What is an important event? Counting 207 million events recursively

Was 195 million events!

Introduction

> In any practical going-on with numbers, what matters is that they can be *made* to work, and *making them work* is a politics. Yet is a politics that completely evades conventional foundationist analysis (Verran, 2001: 88)

This paper concerns the work needed to get contemporary digital data and particularly, large numbers numbers, to do something other than confirm what we already know. This work is somewhat ethnographic in its ambitions. That is, it hopes to be a form of writing that derives from an encounter with some particular differences or alterity that challenges accepted understandings of what is happening. The ethnographic setting here is not a field site, but principally a stream of timestamped event data from a social media platform. Rather than travelling some place to do ethnography, I'm asking: how can can we treat a body of data ethnographically as a kind of field site? That means finding a place to sit in relation to the data and learning some ways of having a conversation with it, a conversation that will include questions about how many and how often, but will also include the experience of being located in the data. The question here is: how might deliberate and somewhat naïve preoccupation with digital data become not only informative but provocative?

The kind of data I have in mind can be glimpsed at places like this: https://api.github.com/events. What you see here is an event stream for a social media platform called Github that hosts collaborative software development or 'coding.' In this paper, I will not spend much time trying to convince you of the general significance of Github. For the moment, you need only accept that this flow is an event stream typical of contemporary digital platforms, devices and media. Such event streams can be found in online media, but also in many other places both in the sciences and in various infrastructures.

As you can see in any web browser, these events are detailed and heavily formatted. If we look at one event a bit more closely, we see that it is a quite a complex format. This is a 'WatchEvent', meaning that someone who calls themselves 'mmetea' has started to 'watch' a *repository* on Github called 'azondi.' That repository 'azondi' belongs to organisation called 'OpenSensorsIO' that develops software for citizen scientists to do city sensing.


```
1.  {
2.    "id": "2111998059",
3.    "type": "WatchEvent",
4.    "actor": {
5.      "id": 1459103,
6.      "login": "mmemetea",
7.      "gravatar_id": "4532d1e4885f579ca7d9aa8748418817",
8.      "url": "https://api.github.com/users/mmemetea",
9.      "avatar_url": "https://avatars.githubusercontent.com/u/1459103?"
10.   },
11.   "repo": {
12.     "id": 14802742,
13.     "name": "OpenSensorsIO/azondi",
14.     "url": "https://api.github.com/repos/OpenSensorsIO/azondi"
15.   },
16.   "payload": {
17.     "action": "started"
18.   },
19.   "public": true,
20.   "created_at": "2014-05-23T08:40:56Z",
21.   "org": {
22.     "id": 5497318,
23.     "login": "OpenSensorsIO",
24.     "gravatar_id": "1e0218942846ec8ef59f5d679dbca782",
25.     "url": "https://api.github.com/orgs/OpenSensorsIO",
26.     "avatar_url": "https://avatars.githubusercontent.com/u/5497318?"
27.   }
28. },
```


The event of 'Watching' is a very simple act in social media terms, although still quite complicated when you look at it. For instance, in terms of data structures, the couple of dozen lines of data here contains around 25 different variables or values. On lines 21-26, some fields that describe the organisation to which the repository belongs are decribed: the organisation's id (line 22), its login name (23), or the visual image used by organization (line 26). Some of these data fields are id numbers, some are dates and times, some are urls, some are names. Even in the simple act of someone watching something, many things are being counted, named and indexed in time.

Much more complicated event formats can easily be found in event streams today. So for instance, on Github an 'IssueCommentEvent' has a much more complicated referential structure:


```
1.  {
2.    "id": "2111998057",
3.    "type": "IssueCommentEvent",
4.    "actor": {
5.      "id": 179265,
6.      "login": "FroMage",
7.      "gravatar_id": "b8932676ae05ecdd084d3839d1f49da8",
8.      "url": "https://api.github.com/users/FroMage",
9.      "avatar_url": "https://avatars.githubusercontent.com/u/179265?"
10.   },
11.   "repo": {
12.     "id": 1287859,
13.     "name": "ceylon/ceylon-compiler",
14.     "url": "https://api.github.com/repos/ceylon/ceylon-compiler"
15.   },
```

```
    "payload": {
      "action": "created",
      "issue": {
        "url": "https://api.github.com/repos/ceylon/ceylon-compiler/issues/1490",
        "labels_url": "https://api.github.com/repos/ceylon/ceylon-compiler/issues/1490/labels{/name}",
        "comments_url": "https://api.github.com/repos/ceylon/ceylon-compiler/issues/1490/comments",
        "events_url": "https://api.github.com/repos/ceylon/ceylon-compiler/issues/1490/events",
        "html_url": "https://github.com/ceylon/ceylon-compiler/issues/1490",
        "id": 23820110,
        "number": 1490,
        "title": "CLI plugins: support distribution",
        "user": {
          "login": "FroMage",
          "id": 179265,
          "avatar_url": "https://avatars.githubusercontent.com/u/179265?",
          "gravatar_id": "b8932676ae05ecdd084d3839d1f49da8",
          "url": "https://api.github.com/users/FroMage",
          "html_url": "https://github.com/FroMage",
          "followers_url": "https://api.github.com/users/FroMage/followers",
          "following_url": "https://api.github.com/users/FroMage/following{/other_user}",
          "gists_url": "https://api.github.com/users/FroMage/gists{/gist_id}",
          "starred_url": "https://api.github.com/users/FroMage/starred{/owner}{/repo}",
          "subscriptions_url": "https://api.github.com/users/FroMage/subscriptions",
          "organizations_url": "https://api.github.com/users/FroMage/orgs",
          "repos_url": "https://api.github.com/users/FroMage/repos",
          "events_url": "https://api.github.com/users/FroMage/events{/privacy}",
          "received_events_url": "https://api.github.com/users/FroMage/received_events",
          "type": "User",
          "site_admin": false
        },
        "labels": [
          {
            "url": "https://api.github.com/repos/ceylon/ceylon-compiler/labels/FEATURE",
            "name": "FEATURE",
            "color": "02e10c"
          },
          {
            "url": "https://api.github.com/repos/ceylon/ceylon-compiler/labels/IN+PROGRESS",
            "name": "IN PROGRESS",
            "color": "02d7e1"
          },
          {
            "url": "https://api.github.com/repos/ceylon/ceylon-compiler/labels/tools",
            "name": "tools",
            "color": "e102d8"
          }
        ],
        "state": "open",
        "assignee": null,
        "milestone": {
          "url": "https://api.github.com/repos/ceylon/ceylon-compiler/milestones/7",
          "labels_url": "https://api.github.com/repos/ceylon/ceylon-compiler/milestones/7/labels",
          "id": 217273,
          "number": 7,
          "title": "1.1",
          "description": "",
          "creator": {
            "login": "tombentley",
            "id": 879487,
            "avatar_url": "https://avatars.githubusercontent.com/u/879487?",
            "gravatar_id": "19f6b3877a3b4d15467481f7eeb0f635",
            "url": "https://api.github.com/users/tombentley",
            "html_url": "https://github.com/tombentley",
            "followers_url": "https://api.github.com/users/tombentley/followers",
            "following_url": "https://api.github.com/users/tombentley/following{/other_user}",
            "gists_url": "https://api.github.com/users/tombentley/gists{/gist_id}",
            "starred_url": "https://api.github.com/users/tombentley/starred{/owner}{/repo}",
            "subscriptions_url": "https://api.github.com/users/tombentley/subscriptions",
            "organizations_url": "https://api.github.com/users/tombentley/orgs",
            "repos_url": "https://api.github.com/users/tombentley/repos",
            "events_url": "https://api.github.com/users/tombentley/events{/privacy}",
            "received_events_url": "https://api.github.com/users/tombentley/received_events",
            "type": "User",
            "site_admin": false
          },
          "open_issues": 21,
          "closed_issues": 175,
          "state": "open",
          "created_at": "2012-11-22T11:43:26Z",
          "updated_at": "2014-05-22T16:34:51Z",
          "due_on": null
        },
        "comments": 4,
        "created_at": "2013-12-05T21:43:00Z",
        "updated_at": "2014-05-23T08:40:56Z",
        "closed_at": null,
        "body": "We need to find a way to support distribution/installation of script plugins. One possible strategy would be to package whatever scripts are located in the `scripts` folder into `module/name/module.name-version.scripts.zip` and have `ceylon plugin --add module.name/version` install it in `~/.ceylon/scripts` and make that folder searched recursively by the `ceylon` tool when looking for plugins like it does for the `$PATH`."
      },
      "comment": {
        "url": "https://api.github.com/repos/ceylon/ceylon-compiler/issues/comments/43984368",
        "html_url": "https://github.com/ceylon/ceylon-compiler/issues/1490#issuecomment-43984368",
        "issue_url": "https://api.github.com/repos/ceylon/ceylon-compiler/issues/1490",
        "id": 43984368,
        "user": {
          "login": "FroMage",
          "id": 179265,
          "avatar_url": "https://avatars.githubusercontent.com/u/179265?",
```

```
        "gravatar_id": "b8932676ae05ecdd084d3839d1f49da8",
        "url": "https://api.github.com/users/FroMage",
        "html_url": "https://github.com/FroMage",
        "followers_url": "https://api.github.com/users/FroMage/followers",
        "following_url": "https://api.github.com/users/FroMage/following{/other_user}",
        "gists_url": "https://api.github.com/users/FroMage/gists{/gist_id}",
        "starred_url": "https://api.github.com/users/FroMage/starred{/owner}{/repo}",
        "subscriptions_url": "https://api.github.com/users/FroMage/subscriptions",
        "organizations_url": "https://api.github.com/users/FroMage/orgs",
        "repos_url": "https://api.github.com/users/FroMage/repos",
        "events_url": "https://api.github.com/users/FroMage/events{/privacy}",
        "received_events_url": "https://api.github.com/users/FroMage/received_events",
        "type": "User",
        "site_admin": false
      },
      "created_at": "2014-05-23T08:40:56Z",
      "updated_at": "2014-05-23T08:40:56Z",
      "body": "We could put the scripts in the `.car`, but then where would the JS compiler put it?"
    }
  },
  "public": true,
  "created_at": "2014-05-23T08:40:56Z",
  "org": {
    "id": 579261,
    "login": "ceylon",
    "gravatar_id": "a38479e9dc888f68fb6911d4ce05d7cc",
    "url": "https://api.github.com/orgs/ceylon",
    "avatar_url": "https://avatars.githubusercontent.com/u/579261?"
  }
}
```


Instead of the event having 2 dozen lines, it now has well over a hundred, and the hierarchical format is much more complicated, with lots of nested fields. These events start to become harder to read because there are too many components or actors indexed here. We see many names, quite a few addresses and a number of date-times.

Specifying site-specific events

If even a single event may have several hundred variables in the timestamped record, how do we traverse hundreds of millions of such events? What kind of work can we do on these large numbers to make them work differently?

In so many settings, STS research has encountered vast and complex forms of expression. In scientific literatures, in technologies, in media platforms, and in forms of organizational and institutional life associated with health, environment, medicine, energy, finance or government, it has confronted problems of seemingly overwhelming scale and diversity. In some ways, formative work in STS has seen concerted attempts to re-scale science and technology, for instance, by locating and localising it. In social construction of technology (Pinch, Bijker, Collins, Mackenzie), in actor-network theory (Latour, Akrich, Callon, Law), in networks of power (Hughes), material-semiotics (Haraway), or ontological politics or choreographies (Mol, Law, Charis-Thompson), scale has often been both a substantive concern and a methodological challenge. Various proposals to do mid-level research have been explored (Anne; etc). In STS, even as scale-related talk (the global, the universal, etc) is problematised, its own has largely

remained scale-invariant. That is, the specific field-site anchors research, occasionally supplemented by the historical archive. We tell selected stories from the field that convey our own participation in that setting or site. As in anthropology more broadly, site-specific narratives offer a kind of literary technology of witnessing not the replicability of the empirical result, but a singular difference, an alterity that somehow transforms things closer to home.

The question I explore here then is how to address large numbers of events in heavily formatted data while retaining the openness to site-specificity that allows ethnography to encounter alterity. Is there work imbued with ethnographic sensibility that can be done on complex and vast data formats? I'm not referring to doing ethnographies of IT companies or users of social media platforms, but something more rudimentary albeit technically complicated to do with the data streams and the flow of inscriptions. What kind of work could be done in the data stream with an ethnographic sensibility open to **displacements** in the same way that ethnographers are displaced in various ways by their field sites? And how would that work on numbers and words do something different to what is being done with this data more generally, which I would suggest, is largely reductive?

My suggestion here is that we need to try to do two things at once in working with such data if we want to open up its ethnographic value:

1. **do device-specific research** that highlights the ways in which the data is formatted by the social life of the devices and platforms that give rise to it;

2. look for the forms of practice, difference, change or becoming that cannot be fully formatted by the event-structures of the platform. That is, look for what **overflows the platform and its formatting of events**.

The interplay of the device-specific and the overflows, I suggest, is an eventful one. The interplay is potentially generative, and it might be interesting empirically and theoretically for STS as it comes to grips with data. We can't always decide whether what we are seeing is specific to the device or platform or comes from elsewhere – from pre-existing social practices, from other layers of organisation. (This mixing generates many of the recursive effects that anthropologists such as Anne-Lise Riles, Marilyn Strathern and Chris Kelty have described using notions such as recursion and

mirroring.) And they also mean that data streams associated with these platforms exist in more than one mode, even if certain modes of publishing and using them tend to be more heavily power-laden. Because of STS's long-standing interest in keeping both the device-specific and the overflow in view at the same time, it might do some generative work.

The device specific: the case of Github

As foreshadowed already, my case study or 'showcase' example is Github.com, a 'social coding' platform widely used by software developers to make software. Github is a social media platform for software development. It claims to be the largest code repository 'on the planet.' Somewhat recursively, it is a general platform for collectively making device-specific things, that is, code. Given that software is generally acknowledged as a pervasive component of our lives, could we say something about the world by viewing it from the perspective of Github? We have already seen from the sample event data above that everything on Github concerns *repositories*. These repositories or **repos** contain code for many different devices, platforms and settings. The hundreds of millions of events recorded in the Github show work done on and between these millions of repositories.

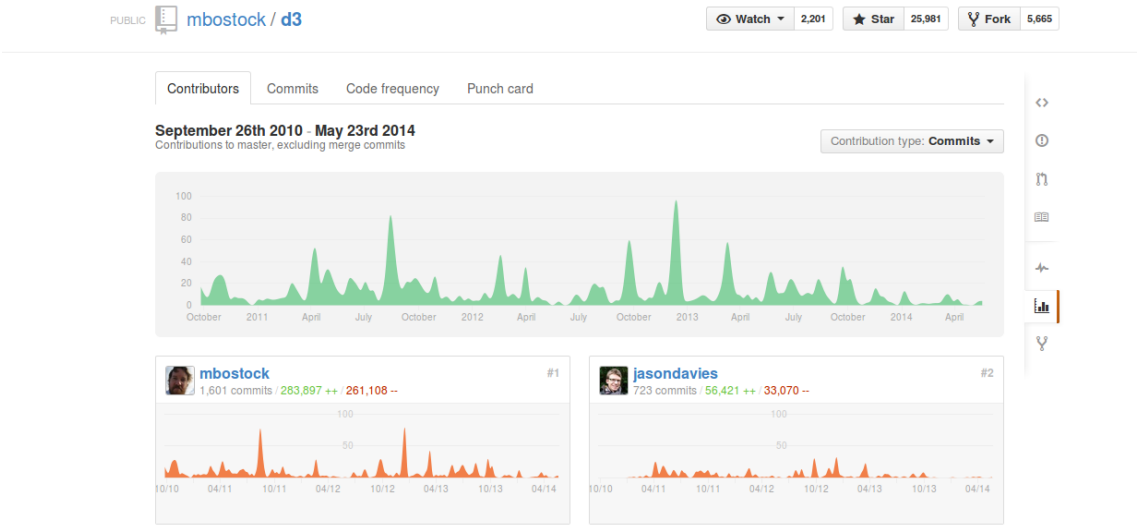
There is a much to say about Github from various angles, including its funding, its importance to software developers, its media visibility, the way it works as a social media platform, what it even means to collaboratively develop software using code repositories and how software development platforms figure in a broader political economy. However, given my focus on data streams, and attempts to inhabit an indeterminate zone between device-specific and social practice in the world, I will leave most of that aside and focus instead on how we inhabit data streams in a more ethnographic mode.

For instance, Github itself makes many data-based claims about the platform. Like many social media platforms, but also many sciences, they count events at various levels and report totals. **They claim, for instance, that Github has 13.2 million repositories used by 6 million people** (https://github.com/press; June 2014). This number of software projects and people suggests both enormous activity around

software development, and the centrality of Github or 'hubbiness' to that activity, and kind of 'many-to-one' relation.

At the same time, Github seeks to avoid the 'information loss' intrinsic to the large numbers by selecting a very small number of repository as 'showcases.' Like many other social media platforms, it presents what is happening on the platform by showing key examples or actors. In the 'showcases,' exemplary repositories become visible as representatives of the spectrum of different kinds of things being done on Github. (These 30 or so categories, each containing a small number of repositories may be representative of everything in the event stream, but it is more likely they are chosen by Github staff for various reasons.)

For any individual repository or indeed individual actor, a great deal of data and data visualization is available.



https://github.com/mbostock/d3/graphs/contributors

For any given repository, detailed graphs of who contributed what can be found. As the Figure above shows, the number of people adding or editing code, the number of people watching, the number of people copying ('forks'), etc., are all visible. This again is typical of contemporary media platform data streams. Individuals and single entities become hypervisible and visualized.

In terms of our ethnographic interest in the data stream, how can we move between the shallow overall count data (12 million repos; 6 million people; 200 million events), the exemplary showcases (a few hundred repositories), and the deeply fine-grained

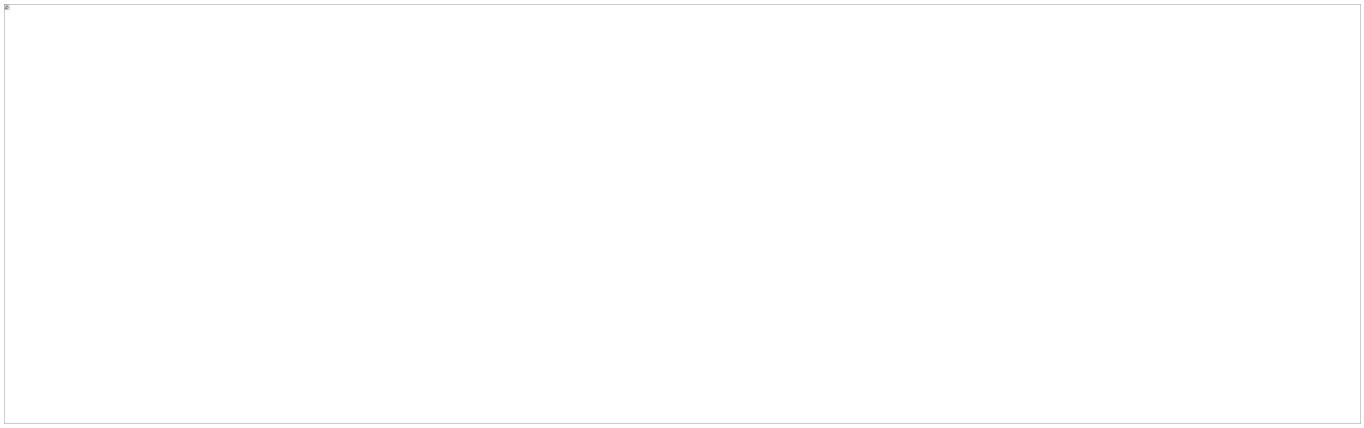
individual repository or user data (1 repository, a couple of people). Does the formatted datastream afford any other kind of movement other than the shallow global counts or the individualised traces of actions?

Publishing data as an archive

The bare fact of being able to access datastreams about Github from a web browser already says something important. This data is **published**. Its publication is not an accident or incidental factor. The availability of the formatted event streams is part of the platform that needs to be analysed. People do various things with these datastreams, and the fact that the datastreams are used in different ways should figure in any sense that we might make of them.

The very fact of the existence of the datastream itself should not be taken for granted here. The existence of a more or less public datastream is a device-specific feature common to many social media platforms. People publish data so that other people will do things with it, and thereby augment or mirror the platform. Publishing of the data, heavily formatted as we have seen, is just as much a part of the Github platform and its tactics of assembling many devices in a network as is making software source code open, or for that matter, as publishing a paper in STS. As in so many settings today, the data is not something created by ethnographic observation, but formatted for publication. How does the fact of the operational availability of the data figure in our encounter with it? We should look at how people see and work on the data. This is more or less equivalent to listening to what informants or interlocutors in a given setting say about that place. What people do with public datastreams is a kind of practice that can be observed.

For one thing, the datastream is something whose very existence and circulation itself has value. One thing people do in response is **archive it** in various ways, sometimes just locally, but other times in quite public ways. For almost three years, Githubarchive.org has archived the public Github data stream. The rationale for this archive is to support further analysis:



> Open-source developers all over the world are working on millions of projects: writing code & documentation, fixing & submitting bugs, and so forth. GitHub Archive is a project to **record** the public GitHub timeline, **archive it**, and **make it easily accessible** for further analysis (Grigorik, 2012)

writes Ilya Grigorik, a 'Developer Advocate @ Google, working on everything web performance related: protocols, standards, browser performance' according to his Twitter feed.

Given Grigorik's role as 'developer advocate,' it's no surprising that all the GithubArchive data can be also found on Google's BigQuery cloud computing service as well as at githubarchive.org:

> GitHub Archive dataset is also available via Google BigQuery. The JSON data is normalized and is updated every hour, allowing you to run arbitrary queries and analysis over the entire dataset in seconds. To get started, login into the BigQuery console (bigquery.cloud.google.com), and add the project (name: "**githubarchive**"), or take a look at the 03/11..05/11 snapshot of the data under "**publicdata:samples**" (http://www.githubarchive.org/)

Much more detail on how to access and use this data can be found on Grigorik's in one of Grigorik's own Github repositories (https://github.com/igrigorik/githubarchive.org/tree/master/bigquery). For instance, the approximately 200 columns in the dataset are described in the 'schema.js' document there (https://github.com/igrigorik/githubarchive.org/blob/master/bigquery/schema.js). But we can see that the transformation of the live datastream into an archive begins to

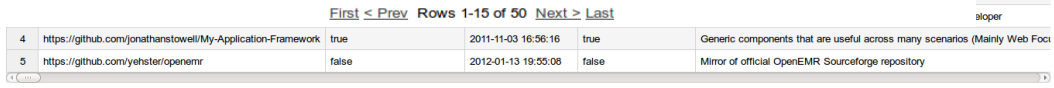
reformat it in platform specific way. Events becomes timestamped rows in a vast table of around 200 million events. So, to take an almost random example, the query

'SELECT created_at, type, actor, repository_name, url FROM [githubarchive:github.timeline] where LIMIT 50'

yields:

Query Results 11:19am, 31 Mar 2014

Row	created_at	type	actor	repository_name	url
1	2014-03-15 09:04:43	PushEvent	vvakame	DefinitelyTyped	https://github.com/borisyankov/DefinitelyTyped/compare/3d3832de3e...93a2313f62
2	2014-03-15 09:04:42	PushEvent	gizmomogwai	plist	https://github.com/gizmomogwai/plist/compare/12eb82d283...83ad8b5f26
3	2014-03-15 09:04:48	IssueCommentEvent	ben-lin	node.inflection	https://github.com/dreamerslab/node.inflection/issues/16#issuecomment-37721104
4	2014-03-15 09:04:48	WatchEvent	cbmd	mongofill	https://github.com/koubas/mongofill
5	2014-03-15 09:04:48	WatchEvent	svett	molokai	https://github.com/tomasr/molokai
6	2014-03-15 09:04:47	GollumEvent	swordray	wiki	https://github.com/ruby-china/wiki/wiki/RubyGems
7	2014-03-15 09:04:46	PushEvent	elfet	purephp	https://github.com/elfet/purephp/compare/dc24b70f00...c94c524e76
8	2014-03-15 09:04:46	PullRequestEvent	nikkypx	rets_data	https://github.com/arcticleo/rets_data/pull/2
9	2014-03-15 09:04:46	WatchEvent	mjaneczek	google-authenticator	https://github.com/jaredonline/google-authenticator
10	2014-03-15 09:04:53	PushEvent	micmath	Rye	https://github.com/micmath/Rye/compare/69c02b1aea...7885ea9e36
11	2014-03-15 09:04:53	PushEvent	fitret	daigon	https://github.com/fitret/daigon/compare/8897fa9404...77e5f48276
12	2014-03-15 09:04:53	WatchEvent	joeyates	spork	https://github.com/sporkrb/spork
13	2014-03-15 09:04:53	CreateEvent	grcnva	mcs-chatboard	https://github.com/grcnva/mcs-chatboard
14	2014-03-15 09:04:51	PushEvent	albatrossen	bungeebouncer	https://github.com/albatrossen/bungeebouncer/compare/57a3771590...7794ffb531
15	2014-03-15 09:04:51	PushEvent	catalinstanciu	ChessEngine-Xboard-cpp	https://github.com/catalinstanciu/ChessEngine-Xboard-cpp/compare/9bedce1744...8b98d2ff45



This is not a meaningfuly, but we can see a transformation in the datastream via this double archiving move. The live data we saw in a web browser was first archived (githubarchive.org) and now appears back in a live web browser in the table format familiar to much speadsheet data analysis. This reformatting of the data, via GithubArchive and Google BigQuery is highly device- specific. While the table of all Github events on BigQuery contains several hundred million events in a form that is more legible than the original event format. Each event is only one line, not a hundred lines.

The data has been reformatted in a familiar and perhaps more 'public' way, but only within the context of a cloud-based data analytics device, Google's BigQuery. So the publication of the datastream does not end the formatting of the data or annul its device-specificity. It flows into new device-specific formats, with their own resistances and affordances.

Data challenges and their imaginaries

Datastreams are not just published and archived. We can begin to see patterns of activity that lie between the total aggregate counts (12 million repos) but are not so localised to one or a few repositories that we explore in detail. Since its inception two years ago, the archived Github data has attracted many different modelling and visualization efforts. This is the second layer of practice that we might attend to as we develop an ethnographic sensibility to datastreams. The fact Github itself organises 'data challenges' (https://github.com/blog/1544-data-challenge-ii-results) suggests that people doing things with this data is part and parcel of the public life and publicity of the platform or device. These data challenges are what AnneLise Riles calls 'momentary apprehensions of depth' showed that drawing networks has long been part of networks (Riles, 2001: 184). Whatever the reason for these data challenges (they are increasingly common in many places today in the form of hackathons, data competitions, etc), they bring to light practical imaginings of what is in data.

In response to these challenges and availability of the public Github datastream, people do things like:

1. the **OpenSource Report Card** (http://osrc.dfm.io/) by Dan Foreman-Mackay, is a prize-winning use of the timeline data. It ingests all the data from the Githubarchive, counting what developers do, when they do it, and using what programming languages. With this data stored, it then builds a predictive model that allows it to both characterise a given Github user, and to predict who that Github user might have affinities with. The admonition from Foreman-Mackay – 'Dear recruiters: While you read this, make sure that you remember that GitHub is not your C.V. and that these stats only provide a biased and one-sided view. This is just a toy. Don't take it too seriously! ' – suggests that this is a playful application of the data, but one that is nevertheless quite typical of what is being done with data streams in social media, mobile communications, retail advertising and marketing, etc. OpenSource Report Card treats the data as a way of seeing similiarities and differences in individual software developers in terms of counting events by type and comparing how often and when an individual does something on Github. Here the massive datastream is brought to bear on finding similiarities between people.

2. People often map datastreams by geographic location. An extraordinary proliferation of maps has occurred as a result. The **mapping of Github contributions by location** performed by David Fischer (http://davidfischer.github.io/gdc2/#languages/C__) is typical in that it too counts events, but this time puts the emphasis on places of work.

3. As is common today in sentiment analysis on social, people look for feelings in data. In 2012, software developers feelings were mined in terms of emotional words present in comments found in the Github events (http://geeksta.net/geeklog/exploring-expressions-emotions-github-commit-messages/), and the presence of words in these message can be cross-linked with programming languages in order to identify what programming language elicit most emotion. The emphasis here is on how software developers **feel** in relation to particular kinds of work.

4. Finally, people make live dashboards for Github. Octoboard (http://octoboard.com/) is a typical dashboard. **Dashboards** are becoming very common as a way to engage with the liveness of data. Data analysis is no longer something done in leisurely rhythms of analysis, but is increasingly framed as a realtime in the form of 'stream analytics.' That is, not also does the data stream, but the analysis is meant to stream as well in order to be timely, lively and responsive to change. Octoboard presents a summary of daily activity in major categories on Github – how many new repositories, how many issues, how repositories have been 'open sourced' **today**. Also it offers realtime analytics on emotions.

5. Finally, and looking slightly more widely, the Github data stream has quickly become a favourite training tool for data mining books and for academic researchers in fields such as software engineering in doing social network analysis (Thung et al., 2013). In his book *Mining the Social Web: Data Mining Facebook, Twitter, LinkedIn, Google+, GitHub, and More,* Matthew Russell makes use of the Github timeline to demonstrate ways of uses social network analysis to highlight the important nodes and links between repositories and users (Russell, 2013). Again, the propensity to apply network analysis approaches is widespread and endemic to the data itself, given the way that the event format is already implicitly framed by a network or 'social media' understanding.

So, the actors in concert with the archives of the datastreams are already doing so much in the data streams. They identify individual behaviours and their similiarities, they

analysis geographies, emotions, social networks, and they offer live summaries and so on. They often invent ways of working with the data using particular devices and platforms. The way in which the OpenSource Report Card models counts what developers do on Github and constructs a simple but vast encompassing database and model for millions of Github developers is quite fascinating. The visual forms of some of the projects are interesting too for the way in which they sometimes provide very simple summaries – bar chart, a some text, a network visualization – of highly entwined processes.

But, isn't everything we have seen, beginning with the simple event histograms produced by GithubArchive.org or Github itself through to the textual descriptions generated by OpenSource Report Card or the maps of contributions to specific code repositories quite familiar to us? Where is the alterity, the difference, the displacements? Do not many of these data analytic devices effectively work to cover over differences in an all encompassing datastream even as they seek to extract meaningful features from it? They perhaps reduce all differences to a known set of differences imagined at the outset: the relatively predictable geographies, work patterns, and popularity of techniques and languages, or live dashboards for the Github platform? Each of their graphics is more likely to validate Github's own count of 12 million repositories as the biggest code repository on the planet?

The open set of irreducible names

As an alternative to the global reduction, and in name of irreductions in data, I want to suggest that an ethnographic sensibility to data might inhabit the space between the specificities of the platform or device and the generalities of emotions, social network and global geographies by beginning to identify more messy and more difficult to count parts of the datastreams. There any many possibilities here, but I will focus on one that relates to the **names of repositories.**

None of the ways of looking at the data stream I have shown you take any account of what people are actually working on or identifying with as they do things on Github. (The only example of a visualization or artefact that goes anywhere near the world in its lived diversity or its multiplicity is Github's own 'showcases', but I suspect this high

profile collection of repositories is not derived from the data streams cut carefully curated by Github marketing or PR teams.) It seems as if these degree of specificity about what sofware is or does is somehow to see.

What if we paid attention to the names, all 12 million of them, on Github? What could we learn from names, and how things are named on Github? There is a long-standing anthropological and a more recent social theory literature on names and naming. Here I'm drawing on just two ideas, one from Judith Butler and another from Bruno Latour and Vincent Lepinay, but set within a broader framing of STS work on ethnographic work. Butler usefully sets out the general conditions of naming:

> **Consider for a moment the more general conditions of naming. First, a name is offered, given, imposed by someone or by some set of someones, and it is attributed to someone else. It requires an intersubjective context, but also a mode of address, for the name emerges as the addressing of a coinage to another, and in that address, a rendering of that coinage proper. (Butler, 1997: 29)**

Clearly, she is referring here to people's names, but things have names too. In Github, there are a vast number of named things – libraries, devices, packages, systems, formats, standards. So we are dealing we a highly intersubjective context, in which many of the subjects are linked with objects by names. Both people and things often have proper names or nouns that have been made into proper names: – Bootstrap, Cassandra, Mahout, Apache, Weibo, Habo, RenRen. Obviously people have names as 'actors' and 'owners' on Github, and, although I don't discuss it here, we can understand much of what happens in writing code in various languages as a practice of developing names and maintaining names in circulation over time. (The concept of the namespace is quite important within coding itself because controlling the conditions under which things are linked to names stabilises repetition.)

Butler asks us to consider the general conditions of naming in terms of acts of offering/giving/imposing, combined with a 'mode of address' to another and something like 'coinage', that is a circulating form of value. These three moments of attributing, addressing, and circulating mean that names are both processual and yet tend to freeze or limit processes in certain ways. As Butler writes, 'a name is not the same as an undifferentiated temporal process or the complex convergence of relations that go under

the rubric of "a situation’" (35). This bundling potential of names means that they act powerfully. They are at the core of most performative processes.

What happens to names on Github? If we approach the 13.2 million repositories in terms of naming practices, what do we find? Obviously, this is a large number of names to deal with, something like the population of a country. But like the population of a country, the flow of names in Github runs in lines or lineages. These lineages develop differently depending on who or what is naming who, and who is *citing* or *imitating* what. Tracking the attribution, mode of address and circulation of repository names then, I am suggesting, might tell us something about how Github comes to be populated by millions repositories. As Gisli Palsson has recently written on personal names in the context of kinship and genetics, 'it is important to explore the variety in current naming traditions in terms of what they do, the impact of the speech acts involved, integrating analyses of such technologies into studies of the constitution of subjects, the politics of belonging, exclusion, and control' (Palsson, 2014: 628).Obviously, there is no space here to map the namescape of Github in any detail, but a few strands of the flow of names might be useful.

A Names as 'proper coinage' or points of identification

The names used for Github repositories mostly display varying degrees of what we might call propriety, or properness. That is, they render the repository proper in an intersubjective context in differently weighted ways.

Names themselves invoke or cite named entities and processes of various coinages or value. Sometimes they invoke well known platforms and devices such as Google, Facebook or Twitter, iPhone or instagram. Sometimes they invoke techniques and styles of code development ('perl', 'php', 'api', 'sdk'). They also index labour processes such as 'learning', 'mining', 'monitoring' etc. Note that the full name of the repository usually includes both a personal or an organisational name ('Hiroki-Takayama' or 'davcro') along with the code entities and processes. This means that a full repository name already bundles people, things and events in ways that resonate that with STS interests in material-semiotic processes.

If it is possible to sustain work on names in the heavily formatted datastream, then we might begin to trace out some of the differently weighted degrees of propriety and value that flow through the aggregate numbers. These differing degrees and weighted values are not explicitly formatted in the data. Just the opposite. They are the most unformatted, non-standardised part of the data. In contrast to the timestamps, URLs, and id numbers we saw in the sample event data earlier, the names are highly mobile yet also mutable forms.

I would like to suggest, without being to evidence it really fully that names in the datastream arelike the *memento mori* anamorphic skull in Hans Holbein's *The Ambassadors* that Latour and Winthereik and Verran discuss as a usefully disfiguring object in accounts of science and technology (Winthereik and Verran, 2012: 42)

Of course, as with any social media platform, we find an enormous amount of distracting complication in these names. A repository called:

'http://github.com/dpackcartonmachine/3layer5layer7layer-corrugated-box-machine--348695'

is unlikely to shed much light on current trends in software development. (For some reason, there are dozens of these 'box machine' repositroies. They were likely part of some Github testing, but might arise from some other source) For us, the existence of all these cardbox box folding machine repositories is not so important, but rather it suggests that unrecognisable names, the names that are difficult to understand might begin to work against the heavy formatting of the data streams that makes them so countable.

B Replicated names and flows of imitation

When we start exploring the name space, a second important feature of the repository names appears: the huge weight of imitation or citationality occurring. Uniques names are exceptional. Of the 13.2 million of so repository names, there are only 5.2 million distinct names. Roughly **4.4 million** repositories are direct copies or citations of others. They are the products of '**forks**', or copies. These patterns of forking or copying suggest large scale processes of imitation running through the activities on Github. Added to that the very large number of names that are only slight variations of each other, and the 13 million repositories on Github begins to change scale a bit.

The copying of names is an important part of naming practices in general since names are hardly ever unique. But in this case, copying a name also means copying a whole body of code, a repository, and these repositories embody very different commitments of time, different participants, organisations and indeed lives. Thus, the massive citation or repetition of names attests to vast processes of copying that allow the Github platform to grow and expand, and but feed into the growth of software cultures more generally in very different ways. For our purposes, it means that any counting of what happens on social platforms has to also deal with the fact that much of what happens is imitation, yet what is imitation is very heterogeneous. Tracking **what is imitated** might say more about what is happening both on Github and with software more generally than counting the number of repositories or users.

What does imitation do? Imitation is a form of identifying or making the same. As Moira Gatens writes 'incorporation into collectivities … involves affective imitation' (Gatens and Lloyd, 1999). When repository names are copied, people not only copy code, but participate in processes of affective imitation. These processes of imitation have been particularly difficult to study because they are neither contained with representations or inscriptions. How can we track the flows of imitation in the massive enviroments we find in contemporary technological cultures? The simple act of copying a name by forking a repository on Github is only the tip or point of an incorporation that attract investments of time and effort, sometimes from individuals, sometimes from informal collectives and sometimes from institutional collectives. But whereever the actors and collective are active, the replication of names attests to power-laden complexities. Convergences or confluences of names arise because people and other

actors are picking up names and using them. Tracking flows of names might allow us to gauge, as Andrew Barry and Nigel Thrift put it:

> the importance of suggestion and imitation in social life, and the limits of forms of analysis which confine themselves to the study of representation or inscription. (Barry and Thrift, 2007: 511)

Although repository names are encounters as attributes in a datastream, the ongoing flow of events that create named repository is a form of 'passionate imitation.' The flow of names in the growth of repositories allows us to begin to conduct 'analysis of the very smallest variations and transformations in style, pronunciation, habit and technology' (Barry and Thrift, 2007) that sometimes, and usually somewhat unpredictability, become new framings or events.

C Analysis of variations

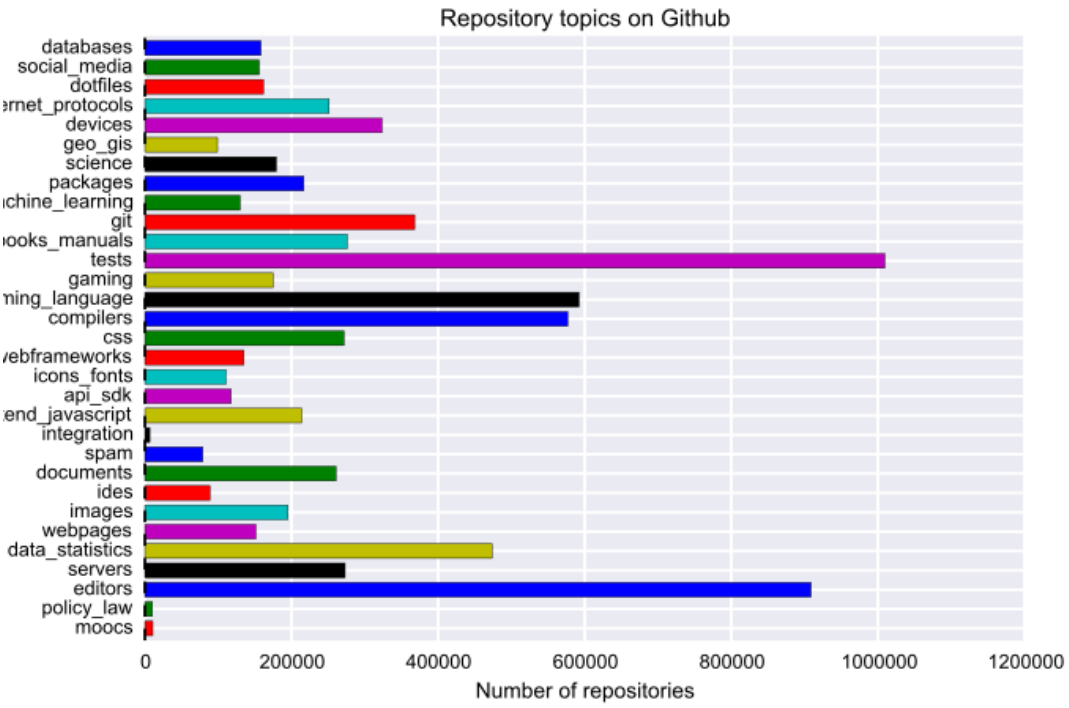
A second outstanding feature of the repository names is the way they combine different particular domains. They are often difficult to classify because they don't easily belong to any one set of concerns. What happens if we start to count both the points of identification conveyed in names and new intersections between these point of identification? As well as following imitations and suggestion, the flow of names in the development of repositories allows us to begin to conduct analysis of the patterns of interference and intersection that generate differences and invention in collective life.

The entities mentioned in repository names sketch economies and collective life associated with software. The names contain points of identification with science, media, finance, business, and government. They traverse a very large variety of different generic and specific techniques associated with software. We could look at many broad domains shown in the figure below. Repositories in Github are neither tagged or labelled as such. Names, as I have said, distort the platform-specific formatting of the datastream. The namespace of the repositories constitute not only an intersubjective context in which people affectively identify with collectives, but a transfinite milieu, more like a thunderstorm full of flashes and movement. Something like a storm of name combinations comes through in the datastream from Github or from the Github archives. The raw count of names doesn't tell us much about conditions on the ground, or it tells us about the same amount as the average rainfall measurements. We don't

know how much these names matter. The question is rather where does the rain fall and where does it go. Indeed, rather than the count of names being something that can be easily measured, the counts fluidly change as you move through the datasets.

We can count names that relate to domains by making or using lists of the most common words found in repository names, and then running queries against the Github archive or API datastream to see how those words appear within repository names. And scaled up through the use of cloud-based 'big data' infrastructures, these queries begin to expand and pluralise the aggregate base numbers of events and entities on the Github platform into a multiplicity with duration and change. Unlike the counts of geographical locations, or programming languages or user activity we saw earlier, mapping domains of activity on the platform in a **namespace** involves unstable trajectories.

For instance, after running queries for around 1000 key terms against the several hundred million events in the Github archive, dataset, we can start to account for large portions of the 12 million repos in a more plural way as made up of a mosaic of intersecting domains. These queries, I should note, are not trivial or unproblematic to construct. They are themselves sediments of much other work with specific devices, platforms, infrastructures and software such as databases and data processing software.



This plot accounts for roughly 5.7 million repositories of the 13 million or so. The counts of repository names that fall under the several dozen domains are still highly reductive, but they begin to suggest something of the distribution of investments in different domains of coding practice. Rather than pointing towards studies of key cases, they begin to suggest the existence of deeper distributions of work and interest around software over the last few years. The 36 domains of software development are neither discrete nor exhaustive. They simply index some potential site of identification in the complex value-laden environments of software as traceable in Github repository names.

D Variations in analysis

For instance, take the list of all the social media platforms. Such lists are easily found on Wikipedia.org. Searches for the 300 or so media platforms run against the Github list of repository names produces around 150,000 results. Most of the query terms are not matching, but enough of them do to begin to see how different domains or fields take shape in Github.

```
In [93]: socialmedia_df.head(30)
```

Out[93]:

	name	full_name	fork
0	PlurkCSS	howar31/PlurkCSS	1
1	ZXing.Net.Mobile	JerryLee7809/ZXing.Net.Mobile	1
2	twitteroauth	mathijsvdhurk/twitteroauth	1
3	Recipes-for-Mining-Twitter	winnerdy/Recipes-for-Mining-Twitter	1
4	DMActivityInstagram	PrincessThan/DMActivityInstagram	1
5	wikipedia_api	gpherguson/wikipedia_api	0
6	facebook	harjeetsingh111/facebook	0
7	symphony-learning-download	teradotact/symphony-learning-download	0
8	facebook-android-sdk	slott/facebook-android-sdk	1
9	facebook-android-sdk	derfshaya/facebook-android-sdk	1
10	SwanFlickrUploader	entercritical/SwanFlickrUploader	0
11	iOSTraining	tokyoaret/iOSTraining	1
12	angular-google-maps	adato/angular-google-maps	1
13	facebook-php-sdk	cabaltho/facebook-php-sdk	1

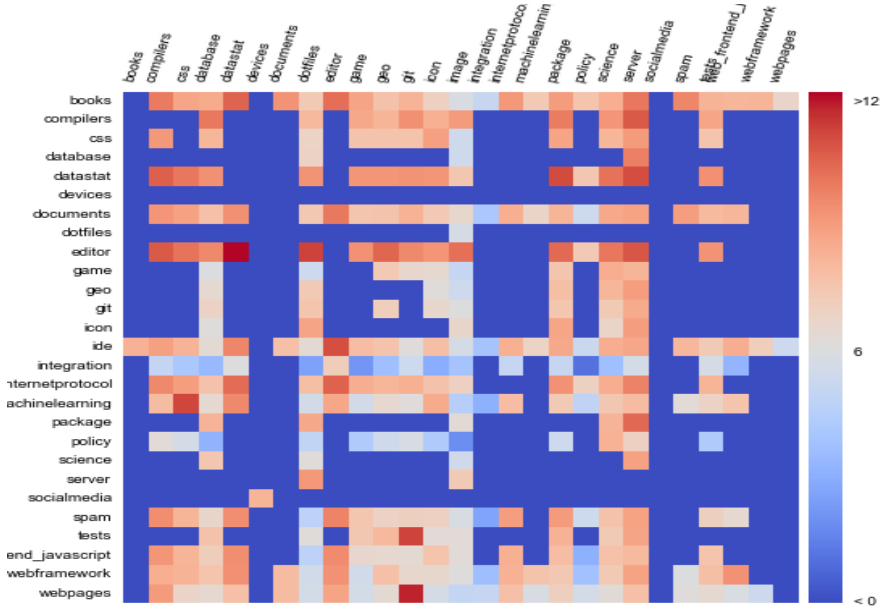
In this small sample of the results, we see the some features of the platform name space starting to appear. We would expect, even in 10 rows of names, to see commonly used social media platforms – Facebook, Instragram, Twitter, Flickr. But they also mix with

less familiar entities such as 'angular', 'php' or 'oauth.' Similar sets of queries for other domains – servers, graphics, code editors – produce other sets of names.

As these entities come together and recombine in different repositories, they also show a heterogeneous mixture of things, some of which seem to have nothing to do with social media. This means that we cannot treat the domains as properly defined by these names, but as constantly shifting in relation to other domains. Indeed the point of these queries on key terms is perhaps less to quantify the amounts of coding done in different domains, but to see how these mixtures and bundles define 'vague wholes' that overflow the device specific formatting of events.

We might expect a lot more growth in code around social media platforms in comparison to word processing software, much more identification with mobile device hardware than with server administration, or in the investment in Big Data, proliferating work on databases and cloud infrastructures. They are very numerous in the Github namespace, and come together in many combinations, including hotspot combinations (see Figure of intersections below). If we see the attribution of names to repositories in Butler's terms, then these names can be seen as tracing out some of the gradients of power and value in social life. Importantly, none of the data analytic approaches described above have any way of identifying these emerging points of identification.

But on the other hand, and I think this is an ethnographic dimension of this kind of data work, our capacity to index these changes, to find different ways to get the numbers to



stack up is an ongoing process, not only that resolves into a definitive count. It changes as we develop different sensibilities for names, and test those sensibilities agains the formatted data streams. The interplay between coining a name and recognising its intersubjective weight or power is an evolving pattern, not a fixed ranking or measurement.

If several thousand queries still only account for 50% of the Github repositories, what is happening in the other millions of repositories? Attempts to find the names of the other repositories are instructive in that they begin to show a kind of slipperiness and plurality in names that makes them difficult to count and categorise. Many repositories on Github come into existence as people try out different things with the platform itself. These repositories in the 'test' domain in the Figure shown above. But switching between querying the Github dataset and looking at Github repositories individually, the domain of testing starts to change shape. From several hundred thousands repositories, it grows to over a million repositories, as I add new terms to the queries that generate the count: 'sample', 'demo', etc, or for instance, finding out the most copied repository on Github is 'Spoon-Knife' and it is a repository for people to test forking of repositories.

We face two different forms of becoming in relation to this data. On the one hand, there is a becoming or change visible in the names as particular domains grow and diminish in size or importance over time. But other concerns the device-specific facets of our own research. Namespace exploration on this scale is not trivial or unproblematic to construct. It entails much work with specific devices, platforms, other infrastructures and software. An important problem in developing an ethnographic sensibility for fields of data is learning to how to interact with data, or how to get data at scale into writing. Some of the literary technologies developed in contemporary science for data analysis such as executable notebooks are very helpful for this. Some software devices developed on Github have been, for instance, important tools in the work I've done with the Github data (for instance, the database Redis https://github.com/antirez/redis; the ipython interactive computing environment https://github.com/ipython/ipython) . But they are only part of the device-specific components of device-specific research. We have glimpsed cloud computing services such as Google's BigQuery. Other infrastructures and devices – programming languages, code editors, models – come from elsewhere. But this complex conjunction involves participatory fieldwork – developing code, learning to use tools, devices and infrastructures. This is the other

'disfiguring' object in working with numbers. The devices we to count events in the datastream morph the counts in particular ways that are sometimes desirable and other times frustrating.

Conclusion

I have been suggesting that STS research might develop its strong ethnographic sensibilities in relation to data event streams by sitting in an ambivalent space between formats and disfiguring elements, between the device-specific and its overflows. There are several different ways to understand what is at stake in this work.

One comes from Latour, Jensen, and Venturini, who ask:

> Is it possible to do justice to … common experieng ce by shifting from prediction and simulation to description and data mining? … Let the agents produce a dynamics and collect the traces that their actions leave as they unfold so as to produce a rich data set' (Latour et al., 2012)

I broadly agree with them but think there are hidden complications in their recommendation, because the data set is pre-formatted for particular actors and agents. If we are interested in alternative agencies, we have to work against the traces in some ways.

I started with Helen Verran's observation about work on numbers, and some ways I think it might be helpful. As she says, 'in any practical going-on with numbers, what matters is that they can be *made* to work, and *making them work* is a politics' (Verran, 2001).

How do we re-animate and open up the foundationist growth-curve numbers that abound in contemporary sciences and technologies as they accumulate and aggregate time-stamped data? Like Verran, I see the interplay between different kinds of numbering as generative. To do this, we need to find ways of working on formatted data that goes against its pre-formatted counting relations and uses. The massive availability of data from online databases is power-laden number work. Its very openness allows a certain kind of capitalising and governing based on data to expand. For instance, does the formatting of data streams tend to simply confirm and expand how the platform

already advertises itself? I think we could say that in many of the data-reuse situations act as mirrors for the platform.

By contrast, device-specific research that attends to the ways in which data is formatted, and how that formatting affects practices around the platform seems to me quite important in alternative to the much advertised (but actually often limitied) opennness of data.

A complementary facet of this work with data concerns indeterminacy or what Winthereik and Verran call 'double vision' (Winthereik and Verran, 2012: 48) in relation to the platform or device and what overflows it. Many contemporary digital platforms seek to capture and format these overflows. Vast and complex infrastructures have been built in the service of that end. We can't always know whether the data we are looking at comes from the platform or from a social practice that overflows that platform. The intersections and collisions between names in the Github event streams are complex and multiple. Even the imitation and copying of names points to the ways in which code repositories are forms of economic and political organization, as well as desires, concerns and passionate interests. The mixing of entities in names suggest that imitation works in plural ways, and that it is recombines different devices, groups, and organisations in a vast array of forms.

If our concern is to begin to explore the indeterminacy in the relations between what is specific to the devices or platforms and what is specific to the social practices that come to those platforms. I think ethnographic work on things like the flow of naming heads in a different direction to the formatting of data for purposes of platform services and enterprises. It cannot remain content with the familiar processes of reduction to social networks, or counting specific kinds of pre-formatted event types. In the case of Github, the names of repositories offer one way to begin to explore this ambiguity and indeterminacy, or what Riles called 'momentary apprehensions of depth.'

References

Barry A and Thrift N (2007) Gabriel Tarde: imitation, invention and economy. *Economy and Society*, 36(4), 509–525.

Butler J (1997) *Excitable Speech, A Politics Of the Performative*. London & New York: Routledge.

Gatens M and Lloyd G (1999) *Collective imaginings : Spinoza, past and present*. London ; New York: Routledge.

Grigorik I (2012) GitHub Archive. Available from: http://www.githubarchive.org/ (accessed 31 March 2014).

Palsson G (2014) Personal Names Embodiment, Differentiation, Exclusion, and Belonging. *Science, Technology & Human Values*, 39(4), 618–630.

Riles A (2001) *The network inside out*. University of Michigan Press, Available from: http://books.google.co.uk/books?hl=en&lr=&id=iOu_LP1w2LEC&oi=fnd&pg=PR9&dq=annelise+riles+networks+2004&ots=SKkvqn85hP&sig=cCS3Jakbv4tK24ttTEzy2wlq7iE (accessed 9 June 2014).

Russell MA (2013) *Mining the Social Web: Data Mining Facebook, Twitter, LinkedIn, Google+, GitHub, and More*. O'Reilly Media, Inc., Available from: http://books.google.co.uk/books?hl=en&lr=&id=_VkrAQAAQBAJ&oi=fnd&pg=PR4&dq=github&ots=JqiqtzTxmK&sig=sfea4ce1ue2XYt_dERD41VpSTS4 (accessed 23 May 2014).

Thung F, Bissyandé TF, Lo D, et al. (2013) Network structure of social coding in GitHub. In: *Software Maintenance and Reengineering (CSMR), 2013 17th European Conference on*, IEEE, pp. 323–326, Available from: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6498480 (accessed 23 May 2014).

Verran H (2001) *Science and An African Logic*. Chicago, London: The University of Chicago Press.

Winthereik BR and Verran H (2012) Ethnographic Stories as Generalizations that Intervene. *Science Studies*, 25(1).