

# Un modello di rendezvous

Università degli Studi di Padova

## Un modello di rendezvous

# SCD

Anno accademico 2003/4  
 Corso di Sistemi Concorrenti e Distribuiti

Tullio Vardanega, [tullio.vardanega@math.unipd.it](mailto:tullio.vardanega@math.unipd.it)

Corso di Laurea Specialistica in Informatica, Università di Padova 1

Università degli Studi di Padova

## Un modello di rendezvous

### Modello base - 1

- **Interazione di tipo cliente-servente**
  - Il processo servente dichiara un certo numero di servizi che esso è disposto a fornire a processi cliente
  - La specifica del processo servente dichiara i corrispondenti punti d'accesso (*entry*), ciascuno dei quali può prevedere un diverso protocollo di scambio parametri
  - Ciascun processo cliente emette una richiesta d'accesso (*entry call*) nella quale nomina servente e servizio
  - Il servente fornisce uno dei servizi richiesti esprimendone esplicitamente la propria accettazione
  - La comunicazione tra servente e cliente è sincrona e **non** richiede necessariamente il passaggio di dati

Corso di Laurea Specialistica in Informatica, Università di Padova 2

Università degli Studi di Padova

## Un modello di rendezvous

### Modello base - 2

```

task type Operator is
entry Query (A_Person : in Name;
             An_Address : in Address;
             A_Number : out Number);
end Operator;
Ann : Operator;

task type User;
task body User is
My_Number : Number;
begin
Ann.Query(
    " ", " ",
    My_Number);
end User;

task body Operator is
begin
loop
accept Query (A_Person : in Name;
              An_Address : in Address;
              A_Number : out Number) do
end loop;
end Operator;
    
```

Specifica di punto di accesso

Invocazione

Realizzazione di accettazione

Corso di Laurea Specialistica in Informatica, Università di Padova 3

Università degli Studi di Padova

## Un modello di rendezvous

### Modello base - 3

- **Modello chiamato rendezvous per implicare che cliente e servente si incontrano presso un punto d'accesso allo stesso tempo**
- **Al momento dell'incontro i parametri di modo in passano dal cliente al servente**
- **Il servente esegue la realizzazione del servizio come una normale procedura e poi restituisce i parametri di modo out al cliente**
- **A quel punto la sincronizzazione si interrompe ed i processi riprendono la loro esecuzione concorrente**

Corso di Laurea Specialistica in Informatica, Università di Padova 4

Università degli Studi di Padova

## Un modello di rendezvous

### Modello base - 4

- **Nella forma base**
  - Il servente si sospende in attesa di una richiesta
    - Come previsto per il Server nel nostro modello di concorrenza (lezione C01)
  - Il cliente si sospende fino alla disponibilità del servizio
  - I clienti sospesi vengono posti in una coda associata al servizio (*entry queue*)
  - L'ordine di accordamento è normalmente FIFO, ma può essere configurato (p.es. su base prioritaria)

Corso di Laurea Specialistica in Informatica, Università di Padova 5

Università degli Studi di Padova

## Un modello di rendezvous

### Modello base - 5

Corso di Laurea Specialistica in Informatica, Università di Padova 6

# Un modello di rendezvous

Università degli Studi di Padova Un modello di rendezvous

**Esempio - 1**

□ **Il crivello di Eratostene rivisitato**

- Una diversa realizzazione dello stesso algoritmo descritto nella lezione precedente
- Invece di usare una risorsa protetta per contenere i possibili numeri primi da esaminare, ogni coppia di processi nella sequenza comunica tramite *rendezvous*
- L'effetto di sincronizzazione rende superflua la mutua esclusione, mentre l'accodamento FIFO sulla coda d'accesso preserva l'ordine dei numeri da esaminare

Corso di Laurea Specialistica in Informatica, Università di Padova 7

Università degli Studi di Padova Un modello di rendezvous

**Esempio - 2**

Corso di Laurea Specialistica in Informatica, Università di Padova 8

Università degli Studi di Padova Un modello di rendezvous

**Sincronizzazione tripartita - 1**

□ **Il modello *rendezvous* è sincrono, asimmetrico e bidirezionale**

□ **Le azioni svolte dal servente durante la sincronizzazione possono coinvolgere processi terzi**

- Ciò consente di realizzare forme complesse di sincronizzazione che però preservano separazione funzionale (desiderabile!) tra i processi coinvolti

Corso di Laurea Specialistica in Informatica, Università di Padova 9

Università degli Studi di Padova Un modello di rendezvous

**Sincronizzazione tripartita - 2**

□ **Il coinvolgimento di processi terzi può avvenire in 2 forme strutturalmente distinte, ma duali**

- **Annidando accettazioni**
  - Modellando in tal modo una macchina a stati, con alcuni stati raggiungibili solo a partire da un dato stato iniziale
- **Invocando accettazioni nella realizzazione di una accettazione**
  - Realizzando così la fornitura di un servizio composto, che racchiude il possibile contributo di più serventi che si siano ripartiti tra loro il lavoro

Corso di Laurea Specialistica in Informatica, Università di Padova 10

Università degli Studi di Padova Un modello di rendezvous

**Sincronizzazione tripartita - 3**

L'astrazione di un dispositivo D che produce valori quando richiesto può richiedere sincronizzazione tripartita

Corso di Laurea Specialistica in Informatica, Università di Padova 11

Università degli Studi di Padova Un modello di rendezvous

**Sincronizzazione tripartita - 4**

```

task User;
task Device;
task Controller is
entry Service (I : out Integer);
entry Start;
entry Finish (K : Integer);
end Controller;

task body User is
...
Controller.Service(Val);
end User;

task body Controller is
begin
loop
accept Service (I : out Integer) do
accept Finish (K : Integer) do
I := K;
end Completed;
end Service;
end loop;
end Controller;

task body Device is
Val : Integer;
procedure Read (I : out Integer) is ... ;
begin
loop
Controller.Start;
Read(Val);
Controller.Finish(Val);
end Device;
    
```

Corso di Laurea Specialistica in Informatica, Università di Padova 12

# Un modello di rendezvous

Università degli Studi di Padova Un modello di rendezvous

## Sincronizzazione tripartita - 5

```

    graph LR
      User((User)) -- Request_Part(.) --> CustomerService((Customer Service))
      CustomerService -- Enquiry(...) --> Warehouse((Warehouse))
  
```

Una fornitura di servizio che nasconde al cliente l'eventuale necessità di approvvigionamento presso componenti "nascosti" del "sistema servente"

Corso di Laurea Specialistica in Informatica, Università di Padova 13

Università degli Studi di Padova Un modello di rendezvous

## Sincronizzazione tripartita - 6

```

    task Warehouse is
    entry Enquiry (Item : Part_Number;
                  In_Stock : out Boolean);
    end Warehouse;

    task Customer_Service is
    entry Request_Part (Order : Part_Number;
                      Part : Spare_Part;
                      Order : Order_Number);
    end Customer_Service;

    task body Customer_Service is
    In_Stock : Boolean;
    begin
    loop
    accept Request_Part (Order : Part_Number;
                       Part : Spare_Part;
                       Order : Order_Number) do
    -----
    if In_Stock then
    Part := The_Part;
    Order := None;
    else
    Warehouse.Enquiry(Order, In_Stock);
    if In_Stock then
    -- go get found part from Warehouse
    Part := The_Part;
    Order := Next_Order_Number;
    end if;
    end if;
    end Request_Part;
    end loop;
    end Customer_Service;
  
```

Corso di Laurea Specialistica in Informatica, Università di Padova 14

Università degli Studi di Padova Un modello di rendezvous

## Punti d'accesso privati - 1

- ❑ Un processo servente non deve esporre necessariamente tutti i suoi punti d'accesso
- ❑ Alcuni possono essere ristretti per motivi di convenienza (incapsulazione) e/o di opportunità (astrazione)
- ❑ La dichiarazione dei punti d'accesso deve allora distinguere tra pubblici e privati

Corso di Laurea Specialistica in Informatica, Università di Padova 15

Università degli Studi di Padova Un modello di rendezvous

## Punti d'accesso privati - 2

La visibilità ai punti d'accesso privati è ristretta al solo ambito (scope) del processo Controller

```

    task User;
    task Controller is
    entry Service (I : out Integer);
    private
    entry Start;
    entry Finish (K : Integer);
    end Controller;

    task body Controller is
    task Device;
    task body Device is
    Val : Integer;
    procedure Read (I : out Integer) is ...;
    begin
    loop
    Controller.Start;
    Read(Val);
    Controller.Finish(Val);
    end Device;
    -- continues in sidebar

    begin -- Controller
    loop
    accept Service (I : out Integer) do
    accept Start;
    accept Finish (K : Integer) do
    I := K;
    end Completed;
    end Service;
    end loop;
    end Controller;
  
```

Corso di Laurea Specialistica in Informatica, Università di Padova 16

Università degli Studi di Padova Un modello di rendezvous

## Casi d'errore

- ❑ Un errore (eccezione) che occorra durante la sincronizzazione ne causa l'abbandono e si propaga presso entrambi i partecipanti
- ❑ Emettere una richiesta d'accesso verso un processo terminato è considerato un errore a tempo di esecuzione e solleva una eccezione presso il chiamante

Corso di Laurea Specialistica in Informatica, Università di Padova 17

Università degli Studi di Padova Un modello di rendezvous

## Stati d'esecuzione di processo

```

    graph TD
      InEsecuzione((In esecuzione)) --> AttivazioneFigli((Attivazione figli))
      InEsecuzione --> SospensioneTemporanea((Sospensione temporanea))
      InEsecuzione --> Sincronizzazione((Sincronizzazione))
      InEsecuzione --> Accodamento((Accodamento presso punto d'accesso  
(cliente)))
      InEsecuzione --> Attesa((Attesa di richiesta d'accesso  
(servente)))
      InEsecuzione --> TerminazioneDipendenti((Terminazione dipendenti))
      InAttivazione((In attivazione)) --> InEsecuzione
      Completato((Completato)) --> InEsecuzione
  
```

Corso di Laurea Specialistica in Informatica, Università di Padova 18