Andrea De Lucia

List of Publications by Year
in descending order

197 papers	8,768 citations	108046 37 h-index	107981 68 g-index
201 all docs	201 docs citations	201 times ranked	3141 citing authors

#	Article	IF	Citations
1	The Effect of Feature Characteristics on the Performance of Feature Location Techniques. IEEE Transactions on Software Engineering, 2022, 48, 2066-2085.	4.3	7
2	On the adequacy of static analysis warnings with respect to code smell prediction. Empirical Software Engineering, 2022, 27, 64.	3.0	6
3	Just-in-time software vulnerability detection: Are we there yet?. Journal of Systems and Software, 2022, 188, 111283.	3.3	16
4	Software testing and Android applications: a large-scale empirical study. Empirical Software Engineering, 2022, 27, 1.	3.0	9
5	A Systematic Literature Review on Bad Smellsâ€"5 W's: Which, When, What, Who, Where. IEEE Transactions on Software Engineering, 2021, 47, 17-66.	4.3	41
6	The Relation of Test-Related Factors to Software Quality: A Case Study on Apache Systems. Empirical Software Engineering, 2021, 26, 1.	3.0	7
7	Comparing within- and cross-project machine learning algorithms for code smell detection. , 2021, , .		11
8	A Test Case Prioritization Genetic Algorithm Guided by the Hypervolume Indicator. IEEE Transactions on Software Engineering, 2020, 46, 674-696.	4.3	33
9	Improving change prediction models with code smell-related information. Empirical Software Engineering, 2020, 25, 49-95.	3.0	34
10	Third-party libraries in mobile apps. Empirical Software Engineering, 2020, 25, 2341-2377.	3.0	16
11	A large empirical assessment of the role of data balancing in machine-learning-based code smell detection. Journal of Systems and Software, 2020, 169, 110693.	3.3	46
12	Developer-Driven Code Smell Prioritization. , 2020, , .		33
13	Testing of Mobile Applications in the Wild. , 2020, , .		17
14	Just-In-Time Test Smell Detection and Refactoring. , 2020, , .		23
15	Splicing Community Patterns and Smells. , 2020, , .		16
16	Refactoring Android-specific Energy Smells. , 2020, , .		8
17	cASpER. , 2020, , .		4
18	A preliminary study on the adequacy of static analysis warnings with respect to code smell prediction. , 2020, , .		6

#	Article	IF	Citations
19	Scented since the beginning: On the diffuseness of test smells in automatically generated test code. Journal of Systems and Software, 2019, 156, 312-327.	3.3	30
20	On the role of data balancing for machine learning-based code smell detection. , 2019, , .		28
21	Comparing Heuristic and Machine Learning Approaches for Metric-Based Code Smell Detection. , 2019, , .		57
22	On the impact of code smells on the energy consumption of mobile applications. Information and Software Technology, 2019, 105, 43-55.	3.0	70
23	Toward a Smell-Aware Bug Prediction Model. IEEE Transactions on Software Engineering, 2019, 45, 194-218.	4.3	63
24	A large-scale empirical study on the lifecycle of code smell co-occurrences. Information and Software Technology, 2018, 99, 1-10.	3.0	64
25	Detecting code smells using machine learning techniques: Are we there yet?. , 2018, , .		138
26	The Scent of a Smell: An Extensive Comparison Between Textual and Structural Smells. IEEE Transactions on Software Engineering, 2018, 44, 977-1000.	4.3	47
27	A Developer Centered Bug Prediction Model. IEEE Transactions on Software Engineering, 2018, 44, 5-24.	4.3	81
28	On the diffuseness and the impact on maintainability of code smells: a large scale empirical investigation. Empirical Software Engineering, 2018, 23, 1188-1221.	3.0	183
29	Crowdsourcing user reviews to support the evolution of mobile apps. Journal of Systems and Software, 2018, 137, 143-162.	3.3	65
30	Impact of Design Pattern Implementation Variants on the Retrieval Effectiveness of a Recovery Tool: An Exploratory Study. , 2018, , .		0
31	OCELOT: a search-based test-data generation tool for C. , 2018, , .		4
32	Dealing with Design Pattern Variants in Reverse Engineering: An Exploratory Study. , 2018, , .		0
33	Automatic Test Smell Detection Using Information Retrieval Techniques. , 2018, , .		43
34	Do developers update third-party libraries in mobile apps?. , 2018, , .		30
35	Enhancing change prediction models using developer-related factors. Journal of Systems and Software, 2018, 143, 14-28.	3.3	49
36	The role of meta-learners in the adaptive selection of classifiers. , 2018, , .		0

#	Article	IF	Citations
37	The scent of a smell. , 2018, , .		7
38	An empirical study on developerâ€related factors characterizing fixâ€inducing commits. Journal of Software: Evolution and Process, 2017, 29, e1797.	1.2	20
39	When and Why Your Code Starts to Smell Bad (and Whether the Smells Go Away). IEEE Transactions on Software Engineering, 2017, 43, 1063-1088.	4.3	156
40	Software-based energy profiling of Android apps: Simple, efficient and reliable?. , 2017, , .		56
41	Predicting Query Quality for Applications of Text Retrieval to Software Engineering Tasks. ACM Transactions on Software Engineering and Methodology, 2017, 26, 1-45.	4.8	27
42	Investigating code smell co-occurrences using association rule learning: A replicated study. , 2017, , .		20
43	Lightweight detection of Android-specific code smells: The aDoctor project. , 2017, , .		59
44	PETrA: A Software-Based Tool for Estimating the Energy Profile of Android Applications. , 2017, , .		27
45	Recommending and Localizing Change Requests for Mobile Apps Based on User Reviews. , 2017, , .		105
46	An Exploratory Study on the Relationship between Changes and Refactoring. , 2017, , .		57
47	Developer-Related Factors in Change Prediction: An Empirical Assessment. , 2017, , .		15
48	Detecting the Behavior of Design Patterns through Model Checking and Dynamic Analysis. ACM Transactions on Software Engineering and Methodology, 2017, 26, 1-41.	4.8	20
49	Smells Like Teen Spirit: Improving Bug Prediction Performance Using the Intensity of Code Smells. , 2016, , .		29
50	Automatic test case generation: what if test code quality matters?. , 2016, , .		41
51	An empirical investigation into the nature of test smells. , 2016, , .		98
52	A textual-based technique for Smell Detection. , 2016, , .		74
53	Search-Based Testing of Procedural Programs: Iterative Single-Target or Multi-target Approach?. Lecture Notes in Computer Science, 2016, , 64-79.	1.0	17
54	Parameterizing and Assembling IR-Based Solutions for SE Tasks Using Genetic Algorithms. , 2016, , .		24

#	Article	IF	Citations
55	On the diffusion of test smells in automatically generated test code. , 2016, , .		47
56	On the role of developer's scattered changes in bug prediction. , 2015, , .		8
57	Defect prediction as a multiobjective optimization problem. Software Testing Verification and Reliability, 2015, 25, 426-459.	1.7	59
58	Mining Version Histories for Detecting Code Smells. IEEE Transactions on Software Engineering, 2015, 41, 462-489.	4.3	192
59	An experimental investigation on the innate relationship between quality and refactoring. Journal of Systems and Software, 2015, 107, 1-14.	3.3	165
60	Adaptive User Feedback for IR-Based Traceability Recovery. , 2015, , .		13
61	User reviews matter! Tracking crowdsourced reviews to support evolution of successful apps. , 2015, , .		118
62	Landfill: An Open Dataset of Code Smells with Public Evaluation. , 2015, , .		35
63	ePadEvo: A tool for the detection of behavioral design patterns. , 2015, , .		6
64	Extract Package Refactoring in ARIES. , 2015, , .		3
65	Towards automating dynamic analysis for behavioral design pattern detection. , 2015, , .		3
66	Improving Multi-Objective Test Case Selection by Injecting Diversity in Genetic Algorithms. IEEE Transactions on Software Engineering, 2015, 41, 358-383.	4.3	100
67	A fine-grained analysis of the support provided by UML class diagrams and ER diagrams during data model maintenance. Software and Systems Modeling, 2015, 14, 287-306.	2.2	4
68	Hypervolume-Based Search for Test Case Prioritization. Lecture Notes in Computer Science, 2015, , 157-172.	1.0	9
69	When and Why Your Code Starts to Smell Bad. , 2015, , .		109
70	Are test smells really harmful? An empirical study. Empirical Software Engineering, 2015, 20, 1052-1094.	3.0	135
71	Anti-Pattern Detection. Advances in Computers, 2014, 95, 201-238.	1.2	25
72	Improving software modularization via automated analysis of latent topics and dependencies. ACM Transactions on Software Engineering and Methodology, 2014, 23, 1-33.	4.8	101

#	Article	IF	Citations
73	Do They Really Smell Bad? A Study on Developers' Perception of Bad Code Smells. , 2014, , .		151
74	Labeling source code with information retrieval methods: an empirical study. Empirical Software Engineering, 2014, 19, 1383-1420.	3.0	32
75	Automating extract class refactoring: an improved method and its evaluation. Empirical Software Engineering, 2014, 19, 1617-1664.	3.0	73
76	Recovering test-to-code traceability using slicing and textual analysis. Journal of Systems and Software, 2014, 88, 147-168.	3.3	47
77	In medio stat virtus: Extract class refactoring through nash equilibria. , 2014, , .		4
78	Cross-project defect prediction models: L'Union fait la force. , 2014, , .		119
79	Methodbook: Recommending Move Method Refactorings via Relational Topic Models. IEEE Transactions on Software Engineering, 2014, 40, 671-694.	4.3	115
80	Enhancing software artefact traceability recovery processes with link count information. Information and Software Technology, 2014, 56, 163-182.	3.0	9
81	Recommending Refactoring Operations in Large Software Systems. , 2014, , 387-419.		30
82	Applying a smoothing filter to improve IR-based traceability recovery processes: An empirical investigation. Information and Software Technology, 2013, 55, 741-754.	3.0	20
83	How to effectively use topic models for software engineering tasks? An approach based on Genetic Algorithms. , 2013, , .		171
84	Automatic query reformulations for text retrieval in software engineering. , 2013, , .		127
85	Query quality prediction and reformulation for source code search: The Refoqus tool. , 2013, , .		11
86	Multi-objective Cross-Project Defect Prediction. , 2013, , .		126
87	The role of artefact corpus in LSI-based traceability recovery. , 2013, , .		10
88	Configuring topic models for software engineering tasks in TraceLab. , 2013, , .		11
89	When and How Using Structural Information to Improve IR-Based Traceability Recovery. , 2013, , .		50
90	Using code ownership to improve IR-based Traceability Link Recovery. , 2013, , .		23

#	Article	IF	Citations
91	Using structural and semantic measures to improve software modularization. Empirical Software Engineering, 2013, 18, 901-932.	3.0	56
92	Improving IRâ€based traceability recovery via nounâ€based indexing of software artifacts. Journal of Software: Evolution and Process, 2013, 25, 743-762.	1.2	54
93	Orthogonal exploration of the search space in evolutionary test case generation. , 2013, , .		10
94	Does software error/defect identification matter in the Italian industry?. IET Software, 2013, 7, 76-84.	1.5	2
95	Evaluating testâ€toâ€code traceability recovery methods through controlled experiments. Journal of Software: Evolution and Process, 2013, 25, 1167-1191.	1.2	18
96	Detecting bad smells in source code using change history information. , 2013, , .		156
97	An empirical study on the developers' perception of software coupling. , 2013, , .		72
98	Generating applications directly on the mobile device. , 2012, , .		3
99	Automatic query performance assessment during the retrieval of software artifacts. , 2012, , .		33
100	Estimating the evolution direction of populations to improve genetic algorithms. , 2012, , .		3
101	Evaluating the specificity of text retrieval queries to support software engineering tasks. , 2012, , .		18
102	TraceME: Traceability Management in Eclipse. , 2012, , .		19
103	Using IR methods for labeling source code artifacts: Is it worthwhile?. , 2012, , .		58
104	Teaching software engineering and software project management: An integrated and practical approach. , 2012, , .		23
105	When Does a Refactoring Induce Bugs? An Empirical Study. , 2012, , .		106
106	On the role of diversity measures for multi-objective test case selection. , 2012, , .		16
107	An empirical analysis of the distribution of unit test smells and their impact on software maintenance. , 2012, , .		104
108	Putting the Developer in-the-Loop: An Interactive GA for Software Re-modularization. Lecture Notes in Computer Science, 2012, , 75-89.	1.0	44

#	Article	IF	Citations
109	Supporting extract class refactoring in Eclipse: The ARIES project. , 2012, , .		15
110	Information Retrieval Methods for Automated Traceability Recovery. , 2012, , 71-98.		57
111	SCOTCH: Slicing and Coupling Based Test to Code Trace Hunter. , 2011, , .		3
112	SCOTCH: Test-to-code traceability using slicing and conceptual coupling. , 2011, , .		28
113	On integrating orthogonal information retrieval methods to improve traceability recovery. , 2011, , .		87
114	Improving Source Code Lexicon via Traceability and Information Retrieval. IEEE Transactions on Software Engineering, 2011, 37, 205-227.	4.3	47
115	Migration of information systems in the Italian industry: A state of the practice survey. Information and Software Technology, 2011, 53, 71-86.	3.0	19
116	Identifying Extract Class refactoring opportunities using structural and semantic cohesion measures. Journal of Systems and Software, 2011, 84, 397-414.	3.3	95
117	Identifying method friendships to remove the feature envy bad smell (NIER track). , 2011, , .		30
118	CodeTopics. , 2011, , .		27
119	Improving IR-based Traceability Recovery Using Smoothing Filters. , 2011, , .		35
120	Introduction to the European Projects Track. , 2011, , .		0
121	Augmented Reality Mobile Applications: Challenges and Solutions. Recent Patents on Computer Science, 2011, 4, 80-90.	0.5	2
122	An experimental comparison of ER and UML class diagrams for data modelling. Empirical Software Engineering, 2010, 15, 455-492.	3.0	41
123	Fineâ€grained management of software artefacts: the ADAMS system. Software - Practice and Experience, 2010, 40, 1007-1034.	2.5	15
124	Recovering traceability links between unit tests and classes under test: An improved method. , 2010, , .		32
125	A two-step technique for extract class refactoring. , 2010, , .		36
126	An Eclipse plug-in for the detection of design pattern instances through static and dynamic analysis. , 2010, , .		18

#	Article	IF	Citations
127	Software Re-Modularization Based on Structural and Semantic Metrics. , 2010, , .		36
128	Playing with refactoring: Identifying extract class opportunities through game theory. , 2010, , .		45
129	On the Equivalence of Information Retrieval Methods for Automated Traceability Link Recovery. , 2010, , .		136
130	Improving Behavioral Design Pattern Detection through Model Checking. , 2010, , .		28
131	Requirements Engineering in Agile Software Development. Journal of Emerging Technologies in Web Intelligence, 2010, 2, .	0.6	70
132	Recovering design rationale from email repositories. , 2009, , .		0
133	On the role of the nouns in IR-based traceability recovery. , 2009, , .		43
134	Design pattern recovery through visual language parsing and source code analysis. Journal of Systems and Software, 2009, 82, 1177-1193.	3.3	75
135	Assessing IR-based traceability recovery tools through controlled experiments. Empirical Software Engineering, 2009, 14, 57-92.	3.0	71
136	Development and evaluation of a system enhancing Second Life to support synchronous roleâ€based collaborative learning. Software - Practice and Experience, 2009, 39, 1025-1054.	2.5	15
137	Evaluating legacy system migration technologies through empirical studies. Information and Software Technology, 2009, 51, 433-447.	3.0	36
138	Development and evaluation of a virtual campus on Second Life: The case of SecondDMI. Computers and Education, 2009, 52, 220-233.	5.1	292
139	Behavioral Pattern Identification through Visual Language Parsing and Code Instrumentation. , 2009, , .		27
140	METAMORPHOS: MEthods and Tools for migrAting software systeMs towards web and service Oriented aRchitectures: exPerimental evaluation, usability, and tecHnOlogy tranSfer. , 2009, , .		2
141	DB-MELIS: An Eclipse Plug-in for Data Migration. , 2009, , .		0
142	Towards automatic clustering of similar pages in web applications. , 2009, , .		1
143	The role of the coverage analysis during IR-based traceability recovery: A controlled experiment. , 2009, , .		5
144	Traceability Recovery Using Numerical Analysis. , 2009, , .		18

#	Article	IF	Citations
145	Concurrent Fine-Grained Versioning of UML Models. , 2009, , .		6
146	Developing legacy system migration methods and tools for technology transfer. Software - Practice and Experience, 2008, 38, 1333-1364.	2.5	35
147	Migrating legacy video lectures to multimedia learning objects. Software - Practice and Experience, 2008, 38, 1499-1530.	2.5	3
148	Assessing the usability of a visual tool for the definition of e-learning processes. Journal of Visual Languages and Computing, 2008, 19, 721-737.	1.8	7
149	Traceability management for impact analysis. , 2008, , .		37
150	SLMeeting. , 2008, , .		28
151	IR-Based Traceability Recovery Processes: An Empirical Comparison of "One-Shot" and Incremental Processes. , 2008, , .		28
152	COMOVER: Concurrent model versioning. , 2008, , .		3
153	Software migration projects in Italian industry: Preliminary results from a state of the practice survey. , 2008, , .		8
154	Supporting Jigsaw-Based Collaborative Learning in Second Life. , 2008, , .		9
155	Adams re-trace. , 2008, , .		39
156	An approach and an eclipse based environment for data migration. , 2008, , .		2
157	Using structural and semantic metrics to improve class cohesion. , 2008, , .		18
158	Assessing the Support of ER and UML Class Diagrams during Maintenance Activities on Data Models. Software Maintenance and Reengineering (CSMR), Proceedings of the European Conference on, 2008, , .	0.0	6
159	Data Model Comprehension: An Empirical Comparison of ER and UML Class Diagrams. , 2008, , .		16
160	A Service Oriented Collaborative Distributed Learning Object Management System. Lecture Notes in Business Information Processing, 2008, , 341-354.	0.8	2
161	A Visual Framework for the Definition and Execution of Reverse Engineering Processes. Lecture Notes in Computer Science, 2008, , 235-246.	1.0	0
162	A Two Phase Approach to Design Pattern Recovery. , 2007, , .		11

#	Article	IF	Citations
163	Improving context awareness in subversion through fine-grained versioning of Java code. , 2007, , .		2
164	Assessing the Effectiveness of a Distributed Method for Code Inspection: A Controlled Experiment. , 2007, , .		7
165	Assessing Legacy System Migration Technologies through Controlled Experiments. Conference on Software Maintenance, Proceedings of the, 2007, , .	0.0	5
166	Clustering Algorithms and Latent Semantic Indexing to Identify Similar Pages in Web Applications. , 2007, , .		11
167	Empirical Studies in Software Maintenance and Evolution. , 2007, , .		1
168	Recovering traceability links in software artifact management systems using information retrieval methods. ACM Transactions on Software Engineering and Methodology, 2007, 16, 13.	4.8	269
169	Identifying similar pages in Web applications using a competitive clustering algorithm. Journal of Software: Evolution and Process, 2007, 19, 281-296.	1.1	14
170	Enhancing collaborative synchronous UML modelling with fine-grained versioning of software artefacts. Journal of Visual Languages and Computing, 2007, 18, 492-503.	1.8	27
171	A Strategy and an Eclipse Based Environment for the Migration of Legacy Systems to Multi-tier Web-based Architectures. Conference on Software Maintenance, Proceedings of the, 2006, , .	0.0	11
172	Working Session: Information Retrieval Based Approaches in Software Evolution. , 2006, , .		4
173	COCONUT: COde COmprehension Nurturant Using Traceability. , 2006, , .		5
174	Using a Competitive Clustering Algorithm to Comprehend Web Applications. , 2006, , .		5
175	Supporting Distributed Software Development with fine-grained Artefact Management. , 2006, , .		16
176	VLMigrator. , 2006, , .		0
177	Incremental Approach and User Feedbacks: a Silver Bullet for Traceability Recovery. Conference on Software Maintenance, Proceedings of the, 2006, , .	0.0	58
178	Assessing effort estimation models for corrective maintenance through empirical studies. Information and Software Technology, 2005, 47, 3-15.	3.0	57
179	Managing coordination and cooperation in distributed software processes: the GENESIS environment. Software Process Improvement and Practice, 2004, 9, 239-263.	1.1	20
180	Assessing the maintenance processes of a software organization: an empirical analysis of a large industrial project. Journal of Systems and Software, 2003, 65, 87-103.	3.3	20

#	Article	IF	Citations
181	Migrating Legacy System to the Web. , 2003, , 151-181.		3
182	Effort estimation for corrective software maintenance. , 2002, , .		18
183	Recovering traceability links between code and documentation. IEEE Transactions on Software Engineering, 2002, 28, 970-983.	4.3	749
184	Business process reengineering and workflow automation: a technology transfer experience. Journal of Systems and Software, 2002, 63, 29-44.	3.3	31
185	Automating the management of software maintenance workflows in a large software enterprise: a case study. Journal of Software: Evolution and Process, 2002, 14, 229-255.	1.1	10
186	Decomposing legacy systems into objects: an eclectic approach. Information and Software Technology, 2001, 43, 401-412.	3.0	27
187	Maintaining traceability links during object-oriented software evolution. Software - Practice and Experience, 2001, 31, 331-355.	2.5	34
188	Decomposing legacy programs: a first step towards migrating to clientâ€“server platforms. Journal of Systems and Software, 2000, 54, 99-110.	3.3	48
189	A DESIGN RATIONALE BASED ENVIRONMENT FOR COOPERATIVE MAINTENANCE. International Journal of Software Engineering and Knowledge Engineering, 2000, 10, 627-645.	0.6	11
190	AN INCREMENTAL OBJECT-ORIENTED MIGRATION STRATEGY FOR RPG LEGACY SYSTEMS. International Journal of Software Engineering and Knowledge Engineering, 1999, 09, 5-25.	0.6	12
191	Identifying objects in legacy systems using design metrics. Journal of Systems and Software, 1999, 44, 199-211.	3.3	41
192	A System for Generating Reverse Engineering Tools: A Case Study of Software Modularisation. Automated Software Engineering, 1999, 6, 233-263.	2.2	2
193	Conditioned program slicing. Information and Software Technology, 1998, 40, 595-607.	3.0	151
194	An integrated environment for reuse reengineering C code. Journal of Systems and Software, 1998, 42, 153-164.	3.3	19
195	An extensible system for source code analysis. IEEE Transactions on Software Engineering, 1998, 24, 721-740.	4.3	20
196	A parsing methodology for the implementation of visual systems. IEEE Transactions on Software Engineering, 1997, 23, 777-799.	4.3	56
197	A Specification Driven Slicing Process for Identifying Reusable Functions. Journal of Software: Evolution and Process, 1996, 8, 145-178.	0.5	46