Oscar Nierstrasz

List of Publications by Year
in descending order

Source: https://exaly.com/author-pdf/7169542/publications.pdf

Version: 2024-02-01

152 papers	3,608 citations	331259 21 h-index	233125 45 g-index
160 all docs	160 docs citations	160 times ranked	1677 citing authors

#	Article	IF	Citations
1	RepliComment: Identifying clones in code comments. Journal of Systems and Software, 2021, 182, 111069.	3.3	3
2	Security Header Fields in HTTP Clients. , 2021, , .		2
3	Caveats in Eliciting Mobile App Requirements. , 2020, , .		2
4	Security code smells in Android ICC. Empirical Software Engineering, 2019, 24, 3046-3076.	3.0	21
5	Mining inline cache data to order inferred types in dynamic languages. Science of Computer Programming, 2018, 161, 105-121.	1.5	0
6	Efficient parsing with parser combinators. Science of Computer Programming, 2018, 161, 57-88.	1.5	2
7	Towards actionable visualization for software developers. Journal of Software: Evolution and Process, 2018, 30, e1923.	1.2	7
8	Improving live debugging of concurrent threads through thread histories. Science of Computer Programming, 2018, 161, 122-148.	1.5	2
9	JIT feedback. , 2018, , .		9
10	Security Smells in Android. , 2017, , .		34
11	Renraku. , 2017, , .		0
12	Moldable Tools for Object-Oriented Development. , 2017, , 77-101.		0
13	Exemplifying Moldable Development. , 2016, , .		2
14	Inferring Types by Mining Class Usage Frequency from Inline Caches. , 2016, , .		6
15	Exploring cheap type inference heuristics in dynamically typed languages. , 2016, , .		3
16	A Case Study on Type Hints in Method Argument Names in Pharo Smalltalk Projects. , 2016, , .		2
17	MetaVis: Exploring Actionable Visualization. , 2016, , .		4
18	The Death of Object-Oriented Programming. Lecture Notes in Computer Science, 2016, , 3-10.	1.0	0

#	ARTICLE	IF	CITATIONS
19	Moldable, context-aware searching with Spotter. , 2016, , .		2
20	The Object Repository. , 2016, , .		1
21	Marea: A Semi-Automatic Decision Support System for Breaking Dependency Cycles. , 2016, , .		5
22	Evaluating an Architecture Conformance Monitoring Solution. , 2016, , .		1
23	Tracking Null Checks in Open-Source Java Systems. , 2016, , .		2
24	CommunityExplorer. , 2016, , .		2
25	When QualityAssistant Meets Pharo. , 2016, , .		1
26	Polymorphism in the Spotlight: Studying Its Prevalence in Java and Smalltalk. , 2015, , .		5
27	Spotter: towards a unified search interface in IDEs. , 2015, , .		3
28	Towards moldable development tools. , 2015, , .		5
29	Bounded seas. Computer Languages, Systems and Structures, 2015, 44, 114-140.	1.4	7
30	Practical domain-specific debuggers using the Moldable Debugger framework. Computer Languages, Systems and Structures, 2015, 44, 89-113.	1.4	21
31	The moldable inspector. , 2015, , .		18
32	GTInspector: a moldable domain-aware object inspector. , 2015, , .		2
33	A Unified Approach to Architecture Conformance Checking. , 2015, , .		23
34	Parsing for agile modeling. Science of Computer Programming, 2015, 97, 150-156.	1.5	7
35	Domain Globalization: Using Languages to Support Technical and Social Coordination. Lecture Notes in Computer Science, 2015, , 70-87.	1.0	4
36	Overthrowing the Tyranny of Alphabetical Ordering in Documentation Systems. , 2014, , .		4

#	Article	IF	Citations
37	Mining the Ecosystem to Improve Type Inference for Dynamically Typed Languages. , 2014, , .		6
38	Talents: an environment for dynamically composing units of reuse. Software - Practice and Experience, 2014, 44, 413-432.	2.5	7
39	Predicting dependences using domainâ€based coupling. Journal of Software: Evolution and Process, 2014, 26, 50-76.	1.2	3
40	Delegation proxies. , 2014, , .		8
41	Mining frequent bug-fix code changes. , 2014, , .		30
42	Evolutionary and collaborative software architecture recovery with Softwarenaut. Science of Computer Programming, 2014, 79, 204-223.	1.5	31
43	How Do Software Architects Specify and Validate Quality Requirements?. Lecture Notes in Computer Science, 2014, , 374-389.	1.0	14
44	The Moldable Debugger: A Framework for Developing Domain-Specific Debuggers. Lecture Notes in Computer Science, 2014, , 102-121.	1.0	18
45	Software Engineering for Self-Adaptive Systems: A Second Research Roadmap. Lecture Notes in Computer Science, 2013, , 1-32.	1.0	317
46	Analyzing PL/1 legacy ecosystems: An experience report. , 2013, , .		1
47	Towards a moldable debugger. , 2013, , .		3
48	Categorizing developer information needs in software ecosystems. , 2013, , .		15
49	Incremental Dynamic Updates with First-class Contexts.. Journal of Object Technology, 2013, 12, 1:1.	0.8	1
50	Farewell editorial.. Journal of Object Technology, 2013, 12, .	0.8	0
51	Agile software assessment with Moose. Software Engineering Notes: an Informal Newsletter of the Special Interest Committee on Software Engineering / ACM, 2012, 37, 1-5.	0.5	2
52	Ownership, filters and crossing handlers. , 2012, , .		3
53	Agile software assessment (Invited paper). , 2012, , .		3
54	Specifying Dynamic Analyses by Extending Language Semantics. IEEE Transactions on Software Engineering, 2012, 38, 694-706.	4.3	0

#	Article	IF	Citations
55	Exploiting Dynamic Information in IDEs Improves Speed and Correctness of Software Maintenance Tasks. IEEE Transactions on Software Engineering, 2012, 38, 579-591.	4.3	34
56	Object-centric debugging. , 2012, , .		22
57	Incremental Dynamic Updates with First-Class Contexts. Lecture Notes in Computer Science, 2012, , 304-319.	1.0	4
58	Modeling Domain-Specific Profilers.. Journal of Object Technology, 2012, 11, 5:1.	0.8	4
59	Seuss: Decoupling responsibilities from static methods for fine-grained configurability.. Journal of Object Technology, 2012, 11, 3:1.	0.8	3
60	Can We Predict Dependencies Using Domain information?. , 2011, , .		9
61	Flexible object layouts. ACM SIGPLAN Notices, 2011, 46, 959-972.	0.2	1
62	SmartGroups: Focusing on Task-Relevant Source Artifacts in IDEs. , 2011, , .		5
63	Talents. , 2011, , .		0
64	Using first-class contexts to realize dynamic software updates. , 2011, , .		2
65	Flexible object layouts. , 2011, , .		13
66	Generating a catalog of unanticipated schemas in class hierarchies using Formal Concept Analysis. Information and Software Technology, 2010, 52, 1167-1187.	3.0	13
67	Software Cartography: thematic software visualization with consistent layout. Journal of Software: Evolution and Process, 2010, 22, 191-210.	1.1	19
68	Embedding spatial software visualization in the IDE. , 2010, , .		13
69	Recovery and analysis of transaction scope from scattered information in Java Enterprise Applications. , 2010, , .		11
70	Lessons in Software Evolution Learned by Listening to Smalltalk. Lecture Notes in Computer Science, 2010, , 77-95.	1.0	3
71	Language Boxes. Lecture Notes in Computer Science, 2010, , 274-293.	1.0	15
72	Domain-Specific Program Checking. Lecture Notes in Computer Science, 2010, , 213-232.	1.0	10

#	Article	IF	Citations
73	Modeling Features at Runtime. Lecture Notes in Computer Science, 2010, , 138-152.	1.0	2
74	Ten Things I Hate About Object-Oriented Programming.. Journal of Object Technology, 2010, 9, .	0.8	8
75	Long Live Metadata!. Journal of Object Technology, 2010, 9, .	0.8	0
76	Transactional memory in a dynamic language. Computer Languages, Systems and Structures, 2009, 35, 21-30.	1.4	8
77	Practical, pluggable types for a dynamic language. Computer Languages, Systems and Structures, 2009, 35, 48-62.	1.4	11
78	CompAS: A new approach to commonality and variability analysis with applications in computer assisted orthopaedic surgery. Information and Software Technology, 2009, 51, 448-459.	3.0	5
79	Model-Centric, Context-Aware Software Adaptation. Lecture Notes in Computer Science, 2009, , 128-145.	1.0	15
80	Autumn Leaves: Curing the Window Plague in IDEs. , 2009, , .		21
81	Assigning bug reports using a vocabulary-based expertise model of developers. , 2009, , .		164
82	Senseo: Enriching Eclipse's static source views with dynamic metrics. , 2009, , .		4
83	Comparative analysis of evolving software systems using the Gini coefficient. , 2009, , .		53
84	Augmenting static source views in IDEs with dynamic metrics. , 2009, , .		27
85	Flow-Centric, Back-in-Time Debugging. Lecture Notes in Business Information Processing, 2009, , 272-288.	0.8	8
86	Dynamic synchronization. , 2009, , .		1
87	Tackling software navigation issues of the Smalltalk IDE. , 2009, , .		2
88	Stateful traits and their formalization. Computer Languages, Systems and Structures, 2008, 34, 83-108.	1.4	48
89	Consistent Layout for Thematic Software Maps. , 2008, , .		30
90	FAMOOSr 2008: Workshop on FAMIX and Moose in Software Reengineering. , 2008, , .		2

#	Article	IF	Citations
91	Test BlueprintsÂ¿ Exposing Side Effects in Execution Traces to Support Writing Unit Tests. Software Maintenance and Reengineering (CSMR), Proceedings of the European Conference on, 2008, , .	0.0	6
92	Exploiting Runtime Information in the IDE. , 2008, , .		14
93	Practical Object-Oriented Back-in-Time Debugging. Lecture Notes in Computer Science, 2008, , 592-615.	1.0	48
94	Enriching Reverse Engineering with Annotations. Lecture Notes in Computer Science, 2008, , 660-674.	1.0	11
95	Back to the Future in One Week â€" Implementing a Smalltalk VM in PyPy. Lecture Notes in Computer Science, 2008, , 123-139.	1.0	17
96	Change-Enabled Software Systems. Lecture Notes in Computer Science, 2008, , 64-79.	1.0	6
97	Context-oriented Programming.. Journal of Object Technology, 2008, 7, 125.	0.8	345
98	Composing new abstractions from object fragments. , 2008, , .		0
99	JExample: Exploiting Dependencies between Tests to Improve Defect Localization. Lecture Notes in Business Information Processing, 2008, , 73-82.	0.8	5
100	Feature driven browsing. , 2007, , .		11
101	Context-oriented programming. , 2007, , .		51
102	Transactional memory for smalltalk. , 2007, , .		6
103	Encapsulating and exploiting change with changeboxes. , 2007, , .		24
104	Practical, pluggable types. , 2007, , .		3
105	The Inevitable Stability of Software Change. , 2007, , .		32
106	Example-Driven Reconstruction of Software Models. , 2007, , .		7
107	Towards faster method search through static ecosystem analysis. , 2007, , .		0
108	User-changeable visibility. ACM SIGPLAN Notices, 2007, 42, 171-190.	0.2	2

#	Article	IF	Citations
109	Modeling Change as a First-Class Entity. Proceedings / Australian Software Engineering Conference, 2007, , .	0.0	0
110	Tracking Objects to Detect Feature Dependencies. , 2007, , .		25
111	A Quantitative Analysis of Developer Information Needs in Software Ecosystems. , 2007, , .		13
112	Stateful Traits. Lecture Notes in Computer Science, 2007, , 66-90.	1.0	9
113	Composing Tests from Examples.. Journal of Object Technology, 2007, 6, 71.	0.8	13
114	DictÅ• , 2007, , .		0
115	User-changeable visibility. , 2007, , .		3
116	Idioms for Composing Games with EToys. , 2006, , .		4
117	On the effectiveness of clone detection by string matching. Journal of Software: Evolution and Process, 2006, 18, 37-58.	1.1	51
118	Traits. ACM Transactions on Programming Languages and Systems, 2006, 28, 331-388.	1.7	181
119	Flattening Traits.. Journal of Object Technology, 2006, 5, 129.	0.8	33
120	Classbox/J. ACM SIGPLAN Notices, 2005, 40, 177-189.	0.2	8
121	A data-centric approach to composing embedded, real-time software components. Journal of Systems and Software, 2005, 74, 25-34.	3.3	14
122	A calculus for reasoning about software composition. Theoretical Computer Science, 2005, 331, 367-396.	0.5	20
123	Classboxes: controlling visibility of class extensions. Computer Languages, Systems and Structures, 2005, 31, 107-126.	1.4	37
124	The story of moose. Software Engineering Notes: an Informal Newsletter of the Special Interest Committee on Software Engineering / ACM, 2005, 30, 1-10.	0.5	18
125	Moose. Software Engineering Notes: an Informal Newsletter of the Special Interest Committee on Software Engineering / ACM, 2005, 30, 99-102.	0.5	12
126	Classbox/J. , 2005, , .		59

#	Article	IF	Citations
127	Moose. , 2005, , .		25
128	The story of moose. , 2005, , .		48
129	On the Revival of Dynamic Languages. Lecture Notes in Computer Science, 2005, , 1-13.	1.0	18
130	Object-Oriented Reengineering Patterns â€" An Overview. Lecture Notes in Computer Science, 2005, , 1-9.	1.0	6
131	Composable Encapsulation Policies. Lecture Notes in Computer Science, 2004, , 26-50.	1.0	15
132	Software Evolution as the Key to Productivity. Lecture Notes in Computer Science, 2004, , 274-282.	1.0	6
133	Traits: Composable Units of Behaviour. Lecture Notes in Computer Science, 2003, , 248-274.	1.0	176
134	A Calculus for Modeling Software Components. Lecture Notes in Computer Science, 2003, , 339-360.	1.0	27
135	Components for embedded software. , 2002, , .		49
136	A Component Model for Field Devices. Lecture Notes in Computer Science, 2002, , 200-209.	1.0	49
137	Explicit Namespaces. Lecture Notes in Computer Science, 2000, , 77-89.	1.0	14
138	Finding refactorings via change metrics. , 2000, , .		148
139	OpenSpaces: An Object-Oriented Framework for Reconfigurable Coordination Spaces. Lecture Notes in Computer Science, 2000, , 1-18.	1.0	12
140	Scripting Coordination Styles. Lecture Notes in Computer Science, 2000, , 19-35.	1.0	5
141	Components, Scripts and Glue. , 2000, , 13-25.		37
142	Design guidelines for â€œtailorableâ€•frameworks. Communications of the ACM, 1997, 40, 60-64.	3.3	28
143	Research directions in software composition. ACM Computing Surveys, 1995, 27, 262-264.	16.1	89
144	Requirements for a composition language. Lecture Notes in Computer Science, 1995, , 147-161.	1.0	33

#	Article	IF	Citations
145	Regular types for active objects. , 1993, , .		71
146	Component-oriented software development. Communications of the ACM, 1992, 35, 160-165.	3.3	104
147	Towards an object calculus. Lecture Notes in Computer Science, 1992, , 1-20.	1.0	26
148	Towards a type theory for active objects. ACM SIGPLAN OOPS Messenger, 1991, 2, 89-93.	0.1	7
149	Towards a type theory for active objects. , 1991, , .		11
150	Class management for software communities. Communications of the ACM, 1990, 33, 90-103.	3.3	89
151	Viewing object as patterns of communicating agents. , 1990, , .		31
152	Viewing object as patterns of communicating agents. ACM SIGPLAN Notices, 1990, 25, 38-43.	0.2	17