CÁC MÔ HÌNH LẬP TRÌNH TIÊN TIẾN

Đỗ Thị Bích Ngọc
PTIT/FIT/SE
dothibichngoc@gmail.com

Nội dung môn học

- TỔNG QUAN CÁC MÔ HÌNH LẬP TRÌNH
- LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG
- LẬP TRÌNH PHÂN TÁN
- LẬP TRÌNH HÀM
- LẬP TRÌNH RÀNG BUỘC
- MỘT SỐ MÔ HÌNH TIÊN TIẾN KHÁC
  - Lập trình giao diện người dùng
  - Lập trình trong kiểm thử tự động
  - Kiến trúc hướng dịch vụ và Lập trình hướng dịch vụ

Tài liệu tham khảo

[1] Roy V. Peter, *Programming paradigms for dummies: What every programmer should know*, New Comput. Paradigms for Comput. Music, 2009.

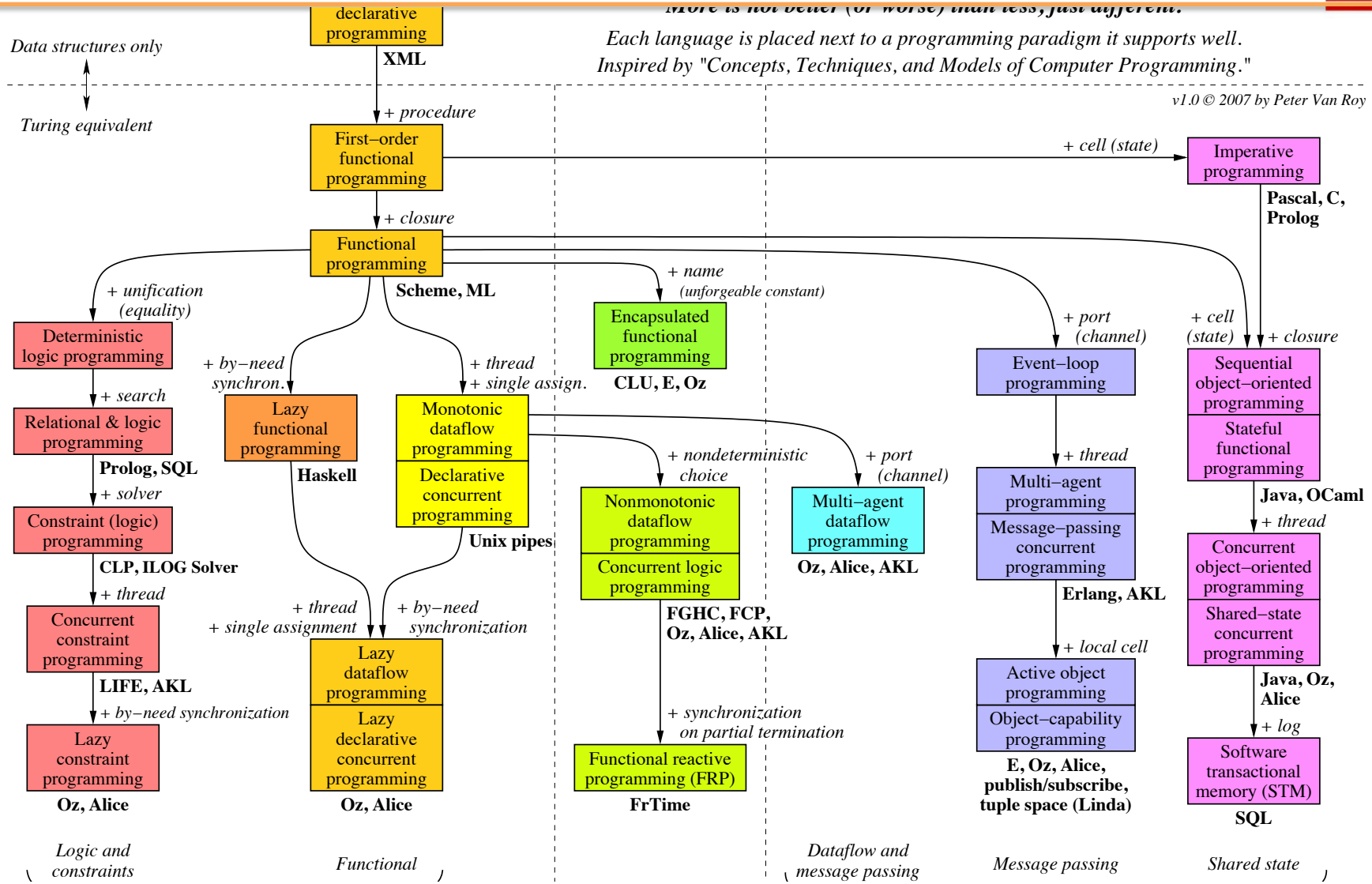
[2] Jeff Magee and Jeff Kramer, *Concurrency: State Models & Java Programs*, 2nd Edition, John Wiley & Sons, 2006.

[3] Eben Hewitt, *Java SOA cookbock*, O'Reilly, 2009

[4] Charles W. Kann, *Creating Components: Object Oriented, Concurrent, and Distributed Computing in Java*, Auerbach Publications, 2004

[5] Programming Paradigms: Lecture Notes:http://www.cse.chalmers.se/~bernardy/pp/Lectures.html

Sơ đồ các mô hình lập trình



Bốn mô hình lập trình chính

Đỗ Thị Bích Ngọc dịch từ bài giảng
Dan Garcia, UC Berkeley EECS

Các mô hình lập trình

- Các mô hình lập trình là gì?
  - Hầu hết là Hybrids!
- Bốn mô hình chính
  - Functional – hàm
  - Imperative – không điều kiện
  - Object-Oriented – hướng đối tượng
  - Declarative – Khai báo

en.wikipedia.org/wiki/Programming_paradigm

Mô hình lập trình là gì ?

- Các khái niệm và sự trừu tượng hoá sử dụng để biểu diễn các thành phần của chương trình (ví dụ, đối tượng, hàm, biến, ràng buộc…), và các bước để hợp thành 1 sự tính toán (gán giá trị, tính toán,luồng dữ liệu...)

- Hoặc, một cách để phân loại kiểu lập trình(classify the style of programming.)

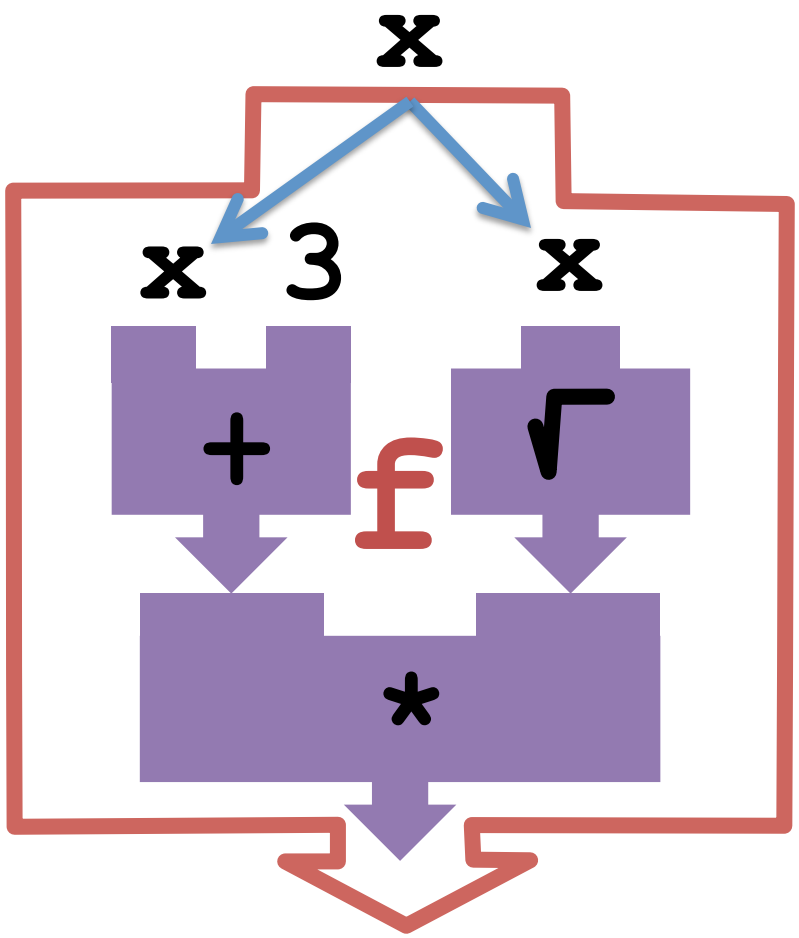
Đa số các ngôn ngữ là Hybrids!

- Hầu hết các ngôn ngữ kết hợp nhiều mô hình
  - Gọi là ngôn ngữ "Multi-paradigm"
  - Ví dụ: Scratch
- Giống như nước hoa quả (với nhiều loại quả) và yêu cầu cảm nhận vị 1 loại quả trong đó ☺

Functional Programming (giới thiệu)

- Chương trình thực hiện tính toán các hàm
  - Kết hợp các pipes
  - Mỗi pipe, hoặc function, có đúng 1 đầu ra
  - Functions có thể là đầu vào !
- Đặc điểm:
  - Không trạng thái
    - Ví dụ, gán giá trị cho biến
  - No mutation
    - Ví dụ, thay đổi giá trị của biến
  - No side effects
- Ví dụ?
  - Scheme, Scratch BYOB

$$f(x) = (x+3) * \sqrt{x}$$



en.wikipedia.org/wiki/Imperative_programming

Imperative Programming

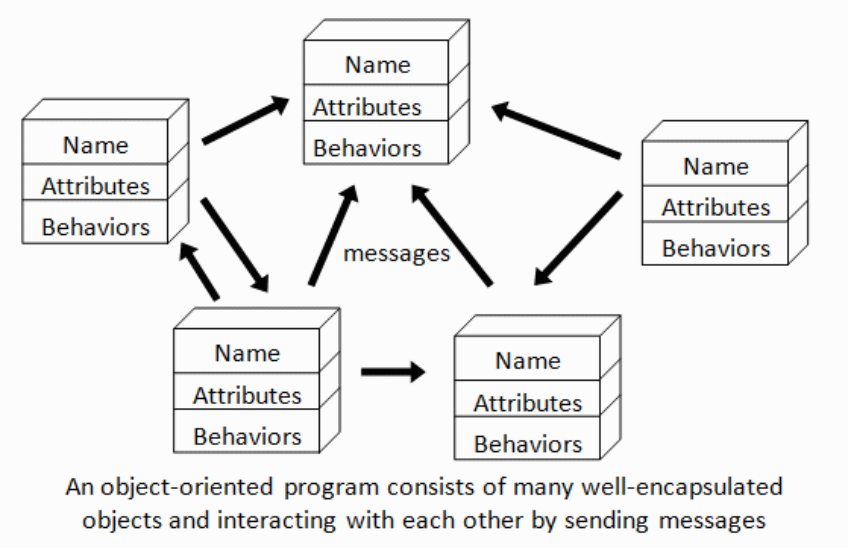
$$f(x) = (x+3) * \sqrt{x}$$

- Lập trình tuần tự Chương trình là một chuỗi các bước
  - Gán giá trị
  - Thay đổi giá trị
- Giống như theo các bước của công thức nấu ăn. Ví dụ ,
- Hàm f(x)
  - ans = x
  - ans = ans
  - ans = ($\sqrt{x}$+3) * ans
  - return ans
- Ví dụ?
  - Pascal, C

en.wikipedia.org/wiki/Object-oriented_programming

Lập trình hướng đối tượng (OOP)

- Objects như các cấu trúc dữ liệu
  - Với các methods
    - Gọi là behaviors
  - Với local state,
    - Gọi là attributes
- Inheritance tái sử dụng code
- Ví dụ?
  - Java, C++



An object-oriented program consists of many well-encapsulated objects and interacting with each other by sending messages

www3.ntu.edu.sg/home/ehchua/programming/java/images/OOP-Objects.gif

OOP in BYOB


new counter
script variables count
set count to 0
the script
change count by 1
report count
report
set counter1 to new counter
set counter2 to new counter
say call counter1 for 2 secs
say call counter1 for 2 secs
say call counter1 for 2 secs
think call counter2 for 2 secs
think call counter2 for 2 secs
say call counter1 for 2 secs

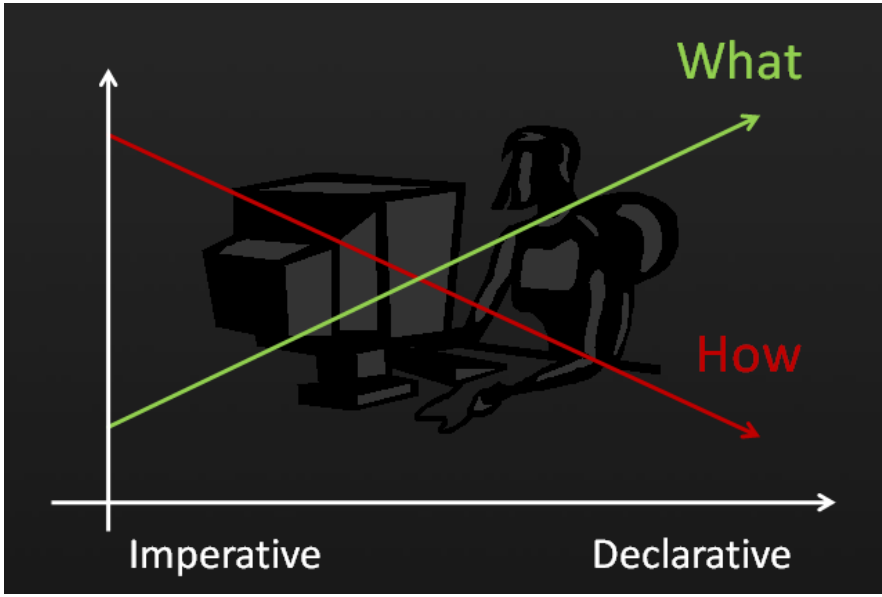

1
2
3
1
2
4


run Dance of Girl

en.wikipedia.org/wiki/Declarative_programming

Declarative Programming

- Biểu diễn chương trình mong muốn gì (what) mà không xác định làm thế nào để đạt được nó (how)
  - Thường là một chuỗi các giả thiết và truy vấn,
  - Cảm tưởng như điều thần kỳ!
- Sub-categories
  - Logic
  - Constraint
- Ví dụ: Prolog



Anders Hejlsberg
"The Future of C#" @ PDC2008
channel9.msdn.com/pdc2008/TL16/

en.wikipedia.org/wiki/Programming_paradigm

Tóm tắt

- Functional
  - tính giá trị của biểu thức và sử dụng kết quả tính toán cho bước tiếp
- Object-oriented
  - Gửi thông điệp giữa các đối tượng để mô phỏng tính toán hiện thời của một tập các hiện tượng trong thế giới thực
- Imperative
  - *Làm cái này, rồi làm tiếp cái kia*
- Declarative
  - Trả lời một câu hỏi thông qua tìm kiếm đáp án

www.cs.aau.dk/~normark/prog3-03/html/notes/paradigms_themes-paradigm-overview-section.html

Kết luận

- Mỗi mô hình có 1 ưu điểm riêng
  - Nếu một ngôn ngữ là máy Turing đầy đủ, nó ngang bằng về sức mạnh
  - Các mô hình đa dạng về tính hiệu quả, khả năng mở rộng, mục đích cho “how” vs “what”
- Các ngôn ngữ lập trình hiện đại thường kết hợp các mô hình tốt nhất
  - Ví dụ., Scratch
    - Có thể là functional
    - Có thể là imperative
    - Có thể là object-oriented
    - Có thể là declarative