Massimiliano Di Penta

List of Publications by Year
in descending order

225 papers	10,265 citations	109321 35 h-index	114465 63 g-index
226 all docs	226 docs citations	226 times ranked	3628 citing authors

#	Article	IF	Citations
1	Recommending API Function Calls and Code Snippets to Support Software Development. IEEE Transactions on Software Engineering, 2022, 48, 2417-2438.	5.6	12
2	Enabling Mutant Generation for Open- and Closed-Source Android Apps. IEEE Transactions on Software Engineering, 2022, 48, 186-208.	5.6	6
3	Why Do Developers Reject Refactorings in Open-Source Projects?. ACM Transactions on Software Engineering and Methodology, 2022, 31, 1-23.	6.0	1
4	Using code reviews to automatically configure static analysis tools. Empirical Software Engineering, 2022, 27, 1.	3.9	9
5	An empirical characterization of software bugs in open-source Cyberâ€“Physical Systems. Journal of Systems and Software, 2022, 192, 111425.	4.5	11
6	Adversarial Machine Learning: On the Resilience of Third-party Library Recommender Systems. , 2021, , .		3
7	An empirical study on the co-occurrence between refactoring actions and Self-Admitted Technical Debt removal. Journal of Systems and Software, 2021, 178, 110976.	4.5	20
8	Self-admitted technical debt practices: a comparison between industry and open-source. Empirical Software Engineering, 2021, 26, 1.	3.9	19
9	How Empirical Research Supports Tool Development. , 2021, , .		0
10	An Empirical Study on the Usage of Transformer Models for Code Completion. IEEE Transactions on Software Engineering, 2021, , 1-1.	5.6	23
11	CI/CD Pipelines Evolution and Restructuring: A Qualitative and Quantitative Study. , 2021, , .		15
12	An NLP-based Tool for Software Artifacts Analysis. , 2021, , .		9
13	Adversarial Attacks to API Recommender Systems: Time to Wake Up and Smell the Coffee?. , 2021, , .		5
14	What kind of questions do developers ask on Stack Overflow? A comparison of automated approaches to classify posts into question categories. Empirical Software Engineering, 2020, 25, 2258-2301.	3.9	34
15	An empirical characterization of bad practices in continuous integration. Empirical Software Engineering, 2020, 25, 1095-1135.	3.9	42
16	CrossRec: Supporting software developers by recommending third-party libraries. Journal of Systems and Software, 2020, 161, 110460.	4.5	37
17	Automatically Learning Patterns for Self-Admitted Technical Debt Removal. , 2020, , .		21
18	Characterizing the evolution of statically-detectable performance issues of Android apps. Empirical Software Engineering, 2020, 25, 2748-2808.	3.9	10

#	Article	IF	Citations
19	Demystifying the adoption of behavior-driven development in open source projects. Information and Software Technology, 2020, 123, 106311.	4.4	10
20	Guest editorial: special section on software analysis, evolution, and reengineering. Empirical Software Engineering, 2020, 25, 1379-1381.	3.9	0
21	On the relationship between refactoring actions and bugs: a differentiated replication. , 2020, , .		21
22	Why Developers Refactor Source Code. ACM Transactions on Software Engineering and Methodology, 2020, 29, 1-30.	6.0	29
23	Understanding and Improving Continuous Integration and Delivery Practice using Data from the Wild. , 2020, , .		0
24	Configuration smells in continuous delivery pipelines: a linter and a six-month study on GitLab. , 2020, , .		17
25	Exploiting Natural Language Structures in Software Informal Documentation. IEEE Transactions on Software Engineering, 2019, , 1-1.	5.6	14
26	A Survey on Online Learning Preferences for Computer Science and Programming. , 2019, , .		10
27	Automated Reporting of Anti-Patterns and Decay in Continuous Integration. , 2019, , .		39
28	FOCUS: A Recommender System for Mining API Function Calls and Usage Patterns. , 2019, , .		69
29	An Empirical Study on Learning Bug-Fixing Patches in the Wild via Neural Machine Translation. ACM Transactions on Software Engineering and Methodology, 2019, 28, 1-29.	6.0	151
30	Self-Admitted Technical Debt Removal and Refactoring Actions: Co-Occurrence or More?. , 2019, , .		20
31	Listening to the Crowd for the Release Planning of Mobile Apps. IEEE Transactions on Software Engineering, 2019, 45, 68-86.	5.6	48
32	Automatic Identification and Classification of Software Development Video Tutorial Fragments. IEEE Transactions on Software Engineering, 2019, 45, 464-488.	5.6	32
33	A large-scale empirical study on the lifecycle of code smell co-occurrences. Information and Software Technology, 2018, 99, 1-10.	4.4	64
34	The relation between developersâ€™ communication and fix-Inducing changes: An empirical study. Journal of Systems and Software, 2018, 140, 111-125.	4.5	9
35	On the diffuseness and the impact on maintainability of code smells: a large scale empirical investigation. Empirical Software Engineering, 2018, 23, 1188-1221.	3.9	183
36	Crowdsourcing user reviews to support the evolution of mobile apps. Journal of Systems and Software, 2018, 137, 143-162.	4.5	65

#	Article	IF	Citations
37	On the diffuseness and the impact on maintainability of code smells. , 2018, , .		23
38	To distribute or not to distribute?. , 2018, , .		8
39	Estimating the number of remaining links in traceability recovery (journal-first abstract). , 2018, , .		1
40	Assessing Test Case Prioritization on Real Faults and Mutants. , 2018, , .		32
41	MDroid+. , 2018, , .		24
42	Was self-admitted technical debt removal a real removal?. , 2018, , .		38
43	Automatically classifying posts into question categories on stack overflow. , 2018, , .		31
44	License usage and changes: a large-scale study on gitHub. Empirical Software Engineering, 2017, 22, 1537-1577.	3.9	25
45	When and Why Your Code Starts to Smell Bad (and Whether the Smells Go Away). IEEE Transactions on Software Engineering, 2017, 43, 1063-1088.	5.6	156
46	Enabling mutation testing for Android apps. , 2017, , .		57
47	Supporting Software Developers with a Holistic Recommender System. , 2017, , .		34
48	Detecting missing information in bug descriptions. , 2017, , .		96
49	How Open Source Projects Use Static Code Analysis Tools in Continuous Integration Pipelines. , 2017, , .		82
50	Patterns of developers behaviour: A 1000-hour industrial study. Journal of Systems and Software, 2017, 132, 85-97.	4.5	19
51	Estimating the number of remaining links in traceability recovery. Empirical Software Engineering, 2017, 22, 996-1027.	3.9	26
52	ARENA: An Approach for the Automated Generation of Release Notes. IEEE Transactions on Software Engineering, 2017, 43, 106-127.	5.6	69
53	A Tale of CI Build Failures: An Open Source and a Financial Organization Perspective. , 2017, , .		37
54	Release planning of mobile apps based on user reviews. , 2016, , .		172

#	Article	IF	Citations
55	Continuous Delivery Practices in a Large Financial Organization. , 2016, , .		28
56	A Quantitative and Qualitative Investigation of Performance-Related Commits in Android Apps. , 2016, , .		21
57	Guest editorial: Special section on mining software repositories. Empirical Software Engineering, 2016, 21, 301-302.	3.9	0
58	Guest editorial: special section on software reverse engineering. Empirical Software Engineering, 2016, 21, 749-752.	3.9	0
59	An empirical investigation into the nature of test smells. , 2016, , .		98
60	Too long; didn't watch!. , 2016, , .		46
61	Parameterizing and Assembling IR-Based Solutions for SE Tasks Using Genetic Algorithms. , 2016, , .		24
62	DECA. , 2016, , .		20
63	Linguistic antipatterns: what they are and how developers perceive them. Empirical Software Engineering, 2016, 21, 104-158.	3.9	85
64	Prompter. Empirical Software Engineering, 2016, 21, 2190-2231.	3.9	31
65	Optimizing energy consumption of GUIs in Android apps: a multi-objective approach. , 2015, , .		59
66	Defect prediction as a multiobjective optimization problem. Software Testing Verification and Reliability, 2015, 25, 426-459.	2.0	59
67	Mining Version Histories for Detecting Code Smells. IEEE Transactions on Software Engineering, 2015, 41, 462-489.	5.6	192
68	The Impact of API Change- and Fault-Proneness on the User Ratings of Android Apps. IEEE Transactions on Software Engineering, 2015, 41, 384-407.	5.6	139
69	An experimental investigation on the innate relationship between quality and refactoring. Journal of Systems and Software, 2015, 107, 1-14.	4.5	165
70	License Usage and Changes: A Large-Scale Study of Java Projects on GitHub. , 2015, , .		21
71	User reviews matter! Tracking crowdsourced reviews to support evolution of successful apps. , 2015, , .		118
72	How Can I Use This Method?. , 2015, , .		40

#	Article	IF	Citations
73	Query-based configuration of text retrieval solutions for software engineering tasks. , 2015, , .		34
74	When and why developers adopt and change software licenses. , 2015, , .		26
75	Development Emails Content Analyzer: Intention Mining in Developer Discussions (T). , 2015, , .		64
76	Improving Multi-Objective Test Case Selection by Injecting Diversity in Genetic Algorithms. IEEE Transactions on Software Engineering, 2015, 41, 358-383.	5.6	100
77	Guest editorial: special section on software maintenance and evolution. Empirical Software Engineering, 2015, 20, 410-412.	3.9	0
78	Guest editorial: special section on mining software repositories. Empirical Software Engineering, 2015, 20, 291-293.	3.9	1
79	Would static analysis tools help developers with code reviews?. , 2015, , .		49
80	How the Apache community upgrades dependencies: an evolutionary study. Empirical Software Engineering, 2015, 20, 1275-1317.	3.9	93
81	Irish: A Hidden Markov Model to detect coded information islands in free text. Science of Computer Programming, 2015, 105, 26-43.	1.9	4
82	When and Why Your Code Starts to Smell Bad. , 2015, , .		109
83	A family of experiments to assess the effectiveness and efficiency of source code obfuscation techniques. Empirical Software Engineering, 2014, 19, 1040.	3.9	42
84	Identifying and locating interference issues in PHP applications: the case of WordPress. , 2014, , .		11
85	Do They Really Smell Bad? A Study on Developers' Perception of Bad Code Smells. , 2014, , .		151
86	How Developers' Collaborations Identified from Different Sources Tell Us about Code Changes. , 2014, , .		32
87	Prompter: A Self-Confident Recommender System. , 2014, , .		27
88	How the evolution of emerging collaborations relates to code changes: an empirical study. , 2014, , .		19
89	How do API changes trigger stack overflow discussions? a study on the Android SDK. , 2014, , .		118
90	CODES: mining source code descriptions from developers discussions. , 2014, , .		53

#	Article	IF	Citations
91	Automatic generation of release notes. , 2014, , .		73
92	Recommending refactorings based on team co-maintenance patterns. , 2014, , .		12
93	On the Impact of Refactoring Operations on Code Quality Metrics. , 2014, , .		49
94	Mining energy-greedy API usage patterns in Android apps: an empirical study. , 2014, , .		160
95	An experimental investigation on the effects of context on source code identifiers splitting and expansion. Empirical Software Engineering, 2014, 19, 1706-1753.	3.9	10
96	Labeling source code with information retrieval methods: an empirical study. Empirical Software Engineering, 2014, 19, 1383-1420.	3.9	32
97	How changes affect software entropy: an empirical study. Empirical Software Engineering, 2014, 19, 1-38.	3.9	50
98	REPENT: Analyzing the Nature of Identifier Renamings. IEEE Transactions on Software Engineering, 2014, 40, 502-532.	5.6	71
99	The market for open source: An intelligent virtual open source marketplace. , 2014, , .		12
100	Mining StackOverflow to turn the IDE into a self-confident programming prompter. , 2014, , .		181
101	SCAN: an approach to label and relate execution trace segments. Journal of Software: Evolution and Process, 2014, 26, 962-995.	1.6	7
102	Applying a smoothing filter to improve IR-based traceability recovery processes: An empirical investigation. Information and Software Technology, 2013, 55, 741-754.	4.4	20
103	A Hidden Markov Model to detect coded information islands in free text. , 2013, , .		9
104	Multi-objective Cross-Project Defect Prediction. , 2013, , .		126
105	A New Family of Software Anti-patterns: Linguistic Anti-patterns. , 2013, , .		56
106	A Study on the Relation between Antipatterns and the Cost of Class Unit Testing. , 2013, , .		7
107	YODA: Young and newcOmer Developer Assistant. , 2013, , .		5
108	The Evolution of Project Inter-dependencies in a Software Ecosystem: The Case of Apache. , 2013, , .		65

#	Article	IF	Citations
109	API change and fault proneness: a threat to the success of Android apps. , 2013, , .		180
110	TIDIER: an identifier splitting approach using speech recognition techniques. Journal of Software: Evolution and Process, 2013, 25, 575-599.	1.6	39
111	Detecting bad smells in source code using change history information. , 2013, , .		156
112	An Empirical Investigation on Documentation Usage Patterns in Maintenance Tasks. , 2013, , .		8
113	LHDiff: A Language-Independent Hybrid Approach for Tracking Source Code Lines. , 2013, , .		31
114	LHDiff: Tracking Source Code Lines to Support Software Maintenance Activities. , 2013, , .		1
115	Message from the PROMISE 2013 Chairs. , 2013, , .		0
116	Who is going to mentor newcomers in open source projects?. , 2012, , .		105
117	Estimating the evolution direction of populations to improve genetic algorithms. , 2012, , .		3
118	SCAN: An Approach to Label and Relate Execution Trace Segments. , 2012, , .		5
119	TRIS: A Fast and Accurate Identifiers Splitting and Expansion Algorithm. , 2012, , .		14
120	Mining source code descriptions from developer communications. , 2012, , .		65
121	Five days of empirical software engineering: The PASED experience. , 2012, , .		0
122	Managing and assessing the risk of component upgrades. , 2012, , .		4
123	When Does a Refactoring Induce Bugs? An Empirical Study. , 2012, , .		106
124	Empirical Studies in Reverse Engineering and Maintenance: Employing Developers to Evaluate Your Approach and Tool. , 2012, , .		0
125	On the role of diversity measures for multi-objective test case selection. , 2012, , .		16
126	Mining developers' communication to assess software quality: Promises, challenges, perils. , 2012, , .		3

#	Article	IF	Citations
127	Do Developers Introduce Bugs When They Do Not Communicate? The Case of Eclipse and Mozilla. , 2012, , .		19
128	Putting the Developer in-the-Loop: An Interactive GA for Software Re-modularization. Lecture Notes in Computer Science, 2012, , 75-89.	1.3	44
129	A Method for Open Source License Compliance of Java Applications. IEEE Software, 2012, 29, 58-63.	1.8	31
130	An exploratory study of the impact of antipatterns on class change- and fault-proneness. Empirical Software Engineering, 2012, 17, 243-275.	3.9	277
131	How Long Does a Bug Survive? An Empirical Study. , 2011, , .		27
132	An Approach for Search Based Testing of Null Pointer Exceptions. , 2011, , .		18
133	Assessing, Comparing, and Combining State Machine-Based Testing and Structural Testing: A Series of Experiments. IEEE Transactions on Software Engineering, 2011, 37, 161-187.	5.6	61
134	Improving Source Code Lexicon via Traceability and Information Retrieval. IEEE Transactions on Software Engineering, 2011, 37, 205-227.	5.6	47
135	Introduction to the special issue on search based software engineering. Empirical Software Engineering, 2011, 16, 1-4.	3.9	2
136	The use of searchâ€based optimization techniques to schedule and staff software projects: an approach and an empirical study. Software - Practice and Experience, 2011, 41, 495-519.	3.6	50
137	Migration of information systems in the Italian industry: A state of the practice survey. Information and Software Technology, 2011, 53, 71-86.	4.4	19
138	What topics do Firefox and Chrome contributors discuss?. , 2011, , .		4
139	Workshop on emerging trends in software metrics (WETSoM 2011). , 2011, , .		0
140	An exploratory study of identifier renamings. , 2011, , .		16
141	CodeTopics. , 2011, , .		27
142	Nothing else matters. , 2011, , .		2
143	MoMS: Multi-objective miniaturization of software. , 2011, , .		10
144	Improving IR-based Traceability Recovery Using Smoothing Filters. , 2011, , .		35

#	Article	IF	Citations
145	Sixth international workshop on traceability in emerging forms of software engineering (TEFSE 2011). , 2011, , .		2
146	Social interactions around cross-system bug fixings. , 2011, , .		31
147	Achievements and challenges in software reverse engineering. Communications of the ACM, 2011, 54, 142-151.	4.5	98
148	Cooperative Co-evolutionary Optimization of Software Project Staff Assignments and Job Scheduling. Lecture Notes in Computer Science, 2011, , 127-141.	1.3	29
149	A Fast Algorithm to Locate Concepts in Execution Traces. Lecture Notes in Computer Science, 2011, , 252-266.	1.3	9
150	An empirical study on the maintenance of source code clones. Empirical Software Engineering, 2010, 15, 1-34.	3.9	144
151	A heuristic-based approach for detecting SQL-injection vulnerabilities in web applications. , 2010, , .		36
152	Understanding and Auditing the Licensing of Open Source Software Distributions. , 2010, , .		49
153	Lawful software engineering. , 2010, , .		6
154	An empirical comparison of methods to support QoS-aware service selection. , 2010, , .		61
155	An eclectic approach for change impact analysis. , 2010, , .		29
156	An exploratory study of the evolution of software licensing. , 2010, , .		57
157	An Exploratory Study of Factors Influencing Change Entropy. , 2010, , .		13
158	Using multivariate time series and association rules to detect logical change coupling: An empirical study. , 2010, , .		41
159	A Heuristic-Based Approach to Identify Concepts in Execution Traces. , 2010, , .		28
160	Identifying licensing of jar archives using a code-search approach. , 2010, , .		24
161	Ldiff: An enhanced line differencing tool. , 2009, , .		38
162	Introduction to the WCRE 2007 special issue. Software Quality Journal, 2009, 17, 305-307.	2.2	0

#	Article	IF	Citations
163	Introduction to the special issue on reverse engineering (WCRE 2008). Journal of Software: Evolution and Process, 2009, 22, n/a-n/a.	1.1	0
164	Using acceptance tests as a support for clarifying requirements: A series of experiments. Information and Software Technology, 2009, 51, 270-283.	4.4	53
165	The life and death of statically detected vulnerabilities: An empirical study. Information and Software Technology, 2009, 51, 1469-1484.	4.4	31
166	Tracking Your Changes: A Language-Independent Approach. IEEE Software, 2009, 26, 50-57.	1.8	38
167	Service-Oriented Architectures Testing: A Survey. Lecture Notes in Computer Science, 2009, , 78-105.	1.3	101
168	Who are Source Code Contributors and How do they Change?. , 2009, , .		10
169	An Exploratory Study of the Impact of Code Smells on Software Change-proneness. , 2009, , .		194
170	METAMORPHOS: MEthods and Tools for migrAting software systeMs towards web and service Oriented aRchitectures: exPerimental evaluation, usability, and tecHnOlogy tranSfer. , 2009, , .		2
171	Welcome from the Workshop Chair. , 2009, , .		0
172	Code siblings: Technical and legal implications of copying code between applications. , 2009, , .		69
173	The effectiveness of source code obfuscation: An experimental assessment. , 2009, , .		47
174	Dynamic composition of web applications in human-centered processes. , 2009, , .		6
175	3rd International Workshop on Designing Empirical Studies: Assessing the Effectiveness of Agile Methods (IWDES 2009). Lecture Notes in Business Information Processing, 2009, , 234-235.	1.0	0
176	Guest editorsâ€™ introduction: special issue from the 13th working conference on reverse engineering (WCRE 2006). Empirical Software Engineering, 2008, 13, 597-600.	3.9	0
177	Search-based inference of dialect grammars. Soft Computing, 2008, 12, 51-66.	3.6	6
178	Special Issue on Searchâ€Based Software Maintenance. Journal of Software: Evolution and Process, 2008, 20, 317-319.	1.1	0
179	A framework for QoS-aware binding and re-binding of composite web services. Journal of Systems and Software, 2008, 81, 1754-1769.	4.5	255
180	Reuse or rewrite: Combining textual, static, and dynamic analyses to assess the cost of keeping a system up-to-date. , 2008, , .		8

#	Article	IF	Citations
181	Trend Analysis and Issue Prediction in Large-Scale Open Source Systems. , 2008, , .		31
182	An empirical study of the relationships between design pattern roles and class change proneness. , 2008, , .		33
183	Software migration projects in Italian industry: Preliminary results from a state of the practice survey. , 2008, , .		8
184	Is it a bug or an enhancement?. , 2008, , .		259
185	Are fit tables really talking?. , 2008, , .		26
186	Frontiers of reverse engineering: A conceptual model. , 2008, , .		8
187	The Evolution and Decay of Statically Detected Source Code Vulnerabilities. , 2008, , .		7
188	Guidelines on the use of Fit tables in software maintenance tasks: Lessons learned from 8 experiments. , 2008, , .		8
189	Towards experimental evaluation of code obfuscation techniques. , 2008, , .		16
190	Smart Formatter: Learning Coding Style from Existing Source Code. , 2007, , .		13
191	An approach for mining services in database oriented applications. , 2007, , .		14
192	How Clones are Maintained: An Empirical Study. , 2007, , .		141
193	An empirical study on the evolution of design patterns. , 2007, , .		61
194	A search-based approach for dynamically re-packaging downloadable applications. Proceedings of CASCON, 2007, , .	0.0	6
195	Search-based testing of service level agreements. , 2007, , .		44
196	IWPSE 2007. , 2007, , .		0
197	Discovery of SOA patterns via model checking. , 2007, , .		4
198	Designing your Next Empirical Study on Program Comprehension. , 2007, , .		25

#	Article	IF	Citations
199	Identifying Changed Source Code Lines from Version Repositories. , 2007, , .		68
200	Relating the Evolution of Design Patterns and Crosscutting Concerns. , 2007, , .		6
201	The Role of Experience and Ability in Comprehension Tasks Supported by UML Stereotypes. , 2007, , .		46
202	"Talking tests": a Preliminary Experimental Study on Fit User Acceptance Tests. First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007), 2007, , .	0.0	2
203	"Talking tests": a Preliminary Experimental Study on Fit User Acceptance Tests. , 2007, , .		8
204	The Effect of Communication Overhead on Software Maintenance Project Staffing: a Search-Based Approach. , 2007, , .		22
205	New Frontiers of Reverse Engineering. , 2007, , .		93
206	Special issue on source code analysis and manipulation (SCAM 2006). Journal of Software: Evolution and Process, 2007, 19, 203-204.	1.1	0
207	How design notations affect the comprehension of Web applications. Journal of Software: Evolution and Process, 2007, 19, 339-359.	1.1	1
208	Web Services Regression Testing. , 2007, , 205-234.		33
209	COCONUT: COde COmprehension Nurturant Using Traceability. , 2006, , .		5
210	On the Use of Line Co-change for Identifying Crosscutting Concern Code. , 2006, , .		31
211	An empirical study on the usefulness of Conallen's stereotypes inWeb application comprehension. , 2006, , .		6
212	Service Composition (re)Binding Driven by Applicationâ€“Specific QoS. Lecture Notes in Computer Science, 2006, , 141-152.	1.3	36
213	A language-independent software renovation framework. Journal of Systems and Software, 2005, 77, 225-240.	4.5	22
214	Towards the Integration of Versioning Systems, Bug Reports and Source Code Meta-Models. Electronic Notes in Theoretical Computer Science, 2005, 127, 87-99.	0.9	15
215	Improving network applications security. , 2005, , .		27
216	The C-Cube framework. , 2005, , .		3

#	Article	IF	Citations
217	An experimental investigation of formality in UML-based development. IEEE Transactions on Software Engineering, 2005, 31, 833-849.	5.6	91
218	An approach for QoS-aware service composition based on genetic algorithms. , 2005, , .		668
219	Using Test Cases as Contract to Ensure Service Compliance Across Releases. Lecture Notes in Computer Science, 2005, , 87-100.	1.3	35
220	TransientMeter: A Distributed Measurement System for Power Quality Monitoring. IEEE Transactions on Power Delivery, 2004, 19, 456-463.	4.3	50
221	Compiler Hacking for Source Code Analysis. Software Quality Journal, 2004, 12, 383-406.	2.2	7
222	Assessing staffing needs for a software maintenance project through queuing simulation. IEEE Transactions on Software Engineering, 2004, 30, 43-58.	5.6	55
223	Assessing and improving state-based class testing: a series of experiments. IEEE Transactions on Software Engineering, 2004, 30, 770-783.	5.6	101
224	Analyzing cloning evolution in the Linux kernel. Information and Software Technology, 2002, 44, 755-765.	4.4	106
225	Object-oriented design patterns recovery. Journal of Systems and Software, 2001, 59, 181-196.	4.5	72