re-implantação, reconfiguração e adaptação

# Critérios para avaliação

- ◆ **Modelo de composição e distribuição**
  - ◆ quais os recursos de composição estão disponíveis e como são respeitados na arquitetura distribuída ?
  - ◆ como obter remotamente os componentes ?
- ◆ **Abrangência tecnológica**
  - ◆ o que pode ser implantado ?
  - ◆ como ter acesso ao nó remoto para controlar a implantação dos componentes ?
- ◆ **Tratamento de dependências**
  - ◆ como o modelo gerencia dependências entre componentes e recursos externos ?
- ◆ **Flexibilidade da implantação**
  - ◆ possível monitorar, reconfigurar, adaptar, re-implantar, co-implantar ?

# Trabalhos estudados

## 2008.1

- ◆ Middlewares de componentes
  - ◆ EJB
  - ◆ OpenCom
  - ◆ Fractal
  - ◆ CCM – único a definir implantação distribuída
- ◆ Ferramentas de implantação para grades
  - ◆ ADAGE
  - ◆ SmartFrog

## 2008.2

- ◆ CoRDAGE – implantação para grades
- ◆ DeployWare – implantação para grades, totalmente implementada em Fractal
- ◆ DaNCE – diretamente ligado ao CCM

# Modelo de composição e distribuição

**repositório** = possui ou não repositório de componentes com a capacidade de obtê-los desse

**composição** = quando pode se dar a composição

**distribuição** = dado um conjunto de nós remotos, como é feita a alocação nesses *hosts*

	<b>repositório</b>	<b>composição</b>	<b>distribuição</b>
<b>ADAGE</b>	n/a	n/a	automática
<b>SmartFrog</b>	n/a	n/a	manual
<b>CCM + DaNCE</b>	sim	fase de projeto	assistente
<b>EJB</b>	sim	projeto e implantação	manual
<b>OpenCom</b>	n/a	fase de projeto	manual
<b>Fractal + DeployWare</b>	n/a	fase de projeto	automática

# Abrangência tecnológica

**o quê?** = quais softwares podem ser implantados

**acesso remoto** = como é concretizado o acesso remoto aos nós de execução/implantação

**linguagem** = componentes programados em qual linguagem

	<b>implanta o quê?</b>	<b>acesso remoto</b>	<b>linguagem</b>
<b>ADAGE</b>	<i>quaisquer*</i>	<i>job submission</i>	<i>quaisquer</i>
<b>SmartFrog</b>	<i>quaisquer</i>	<i>daemon próprio</i>	java
<b>CCM + DaNCE</b>	ccm	<i>app server</i>	<i>corba 2.x</i>
<b>EJB</b>	javabeans	<i>app server</i>	java
<b>OpenCom</b>	opencom	<i>app server</i>	<i>corba 2.x</i>
<b>Fractal + DeployWare</b>	<i>quaisquer**</i>	<i>daemon próprio</i>	java

\* - demanda definir XML com tratadores para cada tipo de tecnologia sendo implantada

\*\* - demanda definir **componentes fractal** para mapear dependências e tarefas a executar

# Tratamento de dependências

**internas** = se existe a noção de dependência entre componentes para a implantação

**externas** = se é possível informar que se depende de softwares externos para implantação

	<b>internas</b>	<b>externas</b>
<b>ADAGE</b>	sim, descritores	não
<b>SmartFrog</b>	sim, descritores	não
<b>CCM + DaNCE</b>	sim, portas	não
<b>EJB</b>	não	não
<b>OpenCom</b>	sim, portas	não
<b>Fractal + DeployWare</b>	sim, portas	não*

\* - dado o **FDF** já disponibiliza componentes wrappers, é possível especificar que um componente **X** depende de um outro Y onde Y é um wrapper

# Flexibilidade para implantação

- ◆ Monitoramento:
  - ◆ DeployWare, ADAGE, EJB (jmx), CCM (sink/sources)
- ◆ Re-implantação:
  - ◆ CoRDAGE
- ◆ Reconfiguração:
  - ◆ ??

# Conclusões

- ◆ DeployWare parece mais interessante
  - ◆ Prós:
    - ◆ é feito em componentes Fractal
    - ◆ Instala componentes Fractal
    - ◆ FDF facilita a criação de componentes que encapsulem softwares externos
    - ◆ Mapeamento automático na grade
  - ◆ Contras:
    - ◆ Novos componentes Fractal só podem ser programados em Java
    - ◆ Não endereça conceitos de re-implantação, reconfiguração
    - ◆ Encapsular softwares externos dependentes de plataforma em componentes obriga tratar multi-plataforma

# Conclusões (2)

- ◆ DaNCE, também muito interessante, segue a abordagem arquitetural D&C da OMG para implantação.
  - ◆ Prós:
    - ◆ Melhor organização das entidades e contratos envolvidos na implantação: contêiners, nós de execução, domínios de aplicações, repositórios
    - ◆ Flexível para implantar componentes em diferentes linguagens
  - ◆ Contras:
    - ◆ Falta dinamicidade para a implantação
    - ◆ Especificação D&C não informa como suporta re-implantação ou reconfiguração
    - ◆ Não trata dependências externas – poderiam ser encapsuladas como em Fractal :-)

# Conclusões (3)

- ◆ Questões pessoais:
  - ◆ Estar associado com um sistema de pacotes que entenda dependências com softwares externos multi-plataforma e multi-linguagem
  - ◆ Mas qual a vantagem disso ao invés de encapsular tudo em componentes ? Até que ponto definir componentes para tudo faz sentido ?
- ◆ Após observação dessas tecnologias, fundamental:
  - ◆ Endereçar conceitos de re-implantação e reconfiguração dos componentes
  - ◆ Ter estratégias de mapeamento automático para distribuir nos *hosts* disponíveis