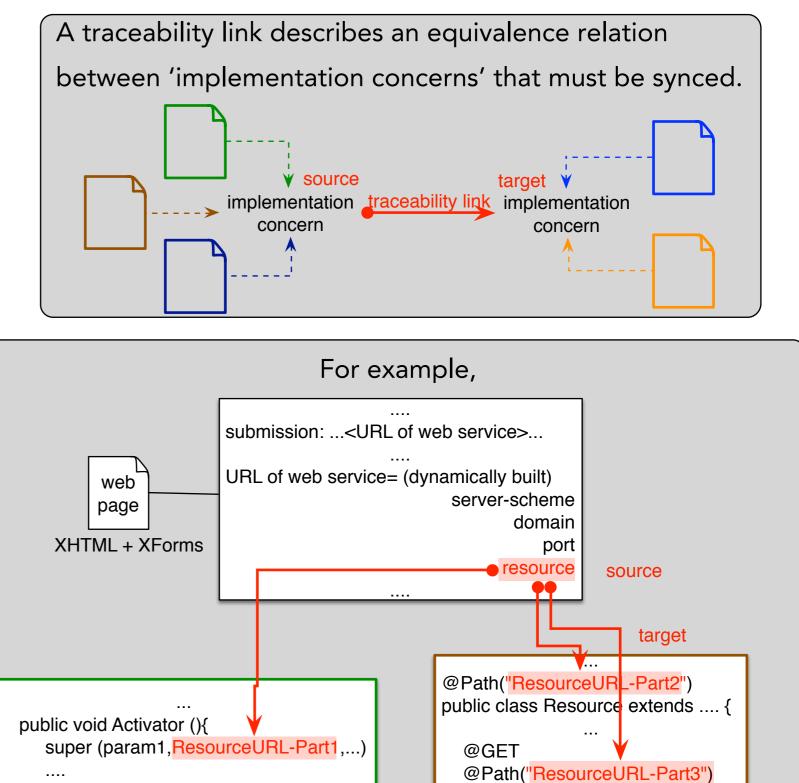


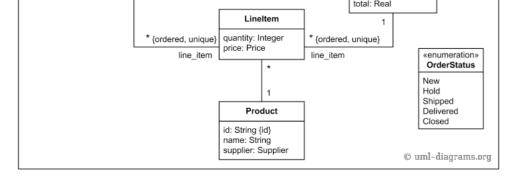
# MaTraCa: Maintaining Traceability Changes

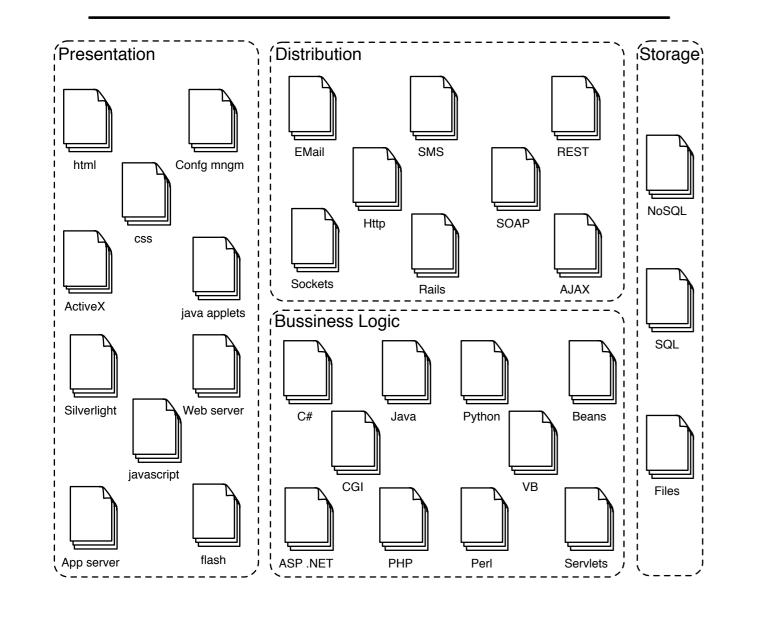
**Research on Tracing changes** 

### Traceability «Subsystem» **Online Shopping** View Items «Service» Authentication winclude ✓ There is a wide variety of software Make Registered Purchase artefacts created along the Customer Identity development process. Provider «include» Web Checkout Customer ✓ Maintaining their internal consistency is a time-consuming and error-prone task. Credit Payment Service New Client Customer Register ✓ Traceability allows you to link related artefacts. © uml-diagrams.org PayPal ✓ Maintaining traceability links can ensure the correct and complete propagation class Online Shopping Customer of changes (i.e., co-evolution). Web User id: String {id} login\_id: String {id} address: Address password: String state: UserState phone: Phone email: String ✓ Research in vertical traceability (i.e., Payment 0..\* id: String {id} «enumeration» UserState across artefacts at different layers of paid: Date total: Real New Active Blocked Banned Account details: String abstraction) focuses on reverse id: String {id} {ordered, unique} lling address: Address \_closed: Boolean engineering these links, but this open: Date closed: Date {ordered unique} extraction is not always cost-effective Order Shopping Cart umber: String {id} (especially at a fine granularity level). reated: Date dered: Date .... shipped: Date ship\_to: Address status: OrderStatu

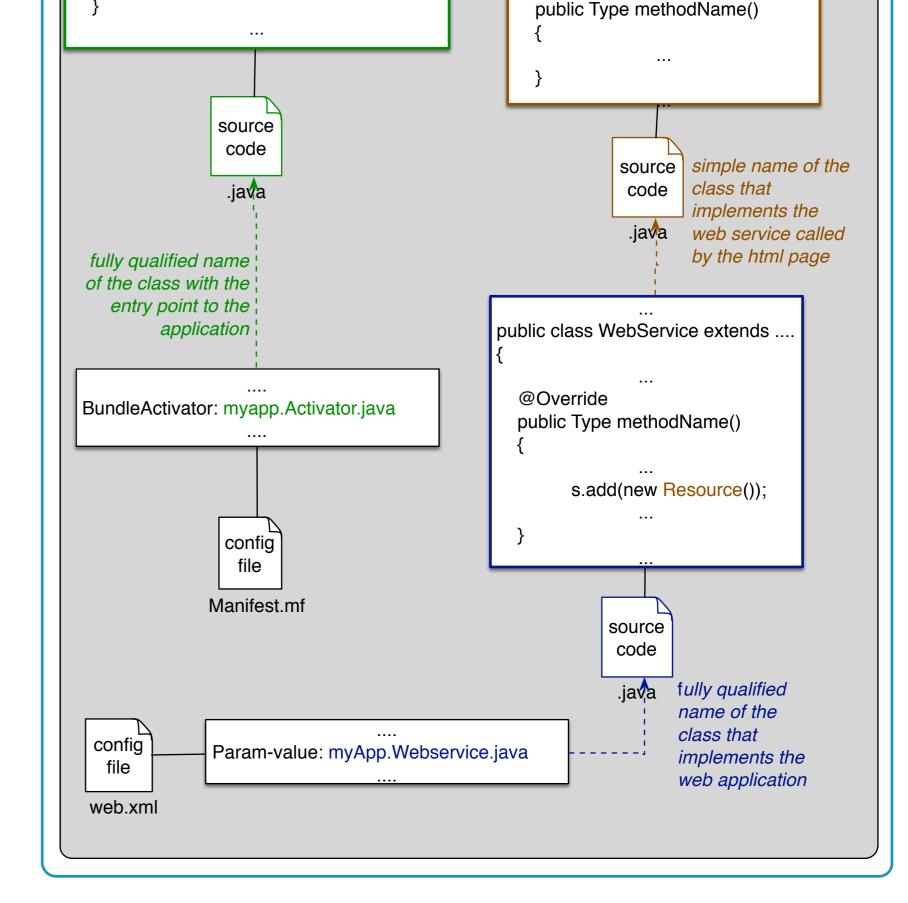
# Defining one traceability link







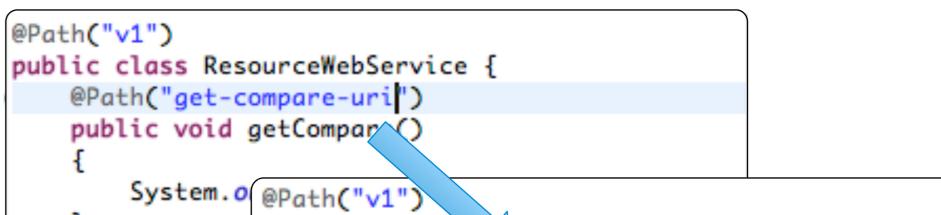
Horizontal traceability (i.e. across artefacts at the same level of abstraction) could provide a costeffective mechanism to evolve complex applications.



### Detecting source & target concerns

## Validating links

🗐 Console 🥥 Link View 🔀		
! Source	Link status	Target
≤ s-do-compare	>	test.ResourceWebService.getCompare()
S−do−merge	>	test.ResourceWebService.getMergeURI(f
s-revert	-X->	
s-get-workspace-name	-X->	
A		

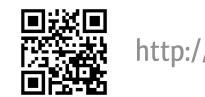


public class k urceWebService {
@Path("s-do-compare")
public void getCompare()
{
 System.out.println("dummy test method");
}

# Choosing interesting links

😑 Console 🥥 Link View 🔀		i i i 🗸 🗖 🗖
! Source	Link status	Target
S-do-compare	>	test.ResourceWebService.getCompare()
S−do−merge	>	test.ResourceWebService.getMergeURI(final Strin
s-revert	-X->	
s-get-workspace-name	-X->	

📮 Console 🥥 Link View 🖾		i i i ▽□□
! Source	Link status	Target
S−do−merge	>	test.ResourceWebService.getMergeURI(final S
S−do−compare	-X->	





Explaining Why Methods Change Together A. Lozano, C. Noguera, and V. Jonckers Proc. of IEEE Int'l Working Conference on Source Code Analysis and Manipulation SCAM, pp. 185 - 194. 2014.





