Candoia: A Platform for Building and Sharing Mining Software Repositories Tools as Apps

Nitin Mukesh Tiwari

Department of Computer Science
Iowa State University
nmtiwari@iastate.edu

POSC Committee
Major Advisor: Dr. Hridesh Rajan
Dr. Gurpur Prabhu
Dr. Steaven Kautz

- Problem
  - Building easily customizable, adoptable and applicable mining software repository tools
- Solution
  - An ecosystem which offers suitable abstractions and computational means to realize the process for building and sharing MSR tools as apps.
- Evaluation
- Related works, Conclusion, & Future Work
  - Existing open source tools and frameworks
  - Open source datasets

Goal

- Reduce the efforts required to build MSR tools
- Ease the process of adopting, customizing and sharing MSR tools
- Allow users to run third-party tools more securely

Scenario 1: MSR Tool Building and Sharing

User wants to build a tool for Association Mining

- Source code
  - Java source code
- Version control systsm(VCS)
  - GIT
- Bug Information
  - Github-Issues Bug Tracker

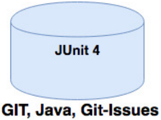


Figure: Software Repository Data

Scenario 1: Tool Building and Sharing

Build a tool for association mining

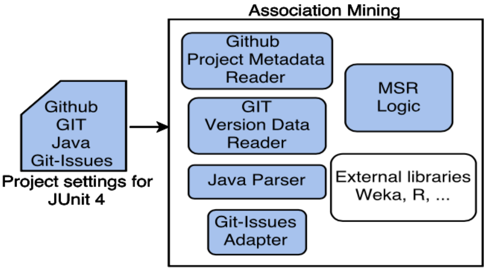


Figure: Association Minging Tool Details

Scenario 1: Tool Building and Sharing

Share the built tool with other researchers and practitioners

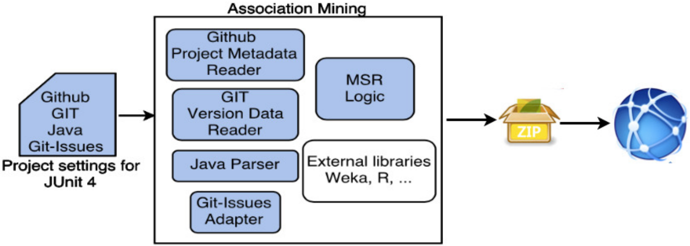


Figure: Complete process of building and sharing tool

Scenario 2: Adopting a shared tool

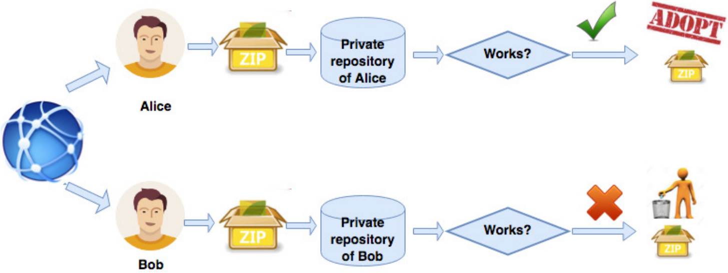


Figure: Repository Mining Tool Building

Scenario 2: Adopting a shared MSR tool

Why Bob is not able to adopt the same tool?
&
What are the possible points of failure?

How MSR tools are build?

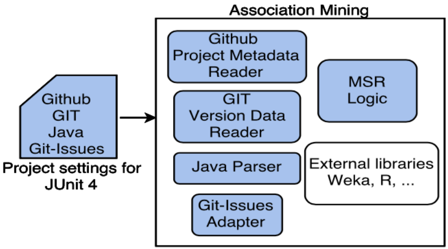
- MSR tools are build for specific project setting.
- A project setting defines types and sources of various MSR artifacts

How MSR tools are build?

- MSR tools are build for specific project setting.
- A project setting defines types and sources of various MSR artifacts
- MSR Artifacts: Any kind of information realted to your software
    - Revision history from version control system (VCS)
    - Source code of programming language(s)
    - Bug data from bug trackers
    - Project metadata
    - users and teams data from forges

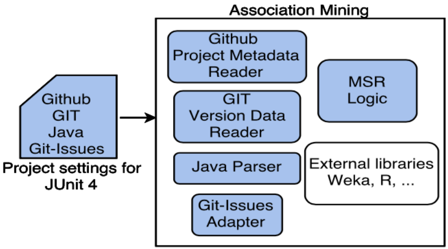
Association Mining Tool Challenges

- User is required to build necessary data preperation tools



Association Mining Tool Challenges

- User is required to build necessary data preperation tools
- Tool are tightly coupled with SCM systems



Potential Points of failure

Failure Cause

An MSR tool build for one project setting may not work for different project settings.

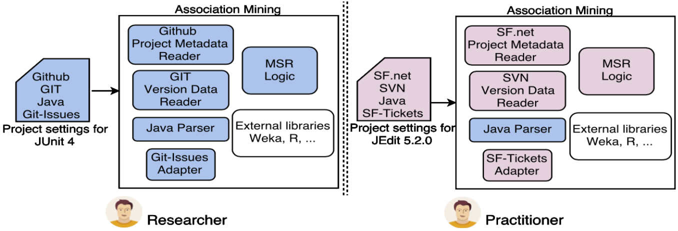

Association Mining
Github Project Metadata Reader
MSR Logic
GIT Version Data Reader
Java Parser
External libraries Weka, R, ...
Git-Issues Adapter
Github GIT Java Git-Issues
Project settings for JUnit 4
Researcher
Association Mining
SF.net Project Metadata Reader
MSR Logic
SVN Version Data Reader
Java Parser
External libraries Weka, R, ...
SF-Tickets Adapter
SF.net SVN Java SF-Tickets
Project settings for JEdit 5.2.0
Practitioner

Figure: A scenario of a practitioner adapting a MSR tool built by a

How to make adopting MSR tools possible?

Our Solution

- Software As Apps
  - Make tool size smaller by pushing generic functionality in the platform
- Programs as Script
  - Make tool components more like scripts, than programs, easier to customize
- Platform and appstore
  - Provide a platform and ecosystem to distribute these tools

How to build Software as Apps?

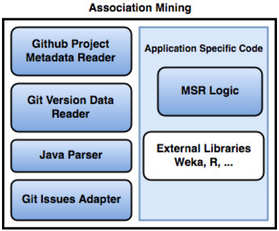

Association Mining
Github Project Metadata Reader
Git Version Data Reader
Java Parser
Git Issues Adapter
Application Specific Code
MSR Logic
External Libraries Weka, R, ...

Figure: An MSR Tool Structure

Pull supporting tools out and make them available as part of platform

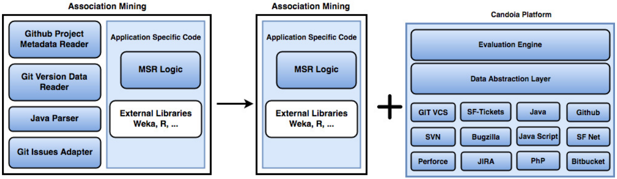


Figure: Repository Mining Tool Building

How to build programs as Script?

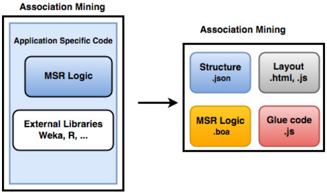


Figure: Transofrmation of MSR program to MSR tool

MSR Tool as Candoia App

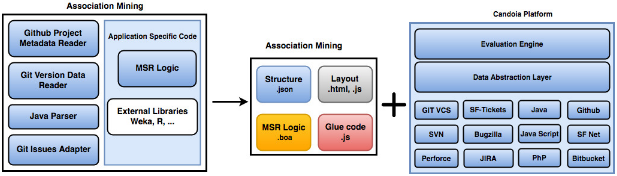
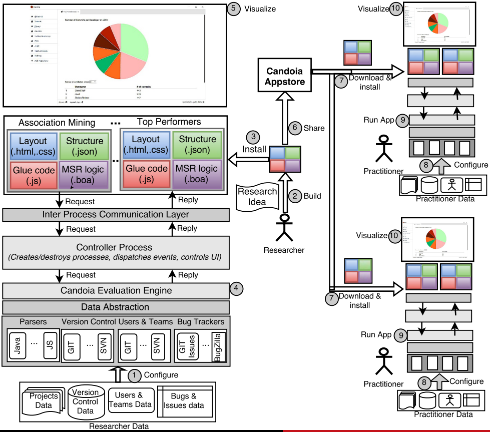


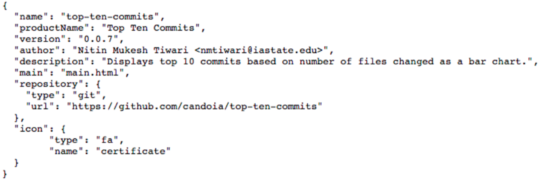
Figure: MSR tool to Candoia App transformation

Process Of Building and Sharing Candoia App



Candoia App Structure

- Mining Logic
  - Extension of Boa DSL
- Visualization Layout
  - Html
  - CSS
- Glue Code
  - Java script
- Structure Decription
  - Json



```
{
  "name": "top-ten-commits",
  "productName": "Top Ten Commits",
  "version": "0.0.7",
  "author": "Nitin Mukesh Tiwari <nmtiwari@iastate.edu>",
  "description": "Displays top 10 commits based on number of files changed as a bar chart.",
  "main": "main.html",
  "repository": {
    "type": "git",
    "url": "https://github.com/candoia/top-ten-commits"
  },
  "icon": {
        "type": "fa",
        "name": "certificate"
  }
}
```

Figure: Structure of an Candoia app

Data Abstraction

- Queries are written over data abstractions
- Data abstractions provide capability of running an app over data collected from diverse sources
- Candoia's data abstraction is extension of Boa DSL.

Candoia Data Schema

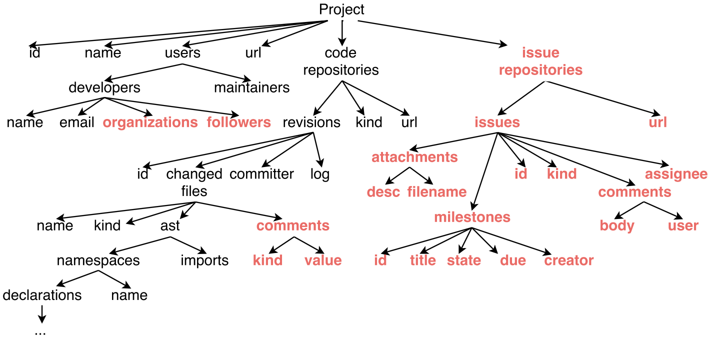


Figure: Candoia's data schema

Customizations

Two levels of customizations

- Data source customizations
- App Customization

Customizations

Two levels of customizations

- Data source customizations
  - Concerned with changing the source of the data
  - No Change in app required
  - Just rerun the application with new datasource
- Customization in Apps

Customizations

Two levels of customizations

- Data source customizations
- Customization in Apps
  - Concerned with customizing different part of the apps
  - App customizations in Candoia are more focused in terms of findings the right component(s) for customizations

Customizations

Two levels of customizations

- Data source customizations
- Customization in Apps
  - Concerned with customizing different part of the apps
  - app customizations in Candoia are more focused in terms of findings the right component(s) for customizations

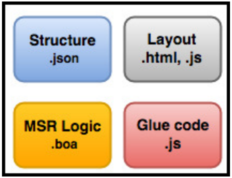


Figure: An MSR App Structure

Customizations

Two levels of customizations

- Data source customizations
- Customization in Apps
  - Concerned with customizing different part of the apps
  - App customizations in Candoia are more focused in terms of findings the right component(s) for customizations
  - Script based app, easire to customize

Candoia Evaluation Engine

- Interpreter based Query evaluator for extended Boa DSL

Candoia Evaluation Engine

- Interpreter based Query evaluator of extended Boa DSL
- Reads data from local and remote software artifacts
  - Forges: Github, Source Forge
  - VCS: GIT, SVN
  - Bug Tracker: Bugzilla, JIRA, Github-Issue, SF-Ticket
  - Programming Language: Java, Javascript

Candoia Evaluation Engine

- Interpreter based Query evaluator of extended Boa DSL
- Reads data from local and remote software artifacts
  - Forges: Github, Source Forge
  - VCS: GIT, SVN
  - Bug Tracker: Bugzilla, JIRA, Github-Issue, SF-Ticket
  - Programming Language: Java, Javascript
- Process level parallelization for running multiple apps
- Thread level prarallelization for dataset creation
- Provides fine grained control to Candoia frontend

Candoia Security Concern

- Private software data
  - Allow access to user data on a need to know basis
- Installing and running third party apps
  - Prevent apps from corrupting each other
  - Prevent apps from accessing system resources directly

Chromium based Candoia frontend

- Candoia builds on the process architecture of Chromium, each window runs as a process
- Process communica tes with controller process via Inter Process Communication
- Controller process mediates interaction between file system, window data etc. by exposing APIs.
- An app can only access resources using exposed APIs.

Few APIs available to Candoia app

- Running MSR queries (api.boa)
- Reading (not writing) files within app (api.fs)
- Saving arbitrary data between instances (api.store)
- Getting its own package info such as version (api.meta)
- Inter-Process-Communication handle (api.ipc)

var data = api.boa.run('myprog.boa')

Candoia Exchange

- A web platform for sharing MSR apps
- Candoia frameworks can connect to exchange for gathering information and installing apps

Evaluation

- Applicability
  - our claim is that MSR tasks and hypotheses can be expressed and evaluated using Candoia platformś capabilities.
- Adoptability
  - Candoia apps are portable across diverse project settings and require no changes
- Customizability
  - Our claim that performing customizations in Candoia requires less efforts in terms of LOC.

Projects used in Evaluation

Projects	VCS	PL	Bugs	#LOC	#Revs	#Bugs	#Devs
Tomcat 8.0.24 (TC)	SVN	Java	Bugzilla	381350	17433	3023	32
Hadoop 2.7.1 (HD)	Git	Java	JIRA	2217636	14301	10333	146
JUnit 4 (JU)	Git	Java	GitHub	30535	2115	148	127
SLF4j 1.7.12 (SLF)	Git	Java	JIRA	20866	1436	332	59
Bootstrap 3.3.5 (BT)	Git	JS	GitHub	65885	11840	213	718
Node.js 0.12.7 (ND)	Git	JS	GitHub	3405739	14695	955	105
Grunt 0.4.6 (GT)	Git	JS	GitHub	3596	1399	155	29
JQuery 2.1.4 (JQ)	Git	JS	GitHub	45212	6153	165	87
PMD 5.3.3 (PMD)	Git	Java	SF	175866	8736	1394	102
JEdit 5.2.0 (JE)	SVN	Java	SF	224127	24509	3926	7

Figure: Test projects.

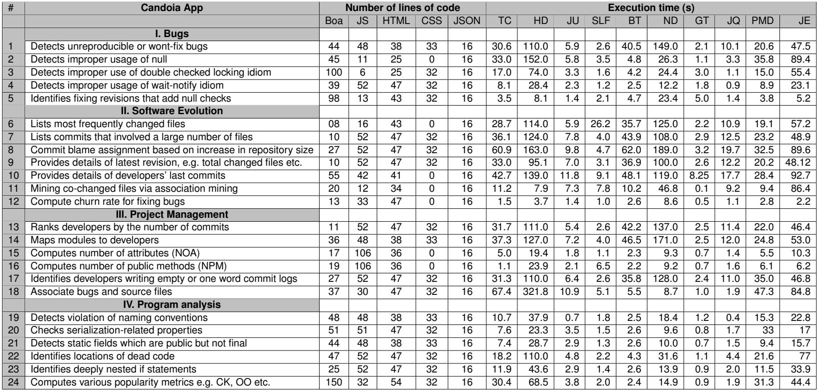
Applicability

Claim: MSR tasks and hypotheses can be ex- pressed and evaluated using Candoia

Evaluation Strategy:

- Created 24 Candoia apps covering
  - Bugs
  - Software Evolution
  - Project Management
  - Source code analysis and Programming practices
- Tested these applications over test projects

Candoia apps used in evaluation



#	Candoia App	Number of lines of code					Execution time (s)									
		Boa	JS	HTML	CSS	JSON	TC	HD	JU	SLF	BT	ND	GT	JQ	PMD	JE
	I. Bugs															
1	Detects unreproducible or wont-fix bugs	44	48	38	33	16	30.6	110.0	5.9	2.6	40.5	149.0	2.1	10.1	20.6	47.5
2	Detects improper usage of null	45	11	25	0	16	33.0	152.0	5.8	3.5	4.8	26.3	1.1	3.3	35.8	89.4
3	Detects improper use of double checked locking idiom	100	6	25	32	16	17.0	74.0	3.3	1.6	4.2	24.4	3.0	1.1	15.0	55.4
4	Detects improper usage of wait-notify idiom	39	52	47	32	16	8.1	28.4	2.3	1.2	2.5	12.2	1.8	0.9	8.9	23.1
5	Identifies fixing revisions that add null checks	98	13	43	32	16	3.5	8.1	1.4	2.1	4.7	23.4	5.0	1.4	3.8	5.2
	II. Software Evolution															
6	Lists most frequently changed files	08	16	43	0	16	28.7	114.0	5.9	26.2	35.7	125.0	2.2	10.9	19.1	57.2
7	Lists commits that involved a large number of files	10	52	47	32	16	36.1	124.0	7.8	4.0	43.9	108.0	2.9	12.5	23.2	48.9
8	Commit blame assignment based on increase in repository size	27	52	47	32	16	60.9	163.0	9.8	4.7	62.0	189.0	3.2	19.7	32.5	89.6
9	Provides details of latest revision, e.g. total changed files etc.	10	52	47	32	16	33.0	95.1	7.0	3.1	36.9	100.0	2.6	12.2	20.2	48.12
10	Provides details of developers' last commits	55	42	41	0	16	42.7	139.0	11.8	9.1	48.1	119.0	8.25	17.7	28.4	92.7
11	Mining co-changed files via association mining	20	12	34	0	16	11.2	7.9	7.3	7.8	10.2	46.8	0.1	9.2	9.4	86.4
12	Compute churn rate for fixing bugs	13	33	47	0	16	1.5	3.7	1.4	1.0	2.6	8.6	0.5	1.1	2.8	2.2
	III. Project Management															
13	Ranks developers by the number of commits	11	52	47	32	16	31.7	111.0	5.4	2.6	42.2	137.0	2.5	11.4	22.0	46.4
14	Maps modules to developers	36	48	38	33	16	37.3	127.0	7.2	4.0	46.5	171.0	2.5	12.0	24.8	53.0
15	Computes number of attributes (NOA)	17	106	36	0	16	5.0	19.4	1.8	1.1	2.3	9.3	0.7	1.4	5.5	10.3
16	Computes number of public methods (NPM)	19	106	36	0	16	1.1	23.9	2.1	6.5	2.2	9.2	0.7	1.6	6.1	6.2
17	Identifies developers writing empty or one word commit logs	27	52	47	32	16	31.3	110.0	6.4	2.6	35.8	128.0	2.4	11.0	35.0	46.8
18	Associate bugs and source files	37	30	47	32	16	67.4	321.8	10.9	5.1	5.5	8.7	1.0	1.9	47.3	84.8
	IV. Program analysis															
19	Detects violation of naming conventions	48	48	38	33	16	10.7	37.9	0.7	1.8	2.5	18.4	1.2	0.4	15.3	22.8
20	Checks serialization-related properties	51	51	47	32	16	7.6	23.3	3.5	1.5	2.6	9.6	0.8	1.7	33	17
21	Detects static fields which are public but not final	44	48	38	33	16	7.4	28.7	2.9	1.3	2.6	10.0	0.7	1.5	9.4	15.7
22	Identifies locations of dead code	47	52	47	32	16	18.2	110.0	4.8	2.2	4.3	31.6	1.1	4.4	21.6	77
23	Identifies deeply nested if statements	25	52	47	32	16	11.9	43.6	2.9	1.4	2.6	13.9	0.9	2.0	11.5	33.9
24	Computes various popularity metrics e.g. CK, OO etc.	150	32	54	32	16	30.4	68.5	3.8	2.0	2.4	14.9	0.9	1.9	31.3	44.4

Figure: Several Candoia apps with their lines of code in different

Results Analysis: App #14 Maps modules to developers

- Software quality can be analyzed using values of [1]
  - NOE: Number of Engineers
  - EF: Component edit frequency
  - DMO: Depth of Master Ownership
  - Bugs: Number of Bugs
- App #14 Computes these matrices

[1] Basili. The influence of organizational structure on software quality, N. Nagappan et.al, ICSE'08

Results Analysis: App #14 Maps modules to developers

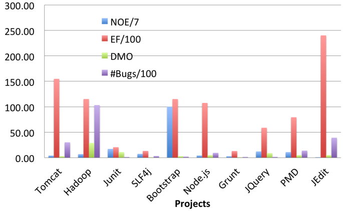


Figure: Repository Mining Tool Building

Adoptability

Claim: Candoia apps are portable across diverse project settings and require no changes while adopting for new settings
Evaluation Strategy:

- Implemted all 24 apps in Java for a project setting, keeping the tool modular and reusable
- Adopt the developed tool to new project setting
- Compared the LOC changes required in Candoia with Java to adopt a tool from a project setting to another

Project Settings Coverd by Test Projects

#	VCS	PL	Bugs
1	GIT	Java	Issues
2	SVN	Java	Bugzilla
3	GIT	Java	JIRA
4	GIT	Java	Tickets
5	SVN	Java	Tickets
6	GIT	JS	Issues

Figure: Six project settings

LOC changes in Boa v/s. Java

	#	Java						Candoia				
		M_{VCS}	M_{Bug}	M_{Forge}	M_{Mining}	$M_{Visualize}$	Total	Boa	JS	HTML	CSS	Total
Nullcheck	1	125	157	20	143	53	498	59	12	34	0	105
	2	148 (-89,+112)	117 (-119,+79)	27 (-15,+22)	156 (-43,+60)	53 (-1,+1)	501 (-267,+274)	59	12	34	0	105
	3	125 (-2,+2)	129 (-110,+82)	20 (-1,+1)	155 (-21,+33)	53 (-1,+1)	482 (-135,+118)	59	12	34	0	105
	4	125 (-2,+2)	115 (-111,+69)	20 (-1,+1)	167 (-18,+42)	53 (-1,+1)	480 (-133,+115)	59	12	34	0	105
	5	148 (-89,+112)	116 (-110,+69)	27 (-15,+22)	154 (-48,+59)	53 (-1,+1)	498 (-263,+263)	59	12	34	0	105
	6	120 (-15,+10)	157 (-1,+1)	20 (-1,+1)	147 (-13,+17)	53 (-1,+1)	497 (-31,+30)	59	12	34	0	105
File Association	1	72	139	20	138	53	422	20	12	34	0	66
	2	125 (-38,+91)	60 (-113,+34)	27 (-15,+22)	140 (-45,+47)	53 (-1,+1)	405 (-212,+195)	20	12	34	0	66
	3	72 (-1,+1)	146 (-120,+127)	20 (-1,+1)	146 (-7,+15)	53 (-1,+1)	437 (-130,+145)	20	12	34	0	66
	4	72 (-1,+1)	115 (-106,+72)	20 (-1,+1)	137 (-4,+3)	53 (-1,+1)	397 (-113,+78)	20	12	34	0	66
	5	125 (-38,+91)	95 (-96,+52)	27 (-15,+22)	133 (-30,+25)	53 (-1,+1)	433 (-180,+191)	20	12	34	0	66
	6	72 (-1,+1)	139 (-1,+1)	20 (-1,+1)	138 (-1,+1)	53 (-1,+1)	421 (-5,+5)	20	12	34	0	66
Churn Rate	1	52	0	20	69	53	194	13	33	47	0	93
	2	104 (-38,+90)	0	27 (-15,+22)	74 (-26,+31)	53 (-1,+1)	258 (-80,+144)	13	33	47	0	93
	3	52	0	20 (-1,+1)	69	53 (-1,+1)	194 (-2,+2)	13	33	47	0	93
	4	52	0	20 (-1,+1)	69	53 (-1,+1)	194 (-2,+2)	13	33	47	0	93
	5	104 (-38,+90)	0	27 (-15,+22)	74 (-26,+31)	53 (-1,+1)	258 (-80,+144)	13	33	47	0	93
	6	52	0	20 (-1,+1)	69	53 (-1,+1)	194 (-2,+2)	13	33	47	0	93
BugSrc Mapper	1	78	152	20	73	53	376	37	30	47	32	146
	2	105 (-49,+76)	79 (-118,+45)	27 (-15,+22)	74 (-41,+42)	53 (-1,+1)	338 (-224,+186)	37	30	47	32	146
	3	78 (-2,+2)	104 (-111,+63)	20 (-1,+1)	78 (-28,+33)	53 (-1,+1)	333 (-143,+100)	37	30	47	32	146
	4	78 (-2,+2)	85 (-106,+39)	20 (-1,+1)	77 (-24,+28)	53 (-1,+1)	313 (-134,+71)	37	30	47	32	146
	5	108 (-44,+74)	85 (-106,+39)	27 (-15,+22)	69 (-45,+41)	53 (-1,+1)	342 (-211,+177)	37	30	47	32	146
	6	78 (-2,+2)	152 (-1,+1)	20 (-1,+1)	78 (-28,+33)	53 (-1,+1)	381 (-33,+38)	37	30	47	32	146

Customizability

Claim: Performing customizations in Candoia requires less efforts in terms of LOC

Evaluation Criteria:

- Customize the Java and Candoia apps
- Customizations include change in Mining logic, visualization, external tool usage etc.
- Compared the LOC changes required to customize the Candoia app with Java tool

c_{10}	Shows number of nullcheck bug revisions in pie chart	c_{23}	Module association instead of file association
c_{11}	Change the output display to column chart	c_{24}	File association without bug data
c_{12}	Display nullcheck issue life time	c_{30}	Churn rate based on revisions
c_{13}	Plot nullcheck date v/s number of modified files	c_{31}	Associate bugs to churn rates
c_{14}	Maps nullcheck to developers	c_{40}	Bugs to source files mapping displayed in column chart
c_{20}	File associations using weka apriori	c_{41}	Change the output display to pie chart
c_{21}	File associations using weka fpgrowth	c_{42}	Top five files with maximum bug fix time
c_{22}	File associations using spmf eclat	c_{43}	Asssociate developers to bugs

	#	Java						Candoia				
		M_{VCS}	M_{Bug}	M_{Forge}	M_{Mining}	$M_{Visualize}$	Total	Boa	JS	HTML	CSS	Total
Nullcheck	c_{10}	125	157	20	143	53	498	59	41	45	26	171
	c_{11}	125 (-1,+1)	157 (-1,+1)	20 (-1,+1)	143 (-2,+2)	53 (-3,+3)	498 (-8,+8)	59	12	34	0	105
	c_{12}	125 (-1,+1)	137 (-29,+9)	20 (-1,+1)	144 (-14,+11)	53 (-2,+2)	479 (-47,+24)	74 (-4,+19)	41 (-2,+2)	45 (-4,+4)	26 (-1,+1)	186 (-11,+26)
	c_{13}	125 (-1,+1)	157 (-1,+1)	20 (-1,+1)	147 (-6,+11)	53 (-1,+1)	501 (-10,+15)	64 (-3,+8)	41 (-4,+4)	45 (-4,+4)	26 (-1,+1)	176 (-12,+17)
	c_{14}	125 (-1,+1)	157 (-1,+1)	20 (-1,+1)	147 (-13,+18)	53 (-1,+1)	502 (-17,+22)	61 (-4,+1)	41 (-4,+4)	45 (-4,+4)	26 (-1,+1)	173 (-13,+10)
File Assoc.	c_{20}	141	157	20	178	23	481	37	12	34	0	83
	c_{21}	141 (-1,+1)	157 (-1,+1)	20 (-1,+1)	178 (-3,+3)	23 (-1,+1)	481 (-7,+7)	37	12 (-1,+1)	34	0	83 (-1,+1)
	c_{22}	141 (-1,+1)	157 (-1,+1)	20 (-1,+1)	183 (-23,+28)	23 (-1,+1)	486 (-27,+32)	37	12 (-1,+1)	34	0	83 (-1,+1)
	c_{23}	141 (-1,+1)	157 (-1,+1)	20 (-1,+1)	178 (-3,+3)	23 (-1,+1)	461 (-8,+34)	37	12 (-1,+1)	34	0	83 (-1,+1)
	c_{24}	141 (-1,+1)	0	20 (-1,+1)	175 (-5,+2)	23 (-1,+1)	359 (-165,+5)	24 (-20,+7)	12 (-1,+1)	34	0	70 (-21,+8)
Churn	c_{30}	52	0	20	69	53	194	13	33	47	0	93
	c_{31}	72 (-1,+21)	0	20 (-1,+1)	73 (-4,+8)	53 (-1,+1)	218 (-7,+31)	42 (-4,33)	33	47	0	122 (-4,+33)
BugSrc	c_{40}	78	152	20	73	53	376	37	30	47	32	146
	c_{41}	78 (-2,+2)	152 (-2,+2)	20 (-1,+1)	73 (-1,+1)	53 (-2,+2)	376 (-8,+8)	37	38 (-28,+35)	47	32	154 (-28,+35)
	c_{42}	78 (-2,+2)	152 (-2,+2)	20 (-1,+1)	137 (-18,+82)	53 (-1,+1)	440 (-24,+88)	41 (-15,+19)	30	47	32	155 (-15,+19)
	c_{43}	78 (-2,+2)	157 (-17,+23)	20 (-1,+1)	99 (-19,+47)	53 (-1,+1)	407 (-40,+74)	46 (-2,+11)	38 (-4,+12)	47	32	163 (-6,+23)

Figure: Compares LOC changes required for a number of customizations in Java and Candoia.

Related Work

Candoia draw inspiration from rich body of work

- Platforms for reusing of tools and allow low cost addition of new tools
- Provides a repository of datasets from open-source repositories

Moose

- A platforms for reusing of data mining tools.
- Provides scripting and visualizations
- Difference lies in the focus of the tools
- Candoia is focused towards MSR and integrates MSR tools

RepoGrams

- Helps researchers gather evaluation targets and calculate metrices on the target project

Kenyon and Sourcerer

- Defiens a databse schema for metadata and source code
- Provide access to the dataset via SQL

Alitheia Core

- Provide a highly extensible framework for analyzing software product
- Process metrics on a large database of open source projects source

FLOSSMole

- Analysis on the project metadata

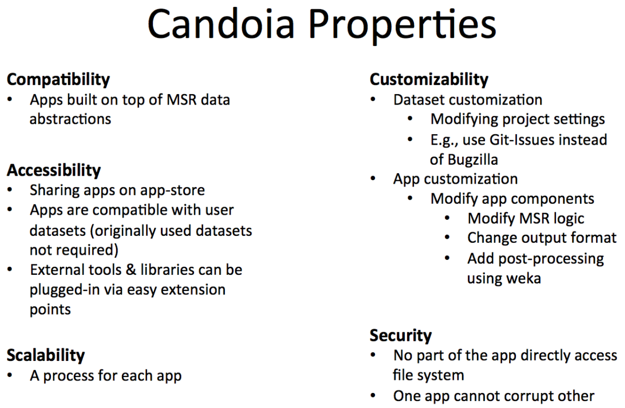
Groundhog

- Infrastructure for downloading and analysing projects from SourceForge

GHTorrent, PROMISE Repository, SourcererDB and Boa

- Provides standard dataset for evaluation
- SourceDB also provides means to create custom dataset

Focus is on lifting the burden of data curation from user



Candoia Properties

Compatibility

- Apps built on top of MSR data abstractions

Accessibility

- Sharing apps on app-store
- Apps are compatible with user datasets (originally used datasets not required)
- External tools & libraries can be plugged-in via easy extension points

Scalability

- A process for each app

Customizability

- Dataset customization
  - Modifying project settings
  - E.g., use Git-Issues instead of Bugzilla
- App customization
  - Modify app components
    - Modify MSR logic
    - Change output format
    - Add post-processing using weka

Security

- No part of the app directly access file system
- One app cannot corrupt other

Future Work

- Adding new software tools and technologies
- Building addiotional useful tools
- Utilizing underlying GPUs for better performance

Acknowledgement

- This work was supported in part by the US National Science Foundation under grants CCF-15-18897, CNS-15-13263, and CCF-14-23370.
- Dalton D. Mills and Trey Erenberger for helping with Candoia frontend implementation
- Eric Lin for implementing several Candoia apps
- Dr. Robert Dyer for valuable feedback

Candoia was awawrded distinguished poster[2] at ICSE’16

Full version of the work is appearing at MSR’17 [3]

[2] *Tiwari, Nitin M., Ganesha Upadhyaya, and Hridesh Rajan. "Candoia: a platform and ecosystem for mining software repositories tools.", ICSE’16*

[3] *Tiwari, Nitin M., Ganesha Upadhyaya, Dr. Hoan Anh Nguyen and Hridesh Rajan. "Candoia: A Platform for Building and Sharing Mining Software Repositories Tools as Apps", MSR’17*

Thank you