H1: The roles of core versus peripheral developers: Greater participation of peripheral developers in open-source projects are associated with greater awareness about open-source products (Setia et al. 2012).

IndepVs:

The degree of participation of peripheral developers (NoOfPerD) is the primary independent variable used in this research, and it is a monthly count of the peripheral developers working on each project.

The number of core developers (NoOfCord) was another variable used in our research. It was measured as the count of developers contributing more than 12% of code to the project.

DVs:

Product Quality of an OSS product is examined as the extent of quality assessment and quality enhancement. These are measured as the number of bugs reported (BugsRep) and bugs fixed (BugsFix), respectively, for each product during the month.

Open-source product awareness and adoption in terms of the number of product watches (PagView) and the number of product forks (Dwnld).

Method:

Hierarchical linear modeling (HLM): for different maturity levels; TO DO: define stages in the ETH project, i.e., Alpha, Beta, and Mature. Setia et al. 2012 takes them from OSS platfrom, in our case, we could rely on the ETH price performance or platform adoption.

[More info is here: https://towardsdatascience.com/hierarchical-linear-modeling-a-step-by-step-guide-424b486ac6a3](More%20info%20is%20here:%20https://towardsdatascience.com/hierarchical-linear-modeling-a-step-by-step-guide-424b486ac6a3)

Core developer (Is\_core = 1):

Step 1. Rolling window: developer contributed *PercChangeCode %*  to the project within last 30 days (1M core developer)

Step 2. Set threshold 12% of code contribution to the project. Check other values of % for robustness. Create const *PercChangeCode*

Script sample:

From (select repo, user, date, rolling\_sum\_30\_d(activities) by user, repository) t1

inner join (select repo, date, rolling\_sum\_30\_d( (activities) by repository) t2 on t1. Repo=t2.repo & t1.date=t2.date

\* rolling\_sum\_30\_d – look up sql function on the web

Peripheral developer (Is\_core = 0):

* All other

Timeline:

* 201508-201509 (GitHub Query DB)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Repository | Date | User\_id | Is\_core (1/0) | Num activities\_by user | Num\_activities\_total\_in\_repo\_past\_30 days |
| A | 2015-09-01 |  |  |  |  |
| … | … |  |  |  |  |
|  |  |  |  |  |  |
| A | 2015-09-30 |  |  |  |  |
| B | 2015-09-01 |  |  |  |  |
| … | … |  |  |  |  |
|  |  |  |  |  |  |
| B | 2015-09-30 |  |  |  |  |

\*\*

Alina’s script 2023-03-19

DECLARE PercChangeCode INT64;

SET PercChangeCode = 12;

SELECT t1.repo.name as GhRepo, ## note that I added the table identificators (t1.) to specify from which tables the fields are taken

cast(t1.created\_at as date) as date1,

t1.actor.id as user,

COUNT(t1.actor.id) as num\_activities,

SUM(COUNT(t1.actor.id)) OVER (ORDER BYcast(t1.created\_at as date) ROWS BETWEEN 29 PRECEDING AND CURRENT ROW) AS rolling\_sum\_30\_d\_user ## renamed the field to avoid confusion.

## Next part to calculate the % of activities by user in total pool of activities in the repository

, t1.rolling\_sum\_30\_d\_user/t2.rolling\_sum\_30\_d\_repo as percent\_activities\_user

, (case when percent\_activities\_user>= PercChangeCode then 1 else 0 end) as is\_core\_developer

FROM (SELECT \* FROM `githubarchive.month.\*` where \_TABLE\_SUFFIX between '201508' and '201509') t1 where t1.repo.name like 'ethereum/%'

## Part where we join with the table grouped by repo to calculate the % of contributions by repository

inner join (select t1.repo.name as GhRepo,

cast(created\_at as date) as date2,

SUM(COUNT(t1.repo.name)) OVER (ORDER BYcast(created\_at as date) ROWS BETWEEN 29 PRECEDING AND CURRENT ROW) AS rolling\_sum\_30\_d\_repo

group by GhRepo, date2

) t2 on t1.GhRepo =t2.GhRepo and t1.date1=t2.date2

## Try change the group by condition from created\_at -> date1

group by t1.GhRepo, t1.date1, t1.user, t1.percent\_activities\_user, t1.is\_core\_developer ## added fields to group by

order by t1.GhRepo, t1.date1, t1.user

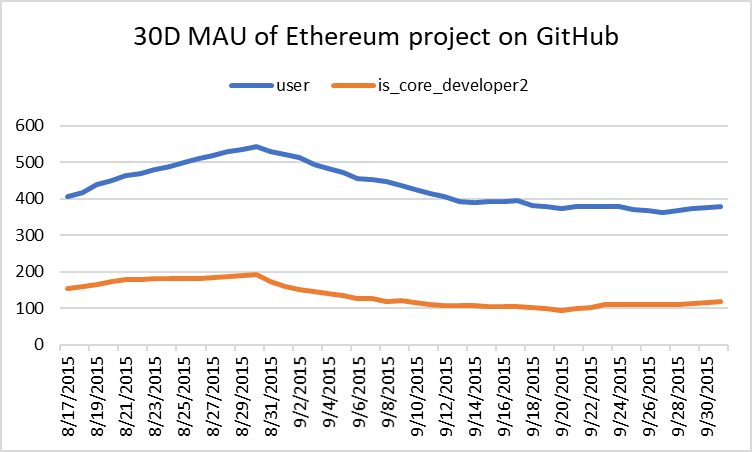
However, the table the query returns is not what I am looking for:

| Row | GhRepo | date1 | user | num\_activities | rolling\_sum\_30\_d |
| --- | --- | --- | --- | --- | --- |
| 1 | ethereum/alethzero | 2015-08-17 | 138296 | 1 | 31 |
| 2 | ethereum/alethzero | 2015-08-17 | 138296 | 1 | 32 |
| 3 | ethereum/alethzero | 2015-08-17 | 138296 | 1 | 32 |
| 4 | ethereum/alethzero | 2015-08-17 | 138296 | 1 | 30 |
| 5 | ethereum/alethzero | 2015-08-17 | 138296 | 1 | 30 |

Graphics for analysis:

<https://mixpanel.com/blog/mau/>





Regression analysis:

**DV: Num\_dist\_issuesopened**

> FEmodel <- plm(Num\_dist\_issuesopened~+Num\_peripheral\_devs

+ +Num\_core\_devs

+ , data = ossdf, model = "within")

> summary(FEmodel)$coefficients

Estimate Std. Error t-value Pr(>|t|)

Num\_peripheral\_devs 0.302521210 0.01011419 29.9105763 7.830256e-119 (<0.05)

Num\_core\_devs 0.005659715 0.03879670 0.1458813 8.840667e-01 (=0.8>0.05)

**DV: Num\_dist\_issuesclosed**

> FEmodel <- plm(Num\_dist\_issuesclosed~+Num\_peripheral\_devs

+ +Num\_core\_devs

+ , data = ossdf, model = "within")

> summary(FEmodel)$coefficients

Estimate Std. Error t-value Pr(>|t|)

Num\_peripheral\_devs 0.32581759 0.01291158 25.234528 7.424812e-95

Num\_core\_devs -0.05484325 0.04952713 -1.107338 2.686158e-01 (=0.26>0.05)

**DV: Num\_forks\_event**

> FEmodel <- plm(Num\_forks\_event~+Num\_peripheral\_devs

+ +Num\_core\_devs

+ , data = ossdf, model = "within")

> summary(FEmodel)$coefficients

Estimate Std. Error t-value Pr(>|t|)

Num\_peripheral\_devs 0.0462129 0.003129408 14.767300 5.177020e-42

Num\_core\_devs 0.0336877 0.012003998 2.806373 5.181832e-03

**DV: Num\_watches\_event**

> FEmodel <- plm(Num\_watch\_event~+Num\_peripheral\_devs

+ +Num\_core\_devs

+ , data = ossdf, model = "within")

> summary(FEmodel)$coefficients

Estimate Std. Error t-value Pr(>|t|)

Num\_peripheral\_devs 0.111790351 0.006123597 18.2556680 5.962869e-59

Num\_core\_devs 0.006455168 0.023489316 0.2748129 7.835597e-01 (=0.7>0.05)