UMONS
Université de Mons

Analysing the evolution of social aspects of open source software ecosystems

Tom Mens & **Mathieu Goeminne**
Service de Génie Logiciel
Département des Sciences Informatiques
Faculté des Sciences - Université de Mons
Belgique

Faculté
des Sciences

Research topic

- Study of *software evolution*
  - in particular, the software *process* and software *quality*
- Focus on *software ecosystems* (see next slide)
- Focus on *social aspects* (see next slide)
- By analysing and visualising *historical* data of *open source projects*
  - obtained by mining and combining data from different types of *repositories*

Goal

- Analyse the success factors (e.g. popularity, quality) of OSS projects?
  - What are good and bad practices of OSS evolution? What lessons can we learn?
- Study how social aspects *co-evolve* with, and affect the software product and process
- Exploit this knowledge to improve the current practice (e.g., guidelines, tool support)

Software ecosystem

- There are many views on a software ecosystem. Our focus:
  - *the ecosystem of a single project*, containing all artefacts (code, documentation, etc.) and persons involved in using, producing or modifying these artefacts
  - the ecosystem of a *coherent collection* or distribution of individual projects
- Example: GNOME desktop environment for Linux, containing hundreds of applications/project
  - The ecosystem of each individual GNOME project can be studied
  - The interaction between all GNOME projects and their contributors can be studied
  - Other examples: Linux OS distributions (e.g. Debian, Ubuntu)

Social aspects

- Which communities (e.g. users, developers) are involved in a software project?
  - relation between project quality and popularity?
- How do communities communicate / interact?
  - e.g. how are developers driven by user requests and how does this influence the project evolution?
- How are communities structured?
  - relation between community structure and software quality or maintainability?
- How is work distributed among persons?
  - relation between distribution of work / responsibility and maintainability?
- Which processes (formal or informal) are used?

Why open source?

- 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

open source™

Methodology

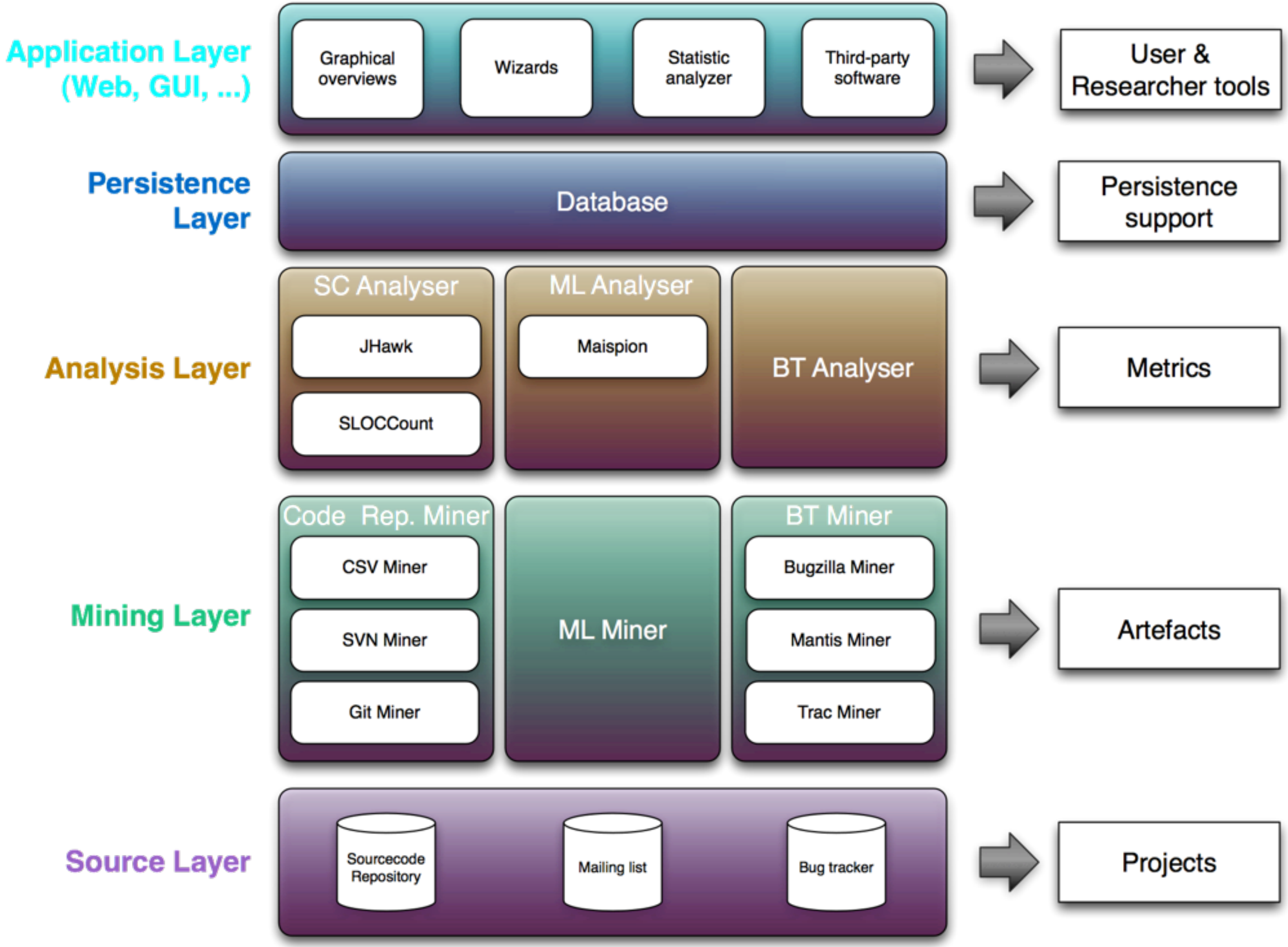
- Exploit available data from different repositories
  - code repositories (e.g. SVN, Git, ...)
  - mail repositories (mailing lists)
  - bug repositories (bug trackers)
- Select open source projects
- Use Herdsman framework
  - Based on FLOSSMetrics data extraction
  - Use identity merging tool
  - Use of statistical analysis and visualisation

Selecting Projects

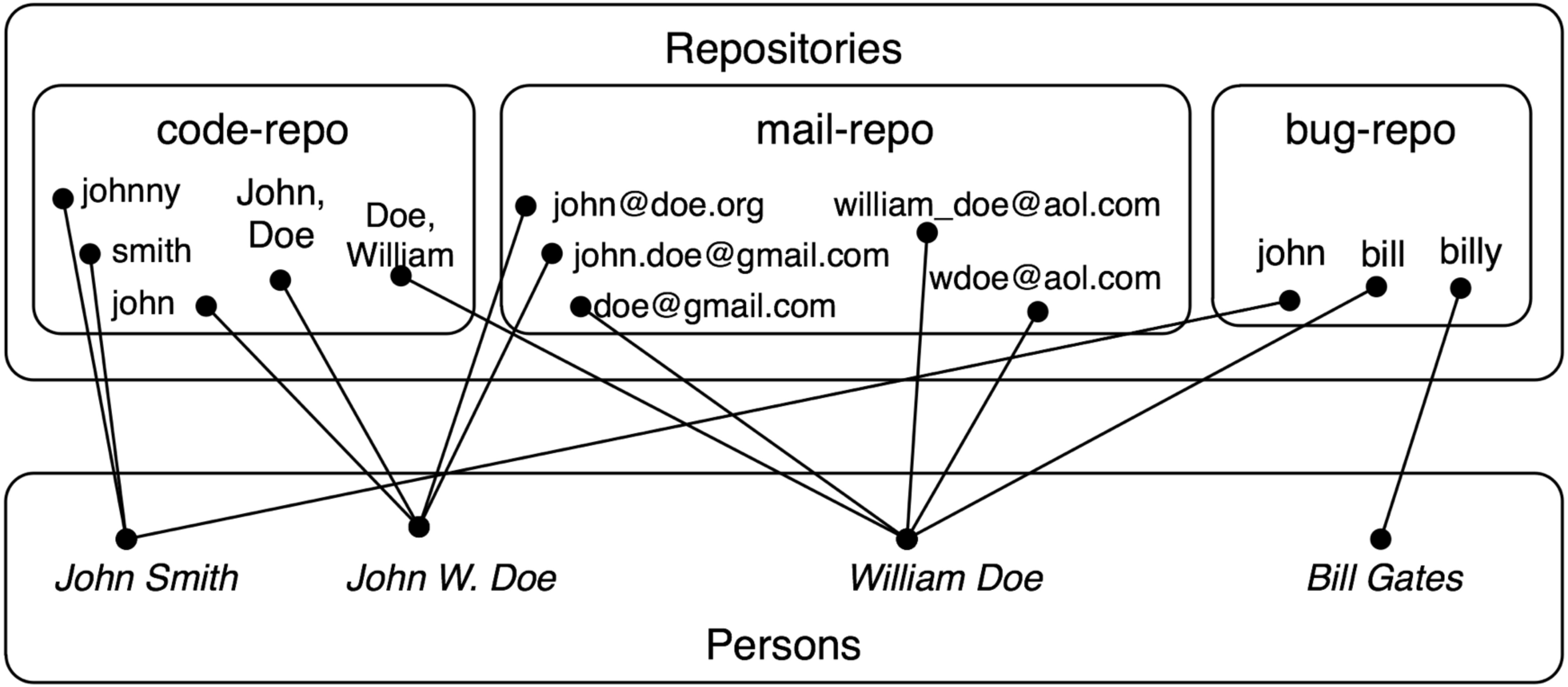
- Criteria for selecting projects
  - Availability of data from repositories
  - Data processable by FLOSSMetrics tools
    - CVSAnaly2, MLStats, Bicho

      Free/Libre and Open Source Software Metrics
  - Size of considered projects: persons involved, code size, activity in each repository
  - Age of considered projects

Herdsman Framework



Identity Merging



UMONS
Université de Mons

Comparing Merge Algorithms

- Based on a *reference merge model*
  - manually created
  - iterative approach
  - relying on information contained in different files (COMMITTERS, MAINTAINERS, AUTHORS, NEWS, README)
- Compute, for each algorithm, *precision* and *recall* w.r.t. reference model

Faculté
des Sciences

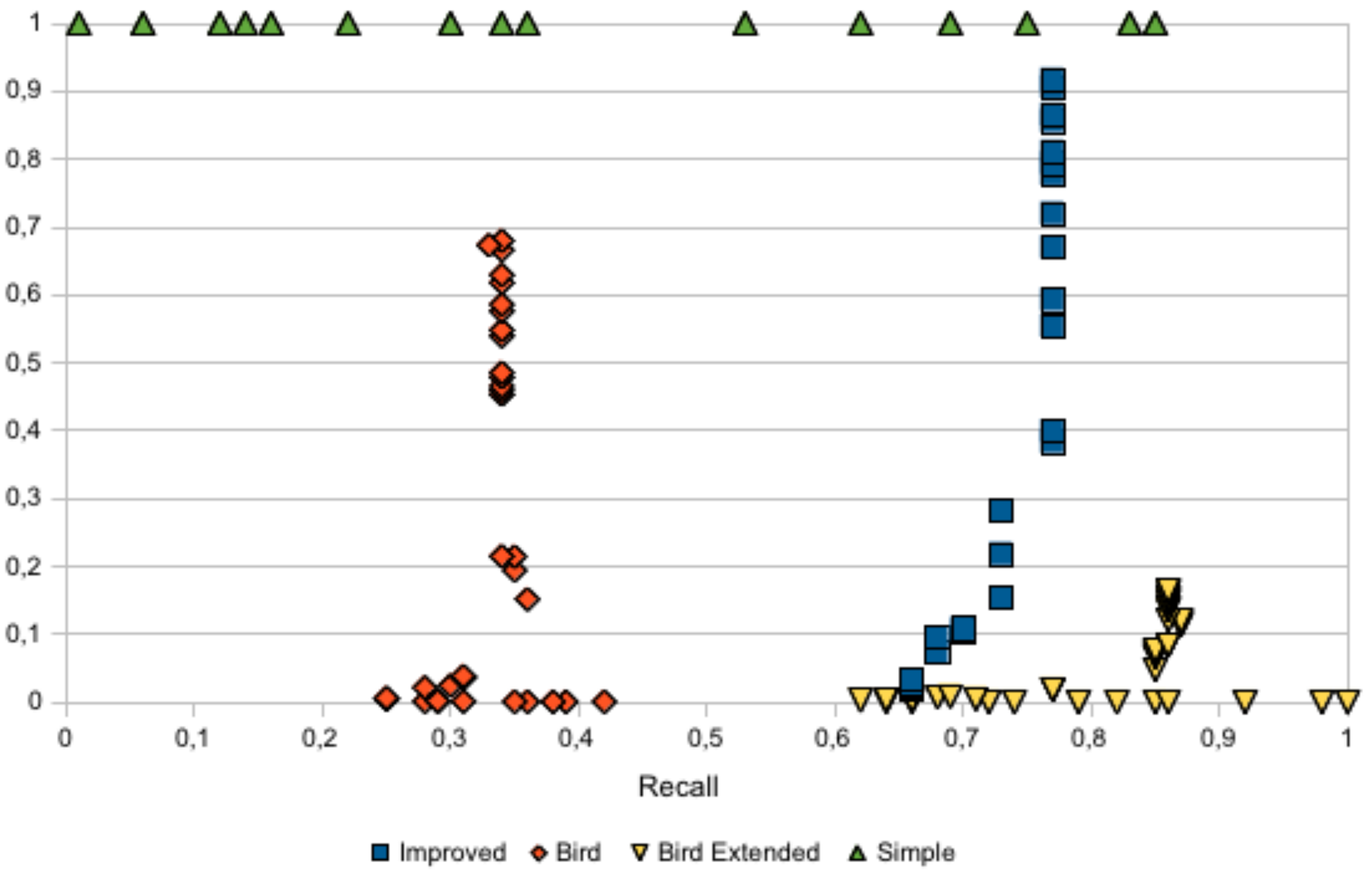
Comparing Merge Algorithms

- *Simple*
- *Bird* (code and mail repositories only)
  - based on Levenshtein distance
- *Bird, extended* for bug repositories
- *Improved*
  - Combining ideas from Bird and Robles

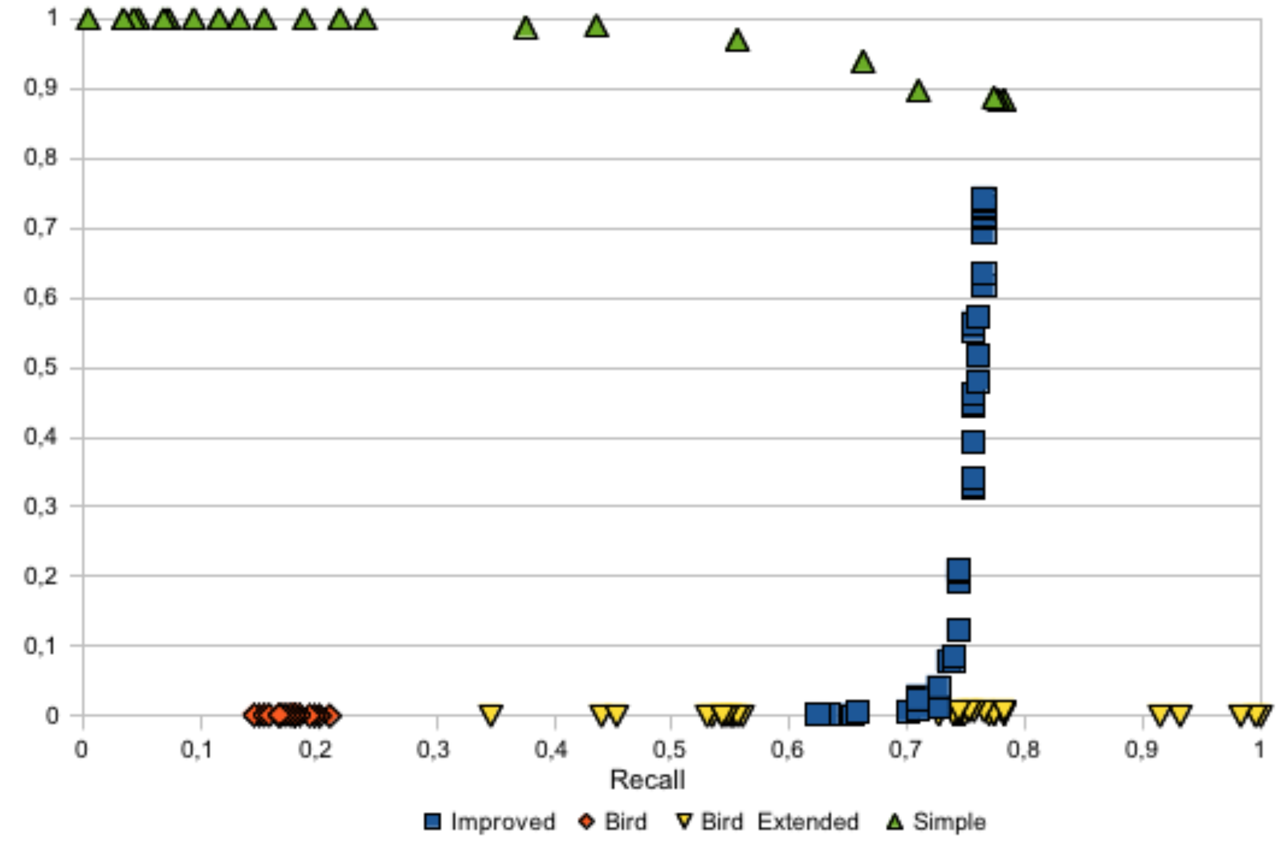
Université de Mons

Comparing Merge Algorithms

Brasero



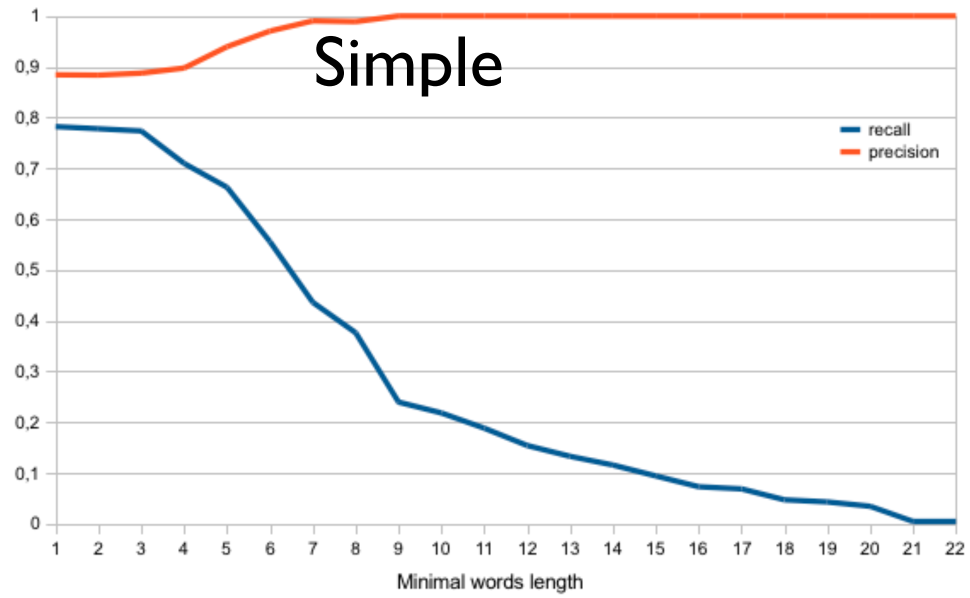
Evince

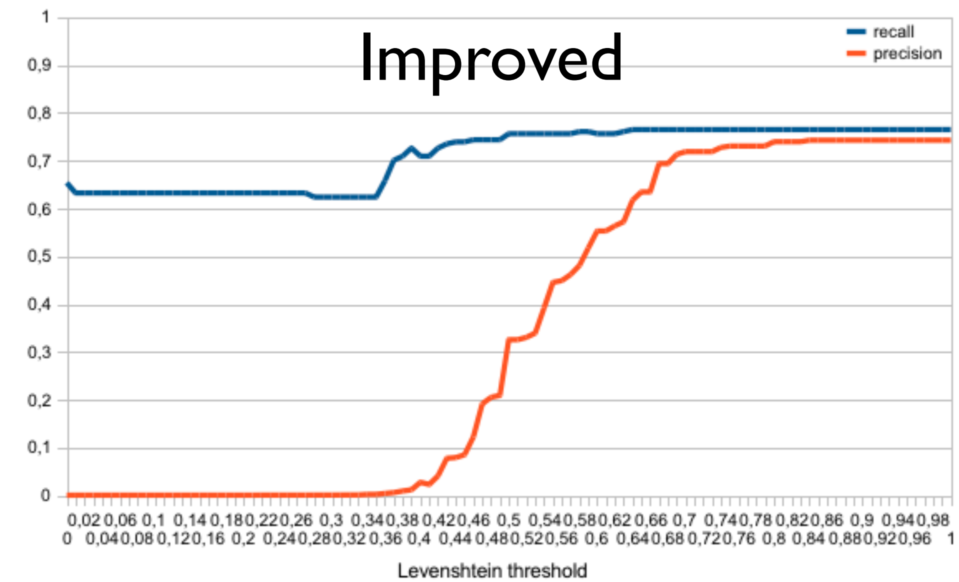


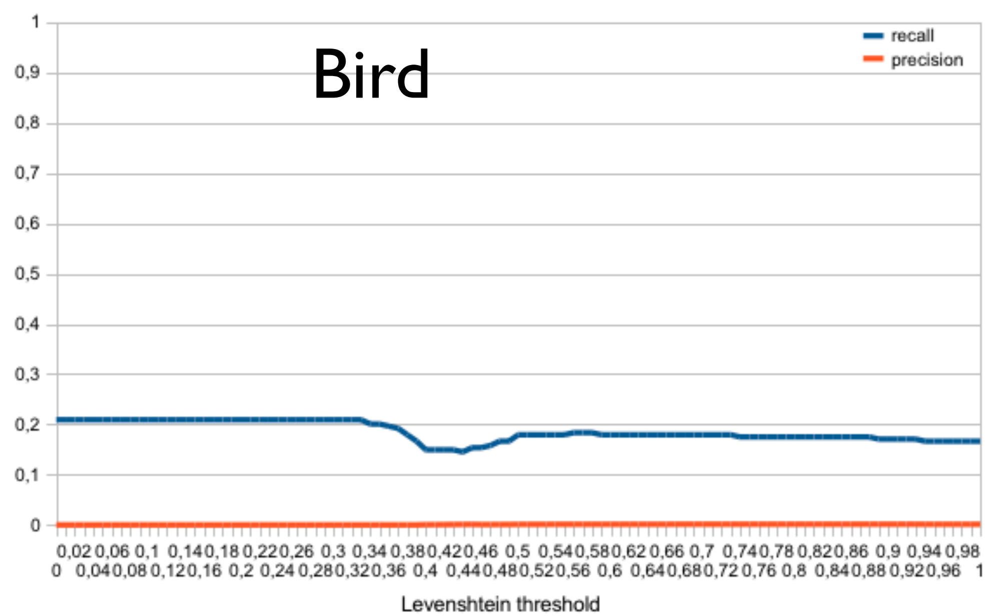
Faculté des Sciences

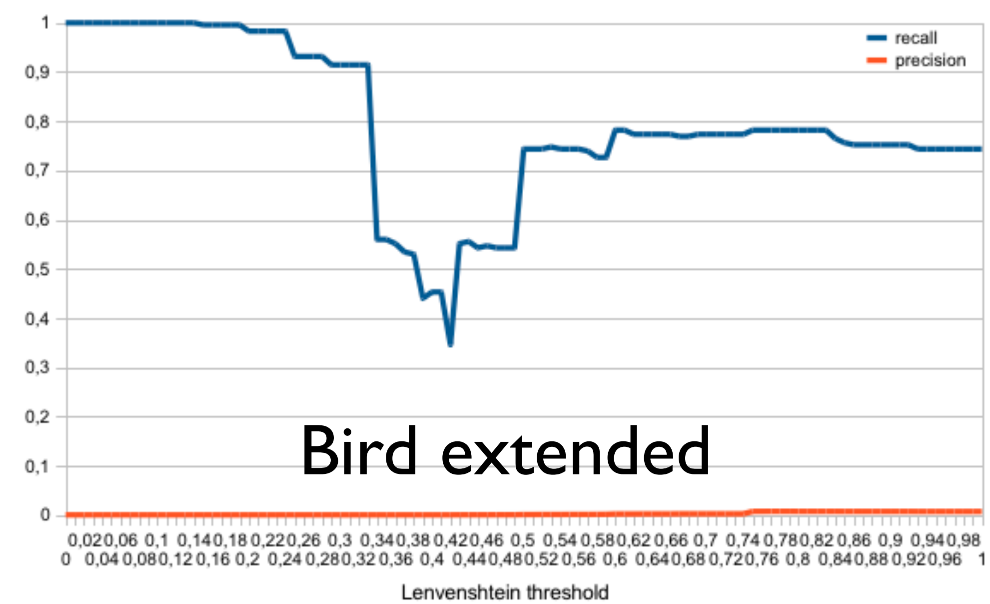
UMONS
Université de Mons

Comparing Merge Algorithms (varying parameter values) - Evince









Faculté
des Sciences

Analysing Activity Distribution

Analysing Activity distribution

Three different longitudinal studies

1. Distribution of work, for a *single project*, using *coarse-grained data* from different repositories

    How are developer activities (commits, mails, bug fixes) distributed across repositories? How is activity distributed across developers? How does this evolve over time?

2. Distribution of work, for a *single project*, using *fine-grained data* from a single repository

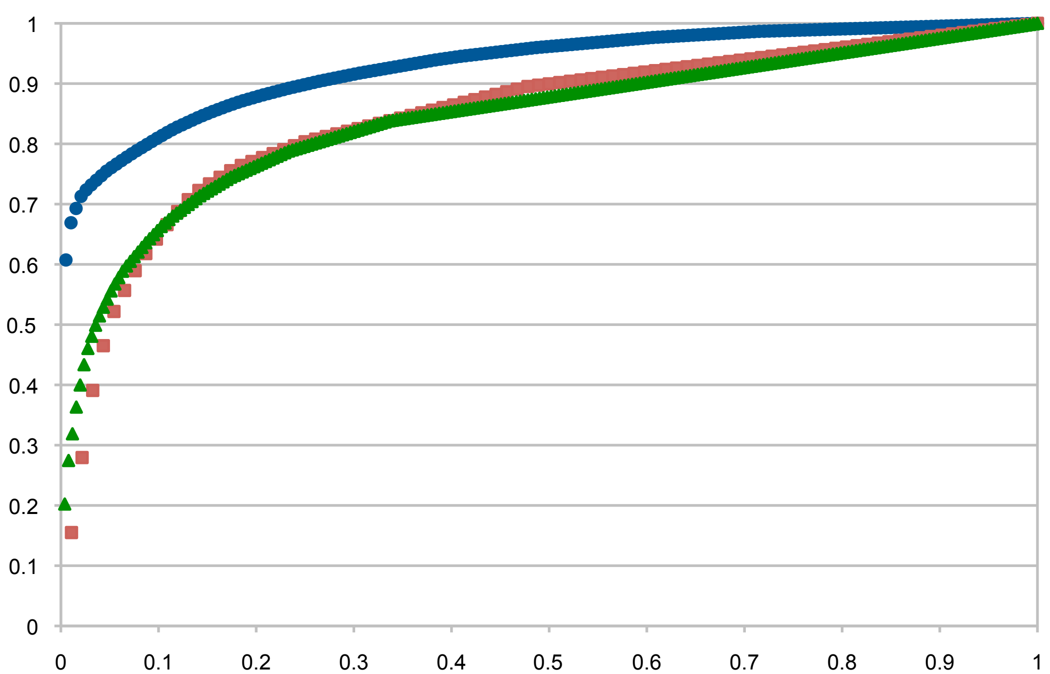
    Based on commits of different file types in version repository

3. Distribution of work across *different projects* belonging to the same community (e.g. GNOME)

Activity distribution First study

- Analyse, for individual Gnome projects, historical data from version repositories, bug trackers and mailing lists
- Focus on 3 types of activities per person: commits, mails, bug report changes
- How are activities distributed: over different persons, over time?

Activity distribution

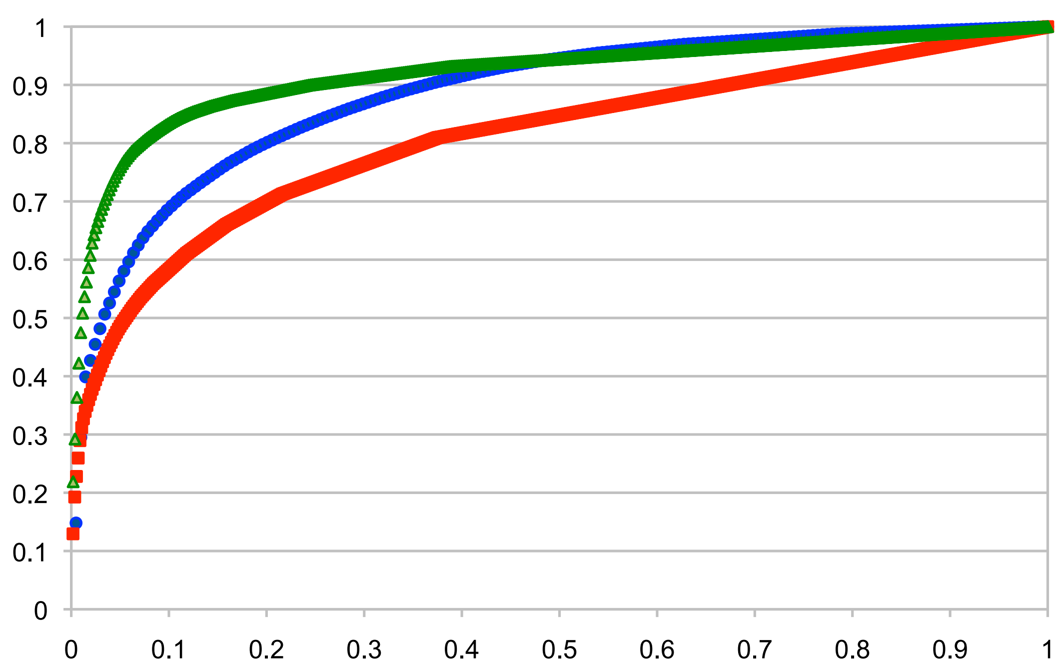


Brasero

commits

mails

bug report changes



Evince

Activity distribution
First study

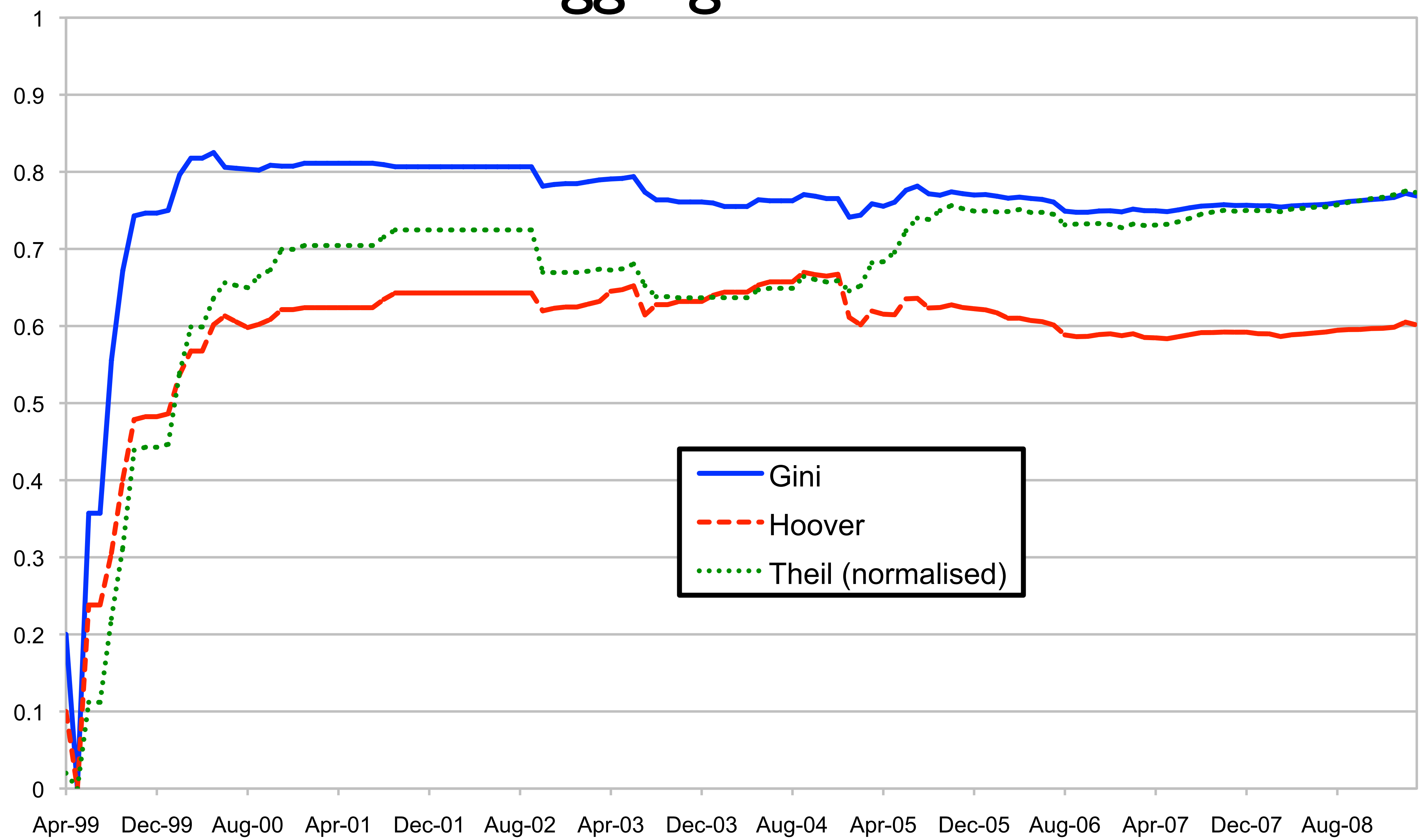
- Evidence of Pareto principle (20/80 rule)?
  - Most activity is carried out by a small group of persons
  - Typically : 20% do 80% of the job
- Distribution of code activity more unequal than mail activity

Activity distribution First study

- Analyse activity distribution *over time*
- Use *econometrics*
  - express inequality in a distribution
  - aggregation metrics:
    Gini, Hoover, Theil (normalised)
  - Values between 0 and 1
    - 0 = perfect equality; 1 = perfect inequality

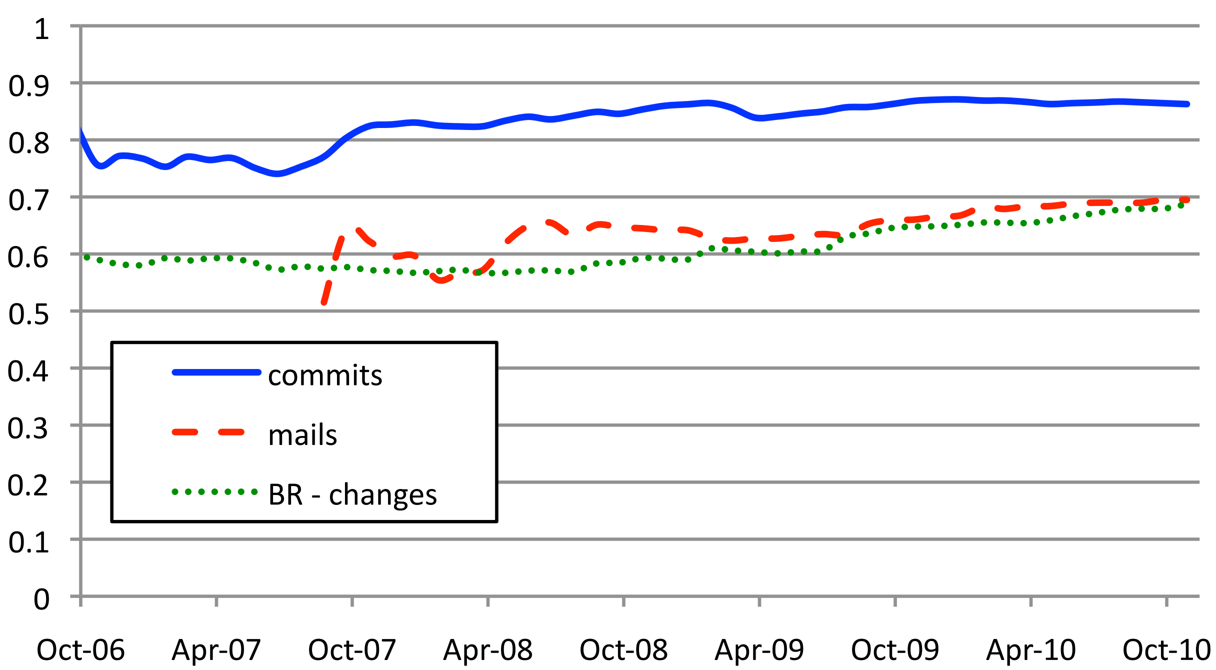
Activity distribution
First study

Evolution of aggregation indices for Evince

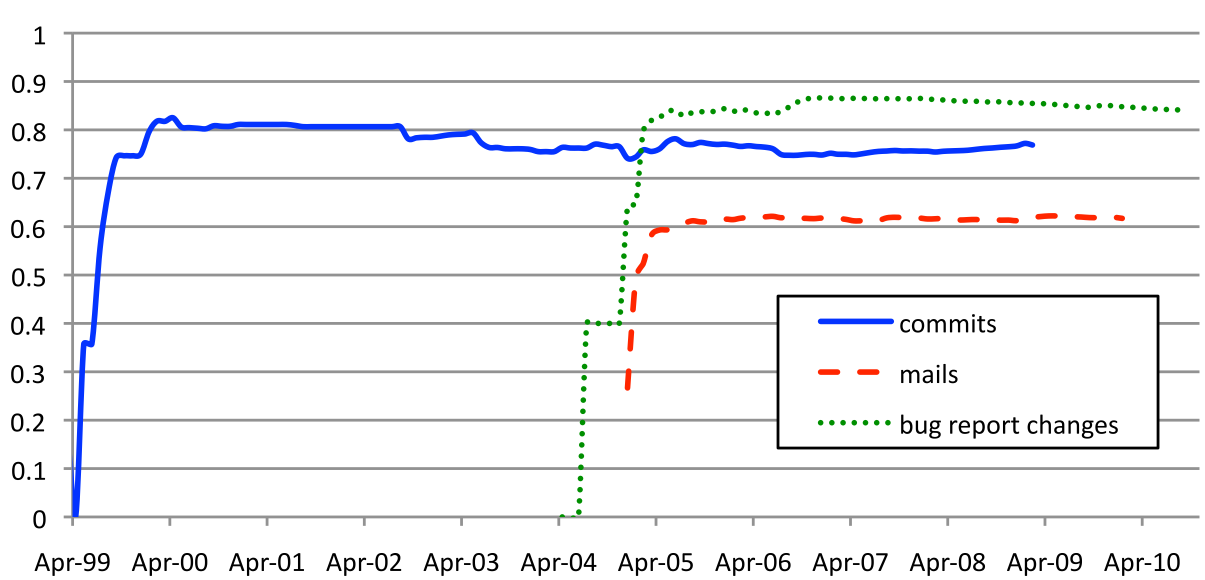


Activity distribution
First study

Evolution of Gini index



Brasero

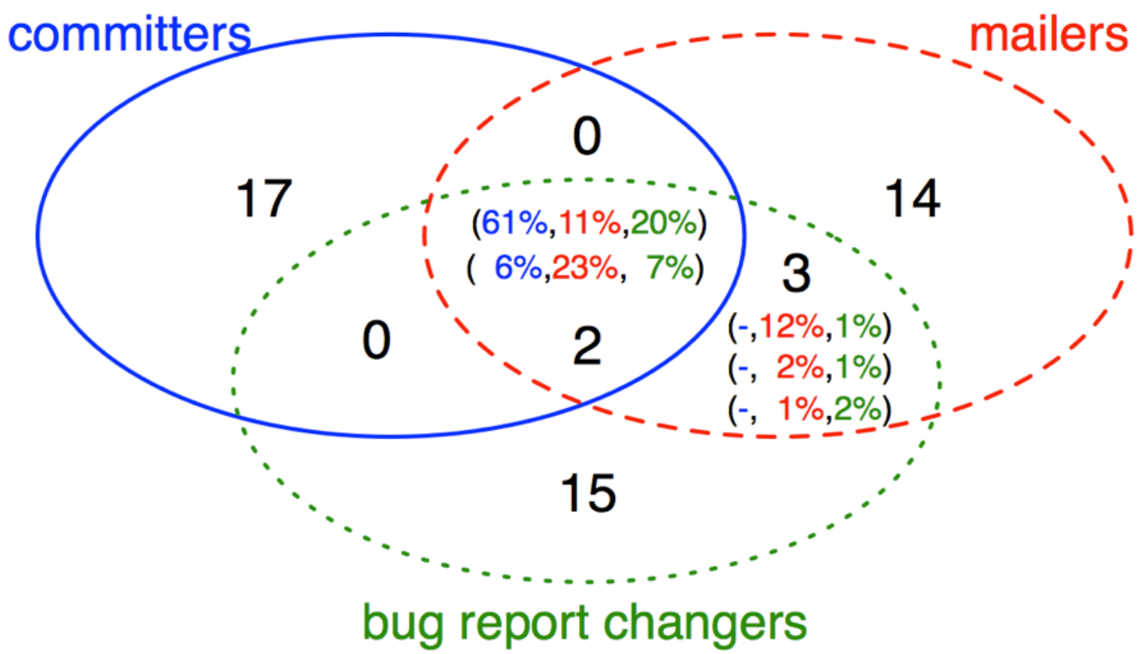


Evince

Activity distribution First study

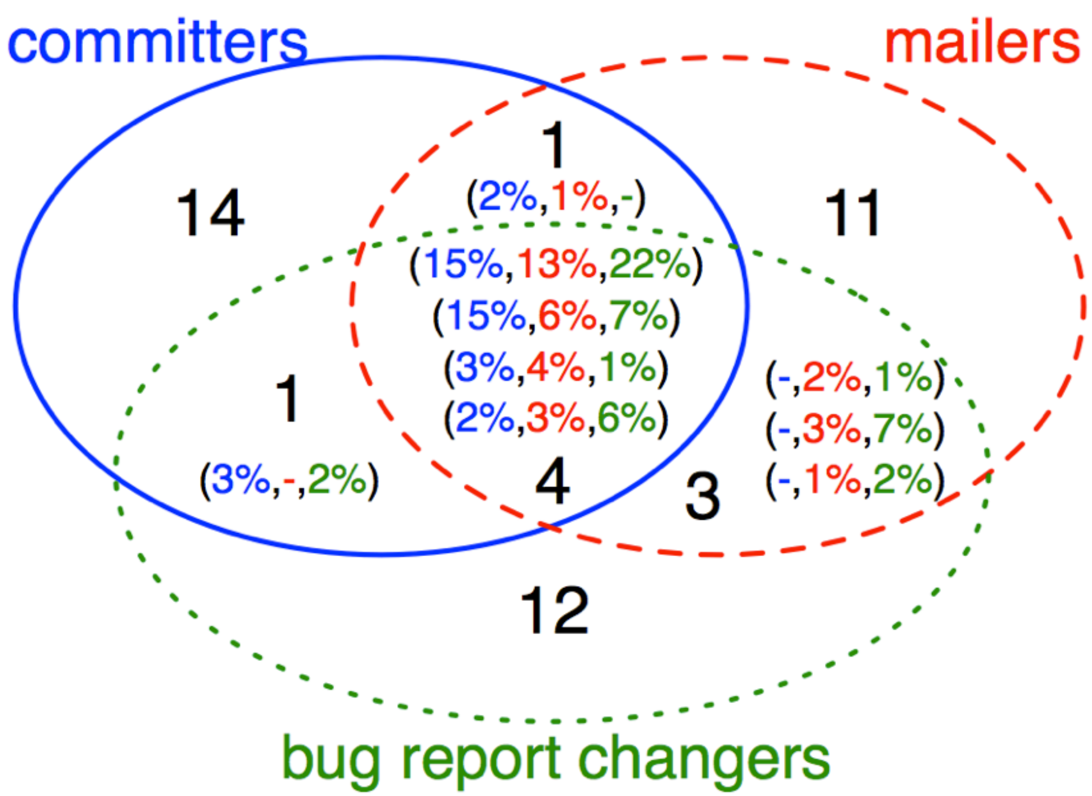
- Identifying core groups
  - Display Venn diagrams of most active (top 20) persons, according to each activity type (committing, mailing, bug report changing)
  - For each person, show percentage of activity attributable to this person
  - Take into account identity merges

Activity distribution



Identifying core groups

Brasero



Evince

Activity distribution
First study

- Conclusion
  - Activity distributions seem to become more and more unequally distributed
  - The Pareto principle is clearly present in studied projects
  - For Brasero and Evince, the activity is led by a few persons involved in 2 or 3 of the defined activities

Activity distribution First study

- Future work
  - Identify the type of statistical distribution
  - Use sliding window approach
    - detect impact of personnel turnover: ignore inactive persons, and discover new active persons
  - Automate detection of core groups
  - Study the evolution of core groups over time

UMONS
Université de Mons

Activity distribution
Second study

Faculté
des Sciences

Activity distribution
Second study

- Analysing fine-grained activities, restricted to source code repository
- Analyse contributors to a project based on types of activity they contribute to

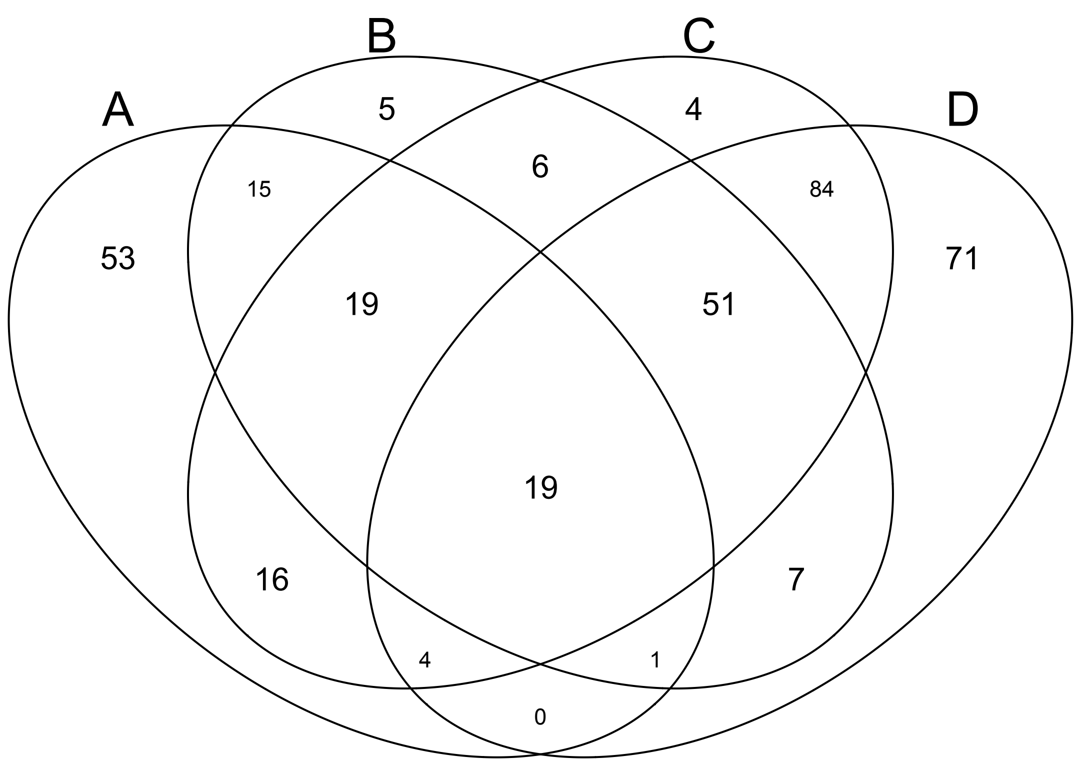
type of activity	file type
coding	.c, .h, .cc, .pl, .java, .ada,.cpp, .chh, .py
documenting	.html, .txt, .ps, .tex, .sgml, .pdf
translation	.po, .pot, .mo, .charset
multimedia	.mp3, .ogg, .wav, .au, .mid
...	

Activity distribution
Second study

- Example: Rhythmbox
  - overlap between 5 activity types

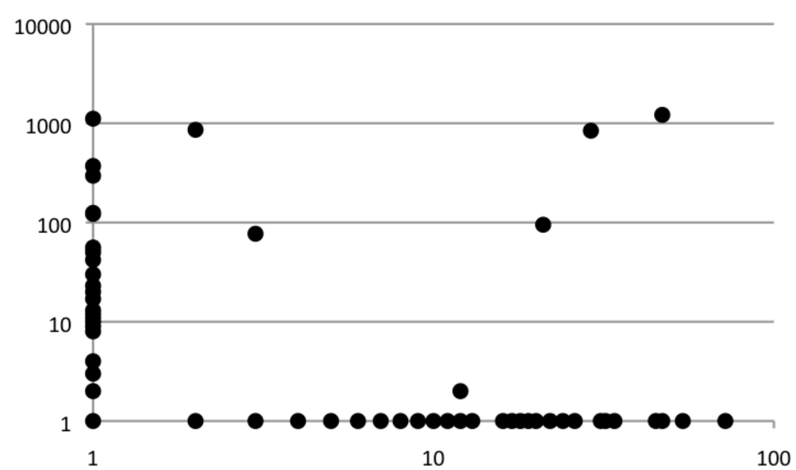
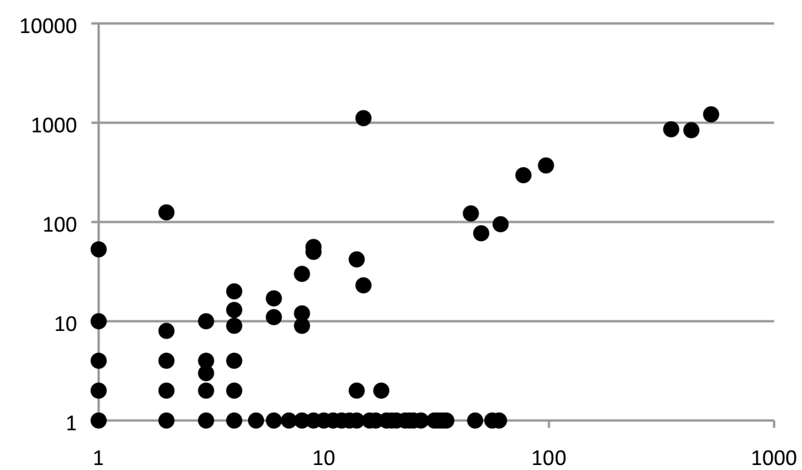
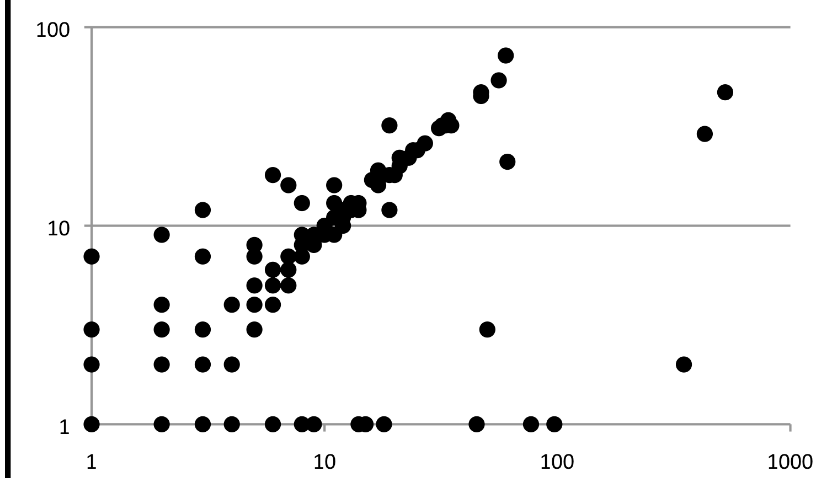
    A = code, B = build, C = devel-doc, D = translation
- Visualised using Venn diagram

  values represent number of persons involved in a particular activity (i.e., having committed at least one file of this type)



Activity distribution
Second study

Evince - scatter plots of correlation

		
coding versus translation	coding versus developer documentation	translation versus developer documentation

Activity distribution Second study

Evince - correlation matrix

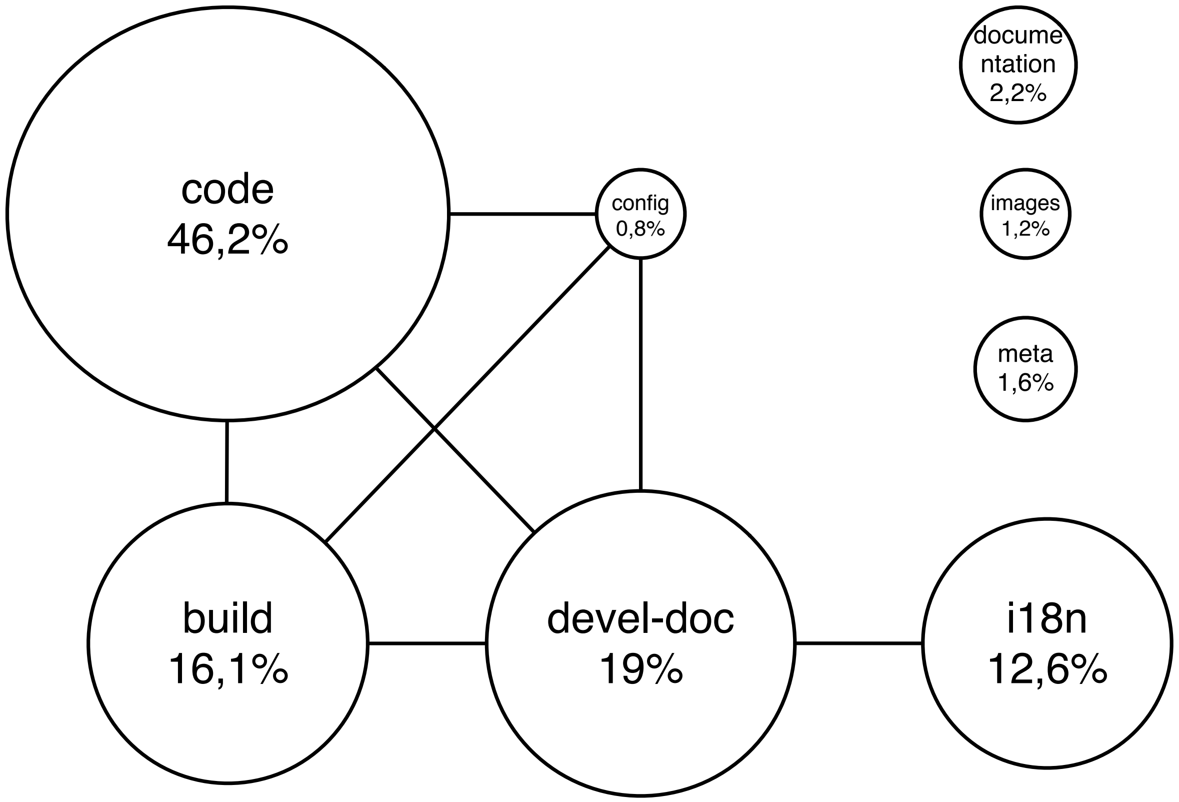
Correlations with p-value > 0.01 not shown;

Medium (>0.5) to strong correlations (>0.8) are coloured

Evince	documentation	images	i18n	ui	multimedia	code	meta	config	build	devel.doc	other
documentation	1	0,14715687									
images		1	0,16954996	0,20213002	0,23079965	0,41266264		0,4596056	0,4757661	0,44918361	0,19411888
i18n			1		0,15575507	*0,13010625*		0,17912559	0,15815786	0,30906678	
ui				1	0,16142962	0,56632106	0,31157703	0,54728747	0,55080516	0,51066149	0,16779495
multimedia					1	0,21869572		0,41933948	0,33361174	0,36632992	0,20712209
code						1	0,30763128	0,8242121	0,90169181	0,87365497	0,4087696
meta							1	0,31914846	0,54909454	0,2434247	0,61723848
config								1	0,89979553	0,95124778	0,43475402
build									1	0,90540444	0,58183399
devel.doc										1	0,41281866
other											1

Activity distribution Second study

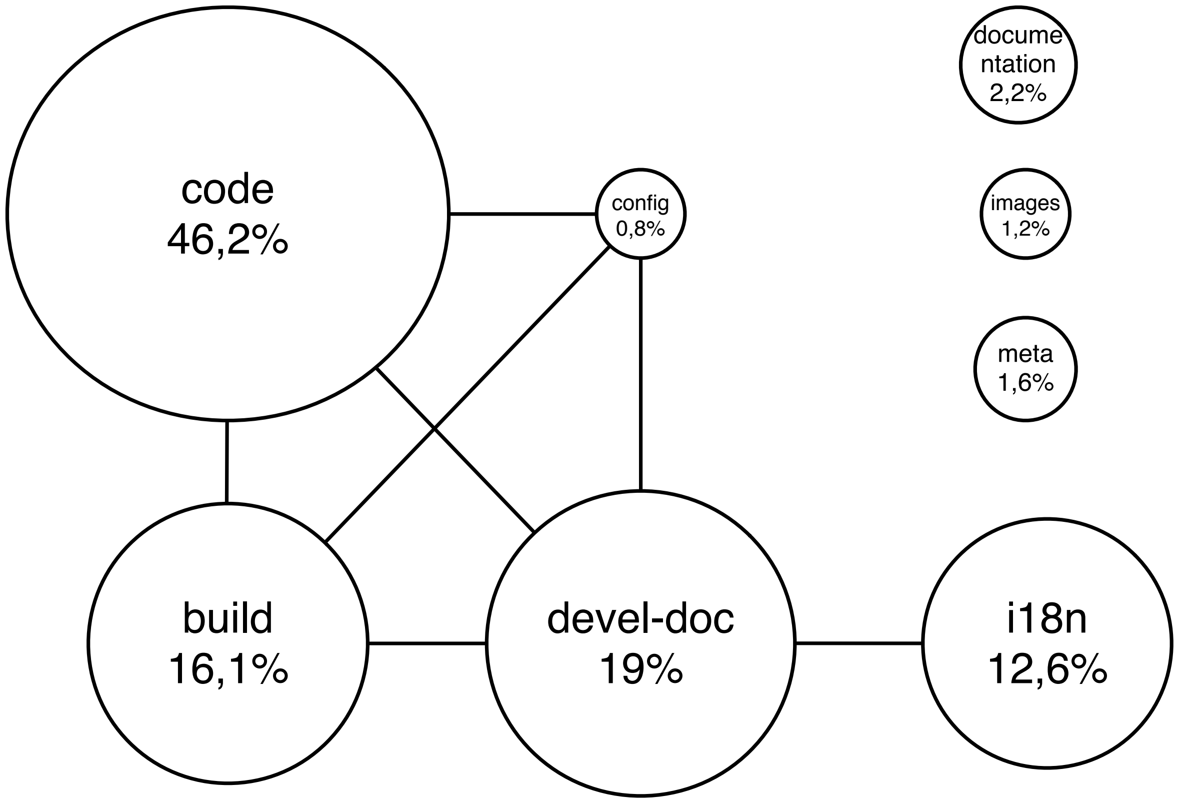
- Correlation graph for Evince
  - Nodes represent relative amount of activity of each type
  - Edges represent a statistically significant ($p<0.01$) high correlation (>0.8)



Edges with weak or not statistically significant correlation, and activity nodes <0,5% are not shown.

Activity distribution Second study

- Evince: Interpretation of results
  - The activities of building, coding and development documentation are done by the same group of persons
- Translators (i18n) are also involved in development documentation but not in coding
- Other documentation is largely independent from these activities



Activity distribution Second study

- Conclusion
  - Some activities are done by same group of persons, other activities are largely independent
- Future work
  - Study and compare the activity patterns on other projects
  - Study how the separation of activities influences the process
  - Compare activity distribution of different types of distributions
    - E.g. translators have a more equal distribution of work than coders

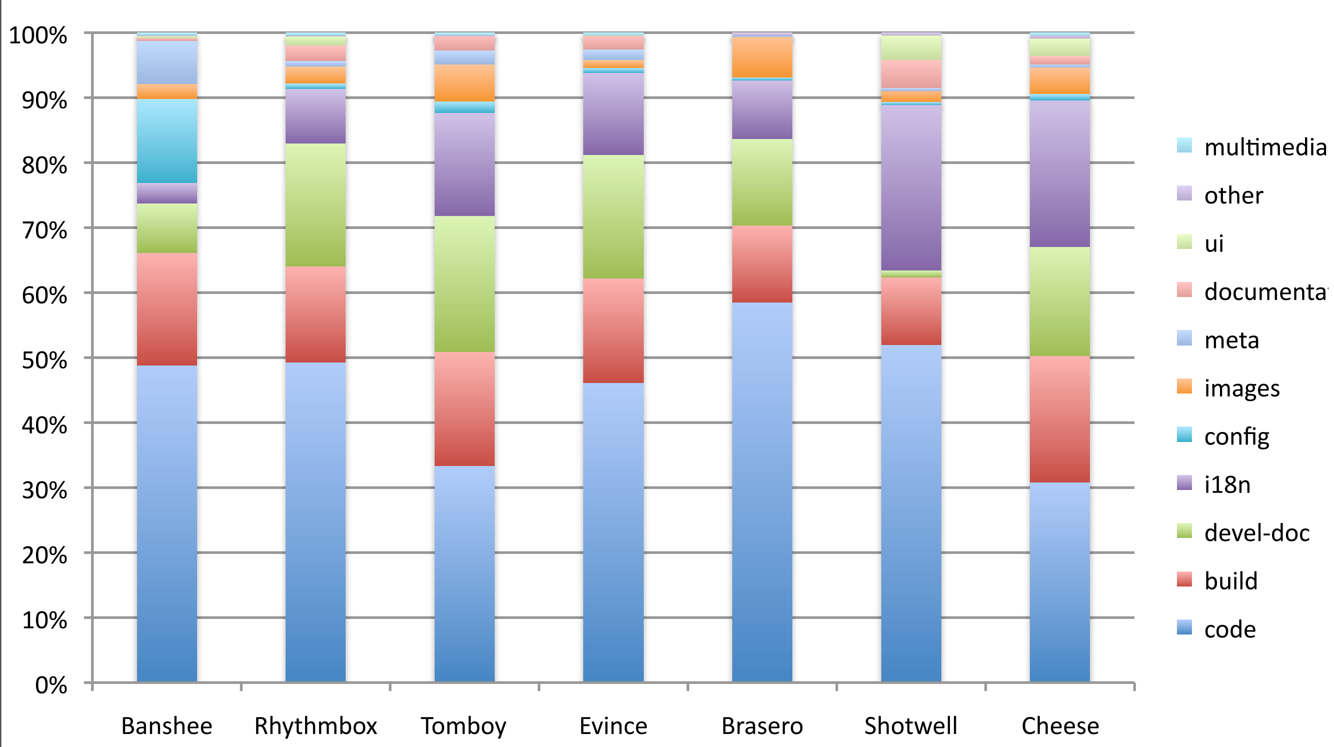
Activity distribution Third study

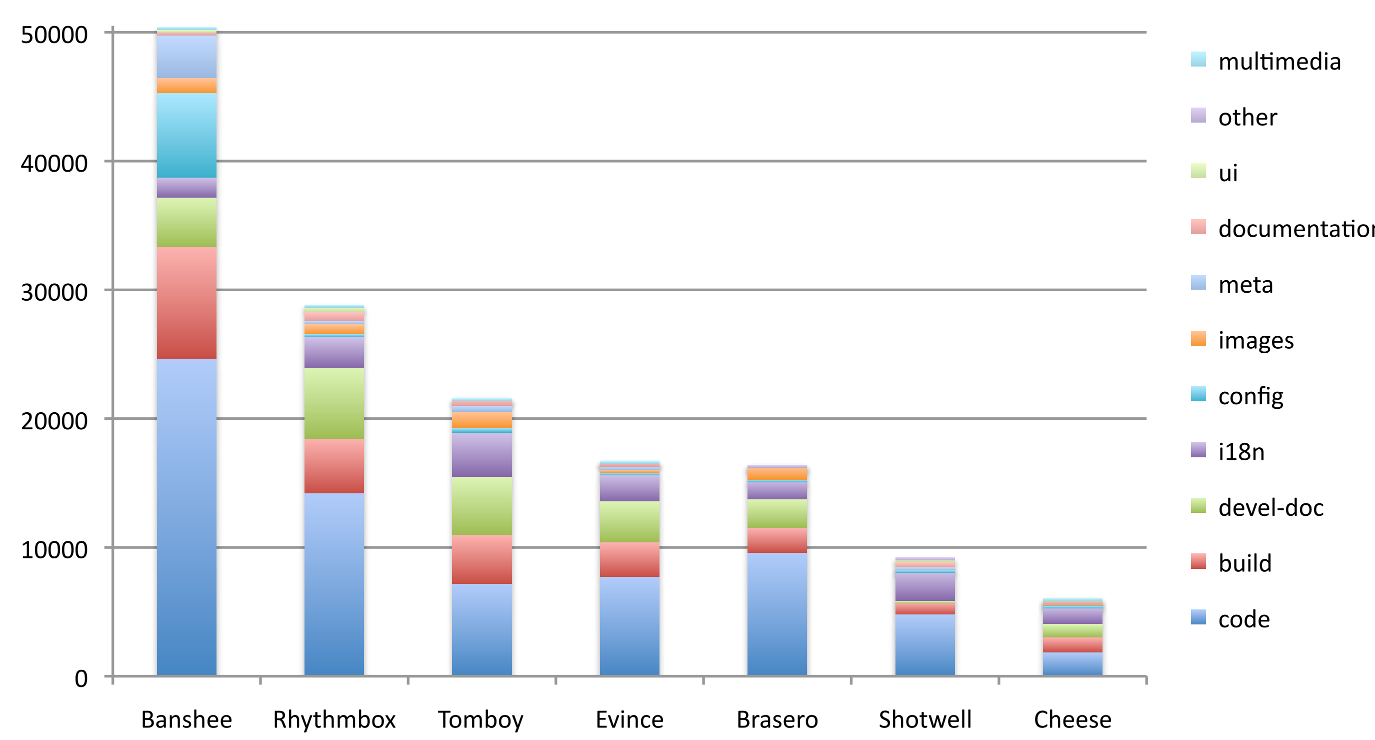
- Study the activity of persons across software projects in the GNOME ecosystem
- Case: Large long-lived GNOME projects using Git

ID	Project name	Authors	Years	Files
A	Banshee	268	5,9	2700
B	Rhythmbox	364	8,9	937
C	Tomboy	290	6,6	766
D	Evince	381	12,1	699
E	Brasero	193	4,2	797

Activity distribution Third study

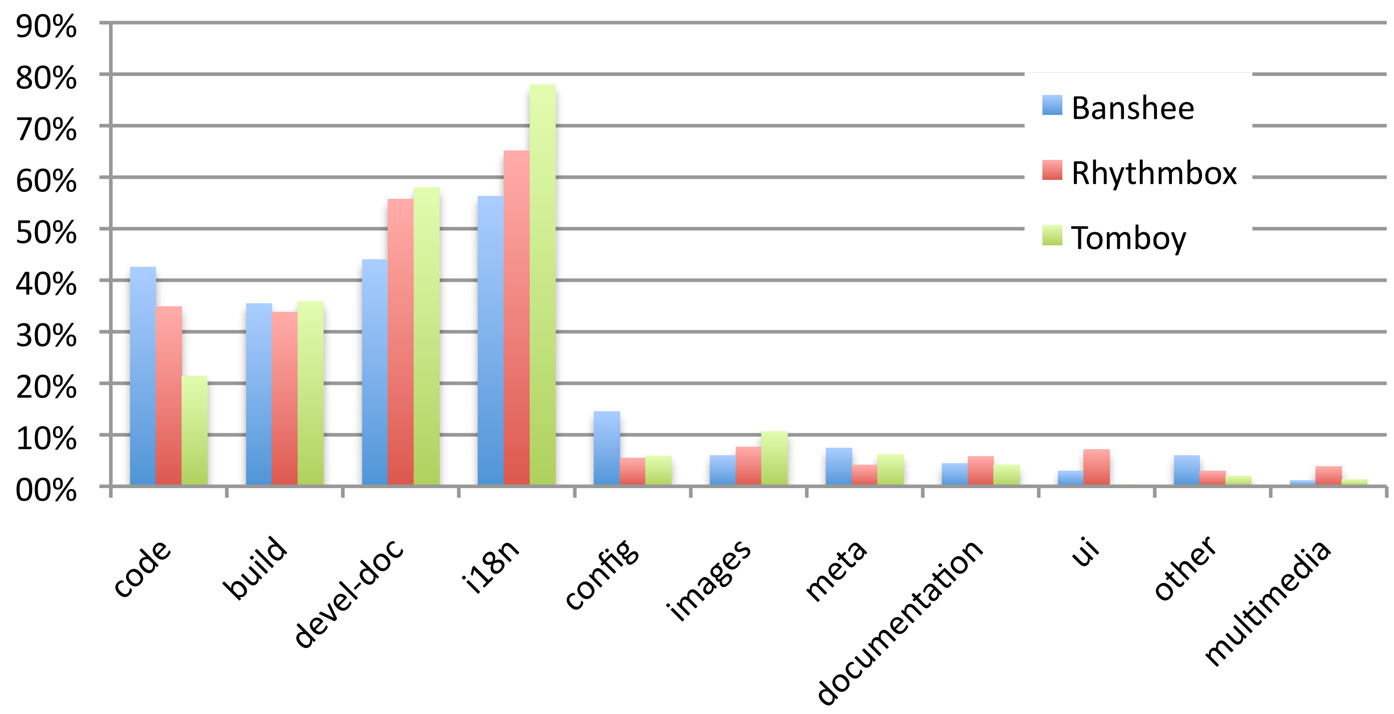
- How are activities distributed in different GNOME projects?
  - counted in terms of number of files modified
  - *code* files are most frequently committed, followed by *build*, *development documentation* and *translation* files





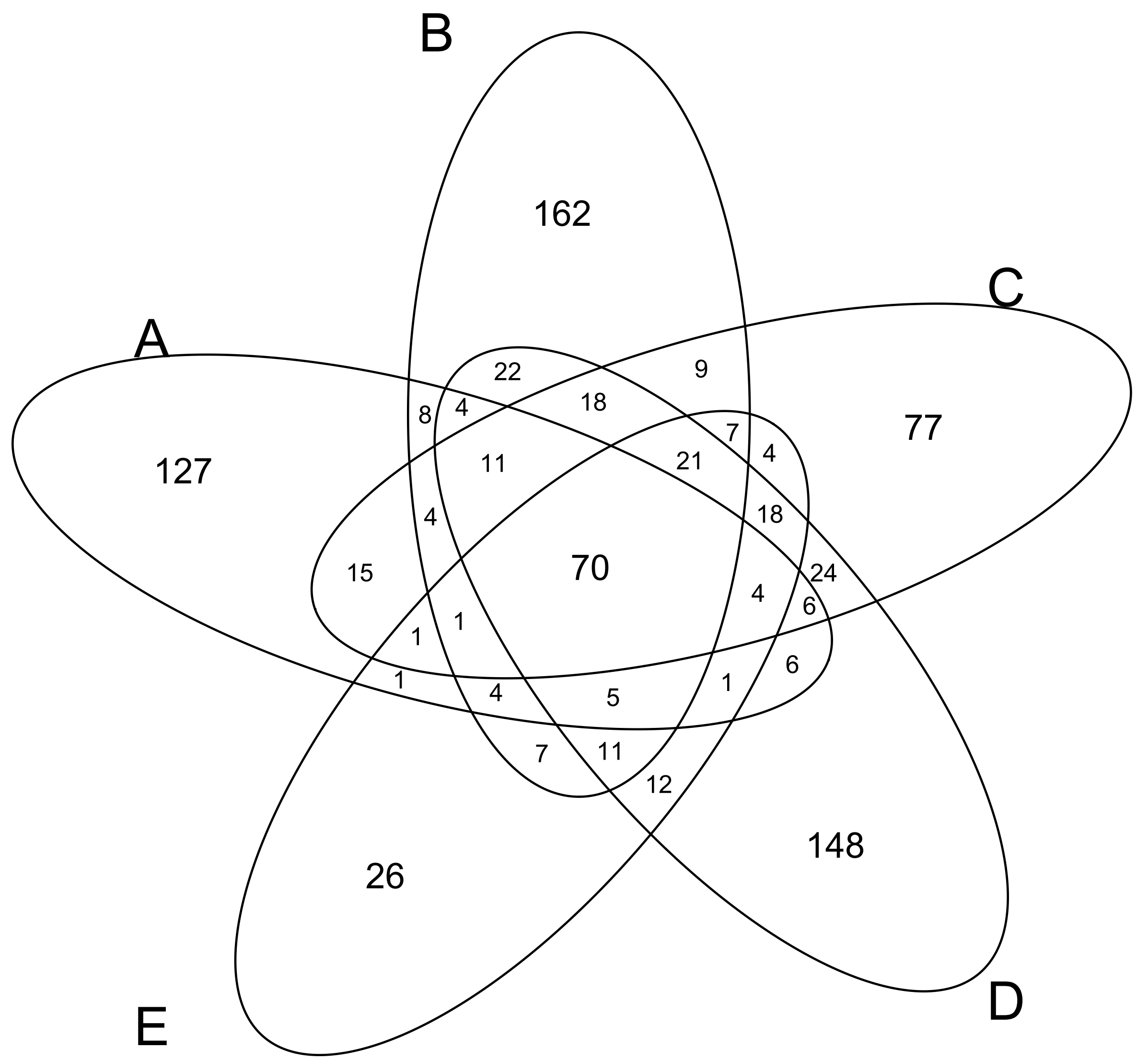
Activity distribution
Third study

- How are activities distributed in different GNOME projects?
    - counted in terms of number of persons involved in modifying files
    - most persons are involved in *translations*,
      followed by *development documentation*,
      followed by *code* and *build*



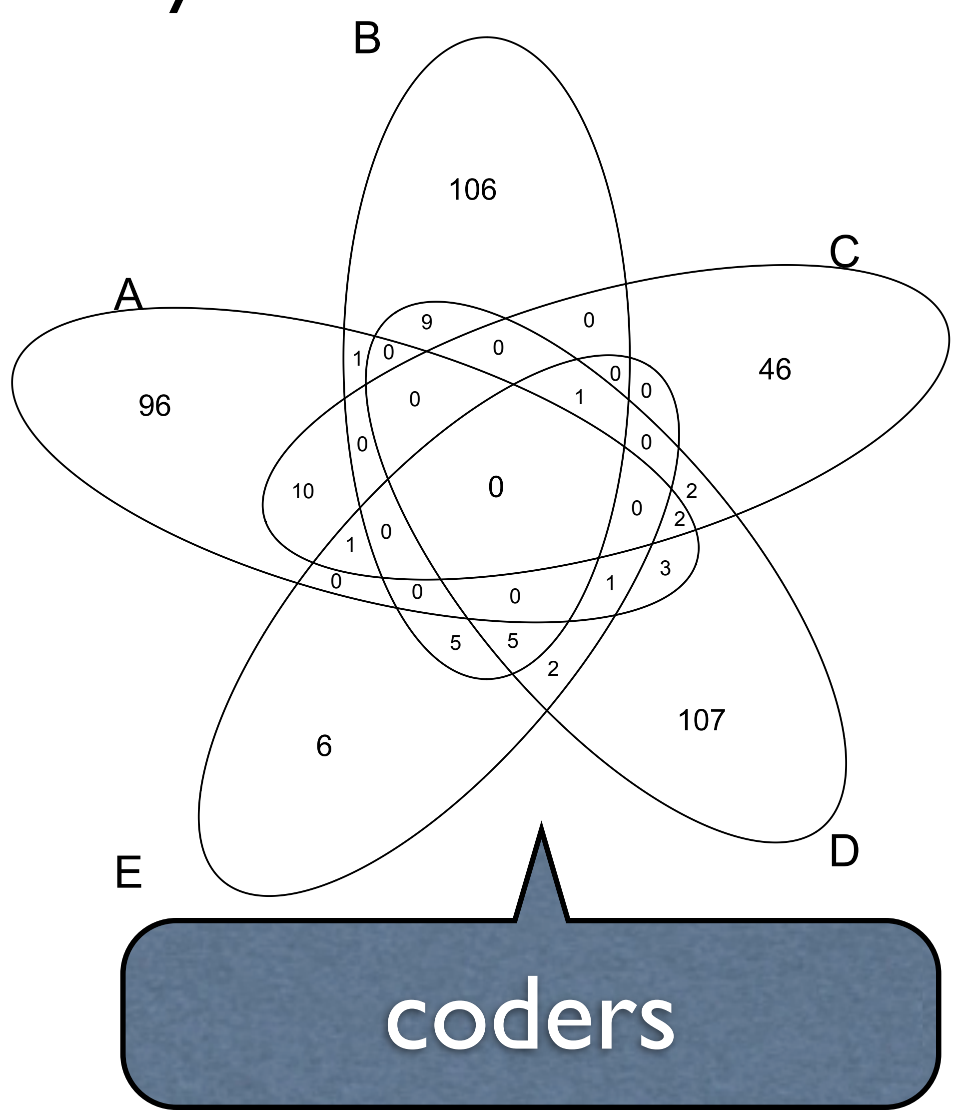
Activity distribution
Third study

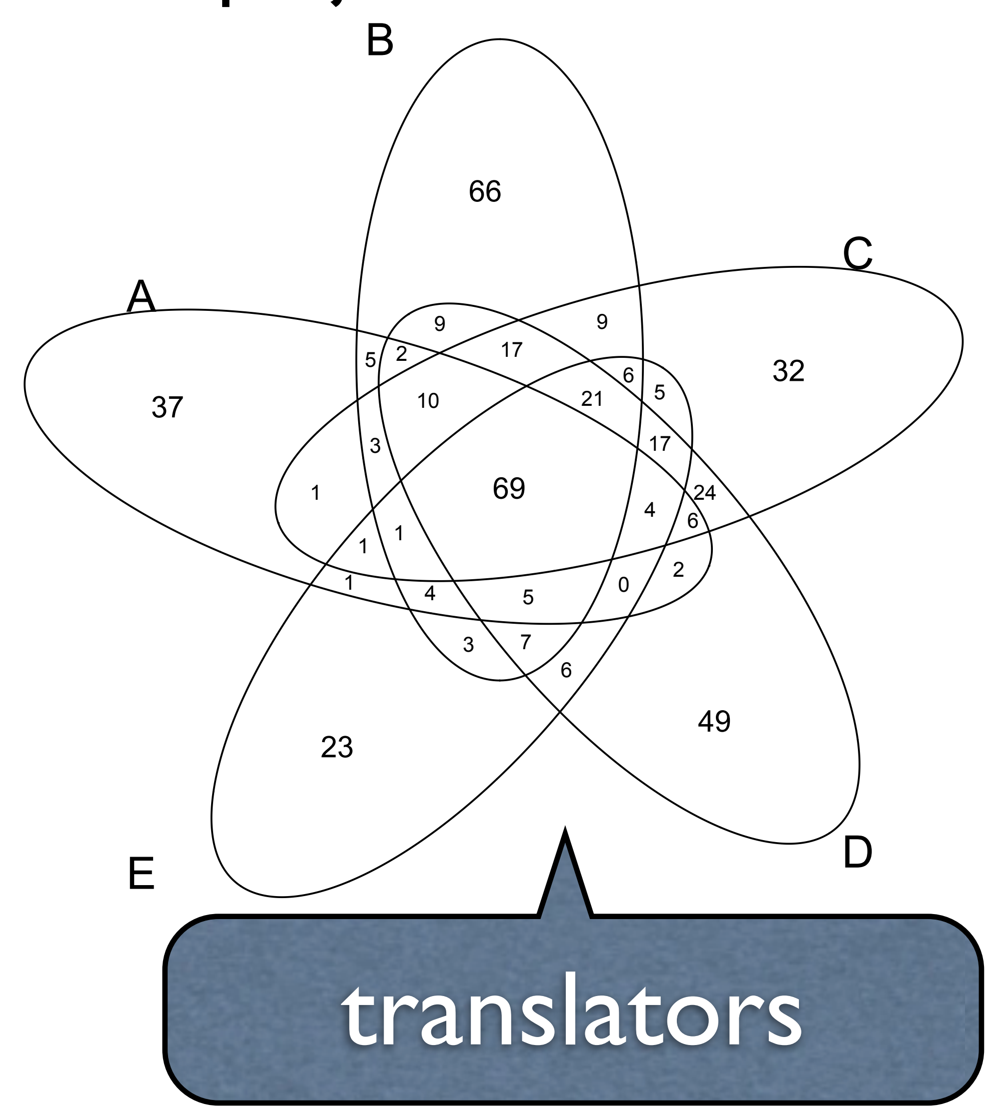
- How many persons are contributing to more than 1 GNOME project?



Activity distribution
Third study

- Which types of activities are carried out by persons involved in multiple projects?
  Mainly translators. Coders tend to stick to one project





Activity distribution Third study

- Work in progress
  - Study activities of developers across different projects
  - How, when and why do developers contribute to different projects?
    - Simultaneously
    - Over time

Thank you