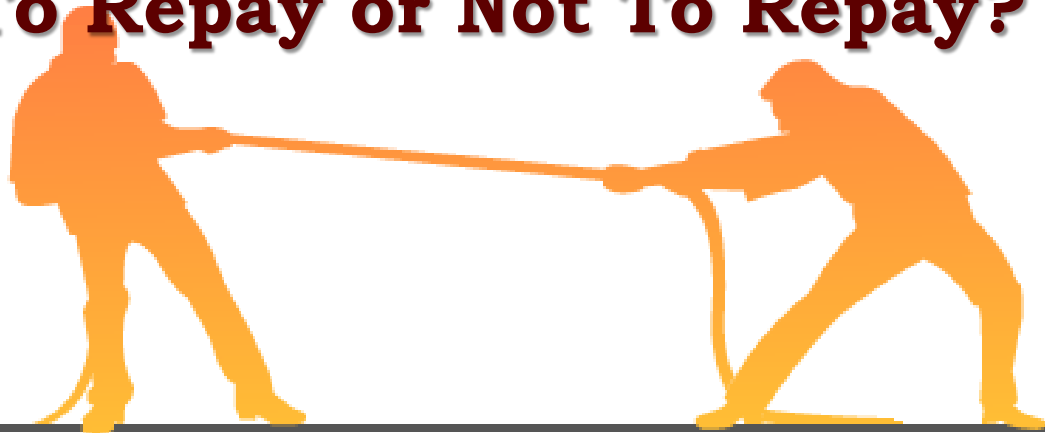


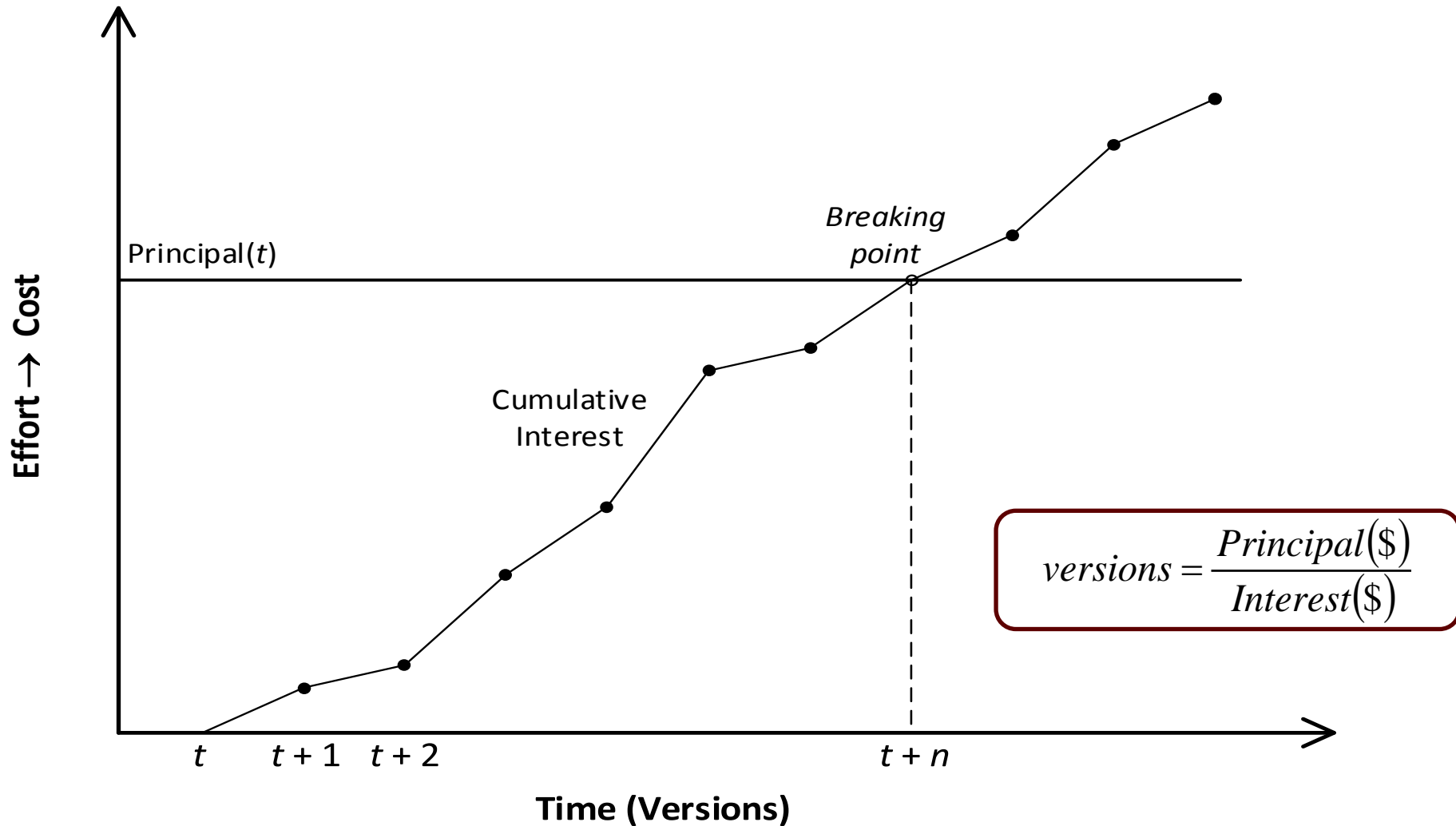
Estimating the Breaking Point for Technical Debt

Alexander Chatzigeorgiou,
 Apostolos Ampatzoglou,
 Areti Ampatzoglou,
 Theodoros Amanatidis



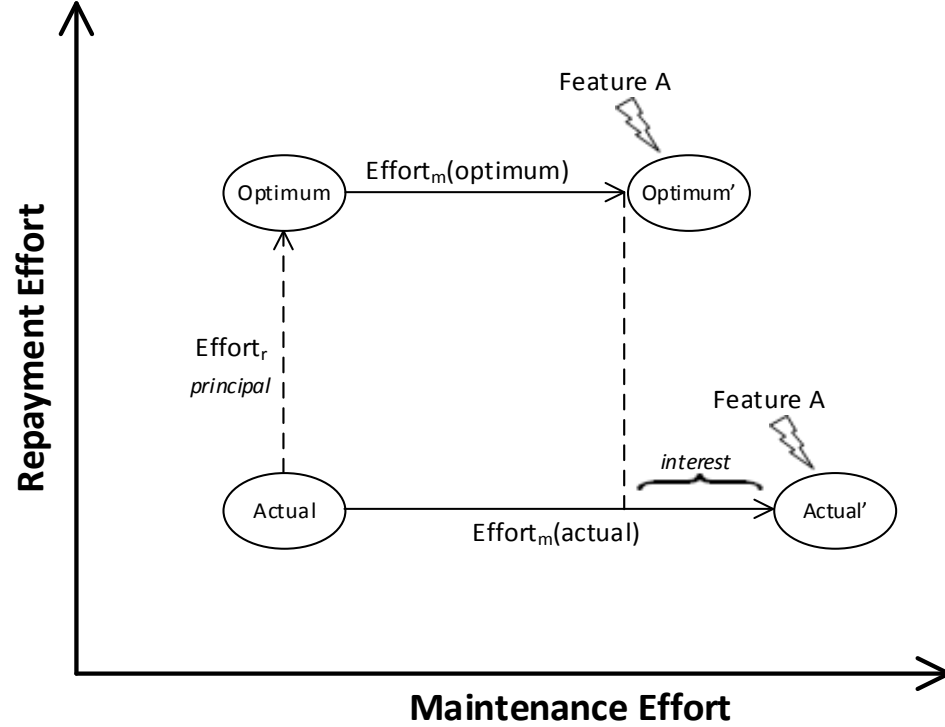
To Repay or Not To Repay?







Proposed Approach



$$Principal = Effort_r$$

$$Interest = \Delta Effort_m$$



Tool: jCaliper

- optimum design

Entity Placement → Coupling & Cohesion



Illustrative Example

RESEARCH DESIGN

JUnit version: 4.1

147 classes

1) Principal Estimation?

2) Interest Estimation?

3) Breaking Point?



Entity Placement				Effort		
actual	optimum	distance	Improvement	#Refactorings	time	Cost
0.819 ^a	0.692	0.127	15.5%	330	41.25h	1,891\$

average added LOC	Actual Effort		Optimum Effort	
	time	Cost	time	Cost
848	33.9h	1,553\$	28.6h	1,310\$

$$\text{Interest} = \text{Effort}(\text{actual}) - \text{Effort}(\text{optimum}) = 243\$$$

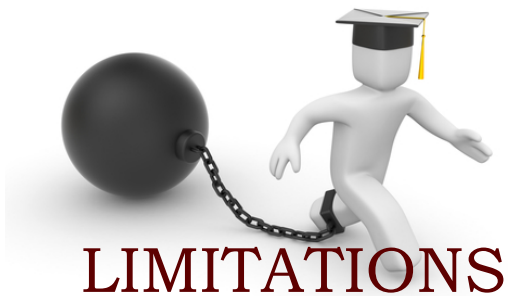
Elapsed time to breaking point = 7.78 versions

Principal estimation:

- AVG 5-10 mins/refactoring (W. C. Wake)
- 5.73\$/refactoring (US Bureau of labor statistics)

Interest estimation:

- AVG 848 added LOC between any two successive versions (T. Chaikalis, E. Ligu, G. Melas, and A. Chatzigeorgiou)
- 25 LOC/hour JAVA productivity (L. Prechelt)
- \$45.81 hourly wage (US Bureau of labor statistics)



- TD has other dimensions beyond coupling and cohesion
- Maintenance effort should account for other activities beyond addition of LoC (e.g. modifications, deletions)
- Future maintenance effort cannot be predicted solely on the basis of past maintenance tasks
- Cost estimates for refactoring application and introduction of new code, do not reflect commonly agreed figures

Questions?

Thank you for your attention!