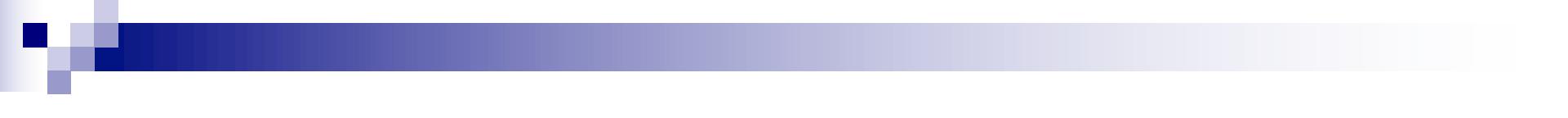




Desenvolvimento de Software para nuvem

Professor: Fernando Antonio Mota Trinta



MULTITENANCY



Contextualização

■ Software como Serviço

□ um modelo de distribuição de software em que o fornecedor armazena a aplicação na Internet, sob sua própria infra-estrutura, e a disponibiliza via browser aos usuários que executam e armazenam seus trabalhos de maneira online, e cujos serviços oferecidos são tarifados sob demanda

■ Modelo cada vez mais popular

■ Contra-ponto ao SaaP (Software as a Product)



SaaS vs SaaP

<i>SaaP</i>	<i>SaaS</i>
<i>Pacote com documentação</i>	<i>Oferta de funcionalidades, documentação online</i>
<i>Executa na Infraestrutura do cliente</i>	<i>Executa na infraestrutura do fornecedor</i>
<i>Pagamento antecipado por todas funcionalidades</i>	<i>Diferentes modelos de tarifação: taxa periódica fixa (mensalidade), quantidade de transações ou usuários</i>
<i>Disponibilizado por licenças</i>	<i>Disponibilizado como um Serviço</i>

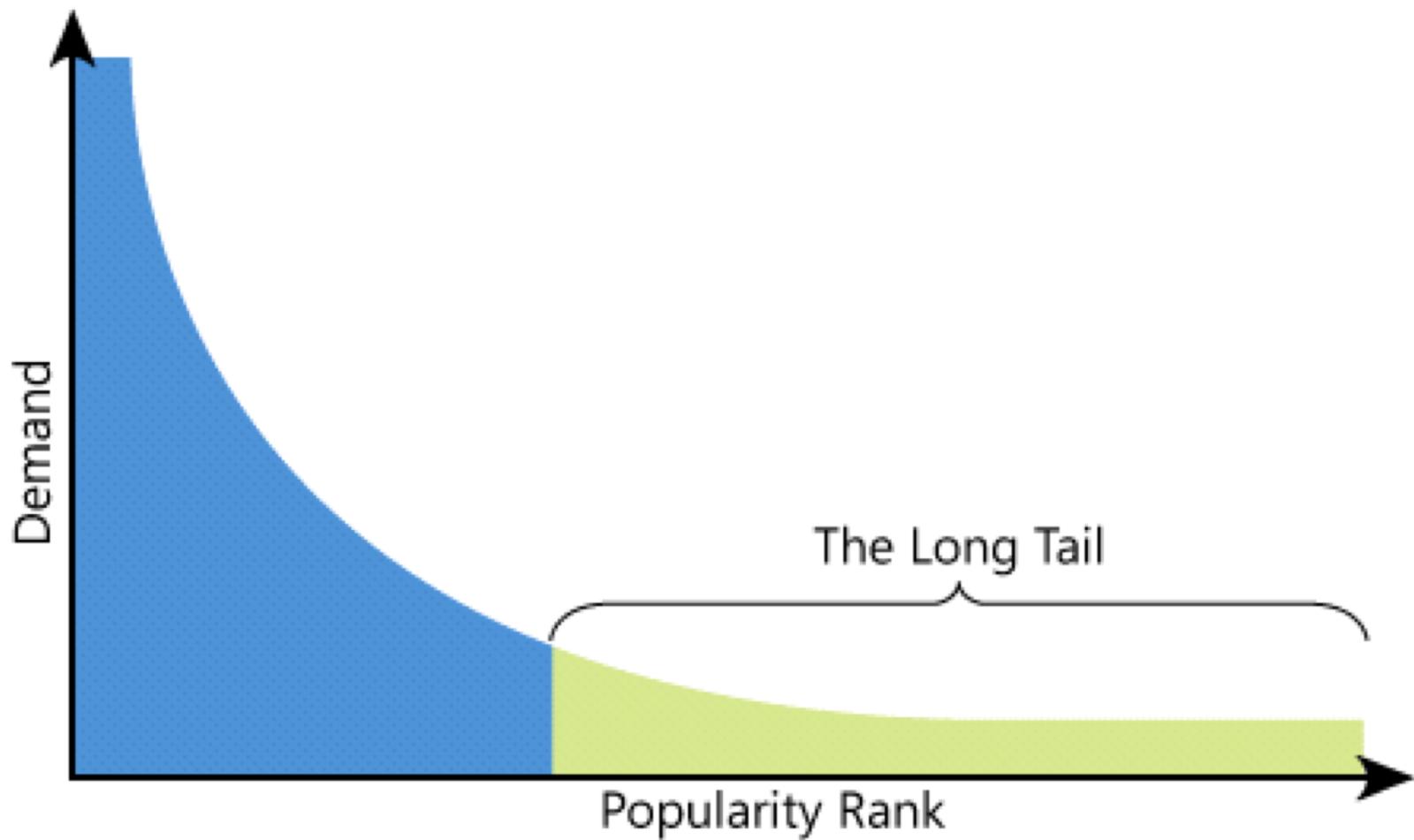


Problemas com o modelo SaaP

- Ao seguir o *modelo tradicional*, clientes enfrentam certos obstáculos
 - *Investimentos são altos*
 - *Máquinas e recursos são subutilizadas*
 - *Dificuldade com manutenção*
 - *Upgrades custosos*
- Estes problemas são especialmente problemáticos para os clientes de pequenos e médios negócios



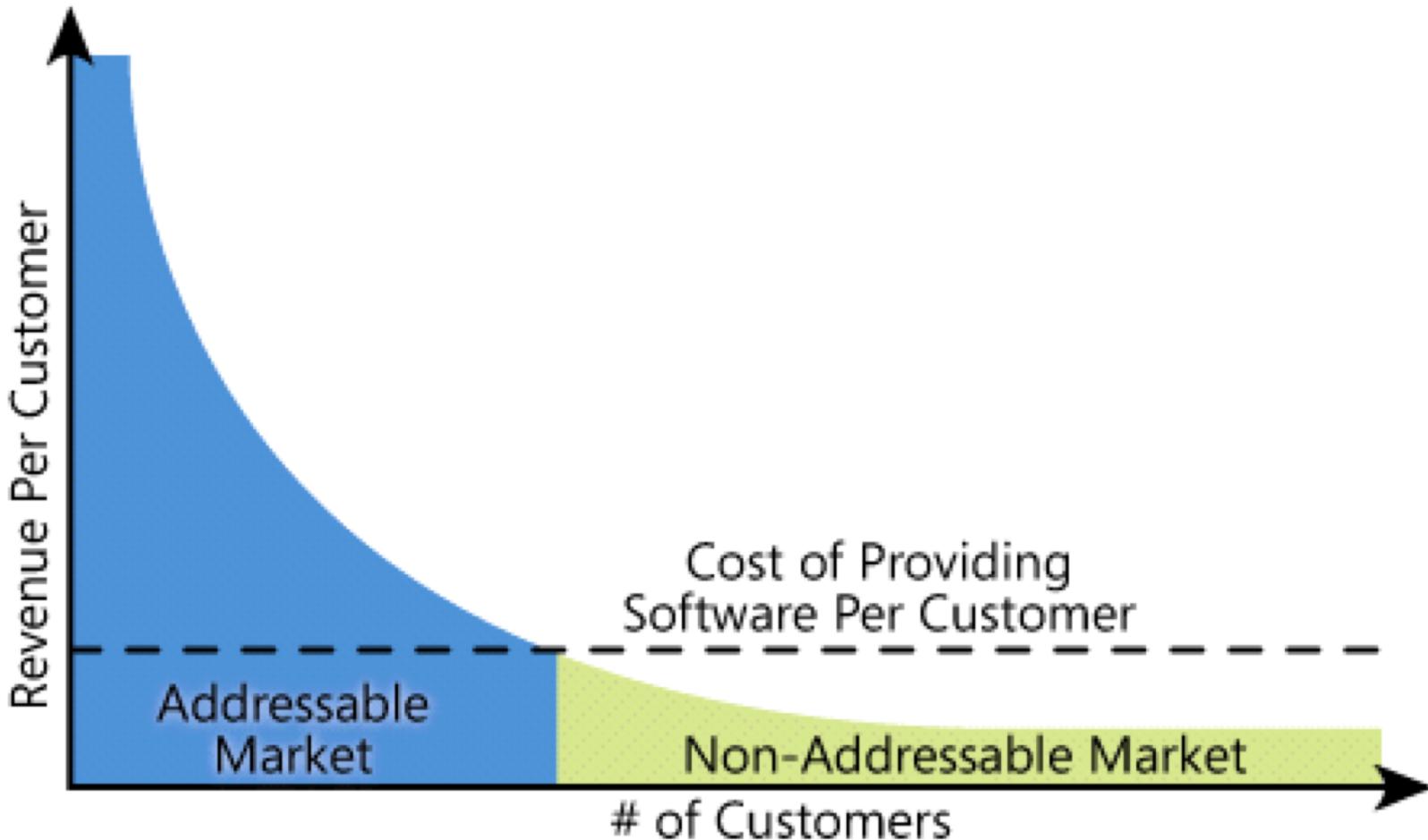
A cauda longa



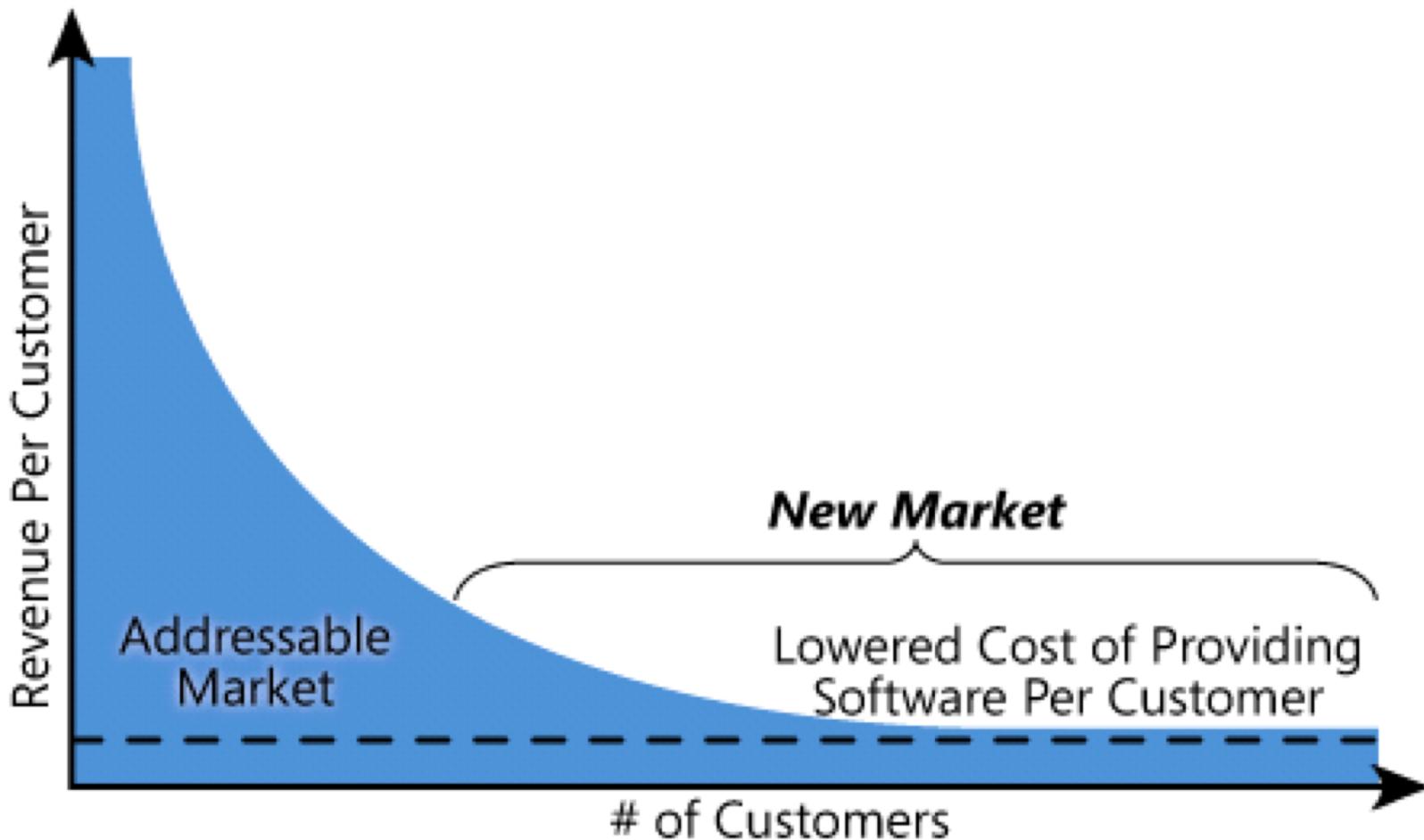
Chris Anderson. "The Long tail", Out. 2008



Cauda Longa na TI



Cauda Longa com SaaS



Então temos um problema

Desenvolver software para atender a um grande número de usuários a baixo custo, no modelo de Software como Serviço



(Estendendo)Então temos um problema

Desenvolver software para atender a um grande número de usuários a baixo custo, no modelo de Software como Serviço...

...tanto otimizando o consumo de recursos quanto reduzindo os custos



Uma analogia



Logo, nosso objetivo...



Multitenancy (Multinquilino)

- *Modelo arquitetural como aplicações que permitem o compartilhamento dos mesmos recursos de hardware, através do compartilhamento da aplicação e da instância do banco de dados, enquanto permite configurar a aplicação para atender às necessidades do cliente como se estivesse executando em um ambiente dedicado.*
- *Tenant (Inquilino) é uma entidade organizacional que aluga uma aplicação multi-tenancy. Normalmente, um tenant agrupa um número de usuários que são os stakeholders da organização.*

Bezemer, C.-P. and Zaidman, A. (2010). Multi-tenant saas applications: maintenance dream or nightmare? IWPSE-EVOL '10, pages 88–92, New York, NY, USA. ACM.



Multitenancy versus Multusuário

■ Multi-usuário

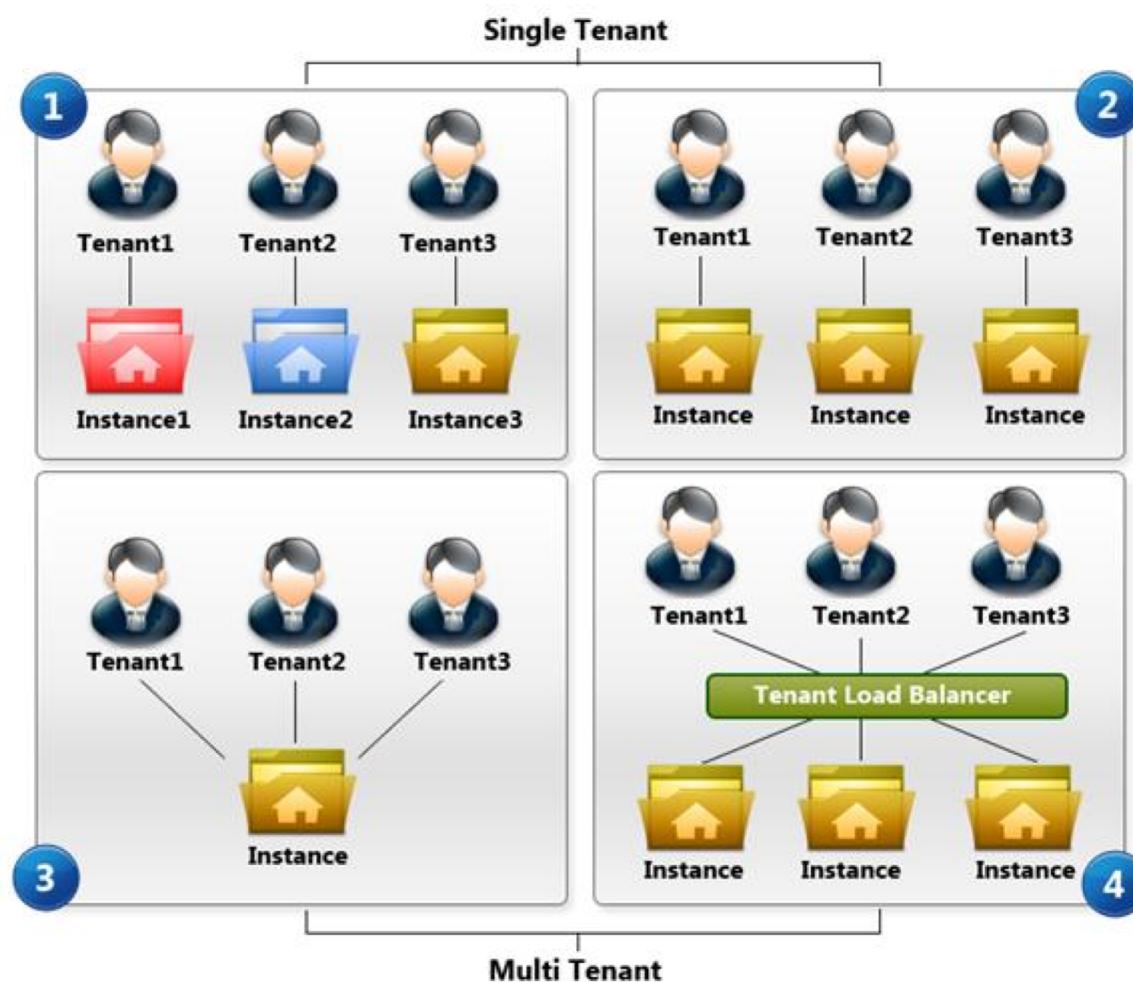
- Todos os usuários usam a mesma aplicação
- Configuração/Customização limitada

■ Multi-tenant

- Grupos de usuários usam instâncias de uma mesma aplicação
- Grande poder de configuração/customização
 - Interface Gráfica (por exemplo)
 - Workflow
 - Acordos de níveis de serviço



Modelos de Maturidade SaaS



Maturidade SaaS (1)

Inquilino Isolado ou Adhoc

- Chamado de Ad-hoc
- Sem compartilhamento
- Cada inquilino tem sua pilha de tecnologia
- Similar ao hosting
- Sem elasticidade/agilidade



Maturidade SaaS (2)

Multi-inquilino via hardware compartilhado (virtualização)

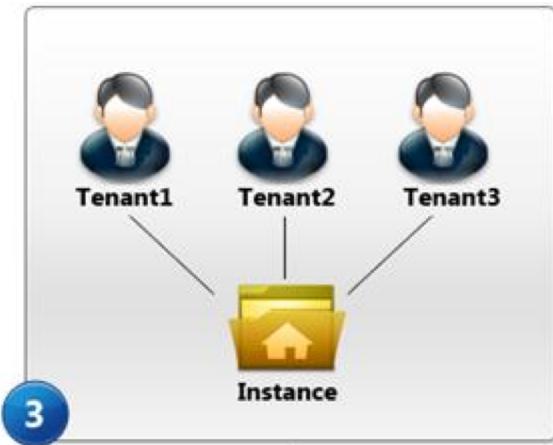
- Chamado também de nível configurável
- Semelhante ao I, mas via virtualização
- Cada inquilino tem sua pilha de tecnologia
- Elasticidade na camada de hardware
- Entrada rápida na computação em nuvem
 - Sem reprojeto da aplicação
- Ainda limitado



Maturidade SaaS (3)

Multi-inquilino via container

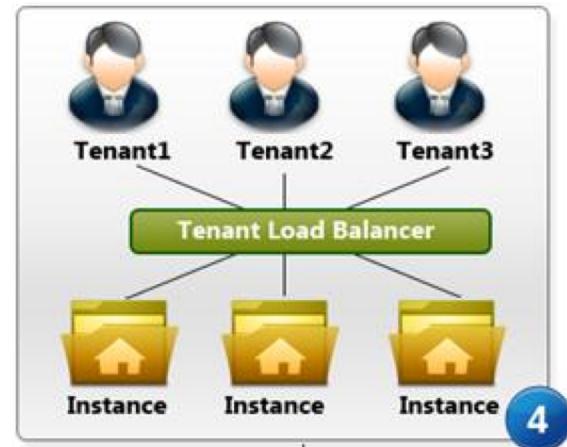
- *Configurável & multi-tenant eficiente*
- *Vários inquilinos em um mesmo container*
- *Bancos de dados separados*
- *Elasticidade e customização*
- *Necessidade de reprojeto da aplicação*



Maturidade SaaS (4)

Multi-inquilino via todo o stack de software compartilhado

- Escalável, Configurável e MultiTenant Eficiente
- Evolução do 3
- Todo o stack é compartilhado
- Única instância de Banco



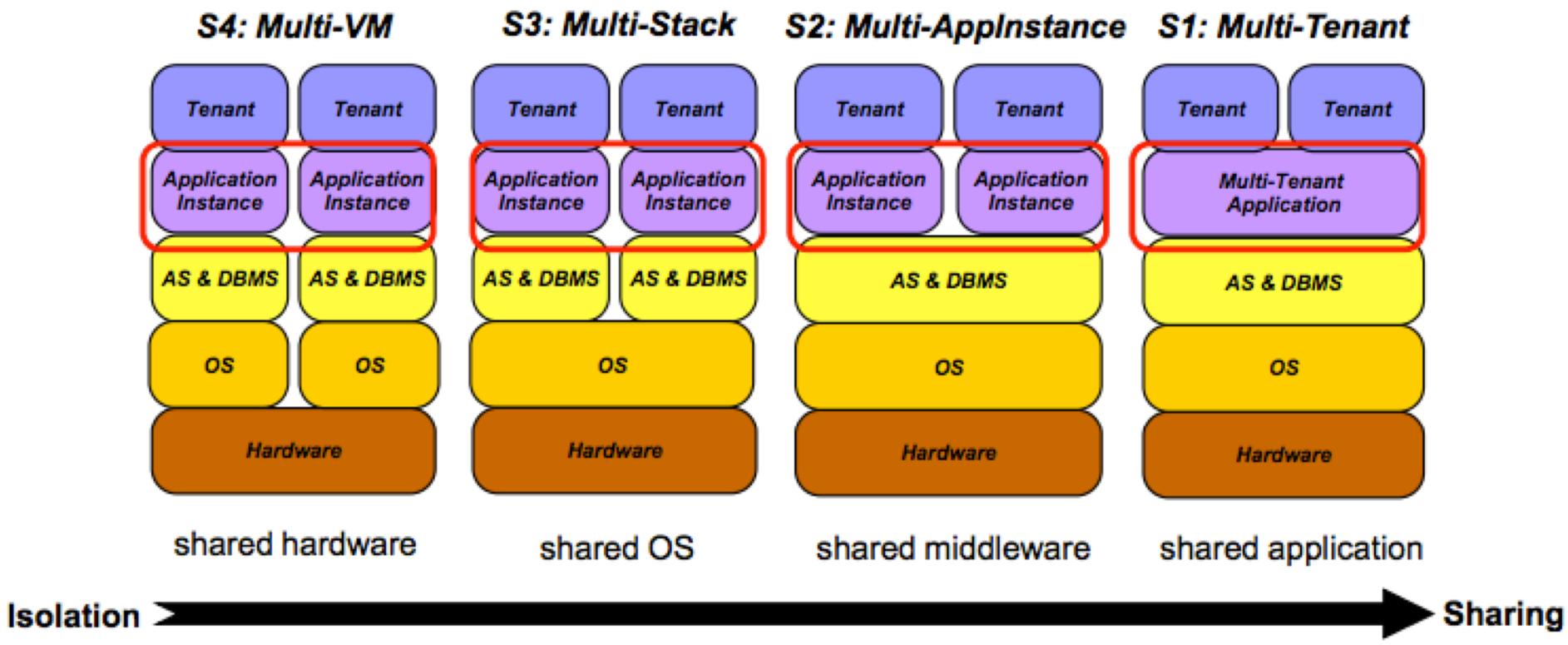
Comparação entre os níveis

- *Modelo 2 permite uma transição SaaS de baixos custo e impacto*
- *Modelos 3 e 4 são mais avançados*
 - *Tendem a ser os predominantes*
 - *Problema: softwares legados*



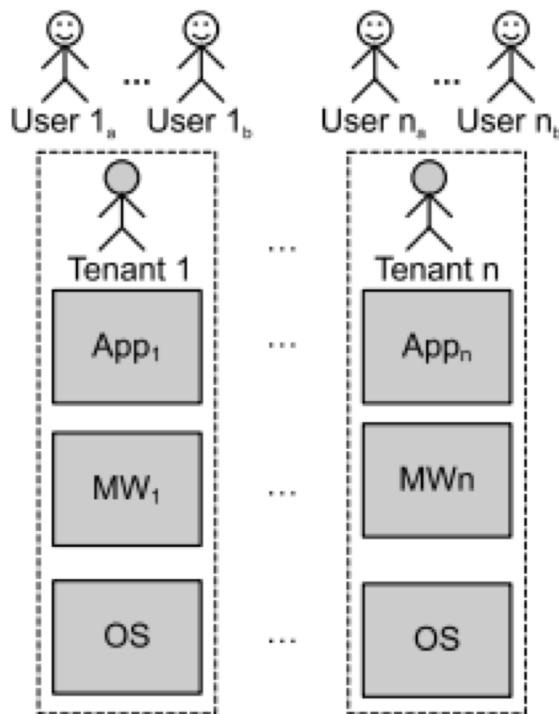
Outra visão

Multi-instance single-tenant applications vs. single-instance multi-tenant applications

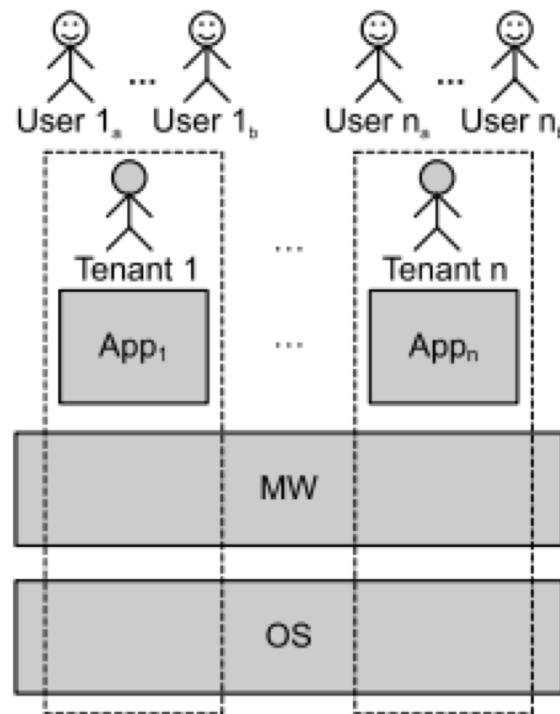


Outra visão

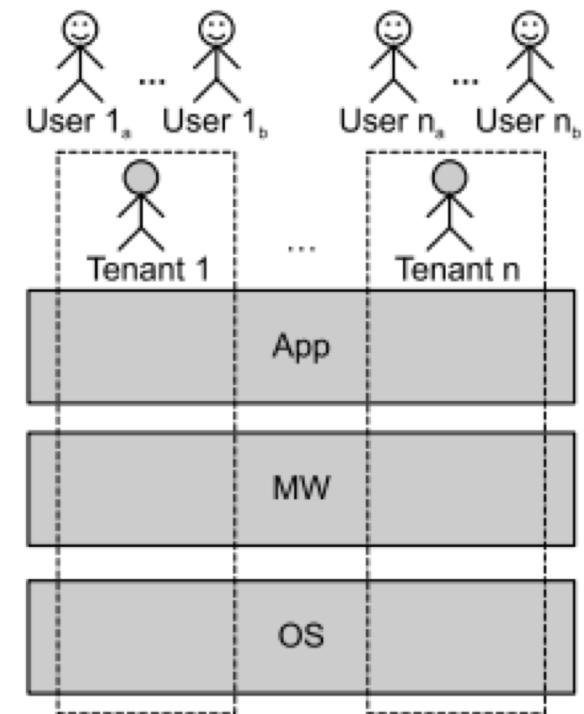
(1) Shared infrastructure



(2) Shared middleware



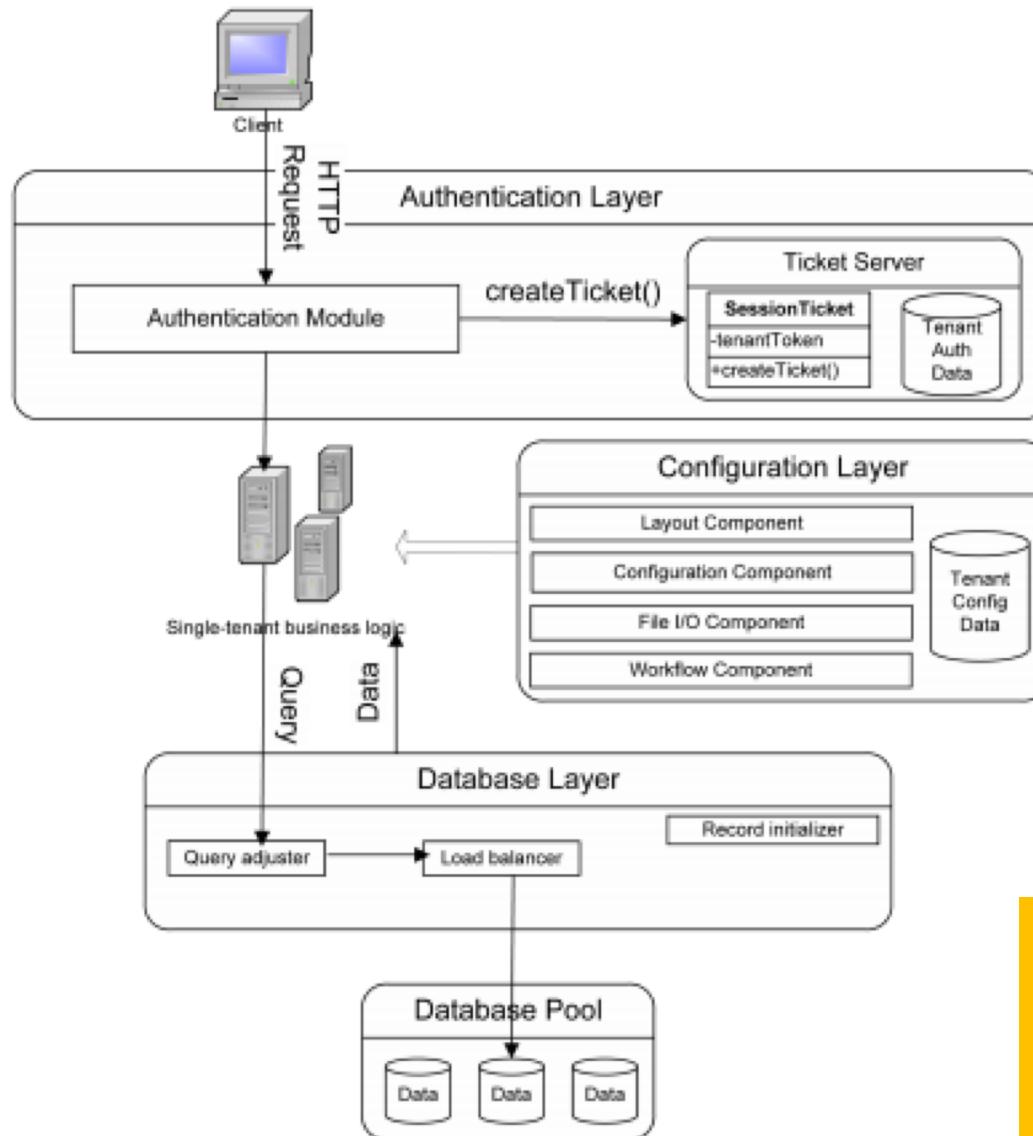
(3) Shared application



Improved scalability, lower operational costs

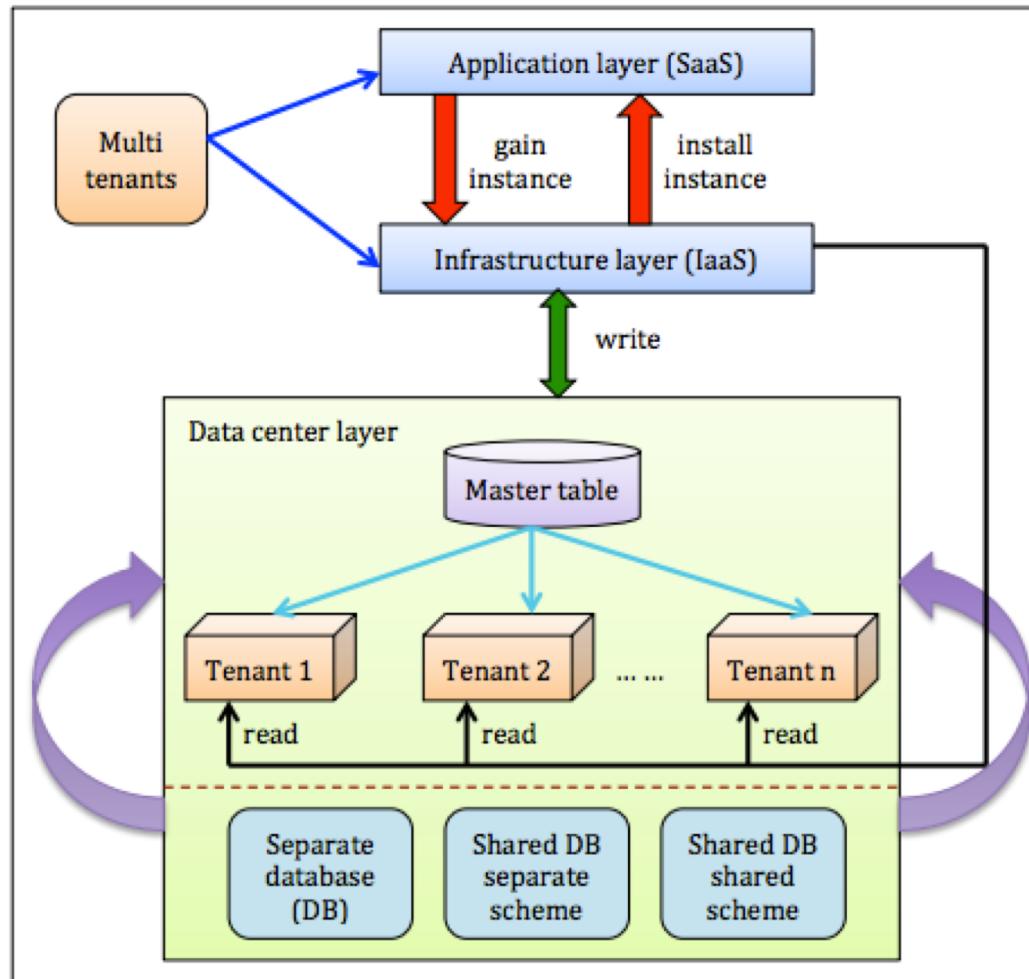
Higher flexibility and isolation, lower engineering complexity

Uma proposta de Arquitetura



Bezemer, C.-P. and
Zaidman, A. Challenges
of Reengineering into
Multi-Tenant SaaS
Applications

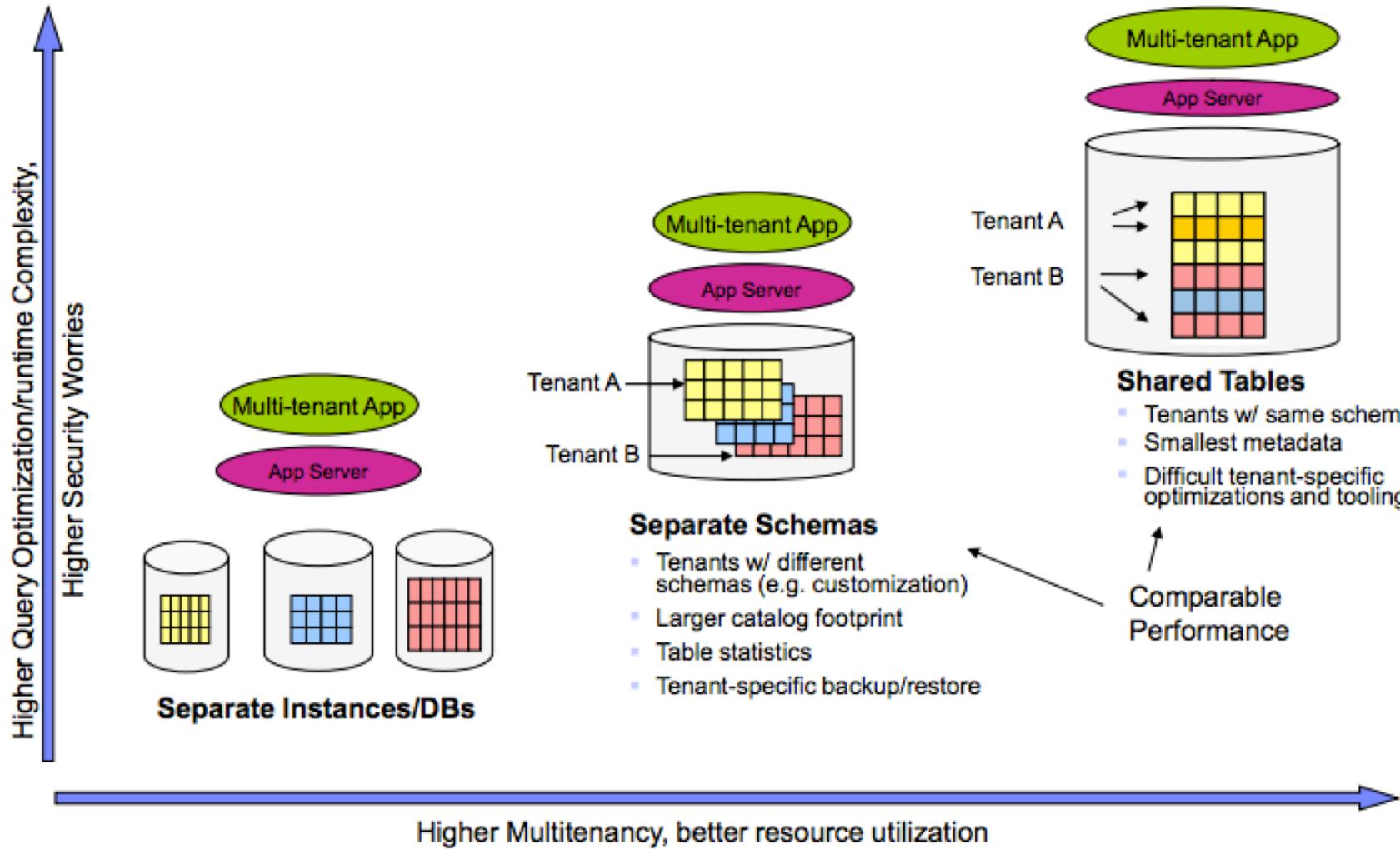
Outra proposta



Jia Ru et al.. 2014. Software engineering for multi-tenancy computing challenges and implications.
In Proceedings of InnoSWDev 2014. ACM, New York, NY, USA, 1-10.



Modelos Multi-Tenancy para Bancos de Dados



Tradeoffs



Benefícios Multitenancy

- *Único código fonte para várias versões da aplicação*
- *Atualização do software de uma só vez para todos os clientes*
- *Facilidade de colaboração e integração entre tenants*
- *Economia nos custos com infraestrutura de hardware*
- *Aumento da margem de lucro*
- *Reuso de regras de negócio com o mínimo de adaptação*
- *Redução dos custos de venda e manutenção do software*
- *Agrupamento de tenants de acordo com a SLA exigida pelo cliente*



Desvantagens

- É difícil calcular os recursos requeridos para cada novo tenant e ao mesmo tempo garantir que as restrições de todos os outros tenants sejam atendidas
- Dificuldade de comparar e otimizar a redução de custos das diferentes formas de distribuição dos tenants entre os servidores, pelo fato de existirem várias variáveis envolvidas



Desvantagens

- *Preocupação das empresas com o custo inicial de reestruturas suas aplicações legadas para multi-tenancy*
- *Preocupação dos mantenedores de software com a possibilidade de multi-tenancy introduzir problemas adicionais de manutenção decorrentes do fato desses novos sistemas serem altamente customizáveis*



Desafios de Pesquisa

- *Alta Disponibilidade, Queda e Balanceamento de Carga*
 - *Alto número de instâncias/bancos de dados*
- *Isolamento: dado, desempenho, manutenção*
- *Customização*
- *SLAs para disponibilidade e desempenho*
- *Benchmarks*
 - *Consumo de recursos/tenant, custo/tenant*

