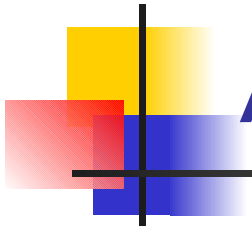
Reconfiguração Dinâmica de Sistemas Baseados em Componentes

Ricardo Koji Ushizaki

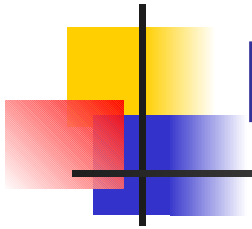
Orientador: Prof. Dr Fabio Kon

Fevereiro/2005



Agenda

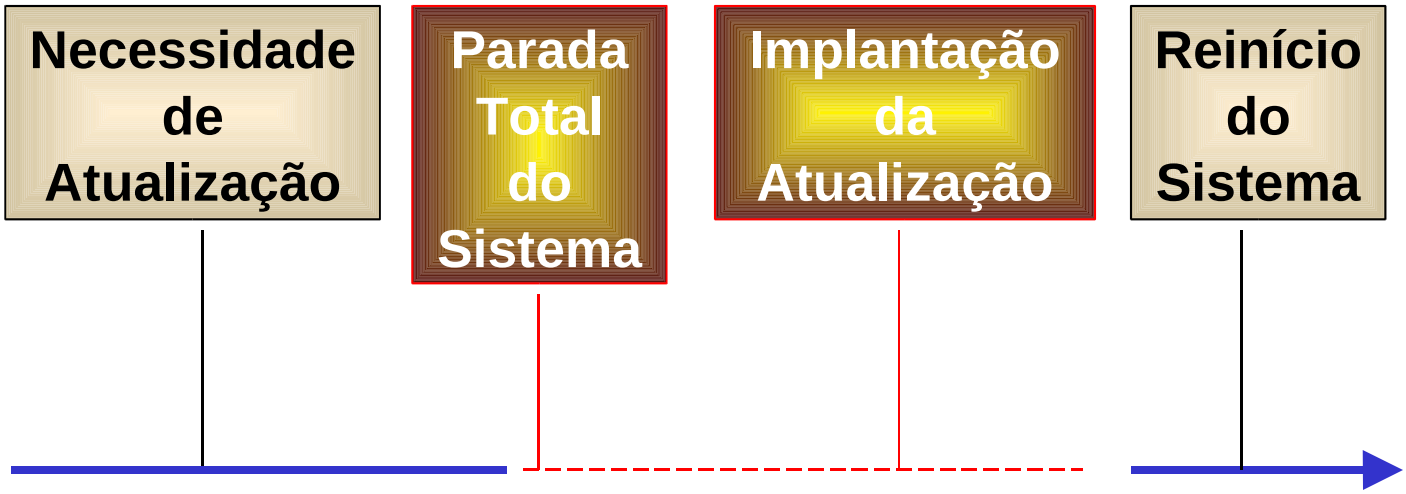
- Reconfiguração Dinâmica
  - Definição e Motivação
  - Etapas e Requisitos
  - Mecanismos existentes
- Trabalhos Relacionados na Área

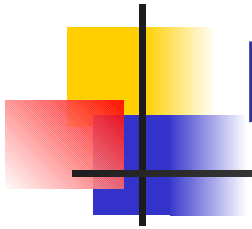


Reconfiguração Dinâmica

- Sistemas computacionais em constante evolução
- Programação orientada a objetos
- Baseada em componentes
- Evolução requer atualizações do sistema

Cenário Típico

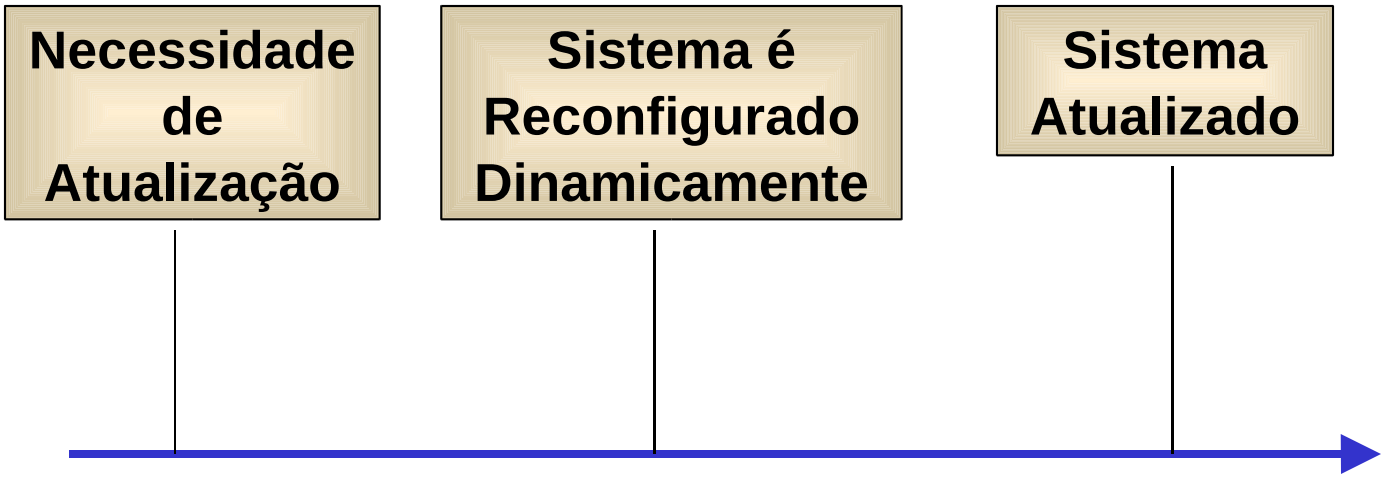


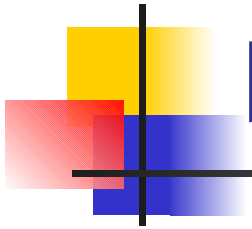


Problemas

- Paradas do Sistema significam:
  - Prejuízos financeiros para empresas;
  - Vidas em perigo em sistemas críticos (hospitais).
- É necessário atualizar dinamicamente os sistemas

Cenário com Reconfiguração Dinâmica

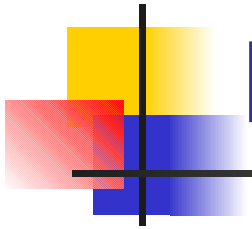




Histórico

- Primeiros trabalhos de reconfiguração dinâmica surgiram na década de 80
- Formalizaram técnicas e identificaram obstáculos para reconfiguração dinâmica

Objetivos da Reconfiguração Dinâmica



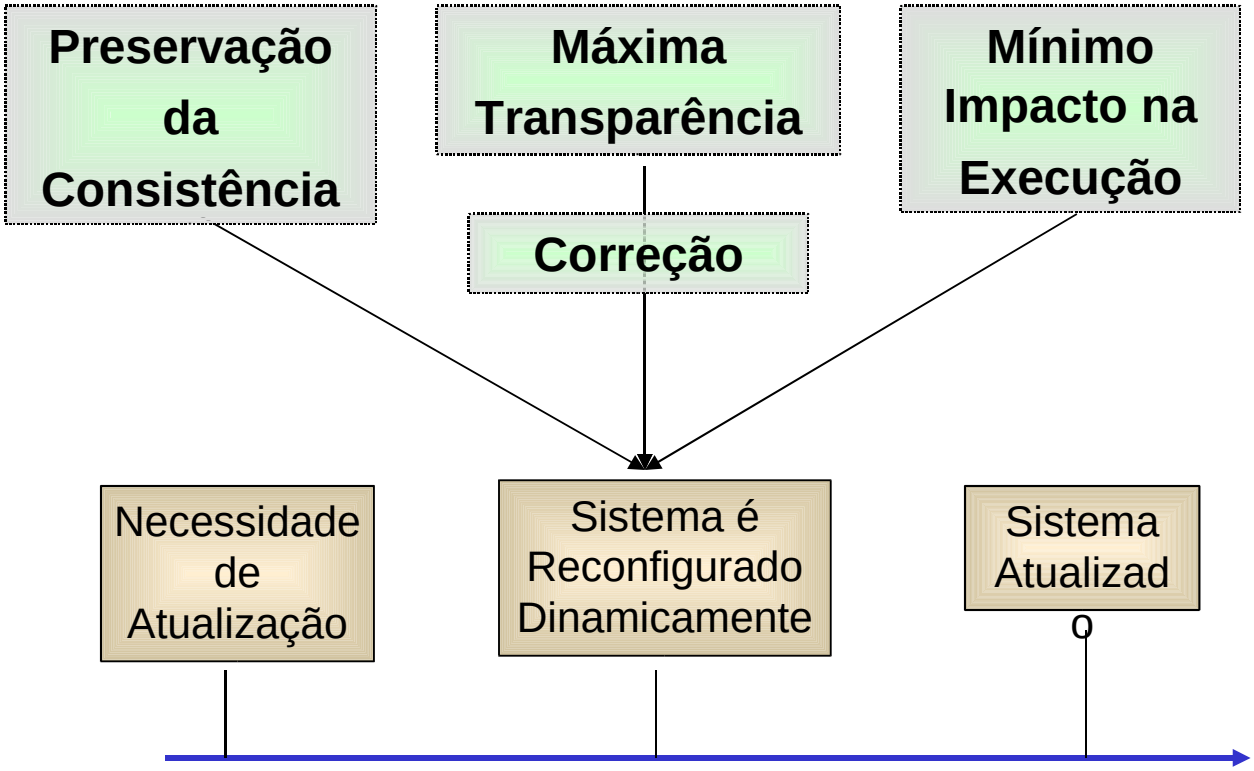
- Permitir a evolução do sistema em tempo de execução

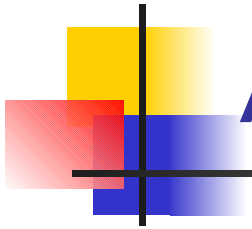
- Pouco ou nenhum impacto negativo no seu desempenho

Operações da Reconfiguração Dinâmica

- Adição e Remoção
  - Adicionar novo componente
  - Remover componente
- Substituição
  - Substituir implementação
- Migração
  - Mover componente

Requisitos

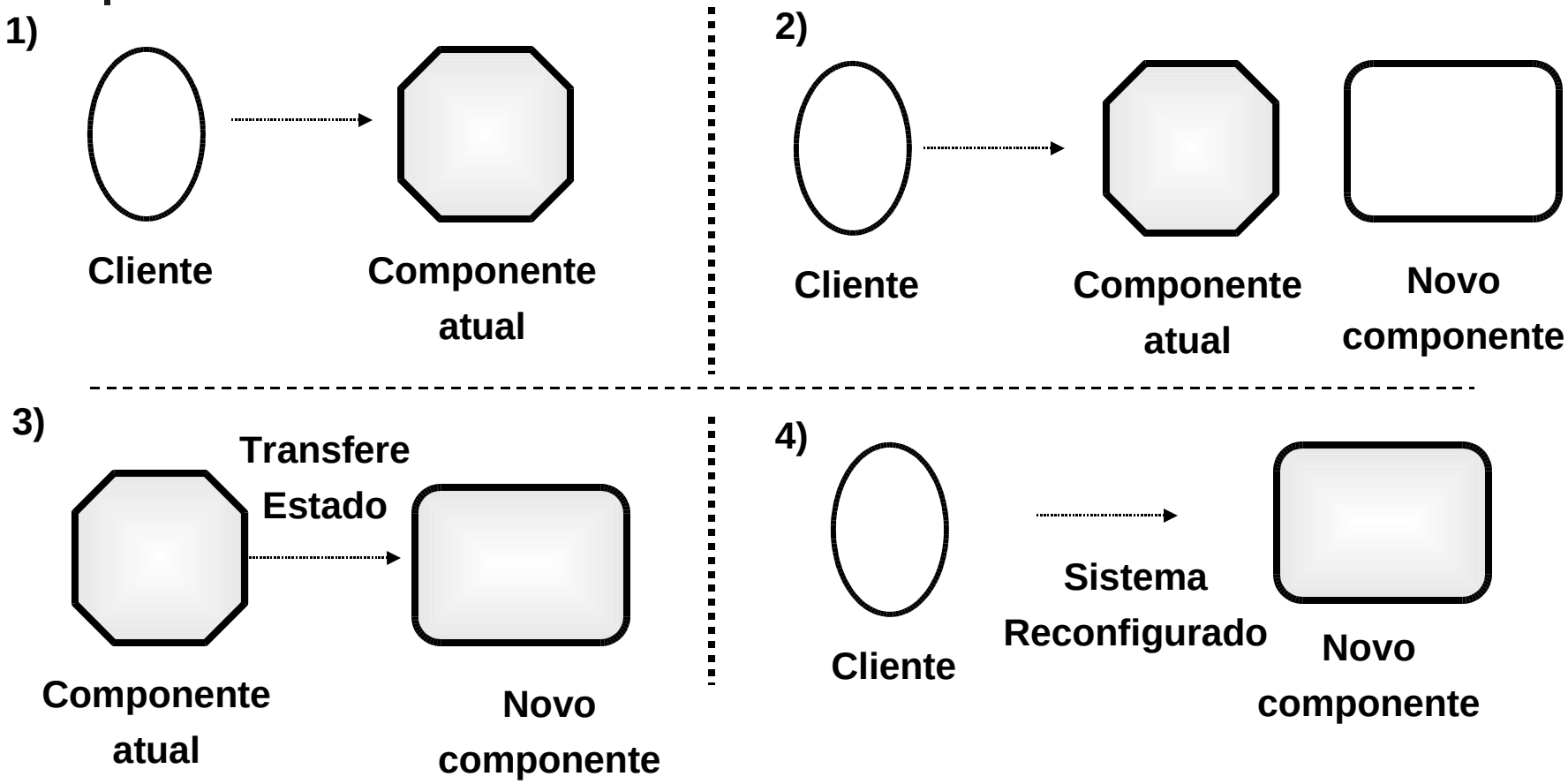

Preservação da Consistência
Máxima Transparência
Mínimo Impacto na Execução
Correção
Necessidade de Atualização
Sistema é Reconfigurado Dinamicamente
Sistema Atualizado

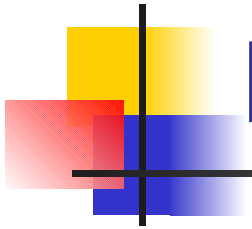


Aspectos Importantes

- Aplicações possuem estado
  - Deve ser mantido correto e consistente
- Garantir a Transferência de Estado
  - Estado deve ser migrado da implementação atual para a nova a ser instalada
  - Transferência de Estado não deve corromper o sistema

Exemplo

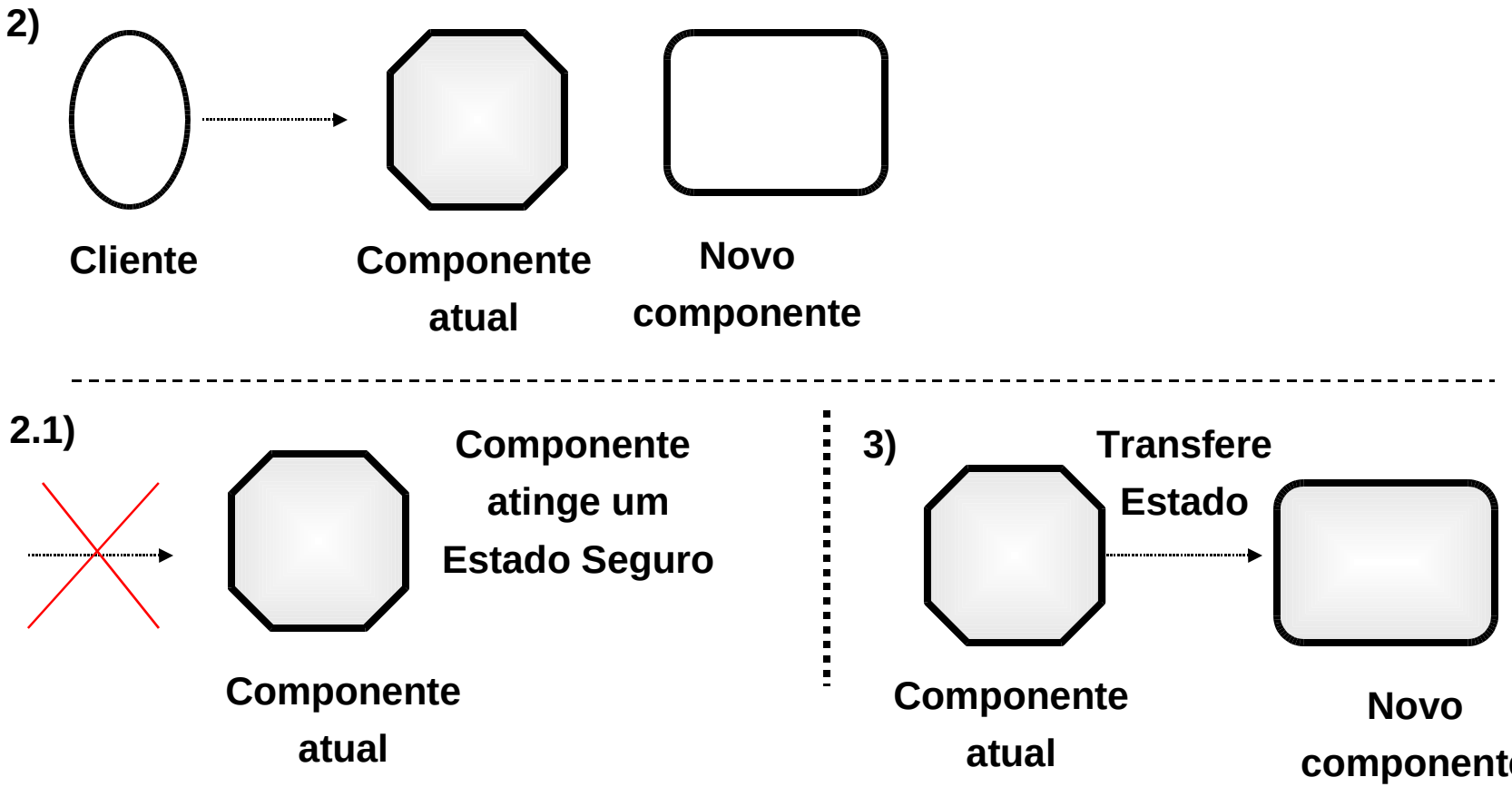

1)
Cliente
Componente atual
2)
Cliente
Componente atual
Novo componente
3)
Transfere Estado
Componente atual
Novo componente
4)
Cliente
Sistema Reconfigurado
Novo componente

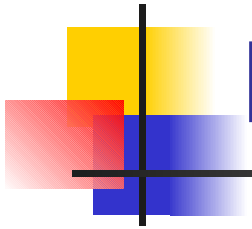


Estado Seguro

- É necessário preservar a consistência do sistema após a reconfiguração
- Para ocorrer a transferência de estado:
  - Componentes devem estar em um **estado seguro**
  - Garantir que permaneçam nesse estado durante a reconfiguração

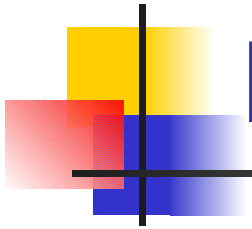
Atingindo Estado Seguro





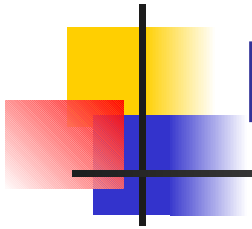
Preservar Consistência

- Sistema deve estar em um estado correto após a reconfiguração
- Estado correto:
  - Integridade estrutural preservada;
  - Partes afetadas continuam interagindo com sucesso;
  - Invariantes do estado preservadas.



Mecanismos

- Foram propostos diversos mecanismos para a reconfiguração dinâmica:
  - Mecanismo de Bloqueio de Chamadas
  - Mecanismo de Indireção de Chamadas
  - Mecanismo de Pontos de Reconfiguração

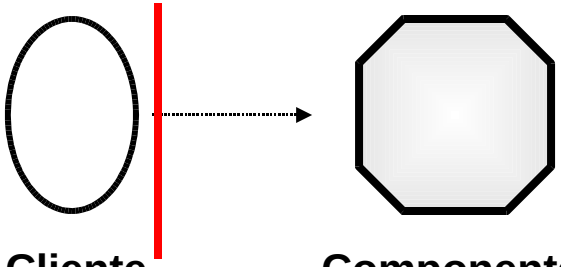


Bloqueio de Chamadas

- Para o componente chegar a um estado seguro:
  - bloquear chamadas a métodos que alterem seu estado
  - Ou abortar chamadas a esses métodos (por exemplo, via *Java Exceptions*)

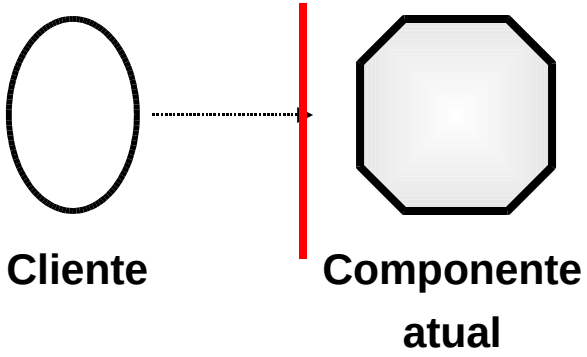
Exemplos de bloqueios

1- Bloquear no lado cliente



- **Mais simples de implementar**
- **Muito intrusiva**
- **Nem sempre podemos alterar cliente**

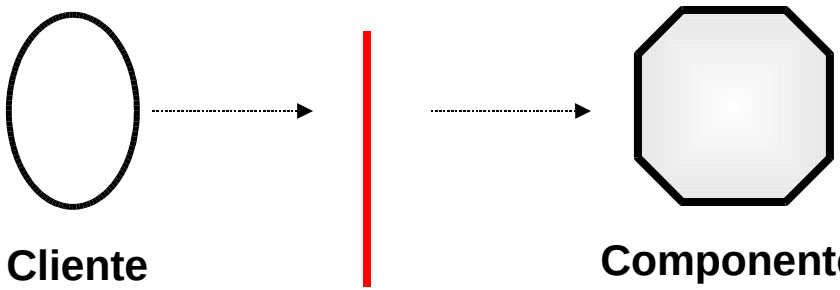
2- Bloquear no lado do componente



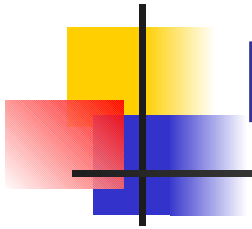
- **Transparente ao cliente**
- **Manipular variáveis locais**

Exemplos de bloqueios

3- Bloquear entre o cliente e o componente



- **Mecanismo de Indireção De Chamadas**

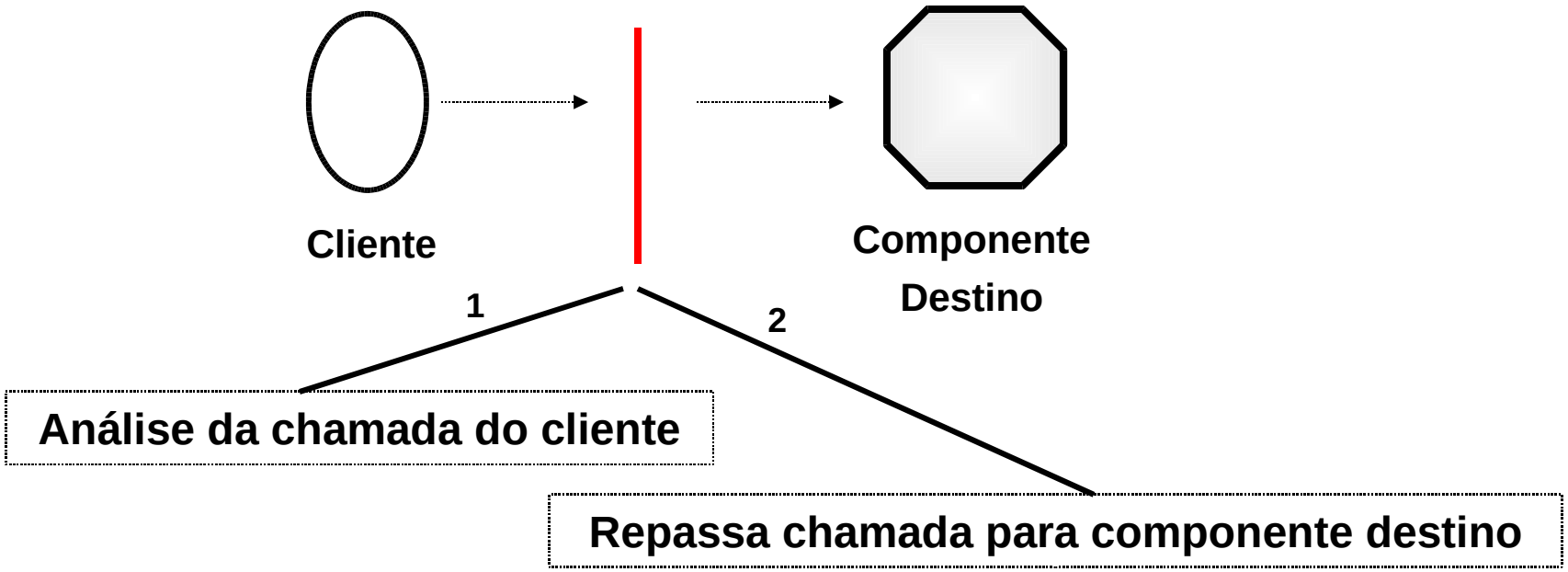


Mecanismo de Indireção

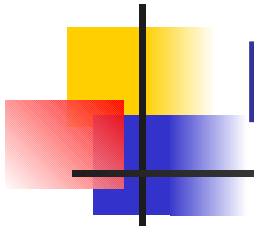
- Utiliza mecanismos do *middleware*
  - CORBA, EJB, .NET
- Redireciona chamadas do cliente para componente
- Existe um serviço intermediário que intercepta toda chamada

Exemplo de indireção

Serviço intermediário intercepta chamada



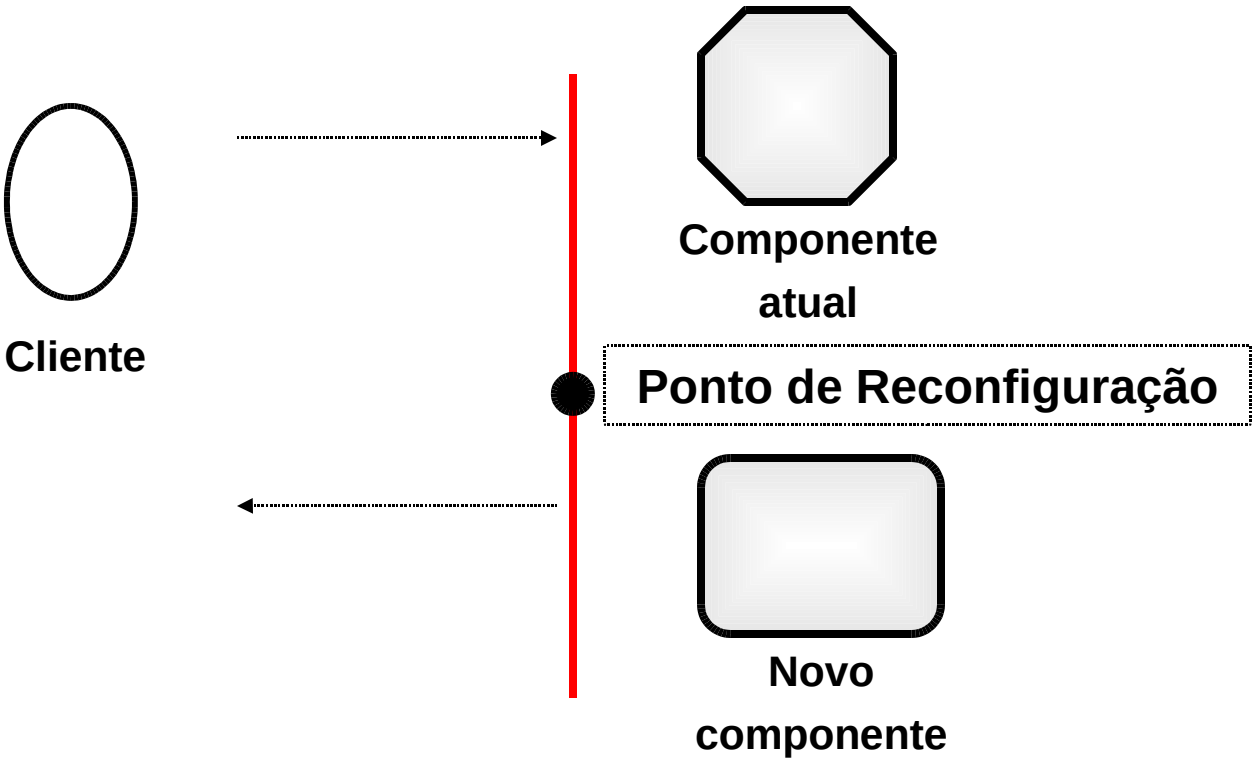
Mecanismo de Pontos de Reconfiguração



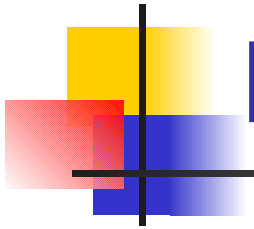
- Pontos no código seguros para se realizar a reconfiguração
- Não é preciso esperar que a chamada termine para iniciar a reconfiguração
  - Execução do método é interrompida no ponto de reconfiguração
  - Cliente inicia chamada em um componente e finaliza no novo componente

Exemplo de Ponto de Reconfiguração

2- Bloquear no lado do componente

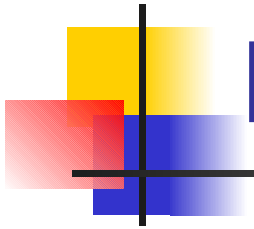


Vantagens de Pontos de Reconfiguração



- Vantagens:
  - Atinge estado seguro durante invocação do método
  - Métodos de longa execução não atrasam a reconfiguração

Desvantagens de Pontos de Reconfiguração



- Desvantagens:
  - Preservar estado do componente e do método invocado
  - Difícil de implementar:
    - nova implementação deve compartilhar o mesmo comportamento após o ponto de reconfiguração
    - Caso contrário, não é possível mapear contexto

Trabalhos Relacionados - Argus

- Substituição Dinâmica no Argus
  - MIT - 1983 - Toby Bloom
  - Formalizou um modelo de substituição dinâmica utilizando Argus
  - Argus é um sistema distribuído composto por módulos chamados de guardiões
  - Definiu mecanismo para substituir manualmente os guardiões

Trabalhos Relacionados - Conic

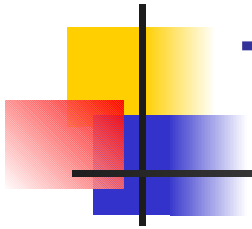
- Configuração Dinâmica com Conic
  - Imperial College – 1985 – Jeff e Magee
  - Conic:
    - ambiente de programação
    - ferramentas para compilar, configurar, depurar e executar programas
  - Como configurar dinamicamente sistema em Conic
  - Definem comandos: *link, unlink, create* e um *ConfigurationManager* para reconfiguração
  - Termo *quiescence*: componente em modo passivo

Trabalhos Relacionados - POLYLITH

- POLYLITH
  - É um sistema distribuído para ambientes heterogêneos
  - Universidade de Maryland – 1990
  - Purtillo e Hofmeister incluem suporte para reconfiguração dinâmica no POLYLITH
  - Estenderam o trabalho de Jeff e Magee
  - Utiliza pontos de reconfiguração para capturar e restaurar estado

Trabalhos Relacionados - Bidan

- Serviço de Reconfiguração Dinâmica para CORBA
  - Bidan em 1998 estendeu a semântica do serviço CORBA de ciclo de vida
  - Criou *Dynamic Reconfiguration Manager:*
    - Interage com componentes
    - Operação *passivateLink*
    - Interface RO_Object



Trabalhos Relacionados

- Reconfiguração Dinâmica de Aplicação Java Baseada em Componentes
  - MIT – 2000 – Ziqiang Tang
  - Define modelo utilizando pontos de reconfiguração
  - Define novas palavras-chaves em Java:
    - *component, decode, encode, fulfills, reconfigurables, reconfigurable*

Trabalhos Relacionados

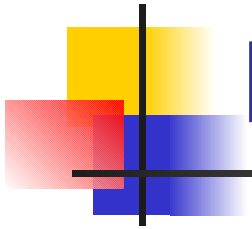
- Reconfiguração Dinâmica Transparente para CORBA
  - Almeida et al – 2001
  - Estendem o trabalho de Bidan adicionando suporte para:
    - Chamadas re-entrantes
    - Substituições atômicas de múltiplos objetos
    - Maior transparência no ORB do CORBA
  - Define arquitetura de componentes:
    - *Reconfiguration Manager, Location Agent e Reconfiguration Agents*

Trabalhos Relacionados

- Sistema Operacional *2K*
  - Universidade de Illinois – 2000 – Kon et al
  - Criaram sistema operacional distribuído baseado em componentes
  - Define um Serviço de Configuração Automática para:
    - Gerenciar pré-requisitos
    - Gerenciar dependências dinâmicas entre componentes

Trabalhos Relacionados

- Adaptação Dinâmica de Sistemas Distribuídos
  - IME USP – 2003 – Francisco Silva e Silva
  - Criou arquitetura para adaptação dinâmica
    - Módulos de monitoração do sistema
    - Detecção e análise de eventos
    - Verificação de pré-condições e aplicar reconfiguração dinâmica



Fim

- Dúvidas
- OBRIGADO!