Software Evolution

„All programming activity that is intended to generate a new software version from an earlier operational version“ *Manny Lehman and Juan Ramil (2000)*

Topic	Paper
1. Overview of Software Evolution	• Meir M. Lehman, Juan F. Ramil, P. Wernick, Dewayne E. Perry, Wladyslaw M. Turski: Metrics and Laws of Software Evolution - The Nineties View. IEEE METRICS 1997. • David Lorge Parnas: Software Aging. Proceedings of the 16th International Conference on Software Engineering, May 16-21, 1994, Sorrento, Italy. Pages 279-287 • L.A. Belady and M.M. Lehman. A Model of Large Program Development. *IBM Systems Journal*, 15(3), 1976, pages 239-248.
2. Software Decay	• S.G. Eick, T.L. Graves, A.F. Karr, J.S. Marron, and A. Mockus. Does code decay? assessing the evidence from change management data. *IEEE Transactions on Software Engineering*, 27(1), 2001. • J. van Gurp and J. Bosch. Design erosion: problems and causes. *The Journal of Systems and Software*, 61(2), 2002.
10. Analysis of Version Control Archives	• T. Ball, J.-M. Kim, A. A. Porter, H. P. Siy. If your version control system could talk ..., ICSE '97 Workshop on Process Modelling and Empirical Studies of Software Engineering. • Audris Mockus, Roy T. Fielding, James D. Herbsleb: Two case studies of open source software development: Apache and Mozilla. ACM Trans. Softw. Eng. Methodol. 11(3): 309-346 (2002) • Audris Mockus, Lawrence G. Votta: Identifying Reasons for Software Changes using Historic Databases. International Conference on Software Maintenance (ICSM'00), 11-14 October 2000, San Jose, California, USA, Proceedings. 120-130
11. Guiding programmers	• Davor Cubranic and Gail C. Murphy. "Hipikat: Recommending Pertinent Software Artifacts", Proc. 25th International Conference on Software Engineering (ICSE), May 2003. • Thomas Zimmermann, Peter Weißgerber, Stephan Diehl, and Andreas Zeller. Mining Version Histories to Guide Software Changes. Proc. 26th International Conference on Software Engineering (ICSE), Edinburgh, UK, May 2004.
5. Feature Location	• T. Eisenbarth, R. Koschke, and D. Simon. Aiding program comprehension by static and dynamic feature analysis. In *Proceedings of the IEEE International Conference on Software Maintenance*, 2001. • W. Zhao et al. SNIAFL: Towards a static non-interactive approach to feature location. In *Proceedings of the 26th International Conference on Software Engineering*, 2004.
7. Refactoring	• W.G. Griswold et al. Tool support for planning the restructuring of data abstractions in large systems. In *Proceedings of the ACM SIGSOFT Symposium on the Foundations of Software Engineering*, 1996. • T. Mens and T. Tourwé. A Survey of software refactoring. *IEEE Transactions on Software Engineering*, 30(2), 2004. • Eclipse-Demo
6. Software architecture analysis	• D. Garlan and D. Perry. Introduction to the special issue on software architecture. *IEEE Transactions on Software Engineering*, 21(4), 1995. • D. Garlan, R. Allen, J. Ockerbloom. Architectural mismatch, or, Why it's hard to build systems out of existing parts. In *Proceedings of the 17th International Conference on Software Engineering*, 1995. • J. Aldrich, C. Chambers, and D. Notkin. ArchJava: connecting software architectures to implementation. In *Proceedings of the 24th International Conference on Software Engineering*, 2002.
3. Dynamic Analysis	• Michael D. Ernst, Jake Cockrell, William G. Griswold, David Notkin:

	Dynamically Discovering Likely Program Invariants to Support Program Evolution. IEEE Trans. Software Eng. 27(2): 99-123 (2001) • Jeff H. Perkins and Michael D. Ernst. Efficient incremental algorithms for dynamic detection of likely invariants. In *Proceedings of the ACM SIGSOFT 12th Symposium on the Foundations of Software Engineering (FSE 2004)*, Newport Beach, CA, USA, November 2-4, 2004.
4. Impact Analysis	• J. Law and G. Rothermel. Whole program path-based dynamic impact analysis. In *Proceedings of the 25th International Conference on Software Engineering*, 2003. • Xiaoxia Ren, Fenil Shah, Frank Tip, Barbara Ryder, and Ophelia Chesley. Chianti: A tool for change impact analysis of Java program. In *Object-Oriented Programming Systems, Languages, and Applications (OOPSLA 2004)*, Vancouver, BC, Canada, October 26-28, 2004.
9. Applied Program Comprehension	• Research: What it program comprehension? • Michele Lanza, Stéphane Ducasse: A Categorization of Classes based on the Visualization of their Internal Structure: The Class Blueprint. OOPSLA 2001: 300-311 • Michele Lanza, Stéphane Ducasse: Polymetric Views - A Lightweight Visual Approach to Reverse Engineering. IEEE Trans. Software Eng. 29(9): 782-795 (2003)
12. Visualization	• Thomas Ball, Stephen G. Eick: Software Visualization in the Large. IEEE Computer 29(4): 33-43 (1996) • Stephen G. Eick, Todd L. Graves, Alan F. Karr, Audris Mockus, Paul Schuster: Visualizing Software Changes. IEEE Trans. Software Eng. 28(4): 396-412 (2002) Plus one paper out of two (your choice): • Michele Lanza. The evolution matrix: recovering software evolution using software visualization techniques. *Proceedings of the 4th international workshop on Principles of software evolution,* 2001 • Evolution Spectrographs: Visualizing Punctuated Change in Software Evolution - (PDF), Jingwei Wu, Claus W. Spitzer, Ahmed E. Hassan and Richard C. Holt, Proceedings of IWPSE 2004: International Workshop on Principles of Software Evolution, Kyoto, Japan, September 6-7
8. Aspect Oriented Programming	• *Communications of the ACM*, 44(10), 2001. T. Elrad, R.E. Filman, and A. Bader. Introduction to AOP. T. Elrad (moderator). Discussing aspects of AOP. H. Ossher and P. Tarr. Using multidimensional separation of concerns to (re)shape evolving software. G. Kiczales et al. Getting started with AspectJ. • S. Breu, J. Krinke: *Aspect Mining Using Event Traces*. Proc. *Automated Software Engineering (ASE 2004)*, Linz, Austria, pp. 310-315, September 2004.