Data Collection from Open Source Software Repositories

GORAN MAUŠA, TIHANA GALINAC GRBAC

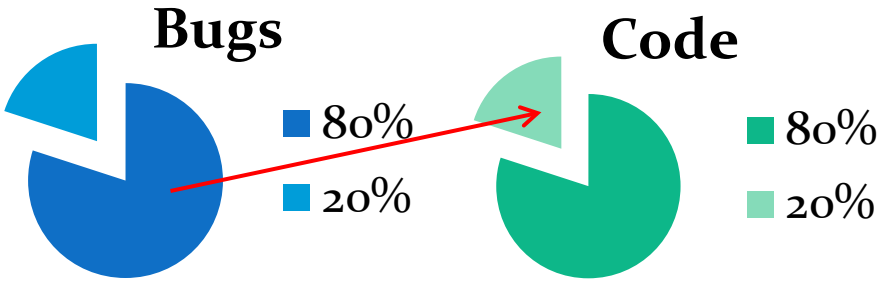
SEIP LABORATORY

FACULTY OF ENGINEERING

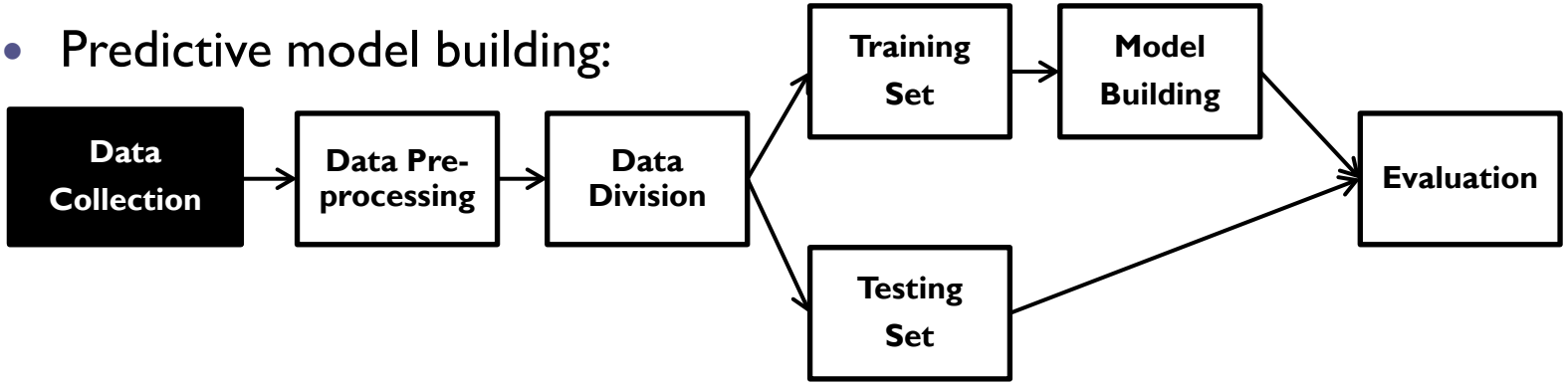
UNIVERSITY OF RIJEKA, CROATIA

Software Defect Prediction (SDP)

- Aim:
  - Focus testing effort to software units with higher fault-proneness probability
- Motivation:
  - High testing costs (80% after release)
  - Pareto principle can be applied



- SDP approach:
  - Classification based on parameters of size and complexity
- Predictive model building:



Data Collection for SDP

- Motivation:
  - The context of project development may influence SDP performance
  - Small number of available datasets => inability to study the context influence

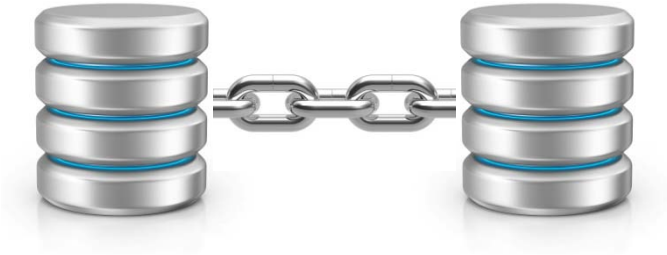
- Problem:
  - Lack of systematic data collection approach
  - Data collection is time consuming and not trivial

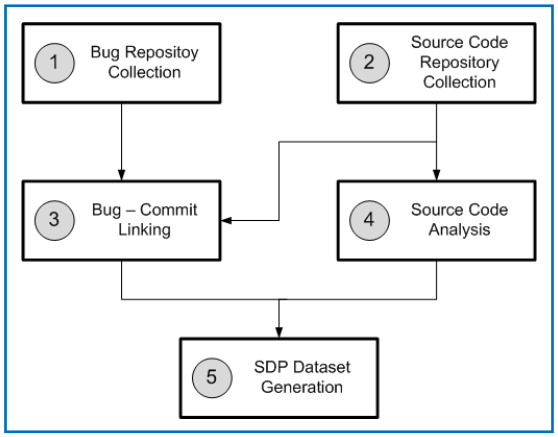
- Potential Source of data:
  1. Industrial (large telecom. software)
     - Rarely available
  2. Open repositories (PROMISE gives NASA datasets)
     - Impossible to validate (missing data collection procedure and source code)
     - Often suffer from: missing values, outliers, duplicated entries, unbalance,...
  3. Open source projects (Eclipse, Mozilla, Apache)
     - Increasingly popular, easily validated, expandable,

Open Source Software Repositories

Linking 2 repositories :

- Source code management & bug tracking
- Structured and unstructured data
- Problem: there is no formal link
- Consequence: different approach -» data bias





Important characteristics :

- **Bug status:** (closed / opened)
- **Bug resolution:** (fixed / otherwise)
- **Bugs severity:**

(blocker - normal / +trivial / +enhancement)

- **Repository search order:**

(start with bugs / source code changes)

- **Declaration of defect-free units**

(all the unlinked units / unlinked & unchanged)

Data Collection for SDP

Linking Techniques :

- Simple search
- Regular expression search
- Authorship correspondence
- Time correlation
- Advanced NLP techniques (ReLink)

Bug tracking repository		Source code repository
Bug ID	→	Commit message
Bug assignee	→	Commit author
Bug closed	→	Commit date
Release	→	Release tag
Bug opened	→	Release date
Comments	→	

Issues :

- **Granularity level** (package / file / class / method)
- **Software metrics** (product / development & process / usage)
- **Bug – File cardinality** (many – to – many)
- **Bug – File duplicated links**
- **Bug ID varying length**

Bug – Code (BuCo) Analyzer Tool [SoftCOM 2014]

Tool developed through :

- **Systematic literature review**

(36 papers from [1] + 35 / 136 / 4447)

- **Exploratory study**

(12 students, observer triangulation, 5 projects, 4 exercises, 5 data forms, 52 tasks)

- **Software product metrics tools review**

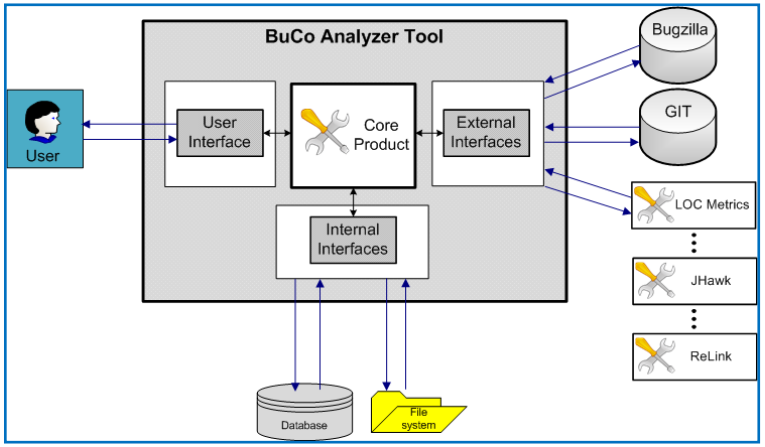
(iterative review 35 / 19 / 5 / 2 tools)

- **Iterative development**

(30 students - 13 groups)

- **Systematic comparison of techniques**

(7 techniques, 5 projects, 37 releases)



Tool properties :

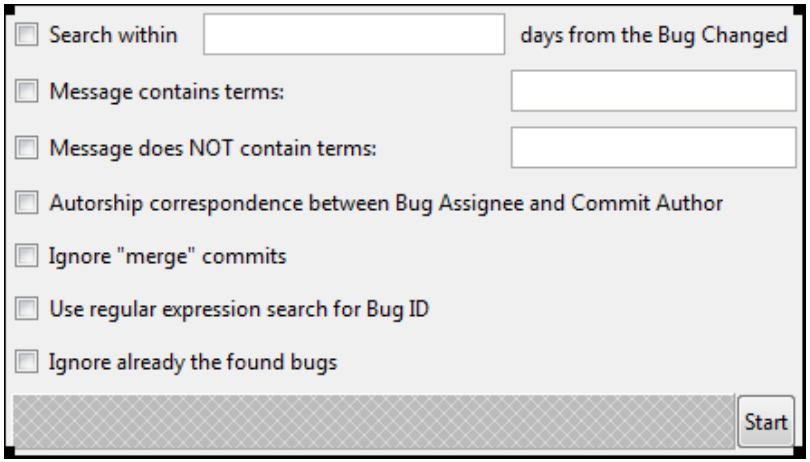
- **Automatic data collection**
- **Simple interface**
- **6 bug-code linking techniques**
- **Calculation of 50 product metrics**
- **Bug counting**
- **Report generation**

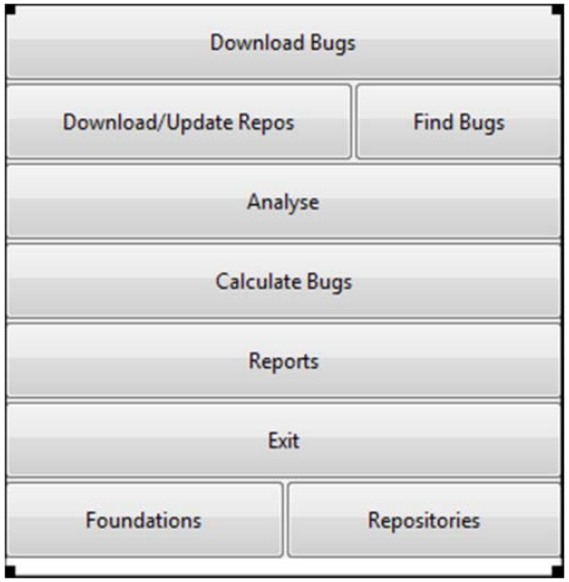
[1] Hall T, Beecham S, Bowes D, Gray D, Counsell S: A systematic literature review on fault prediction performance in software engineering, IEEE Trans Softw Eng 38(6), pp.1276-1304, 2012

Bug – Code (BuCo) Analyzer Tool [SoftCOM 2014]

Tool offers:

- Bug download from Bugzilla of Eclipse, Apache and Mozilla communities
- SCM download from GIT
- Bug-Code linking techniques:

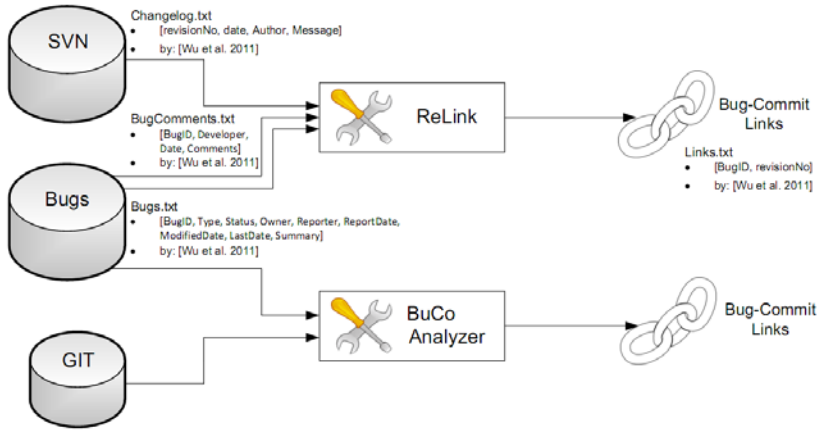




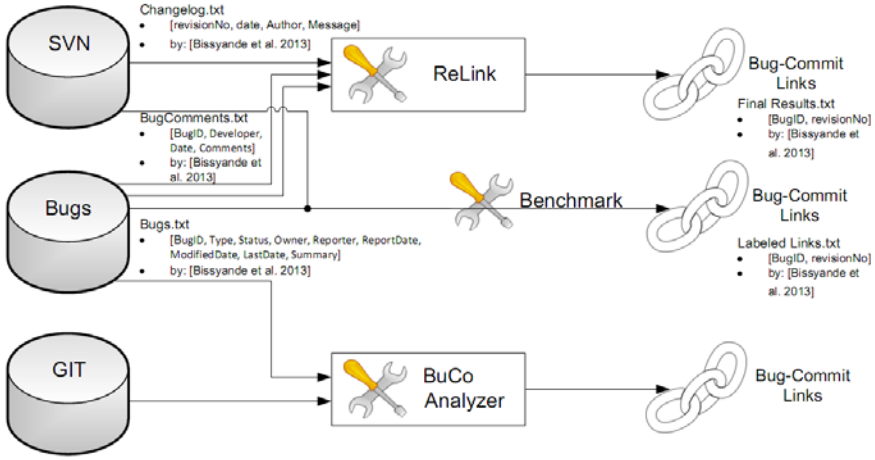
- Automatic calculation of product metrics
- Generate reports

Bug – Code Linking Techniques [SQAMIA 2014]

- Analysis 1 :
  - Comparison: Simple search & ReLink
  - Aim: define Regex Search
  - Project: Apache HTTPD
  - Source: ReLink data, GIT repository



- Analyses 2 & 3 :
  - Comparison: Regex search & ReLink
  - Aim: benchmark evaluation
  - Projects: Apache HTTPD, OpenNLP
  - Source: ReLink & Benchmark data, GIT



Bug – Code Linking Techniques - Results [SQAMIA 2014]

- Analysis 1 – results :
  - Unequal input & linking output:

Analysis	Input			Linking Method	Output			
	Source	Commits	Bugs		Links	Commits	Files	Bugs
1	SVN + Bugs by Relink	43867	673	ReLink	1014	957	1061	673
	GIT + Bugs by Relink	26287	673	Simple search	598	556	993	598

  - Manual investigation revealed:

Equal Links	443	74.1%
Bugs with one link	196	32.8%
Bugs with multiple links	247	41.3%
Different Links	147	24.6%
Incorrect links	120	20.1%
Potentially correct links	27	4.5%
Links From Different Repository	8	1.3%

  - Regular expression: $'(.*[\hat{}0-9]|\hat{})' + bug\_id + '\ (\backslash W|\backslash\backslash r|\$)'$

Bug – Code Linking Techniques - Results [SQAMIA 2014]

- Analyses 2 & 3 – results :
  - OpenNLP – benchmark dataset (equal input), different linking output:

Analysis	Input			Linking Method	Output			
	Source	Commits	Bugs		Links	Commits	Files	Bugs
2	SVN + Bugs by Relink	43867	673	ReLink	1014	957	1061	673
	GIT + Bugs by Relink	26287	673	Regular Expression	703	664	495	621
3	SVN + Bugs by benchmark	847	100	Benchmark	127	125	141	81
	SVN + Bugs by benchmark	847	100	ReLink	115	113	132	76
	GIT + Bugs by benchmark	847	100	Regular Expression	128	126	141	81

  - Manual investigation – **REGEX :**

Bug ID	Commit **Message**	Opened	**Closed**	**Commited**	Bug **Assignee**	**Commiter**
9	OPENNLP-190 Updated to Apache **9** parent pom and removed special version which we needed for the Apache 8 parent pom, namely for the rat plugin and the release plugin.	9.12.2010	13.1.2011	30.5.2011	William Colen	Joern Kottmann

  - Manual investigation – **ReLink :**

Bug ID	Commit **Message**	Opened	**Closed**	**Commited**	Bug **Assignee**	**Commiter**
84	OPENNLP-**84** Corrected method name to sentPosDetect	25.1.2011	25.1.2011	**25.1.2011**	Joern Kottmann	**joern**
115	OPENNLP-**115** Charset should be specified before creating input stream	1.2.2011	11.7.2011	**11.7.2011**	Joern Kottmann	**joern**
471	OPENNLP-**471**: found after we find a name match, we don't jump over the found name but re-process... thanks William for pointing this out	14.3.2012	24.4.2012	19.3.2012	James Kosin	**jkosin**

Bug – Code Linking Techniques - Conclusion [SQAMIA 2014]

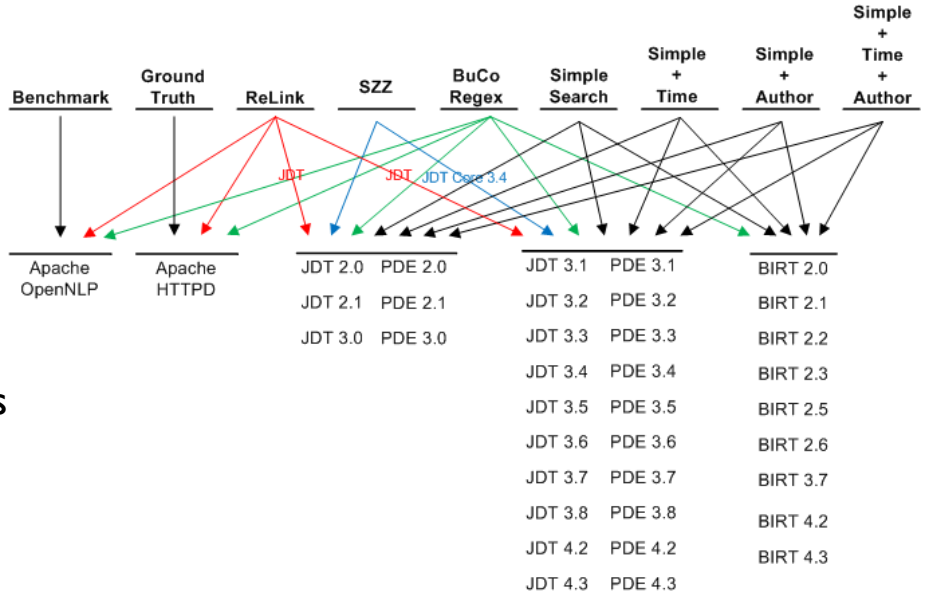
- The generalization of research requires:
  - Datasets from various domains
  - Systematic procedure with limited bias
- Bug – Code linking
  - Proven to be prone to bias
  - Complex technique outperformed by regular expression search
- Future research
  - Compare the whole data collection process approaches
  - Analyze the environment influence to bug-code linking

Current Research

- Developing a systematic data collection procedure for SDP
- Comparison of different linking techniques on various environments:

	JDT	PDE	BIRT	HTTPD	OpenNLP
Releases	13	13	9	1	1
Files	18,752	6,829	8,104	3,744	1,784
Bugs	198,206	42,582	65,173	673	100
Domain	Development Tools	Development Environment	Business Intelligence	Web Server	NLP - Language Processing

- Comparison of the most popular SZZ approach [2] to our own
  - Interactions between different techniques, approaches and datasets used in our experiment



[2] SZZ: "When do changes induce fixes?", SIGSOFT Softw Eng Notes, 2005

Thank you for you attention!

Question?