



Universidade do Minho
Departamento de Informática

Engenharia de Aplicações

Arquitecturas Aplicacionais

António Nestor Ribeiro

anr@di.uminho.pt

Agenda

1. Apresentação do módulo
2. Identificação das principais preocupações
 1. Particularidades dos sistemas Web (discussão)
 2. Requisitos funcionais e não funcionais
 3. Arquitecturas típicas de sistemas web de alta escalabilidade
3. Princípio da separação de camadas
 1. Estratégias de desenvolvimento

Módulo de Arquitecturas Aplicacionais *(ou Arquitecturas Aplicacionais para sistemas multi-camada)*

- A utilização de práticas bem fundadas para a definição da camada aplicacional apresenta-se como determinante na qualidade intrínseca do sistema de software final.
- A correcta utilização das soluções padronizadas existentes e comprovadas, numa lógica de construção orientada à interconexão de componentes estanques e bem-definidos, é uma mais valia no aumento da qualidade e garantia de evolução da aplicação.
- A capacidade de construção de **aplicações complexas e de larga escala** implica uma correcta definição e programação dos serviços existentes por forma a incorporar as necessárias integrações aplicacionais a montante e a jusante do sistema.
- A capacidade de desenvolver uma aplicação com o respeito pela independência de camadas é um requisito chave para a correcta operação do sistema em período de execução, bem como um factor de automatização do desenvolvimento do mesmo.

A tecnologia

- O desenvolvimento de uma aplicação multi-camada em contexto de um servidor aplicacional, seja este **Java**, **.Net** ou outro, implica a aquisição de conhecimentos especializados na programação por objectos, nomeadamente nas frameworks e ambientes de exploração definidos pelo **JEE**.
- A construção de serviços, com a concretização aplicacional como *Web Services*, permite a disponibilização de um nível de *middleware* que facilmente pode incorporar novas funcionalidades.
- Os mecanismos de comunicação e sincronização com as camadas de dados e de apresentação são também importantes e determinam as estratégias de *caching* e lógicas de sessão a desenvolver.

Os resultados da aprendizagem

- Este tema é abordado nesta Unidade Curricular de forma a garantir que o aluno comprehende e assimila as necessidades e pressupostos base para a construção de arquitecturas aplicacionais para sistemas multi-camada e é capaz de explicar as vantagens que a abordagem orientada aos objectos, sustentada num contexto de servidor aplicacional, proporciona.
- Este conhecimento é concretizado num conjunto de **ferramentas** e **técnicas** de que o aluno passa a dispôr por forma a aplicá-las na concretização e operação de sistemas de software.

Programa

- Definição Arquitectural do Sistema de Software:
 - Patterns estruturais e de comportamento;
 - Aspectos avançados de programação orientada aos objectos;
 - Arquitecturas orientadas a serviços;
 - Modelos de programação orientados à construção de componentes reutilizáveis;
 - Manutenção evolutiva de arquitecturas orientadas aos objectos;
- Tecnologias de Programação Multi-Camada:
 - Servidores aplicacionais como contexto aplicacional;
 - Estratégias de desenvolvimento dos mecanismos de independência multi-camada;
 - Mecanismos de caching e de sessão;
 - Serviços como técnicas de integração multi-aplicação;

Competências adquiridas

- Analisar e conhecer os principais patterns estruturais e de comportamento utilizados para o desenvolvimento de sistemas de software complexo e de grande escala. Analisar as especificidades arquitecturais das aplicações multi-camada.
- Escolher os modelos de programação orientados ao objecto adequados ao problema em causa, que respeitem o levantamento de requisitos efectuado e que respeitam a necessária independência de camadas.
- Saber desenvolver camadas computacionais que permitam evolução controlada e independente das camadas de apresentação e dados e que permitam a disponibilização de serviços como mecanismo de integração.
- Identificar as principais características dos servidores aplicacionais por forma a escolher o modelo de programação pretendido. Saber utilizar tecnologia orientada a serviços como mecanismo de criação de arquitecturas de software parametrizáveis.
- Saber criar mecanismos de colocação em produção de aplicações, que sejam independentes do hardware e da configuração das máquinas alvo.

Avaliação

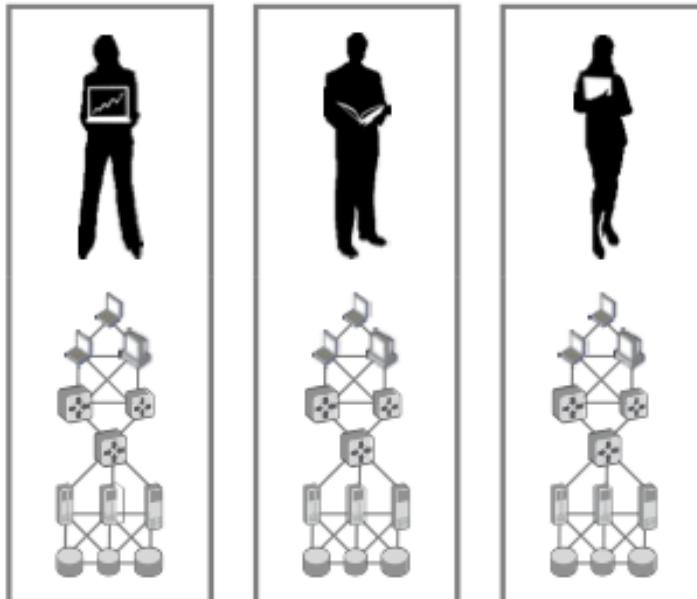
- 1 teste teórico (nota minima: 8.0) + 1 projecto em grupo
 - Data do teste teórico: [18.maio](#)
 - Data do exame teórico: [15.junho](#)
 - Projecto em grupo a ser partilhado com Sistemas Interactivos
 - Grupos com 4/5 elementos, com avaliação por pares (factor multiplicativo da nota do grupo).
 - Avaliação continua: trabalhos ao longo do semestre
- Nota Final = 40% Teste + 40% Projecto + 20% Av. Contínua
- **Projecto:** cada grupo pode sugerir um tema até dia 9.março. Caso não proponham, ou não seja validado, daremos um enunciado até 12.março.
- Entrega/apresentação a 8.junho

Sistemas Web

- Para **discussão e pesquisa**
 - o que é um sistema Web?
 - quais são as principais características (do ponto de vista de quem o constrói)?
- Requisitos funcionais e não funcionais
 - escalabilidade, qualidade de serviço, extensibilidade, etc.
- Arquitectura típicas de deployment

Evolução das aplicações

Yesterday's applications:



- Valuable, but growth is capped
- Costly, brittle, monolithic and proprietary
- Must change structure to evolve

Today's applications:



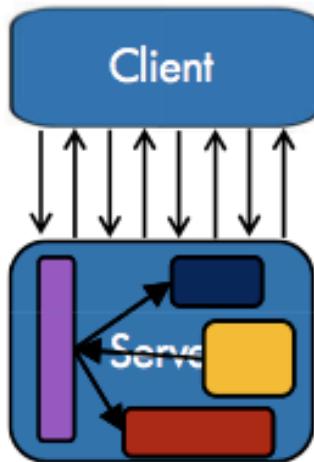
New and Agile Business Capabilities Shared Services / Composite Apps

- Next generations of SAP and Oracle based on SOA
- SOA driving new custom applications and legacy integrations
- New Web 2.0 models
- "Mashups" in the enterprise

Evolução das aplicações Web

Where we've been...

Web 1.0



Simple HTML rendering
Browser as "thin client"

Makes many calls to server
to download pages

Server has all business logic
Multiple applications
connected together
internally communicating

Simply crawl the web site

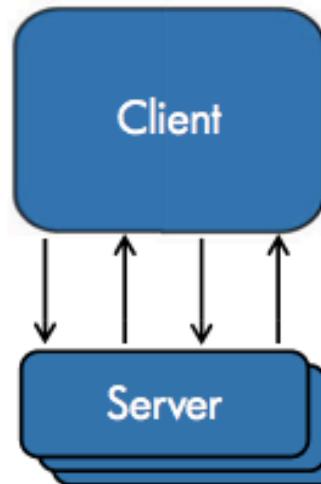
Server 'served' a series of pages with links

Business logic resided inside the organization

Browser was a 'thin' client

Where we're going...

Web 2.0



Code runs in the browser
Business logic pushed to the client
Adaptive and dynamic UI (rich UE)
Thick clients

Less network communication
Primarily focused on user input

Less load on the server
More focus on SOA and Services

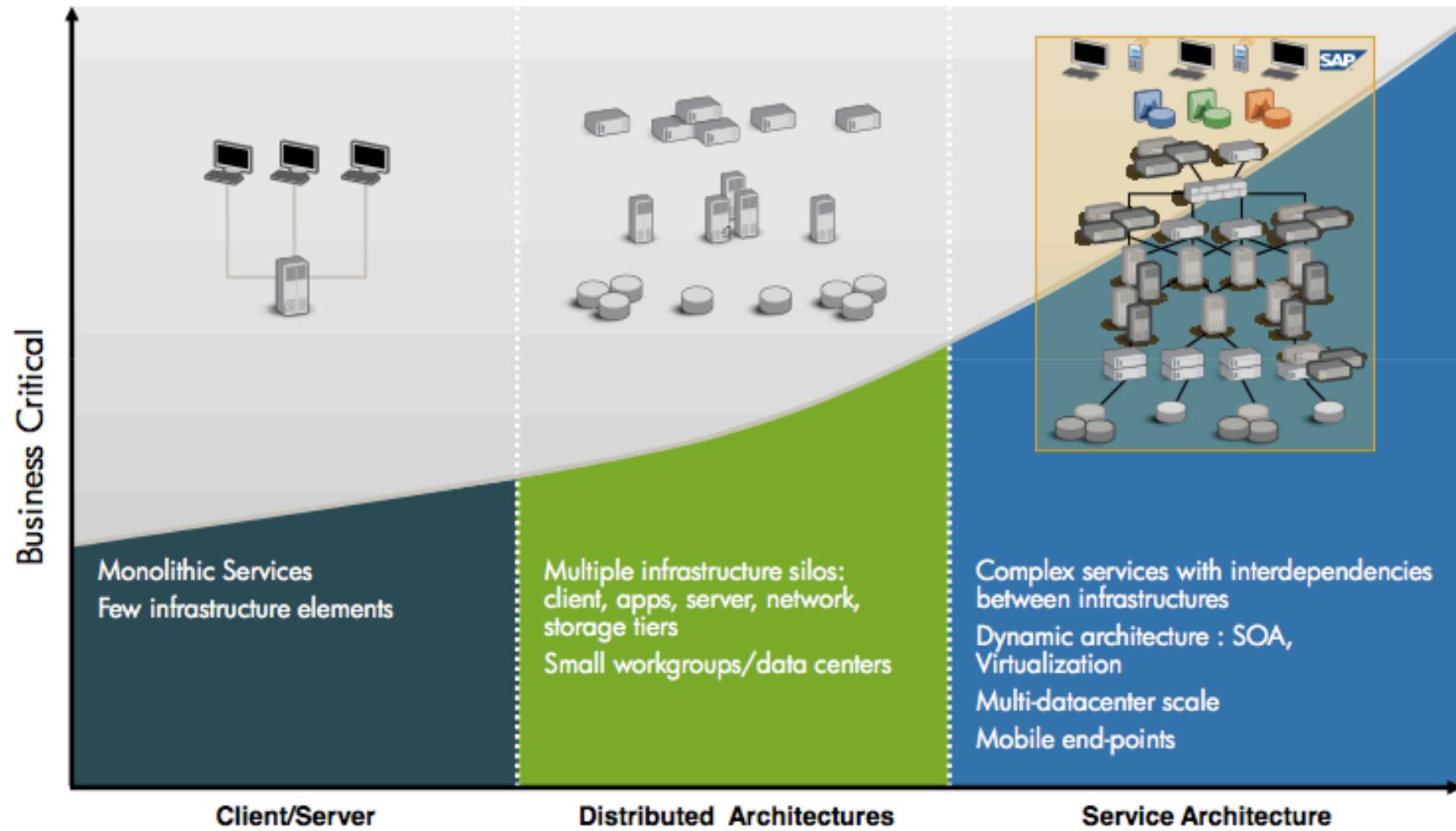
Increased application complexity

Business logic & data pushed to the client

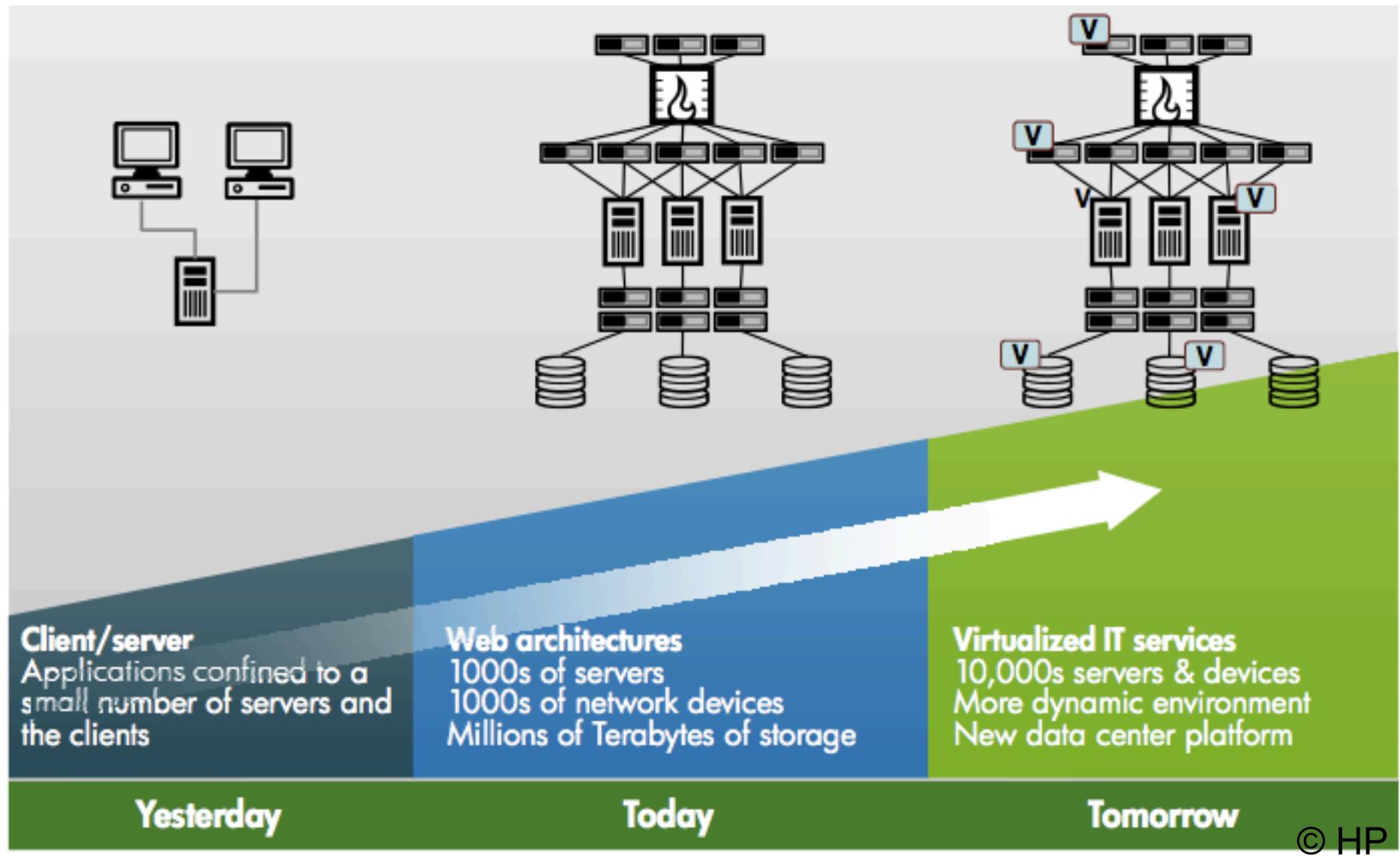
Increase attack surface

New hacking techniques

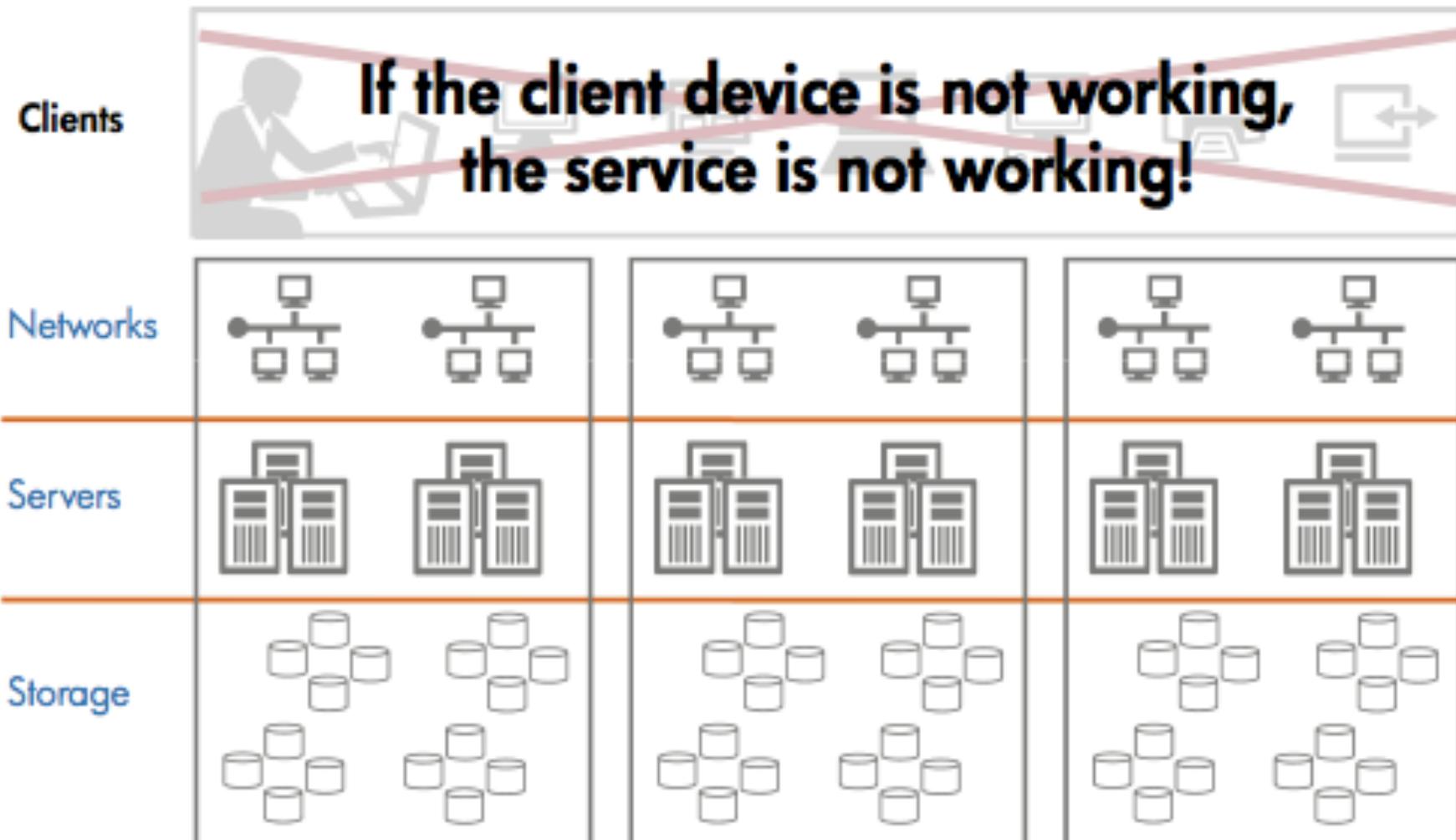
Incremento de complexidade e criticidade



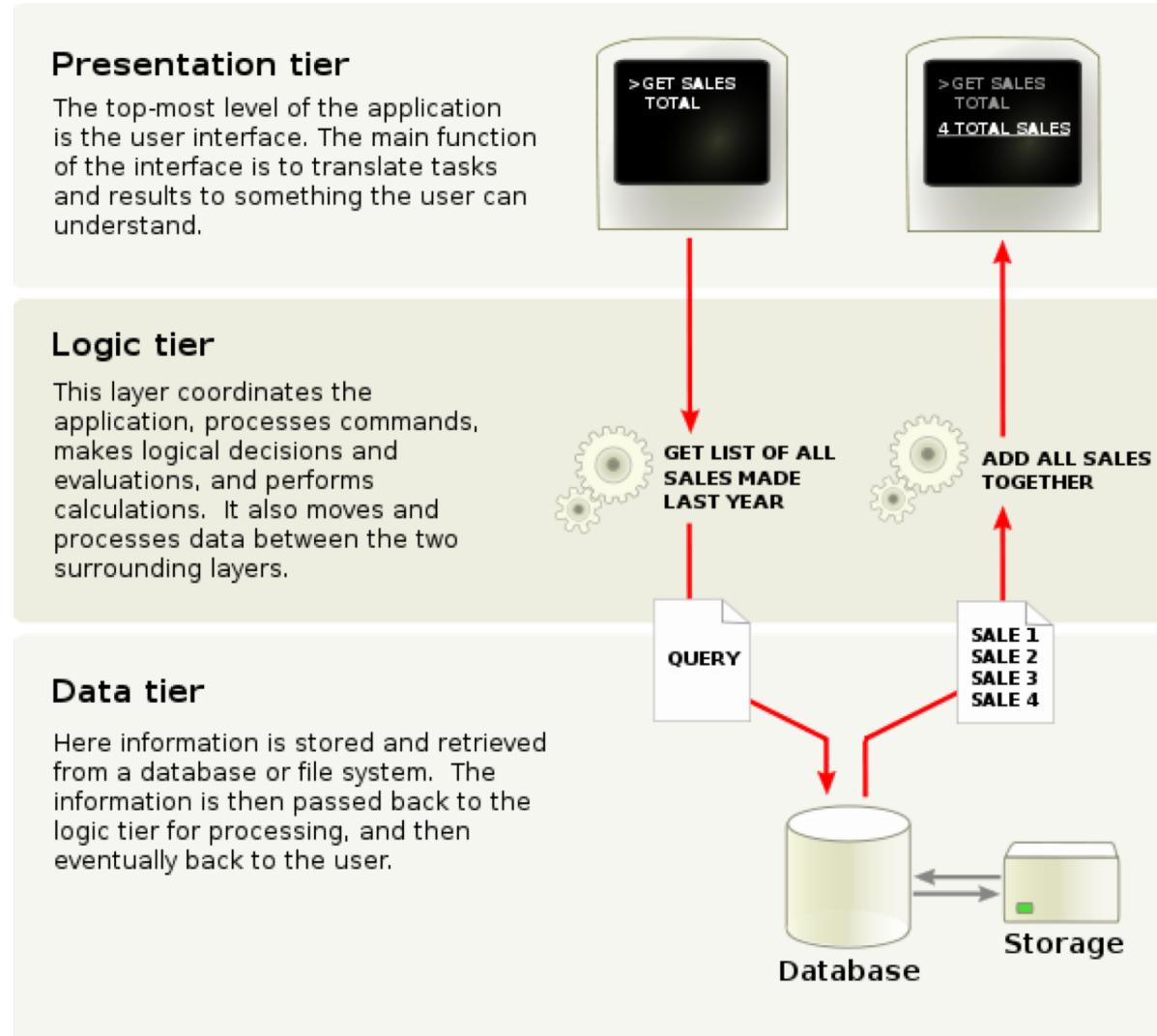
Arquitecturas mais complexas



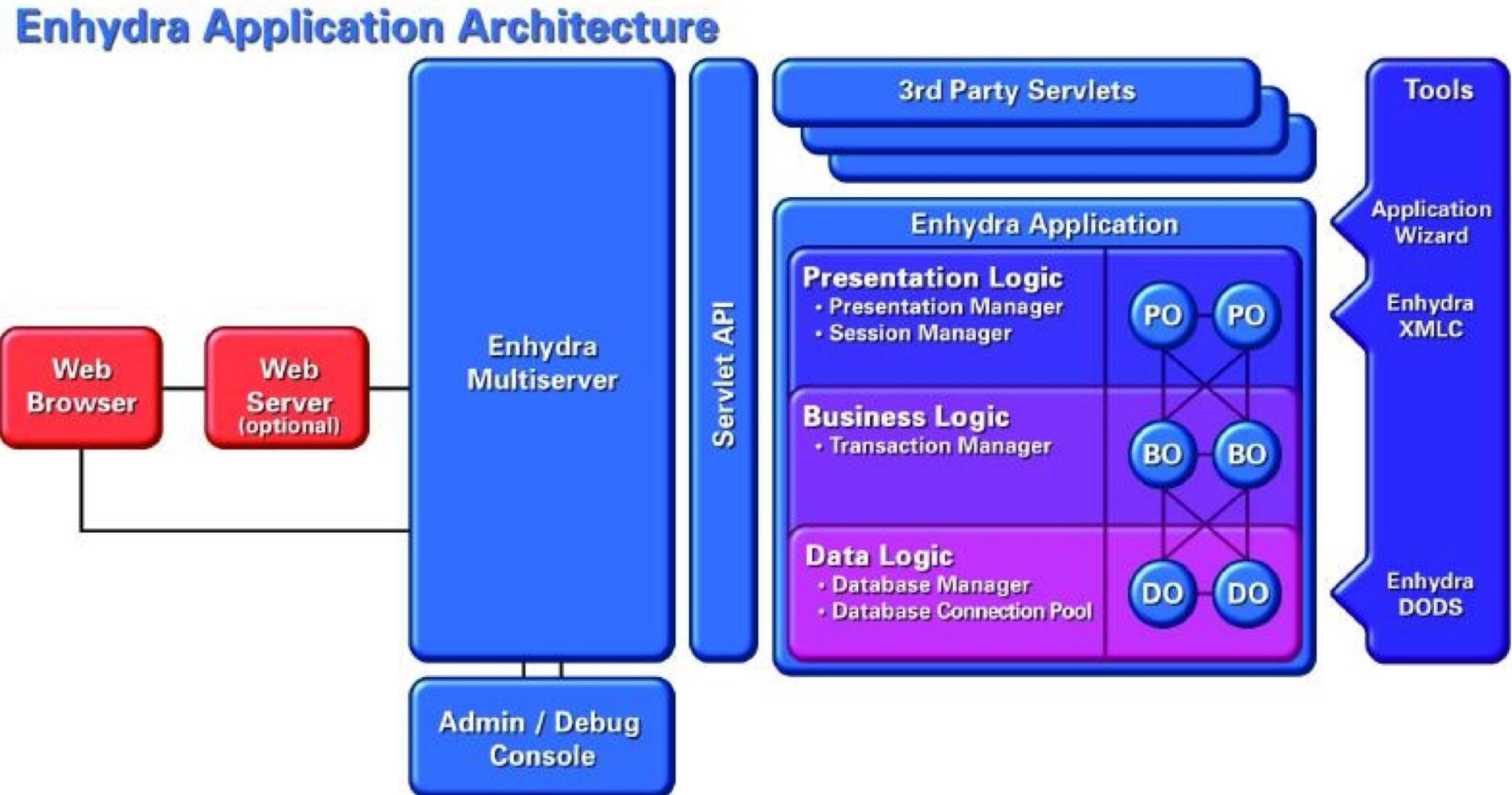
Orientação ao utilizador!



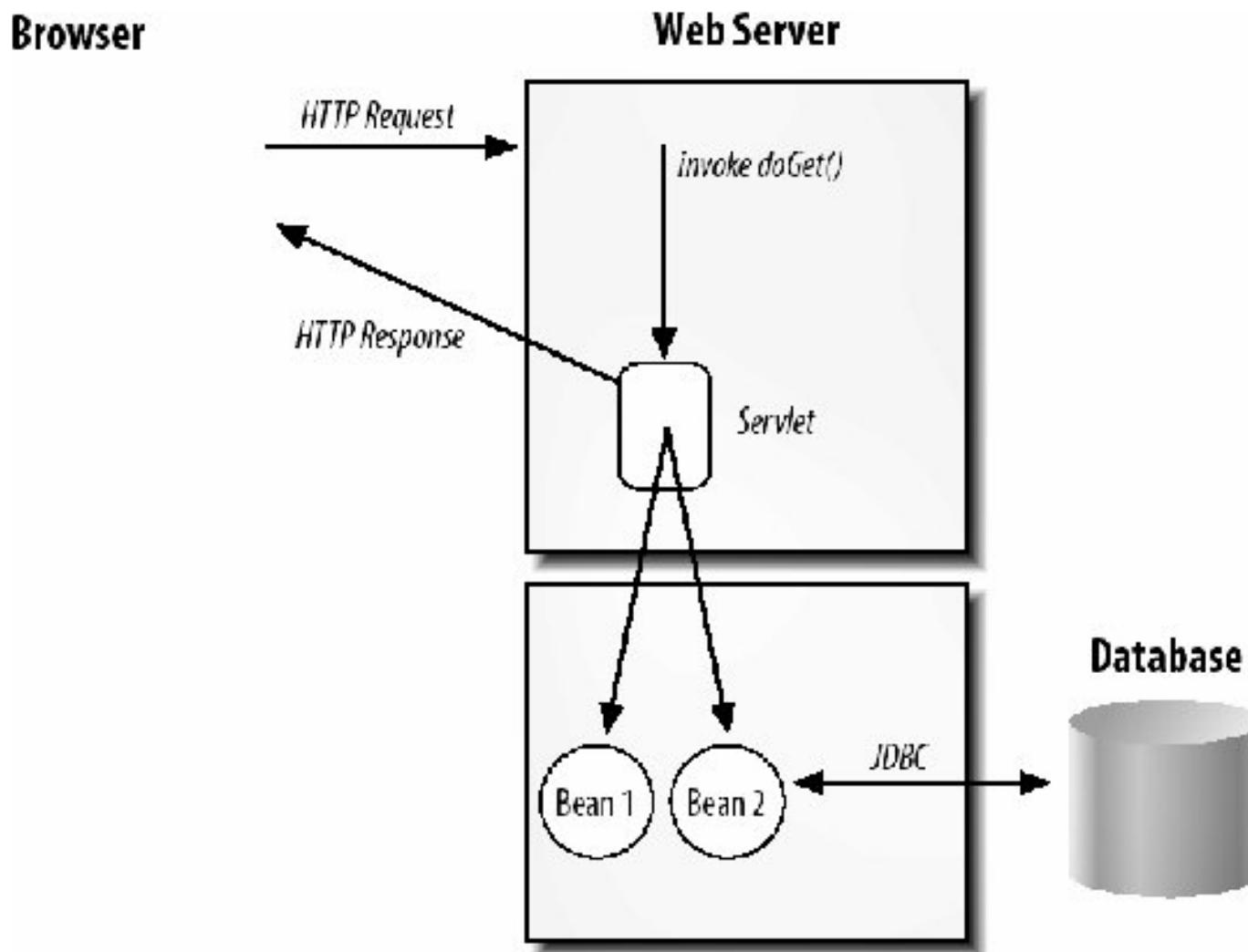
Programação por Camadas (mais refinado)



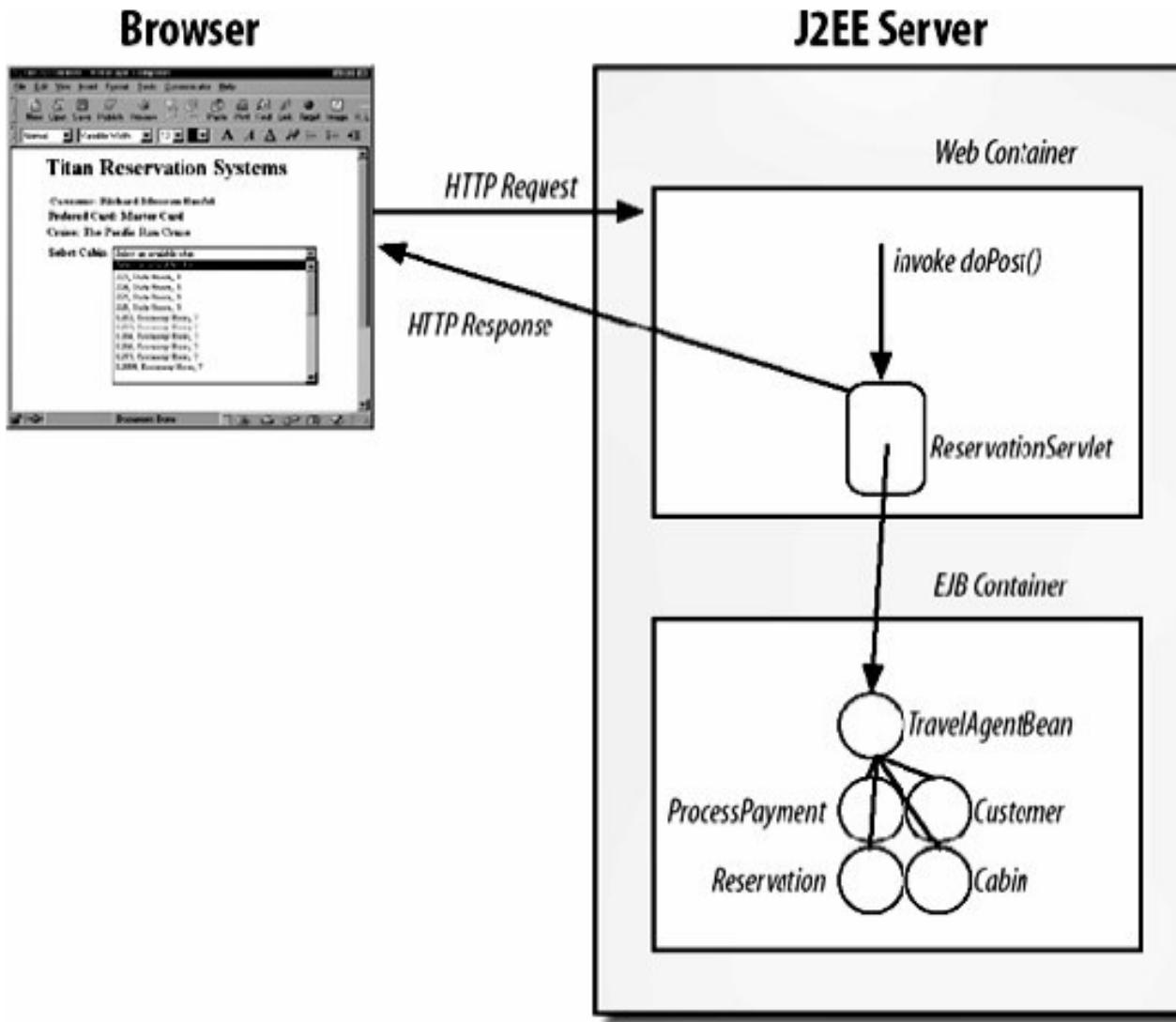
Em contexto de servidor aplicacional?



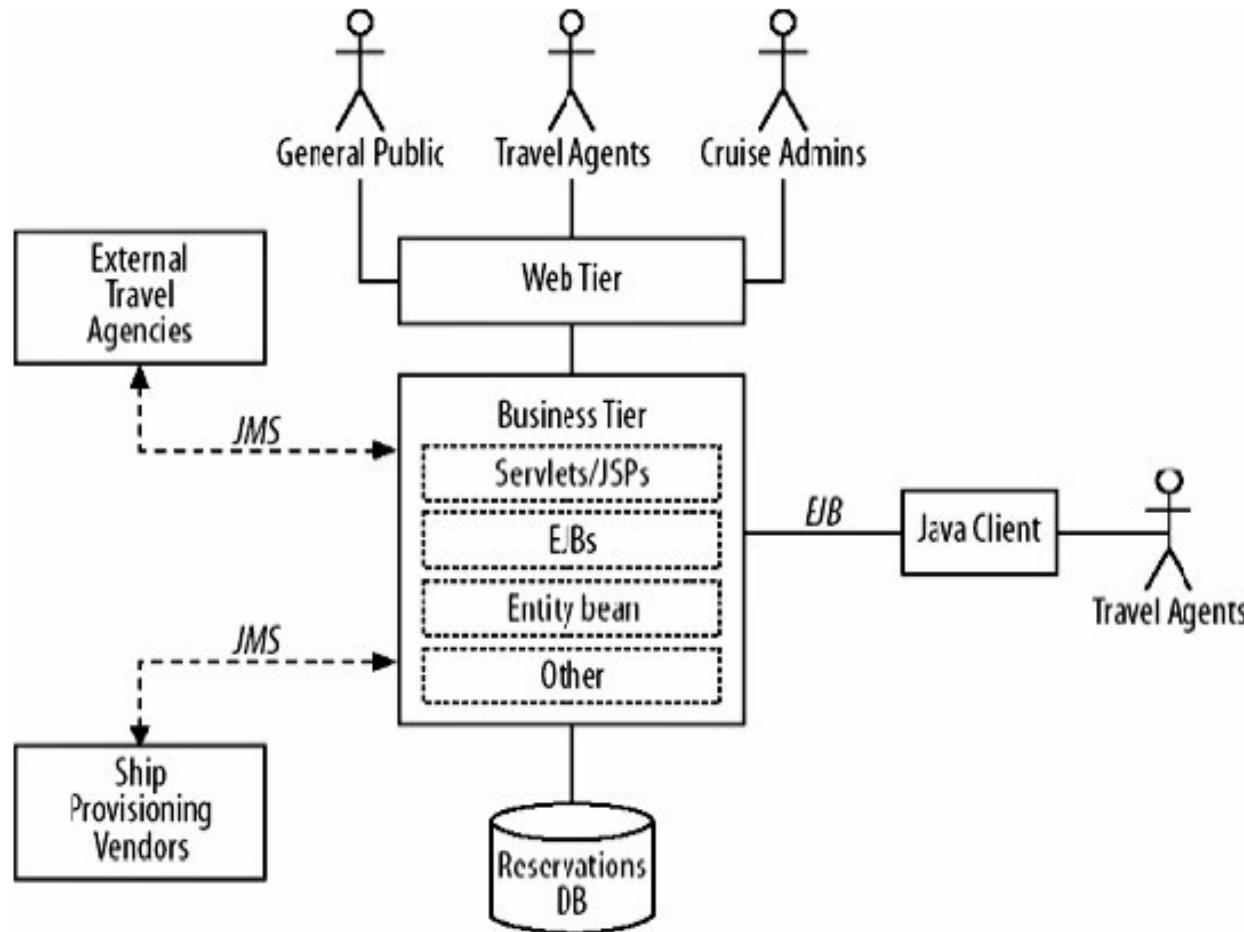
Componentes em contexto de servidor aplicacional



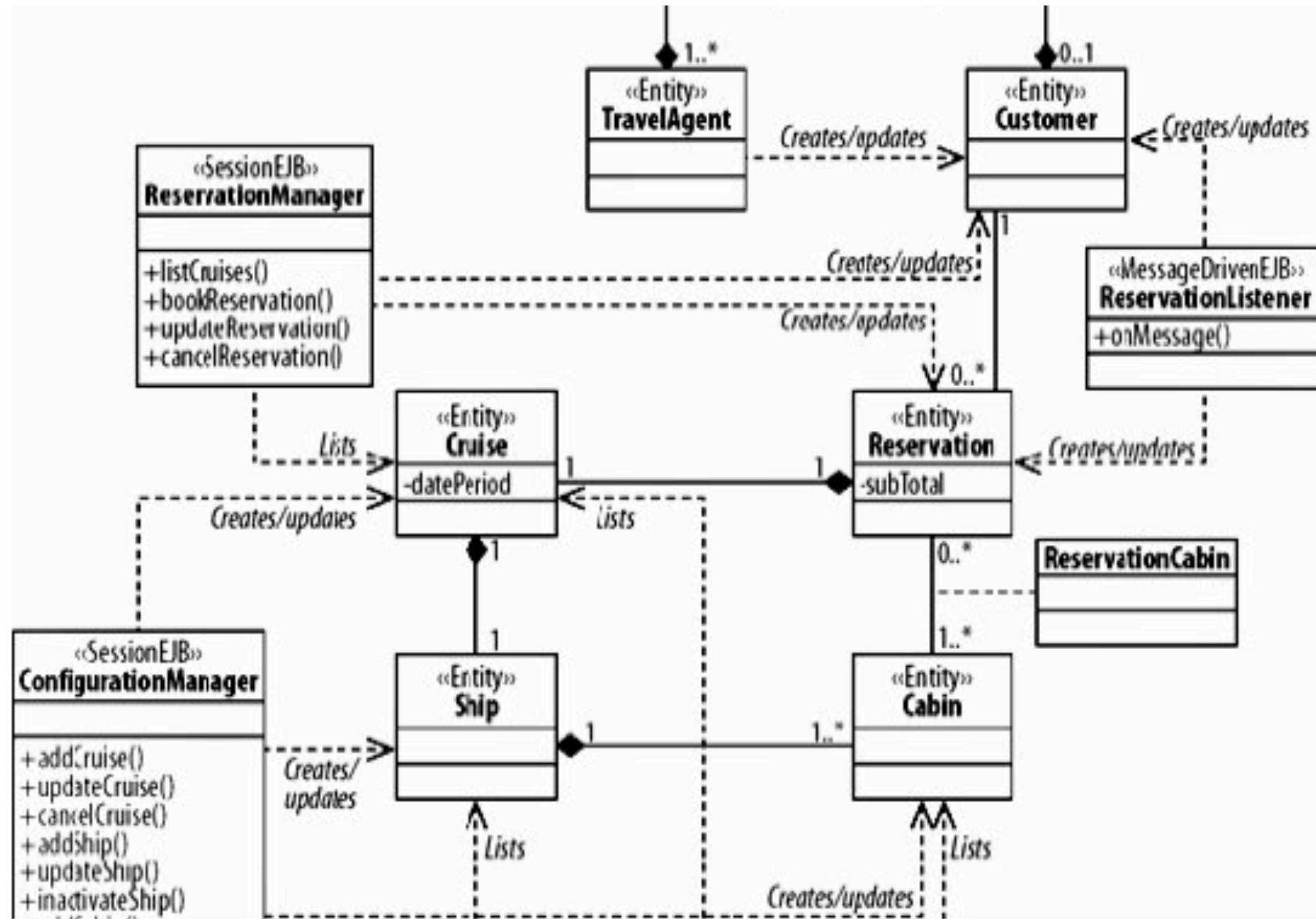
Uma aplicação multi-tier



Aplicação multi-tier - componentes

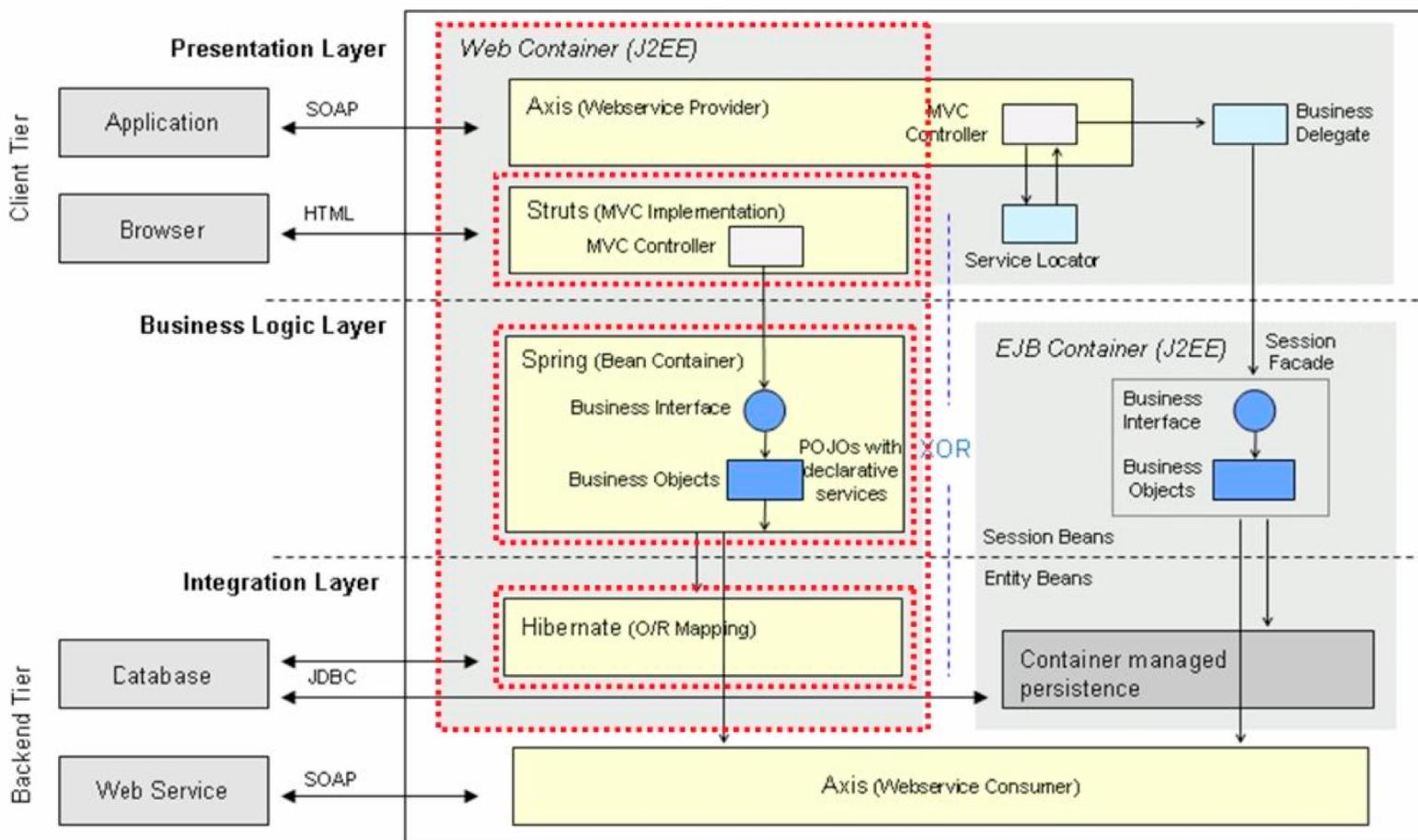


Uma abordagem baseada em modelos



Utilização de frameworks para multi-tier

- Cada equipa tende a escolher a sua estratégia



Exercício

- **Objectivo:** Pesquisa sobre frameworks de separação de camadas (Java/.NET/outras?)
- **Deliverable:** documento a entregar até 1/3 com:
 - identificação de algumas bibliotecas, para cada camada, e pontos fortes de cada framework identificada
 - proposta de uma arquitectura tipo, em função das frameworks estudadas
 - Discussão sobre as frameworks exclusivamente server side e aquelas que são híbridas (server side e client side)
 - trabalho a ser feito na aula de 23/2 e como trabalho autónomo
- **Apresentação:** 15 minutos/grupo na aula de 2/3