UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

INTRODUÇÃO AOS SISTEMAS DISTRIBUÍDOS

Prof. Cesar Augusto Tacla
http://www.dainf.ct.utfpr.edu.br/~tacla/EspSD/

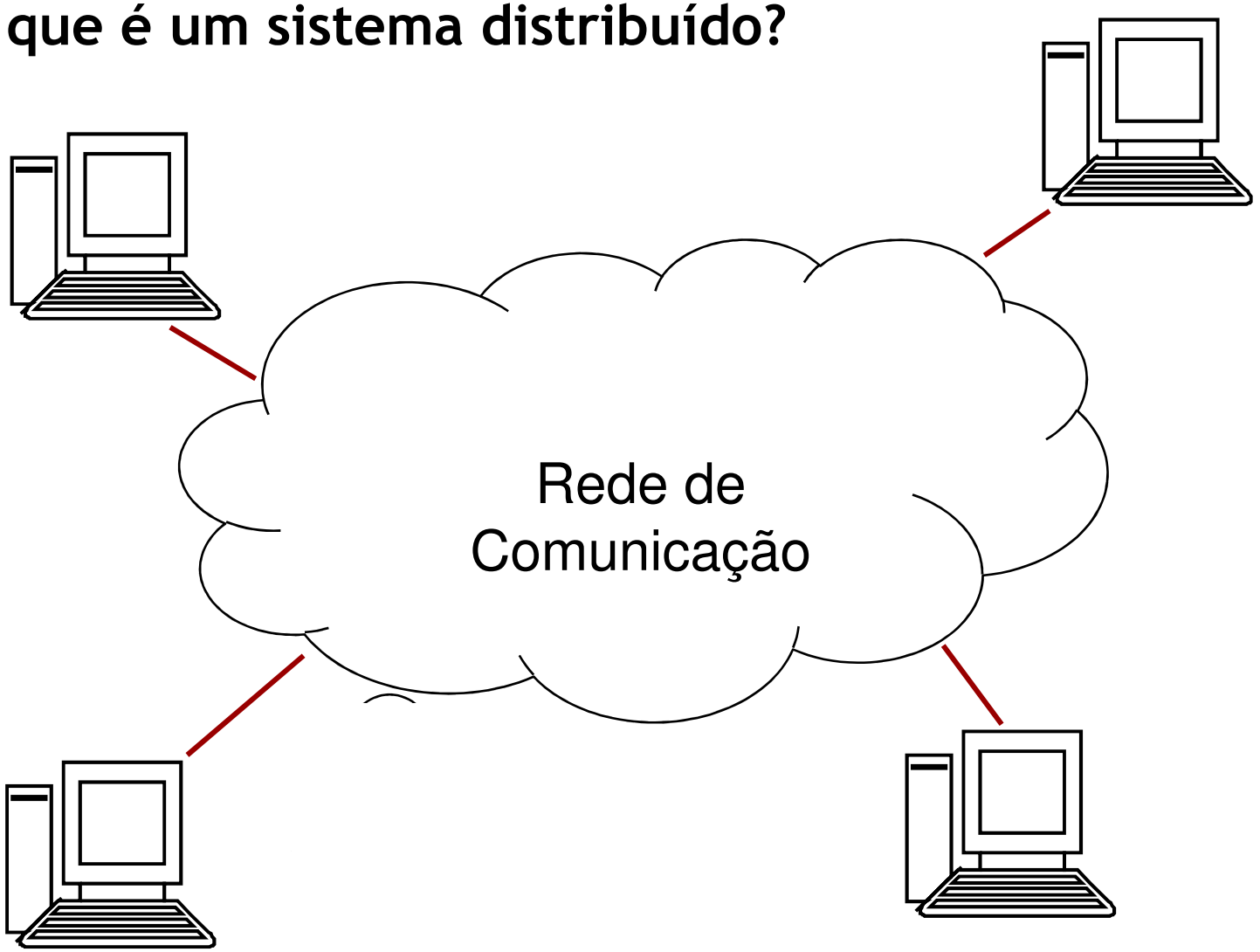
Sumário

1. **INTRODUÇÃO**
   a. Definição de sistemas distribuídos (SDs)
   b. Exemplos de sistemas distribuídos
   c. Tipos de aplicações distribuídas
   d. Vantagens e dificuldades dos SDs

1 a

Definição de sistemas distribuídos

- O que é um sistema distribuído?



Definição

- Um sistema distribuído é uma coleção de computadores autônomos conectados por uma rede de comunicação que é **percebida pelos usuários como um único computador** que provê um serviço ou resolve um problema.

- (Tanenbaum & Steen, 2002)

Esta definição faz referência a um conceito importante em sistemas distribuídos, o da transparência

Definição

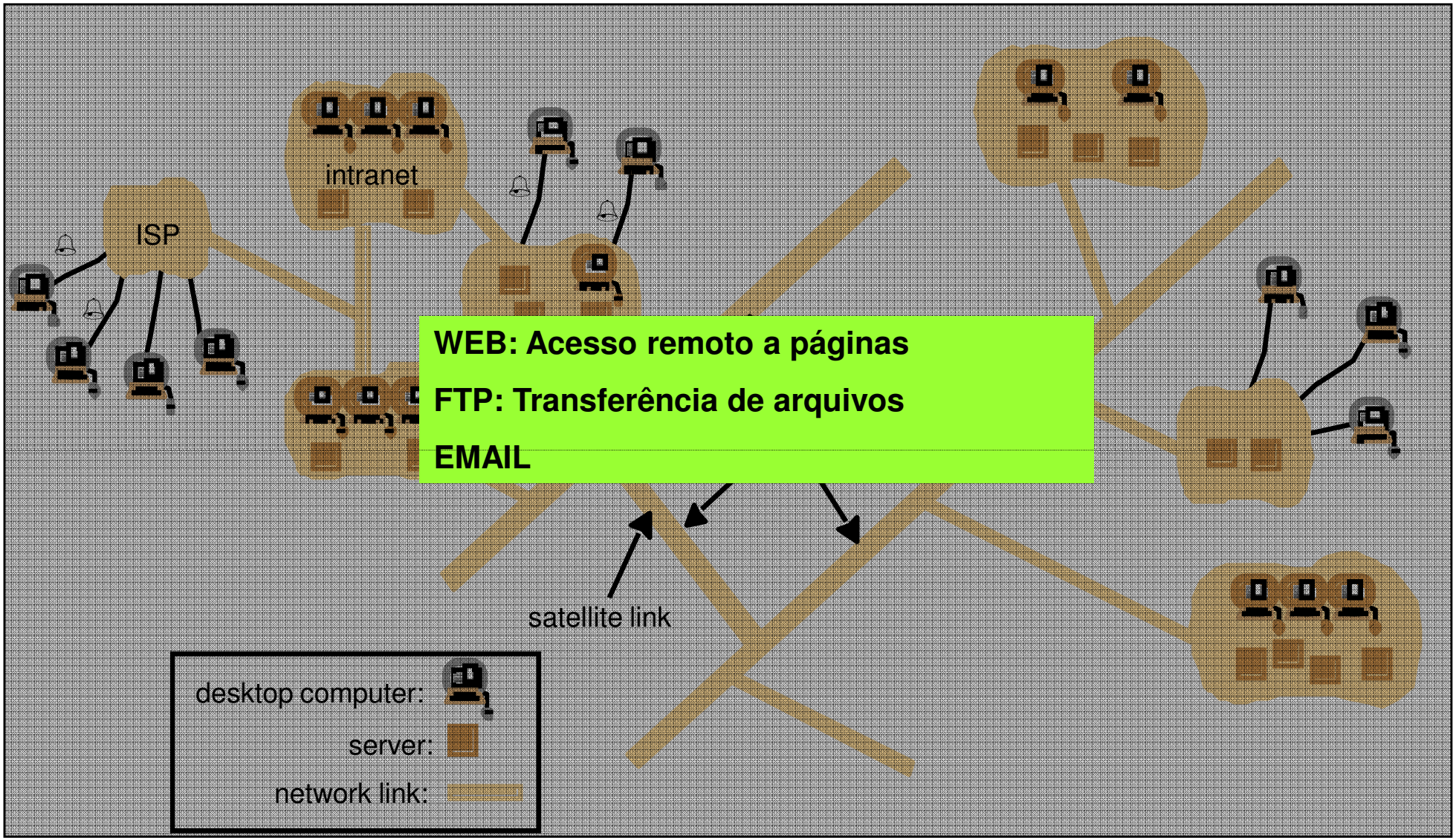
- Um sistema distribuído é composto por computadores conectados em rede (hardware e software) que se comunicam e coordenam suas ações somente **através do envio de mensagens**.

- (Coulouris et al., 2001)

1 b

Exemplos de sistemas distribuídos

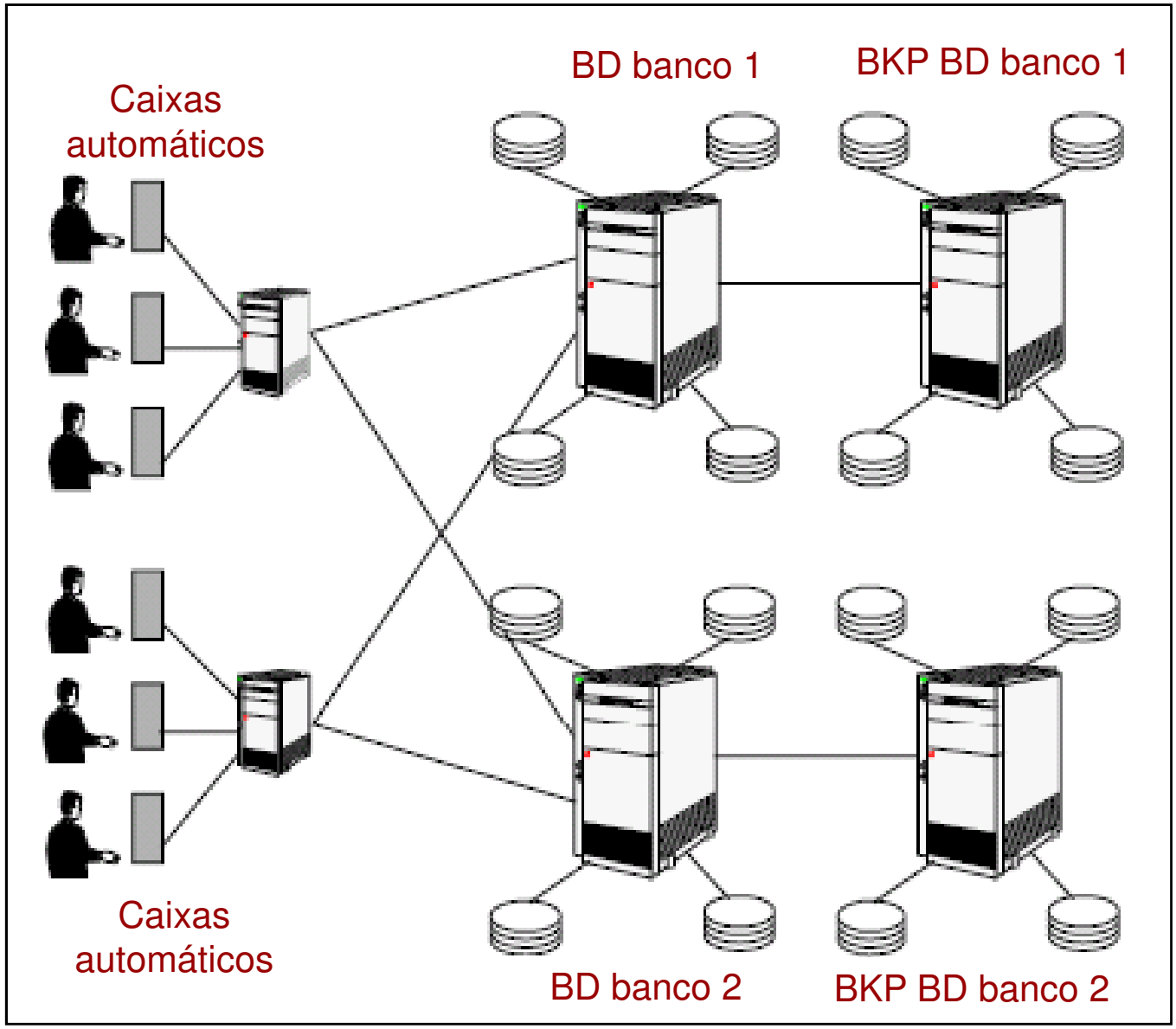
Exemplo 1: WEB



Fonte: figura 1-1 (Coulouris et al., 2003)

ISP: Internet Service Provider = provedor de acesso

Exemplo 2: Bancos

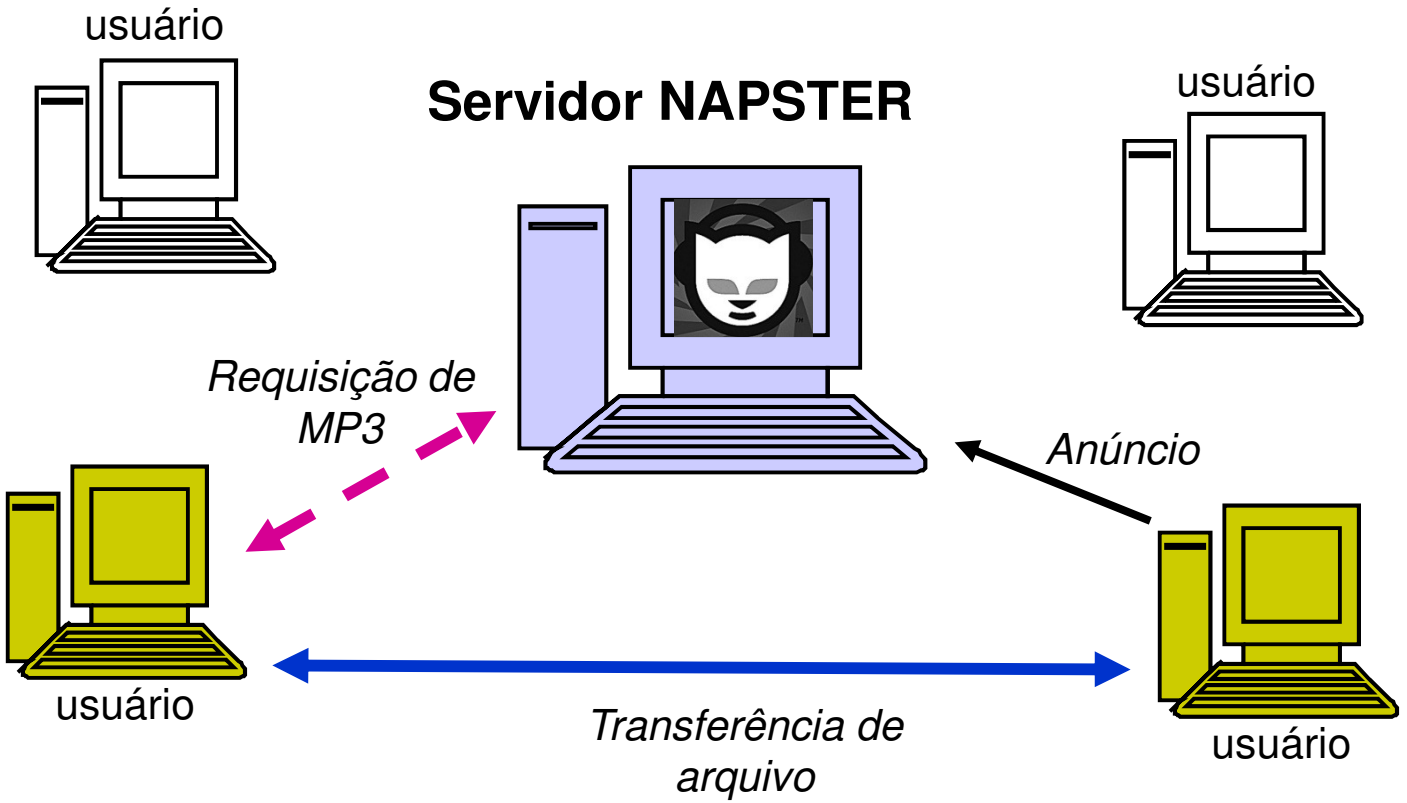


BD = banco de dados

BKP = backup

Fonte: http://www.ida.liu.se/~TDDB37/lecture-notes/lect1.frm.pdf

Exemplo 3: Napster



- **Similares ao NAPSTER**
  - EMULE
  - BITORRENT
  - SKYPE (VoIP)
  - MSN MESSENGER, YAHOO
- **Outros**
  - APLICAÇÕES DE MULTIMÍDIA
    - WEB conferência
    - Difusão de streams on-line (voz, vídeo)

- **CLUSTERS ou AGRUPAMENTOS**

- **Máquinas conectadas por rede de alta velocidade**
- **Necessidade de alto desempenho computacional**
- **Evitar custo de máquinas de alto desempenho**
- **Objetivo: compartilhar recursos computacionais**
  - De processamento
  - De armazenamento
  - De memória
  - Outros

Exemplo 4: Cluster (2)

JAGUAR
Cluster de máquinas do Oak Ridge National Laboratory (EUA)

Primeiro da lista dos TOP 500 SUPERCOMPUTERS

INSTALADO EM 2009
224.162 CORES

1.759.000 Gflops

FONTE http://www.top500.org/list/2010/06/100

Acesso em 9/9/2010

Solução: agrupamento físico de máquinas de menor porte = cluster:

- mesmo espaço geográfico
- Normalmente dentro da mesma organização, com software/hardware homogêneos
- Softwares: OpenMosix, Condor e Oscar

Exemplo 4: cluster (3)

- Clusters são utilizados para...

- **Aumentar a disponibilidade de serviço**
  - se um nodo falha, outro assume

- **Equilibrar carga de trabalho**
  - um ou mais computadores do cluster atuam como distribuidores da carga entre os demais

- **Alto desempenho**
  - Para resolver tarefas complexas que podem ser decompostas em sub-tarefas, cada uma rodando num nodo do cluster.
  - Implementação mais comum: LINUX e software livre para implementar paralelismo = **Beowulf cluster**

Exemplo 5: Grid (1)

- **GRID ou GRADE = as máquinas encontram-se distantes geograficamente.**

- **É uma cooperativa de máquinas**
  - não pertencem necessariamente a mesma organização!
  - são ditas organizações virtuais
  - não há administração central dos recursos computacionais
  - Utilização de protocolos/padrões abertos

Aplicações

- Tarefas complexas decompostas
- Reunião dos resultados parciais

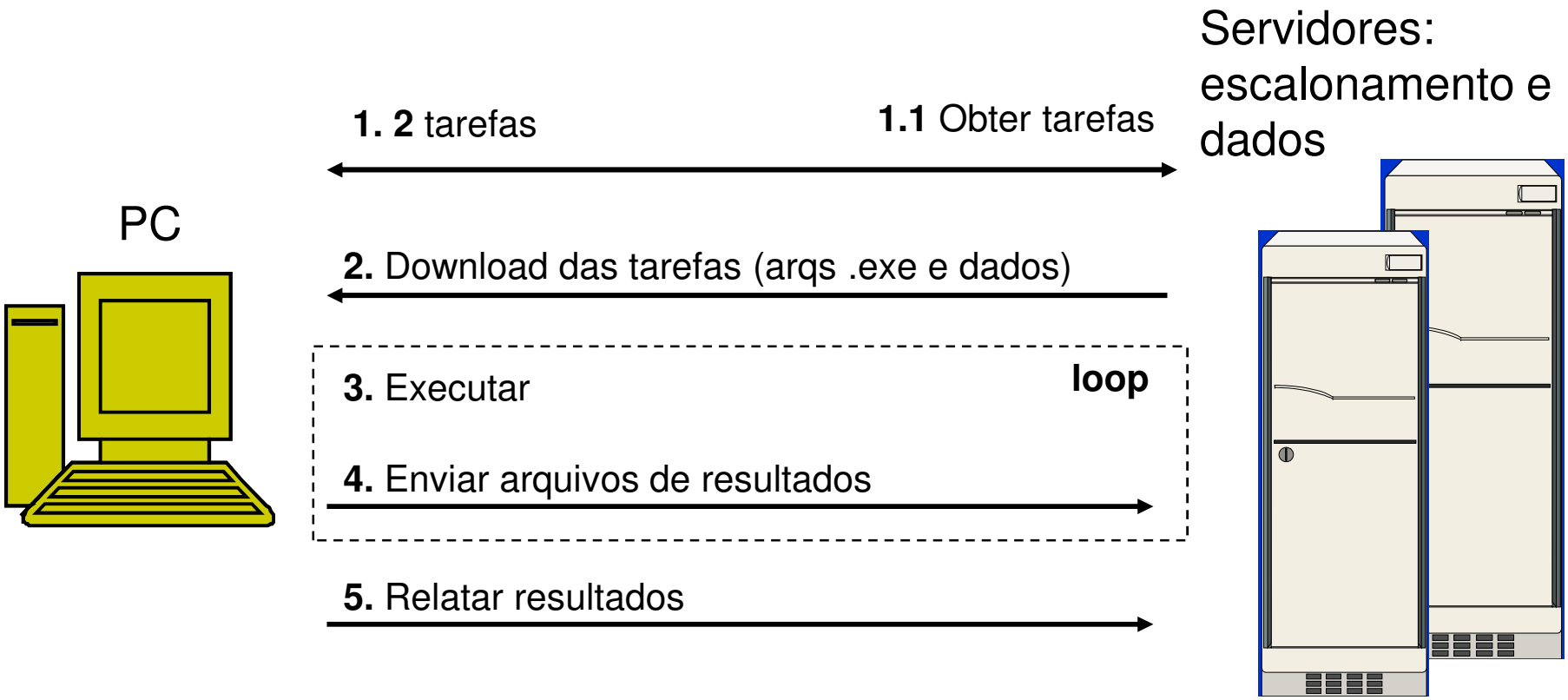
Exemplos de grids de pesquisa

- Laboratório Nacional de Computação Científica: http://portalvcg.lncc.br/
- Campina Grande: http://ourgrid.org

◇ Exemplos – participação de qualquer usuário final

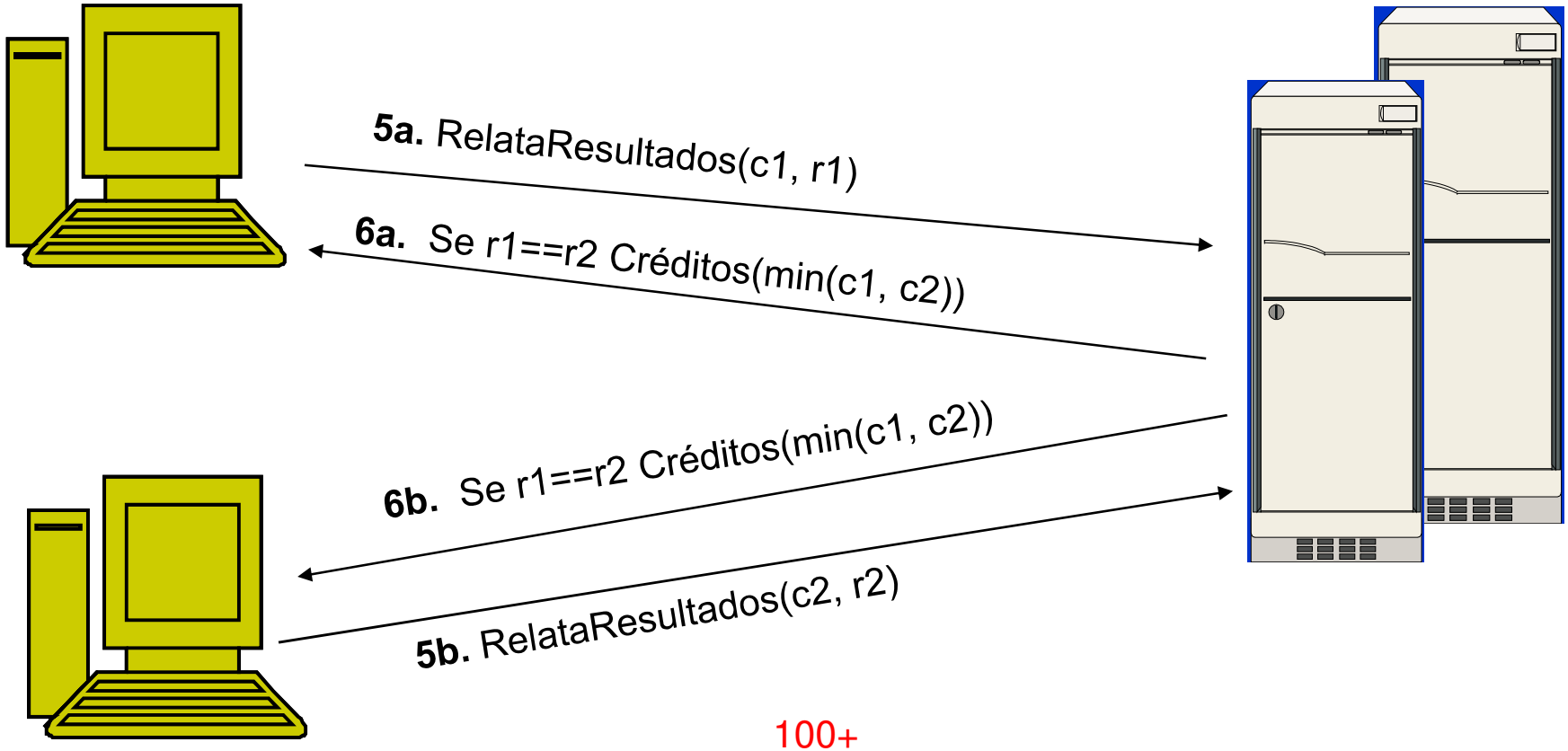
- World Community Grid: http://www.worldcommunitygrid.org
  - Identificar quais defeitos em proteínas produzidas pelos genes humanos podem causar doenças
  - Utiliza BOINC
- TeraGrid da universidade de Purdue (www.teragrid.org)
- BBC Climate Change Experiment: http://climateprediction.net/
  - Tentativa de prever o clima para o século 21
  - Utiliza BOINC
- **Introdução básica**:
  http://www.gridcafe.org/multimedia.html

Exemplo 6: BOINC



Exemplo 6: BOINC Créditos

- Mede quanto cada participante contribuiu
- Cada unidade de trabalho é enviada ao menos a dois PCs



100+
http://boinc.berkeley.edu/chart_list.php

consultar links de Grid

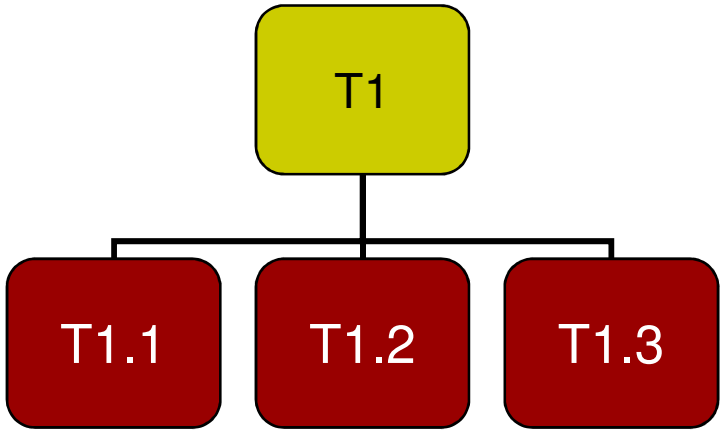
http://www.dainf.ct.utfpr.edu.br/~tacla/EspSD/

1 c

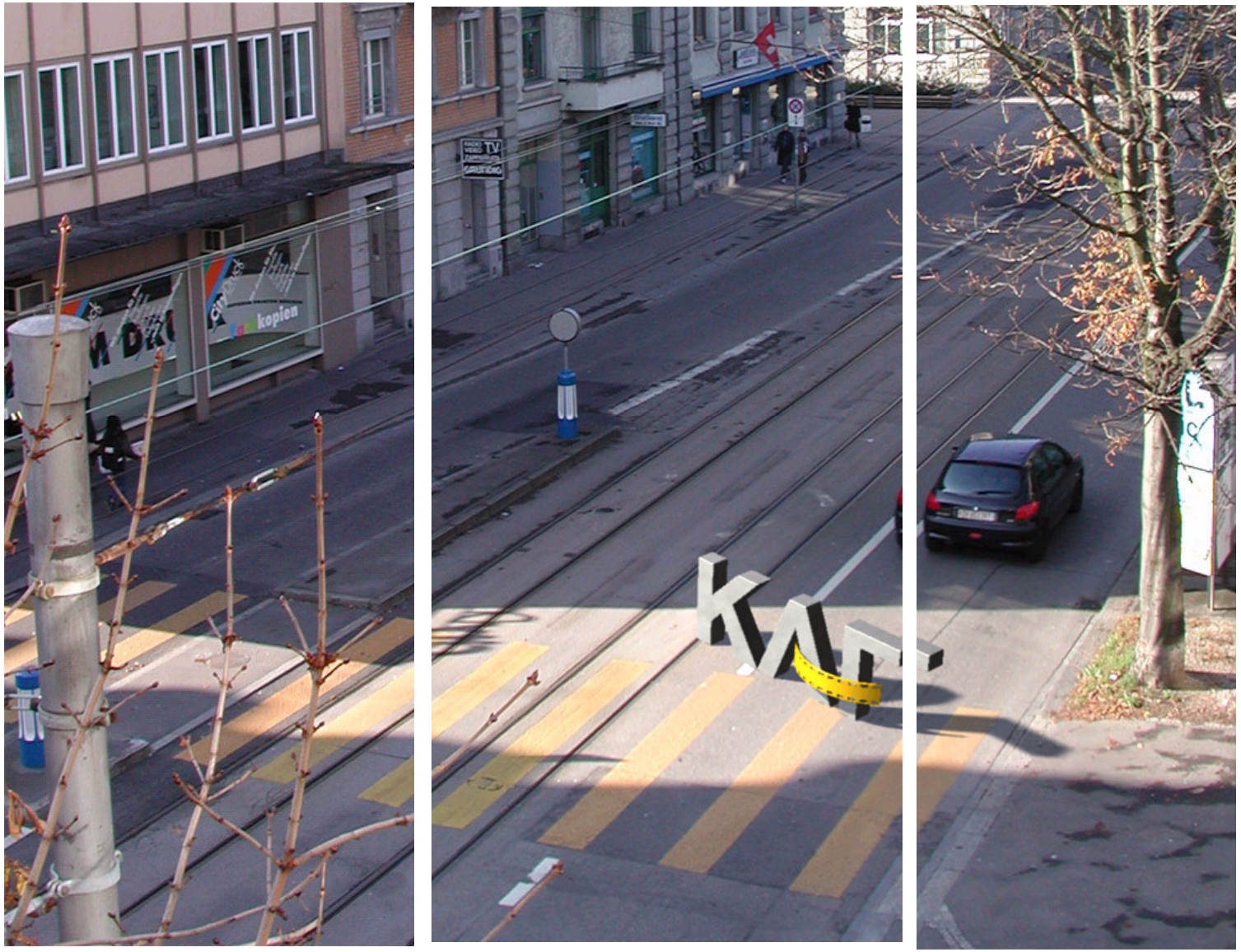
Tipos de aplicações distribuídas

Tipos de aplicação distribuídas (1)

- **Resolução distribuída de problemas**
  - Um usuário deseja resolver um problema complexo (ex. renderização de imagens complexas)
  - Decomposição de um problema em subproblemas
  - Distribuição dos subproblemas para diferentes processos
  - Um dos processos coleta e monta o resultado final



Decomposição de tarefas



Decomposição de tarefas(2)

Tempo

Renderman da Pixar https://renderman.pixar.com/index.html

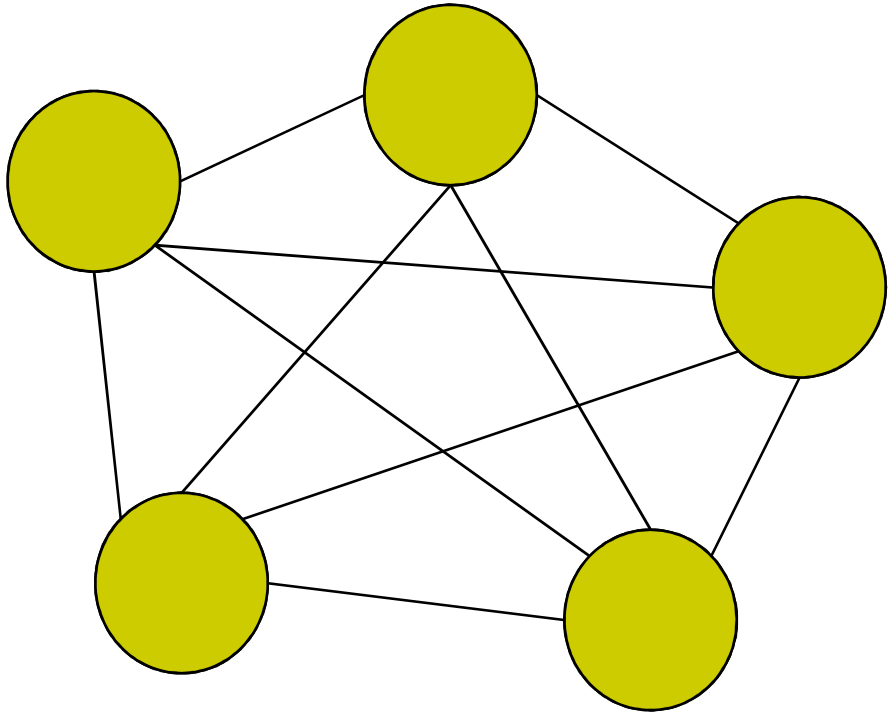
Filme 2 minutos ➔ 30 frames por segundo ➔ 7.200 minutos de renderização

No Distributed Render Environment:
Filme 2 minutos ➔ 30 frames por segundo ➔ 36 minutos de renderização

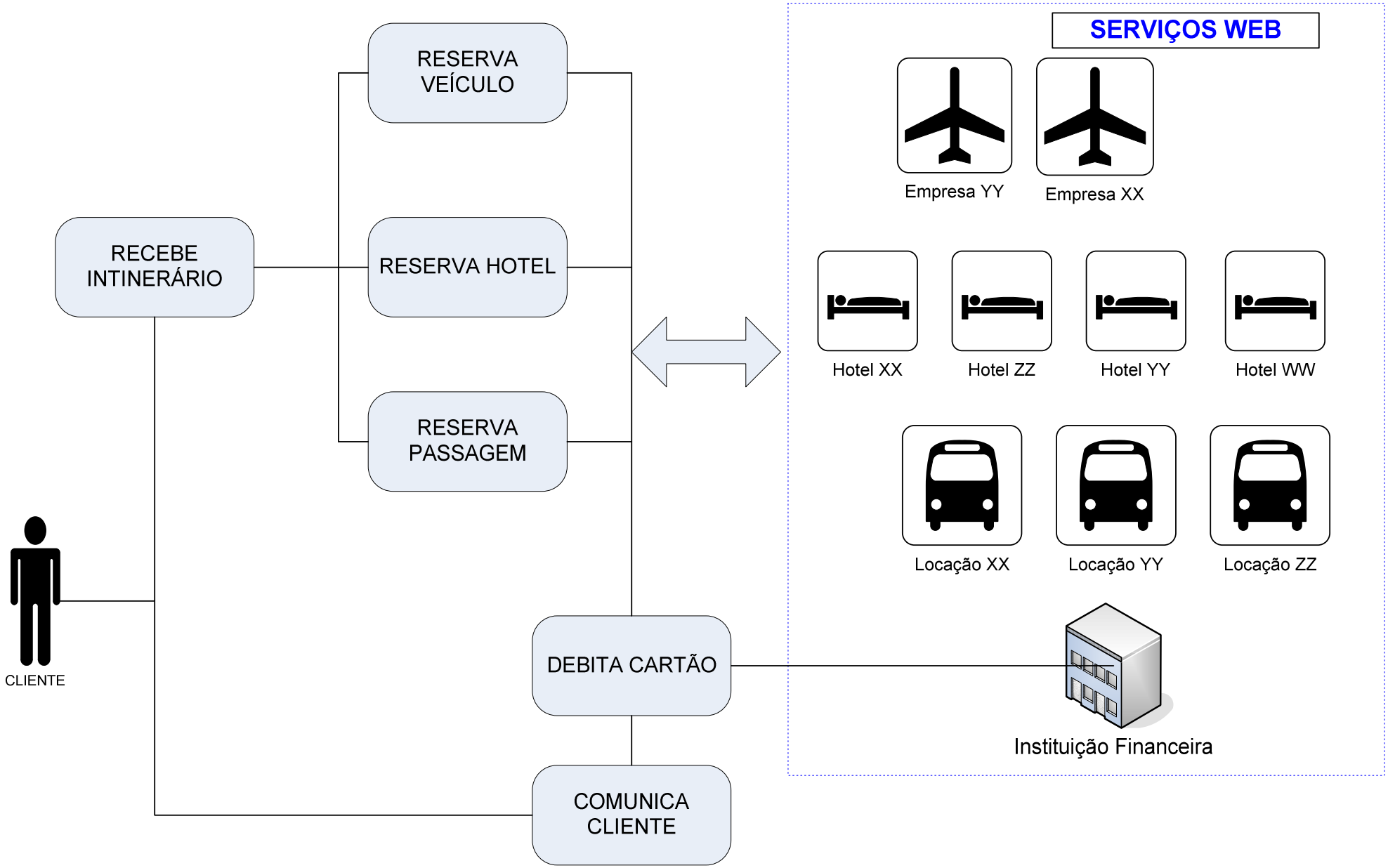
Fonte ~~http://www.i-light.org/docs/IPgrid_fn_report.doc~~ (TeraGrid Purdue)

◇ Composição de serviços

- sistemas multi-agentes/peer-to-peer
- Usuários com diferentes necessidades utilizam serviços/recursos heterogêneos providos por diferentes desenvolvedores
- Foco na interação entre agentes (competição ou colaboração); i.e. solução surge através da interação
- Ex. leilão



Tipos de aplicação distribuídas (3): Exemplo integração de serviços



1 d

Vantagens e dificuldades dos SDs

◇ Razões para construir um SD

- **Desempenho** > *sistema centralizado*
- **Distribuição** natural das aplicações: *bancárias*, *videoconf.*, *etc.*
- **Robustez**: frente a falhas
- **Escalabilidade**: sist. pode ser aumentado sem perda de desempenho?
- **Compartilhamento** de recursos
  - Impressoras e outros hardwares
  - Arquivos (inclusive páginas Web)
  - CPU
  - Memória
  - Espaço em disco
  - dados

- O desenvolvimento de um sistema distribuído é mais complexo do que um centralizado

- **Baseando-se no exemplo da aplicação bancária mostrada no início destes slides e considerando que:**
  - Há acessos múltiplos (concorrentes) à mesma conta;
  - Transações bancárias podem envolver contas armazenadas em diferentes bancos de dados (BD);
  - Não interessa ao cliente se os dados de sua conta estão armazenados no BD1 ou BD2

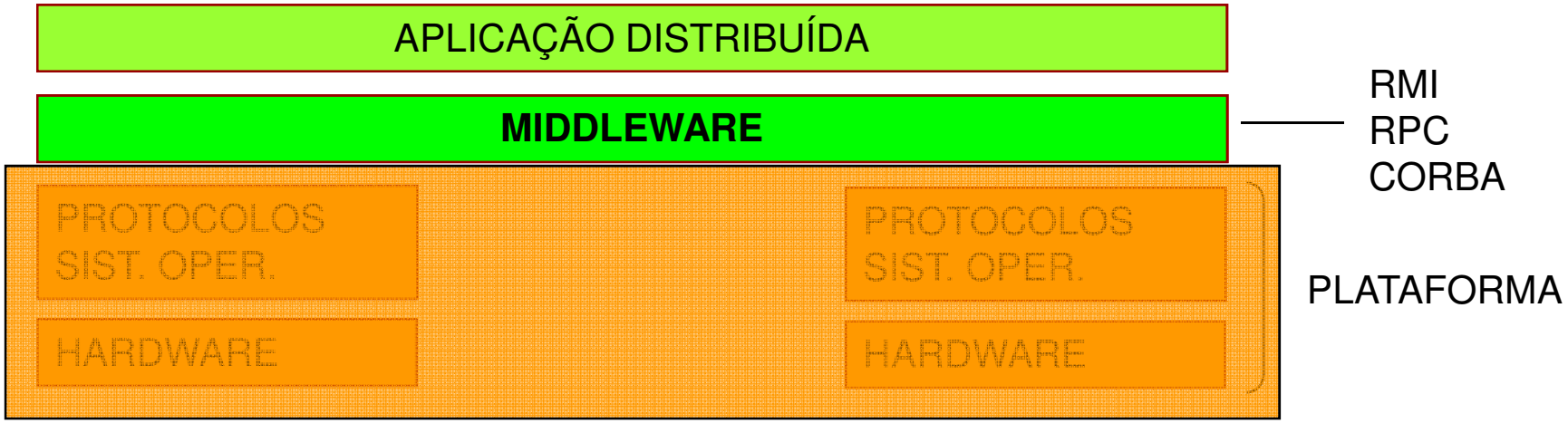
- **pode-se concluir que as características importantes para este tipo de sistema são... (> próximo)**

Exemplo

- **Acessos múltiplos (**concorrentes**)** à mesma conta: vários usuários acessam a mesma conta ou um sistema do banco (emissão de relatórios) acessa ao mesmo tempo em que usuários realizam transações.
- **Transações bancárias** podem envolver contas armazenadas em diferentes bancos ou localidades.
- **O serviço deve ser transparente ao usuário -** não interessa se os dados de sua conta estão armazenados no banco 1 ou banco 2, a aplicação não deve em nenhum momento deixar transparecer aspectos internos do sistema
- **O serviço deve ser** confiável **e** seguro
- **Bases de dados** replicadas **devem ser consistentes**
- **Tolerância a falhas**

- **Lidar com diferentes**
  - Redes
  - hardwares (computadores)
  - sistemas operacionais
  - Linguagens de programação

- **Além de resolver os problemas de heterogeneidade, o middleware define um modelo computacional uniforme para a programação de aplicações distribuídas.**



- **O grau de abertura de um SD depende de:**
  - Facilidade de **adicionar** novos recursos
    - De hardware
    - De software: novos serviços ou re-implementação de antigos
  - SD abertos baseiam-se em:
    - **Mecanismos de comunicação uniformes**
      - Especificados por organismos tais como OMG e IETF
      - Ex. CORBA (OMG)
    - **Publicação de interfaces de serviços**: para esconder a heterogeneidade de implementação dos serviços, as implementações devem aderir a uma forma de publicar as interfaces dos serviços

◇ Segurança envolve:

- "**selar**" informações importantes enviadas através da rede (ex. criptografia de número de cartão de crédito)

- **Identidade**: garantir a autenticidade da fonte.
  - Ex. um médico deseja acessar dados de pacientes de forma remota. É preciso assegurar que o usuário é realmente o médico.

  - Ex. um usuário deseja comprar um livro pela Internet. É preciso assegurar a identidade da loja virtual - que realmente é uma loja e não vai fazer mal uso do cartão.

- **Problemas** :
  - Indisponibilidade de serviço devido a ataques
  - Segurança de código móvel (ex. .exe anexados a emails, applets)

- **Um SD é *escalável* se, ao adicionarmos novos recursos e usuários, seu desempenho permanece satisfatório.**

- **Exemplos:**
  - Endereços IPv4 de 32 bits => IPv6 128 bits
  - DNS, no início, era uma tabela => foi particionada e é tratada localmente contendo replicações.

- **Técnicas para garantir escalabilidade:**
  - Replicação de dados
  - Caching
  - Replicação de serviços

- **Falhas atingem parcialmente um SD**

- **Detecção**
  - Mensagens corrompidas => checksum
  - Crash de um servidor => nunca é certo se o servidor caiu, se a falha está no link ou se a rede está sobrecarregada

- **Esconder falhas**
  - Se uma mensagem não chega no destino => retransmissão
  - Dados podem ser escritos em dois discos, se um está danificado, utiliza-se a cópia.

◇ Tolerância a falhas

- Exemplo: quando um navegador da Internet não consegue acessar um servidor ele não prende o usuário indefinidamente.

◇ *Como tornar um sistema tolerante a falhas?*

- **Redundância** pode tornar sistema tolerante a falhas
  - Replicação de dados
  - No serviço DNS cada entrada é replicada em pelo menos dois servidores
  - Explorar rotas alternativas na Internet

Maior tolerância a falhas aumenta a disponibilidade do sistema

- O sistema deve ser visto pelo usuário e pelo programador como um todo e não como uma coleção de componentes

- **Vários tipos de transparência**
  - **Acesso**: recursos locais e remotos são acessados através de operações idênticas
  - **Localização**: permite acessar um recurso sem conhecimento de sua localização (ex. URL)
    - **Replicação**: múltiplas instâncias de um recurso podem ser utilizadas sem que isto seja explícito para usuários e programadores

- **Vários tipos (cont)**
  - **Concorrência**: vários processos podem **acessar o mesmo recurso** sem interferir na utilização um do outro nem no recurso.
  - **Mobilidade**: permite a **movimentação de recursos** e clientes dentro de um sistema sem afetar a operação

Ex: sistema de telefonia celular (emissor)

SISTEMAS DISTRIBUÍDOS SÃO ÚTEIS PARA

- Melhorar desempenho
- Melhor adaptados a distribuição natural dos elementos

- *Porém são complexos*
- são inerentemente concorrentes
- devem ser confiáveis
- devem ser transparentes
- funcionar em plataformas heterogêneas

- *Middlewares resolvem parte destes problemas*