Identificação rápida de gargalos — Uma forma mais eficiente de realizar testes de carga

Um artigo técnico da Oracle
Junho de 2009

ORACLE

Identificação rápida de gargalos — Uma forma mais eficiente de realizar testes de carga

INTRODUÇÃO

Você está pronto para dar início a uma aplicação Web crítica. Garantir o bom desempenho da aplicação é fundamental, mas o tempo é curto. Como você pode testar a aplicação da melhor forma e ainda atender os prazos?

A Identificação rápida de gargalos (RBI) combina um entendimento abrangente de gargalos com uma metodologia refinada de testes de carga que permitem que as organizações criem aplicações Web altamente redimensionáveis.

A Identificação rápida de gargalos (RBI) é uma nova metodologia de testes que permite que os profissionais de controle de qualidade (QA) descubram rapidamente as limitações de desempenho da aplicação Web e determinem o impacto dessas limitações na experiência do usuário final. Desenvolvida durante anos de envolvimento com testes em todos os tipos de plataformas, a metodologia RBI reduz drasticamente os ciclos de testes de carga ao mesmo tempo em que possibilita uma quantidade maior de testes (e testes mais minuciosos). Através dessa abordagem, as organizações podem aumentar a qualidade das aplicações, melhorar a experiência do usuário e reduzir o custo de implantação de novos sistemas.

DEFINIÇÃO DE TESTES DE DESEMPENHO

Os testes de desempenho podem ser grosseiramente definidos como "testes realizados para avaliar a conformidade de um sistema ou componente com requisitos específicos de desempenho". Entretanto, todas as aplicações têm pelo menos um gargalo e poucos sistemas (caso haja algum) atendem aos requisitos iniciais de desempenho. Para ilustrar essa realidade, vamos redefinir os testes de desempenho como "testes realizados para isolar e identificar os problemas do sistema e da aplicação (gargalos) que impedirão que a aplicação seja redimensionada para atender seus requisitos de desempenho".

Essa mudança filosófica no ponto de vista (de testes como uma avaliação para testes como uma investigação ativa para isolar e resolver problemas) é o que direcionou a criação da metodologia RBI. A RBI combina um entendimento abrangente de gargalos com uma metodologia refinada de testes que permite que as organizações criem aplicações Web altamente redimensionáveis.

ENTENDENDO OS GARGALOS, O THROUGHPUT E A SIMULTANEIDADE

Antes de se aprofundar nos detalhes da metodologia RBI, precisamos estabelecer um entendimento comum sobre os gargalos (e onde eles são encontrados), bem como fazer uma distinção entre teste de throughput e teste de simultaneidade.

Gargalos — Principais inibidores de desempenho

Qualquer recurso do sistema, como hardware, software ou largura de banda, que estipule limites definidos no fluxo de dados ou na velocidade de processamento cria um gargalo. Em aplicações Web, os gargalos afetam diretamente o desempenho e a escalabilidade limitando a quantidade de throughput de dados ou restringindo o número de conexões da aplicação. Esses problemas ocorrem em todos os níveis da arquitetura do sistema, incluindo a camada de rede, o servidor Web, o servidor de aplicações e o servidor de banco de dados. Historicamente, com base em nossa experiência de testes em aplicações reais de clientes, os gargalos aparecem distribuídos por esses componentes conforme mostrado na Figura 1.

Em aplicações Web, os gargalos afetam diretamente o desempenho e a escalabilidade limitando a quantidade de throughput de dados ou restringindo o número de conexões da aplicação.

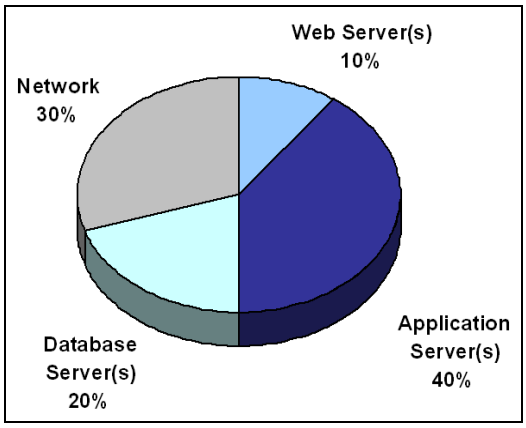

Figura 1. Distribuição estimada de gargalos pela infraestrutura do sistema.

O impacto composto da complexidade dos testes

A abordagem de testes que você escolhe influencia diretamente a dificuldade de isolar e resolver os gargalos. Infelizmente, muitos procedimentos de testes começam com cenários complexos de utilização onde os testadores tentam simular exatamente como a aplicação será utilizada na produção. Isso pode envolver a execução de diversas transações diferentes para simular os diferentes tipos de usuários que interagem com a aplicação de diversas formas. Infelizmente, isso cria uma barreira de testes significativa, pois os cenários mais complexos e que envolvem diversas transações diferentes inserem mais gargalos no teste, o que dificulta a identificação das causas principais.

Por exemplo, o gráfico na Figura 2 ilustra os resultados de teste de uma aplicação de e-commerce padrão que apresentou um gargalo com aproximadamente 2.000 usuários simultâneos. Neste exemplo de teste, os cenários de utilização envolvem

navegação, pesquisa e adição de itens em um carrinho de compras para concluir uma compra. Embora houvesse somente três transações sendo testadas, cada transação interagiu com todos os níveis da arquitetura da aplicação, e qualquer uma delas poderia ter causado o gargalo. Para complicar ainda mais, o gargalo também poderia ter sido causado por um problema no sistema. No final das contas, quanto mais variáveis estão envolvidas em um teste, mais difícil é determinar a causa do problema.

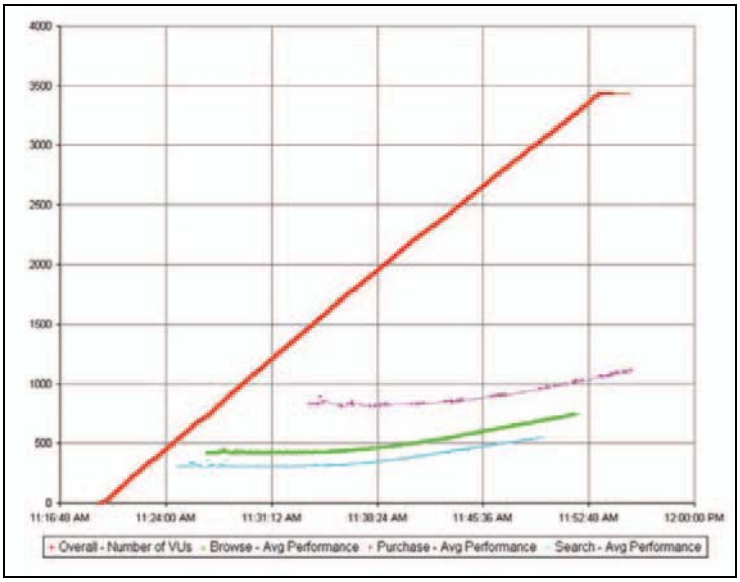

Figura 2: Resultados gráficos do teste de uma aplicação de e-commerce.

Se o problema pode estar em qualquer camada da arquitetura e a probabilidade de estar em uma determinada camada não é significativamente maior que a probabilidade de estar em outra, onde mais você pode procurar orientação?

Dois problemas principais: Throughput e Simultaneidade

A maior parte de todos os problemas de desempenho de sistemas e aplicações é resultado de limitações no throughput.

O throughput é a quantidade de fluxo de dados que um sistema pode suportar, medido em acessos por segundo, páginas por segundo e megabits de dados por segundo. A simultaneidade é o número de usuários independentes conectados simultaneamente e usando uma aplicação. Pela nossa experiência, a maior parte de todos os problemas de desempenho de sistemas e aplicações é resultado de limitações no throughput. Entretanto, problemas de simultaneidade também são críticos para o desempenho de uma aplicação e podem ser até mais difíceis de isolar.

Testes de throughput

Os testes de throughput envolvem a redução até o mínimo de conexões de usuários de um sistema e o aumento até o máximo da quantidade de trabalho sendo realizado por esses usuários. Isso empurra o sistema e a aplicação em termos de capacidade para que todos os problemas sejam revelados.

Para testes de throughput no nível do sistema, arquivos básicos podem ser adicionados aos servidores Web e de aplicações para a realização dos testes. O teste de carga pode então ser configurado para solicitar que esses arquivos de teste avaliem o throughput máximo do sistema em cada camada. Normalmente, os testadores usam um arquivo grande de imagem para testes de largura de banda, um arquivo pequeno de texto ou imagem para testes de taxa de acessos e uma página extremamente simples da aplicação (uma página “Hello World”, por exemplo) para testar a taxa de páginas. Se o sistema não atender aos requisitos básicos de desempenho da aplicação (apenas solicitando essas páginas simples) os testes devem ser interrompidos até que o sistema em si tenha sido melhorado, seja através do ajuste das configurações, o aumento da capacidade de hardware ou o aumento da largura de banda alocada.

Os testes de throughput da aplicação real envolvem o acesso a páginas chave e transações de usuários na aplicação propriamente dita com tempo de atraso limitado entre as solicitações para encontrar o limite da capacidade de "páginas por segundo" de vários componentes funcionais. Obviamente, as páginas ou transações com o pior throughput de página precisam de mais ajustes.

Testes de simultaneidade

Nos níveis do sistema e da aplicação, a simultaneidade é limitada por sessões e conexões de soquete. As falhas de código e configurações incorretas do servidor podem também limitar a simultaneidade. Os testes de simultaneidade envolvem o aumento do número de usuários no sistema e a utilização de tempos de atraso de páginas realistas com um aumento de velocidade lento o suficiente para reunir dados úteis durante todo o teste em cada nível de carga. Assim como com os testes de throughput, é importante testar as páginas chave e transações de usuários na aplicação que está sendo testada.

A diferença entre os testes de throughput e de simultaneidade

A carga gerada por um teste de carga de 100 usuários virtuais com tempos de raciocínio de 1 segundo não é equivalente à carga gerada por um teste de carga de 1.000 usuários virtuais com tempos de raciocínio de 10 segundos. Conforme mostrado na Figura 3, os dois testes são idênticos em termos de throughput; entretanto, em termos de simultaneidade, eles são muito diferentes.

Embora a maioria dos gargalos seja encontrada nos testes de throughput, para testar uma aplicação de forma adequada, você deve testar o throughput e a simultaneidade.

Cenário	Throughput	Simultaneidade	Ponto de gargalo
100 usuários 1 segundo/página	Páginas/segundo = (100VU x 1 página/VU) ÷ 1 segundo = 100	100 Conexões	50 Páginas/ segundo
1.000 usuários 10 segundos/ página	Páginas/segundo = (1.000VU x 1 página/VU) ÷ 10 segundos = 100	1.000 Conexões	25 Páginas/ segundo

Figura 3. Teste de carga de 100 usuários virtuais versus teste de carga de 1.000 usuários virtuais.

No primeiro cenário, o teste de throughput, a aplicação apresentou um gargalo a uma taxa de 50 páginas por segundo. Entretanto, no segundo cenário, um teste de simultaneidade das mesmas transações, a aplicação apresentou um gargalo a uma taxa de 25 páginas por segundo. As únicas diferenças entre esses dois testes foram o número de usuários no sistema e o período de tempo que esses usuários permaneceram nas páginas. No teste de throughput com menos usuários e menores tempos de atraso de visualização de páginas, a aplicação apresentou mais capacidade de throughput; o segundo teste mostra que a aplicação estava limitada por sua simultaneidade. Se os testadores tivessem verificado somente o throughput, o problema de simultaneidade não teria sido detectado até que a aplicação estivesse em produção.

As figuras 4 e 5 na página seguinte mostram os resultados de cada teste e destacam a importância de testar tanto o throughput como a simultaneidade.

As únicas diferenças entre esses dois testes foram o número de usuários no sistema e o período de tempo que esses usuários permaneceram nas páginas.

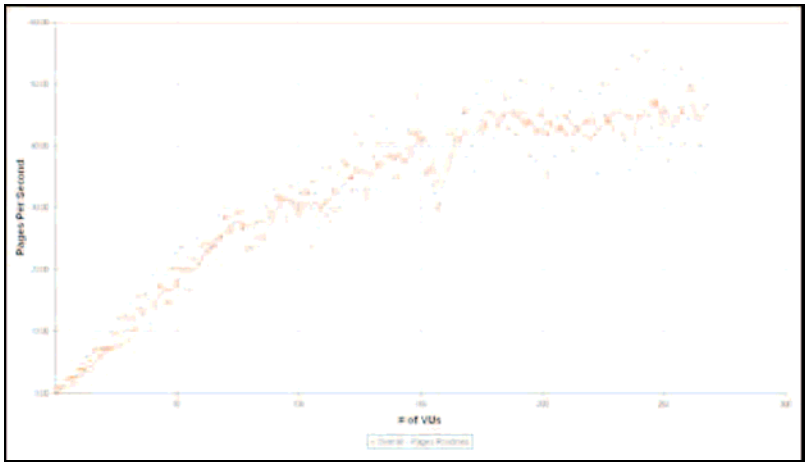

Figura 4. Testes de throughput de 100 usuários com visualizações de página de 1 segundo mostram um gargalo a uma taxa de 50 páginas por segundo.

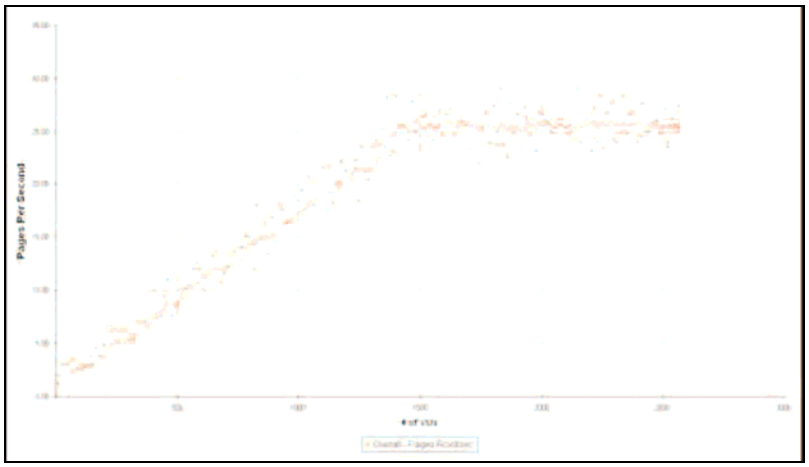

Figura 5. Testes de simultaneidade de 1.000 usuários com visualizações de página de 10 segundos mostram um gargalo a uma taxa de 25 páginas por segundo.

A ABORDAGEM RBI DE PROCESSAMENTO DE TESTES

Tradicionalmente, os testadores de desempenho se concentravam no número de usuários simultâneos, considerando-o a principal métrica para a escalabilidade da aplicação. Entretanto, se a maioria dos problemas no nível do sistema e da aplicação é encontrada em testes de throughput, uma nova abordagem é necessária.

Esses três princípios formam a base da metodologia RBI.

- Todas as aplicações Web têm gargalos.
- Esses gargalos somente podem ser detectados um por vez.
- É necessário se concentrar nos gargalos com maior probabilidade de ocorrência.

Embora reconheça a importância dos testes de simultaneidade, a metodologia RBI se concentra primeiro nos testes de throughput para eliminar pela raiz os gargalos mais comuns e, em seguida, realiza os testes de simultaneidade sob condições de carga que reflitam o número real de usuários esperados na aplicação. O processamento de testes de RBI também começa com testes simples e então aumenta a complexidade de forma que, quando um problema aparecer, todas as outras possíveis causas tenham sido descartadas. Concentrar-se nos testes de throughput e, em seguida, nos testes de simultaneidade usando uma abordagem estruturada para o processo de testes garante que os gargalos sejam isolados rapidamente, o que aumenta a eficiência e reduz os custos.

Benefícios do processamento de testes de RBI

A metodologia RBI permite a realização rápida, (e ainda assim completa) de testes, que descobre sistematicamente todos os problemas do sistema e da aplicação, sejam eles simples ou complexos.

Tempo de testes reduzido

Quanto tempo você pode economizar se concentrando inicialmente em testes de throughput? Considere um exemplo de um sistema com expectativa de lidar com 5.000 usuários simultâneos, com os usuários gastando uma média de 45 segundos em cada página. Se a aplicação tem um gargalo que irá limitar sua escalabilidade para 25 páginas por segundo, um teste de simultaneidade irá encontrar esse gargalo em aproximadamente 1.125 usuários (25 páginas por segundo a 45 segundos por página).

Ao descobrir os gargalos através de uma abordagem em camadas, você pode rapidamente identificar problemas e isolar problemas em componentes sobre os quais você não possui muito conhecimento.

No intuito de não influenciar os dados, o aumento de intensidade de um teste de simultaneidade típico deve ocorrer lentamente. Por exemplo, você pode pensar em aumentar um usuário a cada cinco segundos. Neste exemplo, o gargalo teria ocorrido em 5.625 segundos ou 94 minutos de teste (1.125 usuários, com um usuário sendo adicionado a cada 5 segundos). Entretanto, para validar o gargalo, o teste deveria continuar além deste ponto para comprovar que o throughput não estava aumentando conforme os usuários eram adicionados. Um teste de throughput teria encontrado esse problema em menos de 60 segundos.

Eliminar a complexidade inicial dos testes

Com frequência, os testes de desempenho começam com cenários extremamente complexos operando muitos componentes ao mesmo tempo, tornando mais fácil para que os gargalos não apareçam. A metodologia RBI começa com testes no nível do sistema que podem ser realizados antes mesmo que a aplicação tenha sido implantada.

Melhorar a eficiência do controle de qualidade

A metodologia RBI realiza os testcases mais simples primeiro e, em seguida, avança para aqueles mais complexos. Se os testcases mais simples funcionam e o próximo nível de complexidade falha, o gargalo está relacionado à complexidade que acabou de ser adicionada. Ao descobrir os gargalos através de uma abordagem em camadas, você pode rapidamente identificar problemas e isolar problemas em componentes sobre os quais você não possui muito conhecimento.

Eficiência aprimorada de testes com agregação de conhecimento

A natureza modular e iterativa da metodologia significa que quando um gargalo aparece, todos os componentes testados anteriormente já foram descartados. Por exemplo, se acessar a página inicial não mostra gargalos, mas acessar a página inicial e executar uma pesquisa mostra um desempenho muito ruim, a causa do gargalo está relacionada à funcionalidade de pesquisa.

Processamento de testes de RBI para os gargalos mais comuns do sistema

Todos os testes de desempenho devem começar com uma avaliação da infraestrutura de rede básica que suporta a aplicação. Se esse sistema básico não consegue suportar a carga de usuários antecipada em um sistema, mesmo um código de aplicação infinitamente escalável apresentará gargalos. Os testes no nível de sistema básico devem ser executados para validar a largura de banda, a taxa de acessos e as conexões. Além disso, páginas simples de teste da aplicação devem ser utilizadas (páginas "hello world" simples, por exemplo).

A aplicação

Após validar a infraestrutura do sistema e descobrir que ela atende às necessidades mais básicas dos usuários, volte-se para a aplicação propriamente dita. Mais uma vez, comece com os testcases mais simples possíveis.

Se os testes prosseguiram até agora sem descobrir problemas no nível do sistema (ou se esses problemas já foram resolvidos), todos os problemas restantes serão causados pela própria aplicação. Por exemplo, se uma página de teste da aplicação conseguiu uma taxa de 100 páginas por segundo e a página inicial apresenta gargalos a uma taxa de apenas 10 páginas por segundo, o problema está relacionado à sobrecarga necessária para exibir a página inicial.

A diferença entre o desempenho de uma página de teste e o desempenho de uma página real da aplicação de produção determina o ganho potencial de melhoria no desempenho obtido através do ajuste.

Neste ponto, a página de teste da aplicação fornece duas informações valiosas. Primeiro, como sabemos que o sistema em si não é o gargalo, o culpado só pode ser o código na página inicial. Segundo, podemos ver o quanto seria possível melhorar o desempenho através do ajuste da página inicial. A diferença entre os desempenhos da página de teste da aplicação (100 páginas por segundo) e da página inicial (10 páginas por segundo) determina a máxima melhoria de desempenho que o ajuste poderia fornecer. Da mesma forma, as transações com múltiplas páginas podem ser avaliadas ao separar o desempenho de cada página na transação individualmente e avaliar o quanto cada uma contribui para o desempenho da transação como um todo.

Como qualquer página da aplicação no mundo real provavelmente exige mais capacidade de processamento que uma página de teste "hello world", é razoável esperar uma pequena queda no desempenho. Entretanto, quanto maior essa queda, maior a necessidade (e o ganho potencial) de ajuste. É também importante observar que se a queda de desempenho entre a página de teste e uma página real da aplicação não for significativa e o desempenho ainda for insuficiente para atender às necessidades da base de usuários, será necessário adicionar mais recursos de hardware.

Até agora, os tempos de resposta das páginas não foram mencionados. Embora os tempos de resposta sejam uma das principais métricas do desempenho geral, eles serão os mesmos para 1.000 ou 100.000 usuários a menos que um gargalo seja encontrado. Portanto, nesta metodologia, os tempos de resposta são somente úteis como um indicador de que um gargalo foi obtido (caso eles comecem a apresentar picos) ou como um critério de falha (caso eles comecem a exceder um limite predefinido), com páginas com baixo desempenho (aquelas com erros ou altos tempos de resposta) em sua maioria necessitando de otimização no código.

Processamento de testes de RBI para os gargalos da aplicação

Assim como com o teste no nível do sistema, a metodologia RBI começa com os testcases mais simples possíveis e, em seguida, avança em complexidade. Em uma aplicação de e-commerce típica, você testaria a página principal primeiro e, em seguida, adicionaria páginas e funções de negócios até que transações completas do mundo real estivessem sendo testadas, primeiro individualmente e depois em

padrões de utilização com cenários complexos. Conforme as etapas são adicionadas, quaisquer degradações nos tempos de resposta ou throughput das páginas terá sido causada pela etapa que acabou de ser adicionada, tornando mais fácil isolar o código que precisa ser investigado.

Uma vez que cada uma das funções de negócios e transações tenha sido testada e otimizada (conforme necessário), as transações podem ser combinadas em testes de simultaneidade com cenários completos. Esses testes de simultaneidade devem se concentrar em dois componentes principais. Primeiro, o teste de simultaneidade deve refletir com precisão o que os usuários reais fazem no site: navegar, pesquisar, registrar, efetuar login e comprar. Segundo, as etapas nessas transações devem ser realizadas no mesmo ritmo que os visitantes do mundo real com "tempos de raciocínio" adequados entre cada etapa. Esses dados podem ser reunidos através de uma ferramenta de log da Web que mostra a duração da sessão, as páginas visualizadas por sessão (para determinar o ritmo do usuário) e a porcentagem de acesso às páginas (para determinar as funções de negócios reais usadas).

Os cenários adequados de testes de simultaneidade devem refletir com precisão o que os usuários reais fazem no site e replicar essas ações no mesmo ritmo dos usuários.

Desde que o teste tenha sido projetado com base em dados reais (ou hipóteses bem informadas, no caso de uma aplicação ainda não implantada), ele deve ser executado de forma a reunir informações importantes sobre diversos níveis de carga de usuários. Caso seja esperado que o site lide com 1.000 usuários simultâneos, é importante *não* iniciar todos esses usuários ao mesmo tempo. Em vez disso, aumente a intensidade do seu teste lentamente, adicionando um ou mais usuários em intervalos de tempo predefinidos, até atingir 1.000 usuários. Isso permitirá a você determinar o desempenho geral em cada nível de carga de usuário e também tornará mais fácil identificar problemas de desempenho quando eles começarem a ocorrer.

CONCLUSÃO

A metodologia RBI para o processamento de testes de carga melhora a eficiência do teste ao se concentrar primeiro onde os gargalos normalmente ocorrem, ou seja, no throughput. Uma vez que o throughput tenha sido testado por completo, é possível testar a simultaneidade do sistema e da aplicação para avaliar o desempenho sob cargas de usuário realistas. Ao seguir uma abordagem estruturada (do teste do sistema ao teste da aplicação) e adicionar complexidade lenta e sistematicamente aos testcases, você poderá rapidamente isolar os gargalos e sua causa principal associada.

Embora a intenção principal deste artigo seja descrever a metodologia, é importante ressaltar que boa parte desse processo pode e deve ser automatizada através de uma ferramenta de testes automatizada. O Oracle Application Testing Suite é o componente principal do Oracle Enterprise Manager no que diz respeito a testes funcionais e de carga para aplicações de arquitetura Web e orientada a serviços.

ENTRE EM CONTATO CONOSCO

Para obter mais informações sobre o Oracle Application Testing Suite e o Oracle Enterprise Manager acesse oracle.com/global/br/enterprise_manager ou ligue para 0800-891-4433 para falar com um representante da Oracle. Além disso, visite a Oracle Technology Network em oracle.com/technology/products/oem/

Identificação rápida de gargalos — Uma forma mais eficiente de realizar testes de carga
Junho de 2009

Oracle do Brasil Sistemas Ltda.
Sede no Brasil
Av. Alfredo Egydio de Souza Aranha, 100
São Paulo, SP
Brasil

CNPJ: 59.456.277/0001-76
Fone: 0-800-891-44-33
oracle.com