Computação Distribuída

Modelos de Sistema

António Rui Borges

Modelos

Os *sistemas distribuídos* destinam-se a ser usados no mundo real, o que significa que devem ser projectados para funcionar correctamente mesmo quando sujeitos a uma gama variada de circunstâncias e face a diferentes dificuldades e ameaças previsíveis.

Recorre-se a *modelos de descrição* para descrever as propriedades comuns e os pressupostos de desenho característicos de uma classe particular de sistemas.

O objectivo do modelo é fornecer uma descrição simplificada e abstracta, mas consistente, de um aspecto relevante do sistema sob consideração.

Tipos de modelos

- *modelos arquitecturais* – definem o modo como os diferentes componentes do sistema são mapeados nos nós da máquina paralela subjacente e como interagem entre si
  - *em termos de mapeamento*: procuram estabelecer padrões eficientes de distribuição dos dados e da carga de processamento
  - *em termos de interacção*: descrevem o papel funcional atribuído a cada componente e os padrões de comunicação existentes entre eles
- *modelos fundamentais* – centram-se em características sistémicas que afectam a confiabilidade
  - *tipo de interacção*: abordam questões como a eficiência da comunicação e a dificuldade em se estabelecer limites temporais
  - *tratamento de falhas*: especificam o tipo de falhas a que estão sujeitos os processos e os canais de comunicação intervenientes
  - *problemática de segurança*: discutem as ameaças possíveis que se colocam ao sistema distribuído e que afectam o seu desempenho

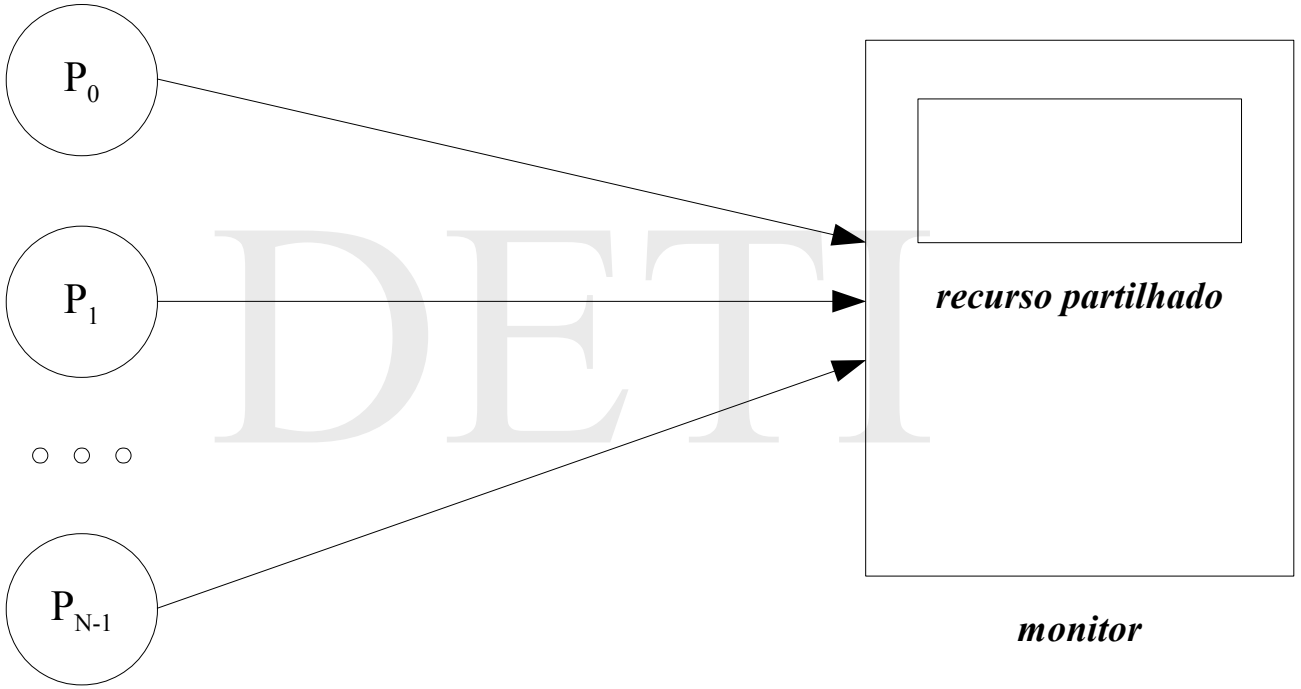
Modelos arquitecturais

Num *sistema distribuído*, os processos intervenientes cooperam entre si na realização de uma tarefa comum. A divisão de responsabilidades entre os diversos componentes activos e o seu mapeamento nos vários nós da máquina paralela subjacente constitui talvez o elemento mais distintivo do projecto.

Uma abordagem possível consiste em considerar em primeiro lugar uma solução concorrente do problema (válida num sistema computacional monoprocessador) e efectuar de seguida um conjunto de transformações conducentes à migração da solução para uma rede de computadores.

Na construção da solução concorrente, qualquer dos paradigmas de comunicação pode ser usado, *variáveis partilhadas* ou *passagem de mensagens*, já que eles são no fundo equivalentes. O primeiro, porém, conduz a implementações mais intuitivas.

Solução concorrente – 1



Solução concorrente – 2

- o recurso partilhado encontra-se no espaço de endereçamento de todos os processos intervenientes
- um *monitor* protege o acesso ao recurso para garantir que se efectua sempre em regime de exclusão mútua
- a sincronização entre processos realiza-se no interior do monitor através do recurso a *variáveis de condição*
- o modelo de interacção é tipicamente *reactivo*: cada processo executa até terminar ou bloquear
- a interacção, propriamente dita, baseia-se na execução de operações sobre o monitor
- alternativamente, se a linguagem de programação não providenciar uma semântica de concorrência, o recurso partilhado pode ser protegido por um semáforo de acesso (ou outro qualquer dispositivo semelhante) e a sincronização ser efectuada também por semáforos, agora colocados fora da região crítica para evitar *deadlock*.

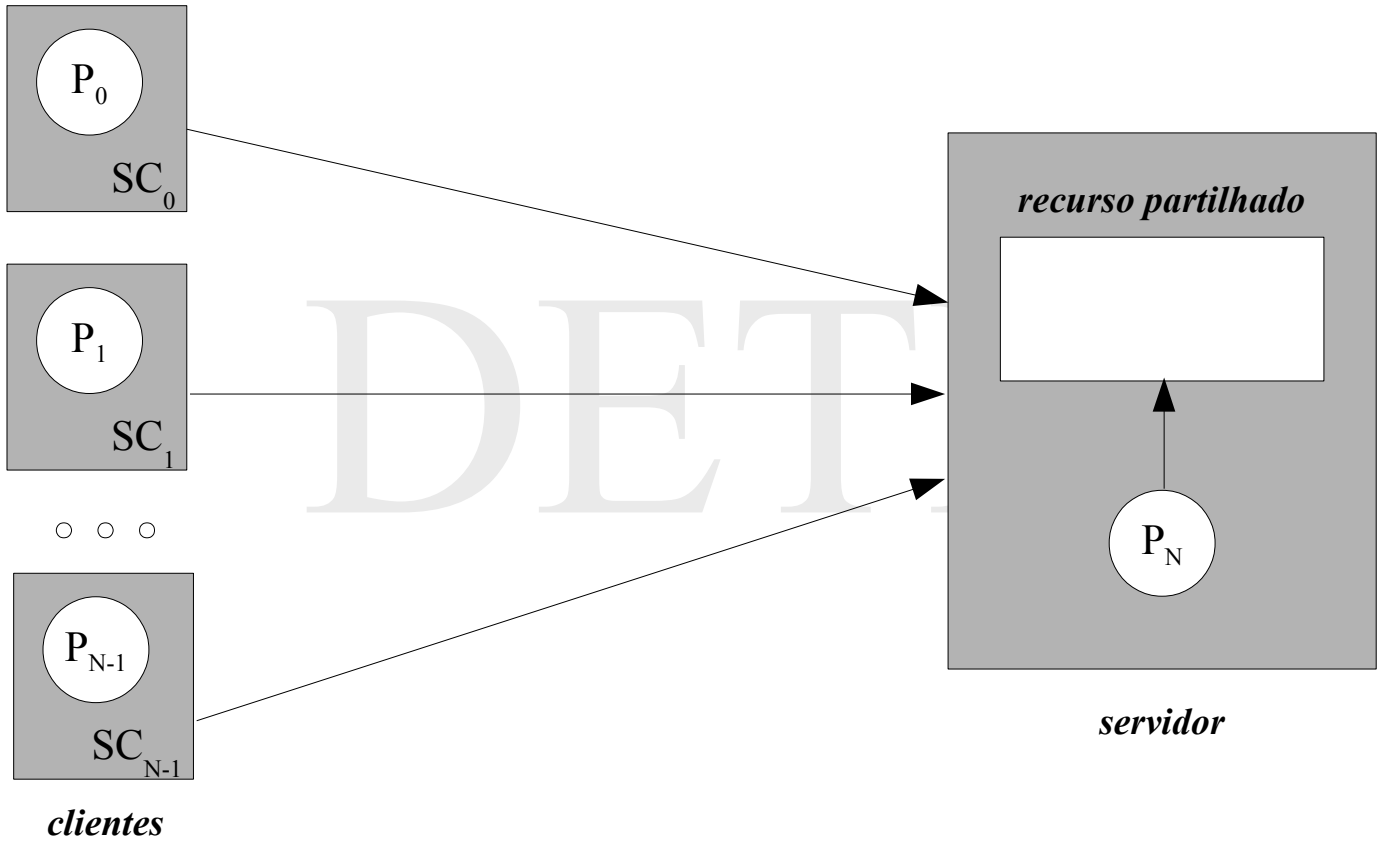
Modelo cliente - servidor – 1

Ao efectuar-se a migração dos processos e do recurso partilhado para sistemas computacionais distintos, o espaço de endereçamento de cada um deles torna-se disjunto e a comunicação passará necessariamente a ter lugar, de um modo explícito ou implícito, por troca de mensagens sobre o canal de comunicação comum.

Um aspecto notável a considerar é que, sendo o recurso partilhado uma entidade passiva, a gestão do seu acesso vai exigir o aparecimento de um novo processo, encarregado não só das comunicações com os outros processos, mas também da execução local das operações sobre o recurso partilhado.

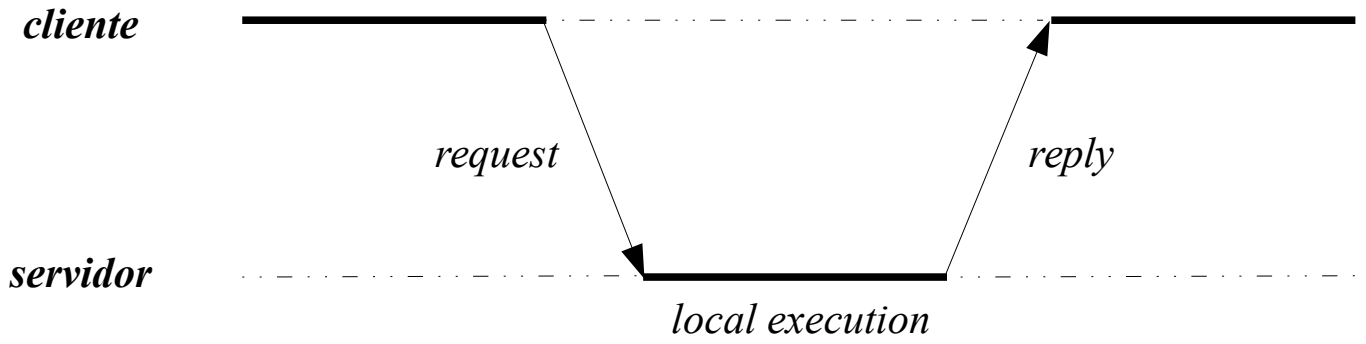
Daqui resulta um modelo operacional em que a manipulação do recurso pode ser entendida como a *prestação de um serviço*. O processo que gere localmente o recurso partilhado é visto como o prestador do serviço, designado genericamente de *servidor*. Os processos restantes que pretendem aceder ao recurso são vistos como as entidades que solicitam o serviço, designados genericamente de *clientes*.

Modelo cliente - servidor – 2



Modelo cliente - servidor – 3

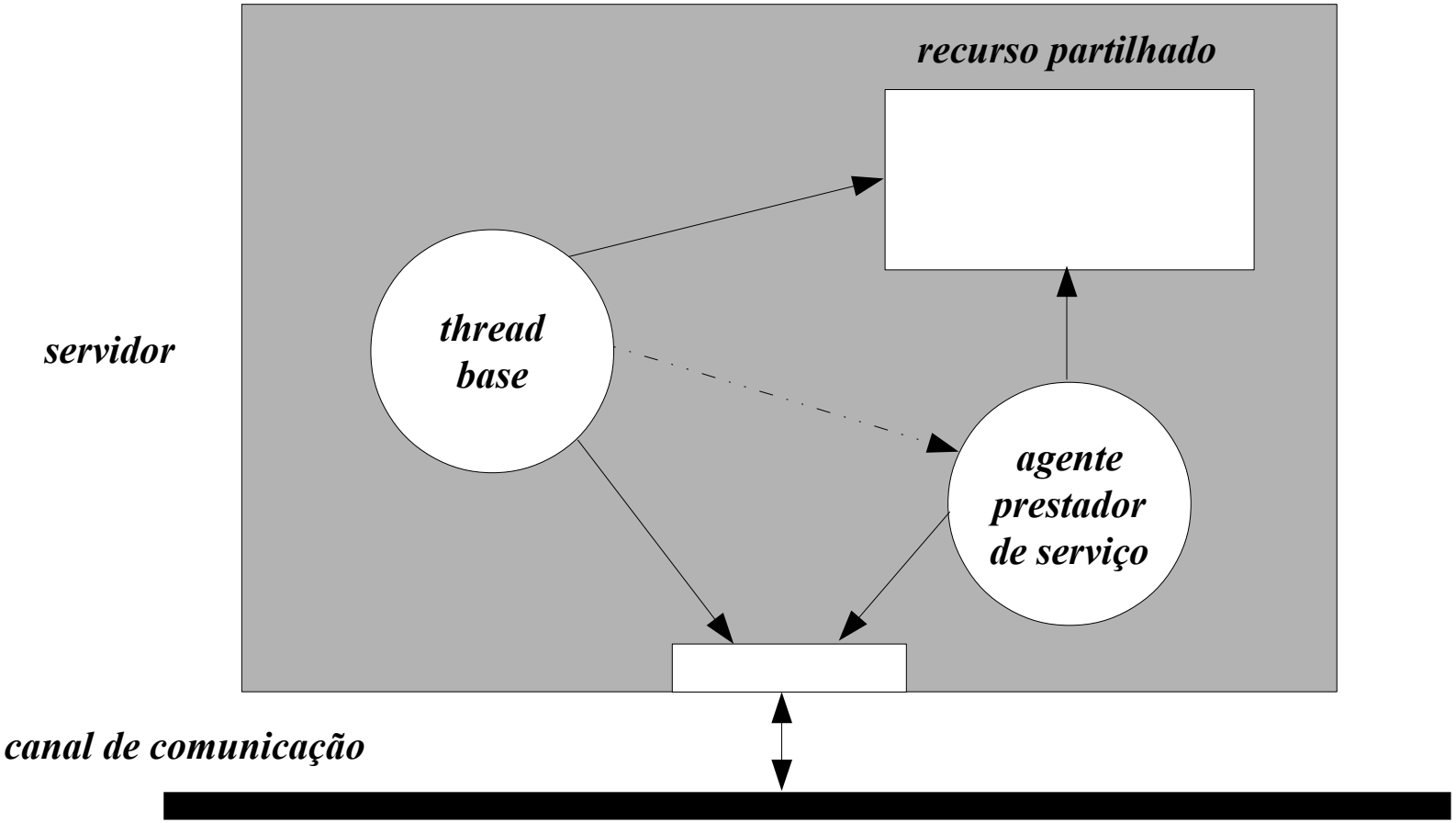
- a realização de operações sobre o recurso partilhado decompõe-se em
  - *request*: pedido por parte do processo *cliente* ao processo *servidor* para que este efectue a operação em seu nome
  - *local execution*: execução por parte do processo *servidor* da operação solicitada sobre o recurso partilhado
  - *reply*: comunicação ao processo cliente do resultado da operação



Modelo cliente - servidor – 4

- o processo servidor realiza tipicamente duas funções
  - *gere as comunicações*: aguarda os pedidos de serviço por parte dos processos *cliente*
  - *actua com um agente prestador de serviço*: efectua as operações sobre o recurso partilhado como procurador dos processos *cliente*
- o modelo de comunicação é assimétrico
  - *servidor* → **público** / *clientes* → **privados**: a operacionalização do serviço exige que o serviço disponibilizado seja conhecido de todos os interessados na sua utilização, enquanto que os fruidores do serviço não precisam de ser previamente conhecidos pelo seu fornecedor
  - *servidor* → **eterno** / *clientes* → **mortais**: a disponibilização do serviço exige que ele esteja permanentemente operacional, enquanto que os seus fruidores só se manifestam em intervalos bem definidos do tempo
  - *controlo centralizado*: todas as operações de prestação do serviço estão concentradas numa única entidade

Arquitectura básica do servidor – 1

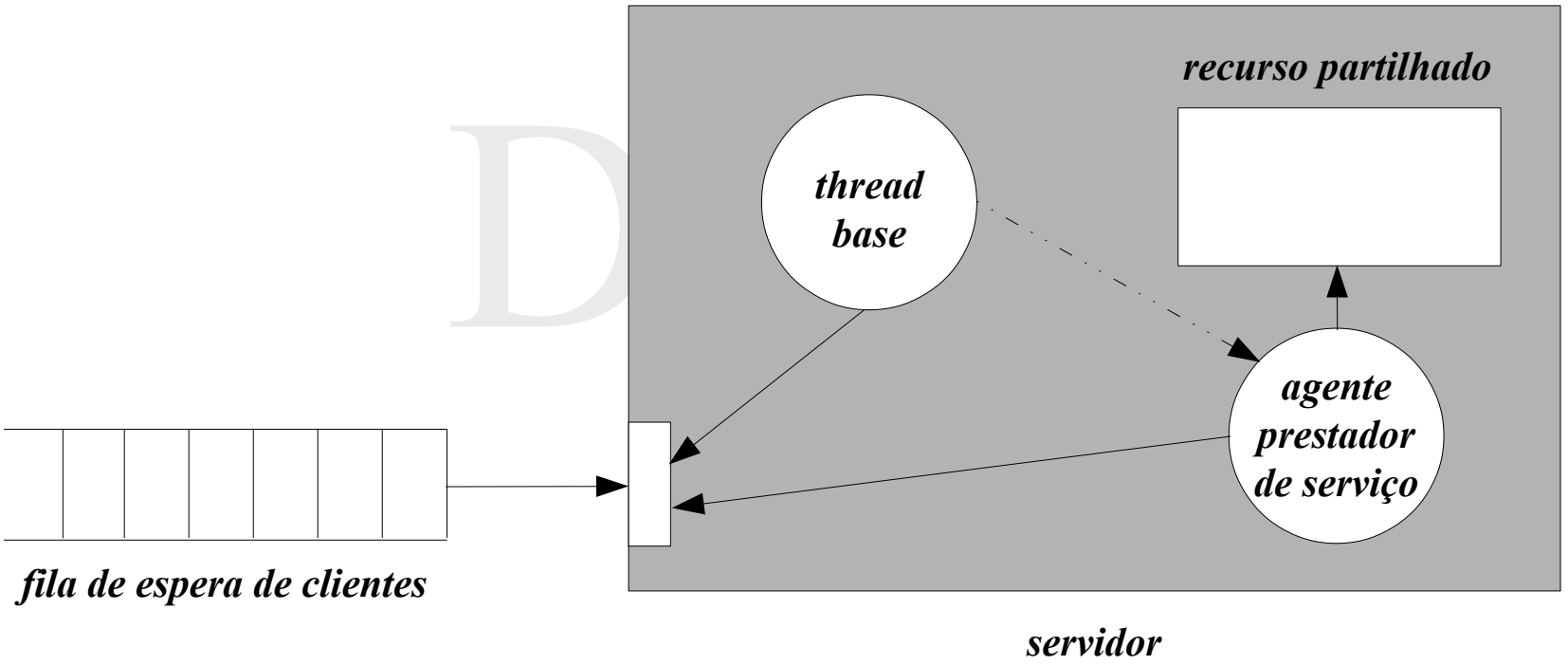


Arquitectura básica do servidor – 2

- papel do *thread base*
  - instanciar o recurso partilhado
  - instanciar o canal de comunicação e mapeá-lo num endereço público
  - colocar-se à escuta no canal de comunicação
  - quando ocorrer uma ligação de um processo *cliente*, instanciar um *thread* de tipo *agente prestador de serviço*
- papel do *thread agente prestador de serviço*
  - recolher do processo *cliente* o pedido de operação a realizar sobre o recurso partilhado (*request*)
  - executar a operação solicitada (*local execution*)
  - comunicar ao processo cliente o resultado da operação (*reply*)

Variantes do modelo cliente - servidor – 1

Serialização de pedidos



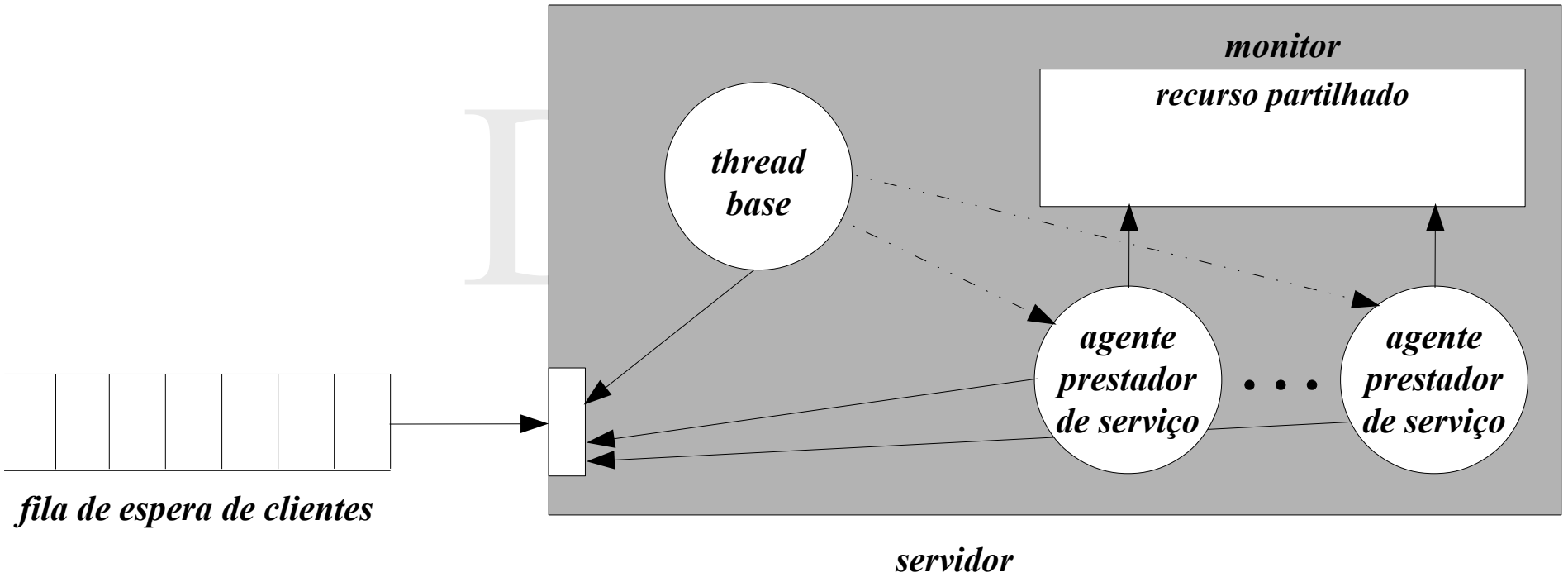
Variantes do modelo cliente - servidor – 2

Variante de tipo 1 (***serialização de pedidos***)

- só um processo cliente é atendido de cada vez: isto significa que o *thread* base, ao receber um pedido de ligação, instancia um *agente prestador de serviço* e aguarda a sua terminação antes de se colocar novamente à escuta
- o recurso partilhado não necessita de protecção especial para garantir exclusão mútua no acesso, porque só um *agente prestador de serviço* está activo de cada vez
- trata-se de um modelo muito simples, mas pouco eficiente
  - o tempo de atendimento não é minimizado, porque sem concorrência não se tira partido dos tempos mortos na interacção
  - potencia a ocorrência de *busy waiting* na tentativa de sincronização de diferentes processos cliente sobre o recurso partilhado

Variantes do modelo cliente - servidor – 3

Replicação do servidor



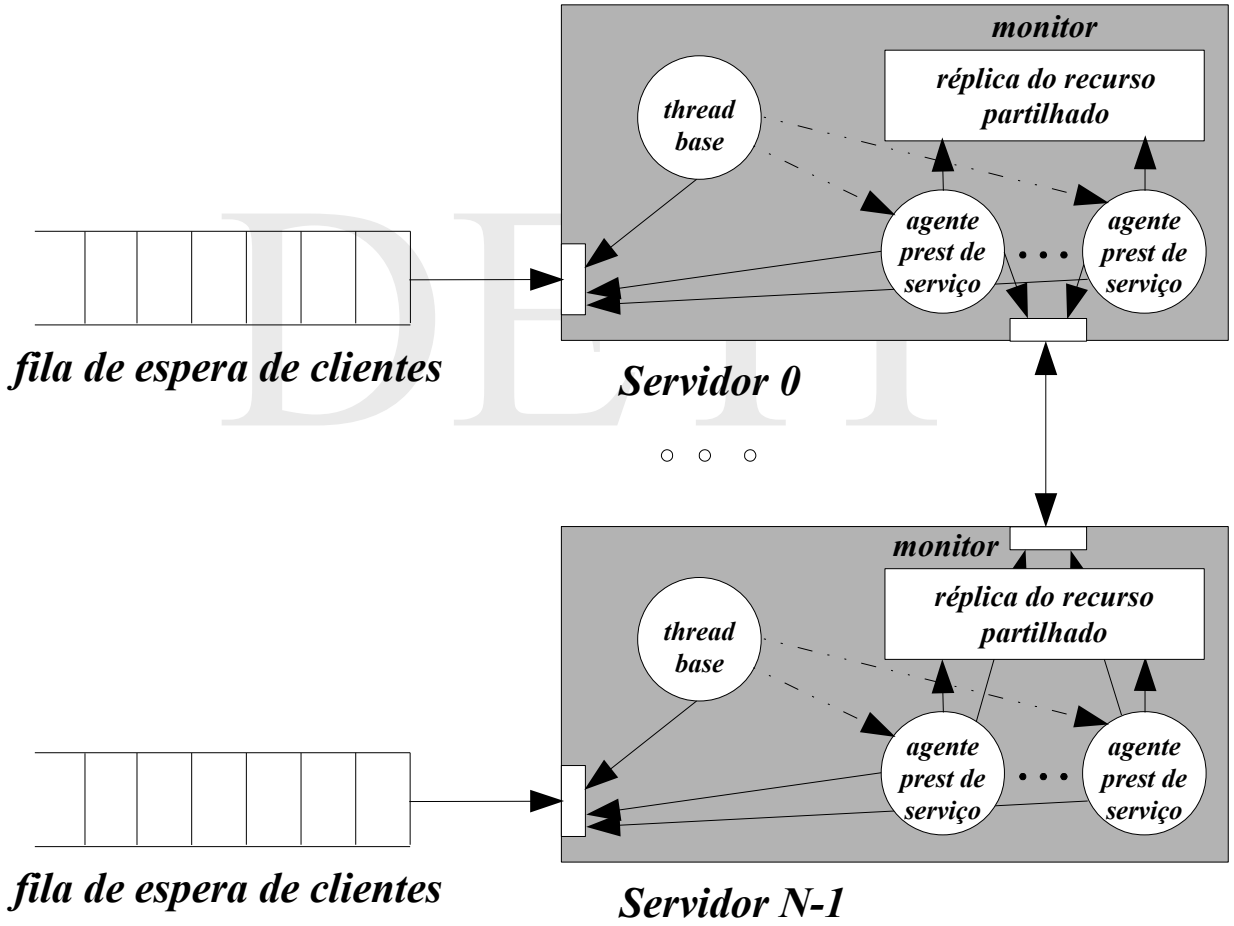
Variantes do modelo cliente - servidor – 4

Variante de tipo 2 (***replicação de servidor***)

- os processos cliente são atendidos concorrentemente: isto significa que o *thread* base, ao receber um pedido de ligação, instancia um *agente prestador de serviço* e volta a colocar-se novamente à escuta
- o recurso partilhado é transformado num monitor para garantir exclusão mútua no acesso, já que agora múltiplos *threads* de tipo *agente prestador de serviço* podem estar activos ao mesmo tempo
- trata-se do modelo tradicional que procura rentabilizar os recursos do sistema computacional onde está instalado o servidor
  - o tempo de atendimento é minimizado, porque se procura através da concorrência tirar partido dos tempos mortos na interacção
  - possibilita a sincronização de diferentes processos cliente sobre o recurso partilhado

Variantes do modelo cliente - servidor – 5

Cópia de recursos



Variantes do modelo cliente - servidor – 6

Variante 3 (***cópia de recursos***)

- o serviço é disponibilizado em simultâneo em vários sistemas computacionais, cada um executando uma variante de tipo 2 do servidor
- o recurso partilhado é, por isso, replicado em cada um dos servidores, originando cópias múltiplas
- trata-se de um modelo sofisticado que procura majorar o período de disponibilização do serviço e minorar o tempo de atendimento, mesmo em situações de picos de carga (potenciando portanto a escalabilidade)
  - o serviço mantém-se operacional face a falhas em servidores particulares
  - as solicitações dos clientes são distribuídas pelos servidores presentes usando um mecanismo de associação rotativa, por parte do serviço de DNS local, dos endereços IP dos servidores individuais ao URL identificador do serviço
  - quando há alteração dos dados presentes no recurso partilhado, surge naturalmente o problema de manter a consistência entre as diferentes cópias presentes para garantir a consistência de informação

Leituras sugeridas

- *Distributed Systems: Concepts and Design, 4th Edition, Coulouris*, Dollimore, Kindberg, Addison-Wesley
  - Capítulo 2: *System models*
- *Distributed Systems: Principles and Paradigms, 2nd Edition,* Tanenbaum, van Steen, Pearson Education Inc.
  - Capítulo 2: *Architectures*
    - Secções 2.1 a 2.3