UMONS Université de Mons

inforTech UMONS RESEARCH INSTITUTE FOR INFORMATION TECHNOLOGY AND COMPUTER SCIENCE

complexys UMONS RESEARCH INSTITUTE FOR COMPLEX SYSTEMS

Evolving Software Ecosystems

Tom Mens, Mathieu Goeminne
Software Engineering Lab
University of Mons

informatique.umons.ac.be/genlog

Current Research Interests

- software evolution
- software quality
- model-driven software engineering
- empirical software engineering
- human-machine interaction

- Use of formal techniques to support the above
  - graph transformation
  - logic-based formalisms
  - model checking
  - statistical analysis
- Develop automated tool support

Ongoing Research Projects

ARC Project « Ecological Studies of Open Source Software Ecoysystems », 2012-2017

- Interdisciplinary research, using ideas from biological ecology to understand and improve evolution/maintenance of software ecosystems

FRFC Project « Data-Intensive Software System Evolution », 2013-2017

- Collaboration with University of Namur (A. Cleve)
- Empirical study of co-evolution of programs and database in data-intensive software systems

COST Action IC1404 « Multi-Paradigm Modelling for Cyber-Physical Systems », 2015-2018

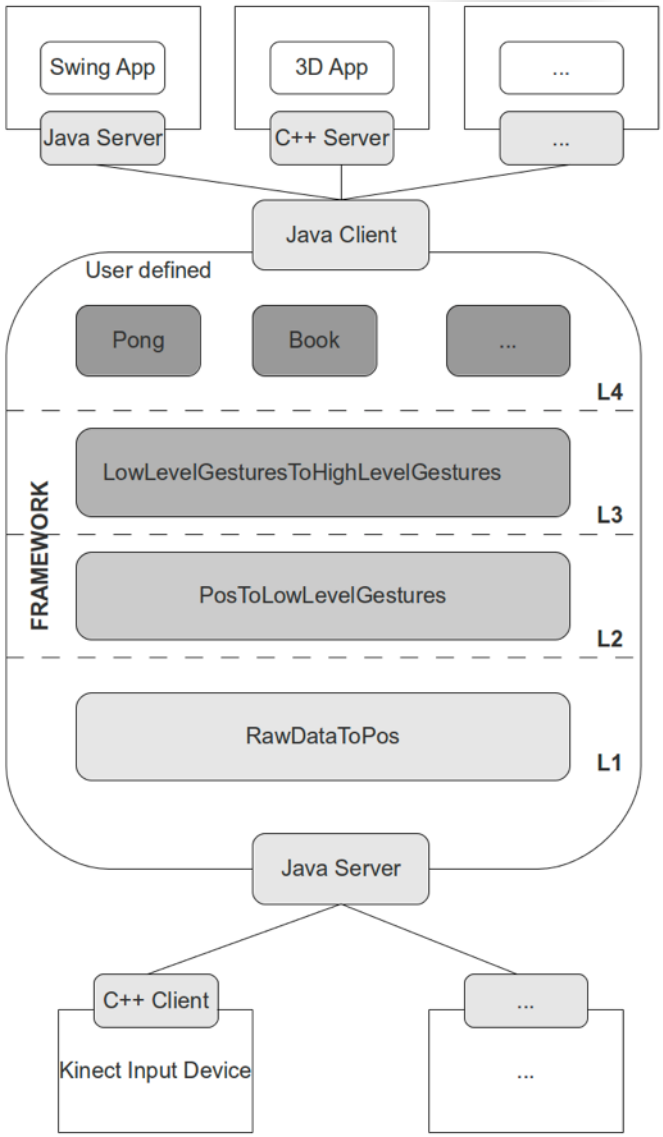
FRIA PhD Scholarship « Executable Modeling of Gestural Interaction Applications », 2011-2015

- *Romuald Deshayes.* Using domain-specific modeling languages, model transformation, executable modelling through high-level Petri nets

Human-machine interaction

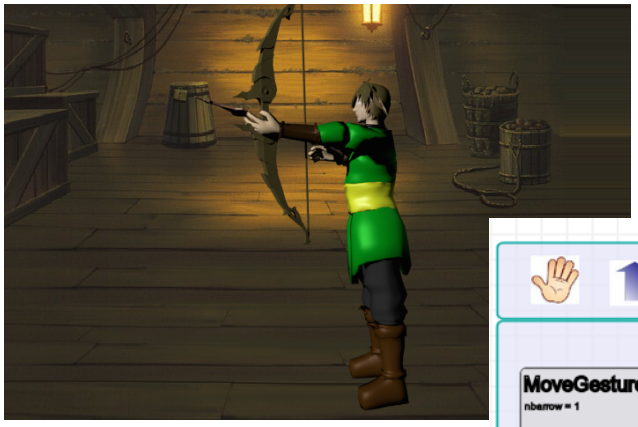
Executable Modeling of Gestural Interaction Applications

Controlling interactive applications
(e.g. games, domotics, …)
using hand gestures

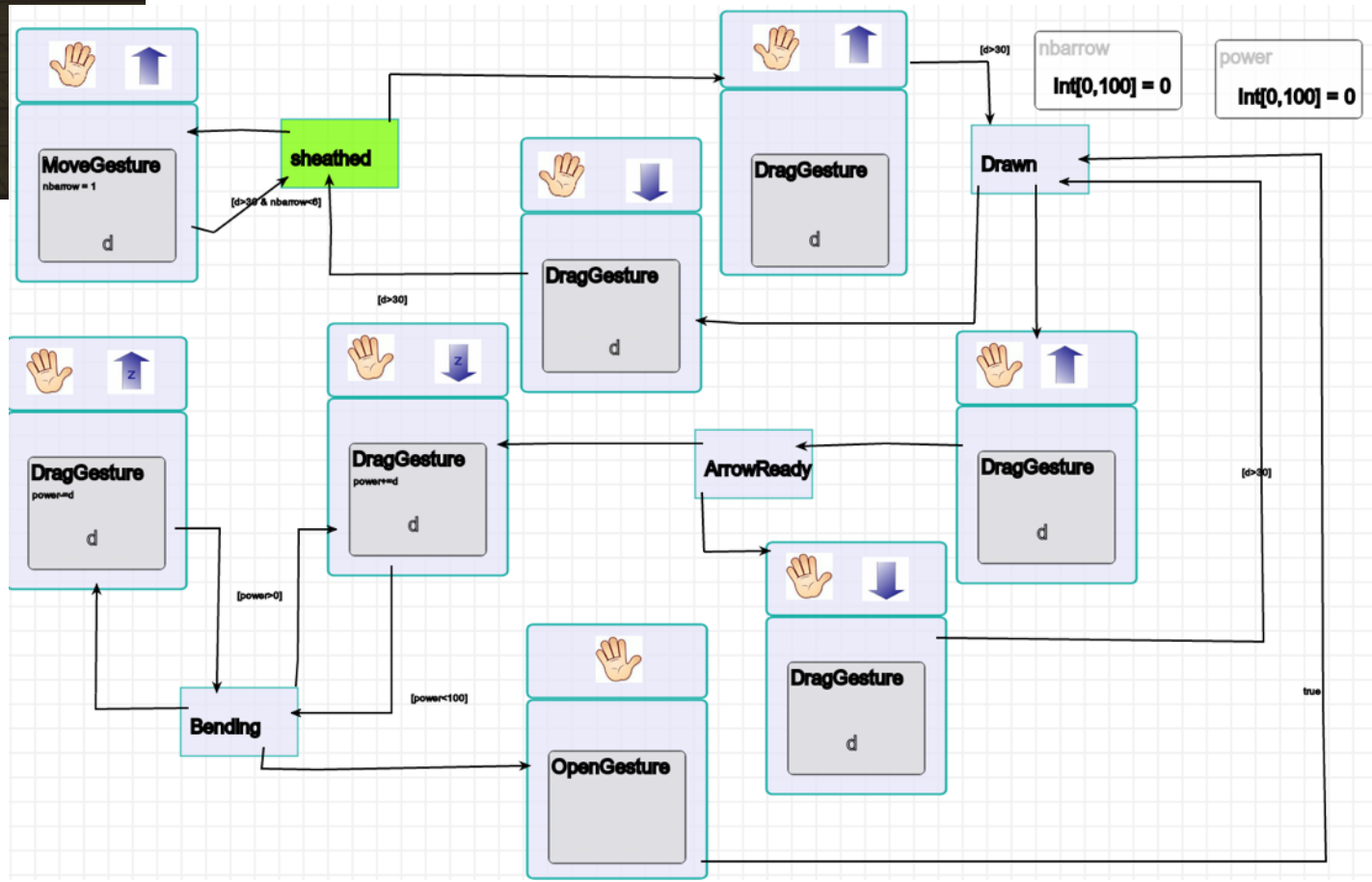


Human-machine interaction

Executable Modeling of Gestural Interaction Applications



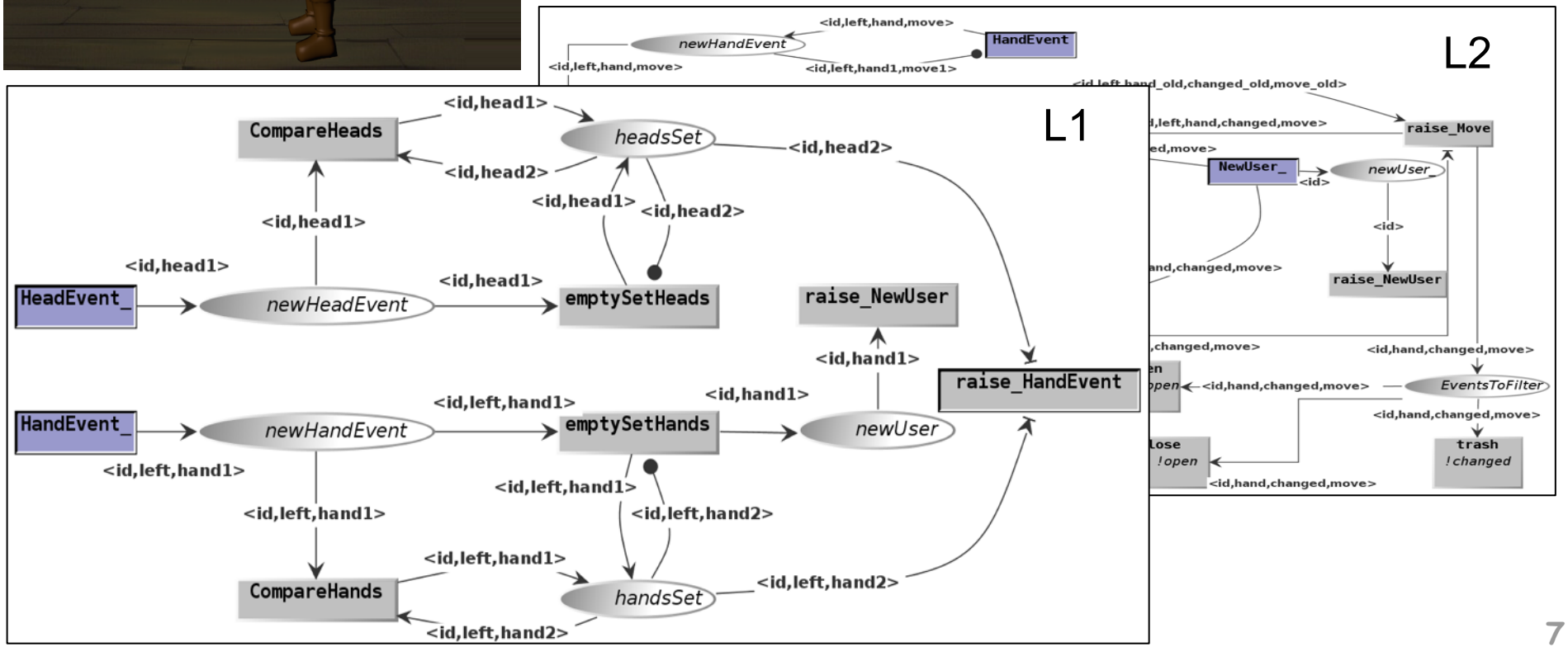
Using an executable, domain-specific, modelling language



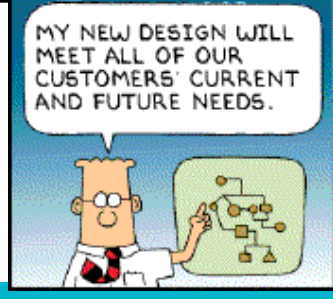
Human-machine interaction

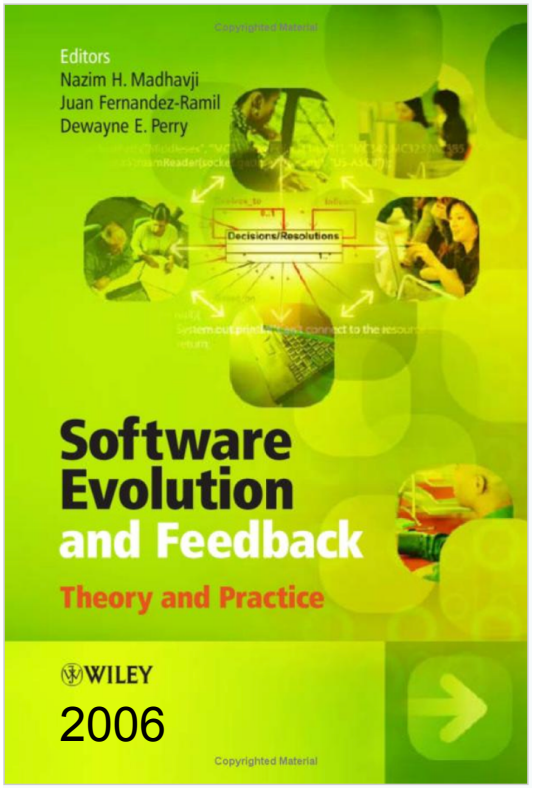
Executable Modeling of Gestural Interaction Applications

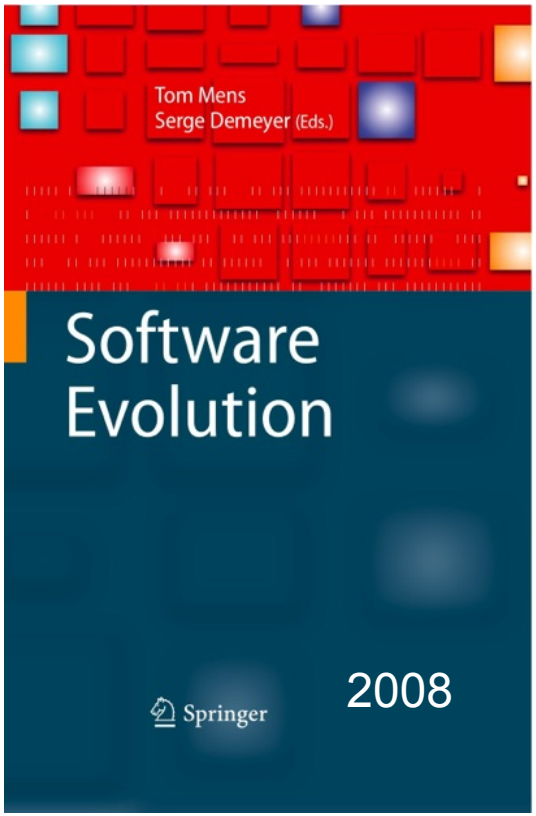
Executable operational semantics
based on high-level Petri nets
(ICO models, IRIT, Toulouse)

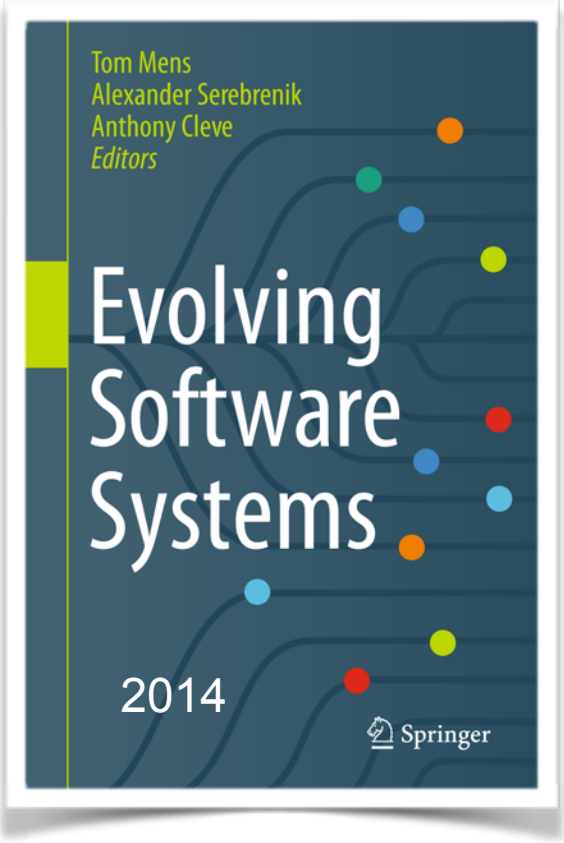


Research Context
Software Evolution









Research Context
Software Ecosystems

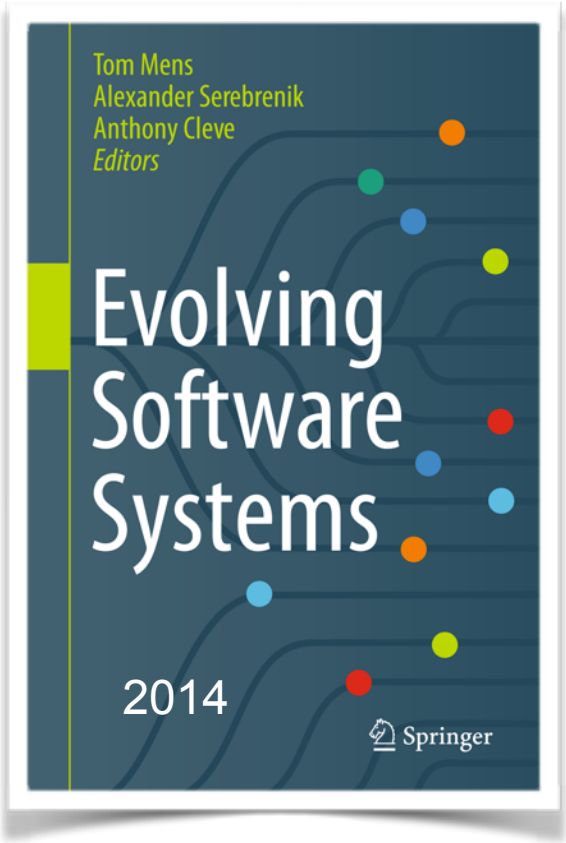
Chapter 10

Studying Evolving Software Ecosystems based on Ecological Models

Tom Mens, Maëlick Claes, Philippe Grosjean and Alexander Serebrenik

Research on software evolution is very active, but evolutionary principles, models and theories that properly explain why and how software systems evolve over time are still lacking. Similarly, more empirical research is needed to understand how different software projects co-exist and co-evolve, and how contributors collaborate within their encompassing software ecosystem.

In this chapter, we explore the differences and analogies between natural ecosystems and biological evolution on the one hand, and software ecosystems and software evolution on the other hand. The aim is to learn from research in ecology to advance the understanding of evolving software ecosystems. Ultimately, we wish to use such knowledge to derive diagnostic tools aiming to analyse and optimise the fitness of software projects in their environment, and to help software project communities in managing their projects better.



Research Context
Software Ecosystems

- Study of **macro-level** software evolution
  - evolution of **large** collections or distributions of software projects or packages
  - E.g. forges like GitHub, SourceForge, Savannah, Google Code

J.M. Gonzalez-Barahona et al. *Macro-level software evolution: a case study of a large software compilation.* Empirical Software Engineering 14(3): 262-285 (2009)

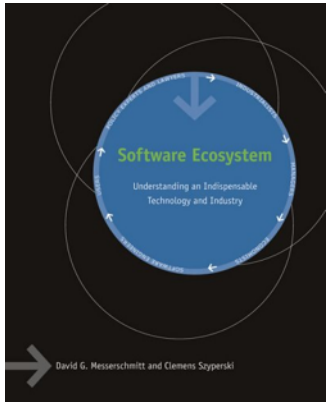
M. Caneill, S. Zacchiroli. *Debsources: Live and historical views on macro-level software evolution*. Int. Symp. ESEM 2014

Research Context
Software Ecosystems

- Study of **macro-level** software evolution
  - evolution of **large** collections or distributions of software projects or packages
  - E.g. forges like GitHub, SourceForge, Savannah, Google Code
- Study **socio-technical aspects** of the community of contributors (end-users, developers, debuggers, …)
- Focus on coherent collections of projects or packages
  - a.k.a. **software ecosystems**
  - E.g. Debian, Ubuntu, Gnome, KDE, CRAN, Eclipse, …

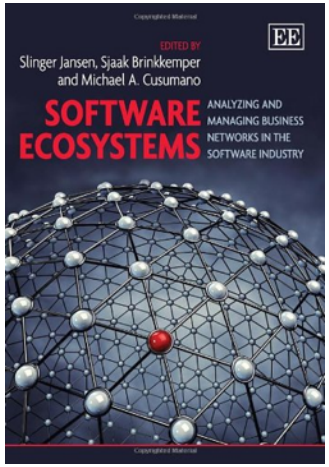
Software Ecosystems Definitions

- David Messerschmitt & Clemens Szyperski, 2003
  - *"a collection of software products that have some given degree of symbiotic relationships."*



- Mircea Lungu, 2008 [Ph.D. dissertation]
  - *"a collection of software projects that are developed and evolve together in the same environment."*

- Slinger Jansen et al., 2013
  - *"a set of actors functioning as a unit and interacting with a shared market for software and services, together with the relationships among them."*



Research Context

- Focus on **open source** software
- Free access to source code, defect data, developer and user communication
- Historical data available in open repositories
  - Observable communities
  - Observable activities
- Increasing popularity for personal and commercial use
- A huge range of community and software sizes

Long-term goals

- Determine and improve the factors that drive *success* or *failure* of OSS projects within their *ecosystem*

- Investigate new techniques and mechanisms to predict and improve *quality* and *survival* of OSS projects
  - Inspired by research in *ecology, social network analysis, …*

informatique.umons.ac.be/genlog/projects/ecos

Research Questions

- Specific questions depend on the software ecosystem under study

CRAN

Debian

Gnome

Research Questions

- Specific questions for **CRAN**
  - R package archive network
    - *Which packages are more likely to cause, upon update, problems in dependent packages?*
    - *When and why is code cloned across packages?*

Daniel German, Bram Adams et al.
"The Evolution of the R Software Ecosystem", CSMR 2013

Maelick Claes, Tom Mens, Philippe Grosjean
"On the Maintainability of CRAN Packages", CSMR-WCRE 2014
"maintaineR: web-based dashboard for maintainers of CRAN packages", ICSME 2014
"An empirical study of identical function clones in CRAN" [In preparation]

Research Questions

- **CRAN** (R package archive)
  - R package description file format:

```
Package: pkgname
Version: 0.5-1
Date: 2004-01-01
Title: My First Collection of Functions
Authors@R: c(person("Joe", "Developer", role = c("aut", "cre"),
              email = "Joe.Developer@some.domain.net"),
           person("Pat", "Developer", role = "aut"),
           person("A.", "User", role = "ctb",
              email = "A.User@whereever.net"))
Author: Joe Developer and Pat Developer, with contributions from A. User
Maintainer: Joe Developer <Joe.Developer@some.domain.net>
Depends: R (>= 1.8.0), nlme
Suggests: MASS
Description: A short (one paragraph) description of what
  the package does and why it may be useful.
License: GPL (>= 2)
URL: http://www.r-project.org, http://www.another.url
BugReports: http://pkgname.bugtracker.url
```

Research Questions

Firefox ▾ | CRAN Package Check Results

cran.r-project.org/web/checks/check_summary.html

Package	Version	r-devel Linux x86_64 (Debian Clang)	r-devel Linux x86_64 (Debian GCC)	r-devel Linux x86_64 (Fedora Clang)	r-devel Linux x86_64 (Fedora GCC)	r-devel OS X x86_64	r-devel OS X x86_64	r-devel Windows ix86+x86_64	r-patched Linux x86_64	r-patched Solaris sparc	r-patched Solaris x86	r-release Linux ix86	r-release Linux x86_64	r-release MacOS X x86_64	r-release Windows ix86+x86_64	r-oldrel Windows ix86+x86_64	Maintainer
A3	0.9.2	NOTE	NOTE	NOTE	NOTE	NOTE		NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	OK	Scott Fortmann-Roe
abc	1.8	NOTE	NOTE	NOTE*	NOTE*	NOTE*	NOTE	NOTE*	NOTE	NOTE*	NOTE	NOTE	NOTE	NOTE	NOTE*	WARN*	Michael Blum
abcdeFBA	0.4	NOTE	NOTE	NOTE	NOTE	NOTE	OK	NOTE	NOTE	NOTE	NOTE	OK	OK	OK	OK	OK	Abhilash Gangadharan
ABCExtremes	1.0	NOTE	NOTE	NOTE	NOTE	NOTE		NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	OK	
ABCoptim	0.13.11	OK	OK	OK	OK	OK	WARN	OK	OK	OK	OK	OK	OK	OK	OK	OK	George Vega Yon
ABCp2	1.1	NOTE	NOTE	NOTE	NOTE	NOTE		NOTE	NOTE	OK	NOTE	OK	OK	OK	OK	OK	M. Catherine Duryea
abctools	0.2-2	NOTE	NOTE	NOTE	NOTE	NOTE	OK	NOTE	NOTE	NOTE	NOTE	OK	OK	OK	OK	OK	Matt Nunes
abd	0.2-5	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	Kevin M. Middleton
abf2	0.7-0	OK	OK	OK	OK	OK		OK	OK	OK	OK	OK	OK	OK	OK	OK	Matthew Caldwell
abind	1.4-0	OK	OK	NOTE	NOTE	OK		OK	OK	OK	OK	OK	OK	OK	OK	OK	Tony Plate
abn	0.83	NOTE	NOTE	NOTE	NOTE	NOTE		NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	Fraser Lewis
abundant	1.0	NOTE	NOTE	NOTE	NOTE	NOTE	OK	NOTE	NOTE	OK	NOTE	OK	OK	OK	OK	OK	Adam J. Rothman
accelerometry	2.0	NOTE	NOTE	NOTE	NOTE	NOTE	OK	NOTE	NOTE	OK	NOTE	OK	OK	OK	OK	OK	Dane R. Van Domelen
AcceptanceSampling	1.0-3	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	WARN	Andreas Kiermeier
ACCLMA	1.0	NOTE	NOTE	NOTE	NOTE	NOTE		NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	Tal Carmi
accrued	1.0	OK	OK	OK	OK	OK		OK	OK	OK	OK	OK	OK	OK	OK		Julie Eaton
ACD	1.5.3	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	Fabio Mathias Correa
Ace	0.0.8	OK	OK	NOTE	NOTE	OK		OK	OK	OK	OK	OK	OK	OK	OK	OK	Brian Claggett
acepack	1.3-3.3	OK	OK	NOTE	NOTE	OK		OK	OK	OK	OK	OK	OK	OK	OK	OK	Jonathan Baron
acer	0.1.2	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	OK	OK	OK	OK	OK	Even Haug
aCGH.Spline	2.2	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	OK	Tom Fitzgerald
acm4r	1.0	NOTE	NOTE	NOTE	NOTE	NOTE	OK	NOTE	NOTE	OK	NOTE	OK	OK	OK	OK	OK	Andrea Benedetti
ACNE	0.7.0	OK	OK	OK	OK	OK	ERROR	OK	OK	OK	OK	OK	OK	OK	OK	OK	Henrik Bengtsson
acopula	0.9.2	OK	OK	NOTE	NOTE	OK		OK	OK	OK	OK	OK	OK	OK	OK	OK	Tomas Bacigal
aCRM	0.1.0	OK	OK	NOTE	NOTE	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	Michel Ballings
acs	1.2	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	Ezra Haber Glenn
ACTCD	1.0-0	OK	OK	OK	OK	OK		OK	OK	OK	OK	OK	OK	OK	OK	OK	Wenchao Ma
Actigraphy	1.2	NOTE	NOTE	NOTE	NOTE	NOTE		NOTE	NOTE	NOTE	NOTE	OK	OK	OK	OK	OK	Berkley Shands
actuar	1.1-6	NOTE	NOTE	NOTE	NOTE	NOTE		NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	Vincent Goulet
ActuDistns	3.0	NOTE	NOTE	NOTE	NOTE	NOTE		NOTE	NOTE	NOTE	NOTE	OK	OK	OK	OK	OK	Saralees Nadarajah
ada	2.0-3	OK	OK	NOTE	NOTE	OK		OK	OK	OK	OK	OK	OK	OK	OK	OK	Mark Culp
adabag	3.2	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	Esteban Alfaro
adagio	0.5.9	OK	OK	OK	OK	OK		OK	OK	OK	OK	OK	OK	OK	OK	OK	Hans W. Borchers
AdapEnetClass	1.0	NOTE	NOTE	NOTE	NOTE	NOTE	OK	NOTE	NOTE	NOTE	NOTE	OK	OK	OK	OK	OK	Hasinur Rahaman Khan
AdaptFit	0.2-2	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	Tatyana Krivobokova
AdaptFitOS	0.45	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	OK	Manuel Wiesenfarth
AdaptiveSparsity	1.3															OK	Kristen Zygmunt
AdaptiveSparsity	1.4	NOTE	NOTE	NOTE	NOTE	NOTE	OK	NOTE	NOTE	OK	NOTE	NOTE	NOTE	OK	NOTE		Kristen Zygmunt
adaptivetau	1.1-1	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	WARN	Philip Johnson
adaptMCMC	1.1	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	NOTE	OK	Andreas Scheidegger

Research Questions

- Tool support for CRAN packages: maintaineR
  - package dependency, conflict and clone analysis

Summary | History | Dependency list | Dependency graph | Namespace | Clones

Minimum clone AST size: 10

Minimum clone LOC: 3

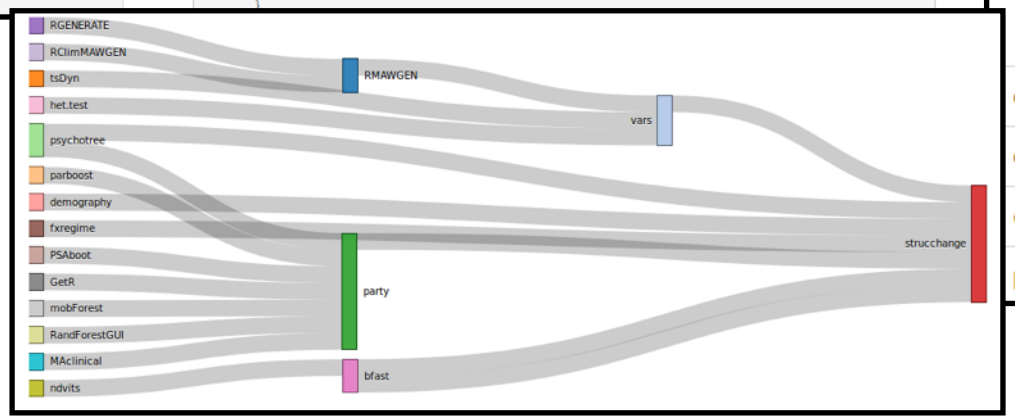
Sort packages by:
- (•) Oldest first
- ( ) Alphabetical

Show only:
- [ ] Last CRAN version

Functions	Size	LOC	Packages	Hash
	1348	214	aplpack_1.2.7 aplpack_1.2.9	da3f45d4292b49364d55f9c251285d7c

```
{
    if (missing(x))
        return("bagplot, version 2012/12/05, peter wolf")
    if (transparency == TRUE) {
        col.loophull = paste(col.loophull, "99", sep = "")
        col.baghull = paste(col.baghull, "99", sep = "")
    }
    win <- function(dx, dy) {
        atan2(y = dy, x = dx)
```

Name	Type	Conflicts
abc	function	gvcm.cat_1.6 forams_2.0-4 pomp_0.49-2
cv4abc	function	None
cv4postpr	function	None
expected.deviance	function	None
postpr	function	None



Research Questions

- Specific questions for **Debian**
  - Open source Linux distribution
    - *Which packages are more likely to cause future co-installation (CI) conflicts with other packages?*
    - *Can I upgrade a set of installed Debian packages without "breaking" my installation?*
      - Based on a formalisation and SAT solving
      - Automated tooling coinst.irill.org

Jerome Vouillon and Roberto Di Cosmo
"Broken Sets in Sotware Repository Evolution", ICSE 2013

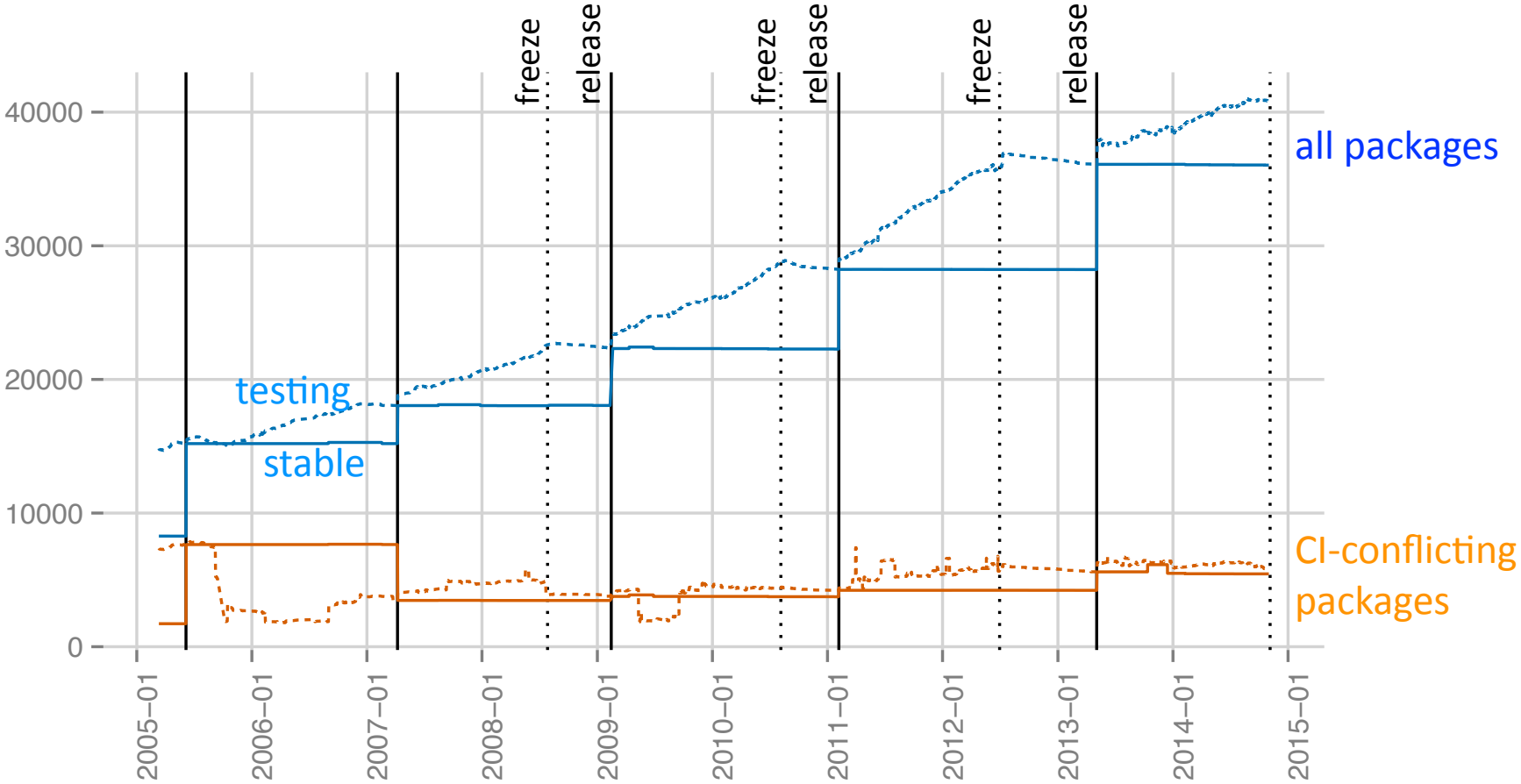
Research Questions

- Specific questions for **Debian**
  - Open source Linux distribution
    - *Which packages are more likely to cause future co-installation (CI) conflicts with other packages?*
    - *Can I upgrade a set of installed Debian packages without "breaking" my installation?*
      - Based on a formalisation and SAT solving
      - Automated tooling coinst.irill.org
    - *How do CI conflicts evolve over time?*

Maelick Claes, Tom Mens, Roberto Di Cosmo
"A historical analysis of Debian package co-installability conflicts", [Submitted]

Research Questions

Debian historical evolution of package CI-conflicts (*testing* and *stable* distribution)



Research Questions

Debian historical evolution of package CI-conflicts

Some results

- Ratio of CI-conflicting packages remains constant over time
- Occasional “jumps” correspond to introduction or removal of problematic packages that spread the problem to (in)direct depending packages
- The more often a package is CI-conflicting, the shorter it tends to live
- The longer it takes for a package to become CI-conflicting, the longer it tends to live
- The most likely causes of introduction or removal of CI-conflicts are the introduction or removal of declared conflicts in Debian package control files

Research Questions

- Specific questions for **GNOME**
  - Linux desktop environment
    - *Which projects have a higher chance of survival?*
    - *How is workload distributed over different projects/contributors?*
    - *What is the "bus factor" risk? Who are the top contributors (for a specific activity type)?*

B. Vasilescu, A. Serebrenik, M. Goeminne, T. Mens
"On the variation and specialisation of workload:
A case study of the GNOME ecosystem community"
Empirical Software Engingeering journal, 2014.

Research Questions

- Gnome visualisation tool support
  - *E.g. Complicity* (Neu et al., University of Lugano)

Data Extraction

Version control repositories store source code and other commits

- E.g., Subversion, Git

Mailing lists for communication between developers and users

Issue tracking systems for recording bug reports and change requests

- E.g., Bugzilla, JIRA

Question and Answer websites

- E.g. StackOverflow

Data Extraction

- Using open source MetricsGrimoire tool suite (https://github.com/MetricsGrimoire)

CVSAnalY

- extracts information from SVN or Git source code repository logs and stores it into relational database

MailingListStats

- extracts mailing list information from mbox format

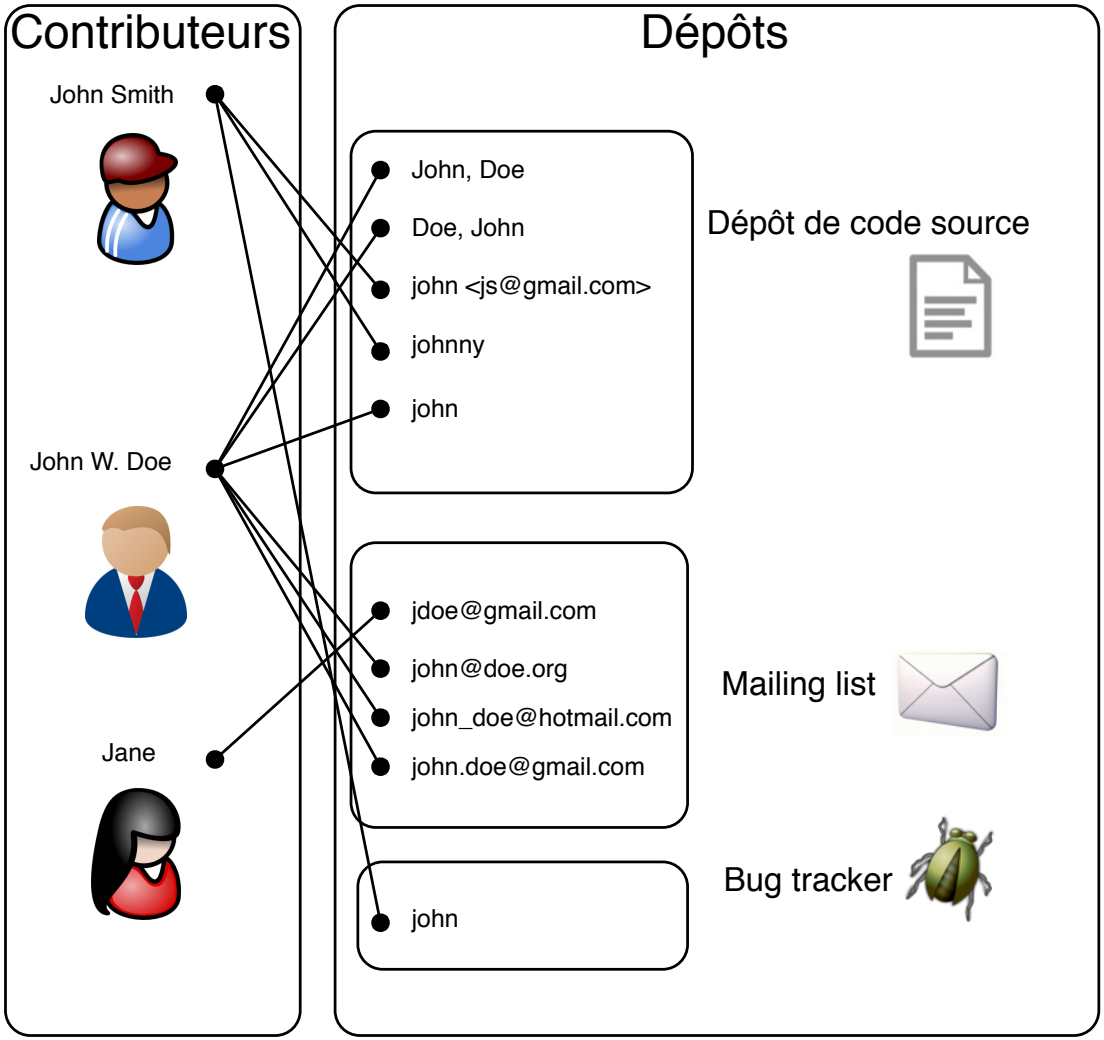
Bicho

- extracts information from issue tracking systems such as Bugzilla and JIRA

Identity merging

- The same contributor may use different aliases

Identity merging



Ordering	Rajesh Sola	Sola Rajesh
Spelling: misspelling, diacritics, punctuation	Rene Engelhard	Fene Engelhard
	Démurget	Demurget
	J. A. M. Carneiro	J A M Carneiro
Middle initials, patronyms, nicknames, additional surnames, incomplete names	Daniel M. Mueth	Daniel Mueth
	Alexander Alexandrov Shopov	Alexander Shopov
	Carlos Garnacho Parro	Carlos Garnacho
	Jacob “Ulysses” Berkman	Jacob Berkman
	A S Alam	Amanpreet Singh Alam
Name variants: transliteration, diminutives	Γιωργοσ	Georgios
	Mike Gratton	Michael Gratton
Software-specific: usernames, projects, tooling artefacts	mrhappypants	Aaron Brown
	Arturo Tena/libole2	Arturo Tena
	(16:06) Alex Roberts	Alex Roberts
Mix	Any combination of those	

Identity merging

Semi-automatic approach:

- eliminate specific quirks observed during extraction
  Example: “(16:06) Alex Roberts”
- compute similarity between each pair of aliases
  (based on Levenshtein distance)
- cluster together aliases with high similarity
- post-process manually
  - rely on external information (websites)
  - precise but labor-intensive

id = 17
{ John Doe,
Doe John,
john@doe.org,
john_doe@hotmail.com,
john.doe@gmail.com }

Identity merging

- several merge algorithms exist
- the “noisier” the data, the worse they perform!
- simple algorithms have higher precision and recall than more complex ones

A Comparison of Identity Merge Algorithms for Software Repositories

Mathieu Goeminne*, Tom Mens*

Institut d’Informatique, Faculté des Sciences, Université de Mons

Science of Computer Programming 28(8), August 2013

Abstract

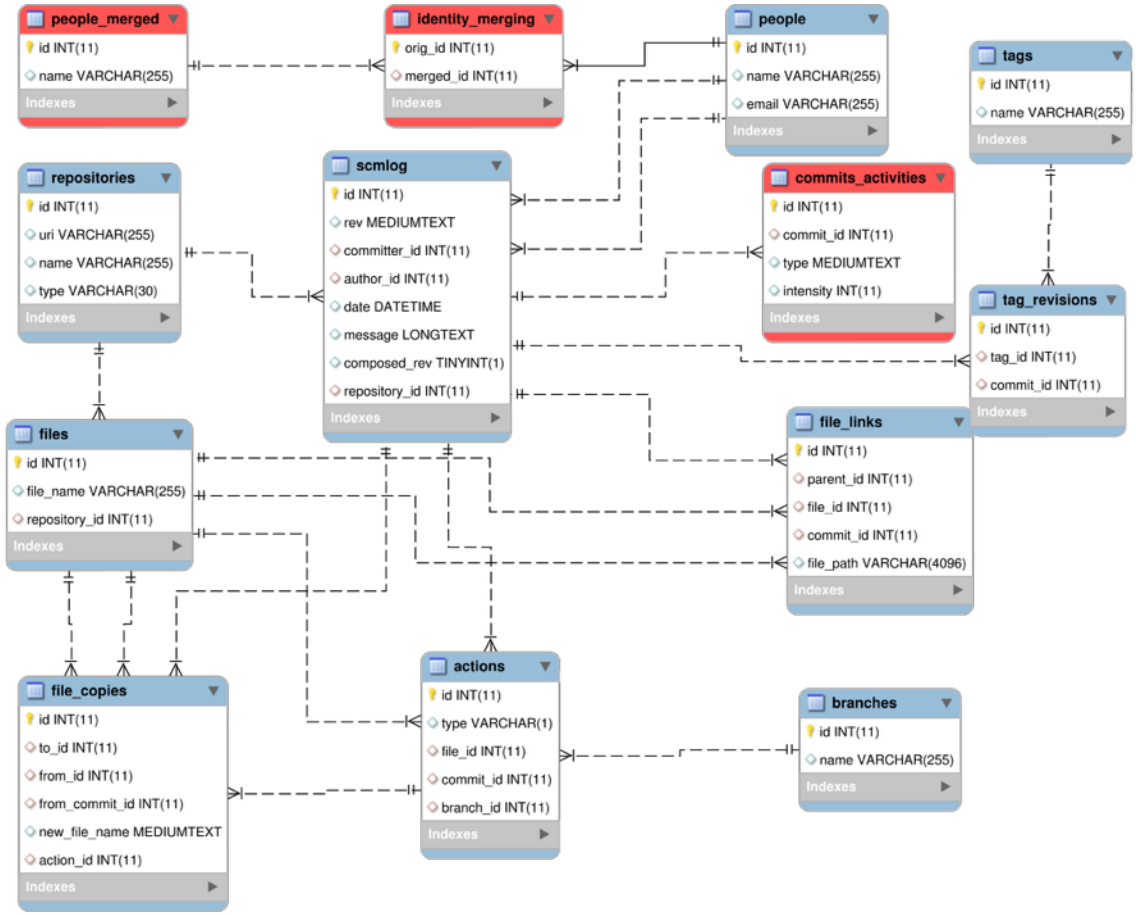
Software repository mining research extracts and analyses data originating from multiple software repositories to understand the historical development of software systems, and to propose better ways to evolve such systems in the future. Of particular interest is the study of the activities and interactions between the persons involved in the software development process. The main challenge with such studies lies in the ability to determine the identities (e.g., logins or e-mail accounts) in software repositories that represent the same physical person. To achieve this, different identity merge algorithms have been proposed in the past. This article provides an objective comparison of identity merge algorithms, including some improvements over existing algorithms. The results are validated on a selection of large ongoing open source software projects.

Keywords: software repository mining, empirical software engineering, identity merging, open source, software evolution, comparison

GNOME Characteristics

Dataset shared on
https://bitbucket.org/mgoeminne/sgl-flossmetric-dbmerge/downloads

FLOSSMetrics compliant
MySQL database



Goeminne *et al.* "A historical dataset for GNOME contributors", MSR 2013

GNOME Characteristics

16 years of activity

> 1.3M of commits

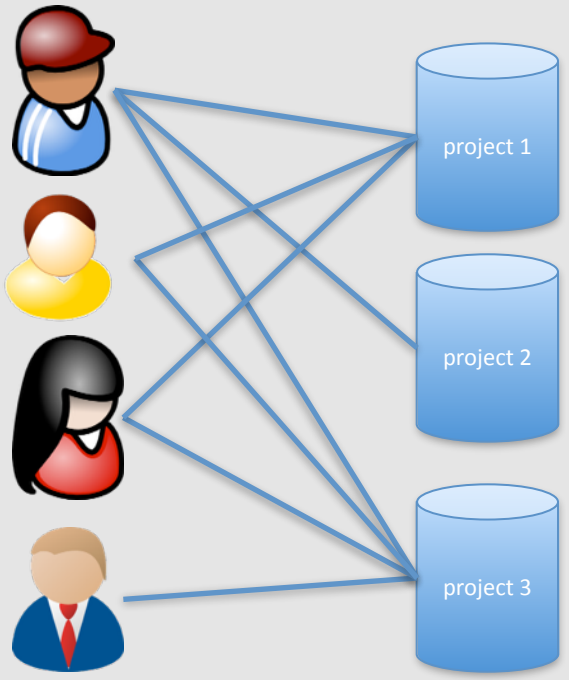
(> 0.6M of code commits)

> 12M of file touches

(> 6M of code file touches)

Mainly C, C++, Python

Bipartite contributor-project graph



> 5800 contributors
(> 4300 coders)

> 1400 projects

GNOME
Top Contributor Distribution

Who are the top GNOME contributors?

in the version repository

in the bug tracker

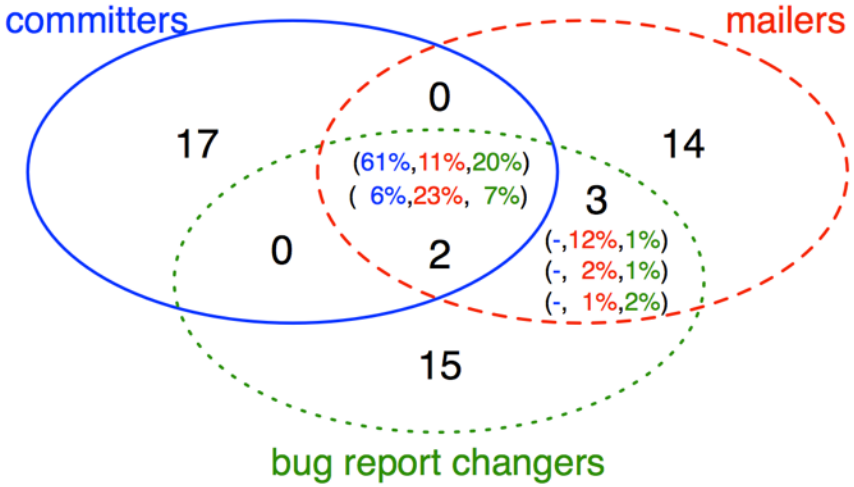
in the mailing list

GNOME
Top Contributor Distribution

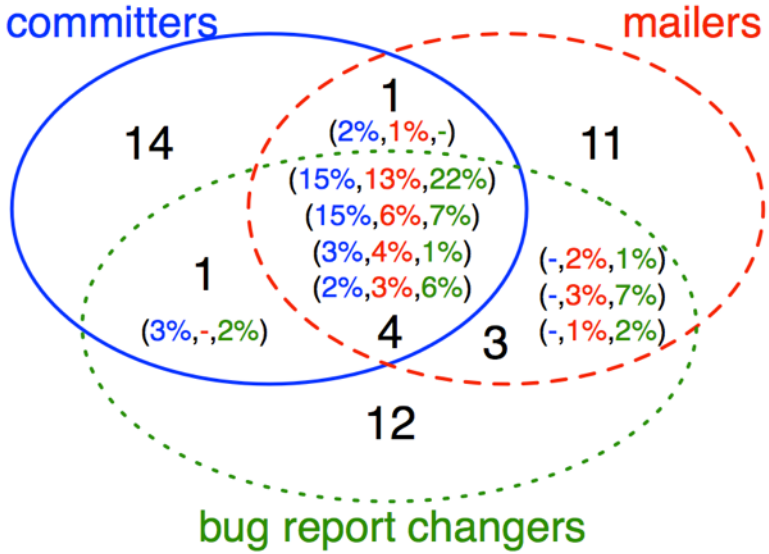
Approach

- Analyse individual GNOME projects
- Identify core groups
  - Compute Venn diagrams of most active (top 20) persons per considered data source
  - Show % of activity attributable to each person
  - Take into account identity merges

GNOME
Top Contributor Distribution



Brasero



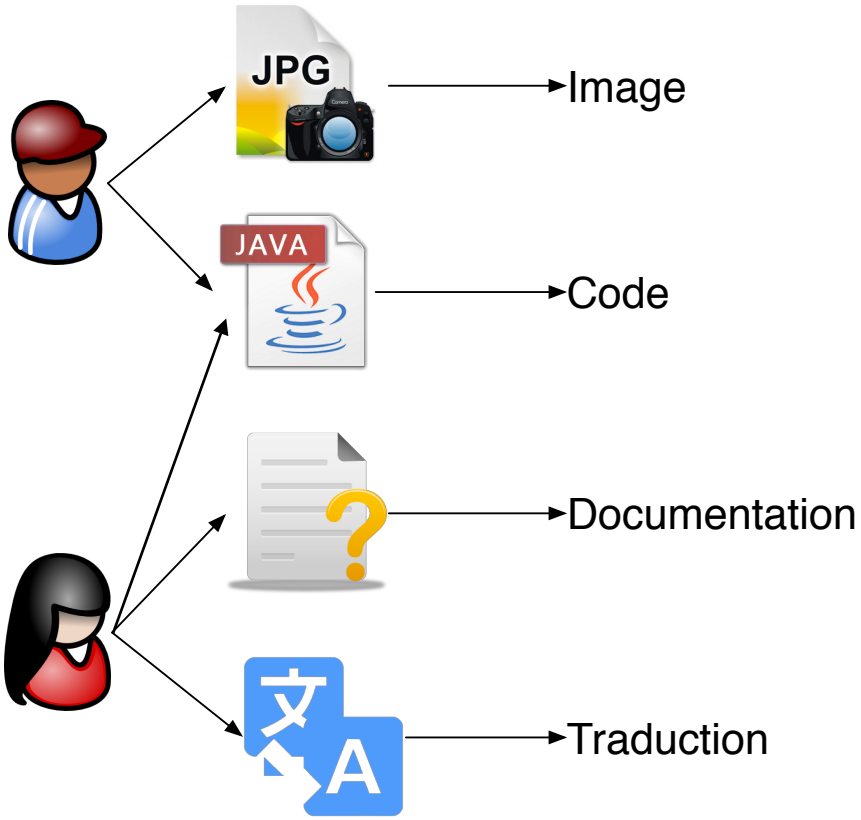
Evince

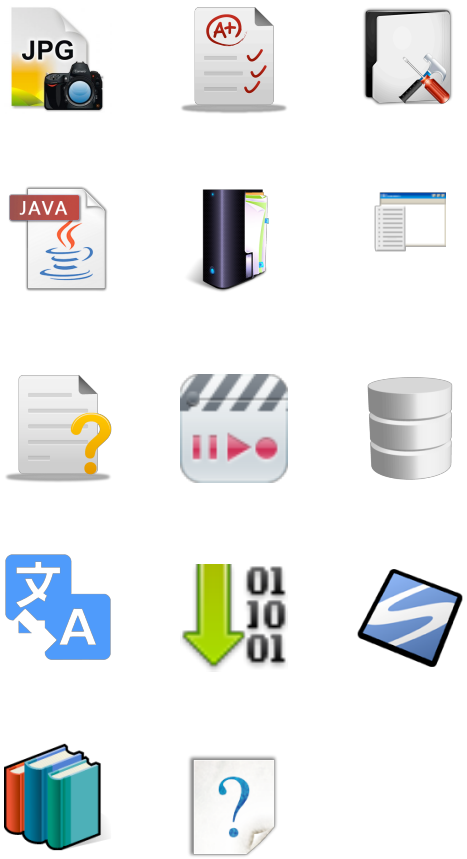
GNOME
Workload Distribution

How is workload distributed over different authors and projects?

GNOME
Workload Distribution

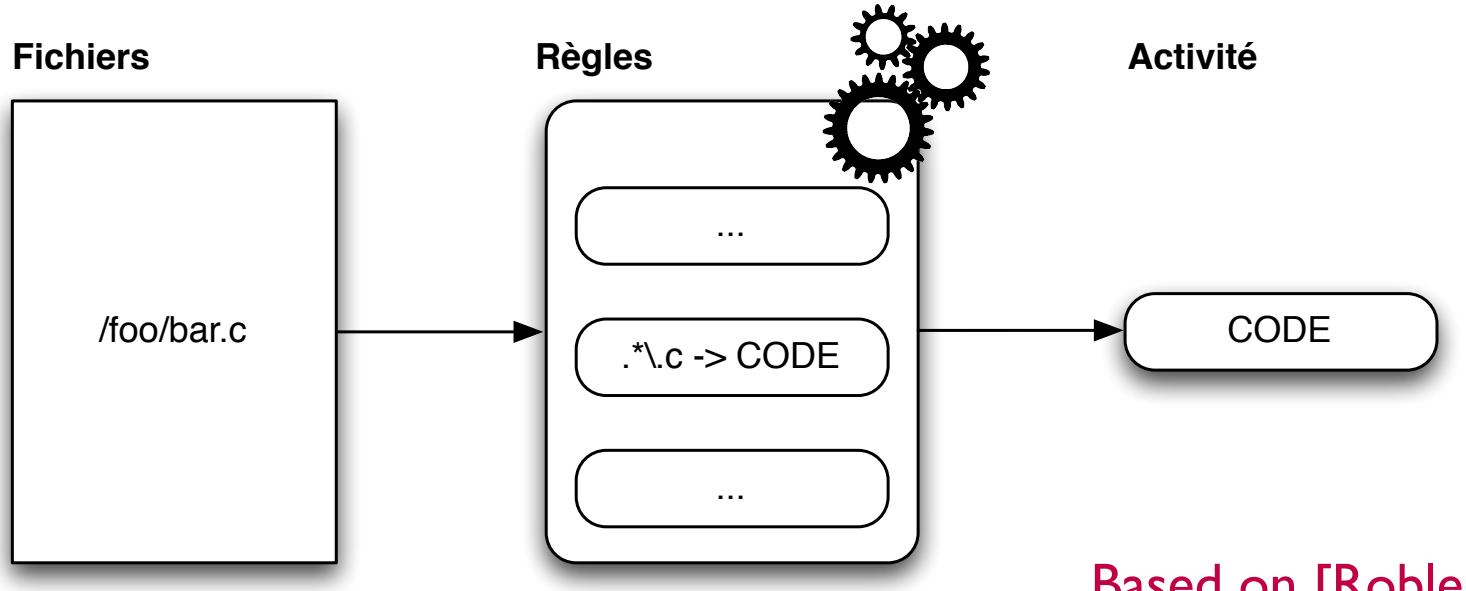
How is workload distributed over different authors and projects **per activity type**?





GNOME
Workload Distribution

- Extract file information for each commit in the git repository of each GNOME project
- Associate a unique activity type *t* to each file
- Count the number of file touches



Based on [Robles2006]

GNOME
Workload Distribution

How is workload distributed over different authors and projects **per activity type**?

- Two *dual* views (cf. contributor-project graph)
- - Distribution of workload over different *projects* per activity type
- - Distribution of workload over different *authors* per activity type?

GNOME
Workload Distribution

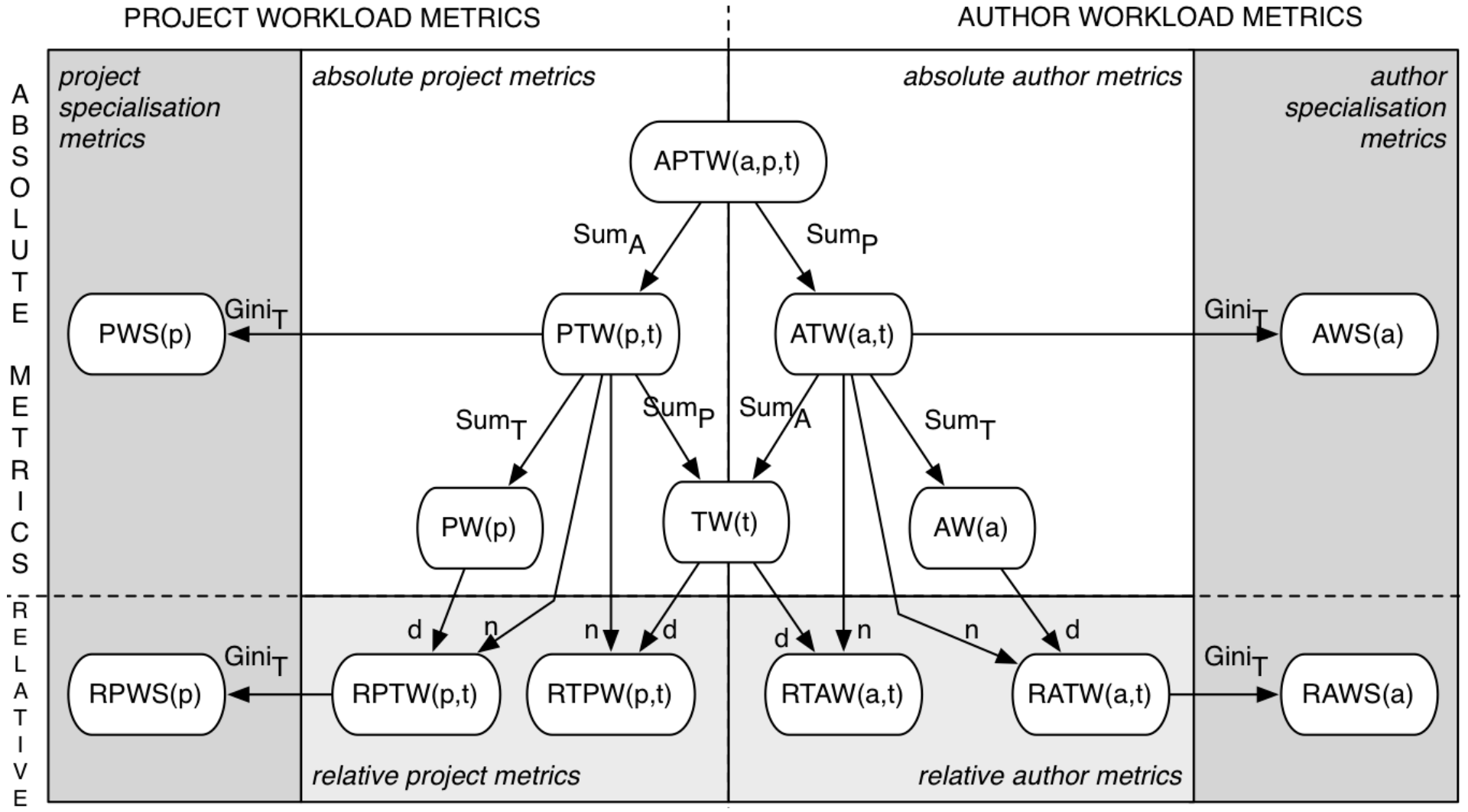
Basic **W**orkload metric **APTW(*a*,*p*,*t*)**

= number of file touches of an **A**uthor *a* for a given **P**roject *p* and activity **T**ype *t*

Many derived metrics

- based on sum and Gini coefficient

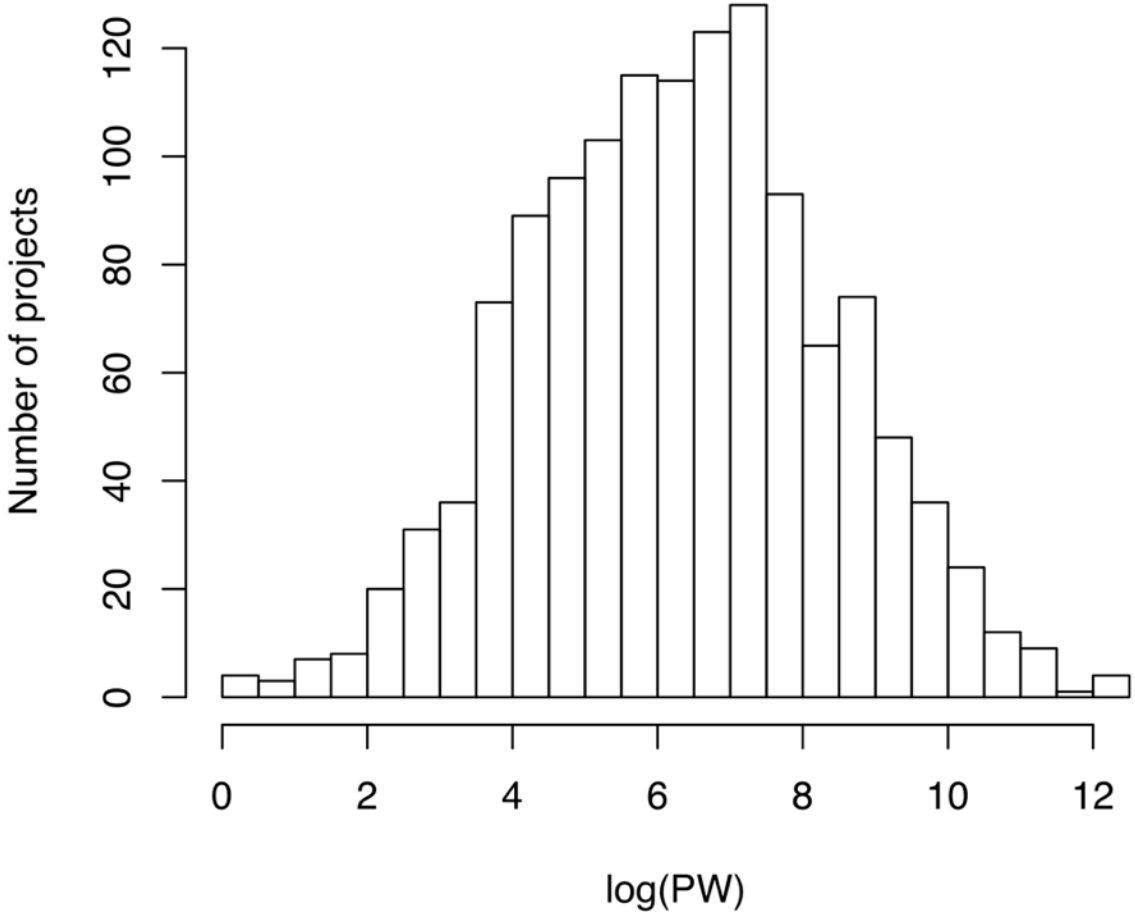
GNOME
Workload Metrics



GNOME Workload Metrics

Main findings

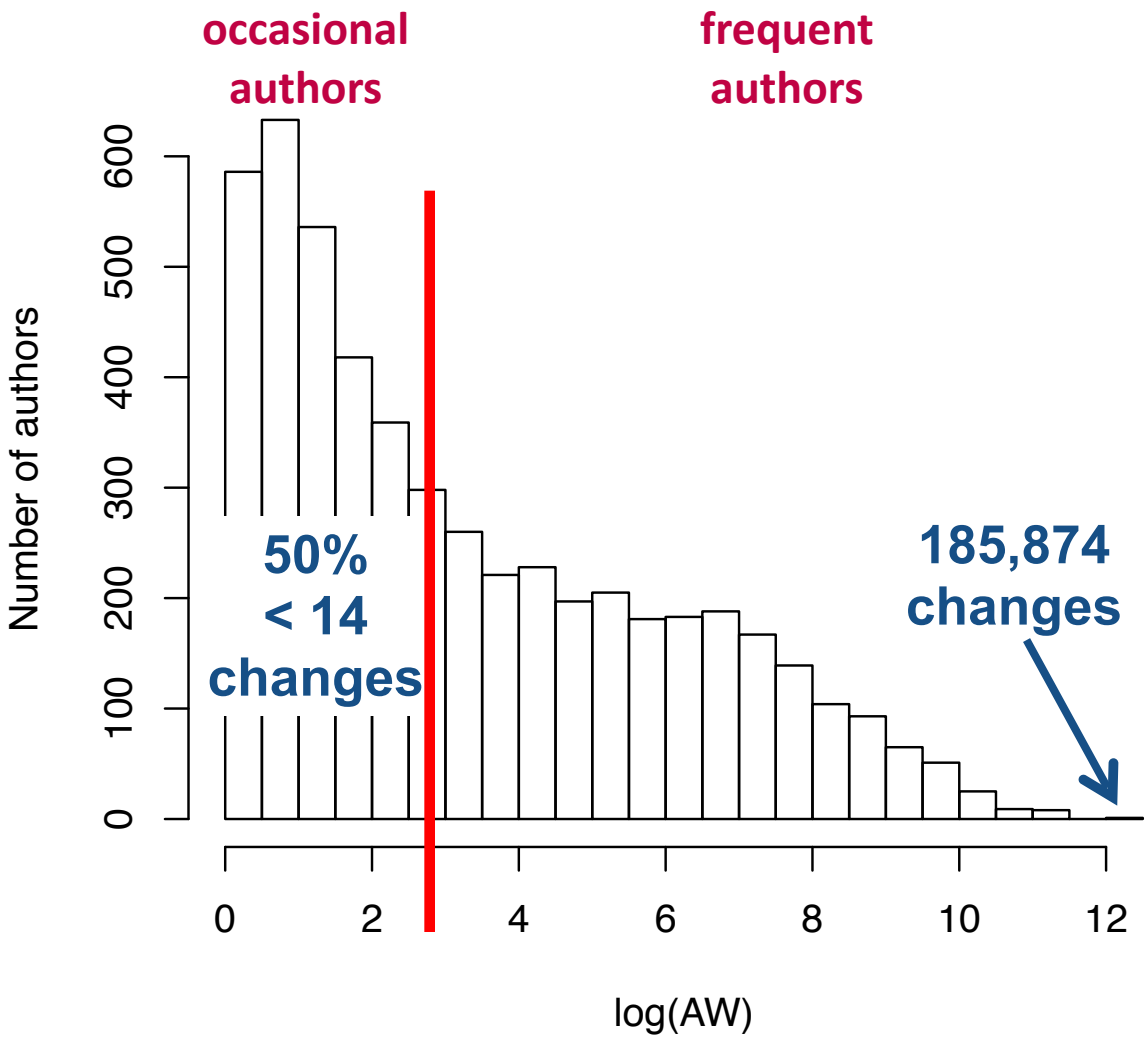
Workload is log-normally distributed over GNOME projects



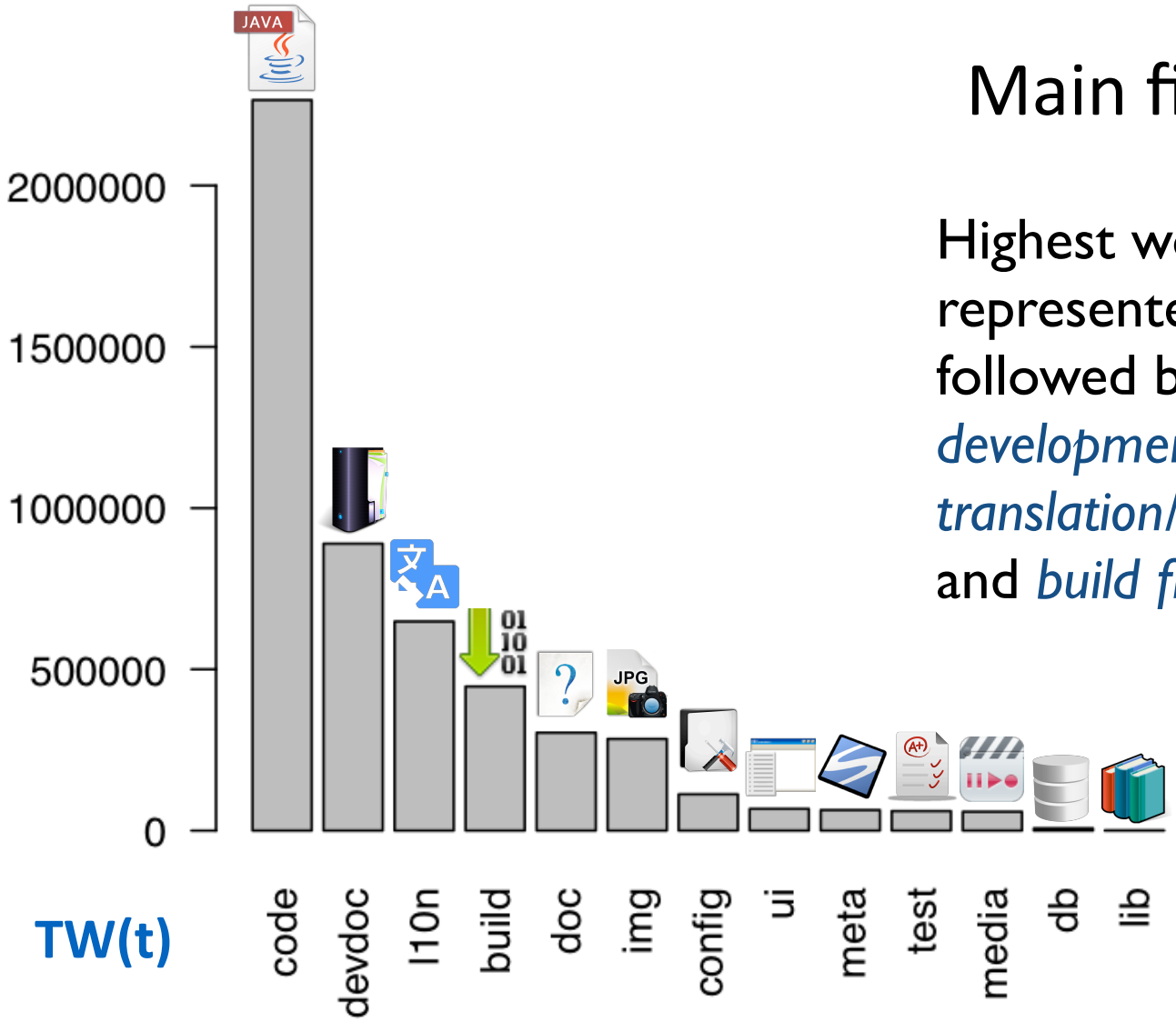
GNOME Workload Metrics

Main findings

The majority of GNOME authors are involved in a very low number of file touches.



GNOME Workload Metrics



Main findings

Highest workload is represented by *coding* activity, followed by activities of *development documentation*, *translation/internationalisation*, and *build file* creation.

GNOME

Relative importance of activity types

What are the favourite activity types for GNOME?

Two dual views

- Relative importance of each activity type *per author*
- Relative importance of each activity type *per project*

GNOME

Relative importance of activity types

What are the favourite activity types for GNOME?

Approach

- Use statistical tests to compare distributions
- Verify if a data set corresponding to an activity type tends to have higher values than a data set corresponding to another activity type

GNOME

Relative importance of activity types

Examples of statistical comparison tests

- (Wilcoxon-)Mann–Whitney *U* test
- Kruskal-Wallis test

Problems with traditional statistical tests:

- Not robust to populations of unequal sizes
- Different tests can be inconsistent with each other
- Pairwise comparison of all activity types requires 78 different combinations (12 * 13 / 2)
- Traditional tests are not transitive

GNOME

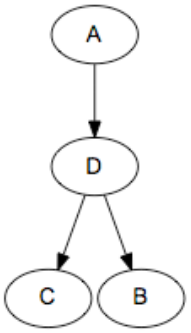
Relative importance of activity types

Solution:

- Use a single test that respects transitivity
- T procedure [Konietschke et al 2012]

Activity type	Developers
A	2 2 2 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 5 5
B	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2
C	1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 3 3
D	1 1 1 2 2 2 2 2 2 2 3 3 3 3 3 3 4 4 4 4

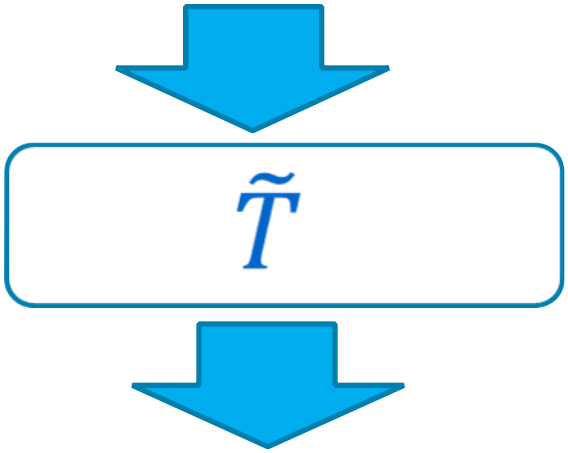
Pair	Lower	Upper	p-value
B–A	-0.560	-0.444	0.000
C–A	-0.503	-0.313	7.536e-10
D–A	-0.320	-0.027	1.997e-02
C–B	-0.014	0.242	9.742e-02
D–B	0.237	0.470	1.200e-06
D–C	0.090	0.404	2.432e-03

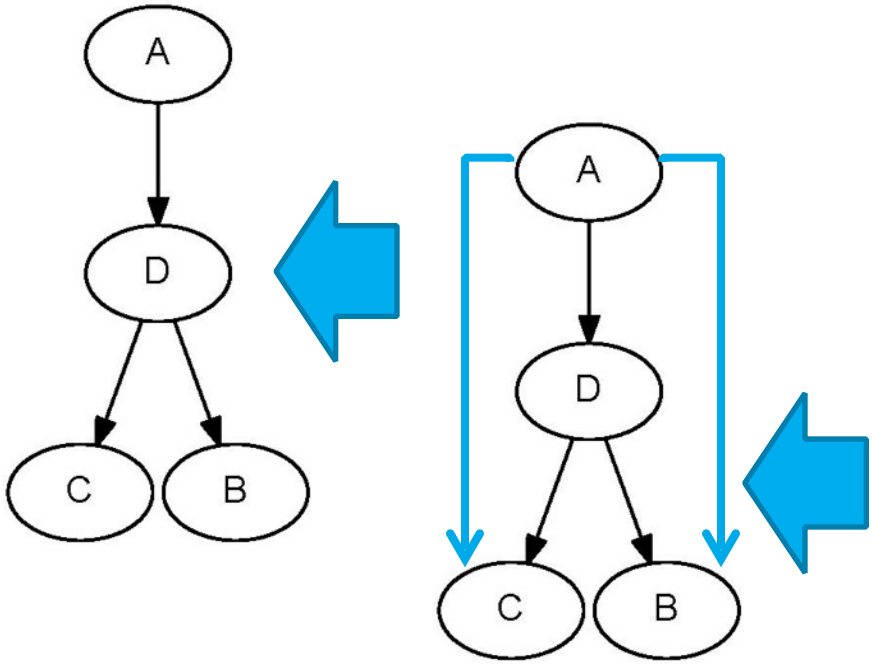


GNOME

Relative importance of activity types

- $\tilde{T}$ procedure

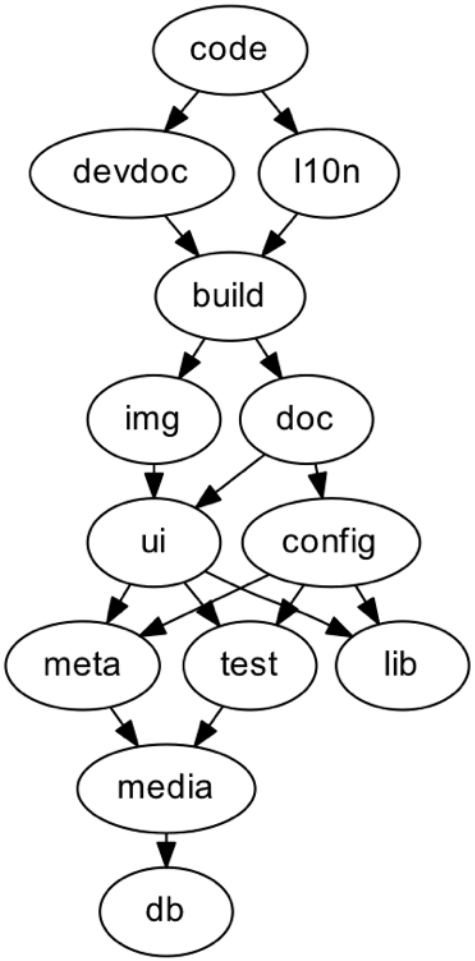




	Pair	Low	High
A→B A→C A→D	B-A	-0.56	-0.44
	C-A	-0.50	-0.31
	D-A	-0.32	-0.03
	C-B	-0.01	0.24
D→B D→C	D-B	0.24	0.47
	D-C	0.09	0.40

GNOME

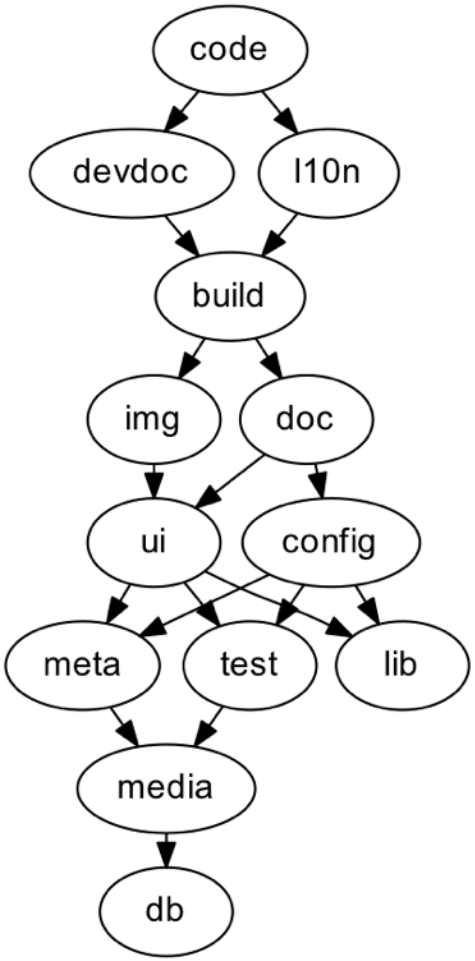
Relative importance of activity types



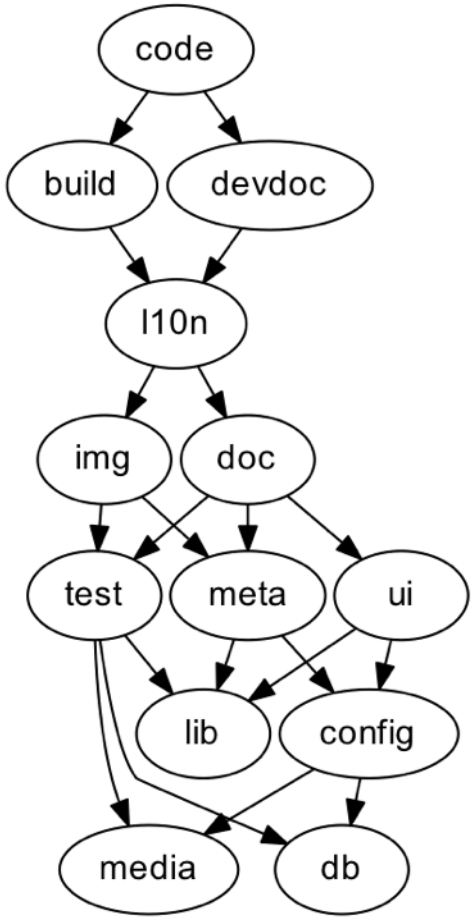
by *author*

GNOME

Relative importance of activity types



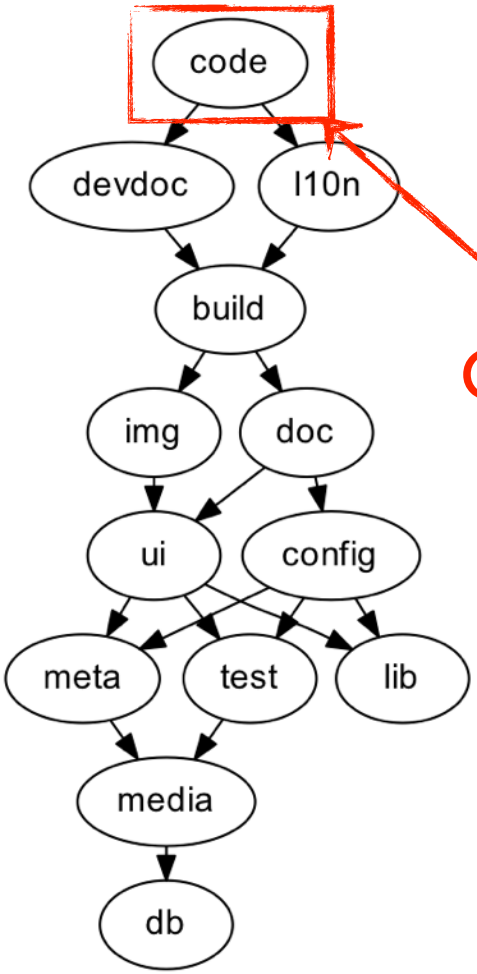
by *author*



by *project*

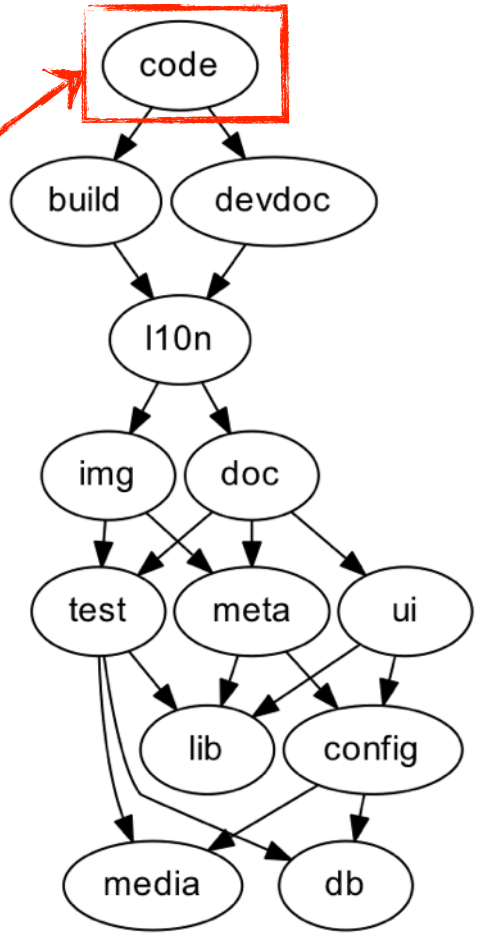
GNOME

Relative importance of activity types



by *author*

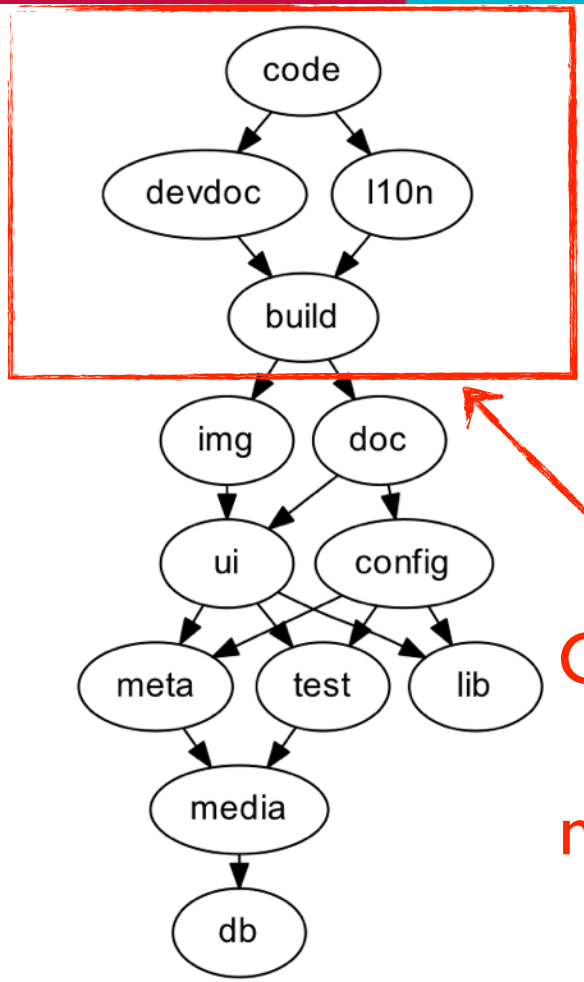
GNOME projects and authors are code-centric



by *project*

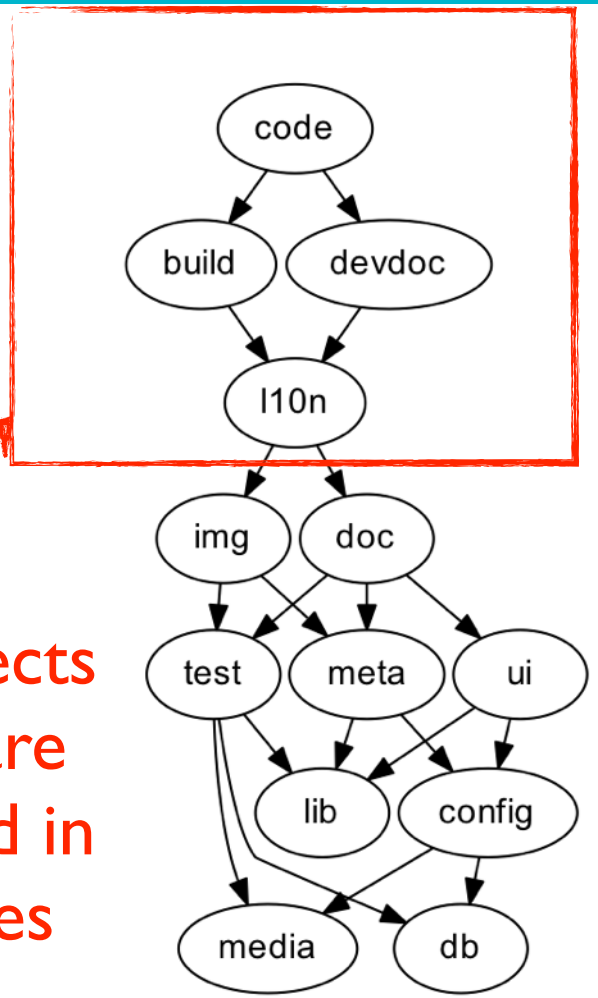
GNOME

Relative importance of activity types



by *author*

GNOME projects and authors are mainly involved in 4 activity types



by *project*

GNOME

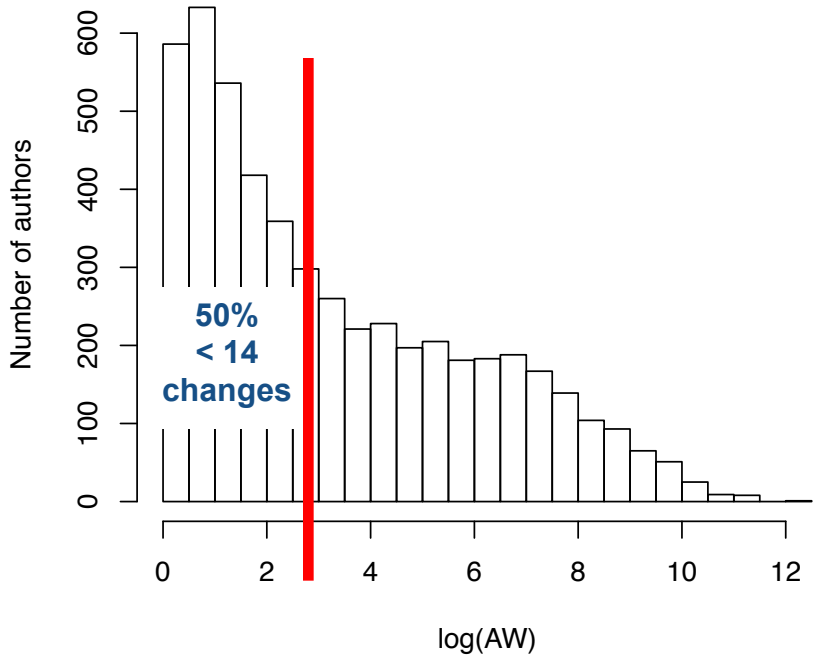
Heterogeneous communities

Does the relative importance of activity types differ between *frequent and occasional* authors?

Idea

Equally split the authors in two bins of more or less equal size, based on the author workload:

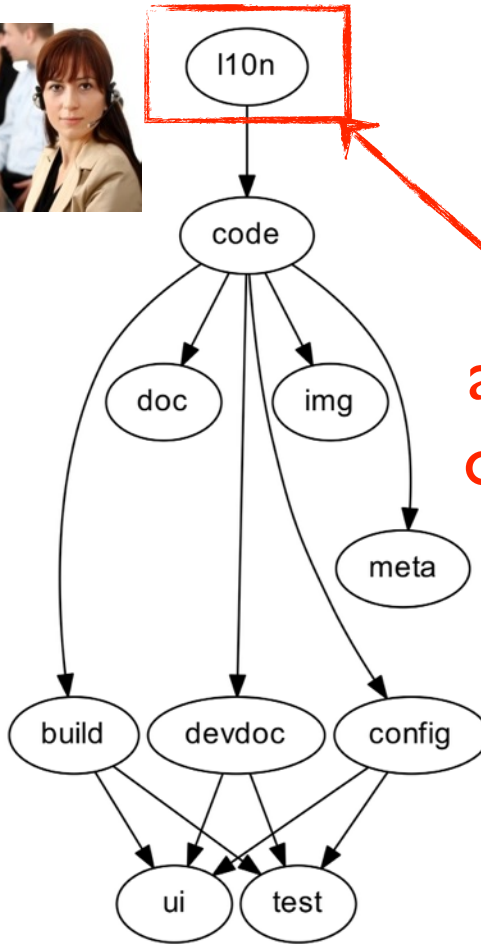
about 50% of all authors were involved in <14 file touches



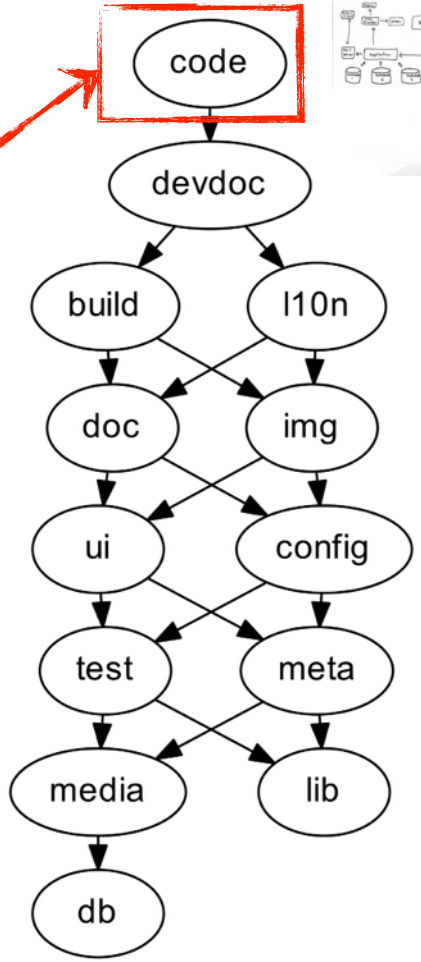
GNOME

Heterogeneous communities

Occasional authors



Frequent authors



Frequent authors are mostly coders, occasional authors are mostly translators.

GNOME
Heterogeneous communities

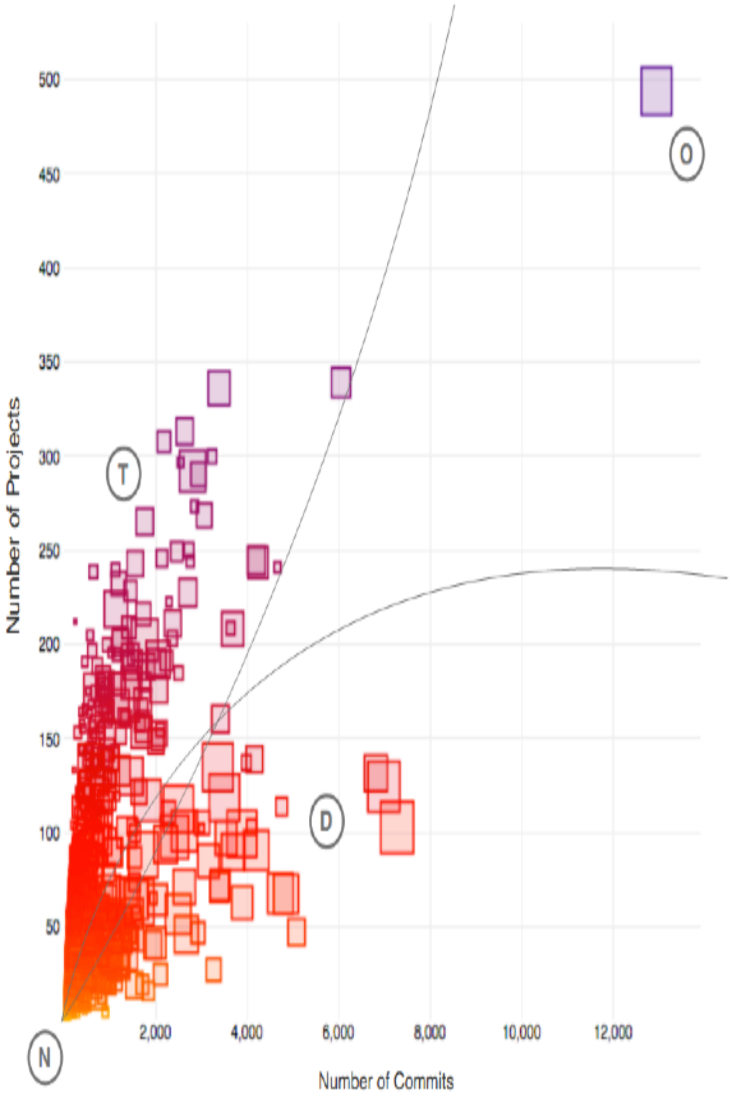
Observations

Coders have a higher workload and are involved in less projects

Translators are less active but are involved in more projects

Can be explained in part by the use of *Damned Lies*, a Web application used to manage the localisation (l10n) activities of the GNOME project

GNOME
Heterogeneous communities



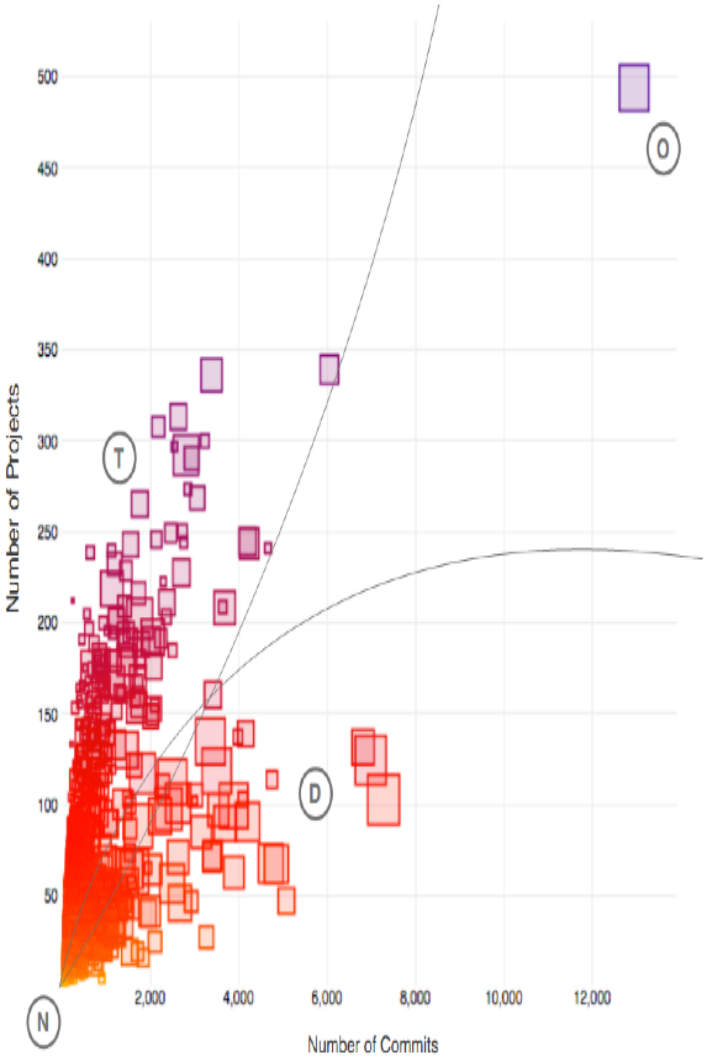
Sylvia Neu *et al.* "Telling stories about GNOME with Complicity", VISSOFT 2011

Complicity is a web-based application supporting software ecosystem analysis by means of interactive visualizations.

Affectional bond view:

- size of rectangle = author's lifetime in days
- color = number of projects

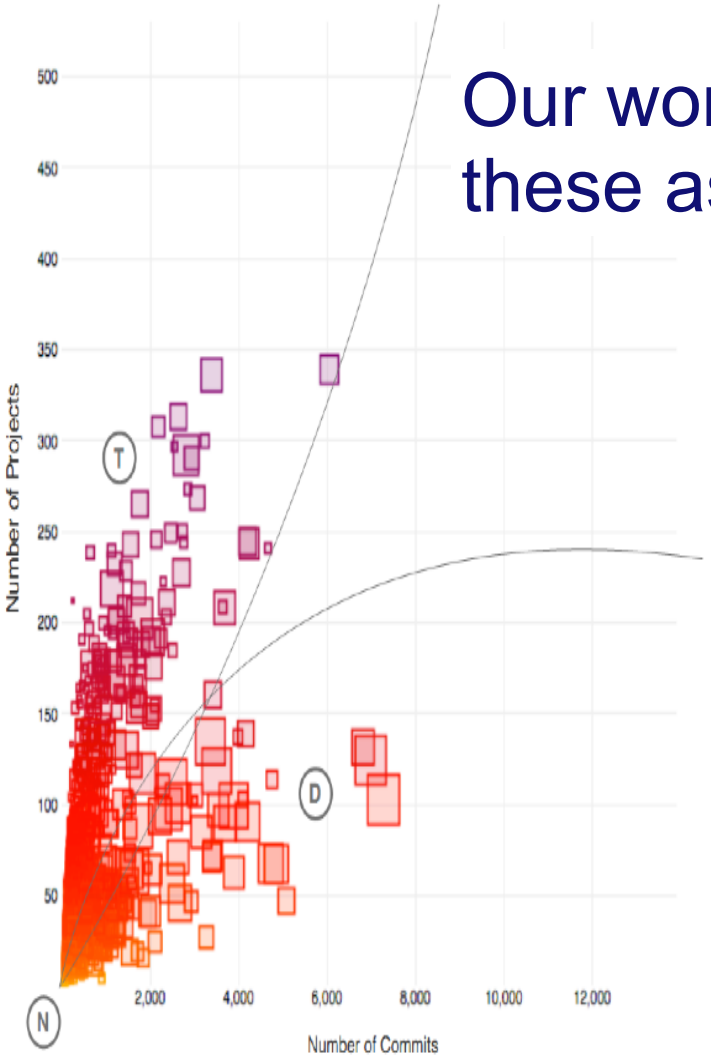
GNOME
Heterogeneous communities



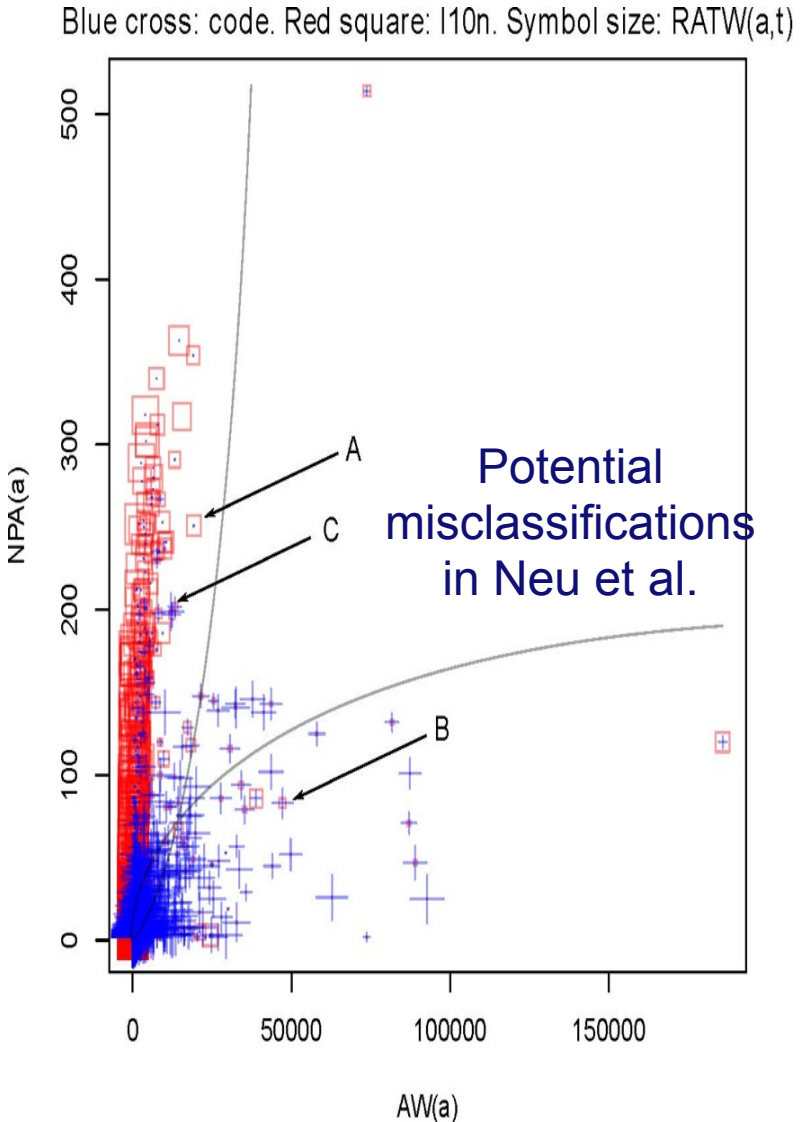
Unverified assumptions:

1. Authors contributing a lot to few projects are likely to be *developers* (D)
2. Authors contributing less often to more projects are likely to be *translators* (T)
3. Authors tend to have an affectional bond to *either* development *or* translation work

Case Study: GNOME
Heterogeneous communities



Our work confirms these assumptions



Potential misclassifications in Neu et al.

Case Study: GNOME Relative Workload

How strongly do authors focus on specific activities?

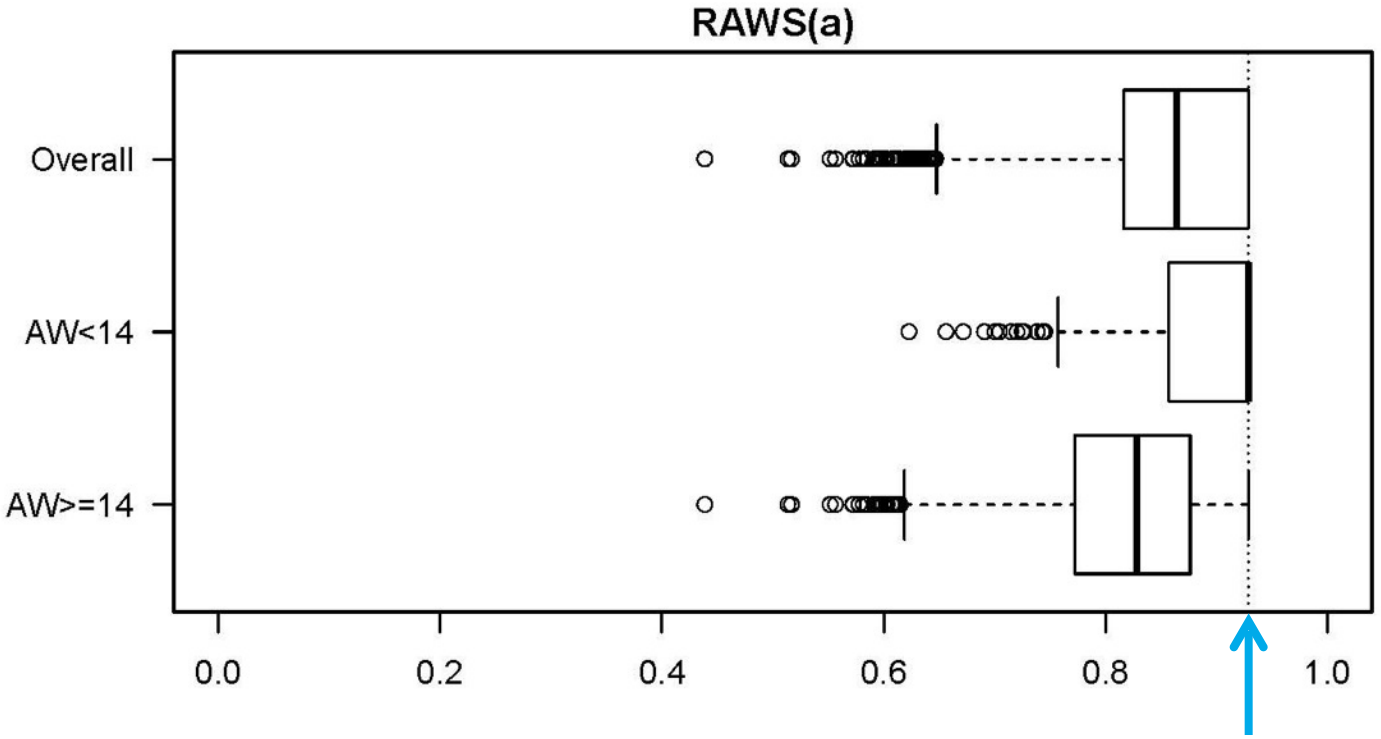
Basic measures:

- RATW(**a**,**t**)
  = % of the total workload of author **a** dedicated to activity type **t**

- RAWS(**a**) = author specialisation
  = Gini index of of inequality of RATW(a,t) aggregated over all activity types

Case Study: GNOME Relative Workload

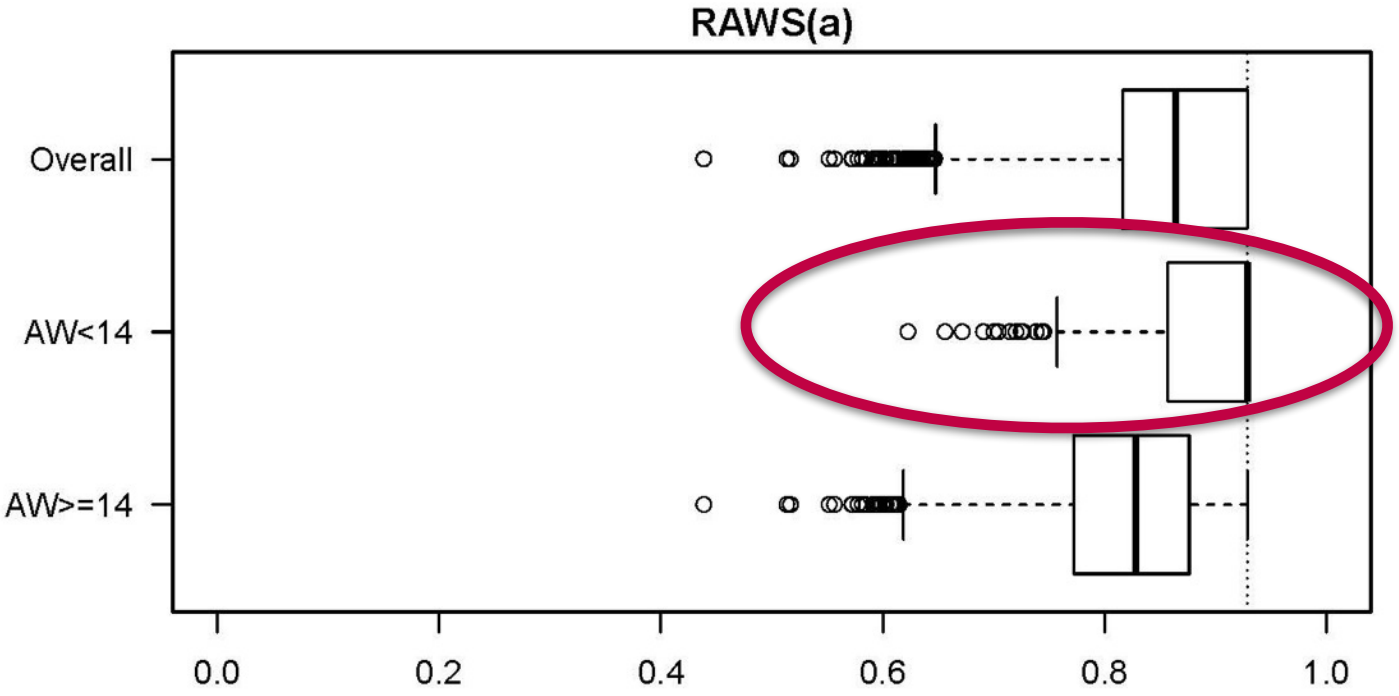
How strongly do authors focus?



max Gini for n = 14: 0.9285

Case Study: GNOME Relative Workload

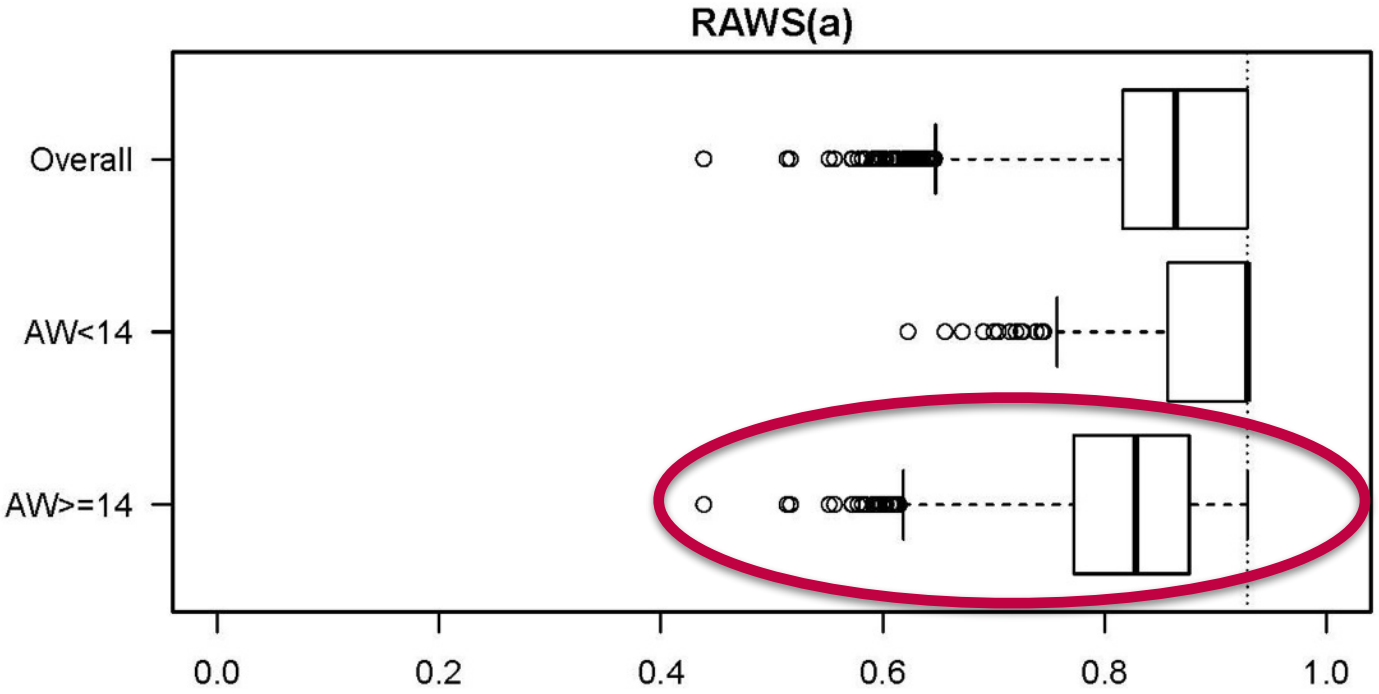
How strongly do authors focus?



Occasional authors tend to focus on a single activity type

Case Study: GNOME Relative Workload

How strongly do authors focus?



Frequent authors tend to focus on few activity types.

Case Study: GNOME Workload Distribution

Main observations for GNOME ecosystem:

- Workload is unevenly distributed over *projects* and *authors*
- Clear distinction between *frequent* and *occasional* authors
- Authors form heterogeneous subcommunities of *coders* and *translators*
- GNOME is *code-centric*:
  workload is concentrated code-related activities
  (coding, build files, development documentation)

GNOME
Next steps

Observation: existing generic tool support does not take the specificities of the ecosystem into account, making the support suboptimal.

Having gained better understanding of the GNOME ecosystem specificities, we hope to come up with better change support mechanisms

- Dedicated to specific sub communities
  - e.g. Damned Lies application for translation community
- Estimation (of cost or effort) and prediction models (e.g. of defects) could be improved
- Tools should be able to focus on those activities/projects a contributor is interested in (based on his historic activity profile)