Refactoring UML models

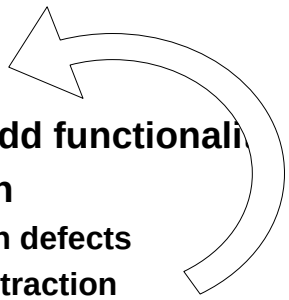
Pr. Jean-Marc Jézéquel
IRISA - Univ. Rennes I

Campus de Beaulieu
F-35042 Rennes Cedex
Tel : +33 299 847 192 Fax : +33 299 842 532
e-mail : jezequel@irisa.fr
http://www.irisa.fr/prive/jezequel

IRISA

Evolutionary software development

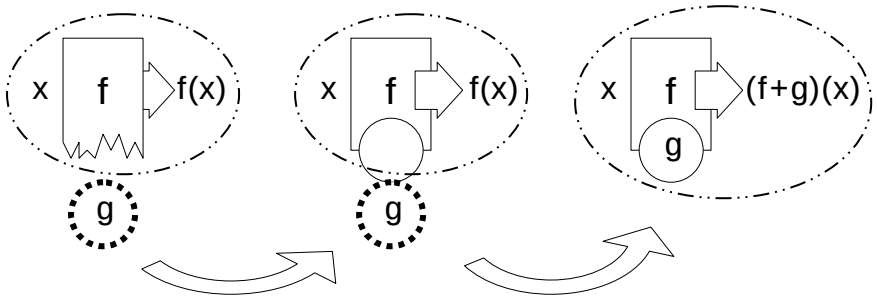
- **Initial design**
- **Evolution**
  - **Expansion: add functionali**
  - **Consolidation**
    - **correct design defects**
    - **introduce abstraction**



"iterative incremental use-case driven"

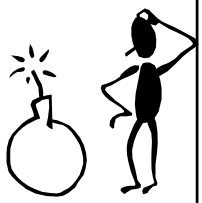
IRISA

What is refactoring ?



refactoring
(behavior-preserving)

integration
(adding behavior)



➔ **impact of each modification ?**

IRISA

Refactoring & UML ?

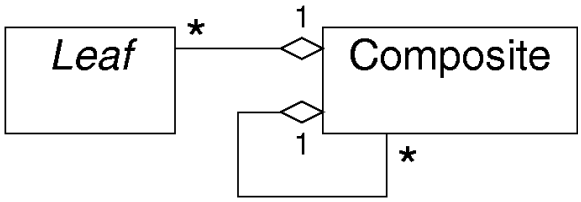
- **Source code refactoring :**
  - **Opdyke, Brant, Roberts, Johnson…**
  - **Standard refactorings for Java available within Eclipse**
- **Recent methods handle evolution**
  - **XP, Joint Application Development…**
  - **Catalysis ➔ UML**

IRISA

Example of UML refactorings for Class diagrams

- renaming, signature changes
- structure modifications

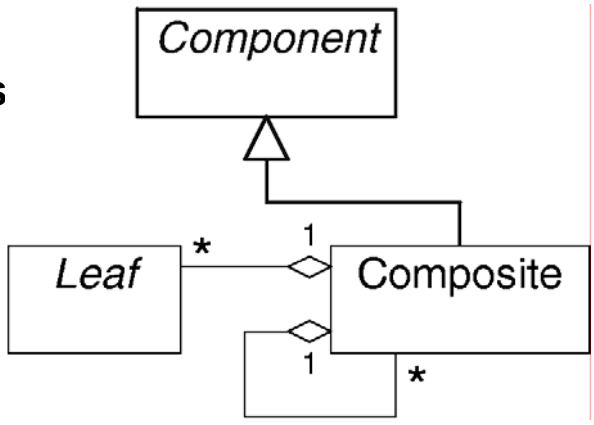
- application of a design pattern:



IRISA

Class diagrams 2/4

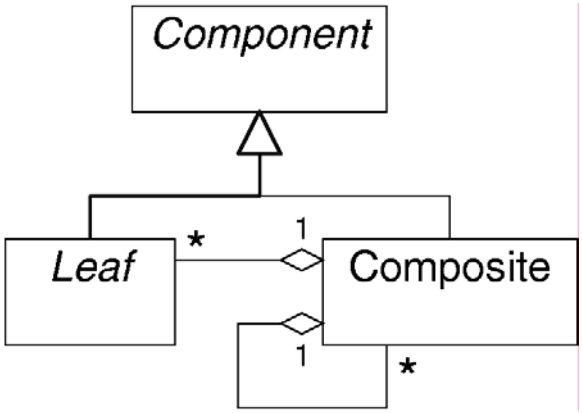
- add a new abstract superclass



IRISA

Class diagrams 3/4

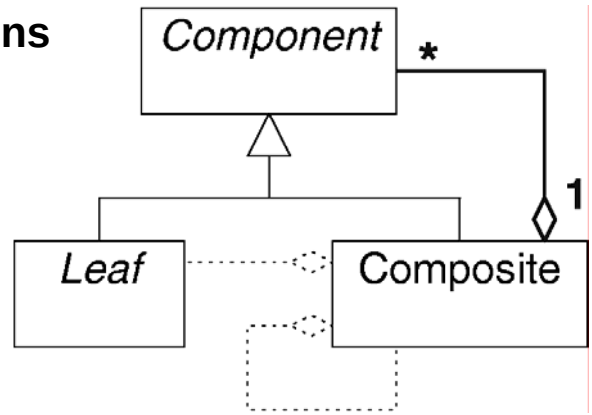
- **make Leaf a subclass**



IRISA

Class diagrams 4/4

- **merge the aggregations**



IRISA

Class diagram refactorings

Structural changes :

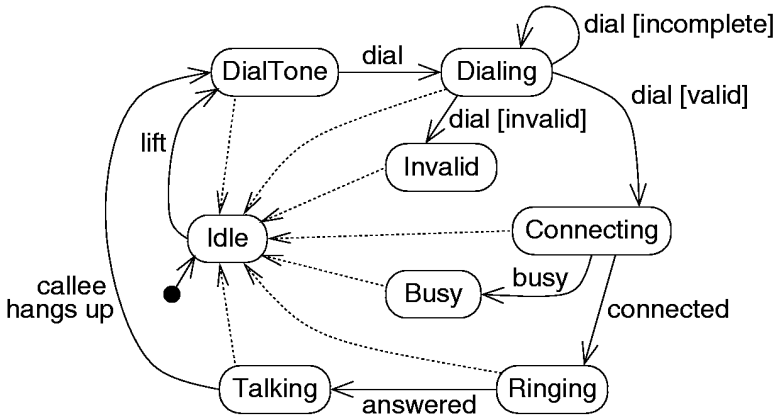
- **add - remove - move - rename**
- **generalize - specialize**
  - **move through inheritance hierarchy**
- **merge Aggregations**

➔ **design pattern introduction**

IRISA

Statecharts

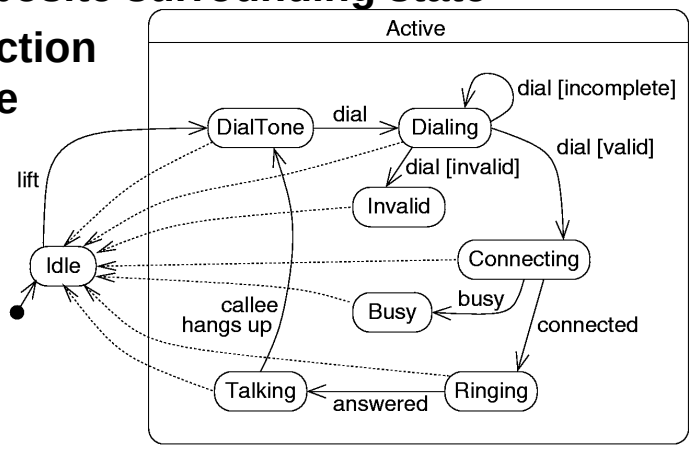
- **phone communication**



IRISA

Statecharts 2/3

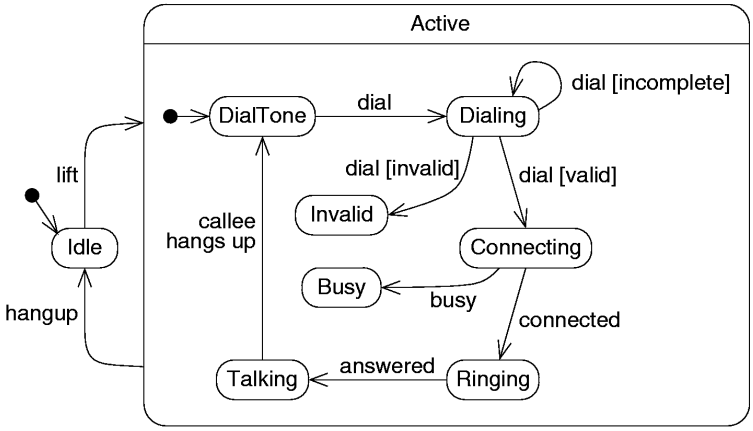
- **Composite surrounding state**
- **extraction of Idle**



IRISA

Statecharts 3/3

- **fold transitions to high-level**



IRISA

Statechart refactorings

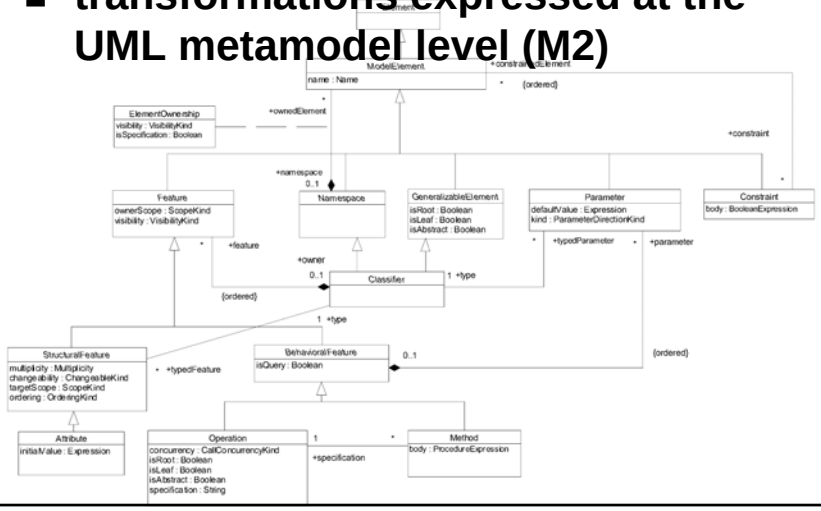
Behavioral changes

- **add, remove state**
- **(un)fold entry / exit actions**
- **(un)fold high-level transitions**
- **group into a composite**
- **move substate into / out of composite**

IRISA

How UML refactoring works

- **transformations expressed at the UML metamodel level (M2)**



Specification of refactorings

- meta-level OCL
- preconditions
  - applicability (behavior preserved)
- postconditions
  - describe transformation effect
  - ensure respect of well-formedness rules

IRISA

Sample pre-post

```
context Classifier :: addMethod(added : Method)
pre:
   added.parameter -> forAll(p1,p2 |
      p1.name=p2.name implies p1=p2) and
   self.allMethods().name -> excludes(added.name) and
   self.allOperations -> exists(op |
      op.hasSameSignature(self))
post:
   let m = self.allMethods() -> select(m |
      m.isEquivalentTo(added))
   in
   m -> notEmpty() and
   self.allOperations -> exists(m.specification)
```

IRISA

Conclusion

- UML model refactoring makes sense
- Initial set of UML refactorings…
- … to be expanded to other views
  - collaborations, activity diagrams…
  - executable models (Action Semantics)
- … to be widened
  - less conservative, weaker preconditions
- Tool support
  - currently being implemented in Umlaut
    `http://modelware.inria.fr`

IRISA