


Università degli Studi di Padova

## Un modello concreto



Annno accademico 2004/5  
 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 1/16

Università degli Studi di Padova

## Un modello concorrente concreto

### Forma sintattica - 1

- **Tipo (classe) task come modello di processo**
  - Deve essere dichiarato per poter essere istanziato
  - La dichiarazione consiste di specifica (*specification*) ed realizzazione (*body*)
  - La specifica contiene
    - Nome del tipo
    - Parte (opzionale) discriminante  
tramite la quale l'istanziazione può passare parametri (ma solo di tipi di dimensione determinabile) all'oggetto, esattamente come accade con un metodo costruttore
    - Parte visibile  
che definisce ciò che di quel processo occorre che gli altri sappiano, per esempio i punti di accesso (*entry*); include il discriminante
    - Parte privata  
che definisce aspetti di implementazione, che non riguardano il resto del sistema

Corso di Laurea Specialistica in Informatica, Università di Padova 2/16

Università degli Studi di Padova

## Un modello concorrente concreto

### Forma sintattica - 2

- **Esempio di specifica di tipo task**

```
task type Controller;
```

Parti discriminante, pubblica e privata vuote

```
task type Agent(Param : Integer);
```

Parte discriminante definita

```
task type Cashier_Attendant(Post : Post_Number := 1);
```

Parte discriminante definita, con valore iniziale predefinito modificabile (*default*)

```
task type Yellow_Pages is
entry Directory_Enquiry(
  Entity : in Name;
  Place : in Address;
  Number : out Telephone_Number);
end Yellow_Pages;
```

Parte pubblica definita con un punto di accesso per sincronizzazione

Corso di Laurea Specialistica in Informatica, Università di Padova 3/16

Università degli Studi di Padova

## Un modello concorrente concreto

### Una dichiarazione completa - 1

```
task type Customer(My_Id : Positive) is
pragma Priority(12);
end Customer;
task body Customer is
Outcome : Boolean;
begin
Status.Signal(Outcome);
if Outcome then
Service.Wait;
else
-- alternate action
end if;
end Customer;
```

```
protected Status is
entry Wait;
procedure Signal(Outcome : out Boolean);
private
In_Line : Max_In_Line := 0;
Present : Boolean := False;
end Status;
```

```
protected Service is
entry Wait;
procedure Signal;
private
Calling_For_Service : Boolean := False;
end Service;
```

```
type Customer_Ref is access Customer;
Customer_1 : Customer_Ref :=
new Customer(1);
Customer_2 : Customer(2);
```

Corso di Laurea Specialistica in Informatica, Università di Padova 4/16

Università degli Studi di Padova

## Un modello concorrente concreto

### Una dichiarazione completa - 2

- ① e ② rappresentano distinte modalità dichiarative di oggetti task
- Il processo dichiarato tramite ① viene attivato solo alla fine dell'elaborazione della parte dichiarativa, all'ingresso della parte esecutiva del modulo dichiarante
- ② invece crea dinamicamente un processo che viene attivato all'ingresso della parte esecutiva del modulo creatore (subito, se ② è in parte esecutiva)

Corso di Laurea Specialistica in Informatica, Università di Padova 5/16

Università degli Studi di Padova

## Un modello concorrente concreto

### Un esempio completo - 1

- **Il crivello di Eratostene**
  - Un processo (Odd) seleziona tutti i numeri dispari in un intervallo fissato
  - Un altro processo (S) seleziona tra i numeri dispari riscontrati quelli non divisibili per il primo numero primo a lui noto a priori (Prime = 3)
    - Il primo di questi numeri è certamente un numero primo (Num)
  - S passa Num e tutti i numeri dispari rimanenti non divisibili per Prime ad un suo clone (Next\_Sieve)
    - Questo clone ripete la sequenza sapendo che il primo numero che gli è pervenuto (Num) è un primo

Corso di Laurea Specialistica in Informatica, Università di Padova 6/16

# Un modello concorrente concreto

Università degli Studi di Padova

Un modello concorrente concreto

## Un esempio completo - 2

Corso di Laurea Specialistica in Informatica, Università di Padova

7/16

Università degli Studi di Padova

Un modello concorrente concreto

## Esercizio 5

- ❑ **Replicare la struttura concorrente proposta nel riquadro precedente, utilizzando il linguaggio Java**
- ❑ **Raffrontare la soluzione proposta con la soluzione replicata**

Corso di Laurea Specialistica in Informatica, Università di Padova

8/16

Università degli Studi di Padova

Un modello concorrente concreto

## Stati di vita di processo - 1

- ❑ **Un processo viene creato dall'elaborazione del modulo che ne contiene la dichiarazione**
- ❑ **L'esecuzione del processo attraverso 3 fasi**
  - **Attivazione** : nella quale viene elaborata la parte dichiarativa della sua realizzazione
  - **Esecuzione** : nella quale vengono eseguiti i comandi nella parte esecutiva della sua realizzazione
  - **Finalizzazione** : nella quale viene eseguito il codice di finalizzazione di ogni oggetto creato dalla sua parte dichiarativa

Corso di Laurea Specialistica in Informatica, Università di Padova

9/16

Università degli Studi di Padova

Un modello concorrente concreto

## Stati di vita di processo - 2

```

declare
  task type T_Type;
  task A;
  B, C : T_Type;
  task body A is ... end A;
  task body T_Type is ... end T_Type;
begin
  ...
end;
    
```

Corso di Laurea Specialistica in Informatica, Università di Padova

10/16

Università degli Studi di Padova

Un modello concorrente concreto

## Stati di vita di processo - 3

Corso di Laurea Specialistica in Informatica, Università di Padova

11/16

Università degli Studi di Padova

Un modello concorrente concreto

## Stati di vita di processo - 4

- ❑ **In un linguaggio a blocchi**
  - I processi possono essere dichiarati in ciascun blocco
  - I blocchi possono essere annidati gerarchicamente
  - Un processo è esso stesso un blocco
- ❑ **Ciò consente di realizzare gerarchie di processi**
  - Il processo padre è soggetto alle stesse regole del blocco contenitore per quanto concerne attesa per l'attivazione dei processi figli

Corso di Laurea Specialistica in Informatica, Università di Padova

12/16

# Un modello concorrente concreto

Università degli Studi di Padova Un modello concorrente concreto

## Stati di vita di processo - 5

- ❑ L'attesa per la terminazione di un processo è responsabilità della regione dichiarativa nella quale è avvenuta la sua attivazione (*master*)
  - Il processo padre non è sempre *master* dei suoi processi figli, ma può esserlo un blocco (*scope*) interno ad esso
  - Nel caso di creazione dinamica di processo (come al punto ② di pagina 4) ne è *master* la regione dichiarativa che ne definisce il tipo
  - Formalmente, un *master* non ha figli ma dipendenti!

Corso di Laurea Specialistica in Informatica, Università di Padova 13/16

Università degli Studi di Padova Un modello concorrente concreto

## Stati di vita di processo - 6

```

task type Dependent;
task body Dependent is ... end Dependent;

declare
type Dependent_Ref is access Dependent;
A : Dependent_Ref;
begin
...
declare
B : Dependent;
C : Dependent_Ref := new Dependent;
D : Dependent_Ref := new Dependent;
begin
...
A := C;
end;
end;
    
```

① è *master* di tutti i processi creati a partire dal tipo `Dependent_Ref`  
 Quali sono?  
 ② è *master* solo del processo B  
 Quando può terminare ②?  
 Quando può terminare ①?

Corso di Laurea Specialistica in Informatica, Università di Padova 14/16

Università degli Studi di Padova Un modello concorrente concreto

## Stati di vita di processo - 7

- ❑ Nel programma che realizza il crivello di Eratostene (pagina 6, esercizio 4) il modulo *package SoE* definisce vari processi (quali?) senza esserne il *master*
- ❑ In questo caso è il programma principale che diventa *master* mediante inclusione (`with`) del modulo creatore

Corso di Laurea Specialistica in Informatica, Università di Padova 15/16

Università degli Studi di Padova Un modello concorrente concreto

## Stati di vita di processo - 8

La presenza di gerarchie di processi aggiunge nuovi stati e nuove transizioni

Corso di Laurea Specialistica in Informatica, Università di Padova 16/16