

Università degli Studi di Padova

Estensioni del modello rendezvous

Estensioni del modello rendezvous



Anno accademico 2003/4
Corso di Sistemi Concorrenti e Distribuiti
Tullio Vardanega, tullio.vardanega@math.unipd.it

Corso di Laurea Specialistica in Informatica, Università di Padova

1

Università degli Studi di Padova

Estensioni del modello rendezvous

Limiti del modello base

- Il processo servente può disporsi ad accettare messaggi da un solo punto di accesso alla volta
 - Lo stesso vale per il processo cliente, che può emettere un solo messaggio di richiesta per volta
- Una volta postosi in attesa sul canale, il servente attende indefinitamente fino all'eventuale arrivo di un messaggio
 - Lo stesso vale per il cliente, che, una volta emesso un messaggio di richiesta, resta indefinitamente in attesa fino alla sua accettazione

Corso di Laurea Specialistica in Informatica, Università di Padova

2

Università degli Studi di Padova

Estensioni del modello rendezvous

Requisiti di estensione - 1

- Lato servente (più critico)**
 - Poter attendere su più di un punto d'accesso alla volta
 - Limitare l'attesa ad un tempo limite (*time-out*), oltre il quale abbandonarla
 - Poter abbandonare immediatamente l'attesa su un punto d'accesso che non abbia messaggi in coda
 - Poter terminare quando nessun cliente fosse più in grado di emettere richieste

Corso di Laurea Specialistica in Informatica, Università di Padova

3

Università degli Studi di Padova

Estensioni del modello rendezvous

Requisiti di estensione - 2

- Lato cliente (meno critico)**
 - Non è strettamente necessario poter emettere più richieste simultaneamente da parte di uno stesso cliente
 - Un processo cliente funzionalmente coeso ha una logica interna sequenziale!
 - È desiderabile porre un limite temporale all'attesa di accettazione di una richiesta effettuata
 - È utile anche poter abbandonare l'attesa di accettazione qualora questa non fosse immediatamente disponibile

Corso di Laurea Specialistica in Informatica, Università di Padova

4

Università degli Studi di Padova

Estensioni del modello rendezvous

Estensioni dal lato servente - 1

- Attesa su più di un punto d'accesso (1/3)**
 - Il servente può fornire più servizi, ciascuno dei quali viene fornito attraverso messaggi scambiati su uno specifico canale tipato (*entry*)

```
task Server is
  entry S1 (...);
  entry S2 (...);
end Server;
```

```
task body Server is
  begin
  loop
    select
      accept S1 (...) do ... end S1;
    or
      accept S2 (...) do ... end S2;
    end select;
  end loop;
end Server;
```

Corso di Laurea Specialistica in Informatica, Università di Padova

5

Università degli Studi di Padova

Estensioni del modello rendezvous

Estensioni dal lato servente - 2

- Attesa su più di un punto d'accesso (2/3)**
 - In questo modo il servente si pone comunque in attesa (della prima prossima richiesta) qualora nessuna fosse disponibile al momento della valutazione
 - Solo il requisito 1 viene così soddisfatto
 - La valutazione avviene simultaneamente su tutti i canali compresi
 - Qualora più di un punto d'accesso avesse una richiesta in attesa, la scelta di uno tra essi è nondeterministica
 - Qualora più di una richiesta fosse in attesa presso lo stesso punto d'accesso, la scelta di una tra esse è FIFO
 - Politiche alternative (p.es. per urgenza) possono essere desiderabili

Corso di Laurea Specialistica in Informatica, Università di Padova

6

Università degli Studi di Padova

Estensioni del modello *rendezvous*

Estensioni dal lato server - 3

- **Attesa su più di un punto d'accesso (3/3)**
 - Conviene anche soddisfare i requisiti posti da Dijkstra (cf. lezione C02, pagina 29) sulle accettazioni di richieste
 - Occorre perciò prevedere "guardie" che esprimano condizioni logiche sull'opportunità (funzionale) di accettare particolari richieste

```
select
  Guard_1 => accept ...;
or
  Guard_2 => accept ...;
or
  --
or
  Guard_N => accept ...;
end select;
```

La guardia è una espressione Booleana, di tipo "when <condizione>", il cui verificarsi abilita la considerazione del canale
Le guardie entro un `select` sono valutate simultaneamente una sola volta all'inizio del comando

Corso di Laurea Specialistica in Informatica, Università di Padova

7

Università degli Studi di Padova

Estensioni del modello *rendezvous*

Estensioni dal lato server - 4

- **Limitare temporalmente l'attesa (1/2)**
 - **Requisito 2** → fissare un tempo limite non nullo entro il quale si è disposti ad attendere l'arrivo di richieste su uno dei canali considerati
 - Potendo naturalmente esprimere il tempo di attesa desiderato come relativo od assoluto, a seconda della necessità
 - **Requisito 3** → abbandonare immediatamente l'attesa in assenza di richieste all'istante di valutazione
 - Del tutto equivalente ad esprimere un tempo limite di attesa nullo

Corso di Laurea Specialistica in Informatica, Università di Padova

8

Università degli Studi di Padova

Estensioni del modello *rendezvous*

Estensioni dal lato server - 5

- **Limitare temporalmente l'attesa (2/2)**
 - Porre un limite temporale di attesa non nullo consente al processo server di preservare il suo comportamento di base restando però disponibile ad accettare richieste entro intervalli fissati
 - L'assenza di richieste su un (od alcun) canale per un dato lasso di tempo può essere interpretato dal server come indizio di una condizione di errore dei processi cliente, che in tal modo non si propaga al server

Corso di Laurea Specialistica in Informatica, Università di Padova

9

Università degli Studi di Padova

Estensioni del modello *rendezvous*

Esempio - 1

```
with Ada.Real_Time; use Ada.Real_Time;
task Sensor_Monitor is
  entry New_Period (Period : Time_Span);
end Sensor_Monitor;

task body Sensor_Monitor is
  Current_Period : Time_Span := Milliseconds(10_000);
  Next_Cycle : Time := Clock + Current_Period;
begin
  loop
    -- read sensor value and post it where required
    select
      accept New_Period(Period : Time_Span) do
        Current_Period := Period;
      end New_Period;
      Next_Cycle := Clock + Current_Period;
      delay until Next_Cycle; ①
    or
      delay until Next_Cycle;
      Next_Cycle := Next_Cycle + Current_Period;
    end select;
  end loop;
end Sensor_Monitor;
```

Comportamento di base periodico, con una lettura di sensore ogni 10 secondi, ma capace di modificare immediatamente l'ampiezza del periodo in caso di richiesta

L'effetto di immediatezza è ottenuto dalla presenza del blocco di comandi in ①

Corso di Laurea Specialistica in Informatica, Università di Padova

10

Università degli Studi di Padova

Estensioni del modello *rendezvous*

Esempio - 2

```
task type Watchdog ( Minimum_Distance : Duration ) is
  entry All_is_Well;
end Watchdog;

task body Watchdog is
begin
  loop
    select
      accept All_is_Well;
      -- client is alive and well
    or
      delay Minimum_Distance;
      -- client may have failed, raise alarm
    end select;
  end loop;
end Watchdog;
```

Nota che il requisito di Dijkstra applica anche all'alternativa di attesa temporale, che può pertanto ammettere una guardia

Corso di Laurea Specialistica in Informatica, Università di Padova

11

Università degli Studi di Padova

Estensioni del modello *rendezvous*

Estensioni dal lato server - 6

- **Attesa nulla**
 - Il processo server può voler prestare attenzione solo a quei canali che abbiano già richieste in attesa al momento del controllo ed altrimenti effettuare azioni alternative
 - Questa modalità rende possibile l'attesa attiva, che resta però indesiderabile!

```
select
  accept A;
or
  accept B;
else
  C;
end select;
```

L'effetto desiderato può essere ottenuto in 2 modi alternativi

Forma esplicita (preferibile)

Forma implicita per T=0.0

```
select
  accept A;
or
  accept B;
or
  delay T;
  C;
end select;
```

Corso di Laurea Specialistica in Informatica, Università di Padova

12

Estensioni del modello rendezvous

Università degli Studi di Padova

Estensioni del modello *rendezvous*

Estensioni dal lato servente - 7

- ❑ **Terminazione in mancanza di clienti (1/2)**
 - L'asimmetria del modello cliente-servente può far sì che il processo servente sopravviva al completamento dei suoi clienti
 - In questo caso, è desiderabile che anche il servente possa terminare
 - La terminazione del servente può essere trattata direttamente a programma
 - Trattandosi però di un requisito generale del modello esteso di *rendezvous* è desiderabile disporre di una soluzione generale
 - Basta consentire al processo servente di aggiungere un'alternativa *terminate* a quelle da considerare in parallelo nel comando *select*

Corso di Laurea Specialistica in Informatica, Università di Padova

13

Università degli Studi di Padova

Estensioni del modello *rendezvous*

Estensioni dal lato servente - 8

- ❑ **Terminazione in mancanza di clienti (2/2)**
 - Un processo servente sospeso su un comando *select* con alternativa *terminate* aperta viene considerato completo allorché
 - Il *master* da cui esso dipende ha completato la propria esecuzione
 - Ogni altro processo dipendente da quello stesso *master* è
 1. Già terminato, oppure
 2. A sua volta sospeso su un comando *select* con alternativa *terminate* aperta
 - La condizione 1. assicura che non vi possano essere nuove richieste di servizio in arrivo
 - La condizione 2. applica la regola transitivamente

Corso di Laurea Specialistica in Informatica, Università di Padova

14

Università degli Studi di Padova

Estensioni del modello *rendezvous*

Ultime volontà ☺

- ❑ Questa forma di terminazione va però arricchita con un meccanismo che consenta al processo terminante di effettuare azioni esplicite di finalizzazione (le ultime volontà)
- ❑ Alcuni tipi speciali esportano un metodo esplicito di finalizzazione, che viene invocato quando l'oggetto corrispondente deve essere rimosso
- ❑ La terminazione di un processo la cui regione dichiarativa contenga istanze di tali tipi comporta l'invocazione automatica dei corrispondenti metodi di finalizzazione

Corso di Laurea Specialistica in Informatica, Università di Padova

15

Università degli Studi di Padova

Estensioni del modello *rendezvous*

Esempio - 3

- ❑ **Il crivello di Eratostene bis-rivisitato**
 - Vogliamo aggiungere controllo di terminazione alle istanze dei processi Sieve
 - In caso di terminazione, vogliamo anche che il processo terminante ce ne fornisca notifica

Corso di Laurea Specialistica in Informatica, Università di Padova

16

Università degli Studi di Padova

Estensioni del modello *rendezvous*

Estensioni dal lato cliente

- ❑ **Per il processo cliente avevamo identificato solo 2 esigenze**
 - Porre un limite temporale non nullo all'attesa di servizio
 - Equivalente al requisito 2 del lato servente e soddisfatto nello stesso modo
 - Il limite temporale riguarda solo la durata massima dell'attesa dell'inizio della sincronizzazione
 - Nessun effetto sulla durata effettiva della sincronizzazione!
 - Lasciare immediatamente il canale qualora il processo servente non fosse immediatamente disponibile
 - Equivalente al requisito 3 del lato servente e soddisfatto nello stesso modo
 - Sta all'implementazione del modello gestire il caso in cui più clienti desiderino simultaneamente conoscere la disponibilità immediata di uno stesso servente

Corso di Laurea Specialistica in Informatica, Università di Padova

17

Università degli Studi di Padova

Estensioni del modello *rendezvous*

Usi del modello cliente-servente

- ❑ Un processo servente è un'entità reattiva capace di garantire mutua esclusione
 - Eseguito una sola alternativa *accept* alla volta
- ❑ L'esecuzione della sincronizzazione rappresenta la sezione critica
- ❑ La risorsa condivisa deve però essere visibile soltanto al processo servente

```
task body Buffer (...) is
  -- the shared resource
  begin
    loop
      select
        when _
          accept Put (...) do _ end Put;
        -- housekeeping
      or
        when _
          accept Get (...) do _ end Get;
        -- housekeeping
      or
        terminate;
      end select;
    end loop;
  end Buffer;
```

Corso di Laurea Specialistica in Informatica, Università di Padova

18

Estensioni del modello rendezvous

Università degli Studi di Padova

Estensioni del modello *rendezvous*

Abusi del modello cliente-servente

- ❑ **Programmazione incauta può dare origine a situazioni di stallo**
 - Il modello *rendezvous*, anche nella sua forma estesa, **non** è capace di impedirne strutturalmente il rischio

```
task T1 is
  entry A;
end T1;

task body T1 is
  begin
    T2.B;
  accept A;
end T1;

task T2 is
  entry B;
end T2;

task body T2 is
  begin
    T1.A;
  accept B;
end T2;
```

Corso di Laurea Specialistica in Informatica, Università di Padova

19

Università degli Studi di Padova

Estensioni del modello *rendezvous*

Una buona prassi

- ❑ **I processi dovrebbero essere usati soltanto per realizzare entità attive oppure *server***
- ❑ **Le entità attive non dovrebbero possedere punti di accesso, ma solo richiederne l'uso**
- ❑ **I *server* dovrebbero accettare richieste di accesso, ma non farne alcuna**

Corso di Laurea Specialistica in Informatica, Università di Padova

20

Università degli Studi di Padova

Estensioni del modello *rendezvous*

Stati d'esecuzione di processo

Corso di Laurea Specialistica in Informatica, Università di Padova

21