Evaluating **Decision-Making Processes** of **Software Enhancement Proposals**

Author
Tim Raaymakers
Supervisor
Dr. Diomidis Spinellis
EEMCS, Delft University of Technology, The Netherlands

1. Background

Enhancement proposals are a common way to track, maintain and evaluate enhancements to OSS projects.

As part of the **decision-making (DM) process**, proposals go through various stages before being accepted and implemented.

Previous case studies of *Python Enhancement Proposals (PEPs)* suggest that DM processes might not be documented properly.

A lack of process transparency can lead to a lack of **trust** and **community engagement**.

2. Objectives

Create a **dataset** covering 10 SEP projects containing **proposal features**, **revisions** & **discussions**.

Replicate the results found by previous Python case studies.

Uncover, **compare**, and **provide feedback** on the DM process of 6 SEP projects.

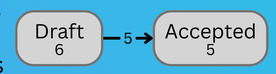
Establish a method for other OSS communities **to extract** and **evaluate** their DM process.

3. Approach

Compare **normative** and **practical** DM processes.

The DM process as a **weighted directed graph**:

- *Proposal states* as vertices
- *State transitions* as edges
- *Number of visits* as weights

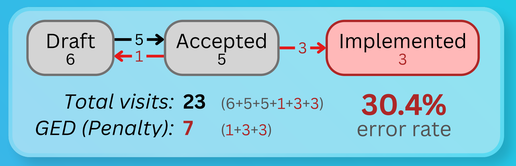


The **normative DM process** can be extrapolated from prescribed methods.

The **practical DM process** gets revealed by state changes between revisions.

Process compliance is measured using a modified version of *Graph Edit Distance (GED)*.

Invalid transitions get penalized based on number of visits.

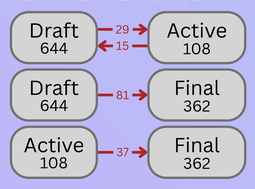


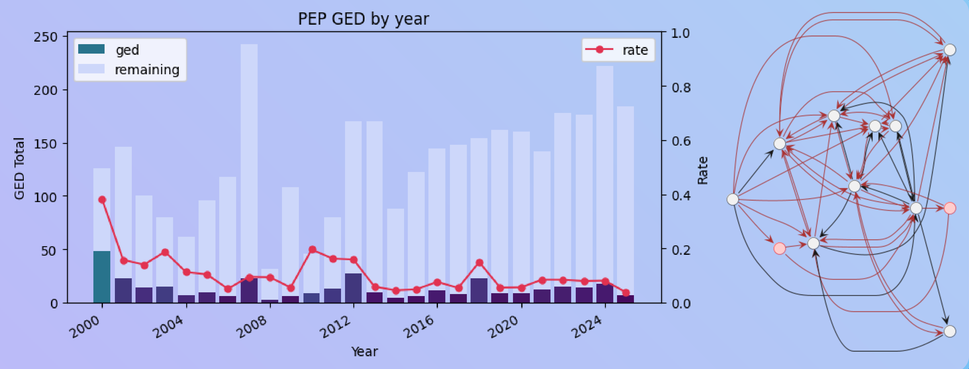
Changes in error rate can be plotted over time.

PEPs (Python)

+ Clear documentation
+ Consistent GED rate
− Lot of noise

To be documented:

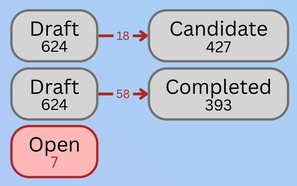


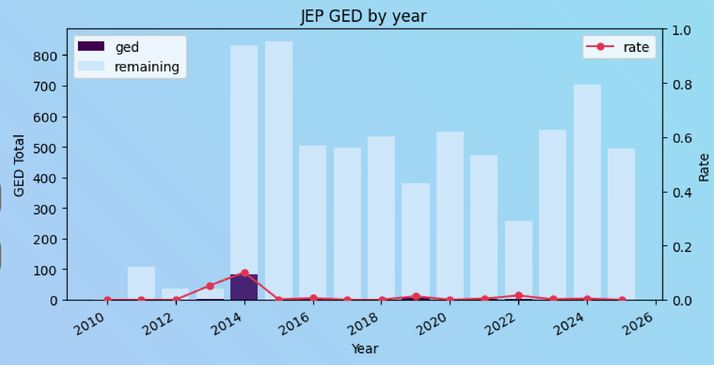


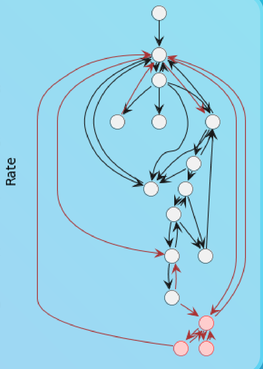
JEPs (Java)

+ Clear documentation
+ Consistent GED rate

To be documented:



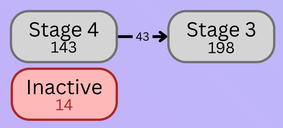


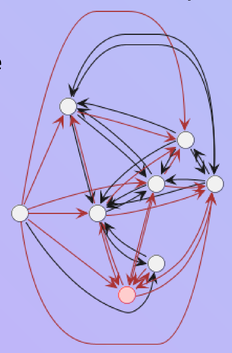


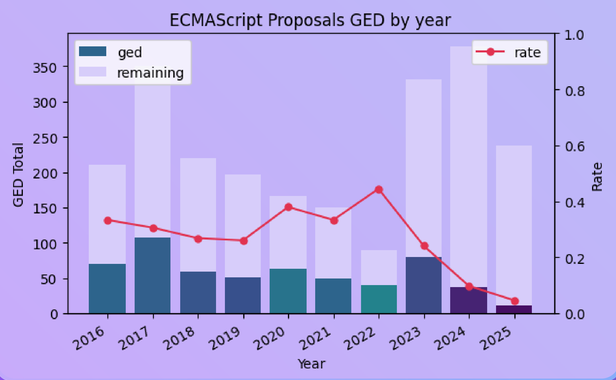
ECMAScript Proposals (JavaScript)

− Ambiguous documentation
+ Downwards trending GED rate
− Lot of noise

To be documented:



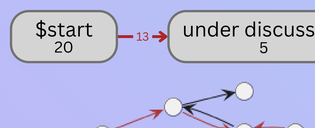


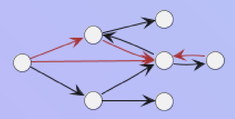


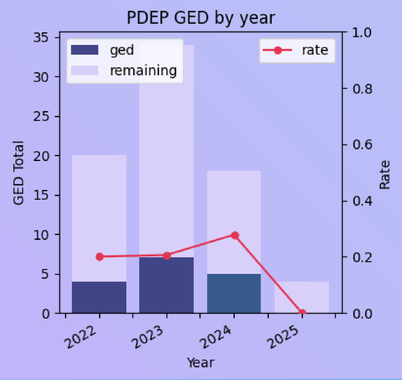
PDEPs (Pandas)

+ Clear documentation
+ Consistent GED rate
− Few datapoints

To be documented:

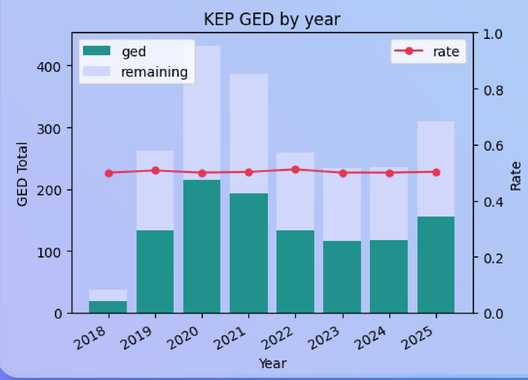


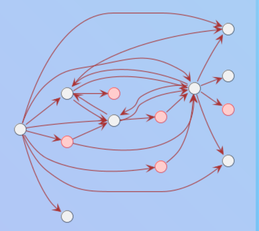




KEPs (Kubernetes)

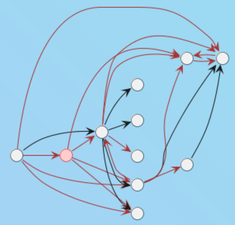
− Ambiguous documentation
+ Consistent GED rate

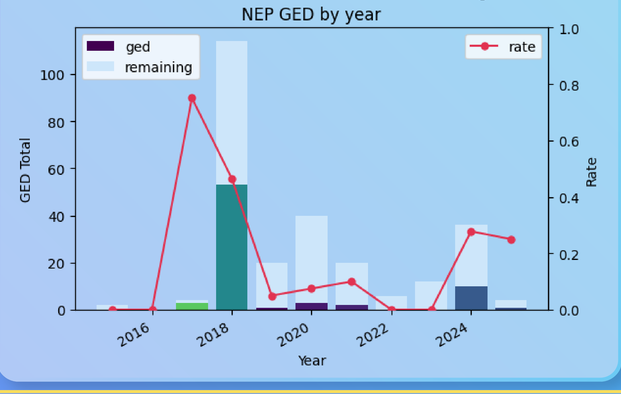




NEPs (Numpy)

+ Clear documentation
− Lot of noise
− Few datapoints





4. Reflection

Prerequisites

- Statuses must be representative of DM process.
- Requires years of data to discern patterns.

Results

- Documentation can be interpreted in multiple ways.
- Does not capture full complexity of DM process.

Applications

- Lightweight and quick.
- More robust alternatives exist.
- Can provide real-time validation.

Future Research

- Expand list of projects.
- Branch out into different contexts, such as software dev. or organizational DM.