Visualizing Work Process in Software Engineering with Developer Rivers

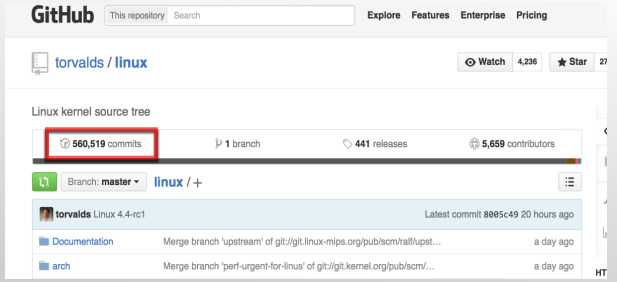
Michael Burch, Tanja Munz, Fabian Beck, and Daniel Weiskopf
VISUS, University of Stuttgart, Germany

Presenter: Arthur Sun

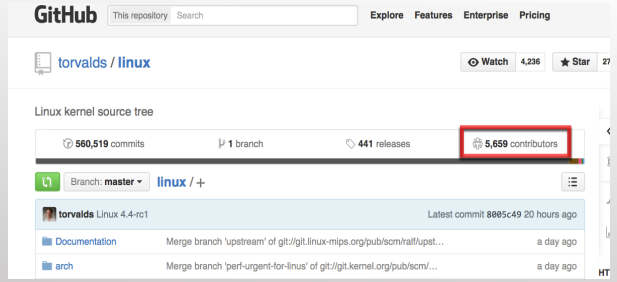
Outline

- What's the current problem for large software projects
- What's the previous solution for large projects InfoVis
- What the paper presents
  - DataSet
  - InfoVis Encoding Technique
  - Visualization Method
  - Sample Usage
- Future Improvement

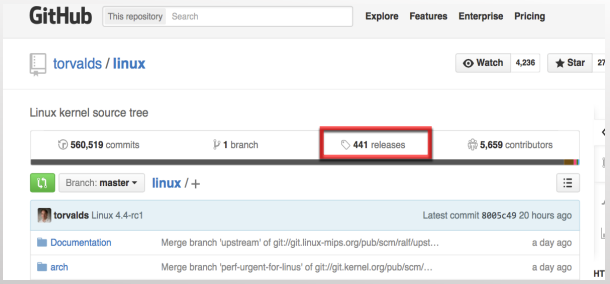
Large open-source projects: 560,519 commits

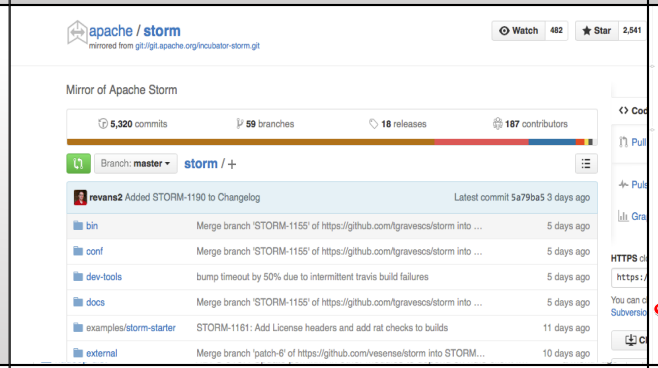


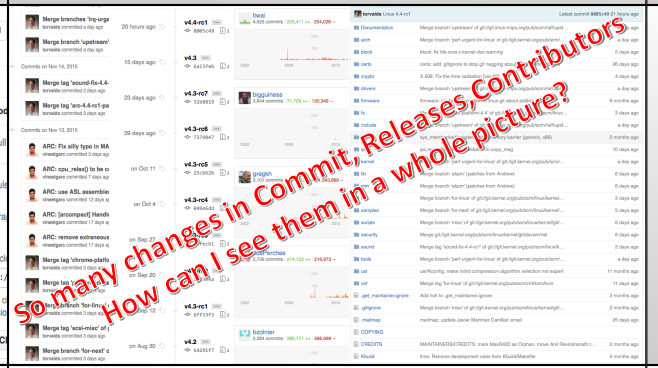
Large open-source projects: 5659 contributors



Large open-source projects: 441 releases







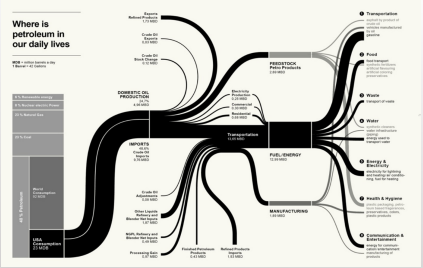
What's do we want

A whole picture of the overall progress of extreme large software engineering project proceeding with time frame in detailed visualization for major participants, their contribution to respective work, how much amount of work they did and their work change

Previous work – Sankey Diagram

Sankey diagrams are a specific type of flow diagram, in which the width of the arrows is shown proportionally to the flow quantity.

Problem:
No Time Frame



Previous work – Gantt Chart

ID	Task Name	Predecessors	Duration
1	Start		0 days
2	a	1	4 days
3	b	1	5.33 days
4	c	2	5.17 days
5	d	2	6.33 days
6	e	3,4	5.17 days
7	f	5	4.5 days
8	g	6	5.17 days
9	Finish	7,8	0 days

A Gantt chart is a type of bar chart that illustrates a project schedule. Gantt charts illustrate the start and finish dates of the involved tasks. Modern Gantt charts also show the dependency

Problem: 1. Doesn't show how many people/resources involved in project
2. Don't have a whole picture about the project

What's the author propose

- A graph flow which can not only show the interconnection of different modules of development along with the timeframe but also the programmer who took part in the whole project with vivid color to show difference

Visualization Technique

DataSet: Developer Activity Model

Encoding: Develop River for Time-Varying Developer Activities

Dataset: Developer Activity Model

1. Abstract commit as c, time as t, developer name as d, files as f, all files as F, file modules hierarchy as H

Partition sequence of commits into equally-sized intervals for each interval of commits and every module

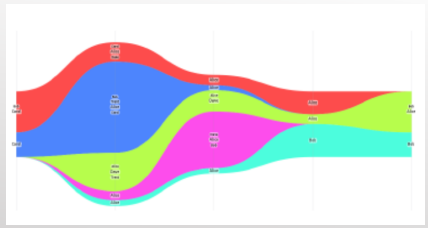
Calculate individual developer activity of their files for each module and reach module specific developer activity

Calculate weighted transition matrix for each developer Mi = Mat((I+1)*(I+1))

Calculate the weighted transition matrix for all developers by summing up Mi

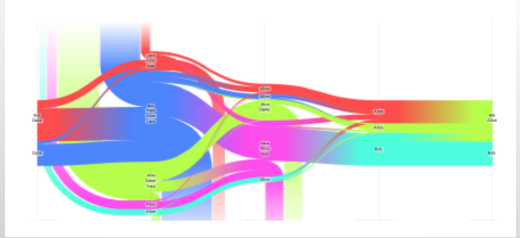
Paper didn't show how to map real data into Activity Model Matrix

Developer Rivers



Mapping Activity Model Matrix into Develop Rivers without intersection

Developer Rivers

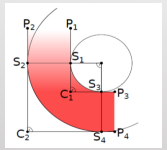


Mapping Activity Model Matrix into Develop Rivers with intersection

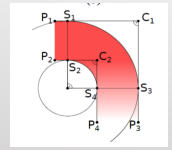
Developer Rivers Curves

1. Transition: how developers change their behavior between different module groups using cubic Bezier Curves
2. Transition color is a linear gradient from color of start module to target module

Paper didn't show how to link Matrix Data with Bezier curve creation

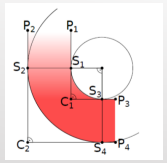


Influents
Developer join current step

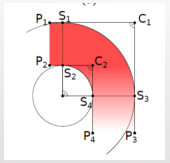


Effluents
Level the main river

Developer Rivers Curves



Effluents
Leave the main river



Effluents
Leave the main river

Diagram

- Inflow/Outflow: A transition from or to the outside of the diagram identifies a developer enter or leaving the project
- Constant Flow: An intra-transition with a constant width indicating a group of developers constantly working on the same module
- Growth/Decline: An intra-transition with an increasing or decreasing strength hints at a group of developers that keep working on a module but with changing total effort
- Split/Merge: A module that is split into or merged from multiple flows shows a qualitative change of developer activity (i.e., developers' relative focus switches between modules). While at least one inter-transition is required for this pattern, one of the flows can be an intra-transition.
- Exchange: A pair of intra-transitions connecting two modules in opposite directions at the same time is a specific qualitative change of activity: some developers move between the two modules in both directions.

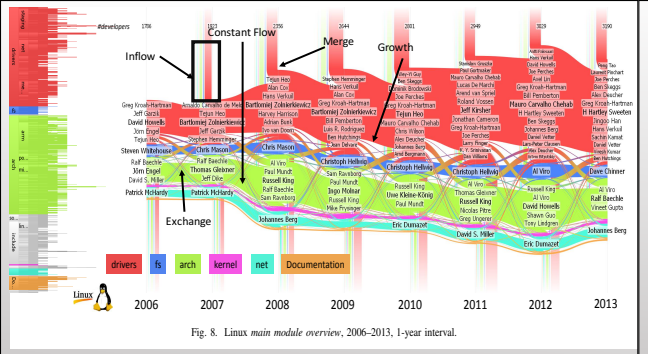
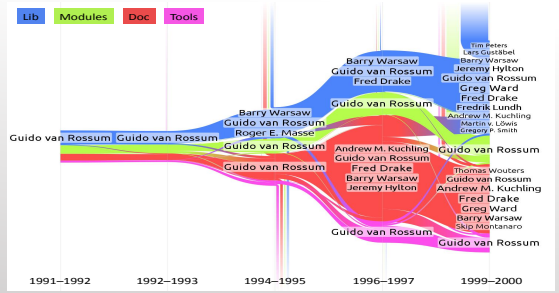


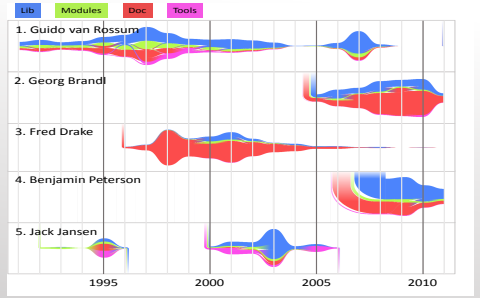
Fig. 8. Linux *main module overview*, 2006–2013, 1-year interval.

Visual Patterns

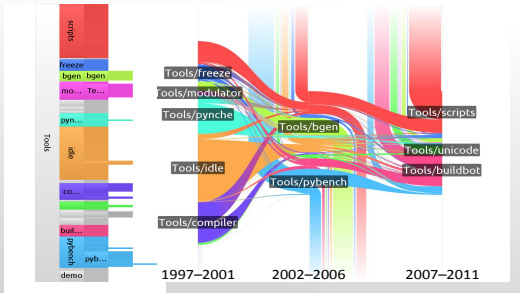
- Main Module Overview
  - Consists main directories, developers and their contributors
- File Type Overview
  - Automatic definition of modules by file types
- Developer Sparklines
  - Highlight top 5 star developer contributing most to the whole project
- Subsystem Details
  - Modules in a subdirectory of the system shows details of a specific system



Python main module overview



Python Developer Sparkline of top 5 developers



Python Subsystem details of Tools Directory

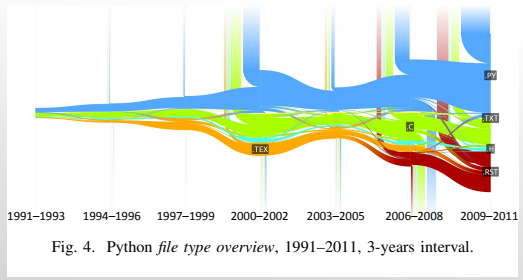
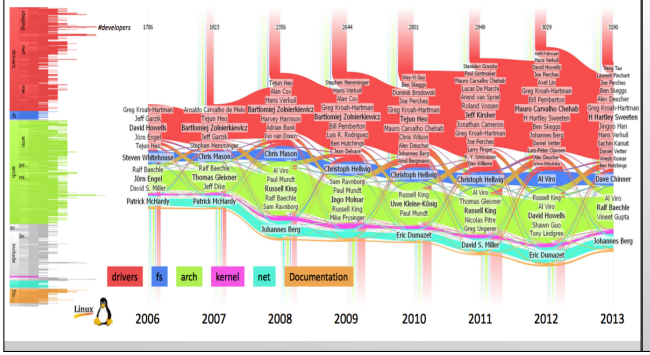
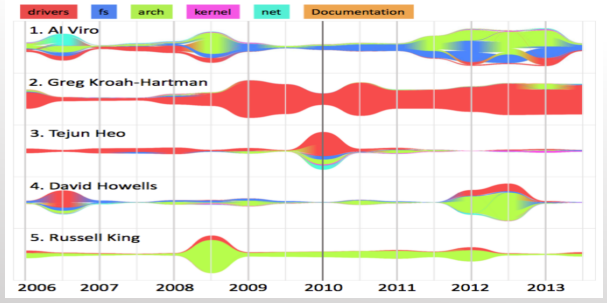


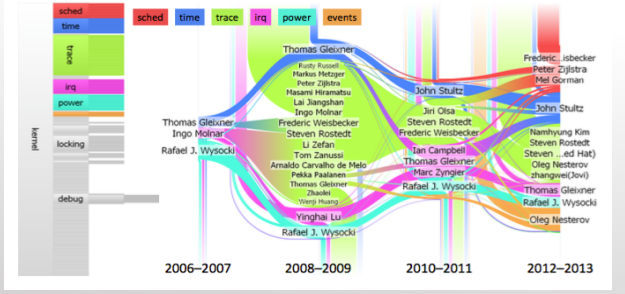
Fig. 4. Python *file type overview*, 1991–2011, 3-years interval.

Python file type overview





Linux Developer Sparkline of top 5 developers



Linux Subsystem details of Tools Directory

Future Improvements

- Show us how do the author organize the data(Data->Matrix)
- Show how to transfer the data into influents and effluents (Matrix->Influents)
- Provide tool ready for practitioners who can use developer river directly(No description about how to tackle the dataset)
- Distinguished colors may be up to 10 colors, otherwise graph may be hard to see
- Transfer the way to study software engineering research into social-technical aspects of engineering research