

Event-based Aspect-Oriented Programming

`www.emn.fr/x-info/eaop`

Kick-off meeting, ACI Dispo, 18/11/2003

Aspect oriented programming in a nutshell

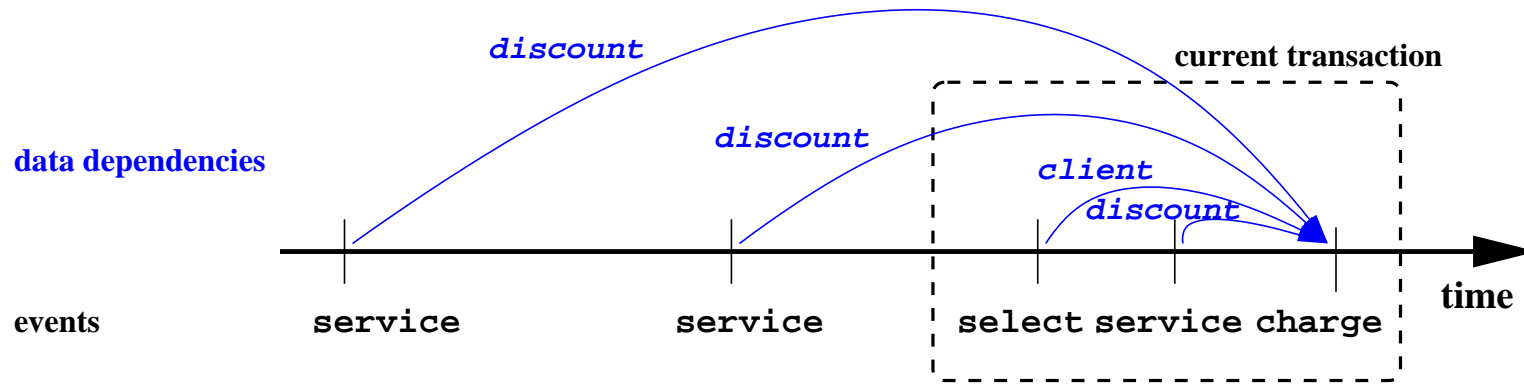
- Base program + aspects = woven program
- Operating system + security aspect + debugging aspect = OS with encrypted logs
- An aspect = a module?
No! Aspect code is **distributed everywhere** in the base program
- Linguistic support and compiler (weaver) required
- Anatomy of an aspect
 - Crosscut: where the aspect code is inserted
 - Action: what aspect code is inserted
- Weaver: how to insert code at the right place

Event-based AOP

- Abstract execution points by events
- Aspect definitions
 - Crosscut \rightarrow Action
 - Crosscut: sequences of execution events
 - Action: executed once corresponding crosscut matched
- Advanced facilities for aspect composition
 - Resolution of interacting aspects
 - Ex.: impose order, do not apply aspects in certain contexts
 - Dynamically create aspects
 - Application of aspects to other aspects
- Weaver (conceptually):
Execution monitor matching crosscuts and triggering actions

Ex.: crosscuts in e-commerce (1)

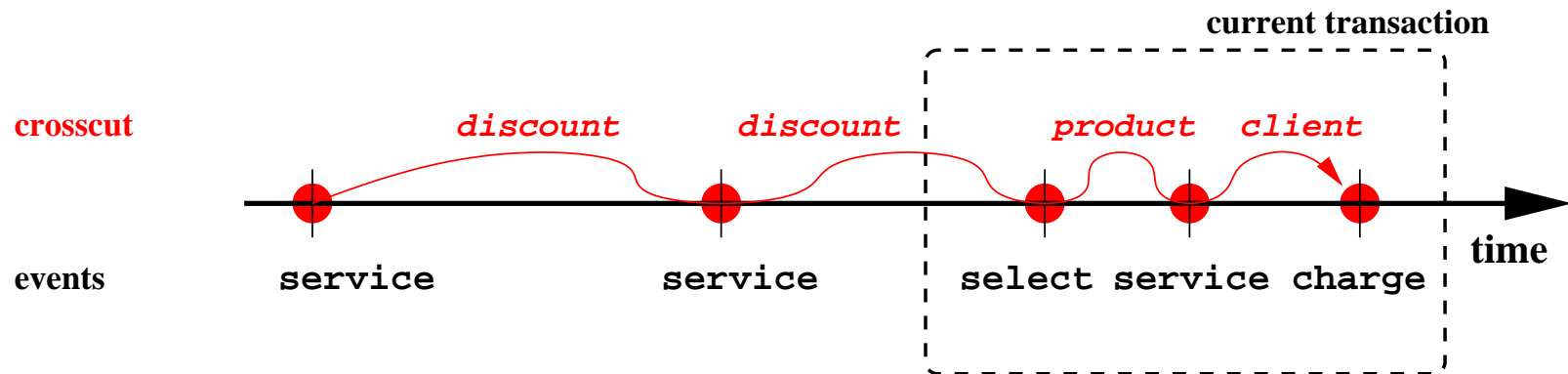
Problem: extending e-commerce application with new functionality (here: billing)



- Payment depends on preceding purchases (discount) and authentication (client info)

Ex.: crosscuts in e-commerce (2)

- EAOP: non-atomic crosscuts including relations between events
- Ex.: precise definition of the billing crosscut which relates authentication events, purchase events, etc.



Current results

- Theoretical basis
 - Formal def. for Turing-complete crosscut language
Manual proofs of equivalences between aspects
 - Crosscut definitions restricted to regular expressions (\rightarrow joint work with Pascal)
Automatic static analysis of aspect interactions; aspect composition operators for conflict resolution
- Tool support: EAOP for Java
 - Turing-complete aspects
 - Aspect composition operators for conflict resolution, dynamic aspect instantiation, aspects of aspects
 - Different implementation strategies: trade-off conceptual simplicity vs. efficiency
- Applications
 - E-commerce
 - Re-engineering of OS code

EAOP and DISPO

- Security: prime example of an aspect
- Incorporate formally-defined security properties into aspect language
- Provide provably correct implementation from scratch or by re-engineering of existing code.