Metapatterns

Prof. Rushikesh K. Joshi

Department of Computer Science and Engineering
Indian Institute of Technology Bombay

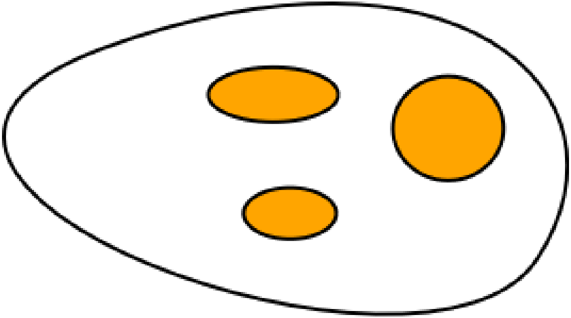
Outline

1. Inheritance Structures
2. Non-conceptual Inheritance
3. More on Conceptual Inheritance
4. Accepting one Abstraction in Place of Another: Subtyping
5. Frameworks, Metapatterns

Application Framework

- Building blocks are ready to use
- They may be semi-finished
- Specific applications can be produced by adjusting the semi-finished blocks

Frameworks and Hot-spots

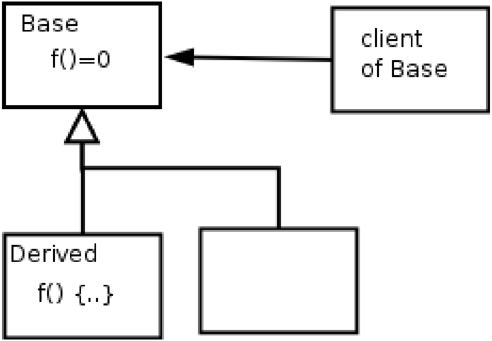


Rigid + Flexible

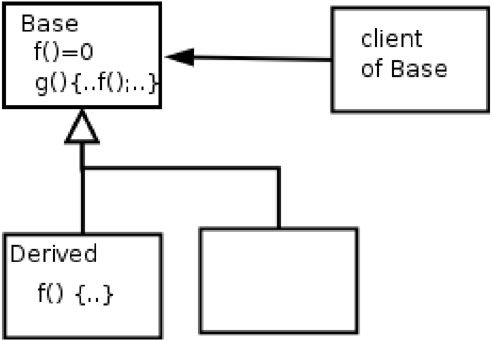
Meta-patterns

- =set of design patterns
- Describe how to construct frameworks independent of a specific domain
- i.e. the basic ingredients of framework making
- Very close to principles of object orientation
- Are at meta level, Complementary to main-stream design patterns

Hook Methods

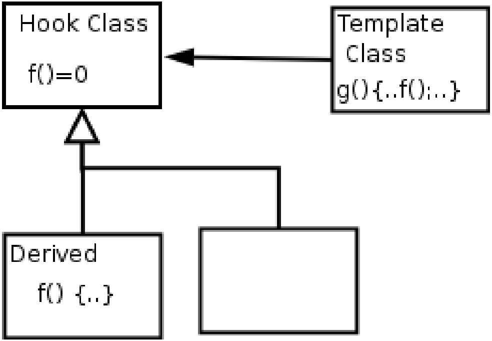


Template Methods



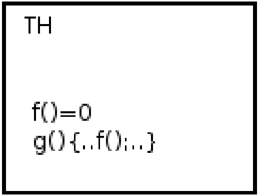
* We are not talking about type templates such as those in C++

Template Methods may be located in a Different Class



* where are template methods located? —- in subclass? in client of hook class?
* how many instances of hook class does the template class refer to?

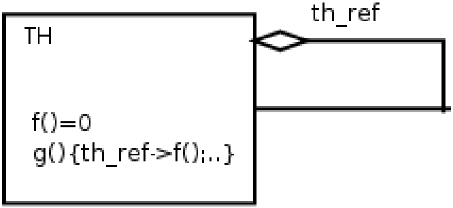
'Unification' Pattern



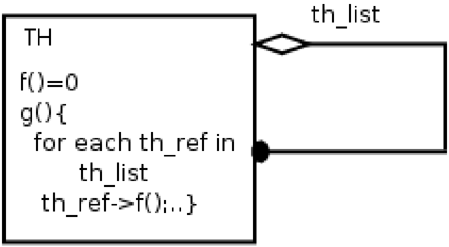
TH

```
f()=0
g(){..f();..}
```

’Recursive 1:1 Unification’ Pattern



'Recursive 1:N Unification' Pattern



Connection Patterns: When T is not located with H

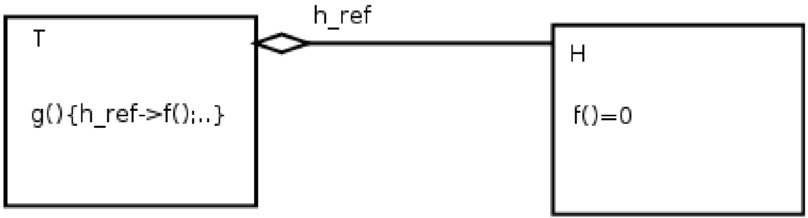
* 1:1 Connection

* 1:N Connection

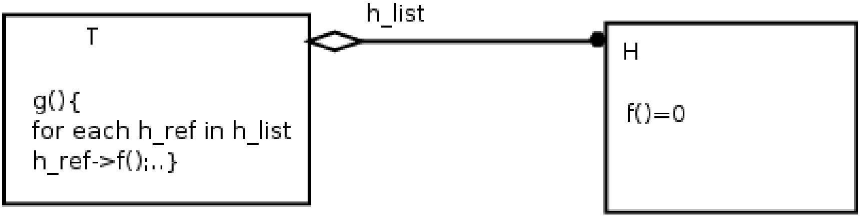
* 1:1 Recursive connection

* 1:N Recursive connection

1:1 Connection Pattern



1:N Connection Pattern



T

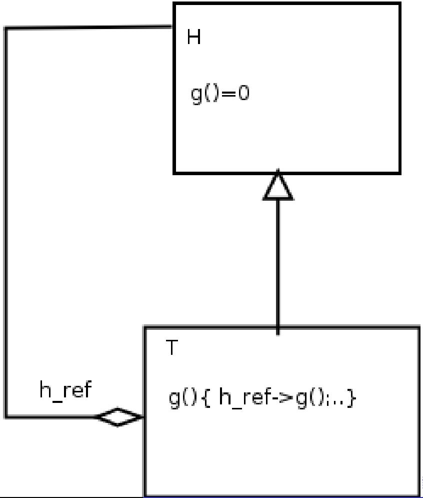
```
g(){
for each h_ref in h_list
h_ref->f();..}
```

h_list

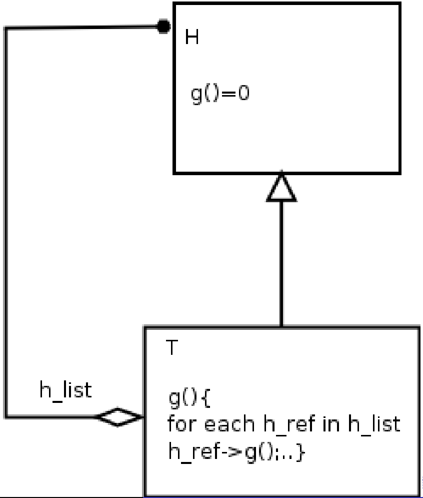
H

f()=0

1:1 Recursive Connection Pattern



1:N Recursive Connection Pattern


H
g()=0
h_list
T
g(){
for each h_ref in h_list
h_ref->g();..}