

# MC714

Sistemas Distribuídos

1º semestre, 2017

# Metas - Escalabilidade

- Meta importante devido à popularidade de dispositivos computacionais conectados em rede.
- Escalabilidade em três dimensões:
  - Em tamanho – fácil adicionar usuários e recursos.
  - Em termos geográficos – usuários e recursos podem estar distantes.
  - Em termos administrativos – fácil/possível de gerenciar mesmo abrangendo muitas organizações e recursos.
- Sistema escalável em uma ou mais dimensões frequentemente apresenta perda de desempenho com ampliação.

# Metas - Escalabilidade

- Necessidade de ampliar um sistema encontra barreiras de tipos diferentes.
- Escalabilidade em relação ao tamanho:
  - Mais usuários e mais recursos se deparam com o problema de centralização de serviços, dados e algoritmos.
  - Ex.: serviços centralizados implementados em um único servidor em uma máquina específica do sistema distribuído → gargalo
  - Gargalo pode ser de processamento, armazenamento, comunicação.
  - Pode ser inevitável centralizar (ex.: sistemas críticos).

# Metas - Escalabilidade

- Serviços centralizados. Ex.: único servidor para todos os usuários.
  - Problema: muitos usuários / aplicações.
  - Pode ser necessário por questões de segurança.
- Dados centralizados.
  - Exs.: um único catálogo de telefones online; DNS centralizado.
  - Armazenamento: factível.
  - Problema: gargalo no acesso – rede e disco.
- Algoritmos centralizados. Ex.: realizar roteamento baseado em informações completas.
  - Problema: colher e transportar toda a informação a cada alteração sobrecarrega a rede.
  - Preferência por algoritmos descentralizados.

# Escalabilidade

- Características de algoritmos descentralizados:
  - Nenhuma máquina tem informações completas sobre o estado do sistema.
  - Máquinas tomam decisões baseadas em informações locais.
  - Falha em uma máquina não arruína o algoritmo.
  - Não há suposição implícita que um relógio global existe.

# Escalabilidade

- Relógio global:
  - “Precisamente às 12:00:00 todas as máquinas anotarão o tamanho da sua fila de saída.” – precisaria de sincronização exata dos relógios. Quanto maior o sistema, maior a incerteza na sincronização.
- Escalabilidade geográfica
  - Difícil ampliar sistemas de LANs para larga escala distribuída, pois muitas vezes são baseados em comunicação síncrona.
  - Funciona bem quando comunicação é rápida.
  - Comunicação em longa distância: inerentemente não confiável.
  - Localizar um serviço: broadcast em LAN vs. sistema distribuído.

# Escalabilidade

- Escalabilidade entre diferentes domínios administrativos
  - Políticas conflitantes de utilização e/ou pagamento.
  - Extensão de um sistema a outros domínios demanda medidas de segurança.
    - Proteger-se contra ataques do novo domínio (ex. acesso somente leitura aos novos usuários)
    - Restringir acesso a componentes e/ou dados críticos.
    - Impor limites de acesso a códigos potencialmente perigosos.
    - Problema: como impor tais limites.

# Escalabilidade / técnicas

- Como resolver problemas de escalabilidade?
- Muitas vezes esses problemas aparecem na forma de queda de desempenho com aumento do número de usuários/componentes.
  - Capacidade limitada de servidores e da rede.
- Três técnicas:
  - Ocultar latências de comunicação (comunicação assíncrona);
  - Distribuição;
  - Replicação.



# Escalabilidade / técnicas

- Ocultar latências de comunicação
  - Não aguardar por resposta: comunicação assíncrona.
  - Útil em processamento em lotes e aplicações paralelas nas quais tarefas mais ou menos independentes podem ser escalonadas para execução enquanto outra espera comunicação.
  - Há aplicações onde não é possível fazer comunicação assíncrona
    - ex: aplicações interativas: nada melhor a fazer que esperar a resposta.
    - Solução melhor: reduzir comunicação global.
    - Ex.: verificação de erros de sintaxe em validação de formulário: código no servidor vs. código no cliente. Fig 20.

# Escalabilidade / técnicas

- Distribuição
  - Dividir componente e espalhar pelo sistema.
  - Ex.: DNS. Fig 21.; Web.
- Replicação
  - Aumenta disponibilidade
  - Diminui latência
  - Equilibra carga

# Escalabilidade / técnicas

- Replicação
  - Ex.: Cache (forma “especial” de replicação);
    - Cache: decisão tomada pelo cliente; replicação pelo proprietário.
    - Cache: sob demanda; replicação: planejada.
    - Problema: consistência.
  - Inconsistência pode ser tolerada dependendo da utilização de um recurso.
    - Cache web: aceitável documento não atualizado por alguns minutos.
    - Bolsas de valores e leilão eletrônico: não aceitável.
    - Forte consistência demanda atualização propagada para todas as outras cópias imediatamente → mecanismos de sincronização global.
      - Difícil e ineficiente em sistemas de larga escala.
  - Tolerar inconsistências reduz necessidade de sincronização global, mas isso é dependente de aplicação.

# Escalabilidade / técnicas

- Escalabilidade de tamanho em muitos casos é a menos problemática → simples aumento da capacidade de uma máquina resolve a questão, ao menos temporariamente.
- Escalabilidade geográfica mais difícil, depende da natureza.
  - Combinar técnicas de distribuição, replicação e cache com diferentes formas de consistência é suficiente em diversos casos.
- Escalabilidade administrativa é mais difícil porque envolve não só problemas técnicos (políticas de organizações e colaborações humanas).
  - Solução parcial: P2P, onde usuários finais tem o controle administrativo.

# Armadilhas

- Suposições falsas comuns feitas por desenvolvedores:
  - Rede é confiável;
  - Rede é segura;
  - Rede é homogênea;
  - A topologia não muda;
  - Latência é zero;
  - Largura de banda é infinita;
  - Custo do transporte é zero;
  - Há um administrador.

# Tipos de sistemas distribuídos

# Tipos de SDs

- Sistemas de computação distribuídos
- Sistemas de informação distribuídos
- Sistemas embutidos (embarcados) distribuídos

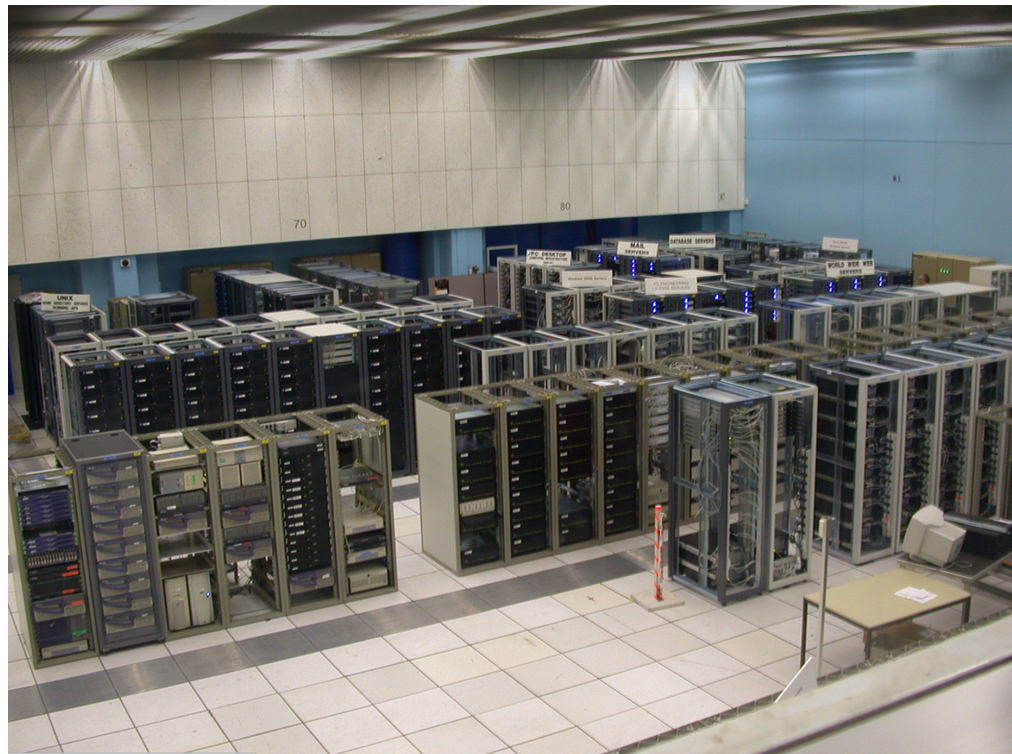
# Sistemas de Computação Distribuídos

- Utilizado para tarefas de computação (frequentemente de alto desempenho).
- Computação em cluster
- Computação em grade
- Computação em nuvem



# Cluster Computing

- Tornaram-se populares pela razão preço/desempenho.
  - Estações mais potentes e mais baratas
  - Rede melhor
- Computação intensiva paralela.



# Cluster Computing

- Ex.: Beowulf. Fig 22.
  - Mestre
    - Alocação de nós / fila / escalonamento
    - Interface para usuário
    - Executa middleware para execução de programas e gerência do cluster.
    - Nós de computação: SO padrão pode ser suficiente.
  - Bibliotecas de execução em sistemas paralelos: facilidades para comunicação por troca de mensagem.
  - Migração de processos: movimento transparente de processos de um nó nativo para qualquer outro nó.

# Computação em grade

- Cluster: homogêneo.
- Grade: heterogênea, nenhuma premissa adotada em relação a hardware, S.O., rede, domínio administrativo, política de segurança, etc.
- Recursos de diferentes organizações reunidos para permitir colaboração.
- Organização virtual: pessoas em uma organização virtual têm direitos sobre recursos dessa organização.
  - Servidores de computação (inclusive clusters), armazenamento, instrumentos, bancos de dados, equipamentos, sensores, telescópios, etc.

# Computação em grade

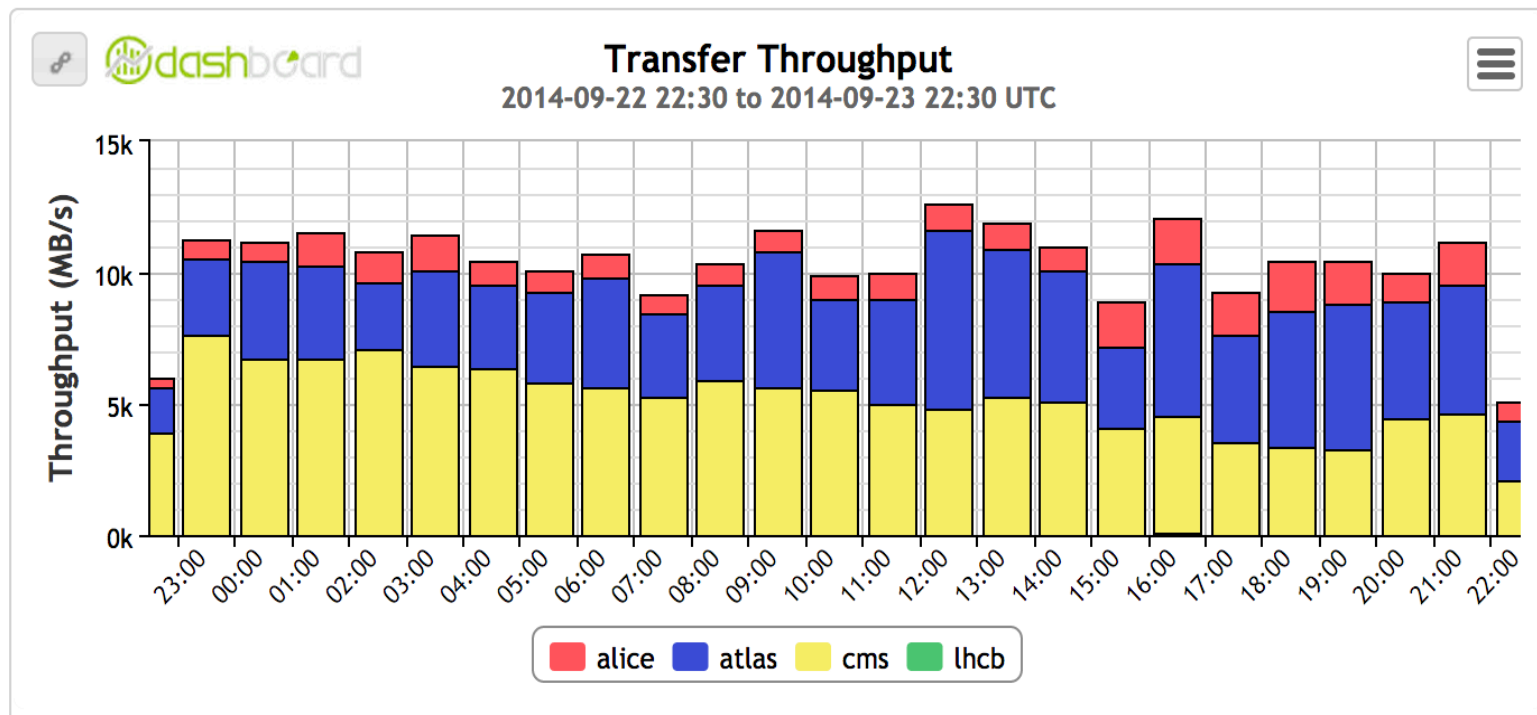
## Current WLCG sites



# Computação em grade

## Welcome to the Worldwide LHC Computing Grid

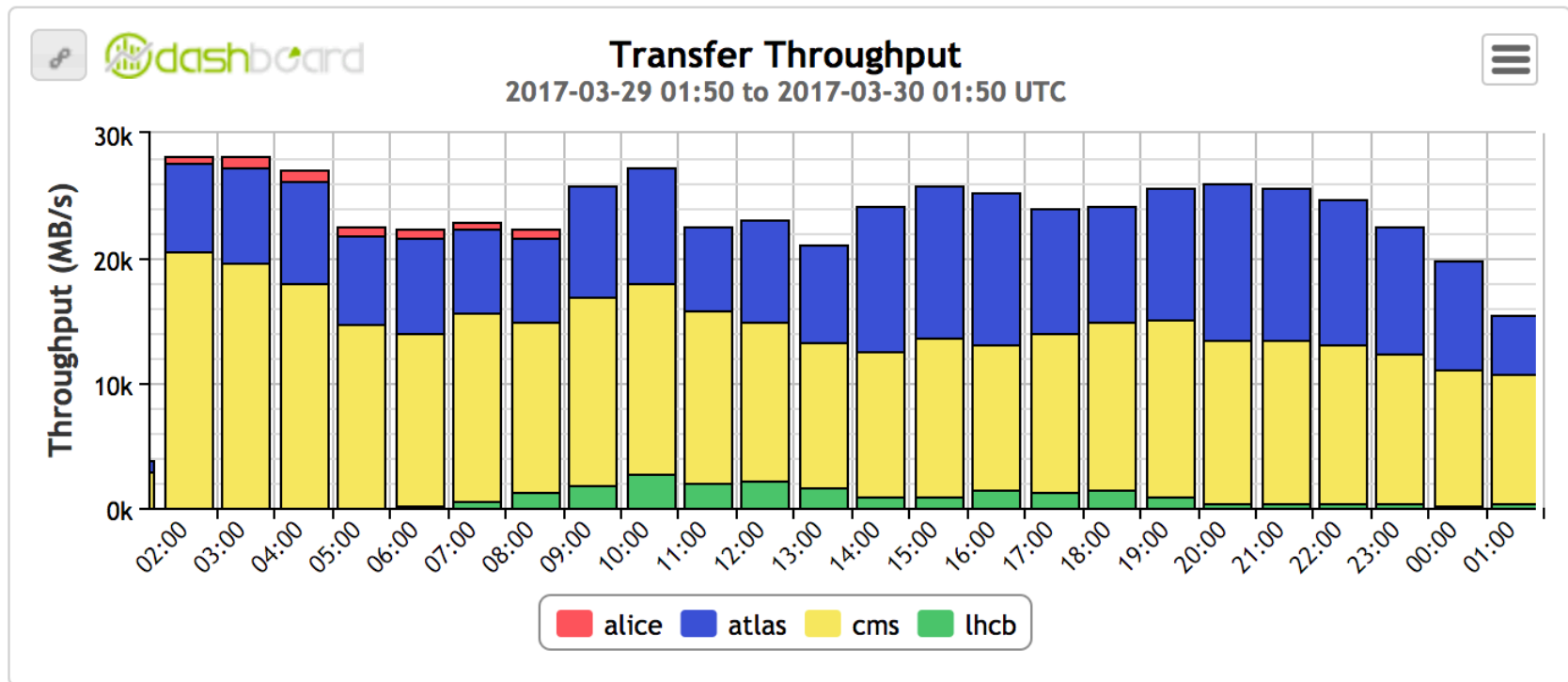
Last 24 hours



<http://wlcg.web.cern.ch/>

# Computação em grade

Last 24 hours



# Computação em grade

- Software de grade: grande parte com finalidade de prover acesso a recursos de diferentes domínios administrativos.
- Fig 23.
- Camada base: interfaces para recursos locais em site específico. Consultar estado de recursos e gerência de recursos locais.
- Camada de conectividade: protocolos de comunicação para suportar transações utilizando múltiplos recursos. Ex.: transferência de dados, autenticação, protocolos de segurança.
- Camada de recurso: gerência de um único recurso. Ex.: criar processo, ler dados. Responsável por controle de acesso – necessita autenticação.

# Computação em grade

- Camada coletiva: manipular acesso a múltiplos recursos. Serviços de descoberta de recursos, alocação / escalonamento de tarefas para múltiplos recursos, replicação de dados, etc.
  - Muitos serviços - pode consistir de muitos protocolos independentes.
- Camada de aplicação: aplicações que usam o ambiente virtual dentro da grade.
- Coletiva + conectividade + recursos = middleware da grade.
- Noção de “site” é comum: tendência do SOA (acesso às camadas através de serviços web).



# Computação em grade

- Camadas Coletiva+conectividade+recurso formam o cerne de um middleware de grade.
  - Dão acesso e gerenciam recursos dispersos por vários “sites”.
- Noção de “site” é comum: tendência do SOA (acesso às camadas através de serviços web).
- OGSA → Open Grid Services Architecture
  - Várias camadas e muitos componentes voltados a definir uma grade aberta orientada a serviços

# Cloud Computing

- Introduz noção de “computação como serviço”.
- Relação com *utility grids*.
- Recursos virtualizados.
- Diferentes níveis de serviço. IaaS, PaaS, SaaS.
- Pagamento pelo uso.
- Nuvem pública / privada / híbrida / comunitária.
- Evita alto investimento inicial.
- Fig. 24
- Visões: provedor e cliente.

# Sistemas de informação distribuídos

# Sistemas de Informação distribuídos

- Organizações se defrontaram com uma profusão de aplicações em rede.
  - Interoperabilidade complicada.
- Algumas soluções de middleware existentes são resultado da integração de tais aplicações em um sistema empresarial
  - Mais fácil que desenvolver todas novamente.

# Sistemas de Informação distribuídos

- Integração em vários níveis.
- Aplicação (servidor + banco de dados) disponibilizada a clientes remotos.
- Cliente envia requisição, recebe resposta.
- Integração em nível mais baixo: clientes poderiam empacotar várias requisições para diferentes servidores em uma única requisição maior.
  - Envio para execução em forma de transação distribuída.
  - Idéia fundamental: ou todas ou nenhuma seria executada.
- Ex.: reserva passagem, hotel, aluguel de automóvel, restaurante (groupon), passagem de trem.
  - Com e sem sistema computacional

# Sistemas de Informação distribuídos

- Aplicações tornaram-se gradualmente separadas em componentes independentes
  - Componentes de banco de dados e de processamento
- Integração deveria ocorrer em nível mais alto
  - Comunicação também entre aplicações
- Indústria de integração de aplicações empresariais (Enterprise Application Integration – EAI).