Università degli Studi di Padova

Interazione con OOP

SCD

Anno accademico 2004/5
Corso di Sistemi Concorrenti e Distribuiti

Tullio Vardanega, tullio.vardanega@math.unipd.it

Inheritance anomaly - 1

- **L'interazione tra concorrenza ed OOP produce effetti indesiderabili**
  - **Metodi che contengono codice di sincronizzazione (per mutua esclusione, sospensione, risveglio) possono non essere ereditabili senza preventiva analisi dettagliata e/o modifiche della classe sorgente**
    - S. Matsuoka and A. Yonezawa
      Analysis of inheritance anomaly in object-oriented concurrent programming languages
      In: G. Agha, P. Wegner, and A. Yonezawa (editors), **Research directions in concurrent object-oriented programming**, pp. 107-150. MIT Press, 1993
  - **Tale eventualità viola il principio di incapsulazione**

Inheritance anomaly - 2

- **Situazioni canoniche di sincronizzazione**
  - **Esecuzione condizionale**
    - Le operazioni effettuabili possono dipendere da
      - **Stato interno** della risorsa
      - **Storia** di esecuzione
      - **Parametri** della richiesta
    - La natura della risorsa può necessitare accesso esclusivo
  - **Controllo di ordinamento**
    - Le richieste non immediatamente soddisfacibili possono venire accodate

Inheritance anomaly - 3

- **Ha luogo tipicamente all'introduzione di**
  - Diversi criteri di ammissibilità
    - La sottoclasse può introdurre diversa corrispondenza tra stati interni ed operazioni ammissibili
  - Modifica degli stati interni
    - La sottoclasse può introdurre nuovi stati interni, ignoti e non trattati dalla classe sorgente
  - Modifica delle dipendenze dalla storia d'esecuzione
    - La sottoclasse può imporre nuovi e diversi vincoli sulla ammissibilità di particolari richieste
- **Il che richiede conoscenza precisa ed eventuale modifica della logica originaria della classe sorgente**

Esempio

- ***Bounded buffer*** **con `get()` e `put()`**
  - Non vuoto → `get()`
  - Non pieno → `put()`
  - **Diversi criteri di ammissibilità**
    - Prelievo simultaneo di 2 articoli: `get2()`
      - Stato non originario → modifica della classe sorgente
  - **Modifica degli stati**
    - Risorsa inaccessibile → `lock()`
    - Risorsa accessibile → `unlock()`
      - Stato non originario → modifica della classe sorgente
  - **Modifica delle dipendenze della storia**
    - Preleva 1 articolo ogni N immissioni: `Nget()`
      - Stato non originario → modifica della classe sorgente