Design pattern advantages

- A lot of work
  - Good the employment
  - From 20 to 100 lines of code
- For experts only
  - Impossible to understand for non experts
  - So youngster, you don't know patterns ahum
- Obfuscation
  - The functionality cannot be recognized from the code
  - Hard to maintain

Abstraction

- Java hasn't enough abstraction
  - In Lisp 16 from the 23 GOF pattern disparate
  - The same for aspectJ, scala
- Design pattern in java are
  - Function pointers
    - Interfaces, anonymous inner class
  - Aspects
    - Delegation, proxy
  - Metaprogramming
    - Runtime programming, generators

Antipattern

- Design pattern were a hype
  - Badly understood
  - To often used
- NOT a catalog of code templates and models
- They are design directions
  - Intent, motivation, implementation, consequences
  - Nobody really read the GOF book

New directions

- GOF book was written in 1994
- Many new developments
  - Refactoring
    - Patterns as refactoring directions
  - Containers, frameworks, generators
    - Patterns as skeletons
  - Aspect oriented, meta programming
    - Same problems, completely different solutions
  - OO + functional programming
    - SCALA

Complexity

- Programmer
  - Frustration, tunnel vision, hiding
- Team
  - Islands, walls, throwing over wall, pointing
- Managers
  - Say it ain't so, it cannot be that difficult, divide and conquer
- Customer relation
  - Tactics, politics
- Projects fail

Conclusion

- Only use pattern when really necessary
- Patterns are a solution to a problem
  - Ask yourself: ”What is the problem?”
  - If they don't solve a problem they create a problem
- Use functions
- See patterns as a catalog of directions