Li WANG, Xianglong KONG, Jiahui WANG, Bixin LI, 2022. An incremental software architecture recovery technique driven by code changes. *Frontiers of Information Technology & Electronic Engineering*, 23(5):664-677. https://doi.org/10.1631/FITEE.2100461

An incremental software architecture recovery technique driven by code changes

Key words: Architecture recovery; Software evolution; Code change

Corresponding author: Bixin LI
E-mail: bx.li@seu.edu.cn
ORCID: https://orcid.org/0000-0001-9916-4790

Motivation

1. It is difficult to keep software architecture up to date with code changes during software evolution.

2. The inconsistency is caused by the limitation of standard development specification and human-power resource, which may impact software maintenance and evolution.

3. In practice, most large-scale projects have one or more high-quality architecture documents which are generated at the beginning of development or are revised through the maintenance.

Main idea

1. Well-documented architecture usually presents the initial structure of specific versions.

2. We can track code changes during software evolution, and we aim to build a mapping mechanism between code-level changes and architecture-level updates.

Method

1. The changed code files and file-level dependency graph are obtained by analyzing code before and after software evolution.

2. The changed code files and file-level dependency graph are preprocessed to determine the changed elements.

3. These incremental entities are processed using double classifiers to achieve the top-down incremental update of the software architecture.

Method (Cont'd)

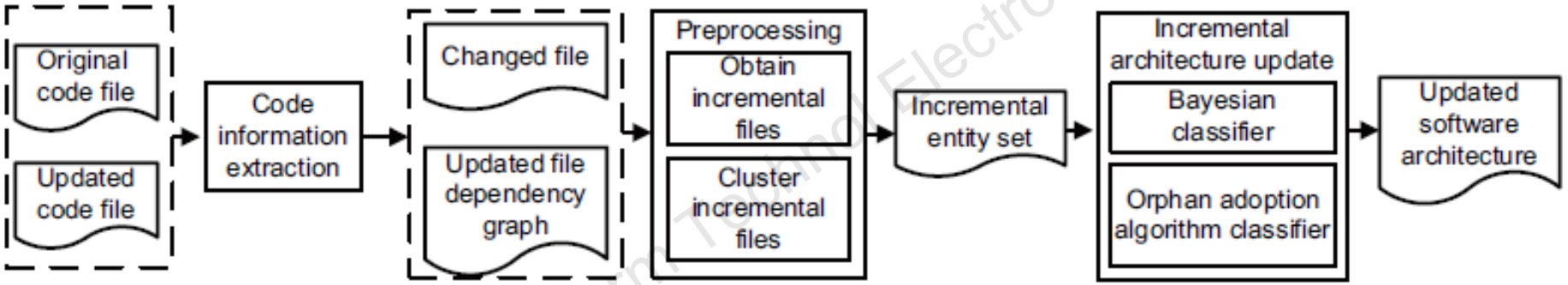

Fig. 1 Incremental software architecture recovery (ISAR) framework

Major results

Table 4 Subject system statistics

Project	LOC	Number of files	Description
Okhttp	53 114	325	An Android lightweight framework for network requests
Mabatis	51 044	918	A persistence layer framework to support customized SQL
Mockito	40 411	863	A simulation test framework for simple verification error production
Junit	3512	47	A regression testing framework for Java
Retrofit	19 193	235	A restful HTTP network request framework
Jadx	45 619	574	An open source tool to decompile APK files
Terrier	55 485	1122	A program for rapid development of web and desktop search engines
Clone	10 198	91	Game written in Java
Freecol	118 428	773	A turn-based strategy game; the open source version of colonization
Fastjson	117 300	1972	An open source tool for parsing and packaging JSON formatted data

Major results (Cont'd)

Table 5 MoJoSim scores of the studied techniques

Project	MoJoSim score		
	ISAR	Bunch	DBDP
Okhttp	0.92	0.71	0.88
Mabatis	0.95	0.73	0.89
Mockito	0.92	0.72	0.88
Junit	0.95	0.65	0.75
Retrofit	0.85	0.57	0.85
Jadx	0.76	0.55	0.64
Terrier	0.90	0.68	0.88
Clone	0.91	0.77	0.90
Freecol	0.91	0.61	0.75
Fastjson	0.92	0.62	0.82
Average	0.90	0.66	0.82

Table 6 Turbo MQ scores on the projects of ISAR

Project	Turbo MQ score			
	Version 2	Version 3	Version 4	Version 5
Okhttp	12.78	10.89	8.47	8.32
Mabatis	13.01	11.66	9.65	8.88
Mockito	11.89	10.08	8.23	7.95
Junit	2.55	2.51	2.41	2.12
Retrofit	6.89	6.88	5.64	5.44
Jadx	7.11	6.84	5.78	4.78
Terrier	14.55	12.66	10.44	10.32
Clone	13.24	13.04	10.47	9.19
Freecol	20.64	16.55	14.34	12.08
Fastjson	19.87	14.52	13.94	12.99
Average	12.25	10.56	8.94	8.21

Major results (Cont'd)

Table 7 Time consumption of studied techniques

Project	Time (s)											
	Version 2			Version 3			Version 4			Version 5		
	ISAR	DBDP	Bunch	ISAR	DBDP	Bunch	ISAR	DBDP	Bunch	ISAR	DBDP	Bunch
Okhttp	8378	32 175	22 547	8641	30 445	25 963	9655	30 927	21 874	7931	29 488	22 554
Mabatis	12 456	72 866	57 412	11 456	68 543	55 478	7645	39 012	36 885	6468	58 238	54 365
Mackito	14 748	57 915	38 452	12 658	59 611	36 462	18 984	65 025	60 742	12 750	60 924	58 168
Junit	1988	3139	2151	1964	3830	2485	1648	3295	2185	2277	3934	2987
Retrofit	2834	7032	5647	2904	7397	5784	3545	8290	5620	2733	7128	5583
Jadx	12 445	51 904	41 875	14 489	55 391	39 845	15 794	55 031	41 321	10 494	50 415	40 871
Terrier	9897	63 008	52 965	8478	57 687	51 492	10 021	59 422	50 774	10 487	61 140	49 863
Clone	6291	11 008	8456	4445	9313	7458	5156	10 420	6985	5265	9772	7059
Freecol	16 854	121 856	98 635	15 687	119 743	88 919	18 900	131 695	108 990	15 805	118 097	106 983
Fastjson	19 124	149 658	99 873	18 871	139 198	97 668	18 844	133 109	90 869	19 657	121 886	110 493

Conclusions

1. We proposed an incremental software architecture recovery technique.

2. We built a mapping between code-level changes and architecture-level updates.

3. Our approach can generally improve the effectiveness and efficiency.

Li WANG received her MA degree from Tongji University, Shanghai, China, in 2009. She is currently a PhD candidate in School of Computer Science and Engineering, Southeast University, Nanjing, China. Her research interests include architecture recovery and software testing.

Xianglong KONG is a lecturer at School of Computer Science and Engineering, Southeast University, Nanjing, China. He received his BE and PhD degrees (under the supervision of Prof. Bixin LI) from Southeast University, in 2009 and 2018, respectively. His research interests include API recommendation, program repair, and architecture recovery.

Jiahui WANG received her BE degree from Nanjing Audit University in 2016, and her MA degree from Southeast University in 2019. She is currently a senior engineer at Huawei Digital Technology Lab, Suzhou, China. Her research interests include architecture recovery and software testing.

Bixin LI is a professor at Southeast University. He is director of Institute of Software Engineering, Southeast University. His research interests include intelligent software development, software testing and defect detection, and software security.