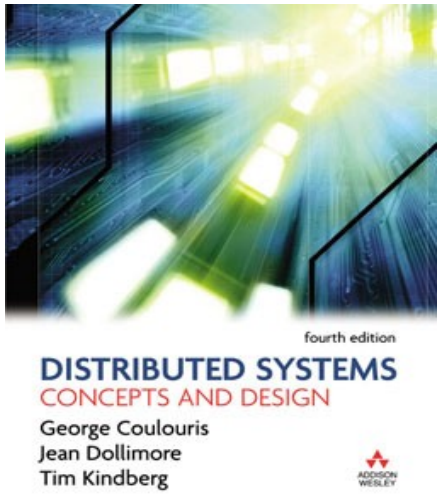
Slides for Chapter 1
Characterization of Distributed Systems



From Coulouris, Dollimore and Kindberg
Distributed Systems:
Concepts and Design

Figure 1.1
A typical portion of the Internet

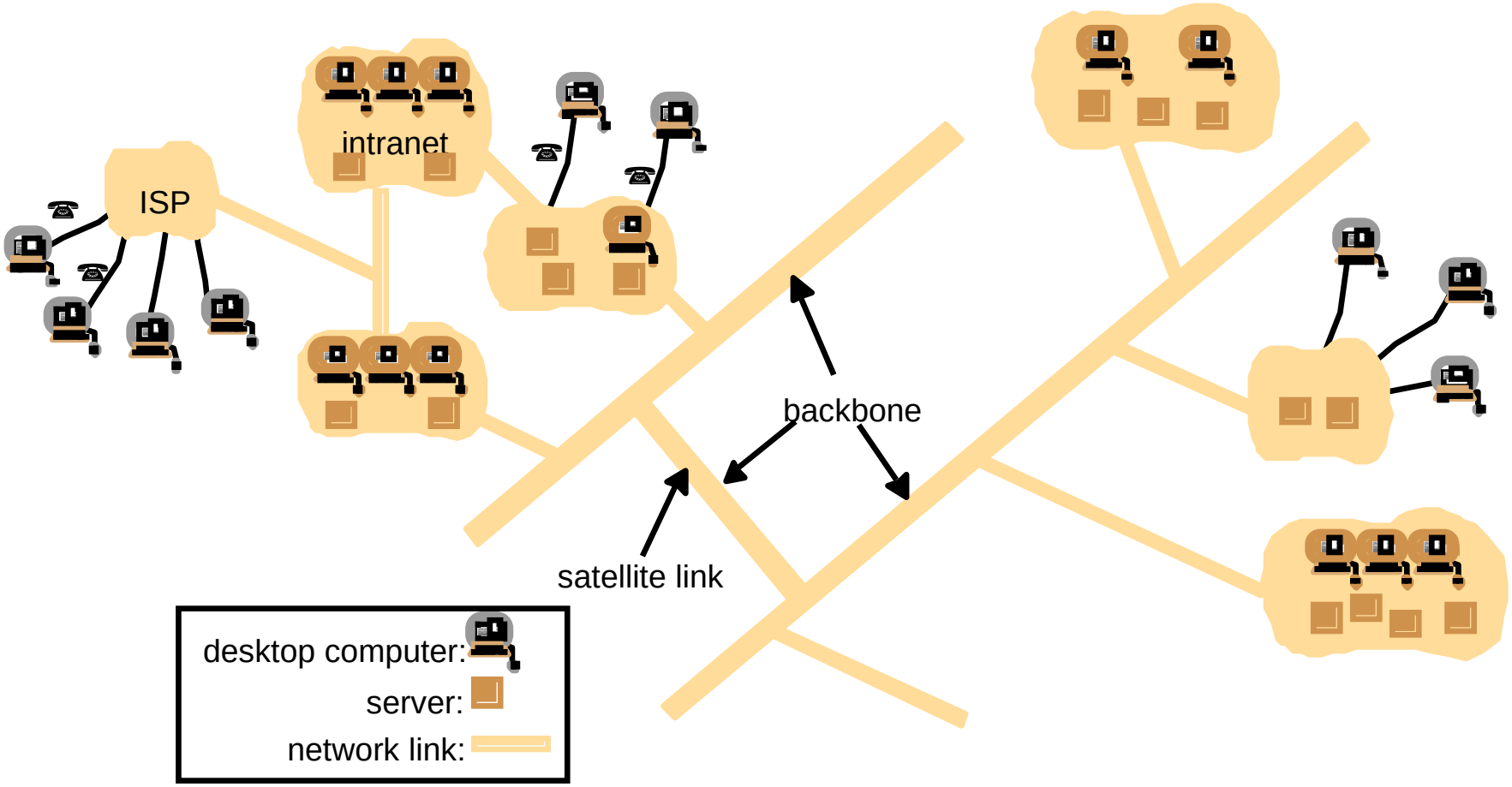


Figure 1.2
A typical intranet

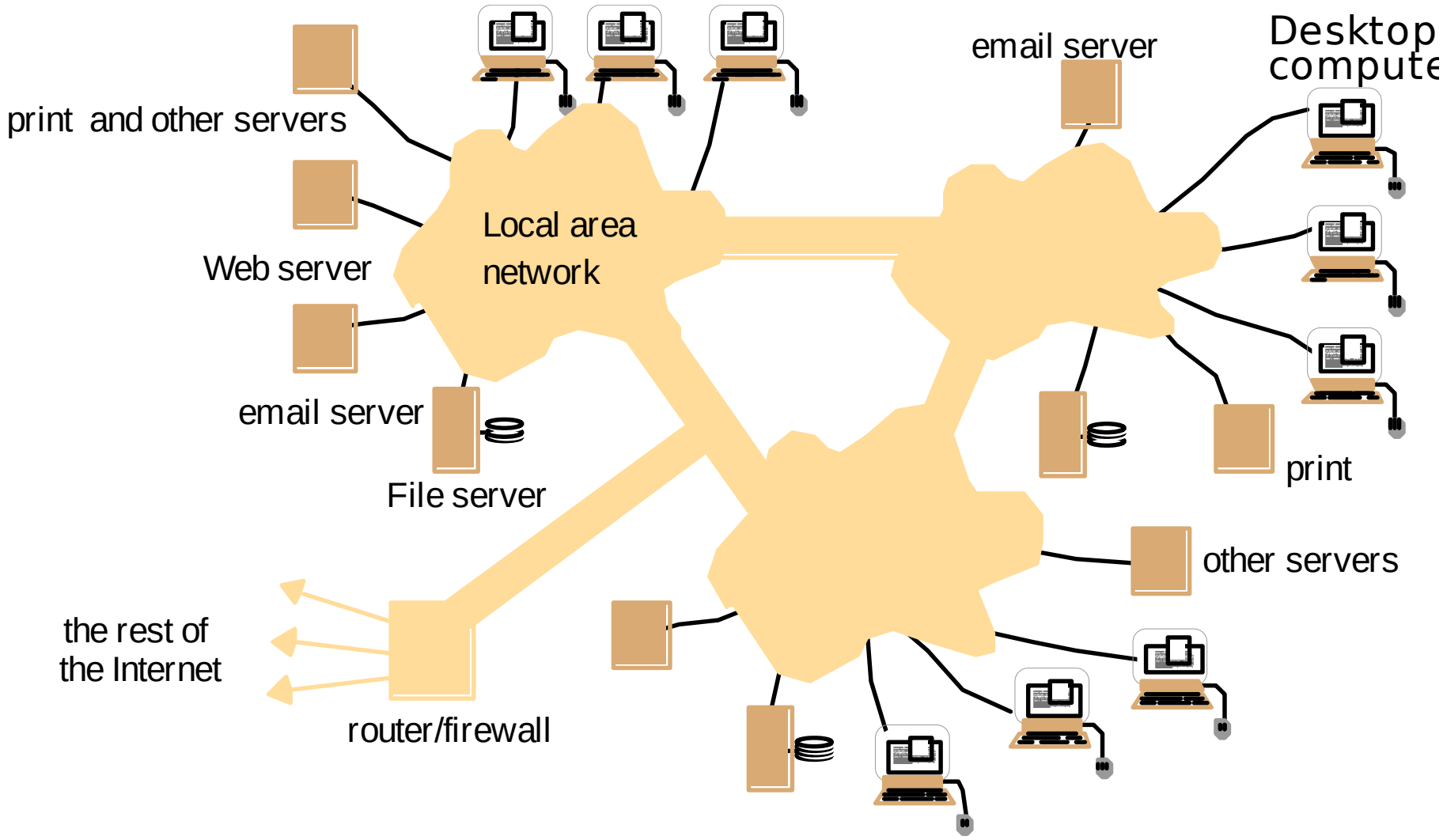


Figure 1.3
Portable and handheld devices in a distributed system

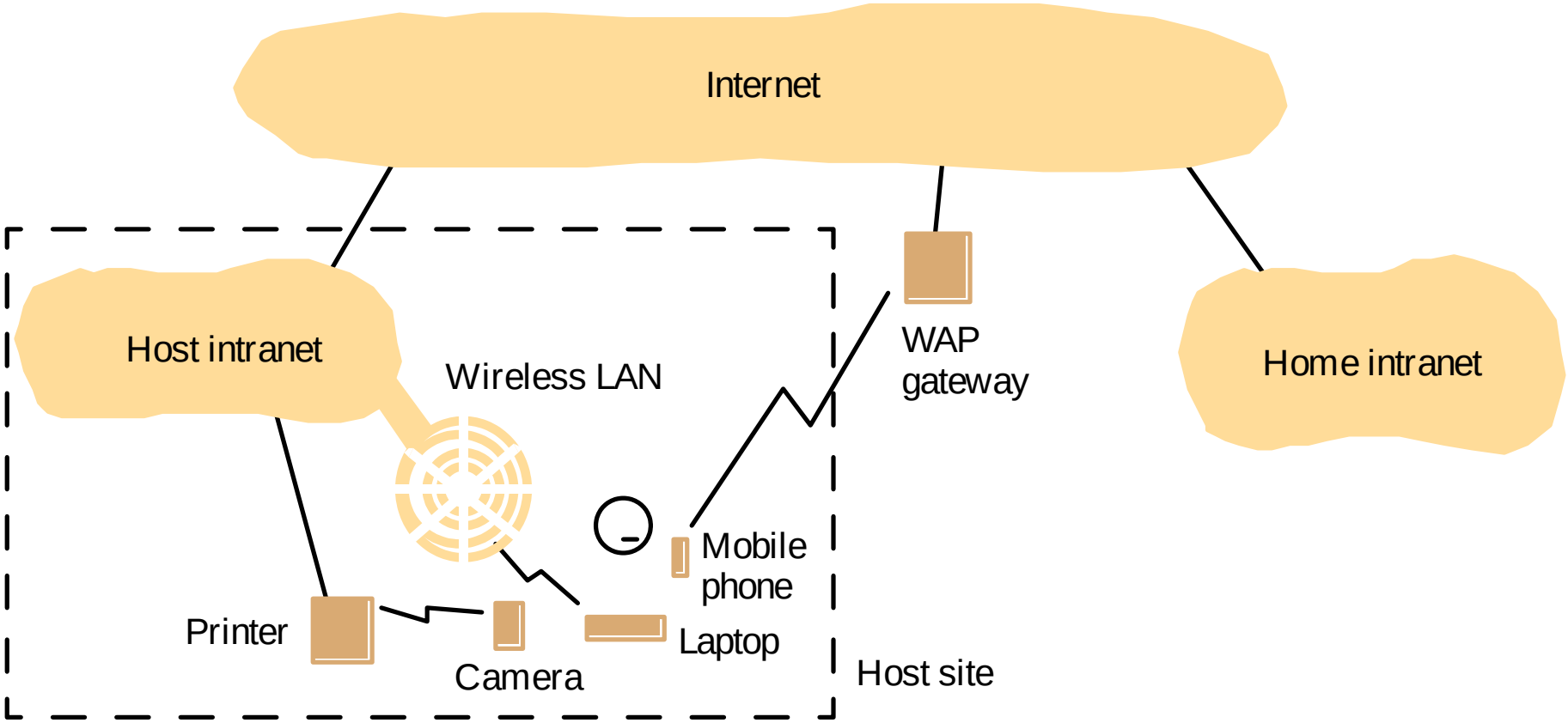


Figure 1.4
Web servers and web browsers

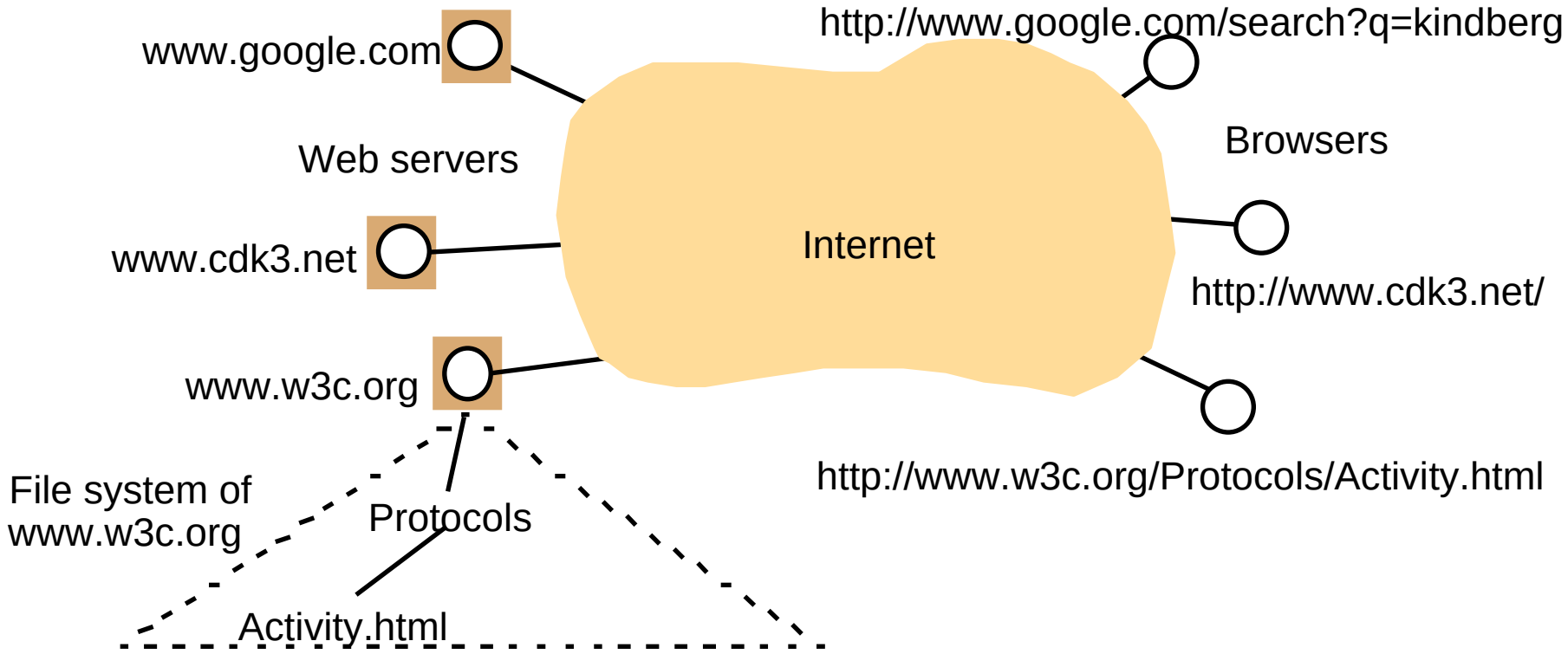
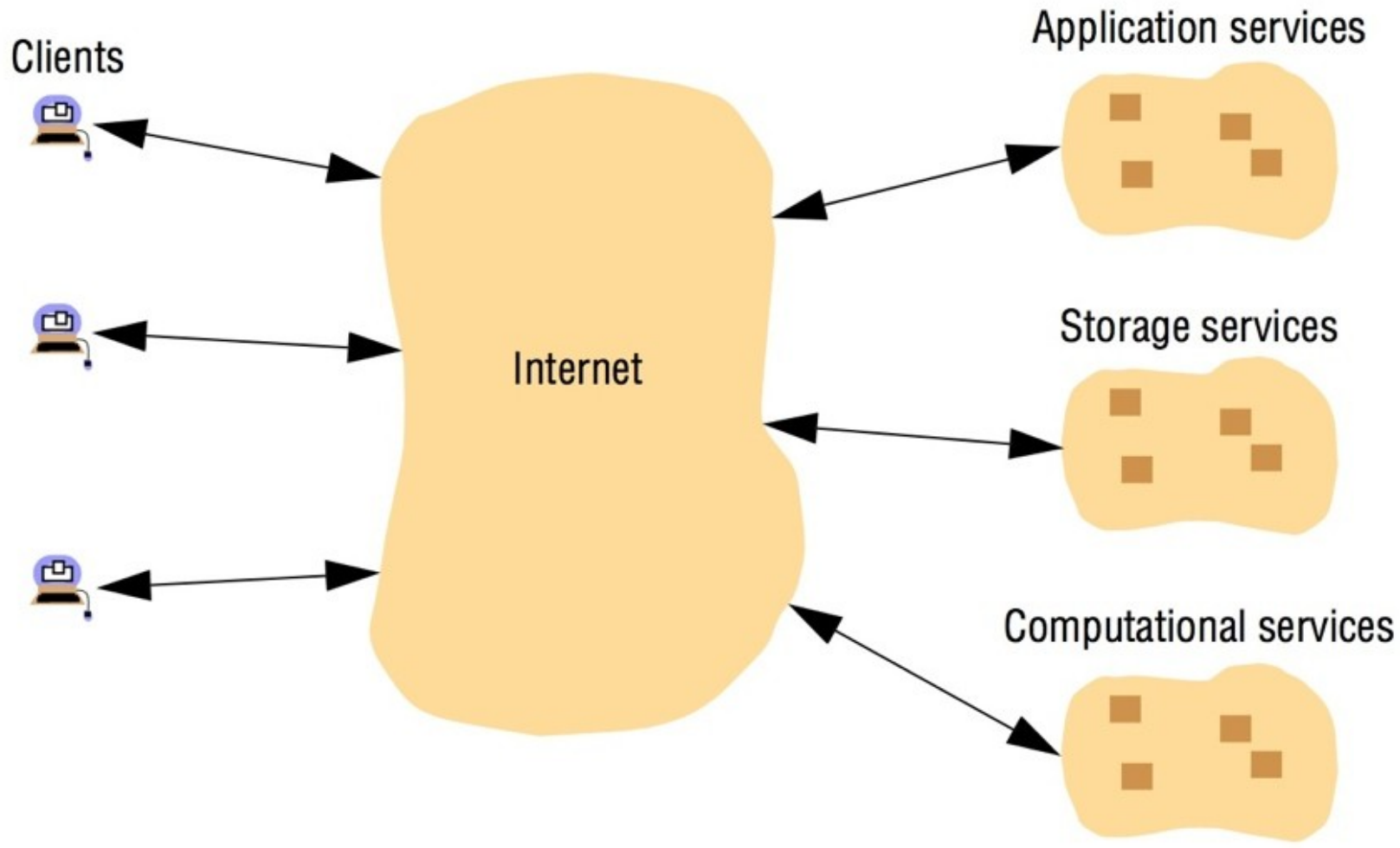


Figure 1.5
Cloud computing



Sistema distribuído vs. rede de computadores

- **Rede**: um meio para **interconectar** computadores e **trocar mensagens** através de protocolos bem definidos. Entidades da rede são visíveis e endereçadas **explicitamente**
- **Sistema distribuído**: a existência de múltiplos computadores autônomos, de forma transparente
- Muitos problemas (e.g., abertura, confiabilidade) são comuns a ambos, mas tratados em diferentes níveis
  - As redes tratam no nível de **pacotes**, **roteamento**, etc, enquanto os sistemas distribuídos tratam no **nível das aplicações**
  - Todo sistema distribuído depende dos **serviços oferecidos** por uma ou mais redes de computadores

Definição de Sistemas Distribuídos

- Definição adotada no curso:
  - Um sistema no qual componentes de hardware e/ou software, localizados em diferentes computadores conectados em rede, se **comunicam** e **coordenam** suas ações apenas através da **troca de mensagens** [Coulouris et al. 05]
- Definição implica em três características:
  - Concorrência
  - Ausência de relógio global
  - Falhas independentes

Desafios

- Heterogeneidade
- Aberta
- Segurança
- Escalabilidade
- Tratamento de Falhas
- Concorrência
- Transparência

Heterogeneidade

- Variedade e diferença em termos de:
  - Hardware
  - Sistemas operacional
  - Rede
  - Linguagem de programação
  - Implementações de diferentes desenvolvedores
- Exemplos de heterogeneidade na Internet
  - Diferentes implementações do mesmo conjunto de protocolos para diferentes tipos de rede: IP, TCP, UDP, SMTP
  - Diferentes padrões de representação de dados: IDL, XML
  - Diferentes padrões de bibliotecas: POSIX, DLL
  - Diferentes padrões de invocação de serviços: COM, CORBA,RMI, SOAP
  - Diferentes plataformas de execução: JVM (Java), CLR (.NET)

Heterogeneidade

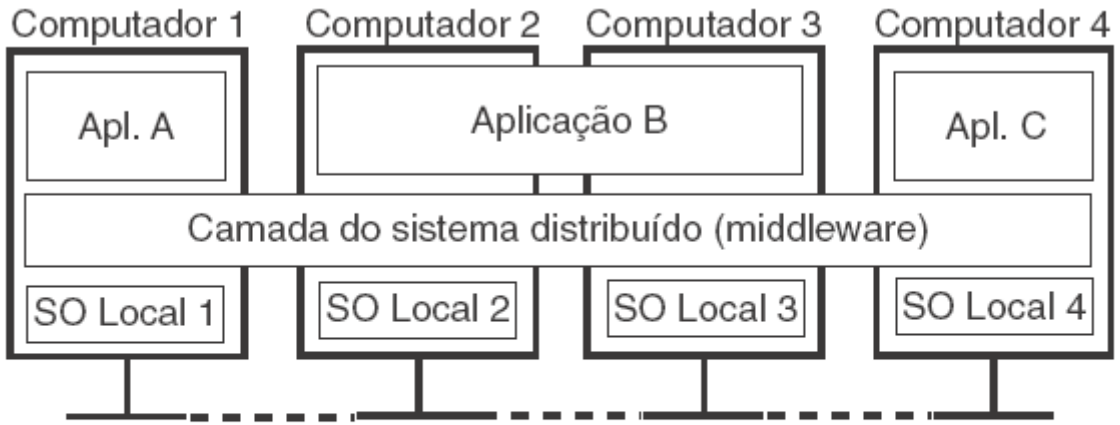


Figura 1.1 Sistema distribuído organizado como middleware. A camada de middleware se estende por várias máquinas e oferece a mesma interface a cada aplicação.

Abertura

- Facilidade de extensão e atualização
  - Adição de **novos recursos** e serviços
  - Re-implementação de **serviços existentes**
- Depende que as **interfaces** de acesso aos principais componentes do sistemas sejam conhecidas e estejam disponíveis para os programadores
- Exemplos de abertura na Internet
  - **Especificações** controladas e atualizadas por um **Comitê Gestor**
  - **Novos produtos e serviços** implementados de acordo com as especificações vigentes
  - **Conformidade da implementação** deve ser testada e verificada para garantir o correto funcionamento do sistema

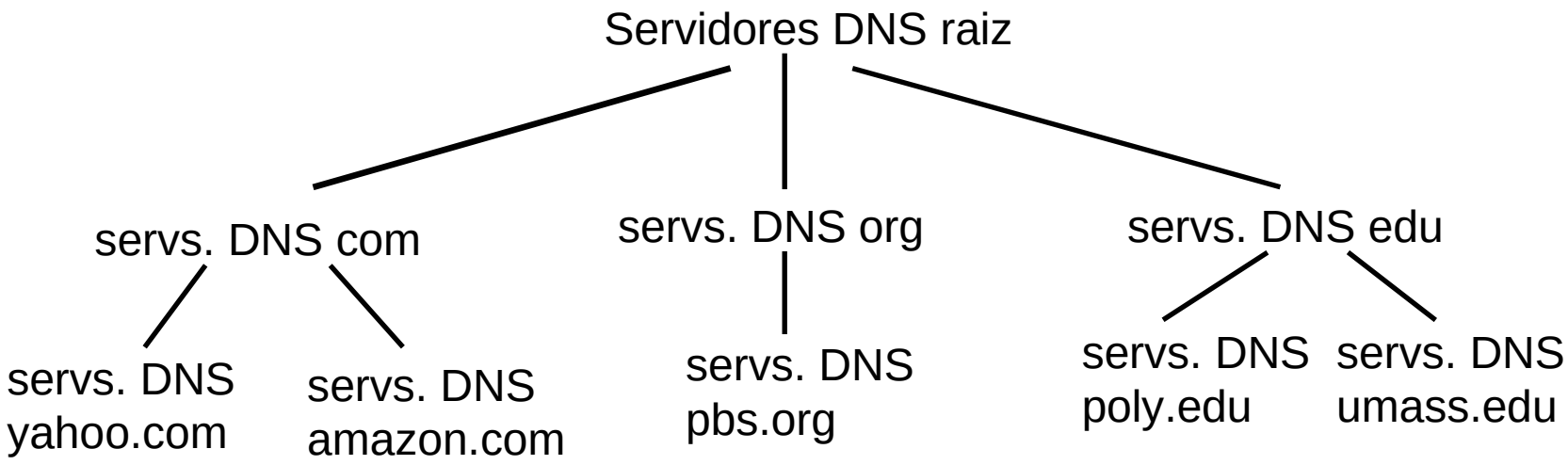
Segurança

- Proteção para recursos compartilhados
  - **Confidencialidade** (proteção contra usuários não autorizados)
    - Ex.: Acesso a dados sobre salário, histórico médico
  - **Integridade** (proteção contra alteração e corrupção)
    - Ex.: Alteração indevida de dados usados em transações bancárias
  - **Disponibilidade** (proteção contra interferência ao meio de acesso)
    - Ex.: Queda ou sobrecarga do servidor ou do meio de comunicação
- Principais mecanismos de segurança na Internet
  - Firewall
  - Assinaturas digitais
  - Canais seguros de comunicação
- Desafios recentes
  - Ataques de **negação de serviço**
  - **Segurança** para código móvel

Escalabilidade

- Capacidade do sistema permanecer operando de forma efetiva mesmo diante de um aumento significativo do número de usuários e/ou dos recursos disponíveis
- Principais desafios:
  - Controlar o **custo** dos recursos físicos – *O(n)*
  - Controlar **perdas de desempenho**
  - Prevenir o **esgotamento dos recursos** de software
  - Evitar "**gargalos**" de desempenho na rede ou nos próprios servidores
- Principais técnicas:
  - Replicação
  - Caching
  - Concorrência e paralelismo

Escalabilidade



Escalabilidade

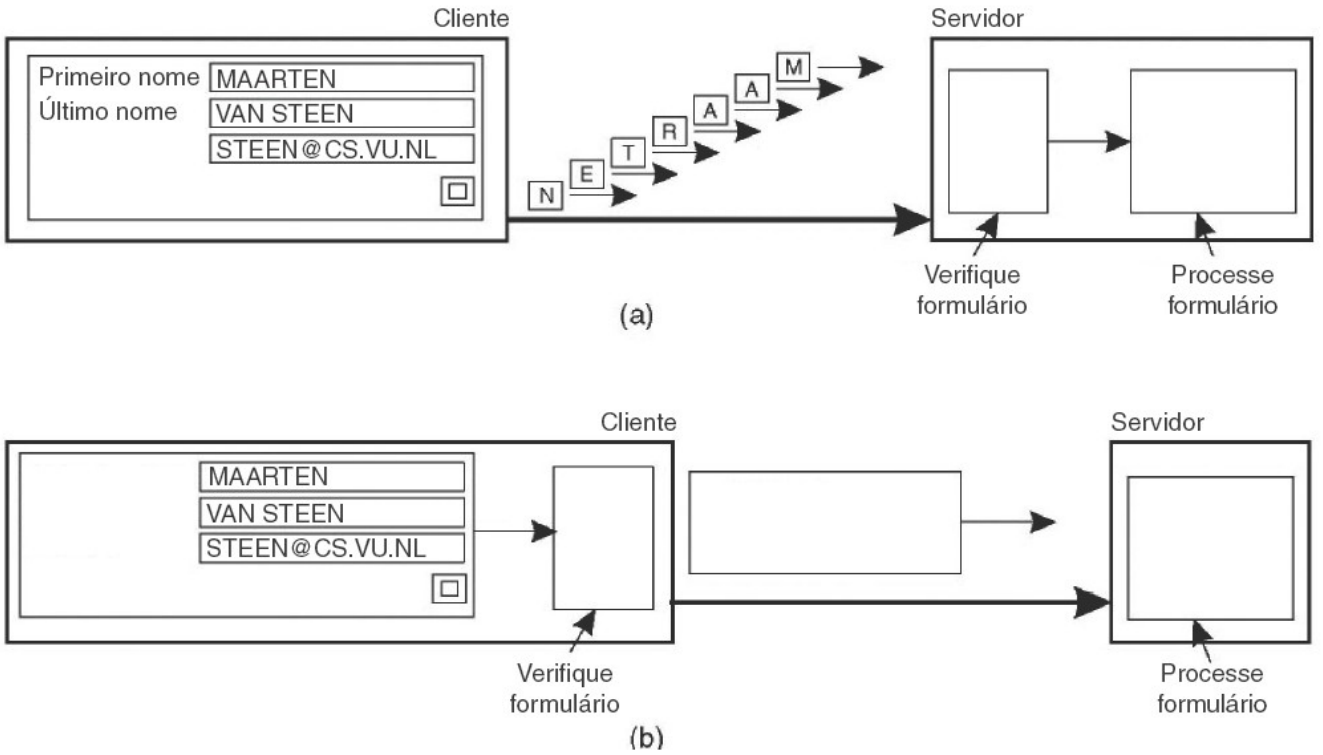


Figura 1.2 A diferença entre deixar (a) um servidor ou (b) um cliente verificar formulários à medida que são preenchidos.

Figure 1.6
Growth of the Internet (computers and web servers)

Date	*Computers*	*Web servers*	*Percentage*
1993, July	1,776,000	130	0.008
1995, July	6,642,000	23,500	0.4
1997, July	19,540,000	1,203,096	6
1999, July	56,218,000	6,598,697	12
2001, July	125,888,197	31,299,592	25
2003, July	~200,000,000	42,298,371	21
2005, July	353,284,187	67,571,581	19

Computadores com endereços IP registrados na Internet

Tolerância a falhas

- Falhas são **inevitáveis** em sistemas computacionais
  - Resultados incorretos
  - Interrupção não planejada do serviço antes de sua conclusão
- Falhas em sistemas distribuídos são **parciais**
- Técnicas de tratamento de falhas mais comuns:
  - **Detecção**
    - (ex. bits de paridade, somas de verificação)
  - **Ocultamento**
    - (ex. retransmissão de mensagens)
  - **Tolerância**
    - (ex. informar o usuário do problema)
  - **Recuperação**
    - (ex. transações em BD's)
  - **Redundância**
    - (ex. replicação de tabelas no DNS)
- Sistemas distribuídos devem oferecer **alta disponibilidade** de recursos mesmo diante da ocorrência de falhas
  - **Disponibilidade**: medida da proporção do tempo que um recurso está disponível para uso

Concorrência

- Suporte para múltiplos acessos simultâneos a um ou mais recursos compartilhados
  - Possibilidade de inconsistências quando os recursos são alterados
- Serviços que representam recursos compartilhados devem ser responsáveis por garantir que as operações de acesso os mantenham em um estado consistente
  - Válido para servidores e objetos de aplicações
- Técnicas mais comuns:
  - Sincronização de acesso (ex.: exclusão mútua distribuída)
  - Protocolos de controle de concorrência (ex.: 2PC)

Transparência

- **Abstração/Ocultação,** para os usuários e programadores de aplicação, da separação dos componentes em um sistema distribuído
  - Sistema percebido como um "**todo**" coerente ao invés de uma coleção de partes independentes
- Uma medida de sucesso de um sistema distribuído é dada pela sua transparência:
  - Em que medida é indistinguível de um sistema centralizado com a mesma funcionalidade?

Transparência

- ***Transparência de acesso:***
  - permite o acesso a componentes remotos e locais através das mesmas operações
  - Ex:
    - *Network File System*
    - *Google Docs*

- ***Transparência de localização***:
  - permite o acesso a componentes sem conhecimento da sua localização física
  - existência de um mecanismo que determina a localização baseada num nome
  - Ex:
    - URL

Transparência

- ***Transparência de concorrência***:
  - permite a execução concorrente de múltipla operações sobre o mesmo conjunto de recursos sem causar interferência entre elas
  - Ex:
    - Impressoras compartilhadas
    - Leilão virtual
- ***Transparência de escala***:
  - permite a expansão do sistema e de suas aplicações sem exigir mudanças significativas na infra-estrutura existente
  - o sistema não possui gargalos
- ***Transparência de mobilidade*** *(migração)*:
  - permite a realocação de recursos e aplicações sem afetar o seu uso

Transparência

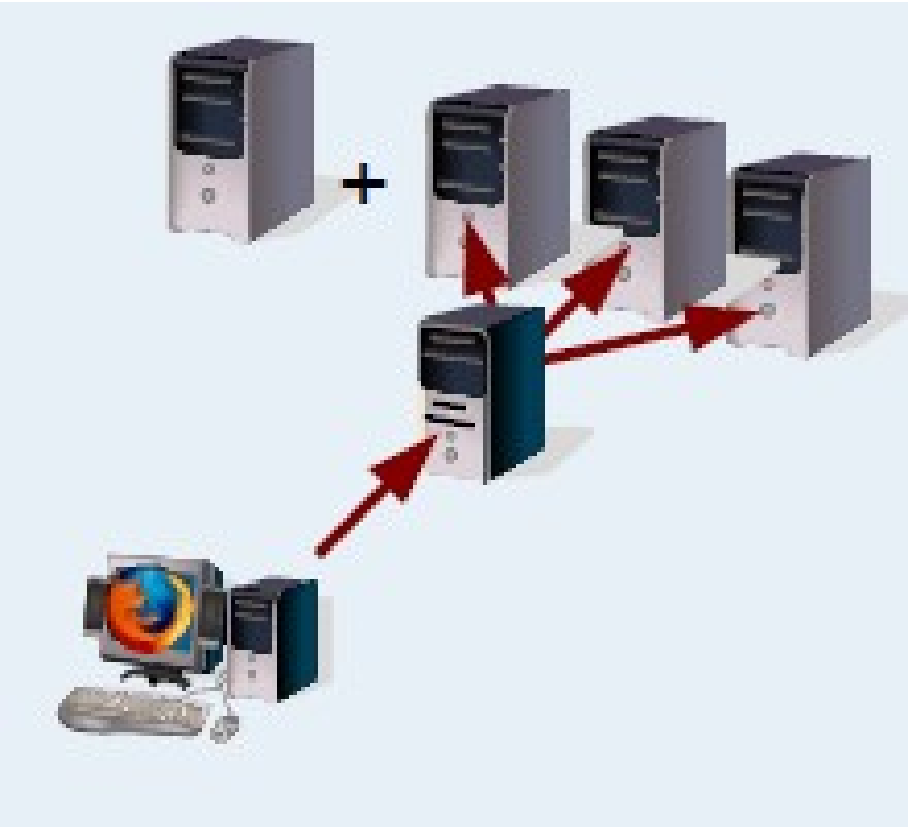
- Transparência de Replicação
  - O usuário desconhece a existência de várias cópias do recurso
  - Fundamental para desempenho e tolerância a falhas

Transparência

- Transparência de falhas
  - A presença de falhas no SD passa desapercebida pelos usuários.
  - Implica na ausência de um ponto único de falha

Transparência

- Transparência de Desempenho
  - Recurso adicionais são adicionados para suprir a nova demanda.
  - Serviços oferecisdos pela Amazon (elasticidade)



Transparência

- As duas formas mais importantes são:
  - ***acesso*** e ***localização*****!**
  - Suas presenças (ou ausências) afetam profundamente a maneira como os recursos são utilizados em um sistema distribuído
  - Também conhecidas como ***transparência de rede***
- Exemplos de falta de transparência:
  - Sistema distribuído onde só é possível acessar arquivos remotos via **FTP**
  - Serviço de jogos online que precisa ser **tirado do ar** para acrescentar ou trocar um servidor
  - Mais algum?

Transparência

- Níveis de transparência
  - **Nível do usuário**: distribuição física dos recursos é imperceptível para os usuários das aplicações (ex.: navegador da Web)
  - **Nível do programador**: distribuição física dos recursos é imperceptível tanto para os usuários quanto para os programadores das aplicações (ex.: programação com middleware ou SO distribuído)
- Importante: transparência total pode ser indesejável ou até mesmo impossível na prática!!

Ciladas

- Premissas falsas que programadores inexperientes podem adotar ao implementar um Sistema Distribuído pela primeira vez:
  - A rede é confiável
  - A rede é segura
  - A rede é homogênea
  - A topologia não muda
  - A latência é zero
  - A largura de banda é infinita
  - O custo do transporte é zero
  - Há só um administrador