2018

Algorithmic Sovereignty

Roio, Denis

http://dx.doi.org/10.24382/551
University of Plymouth

Copyright Statement

University of Plymouth

ALGORITHMIC SOVEREIGNTY

by

DENIS ROIO

A thesis submitted to University of Plymouth
in partial fulfilment for the degree of
DOCTOR OF PHILOSOPHY

January 2018

Acknowledgments

It has been an honor for me to be part of the Planetary Collegium and to get to know professors, colleagues and the visionary researches being conducted. I'm particularly grateful to Francesco Monico for engaging my curiosity about this research group, back at ISEA 2008 when we met on a terrace in Singapore, and later making it possible for me to enroll in this journey under the supervision of Antonio Caronia, with the benefit of engaging some of the most inspiring mentors I could ever desire as Derrick De Kerckhove, Pierluigi Capucci, Mike Phillips, Jill Scott, Geoff Cox, Joasia Krysa, Jane Grant and professor Roy Ascott.

All my loving gratitude for encouraging this research and supporting it through all difficulties goes to my partner Debra, who also inspired an important part of this thesis, and my mother Ines and my father Giancarlo.

Among the fellow travellers that have been indispensably precious to complete this thesis are Francesca Bria who lead the D-CENT project, Marco Sachy whose doctoral research at Leicester University should be seen as complementary to mine.

As it is evidently described in the research, communities have a very important role through all the work done at Dyne.org. For this thesis in particular I would like to thank the community of Devuan GNU+Linux and other fellow developers Franco "Nextime" Lanza, Enzo “Katolaz” Nicosia, Daniel "Centurion" Reurich, Gabriele "Asbesto" Zaverio, Ivan "Parazyd" Jelinčić and the inspiring and modest women, activist and Zen elder known as "Golinux".

I'm also extremely grateful for all the encouragement, suggestions and inspiring dialogues to: Annalisa Pelizza, Cecile Landman, Stefania Milan, Federico Bonelli, Geert Lovink, Evgeny Morozov, Hellekin O'Wolf, Natacha Roussel, Adnan Hadzi, Francesco Nachira, Christian Marazzi, Stephanie Rothenberg, Renée Ridgeway, Stefano Lucarelli, Andreas Broeckman, Christian Nold, Susanne Jaschko, Rob van Kranenburg, Patrice Riemens, Brian Holmes, Mario Canali, Inke Arns, John Hopkins, Ted Byfield, Simona Levi, Andrea Mayr, Felix Stalder, Maurizio Teli, Birgitta Jónsdóttir, Richard M. Stallman, Edward A. Shanken, Josephine Bosma, Aviva Rahmani, Marc Garrett, Andreas Broekmann, Susanne Jaschko, Armin Medosh, Ina Zwerger, John Cooper QC, George Pór, Adam Arvidsson, Tiziano Bonini, Cornelia Sollfrank, David Garcia, Alessandro Ludovico, Tiziana Gemin, Vito Campanelli, Yann Moulier-Boutang, Giorgio Griziotti, Tiziana Terranova, Andrea Fumagalli, Margrit Kennedy, Christina von Braun, Bernard Lietaer, Lavinia Hanay Raja, Amos Bianchi,

Ciro Aniello, Hanieh Abbasinik, Haytham Nawar, Antonio Radici, Andrea Guzzo, Alvise Gottieri, Gadalla el Badawi, Venzha Christawan, Irene Agrivine, Bronac Ferran, Matt Ratto, Fabi Borches, Vicky Sinclais, Daniel Hassan, Anja Kanngieser, Tatiana Bazzichelli, Enrico Bisenzi, Francesco Nachira, Walter Palmetshofer, Ted Byfield, August Black, Tatiana de la O and Pablo Caedes.

I would also like to thank all the organisations (and all the colleagues working in them) that have valued my work and that of the team at Dyne.org throught the years, as well partnered with us in project: the European Commission, Waag Society, Servus.at, Stadtwerkstatt, iDAT (Plymouth University), the Institute of Network Cultures, The City of Amsterdam, The City of Barcelona, the National Endowment for Science Technology and arts (NESTA), Centre d'Économie de la Sorbonne (CES), The Centre for Peace Studies in Croatia, Princeton Univ. (Math dept.), Trento Univ., NLNet, SIDNfonds, Copenhagen Institute of Interaction Design, IT University of Copenhagen, The Danish Society of Engineers, The Ars Electronica Center, The Transmediale festival, Distrowatch, Vereniging NLUUG, Hartware MedienKunst Verein, OSAlliance and PUBLICVOICELab, the Free Sofware Foundation, all the squatters of Amsterdam, The Disruption Network Lab, Neural.it, the Kingdom of Piracy, the MUTE magazine, the Hackmeeting, the Freaknet brotherhood, the Metro Olografix association and all my former colleagues at the Netherlands Media art Institute (Montevideo/Time Based Arts).

This research is dedicated to the memory of my beloved professor:

Antonio Caronia.

Author's declaration

At no time during the registration for the degree of Doctor of Philosophy has the author been registered for any other University award without prior agreement of the Doctoral College Quality Sub-Committee.

Work submitted for this research degree at the University of Plymouth has not formed part of any other degree either at the University of Plymouth or at another establishment.

This study was financed with the aid of a studentship from Gruppo Cabassi.

Publications:

- Sachy, M., Roio, D., Lucarelli, S., Lietaer, B., & Bria, F. (2017). D4. 4 Design of Social Digital Currency, DCENT Project Deliverable submitted to the European Commission.
- Bianchi, A. & Roio, D. (2011). Frames from the life and death of Jean Charles de Menezes. In *Computers, Privacy and Data Protection: an Element of Choice* (pp. 101-109). Springer
- Roio, D. (2011) "Re/think Re/design." *Technoetic Arts: A Journal of Speculative Research* 9

Word count of main body of thesis: 48535.

Signed….. Denis Roio

Date….. 13 March 2018

Abstract

This thesis describes a practice based research journey across various projects dealing with the design of algorithms, to highlight the governance implications in design choices made on them. The research provides answers and documents methodologies to address the urgent need for more awareness of decisions made by algorithms about the social and economical context in which we live. Algorithms consitute a foundational basis across different fields of studies: policy making, governance, art and technology. The ability to understand what is inscribed in such algorithms, what are the consequences of their execution and what is the agency left for the living world is crucial. Yet there is a lack of interdisciplinary and practice based literature, while specialised treatises are too narrow to relate to the broader context in which algorithms are enacted.

This thesis advances the awareness of algorithms and related aspects of sovereignty through a series of projects documented as participatory action research. One of the projects described, Devuan, leads to the realisation of a new, worldwide renown operating system. Another project, "sup", consists of a minimalist approach to mission critical software and literate programming to enhance security and reliability of applications. Another project, D-CENT, consisted in a 3 year long path of cutting edge research funded by the EU commission on the emerging dynamics of participatory democracy connected to the technologies adopted by citizen organizations.

My original contribution to knowledge lies within the function that the research underpinning these projects has on the ability to gain a better understanding of sociopolitical aspects connected to the design and management of algorithms. It suggests that we can improve the design and regulation of future public, private and common spaces which are increasingly governed by algorithms by understanding not only economical and legal implications, but also the connections between design choices and the sociopolitical context for their development and execution.

Contents

1 Establishing a theoretical framework **5**

1.1 Beyond the ethics of free and open source 7
1.2 Practical scenarios for algorithmic sovereignty 8
1.3 Practice based methodology 13
1.4 Personal facts 15
1.5 Algorithms in the digital dimension 17
1.6 Economy in the digital dimension 20
1.7 Labour in the digital dimension 25
1.8 Literature and algorithms 32

2 The Bitcoin experience **34**

2.1 The glory of algorithms 37
2.2 The activist sacrifice 40
2.3 The utopia of neutrality 41

3 Decentralized Citizen Engagement Technologies **43**

3.1 Research across fields and disciplines 46
3.2 Scaling down technological expectations 49
3.3 Freecoin's readable and sharable keys 49

4 Dowse: making visible the invisible **53**

4.1 Methodology: artist lead tech iterations 56
4.2 The Narrative Counts 58
4.3 Functional transparency 58
4.4 Status of the project 59

5 Devuan: the anatomy of a fork **62**

5.1 The Debian project 63
5.2 The legacy init system: sysvinit 65
5.3 The controversial innovation: systemd 66
5.4 Summary of arguments for sysvinit 67
5.5 The General Resolution vote 69
5.6 Response to the GR vote 74
5.7 The aftermath of systemd in Debian 78
5.8 The Debianfork declaration 82
5.9 General response to Devuan 85
5.10 Devuan is born 85
5.11 Conclusion 87

6 Artistic interlude: entropy and design for living settlements **90**

6.1 OBREDIM: a design methodology going well beyond Permaculture . 93
6.2 Seven Layers and REALBOTANIK 94

7 sup: an alternative model for UNIX privilege escalation 97
7.1 Implementation details 99
7.2 sup's sourcecode 102

8 Gufo Volante: live coding for data analysis 116

9 Decentralised Citizens Owned Data Ecosystem 120
9.1 Concept and economic analysis in DECODE 124
9.2 Interdisciplinary methodology to bootstrap a privacy aware, decentralised innovation ecosystem 126
9.3 Scientific and technological advances beyond the state of the art . . . 129
9.4 DECODE, context and perspectives 132

10 Conclusion 134
10.1 For an holistic analysis of algorithms 144
10.2 New forms of rationality as new forms of liberation 145

11 Appendix 147
11.1 Quick reference of projects 147
11.1.1 Git repositories 147
11.1.2 Fora 147
11.1.3 Audiovisuals 147
11.2 Critical Engineering Manifesto 148
11.3 Dowse and the tactical withdrawal 149
11.3.1 I 149
11.3.2 II 150
11.3.3 III 151
11.3.4 IV 153
11.3.5 Concretely? 156
11.4 Community reactions to Devuan 156
11.4.1 Sysvinit scripts vs. systemd service files 176

Bibliography 182

List of Figures

1 "Control" by Pawel Kuczynski, 2016 10
2 Forkbomb tattoo on an anonymous programmer in India 19
3 Semiotic square of creation 20
4 Semiotic square of proprietary creation 21
5 Semiotic square of virtuose creation 22
6 Folded semiotic square of creation 24
7 The Mechanical Turk as represented in an ancient illustration, signed P.G. Ponty 26
8 Sample screenshot of an early version of the "ESP game" 27
9 Excerpt from "The Sheep Market" by Aaron Koblin 28
10 Exerpt from "Learning to love you more" by Harrell Fletcher and Miranda July 29
11 Exerpt from "Artists' Cookbook" by Allison Wiese 29
12 Self-portraits of M-Turk workers have been commissioned as a paid task 32
13 Price oscillation and volume transaction chart for Bitcoin between September 2011 and 2013 36
14 The Alpaca Coin, an early symbol in Bitcoin, still present in the works of Satoshigallery.com 39
15 D-CENT pilots 44
16 Spiral representation of LEAN UX methodology in D-CENT 46
17 A sketch of the FXC Secret Sharing encryption scheme as adopted in Freecoin 51
18 Dowse has found a place in the practice of a growing number of artists and was featured in a workshop at the Live Peformers Meeting in Amsterdam during 2016 (photo courtesy of Federico Bonelli) 57
19 Dowse pendulum photo by Anatole B. Shaw, 2014 61
20 Summary of Debian's GR resolution votes dataset by dasein 77
21 A graphical representation of the Continous Integration infrastructure planned for Devuan 86
22 Graph of ideas and inspiration by the lecture "Design patterns between Free Software and Permaculture" contributed by an attendee to a workshop in held in 2012 on the premises of RuralHub 91
23 Photo of the REALBOTANIK installation at display in 2015 inside the "Glazenhuis" pavillon inside Amsterdam's Amstel Park 95
24 Reproduction of the artwork "Seven Layers", part of Entropical . . . 95
25 Bar graph resulting from the code above 118
26 Illustration of DECODE methodology of LEAN development cycle in relationship to city pilots 127

27 The technical architecture of DECODE explained on the ISO/OSI 7 layers model 130

28 The research and development on distributed ledger technology in DECODE focuses not only around its cryptographic model, but also on the semantic analysis of language 132

List of Tables

1 Table of projects covered by this thesis 14

2 Example of a project information table 15

3 Information on the research about the Bitcoin project 34

4 Information about the D-CENT project 43

5 Information about the Dowse project 53

6 Information about the Devuan project, fork of Debian 62

7 Information about the Entropical artistic project 90

8 Information on sup UNIX privilege escalation project 97

9 Information on Gufo Volante live coding project 116

10 Information about the DECODE project 120

11 Reference table of all listed projects 147

1 Establishing a theoretical framework

When this research started in 2010 its general direction was towards connecting my practices in art and technological development, while I was prompted by a rather naïve idea of what the thesis question would be. For the first two years I mostly worked to refine the question and contextualise it within practice lead research paths, rather than providing any answer to it.

As a matter of fact the deviant paths are as much a part of this research work as those whose influence is clearly formulated in the main research question. It is said that asking the right questions is actually 50% of the research work. I have taken this assumption rather seriously and found that once the question was ready, some projects fell nicely in place to form an answer, even if conducted in most disparate contexts.

To explain the question answered by this research is first and foremost necessary to acknowledge the growing importance of algorithms when studying governance mechanisms. It is the increasingly relevant role of algorithms within society that makes so important the questions and the answers this thesis provides.

The governance of an algorithm, for which the term "algorithmic sovereignty" is proposed, is the condition this thesis takes into account. Opposed to sovereignty, what we can observe today in many context where algorithms are deployed is a condition of subjugation for which the living participants to these systems do not even share knowledge of the algorithms governing their spaces.

The logic of algorithms is often invisible, while only their results are manifest. Communities around the world adopt their use, ending up in a regulatory framework with imposed rules that can only be guessed and, in most cases, never negotiated. For example in the well developed case of algorithms governing Internet search mechanism, researchers note that we are dealing with an unquestionable hidden organisation of knowledge that is impossible to review and is in fact kept secret. As Renée Ridgway points out:

> Search algorithms crawl vast amounts of data and organise it according to, for example, what the advertiser has paid the programmers of algorithms to find. As they sort through the data a hyper-complex infrastructure of daily search requests emerges. We cannot, however, see the mechanisms of how our searches are manipulated by the assumed 200+ proprietary algorithms employed by Google. Search is thus a 'hidden organisation' - or a hidden organising process that keeps its secrets of control sequestered from the user. (Ridgway, 2015)

Such hidden algorithms are growing in power and importance not only for their role on the market, but also for their role in informing public choices, public life and the economies that drives societies at large. The metaphore of a "black box" is perfect

to understand the role that algorithms have taken in our contemporary and highly digitised world, a metaphore fitting many industries and in particular reputation, search and financial sectors, where the intelligibility of algorithms is not only made impossibile by cognitive limits, but also by trade secrets. While the solution to the growth of influence of the "black box" can be envisioned as a movement for an "Intelligible Society" (Pasquale, 2015), it is still difficult to understand under which conditions this can happen. On this topic Frank Pasquale concludes his book "The Black Box Society" with a call for "educated citizenship":

> Educated citizenship today requires more than an understanding of government, which is just the tip of an iceberg of social organization. It also demands an understanding of the companies that influence our government and culture. [...] It is time for us as citizens to demand that important decisions about our financial and communication infrastructures be made intelligible, soon, to independent reviewers - and that, over the years and the decades to come, they be made part of a public record available to us all. Black box services are often wondrous to behold, but our black box society has become dangerously unstable, unfair and unproductive. [...] Those are the tasks of a citizenry, which can perform its job only as well as it understands the stakes.

Taking up the challenge from Pasquale's conclusions, this thesis aims at defining what technical and social processes can be in place for making algorithms "intelligible" to "independent reviewers" and to envision the conditions in which a "citizenry" can "understand the stakes". As it goes with my interpretation of this task, implies that citizens, or simply participants to a system, may gain sovereignty on their algorithmic infrastructures.

The use of the term sovereignty here and in the title of this thesis can be well related to the contemporary notion of it in the debate for "food sovereignty" as defined by associations of rural smallholders in Latin America as "the right of each nation to maintain and develop its own capacity to produce its basic foods while respecting cultural and productive diversity" (Campesina and others, 2003). Furthering this anaology, a loss of sovereignty refers not only to the distribution and reproduction of seeds, a consequence of the globalised patent-based food market that activists of Via Campesina warn us about (Desmarais, 2003); but also to a "monoculture of the mind" (Shiva, 1993) orienting or even forcing participants to be mere consumers (Ghose, 2004) and fostering a reductionist system of knowledge which ignores the complex socio-political relationships laying beyond the mere adoption of a technologic solution.

The path engaged by this thesis to solve this challenge is not a theoretical analysis and not even a laboratory "in-vitro" experiment. Due to the interdisciplinary nature of this quest, I believe it is necessary to take a practical and participatory approach to algorithmic sovereignty, observing contexts and cases where it materialises "in-vivo",

informing participants and even motivating them to take control of algorithms governing their interactions. The answers provided by this thesis are projects illustrating the means for such conditions to take place, the technical, political and economic solutions that can be engaged by this educated citizenry to realise an intelligible society.

1.1 Beyond the ethics of free and open source

When speaking of intelligible algorithms it is rather obvious to think of the free software movement and of open source business models that enable the scrutiny of algorithms by thousands of experts, as well the freedom to modify them and distribute modifications.

The free software movement explicitly put forward more than 30 years ago an ethical and legal framework that addresses this situation by establishing four fundamental freedoms:

0. The freedom to run the program as you wish, for any purpose.
1. The freedom to study how the program works and to change it so it does your computing as you wish.
2. The freedom to redistribute copies so you can help your neighbour.
3. The freedom to distribute copies of your modified versions to others.

The question posed by this thesis can be now reformulated also in relation to this important historical heritage, which has unfortunately played only a marginal role when compared to the industrial scale of digital economies worldwide. Assuming different degrees of knowledge of an algorithm can be gained, are these four freedoms enough, or is there anything else defining the ethical dimension for an algorithm to allow the sovereignty of its participants? Arguably the freedoms enunciated above are not applied where they should be, in contexts in which the use of algorithms is invested by enormous responsibilities.

But even when the access to algorithms complies to these ethical concerns, would that be enough to verify a condition of sovereignty for all living participants? Is the dichotomy between "open" and "closed" algorithms enough to fully interpret the conditions of governmentality (Foucault, 1991) we are observing and envisioning? Is transparency, as opposed to secrecy, enough of a condition to make algorithms functional to the creation of an intelligible society?

By moving forward with the question this thesis poses I affirm that "openness" and "transparency" are not sufficient conditions to realise the good ethical propositions of the free software movement. It now becomes even more urgent to understand this limit and envision how to move forward as computing becomes more pervasive and relevant to many functions supporting the living economies of our planet.

Then furthering the horizon of this investigation in a more interdisciplinary

framework that doesn't neglects the mutual influence that social dynamics have on technology, the question may turn to the methodology adopted: how can algorithms be studied in the context of governance? How can we recognise and what are the important traits for an algorithm to enable the sovereignty of those adopting it? What sort of knowledge and interaction, what sort of literature we need?

Algorithms are active in embedding sensitive decisions in their technical meanders. However, the quest for an answer cannot stop at this basic assumption, cannot stop at the analysis of source code, nor the analysis of code execution is enough.

By contributing to the modern discourse on sovereignty (Bratton, 2016), which requires a move beyond the conventional notion of nation-state, it is sensible to assume that the digital dimension constitutes a territory, or rather an infinite abundance of territories. Then, as for every territory, the pattern language (Alexander, 1977) of architecture has a social, political and ethical relevance, as it defines the conditions of inclusion and exclusion of inhabitants.

To define a valid approach to the problem is also important to note that algorithms govern the living not by intervening and reeducating their bodies and minds, but by making inferences from data gathered from them and acting on these. It is not by discipline, but by Deleuze's notion of control and by Foucault's notion of government that a political paradigm can adopt algorithms as technologies of control (Amoore, 2011). What becomes observable then are dynamics of de/subjectivation (Ek et al., 2007) taking place within a social architecture (Bianchi and Roio, 2011) that is governed by algorithms. When imposed as a black box sealed by trade secrets and intelligibility, algorithms can be used to create a topological space where new relations between derived values can be created and calculated in ways that pay no attention to other aspects of the life of participants (Arvidsson, 2016).

1.2 Practical scenarios for algorithmic sovereignty

It is important to extend this reflection to an array of real-world examples and engage a dialog with developers to raise their awareness. Due to their inherently higher ethical standards, free software projects may provide advanced starting points from which to contextualise and formulate algorithmic governance studies focusing on the problematic of sovereignty. On the other hand, large deployments of proprietary software algorithms constitute a starting point for critical analyses, because they provide concrete cases of de/subjectivation, loss of sovereignty and alienation of large masses of participants.

Unfair situations then arise in which participants in a system not only lack agency, but are ill-informed about the algorithms governing it: these situations can be considered as root factors for societal discontent. Examples of protests and constructive acts of resistance are proliferating nowadays across the media-landscape

of Internet based social-networks, offering an interesting variety of approaches and analyses (Lovink et al., 2013). While privacy concerns seem to be the most aired discontent by non-sovereign participants of social networks, the few communities that are well conscious of the sovereignty implications and are empowering themselves to develop alternatives:

> Contributing to the design and development of technopolitical tools enhances 'technological sovereignty'. There are examples of such a rich contribution by citizens, for example the development of communal radio and television broadcasting, the launch of the first non-military satellite into orbit, the invention of free software and licenses, and even the first news portal on the internet with an open and anonymous publication system, set up by the Indymedia network in 1999. (Cabello et al., 2013)

In case of Facebook, the most popular social network in the western hemisphere, all efforts of "liberation" are representative of this discontent[1] but despite their popularity and critical momentum haven't managed to empower the realisation of any constructive alternatives. Following up with the tools of analysis provided by this thesis, it is evident how Facebook explicitly counter-acts any possibility for its participants to gain algorithmic sovereignty: it denies all possibilities of participation in the decision-making process of its own algorithms, as well strictly regulates the opportunities of interoperability for the data it gathers. While Facebook's vast database can be considered as knowledge commons (Vercellone et al., 2015), it is very far from being managed according to any of the principles of commons based economies. The modus operandi of the Facebook company is tailored to disengage any attempt of its participants at turning critique into action and at creating alternatives, such that the only way for participants to gain sovereignty is by leaving Facebook to create an entirely new territory. Few communities have enough time, technical skills and awareness of sovereignty dynamics to engage in such a task and succeed in gathering a population large and diverse enough to thrive like its predecessor. This has been partially possible by crossing different language domains, when at the beginning of the Facebook phenomenon its language base was prominently English.

A situation like the one described above does not offer any constructive possibility to advance the sovereignty of participants within the platform, whereas the company governing it has no incentive to grant any of its sovereignty to them. The only opportunities for participants of such a system to gain algorithmic sovereignty lie in two possible lines of action: either "fork" the project, with great expense of resources and effort and with an uncertain outcome (Tkacz, 2014) or rely on a strong regulatory framework that opens Facebook's governance to its participants; both options are

[1] See the "Facebook Liberation Army" (FLA) initiative promoted by the Waag Society in Amsterdam https://waag.org/en/project/facebook-liberation-army has developed a platform for engagement and debate also in cooperation with notable local institutions http://fla.waag.org

very uncertain and unlikely to succeed.

Figure 1: “Control” by Pawel Kuczynski, 2016

While writing this thesis, the popular phenomenon of a game called PokemonGO (Serino et al., 2016) offers another concise example while expressing well the urgency of this research, artistically depicted by an artwork by Pawel Kuczynski. This proprietary game creates an economy based on the collection of digital icons that are infinitely reproducible by the company owning the game. Players can collect the icons by reaching certain physical locations. Locations are mapped in the real world without an apparent logic (the algorithm is concealed) and only in part relate to the movement and position of other players. It is deducible that the way this algorithm creates incentives is also disconnected from the physical world, as it often make puppets appear in places that are unreachable or dangerous to reach for young people distracted by their own mobile phone. The enormous popularity of this game has created embarrassing situations in which large amounts of adolescents concentrated at a single point, motivated by playing with the icons and gaining points that are arbitrarily

created by an algorithm, at the risk of stampede or situations in which users are at risk of abduction[2]. In case of PokemonGO it became immediately clear to the institutions in charge of safety for civil society how the control of this algorithm is relevant for their own sovereignty, as the game influences the access and occupation of physical space by large portions of population, whilst remaining outside of participant's control. It is plausible that grim scenarios can be caused by errors or mischievous use of the algorithm behind such a popular game, with extreme consequences of public unrest. In any case, the creation of incentives for the economy of this game is unknown to all players and out of reach for any form of governance connected to them, despite the game can directly affect the conditions of the public space they move across.

Another relevant example is provided by the debate ongoing in the United States about algorithms deployed for criminal profiling. In a recent and popular article researchers and journalists[3] advanced concerns and produced data analyses proving that "the software used across the country to predict future criminals" produces results that are biased against people of colour (Angwin et al., 2016). While it is obvious that the ethical and political implications of such findings are enormous, the assumption of these findings has been immediately challenged in a "rejoinder" published by academics and public sector officials, arguing that the analysis was not properly conducted, pointing out how actuarial risk assessment instruments (ARAIs) "hold considerable promise for criminal justice reform in that they are capable of better informing what were previously subjective and indefensible criminal justice decisions" (Flores et al., 2016). Within this episode the most interesting aspect related to my research is the methodology adopted by both factions opposing each other in the debate: they both refer to data samples and data analyses, whilst superficially describing and enumerating ARAI software products commonly deployed in the USA[4]. There is no literate commentary on these algorithms, only a heated debate on the variety of the sampling and on the validity of the data analysed. The code of the algorithms is barely debated and it can only be evinced (should we say reverse engineered) from the results that have been produced. It is fair to say that a delicate discussion on the ethical nature of algorithms cannot be exempt from an analysis of

[2]"Governor Andrew M. Cuomo today directed the New York State Department of Corrections and Community Supervision to restrict sex offenders under community supervision from using Pokémon GO and similar games. In an effort to safeguard New York's children, the Governor also sent a letter to software developer Niantic, Inc. requesting their assistance in prohibiting dangerous sexual predators from playing Pokémon GO." retrieved on 7 August 2016 from https://www.governor.ny.gov/news/governor-cuomo-directs-department-corrections-and-community-supervision-restrict-sex-offenders

[3]"Facial-Recognition Software Might Have a Racial Bias Problem. Depending on how algorithms are trained, they could be significantly more accurate when identifying white faces than African American ones." retrieved on 7 August 2016 from http://www.theatlantic.com/technology/archive/2016/04/the-underlying-bias-of-facial-recognition-systems/476991/

[4]Among the ARAIs is prominently debated the use and results of "Northpointe COMPAS", but also acknowledged the use of "FaceFirst facial-recognition software" and Cognitec's "AFIS" system for facial recognition and fingerprint retrieval,

the computational literature at the core of ARAIs. Unfortunately this is not possible, as most ARAIs are proprietary software whose code is considered an industrial asset covered as trade secret: beyond the consideration of the results produced by these algorithms. The adoption of these algorithms is therefore unethical because it denies the very possibility to analyse and debate them in detail.

Another relevant example is Tor, the most popular software to establish anonymous connections that can easily bypass censorship and firewalls, well useful to support freedom of speech in situations of crisis and tyrannical prevarication. The software has an ethical mission: it is free and open source and gathers the praise and participation of a community with high ethical standards. Tor progresses in the flux of its active development with changes to its core algorithms contributed by the small number of people who can read its code, while such changes may have a huge impact on operations and the life of millions of people around the world judging from its popularity and statistics of usage. It is impractical to think of an efficient way to implement a democratic and participatory process for the sovereignty of Tor's algorithms, considering the complexity of the software. However steps were taken by Tor project leaders in this direction: a non-profit foundation has the responsibility to ensure the original integrity of the project's ethical mission; a well qualified board of trustees oversees its operations; and a larger team of people actively informs the public about changes operated on algorithms, relating them to the general direction taken by the project and the features it can offer. While it may be confusing at times to distinguish such efforts from marketing campaigns, it is important to understand the differences. The Tor project works not only to develop its algorithms, but also to maintain the sovereignty of its algorithms in the hands of the people who are affected by them, beyond the cognitive difficulties that such an effort presents. Among the political difficulties, there is the incumbent need of law-enforcement apparata that claim the possibility to tap into Tor's communications to investigate its criminal uses: this political negotiation is something that ultimately undermines Tor's mission. Many other free and open source projects can be found to progress more or less consciously in ways that require focus not just on technical or political problems, but that need to relate to very complex and hardly predictable scenarios. This thesis then aims at demonstrating how a interdisciplinary approach can be beneficial to document and navigate these situations towards ethically sound horizons of development, to respect the freedom of the communities of adoption, to help comprehend their motivation in light of uses that can benefit societies.

Lets take another example, that of big and influential software projects as the SAP[5] line of proprietary products: born and grown to store, manage and analyse large amounts of data in diverse environments, from medium sized financial accounting to

[5]SAP is a software for system analyses and programme networking, its acronym stands for "Systems, Applications & Products in Data Processing".

nation wide public procurement. The responsibility for SAP's algorithms is huge, yet the accountability for errors that may occur is only established on a legal level, with corporate liability formulas. For a long while there had been barely any understanding of the underlying algorithms and changes operated by SAP on its code by any of the constituencies relying on it for mission critical operations. Though the company exists since 1973, only 40 years later in 2013 did it promote an "open source" program, mostly to offer training to customise front-end algorithms useful only for *appropriation* and *distribution*. However SAP protects from view the algorithms constituting its back-end and especially its most complex (and less maintainable) ones, for instance artificial intelligence algorithms operating on the data. One may argue that corporate governance is a walled garden and that knowledge transfer happens inside, but the influence of SAP's algorithms on the living world is immense and corporate culture can barely include and give access to the majority of participants affected by these algorithms, beyond the small minority that have access to trade secrets and corporate assets.

In the case of proprietary software like SAP, where algorithms are literally implementations of forms of governance, those who know how they work and evolve can take informed decisions in their own interest. If algorithms are not known by all participants, only a few will be able to make informed decisions about their own positioning within the digital space governed by these algorithms.

> Black box insiders are protected as if they are wearing a Ring of Gyges - which grants its wearer invisibility but, Plato warns us in The Republic, is also an open invitation to bad behavior. (Pasquale, 2015)

1.3 Practice based methodology

The corpus of this thesis are the practical projects described in it. In these projects the algorithms are the expression of socio-political conditions and can be observed as conceived and deployed to enable different levels of security, reliability, trust and confidence, as well as nurturing the constituency of people's movements, their values and their aspirations.

This is an explicit choice: to conduct this research "in-vivo" rather than "in-vitro", operating inside living conditions to effectively play out assumption and consequences in a real socio-technical context. This approach has many collateral effects and primarily that of relating directly with my professional life as a software developer, informing a practice that has always been geared towards community based development.

Arguably a participatory research methodology engaging community based development has no other choice than operating in-vivo, to present the concrete achievements

of each project as the very result of the research. The grade of adoption and the community response to each projects are the main indicators of this research outcome.

I participated in first person to every project documented by this thesis, addressing concretely and directly the research questions. Some present source-code designed to improve security by adopting a minimalist approach; another explores the reasons for a fork of an entire operating system operated by its own community of users and developers; another investigates the reasons and motivations that move large numbers of people to participate and invest their resources in a decentralised network; another shows the process of designing algorithms starting from a methodology based on field research to end with the specification of modular software components and cryptographic algorithms. In all such projects I have taken an active and sometimes leading role. This thesis contextualises all the projects as a concrete outcome for a research that is practice based and interdisciplinary.

Below a brief overview of the projects:

Table 1: Table of projects covered by this thesis

Project	URL	Year	Nature	Advancement
D-CENT	http://dcentproject.eu	2013	R&D	Methodology
Bitcoin	http://bitcoin.it	2010	Research	Context
Devuan	https://devuan.org	2014	Software	Participatory
Dowse	https://dowse.eu	2013	Software	Practice based
Entropical	http://entropical.org	2015	Artworks	Context
sup	https://sup.dyne.org	2016	Software	Practice based
Gufo Volante	https://gufo.dyne.org	2016	Software	Practice based
DECODE	https://decodeproject.eu	2016	R&D	Practice based

Ultimately, seeking a practical outcome, the ambition of this research is to inform the growth of the research organisation I have contributed to create, Dyne.org, whose horizon of progress and future priorities are outlined in light of the conclusions.

The making of Dyne.org responds directly to the need for new forms of institutions that cultivate, preserve and develop further a sort of "societal awareness of algorithms", drawing through the complexity of various disciplines that are progressively affected and enhanced by algorithms. However, this conclusion is not just calling for the institutionalisation of new knowledge, simply projecting the problematic of governance into the digital dimension. It highlights a concrete, urgent need for more socio-political awareness on processes that are currently taking place. We can already relate to this urgency by bringing our observation far from the pages of an essay and straight to the world in which we live.

Table 2: Example of a project information table

Project name	Dyne.org foundation
Home online	https://dyne.org
Started in	1999, 2013 as EU research org
Nature	Think/do tank, Software house
Advancement	New research institution

A concrete outcome of this research has been that of consolidating the Dyne.org project with a solid and sustainable practice that can well be related with institutional and public needs. Dyne.org is a community based organisation whose aims are directly inspired and informed by this thesis: it is a interdisciplinary community of hackers, activists and artists gathered around the notion of an "independent software house". The meaning of "independent" refers to the nature of the incentive motivating the work of developers. Most software houses act on commission from clients, are supported by commercial or speculative interests and are driven by profit, whilst they marginally relate to the communities which will adopt their code. In these cases software is perceived as a product and the relationship to its community of adoption is defined as a service model. In Dyne.org all this is different: the approach is that of maintaining a long term process around the development of software, facilitating a close relationship between the code and its communities of adoption. There is no provision of services, rather than an articulated process of knowledge transfer through phases of *Creation*, *Sharing*, *Distribution* and *Appropriation* of algorithms. Dyne.org research and development activities all aim at empowering communities with new and existing algorithmic knowledge to facilitate their sovereignty and stewardship of knowledge commons. A fundamental belief motivating Dyne.org's operations is that, well beyond popular campaigns for privacy awareness and open access to public data, it is important to develop algorithmic literacy and facilitate the agency of communities in these endeavours. This belief goes well beyond a generic idea of neutrality for algorithms: it facilitates a new sort of rationality, still recognisable as an underpinning of democratic societies, that values transparency, freedom and participation also in relation to the digital territory and in particular with algorithmic governance.

1.4 Personal facts

As a brief self-ethnographic effort, this chapter briefly informs the reader about the different contexts and readings that help me draw this research path, a journey for the conciliation of different passions in humanities and computer system engineering.

I started my PhD studies back in 2010 with supervisor Antonio Caronia, a renowned

expert of cybernetics in Italy. Admittedly, from the early days of my studies I had a hard time reaching the conciliation of different passions for humanities and for computer system engineering. To my great relief, this dichotomy was considered as a resource in the studies that Antonio Caronia conducted and that the Planetary Collegium encouraged.

I thought of my work as that of a software artisan and the choice of low-level languages like C was fundamental for my interest, seeking as little mediation as possible between the code and its visible results, shortening the route across the ISO/OSI[6] layers. For more than 25 years, I have considered the elegance of algorithms as an admirable form of beauty. My practice has been inscribed in the "net.art" generation of artworks, particularly between 1995 and 2005 (Lampo et al., 2005), but the introduction to the world of net.art was mostly accidental. In the Italian "media activist" scene of the '90s there were many people gathered at the "hackmeeting" as "AHA" network (Activist Hacker Artist).

Having being trained in philosophy, semiotics and linguistics, my studies are grafted into the continental tradition of post-modernism and structuralism. During this study period I have engaged in reading several authors and, strongly encouraged by Caronia's last cycle of lectures (Caronia and Bianchi, 2012), deepened my knowledge of Michel Foucault's oeuvre, in particular regarding his definition of de/subjectivation processes, biopolitics and their relation to the birth of the liberal political tradition and the concept of neutrality in governance (Foucault, 2008). Among many other substantial readings are the trilogy "Homo Sacer" (Agamben, 2005) and the critical views on contemporary global governance shared in "Commonwealth" (Hardt and Negri, 2009).

As an activist, my agency has been mostly directed at the dynamics of bottom-up participation, moved by the ethos and joyous discovery of collective autonomy. While engaging in sometimes difficult journeys, I found myself in the middle of situations to which I deeply relate while simultaneously making an effort to consider them as a subject of study, collecting first hand materials about what I am witnessing and suspending my judgment while reporting them. Following this endeavour and contextualising it for this PhD, I found that the methodology of participatory action research is best suited to analyse the process of my involvement in larger community-based projects such as the design and realisation of operating systems; one in particular is discussed in this thesis.

[6] A general architecture model established by the International Standards Organization in 1978 to describe communication systems, allowing open systems interconnection.

1.5 Algorithms in the digital dimension

The first working title for this thesis was "Libre made Flesh. Generative Patterns for Alternative Economies". This diversion deserves an explanation. This early title refers to another PhD thesis by Prof. Florian Cramer, defended in 2006 at the Department of Philosophy of the Free University of Berlin, originally titled "Exe.cut[up]able statements, Poetische Kalküle und Phantasmen des Selbstausführenden Texts" and later translated into English as "Words Made Flesh". I report here the English abstract of Cramer's thesis:

> This study reconstructs a literature whose language performs computations and algorithms and whose history spans from the antiquity to present. With magic and Pythagorean mathematical poetics as its prototypes, it includes such diverse forms as kabbalist and lullist language combinatorics, word permutation poetry, ludistic poetry, calculi of text collage, aleatoric, stochastic and recursive texts, oulipotic constraints, computer-generative literature, poetry in programming languages, and codeworks. They all expose calculi and algorithmics as a dimension of language, writing and literature, in the same way visual and sound poetry extrapolate graphetics and phonetics.

> A common characteristic of algorithmic literature is its semantization of computations. Computation are treated less as symbolic processes separate from the text, but as poetic languages in themselves. The common denominator both of Renaissance permutational poetry and contemporary codeworks is the text that executes itself, as both an idea and a utopia. As a result, the imaginations superimposed onto computations become more remarkable than the computations themselves. Sometimes within the same century, total art and anti-art, mysticism and technicism, order and chaos are programmatically inscribed into identical forms of language computation, yielding a phantastic literature that, instead of merely describing its phantasms, actually performs them on the level of its signifiers.

Cramer's research is a unique and fascinating account on the history, aesthetics and power of algorithms that moves forward by specifically describing their power by focusing on the art history context and what could be definied as a liturgy around code. Its English title echoes an evangelic prophecy, that of "words made flesh", whose comprehension beyond the religious anecdote resonates also in Agamben's opera, "Homo Sacer" (Nikolopoulou et al., 2000), and once again suggests the proximity of the digital territory with the birth and history of alphabets, a thesis looming through the body of work by Prof. Derrick De Kerchove.

Algorithms are not a mere mathematical or logical construction but can be used to describe legal and social conditions on a large scale. In my own perception and interpretation of their nature, algorithms are strictly bound to their digital dimension: an abstract dimension that presents peculiar conditions for everything that exists in

it. First and foremost, this is because algorithms are expressed in alphabetical form and the digital dimension can be traced back to the first uses of the alphabet. Such uses were intended to establish contracts and the service of carving cuneiform symbols into stone was mostly provided by literate scriba to settle important agreements and refer to them over longer time periods. Fascinating artefacts of such kind are collected in the Anatolian Museum in Ankara and contain contracts comprising weddings and trades of large herds of cattle. Considering the proliferation of known algorithms and their constant modification and adaptation, the field we are talking about spans across millennia and many different cultures. The etymology of the word "Algorithm" comes from the name of a researcher in astronomy and geometry, Muhammad ibn Musa al-Khwarizmi, active in Baghdad at the court of the Abbasid Caliphate, who is also believed to have given the name to Algebra.

Even before algebra, alphabets can be considered the early manifestation of what is defined today as "the digital dimension". The "digital dimension" is a territory with a peculiar property: the possibility to duplicate all that populates it, de-facto creating a space for infinite abundance of what exists.

Such a condition for the replication of digital artefacts is well explained as a progression of the condition for the mechanical reproduction of art (Benjamin, 2008) and it is manifested in several contemporary instances of art currents and art movements embracing duplication as a medium for the realisation of social sculptures (Goriunova, 2016). Much as with alphabets, the social character is an underpinning of the digital dimension and the artwork is also a process with which other artists can engage.

Digital algorithms are the medium used for software art (De Souza, 2010). This approach has been the underpinning of my artistic production since the very beginning, with the creation of HasciiCam (Carettoni and Laniado, 2005) and its code being used in various performances and interactive installations, in contrast with other ASCII-art based artworks which were focused on the final product and did not release the code to realise it.

With the Shell Forkbomb (Horst, 2008) I've released "in the wild" and encouraged the reproduction of a concise and elegant logic bomb (Dencker, 2011). This forkbomb is nowadays found in several tattoos on people around the world (Mendieta, 2013), which is the ultimate statement of its viral nature (Raley, 2014) and the viral nature of critical approaches to computing (Brock, 2012).

The way to artistically engage with the digital world in my own practice has been always that of looking beyond the form and conditions offered by the medium, but dig deeper in the social, cultural and perceptual connotations attributed by participants to the same space, be it a reformulation of vision according to alphabetical patterns or the elegant representation of a virus. This because the digital dimension is not a medium, but primarily an social and cultural space shaped by algorithms and at

Figure 2: Forkbomb tattoo on an anonymous programmer in India

the same time a pre-individual milieu (Stiegler, 2009) where affectivity can travel even faster than perception and the opportunities are open for psychic and collective individuation. The condition of "being digital" (Negroponte, 1996) should not naively suggest to de/materialise atoms in bits, but lead to the elaboration of new and often critical ways of relating with power, perception and cultural boundaries. Algorithms, as immersed in the social political and cultural conditions of the interactions they govern, are to be considered "digital objects" (Hui, 2016) as relational milieus where we can dare to move our first steps of comprehension, hereby explicitly adopting the fascinating new interpretation Hui gives of them, yet narrowing down our focus on the governmentality of a variety of relations in place around specific socio-technical artefacts.

1.6 Economy in the digital dimension

The second part of the title "Generative Patterns for Alternative Economies" reflects my enthusiasm for the potential of how free / open source software (F/OSS) can work both as a community and as an economy (Coleman, 2009). This section elaborates on the economic aspect of the digital dimension, providing a useful cathegorisation that will be used as a reference in the rest of this thesis to define and relate the economic feature of algorithms.

With the definition of "Alternative Economies", I have tried, moving across the abstraction to patterns, to describe what is generative in the F/OSS approach to "copyleft" (Stallman, 2010) economies, juxtaposing it to modern economic models based on patents and scarcity.

I later rejected the theoretical exploration of such concepts, as it is out of the scope of this thesis to analyse them in the context of economic literature. However, the study I have conducted on this matter is worth mentioning here to provide more context on F/OSS and its qualitative innovation for knowlege.

In order to describe the alternative traits of such economies I have used the Greimas Semiotic Square (Greimas, 1983) and organised the four main moments of immaterial value circulation along the lines of this narratologic device.

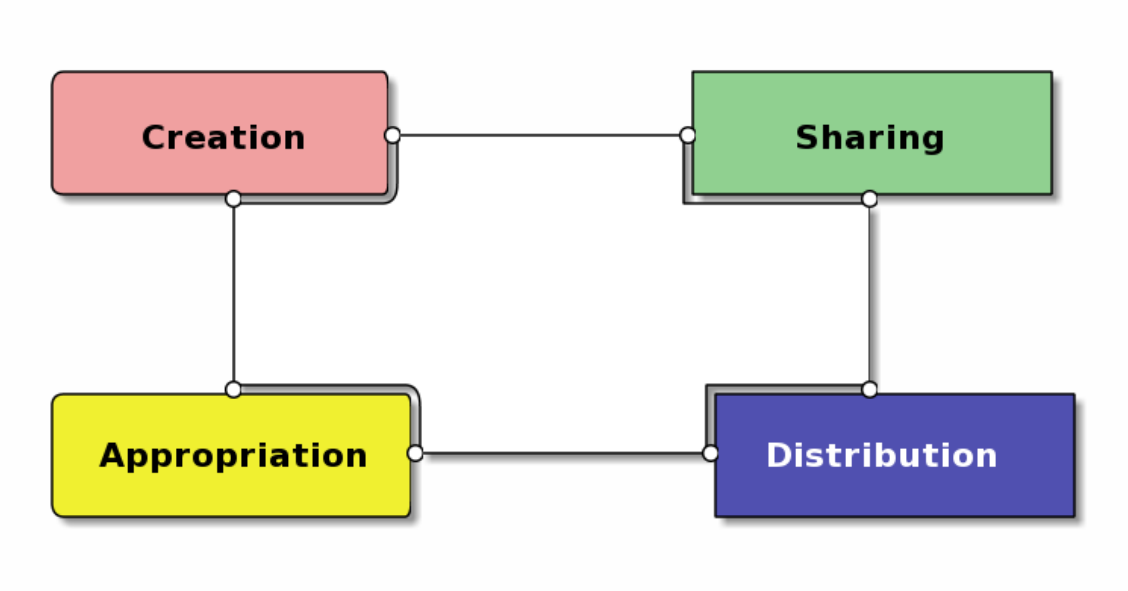

Figure 3: Semiotic square of creation

I have chosen the following nouns to describe the four moments:

- **Creation** is the inception and realisation of algorithms.
- **Appropriation** is the adaptation of algorithms to a specific context.
- **Sharing** is the circulation of algorithms, open to study by others.
- **Distribution** is the packaging and documentation of algorithms.

My use of this conceptual device here is mostly to narrate a possible new flow

for virtuous (and sustainable) creativity in the domain of knowedge economies. I proceed from here by representing a scheme of the flow of value in the corporate industry of computer programming.

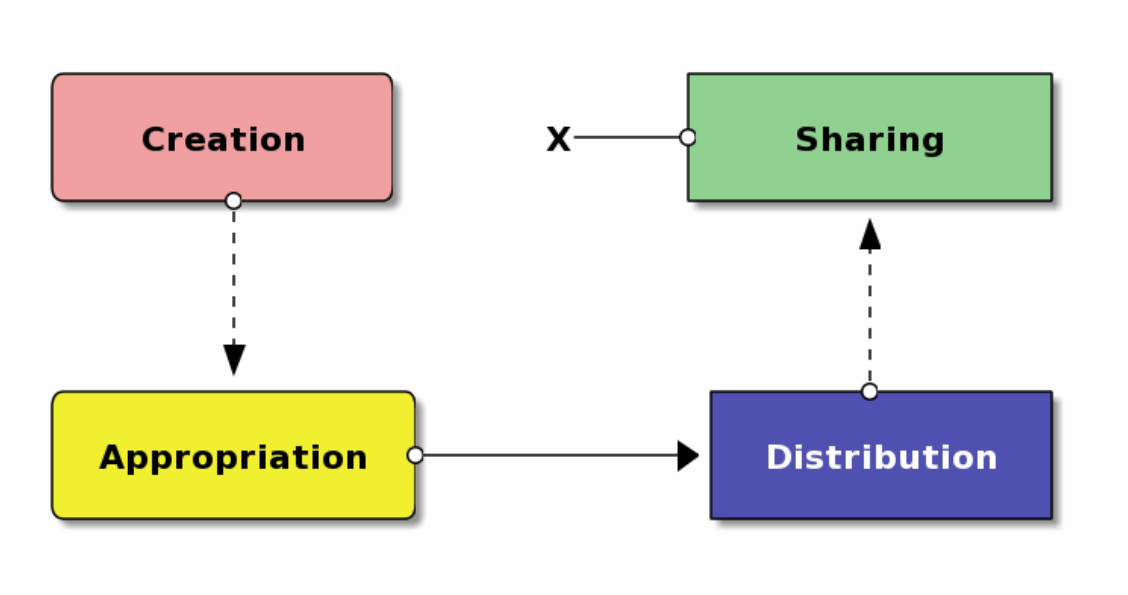

Figure 4: Semiotic square of proprietary creation

This configuration of the semiotic square flow represents how most of the immaterial economy is implemented by modern industry dealing with immaterial production, for instance art, entertainment or software. The *Creation* moment is closely bound to the concept of authorship and is dealt with as a contract for the appropriation and management of such rights. The *Appropriation* is a legal conundrum where rights are reserved for the producer, while different sorts of low-value deals may be in place between the produced and the author. The link between *Creation* and *Appropriation* is not generating value per-se, but consolidating a deal on how the value creation will be distributed.

The passage between *Appropriation* and *Distribution* is where modern industry really generates profits, with packaged products that are distributed to a large audience of consumers, be it via material (CD, DVD, mass storage supports, etc.) or via immaterial channels (online shops and proprietary markets like iTunes, Play, etc.). Further on, from Distribution to Sharing there may be different sorts of agreements on how the acquired materials can be shared. Such agreements may vary in nature and quality of implementation and are often a contested ground since it is in the interest of the industry to regulate heavily, to monitor and sanction the violation of restrictions put in place here. To this link belongs the techno/political debate about DRM[7] (Digital Restrictions Management), the right to copy and share with friends, the potential regulation for clubs of people sharing content, but also the way Public

[7]Controversially called "Digital Rights Management", it is widely considered a threat to the digital rights of users; see drm.info

Libraries function.

Finally, the link between *Sharing* and *Creation* is broken, with some rare exceptions. While it seems natural to presume that creations (and authors) are inspired by the cultural context created when *Sharing*, there is a mutual interest for authors and industry to deny this link and block most "derivative" works, denying the fact that many "original" works are in fact derivatives of works shared in the wild. An example of this is the appropriation of Propp's popular fables collection by Disney industries (Pallant, 2011), or more recent debacles on the "remix culture"[8] and widespread practices in art and entertainment like DJ and VJ (Dekker, 2011) performative sets or sub-cultural phenomena like "Bastard POP" (Shiga, 2007).

This scheme is a simplification of economic relations for the distribution of immaterial goods, I intend to propose it as a viable base for debate and representation of qualitative changes introduced by the Free Culture Movement and F/OSS based economies. To illustrate the latter, here I draw another scheme also based on the same semiotic square:

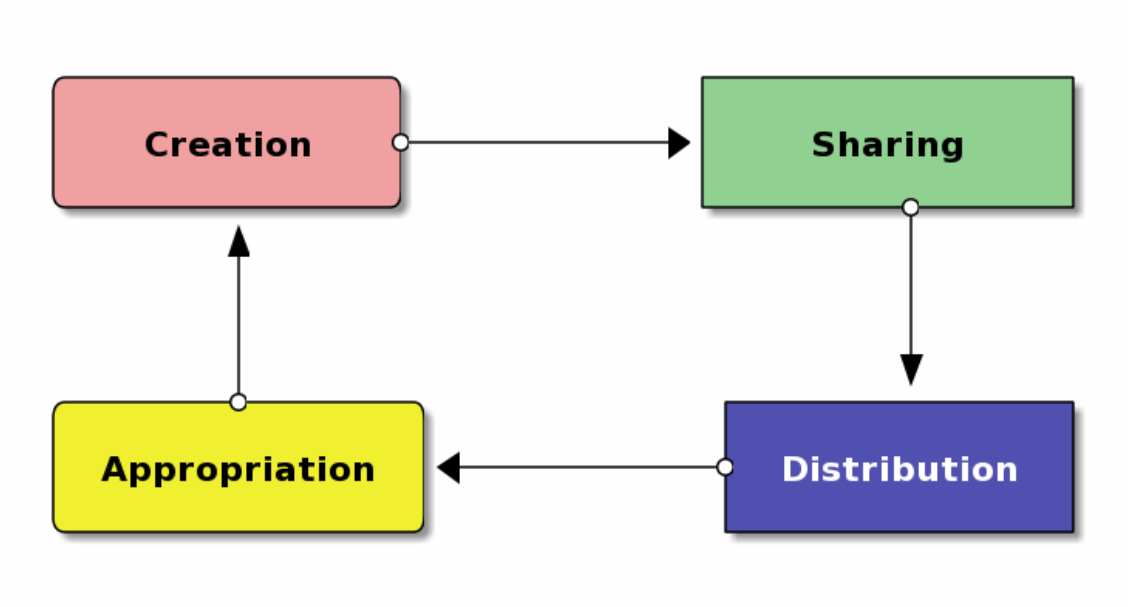

Figure 5: Semiotic square of virtuose creation

The first striking difference I observed here is that the direction is inverted. The circularity suggests that any moment represented can be a starting point for agencies that focus on the process rather than a final result. The ideal starting point is Sharing which, in the semiotic square formulation is the node furthest from Appropriation. Sharing is the space for "moist media" (Ascott, 2001); it represents the space for a multiplicity of media to be configured and manipulated in an infinite number of

[8] Also see the "Remix Culture" festival to which I have contributed with workshops, held at the Netherlands Institute of Media Art in collaboration with the Master of Arts Education, Amsterdam School of the Arts, Stedelijk Museum Amsterdam and FOAM Museum of Photography Amsterdam http://nimk.nl/eng/remix-culture

ways, further in the flow, by distributed authorship, appropriation and distribution.

The Distribution moment here acquires the form of a free and open market where interactions can be structured, traced and categorised: brilliant examples for such distribution models are on-line platforms like Soundcloud for musicians or Github for programmers, whose popularity and sustainability is noticeable despite the apparent lack of a "business model" and even the absence for monetisation through advertisements.

Appropriation thus becomes not a moment for the restriction and management of author rights, but brings us back to the meaning of "Appropriate Technology" (Schumacher, 2011) and includes the contextual and distributed agency for translations, adaptations, customisations and user experience improvements within certain communities. In linguistic terms this could be defined as the acquisition within a "Parole", the concrete instance of use for distributed value. In practical terms this is the declination of values and ideas into usable services, art movements, styles and forms of representation that are easily comprehended and nurture a particular language or cultural context.

Finally, Creation here follows Appropriation and indicates that the moment of authorship is closely bound to that of appropriated education as an intelligible passage of values between a concrete instance of knowledge and the individual. The individual or collective moment of Creation is less isolated from other moments and, rather than representing the enclosing act of immaterial property (often and wrongly referred to as intellectual property (Stallman and others, 1985), it highlights that creation by individual or collectives depends on how vibrant is the context of Sharing and how effective is the Appropriation of value circulation within the language and schemes that form practitioners.

The last closing passage from Creation to Sharing is an economic moment characterised by mechanisms of reputation and curatorial activities intended to facilitate and relate to authors as Sharing is, in fact, crucial to the very existence of authorship; also, in the sense of sharing the place for new authorship within the same or similar creations, a dynamic is evident in art movements as much as in software development.

My intention with this study is mostly to represent the qualitative gap standing between the perception of modern industrial processes in the circulation of immaterial values and emerging forms of "sharing economies" (Sacks, 2011), a definition often misinterpreted yet important to comprehend for a contemporary debate on copyright reform (Towse, 2005).

However, the question posed in this thesis is not limited to such economic analysis, but intends to move further in the domain of governance, sovereignty and integrity for which economy is an underlying structure. In this sense, I took one step further and experimented with the creation of a "folded semantic square", mirroring each

element to reproduce in an outward direction the same relationships defined by Greimas for the economy of his semantic square. The result is the scheme below which concludes this brief study and extends the potential terminology to a more refined representation of dynamics that are internal or contingent to each moment, while referring to terms found across the text of this thesis:

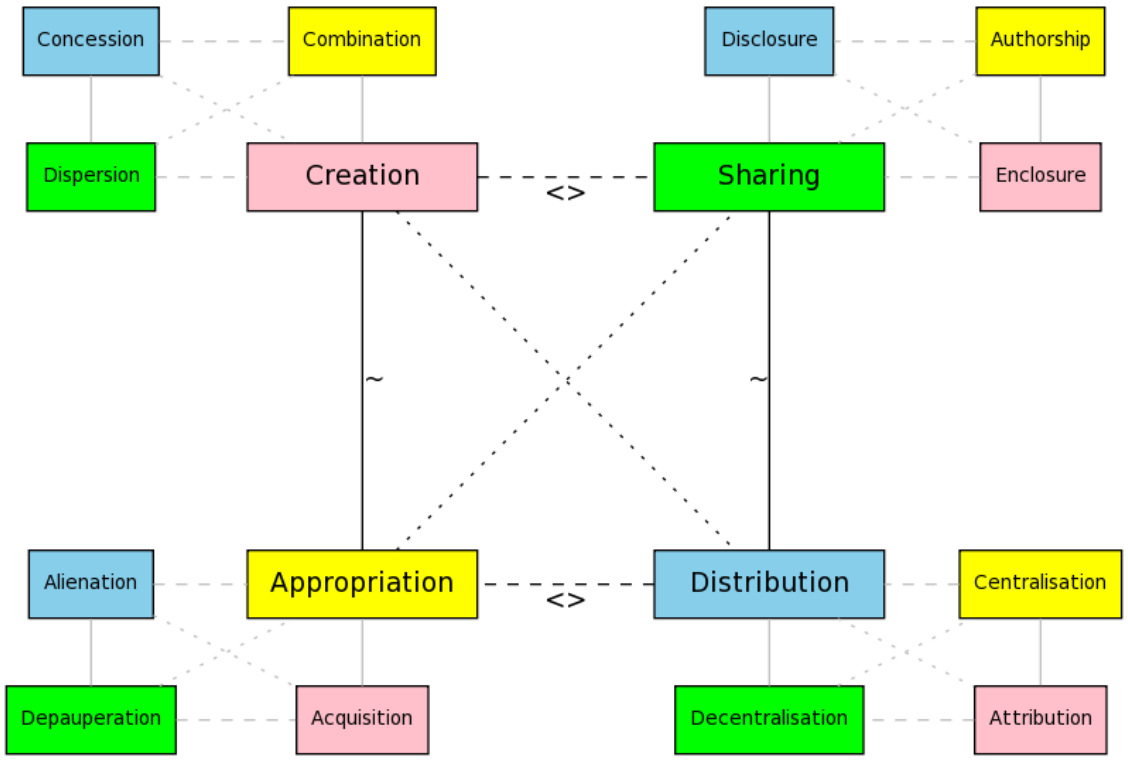

Figure 6: Folded semiotic square of creation

This study serves to lay down a terminology and general context for the description of open source economies. It responds to the first intention of this thesis to focus on the "generative patterns" for the open source economies that have marked a relevant passage in the modes of production of the past 20 years. While getting us closer to the question posed with this thesis, my first supervisor, Antonio Caronia, and I agree that this does not completely satisfy the need for a context. Free and open source economies are praised for their implementation of the principles of "wealth of networks" (Benkler, 2007) and this model provides a limited idiolect for its dynamics. Yet there are critical aspects left behind by such overly-optimistic analysis which clearly emerge from adopting a political economic analysis that puts labour at the centre of the debate (Terranova, 2000). The consideration of labour dynamics and appropriation and subsequent enclosure by the industry of the F/OSS model inside the proprietary modes of production is a reality that cannot be ignored. Therefore, to complete this context I moved forward to include this conflictual aspect, a final step leading to the definition of the theoretical context posed.

1.7 Labour in the digital dimension

A way to describe the dynamics of F/OSS in the political economy debate can be to relate it to the almost parallel discourse on "crowdsourcing economies". The economies commonly referred by this term are different from F/OSS, but there are analogies in the way they include and involve people in the processes of production. Crowdsourcing has been heavily enabled by web technologies to achieve business goals and adopted commercially as a service model called "Mechanical Turks" by the multi-national company Amazon.

The term crowdsourcing indicates the act of outsourcing tasks, traditionally performed by an employee or contractor, to a large group of people, a crowd. It is characterised by the trend of leveraging mass collaboration on repetitive tasks, often using a "gamification" approach to them, even creating reward systems that allow such economies to provide incentives to participation. According to Howe: "Crowdsourcing constituted a new form of corporate outsourcing to largely amateur pools of volunteer labor that create content, solve problems, and even do corporate R & D" (Howe, 2006).

The Mechanical Turk was originally a creation of Hungarian nobleman Wolfgang von Kempelen, presented in 1769 in a conjuring show at the imperial court of Maria Theresa, Empress of Austria-Hungary. Von Kempelen presented a mechanic man sitting behind a table: fashioned from wood, powered by clockwork and dressed in a Turkish costume, it was capable of playing chess.

The fascination of this creation is very actual. The pseudo-automation of labour, its alienation from view, is defined by Amazon as "artificial artificial intelligence": the repetition here has significance, it signal the presence of an AI is fake, substituted by invisible humans. This was indeed the secret of Von Kempelen's creation, made uniquely to impress his audience: it hid a human inside it, operating the chessboard through a set of levers, almost suggesting that one day this role would be substituted by a machine. This episode has been recalled through the centuries, echoing in the works of the famous writer, Edgar Allan Poe, in one of his novels where he describes the Mechanical Turk:

> Yet the question of its modus operandi is still undetermined. Nothing has been written on this topic which can be considered as decisive - and accordingly we find every where men of mechanical genius, of great general acuteness, and discriminative understanding, who make no scruple in pronouncing the Automaton a pure machine, unconnected with human agency in its movements, and consequently, beyond all comparison, the most astonishing of the inventions of mankind." (Poe, 1836)

Back to our times, at the origin of the modern crowdsourcing phenomenon are so-called ESP games (Ahn, 2006) which were first used to address the problem of

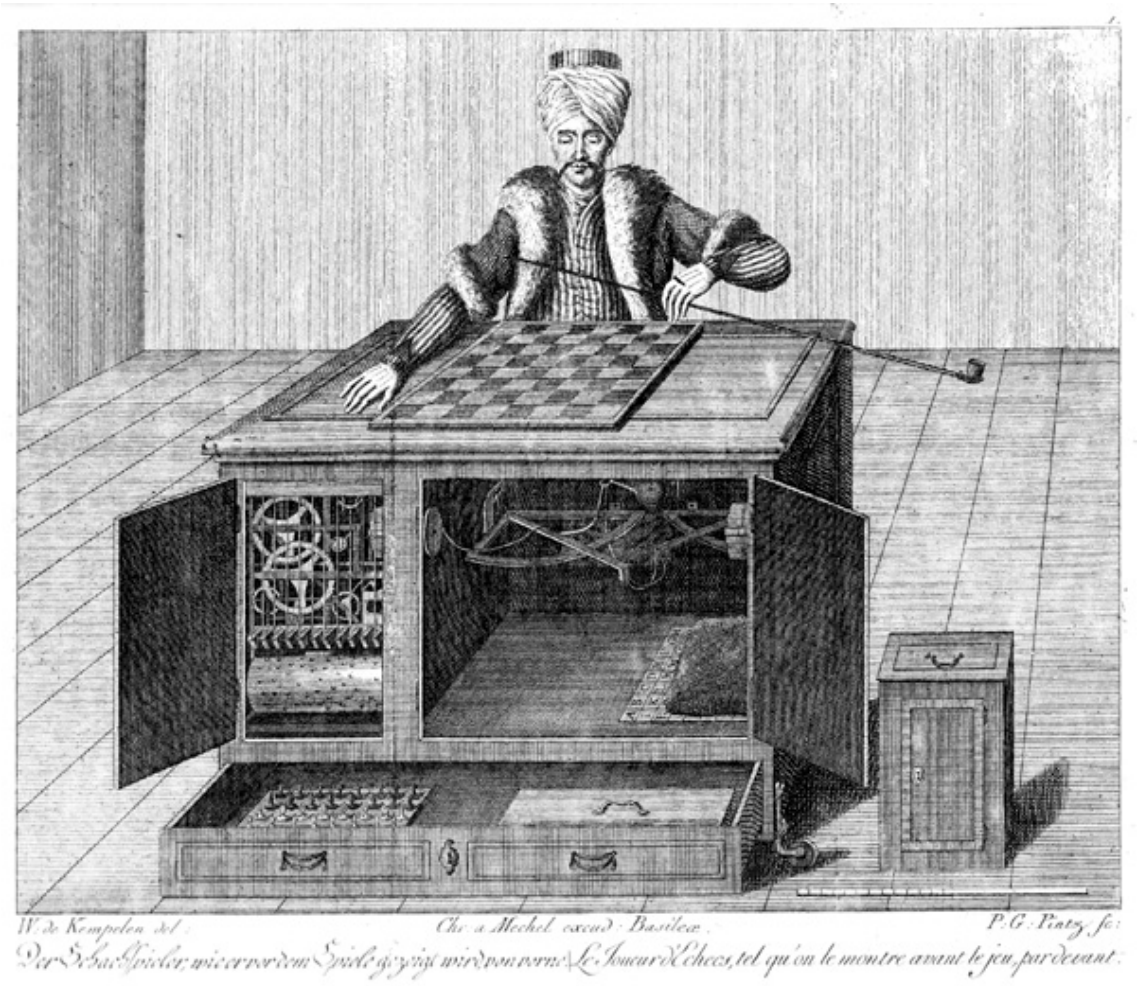

Figure 7: The Mechanical Turk as represented in an ancient illustration, signed P.G. Ponty

creating difficult metadata on non-textual information that is easily recognised and categorised by humans. ESP uses the computational power of humans to perform a task that computers cannot by packaging the task as a "game".

Figure 8: Sample screenshot of an early version of the "ESP game"

> 5000 people simultaneously playing an ESP game on image recognition can label all images on Google in 30 days. Individual games in Yahoo! and MSN average over 5000 players at a time. (Ahn, 2006)

Crowds are gathered by playing games and players invest their intelligence into some problem solving: voice recognition, tagging, optical recognition, describe the tune (tag-a-tune), etc. They are assigned points that are worth nothing outside the game but produce an illusion of an incentive system for it.

With the evolution of such systems, it is now clear that this incentive is provided outside their own function, for instance with the popular Re-captcha system which grants access to content or subscriptions on-line while keeping "non-humans" outside, as spam protection from bots.

Another notable adoption of the crowdsourcing approach is in the electronic industry with "Electronic Design Automation" approaches (EDA) where complex problems can be broken up into modules where the input/output of logic circuits is tested against combinations computed by humans (Romero, 2009). It is interesting

to note that by adopting this approach it is possible to hide the product of the final design completely from workers employed on it, alienating them from the value they produce. This again allows creation of different incentive models that are based on fictitious point systems that are contextual to the "game". Arguably, this alienation could go as far as involving people for the realisation of ethically questionable products they may not approve consciously; for instance, the design process of the ballistic circuitry of weapons could be broken down into logical games to be solved by minors.

The phenomenon of crowdsourcing has inspired a number of artists to produce works as an extended commentary on it. In an exhibition titled "Phantom Captain" (Grover, 2006), we find various artworks that were materially produced by crowdsourcing the artist's goal, raising interesting questions regarding the distribution of authorship, questions ultimately answered by the fact that the "authors" as artists were credited for those productions.

Figure 9: Excerpt from "The Sheep Market" by Aaron Koblin

> A collection of 10,000 sheep made by workers on Amazon's Mechanical Turk. Workers were paid 0.02 ($USD) to "draw a sheep facing to the left." Animations of each sheep's creation may be viewed at TheSheepMarket.com.

> Easy numbered assignments for anyone—artist or non-artist—to complete and upload his or her results (known as "reports").

> Based on the 1977 Museum of Modern Art Artists' Cookbook by Madeleine Conway and Nancy Kirk, and composed of free "recipes" submitted by contemporary artists.

The discourse unfolds within the political economy debate in defining the interesting concept of "Flexitariat" (Da Rimini, 2011), a new category for labour describing how the knowledge workers of precariat (Standing, 2011) accept increasingly flexible work schemes and further alienate from their career portfolios to withstand changes in the labour market. This dynamic is nowadays very clear in the rhetoric of neo-liberist economic notions leading change in many sectors of cultural and material production. Business companies invest more in attracting than in retaining talent and skills: they

Figure 10: Exerpt from “Learning to love you more” by Harrell Fletcher and Miranda July

Figure 11: Exerpt from “Artists’ Cookbook” by Allison Wiese

seek the speed, creativity and volume of large and diverse populations to develop and to consume new formats.

It is possible to draw analogy with the definition of an immigrant worker given by Rancière:

> The emancipation of the workers is not a matter of making labour the founding principle of the new society, but rather of the workers emerging from their minority status and proving that they truly belong to the society, that they truly communicate with all in a common space; that they are not merely creatures of need, of complaint and protest, but creatures of discourse and reason, that they are capable of opposing reason with reason and of giving their action a demonstrative form. [...] The immigrant is first and foremost a worker who has lost his name, a worker who is no longer perceptible as such. Instead of the worker or proletarian who is the object of an acknowledged wrong and a subject who vents his grievance in struggle and disputation, the immigrant appears as at once the perpetrator of an inexpiable wrong and the cause of a problem calling for the round-table treatment. Alternately problematised and hated, the immigrant is caught in a circle, one might even say a spiral: the spiral of lost political otherness. (Rancière, 1995)

The human labour activity serving algorithms is less and less considered labour itself, alienated in the domain of games or social interaction while exploited for its value in the markets. At the same time, the worker is no longer perceptible as such: everyone may fit Rancière's category of an "immigrant" within the sort of agency established by new forms of immaterial and deterritorialised labour.

This issue has been extensively researched and analysed by Trebor Scholz in the framework of contemporary political economics and mostly in relation to the ownership and governance of the platforms that organise the labour and the condition of workers. Scholz puts forward a very important call for the "humanisation of digital labour" in the ever increasing number of societies that are facing the phenomenon, advising to not tolerate exploitation but encourage cooperation, co-ownership and democratic governance (Scholz, 2016). This analysis resonates with many aspects examined in this thesis, yet a distinction among different approaches must be made: when focusing on labour conditions it is easy to represent a line of conflict as present between humans and algorithms, but this is not the sort of conflict that this research tries to describe.

To dive deeper in the techno-political dynamics at play I'll dare to digress into an effort that may be well regarded as unnecessary by political economists: an important distinction must be made between the definition of "algorithms" and the term "platform". A platform is the implementation of an algorithm in conjunction with a particular base of data and participants. In other words, a platform is the deployement of a sourcecode which, when executed, represents the "production means"

by which the workforce is organised. The algorithms on the other hand are the codified implementation that define the mode of operation of a platform and may be adopted to run multiple instances: they are the "blueprints" of the production means. This also defines two different levels of governance: one that operates the platform and one that re/designs the algorithms. Arguably algorithms are not necessarily subjugating humans, to the contrary they may also be used to empower the living world as a whole: yet the difference is not only made by the operations defined by a source code, but also by the conditions for its deployment as a platform. In an effort to relate to the seminal work that Scholz and other researchers have made on the topic of immaterial labour (Lazzarato, 1997) I conclude this digression with the attempt to define the meaning of a condition of sovereignty in this context. For the workers who are exploited by crowd-sourcing platforms, algorithmic sovereignty means to know their own ratings, to influence the algorithms that establish them as well the algorithms that define the negotiation regarding the acceptance and validation of work done. On the other end, the co-ownership of a platform would mean to share the costs and the profits made by that particular instance, administrating the execution of the algorithms that define it and sharing the liability of making the platform available. It can also be envisioned that, in case the "blueprint" for a certain platform is available to the workers (for instance the code open source) then the workers themselves may decide to "re-deploy" it, or better said fork it, into a platform that they own; but then as we will discuss later in this thesis the forking process has a very high price and its success depends from various conditions of resource usage, literacy, licensing, critical mass and level of establishment of the originating platform.

For completeness, another qualitatively different and notable attempt has been made to design a protocol that can quantify digital labour interactions: the User Labor Markup Language (Arikan and Erdogan, 2008) as an "open data structure (based on XML syntax) to outline the metrics of user participation in social web services". The ULML effort aimed to construct criteria and context for determining the value of user labour (Ridgway, 2014), to constitute a monetised asset not just for the service provider but also for the user him- or herself, something that has only been realised by Amazon in the context of its Mechanical Turk service.

Interestingly, the effort of ULML never gained momentum and was substituted by the Activity Streams (AS) standard, adopted by Facebook, MySpace, Windows Live, Google Buzz, BBC, Opera, TypePad, Gowalla, Gnip, Superfeedr, YIID, and many others. The AS protocol is mostly focused on the semantic organisation of events occurring, but hardly contemplating the value attached to human interaction.

Going back to the actual research question then, lets think of algorithms and ask who else is in charge for the definition of labour? Who has an overview and validates work? who knows what the work is for?

The questions here need to be deeply reformulated to address a larger picture, yet

Figure 12: Self-portraits of M-Turk workers have been commissioned as a paid task

here we have found two very important elements emerging: the role of point-based systems or gaming incentives and the role of algorithms in defining the mode of production, access, perception and reproducibility of labour.

1.8 Literature and algorithms

The main subject for this study is literature, algorithms and the passion and knowledge necessary to engage writing and maintaining them. The question this thesis intends to answer is articulated by the concepts of algorithms and sovereignty. The answer will emerge from the practical "in-vivo" observation of important roles such concepts play for the correct interpretation of highly influential projects.

As the demand and production of well-connected vessels for the digital dimension has boomed, machinic code today functions as a literature informing the architecture in which human interaction happens. The telematic condition is realised by an integrated datawork continuously engaging the observer as a participant. Such a "Gesamtdatenwerk" (Ascott, 1990) may seem an abstract architecture, yet it can be deeply binding under legal, ethical and moral circumstances.

My quest in this research has been to individuate and understand the literature of algorithms, an interdisciplinary zone which is perhaps best understood by artists and architects and political scientists, shaping architectures of interaction between humans, machines and the natural environment in contemporary times.

Leaving aside quantitative indicators based on human interaction and profits, going beyond purely axiomatic analysis of systems and their throughput, can we

learn to recognise the quality of such a literature? Is it possible to establish the ground for a shared language that informs digital architects about their choices and inhabitants about the digital territory? Going past assumptions about the strong role algorithms have in governance and accountability (Diakopoulos, 2016), how can we inform digital citizens about their condition?

When describing the virtualisation of economic activity in the global context, Saskia Sassen describes the need we are observing as that of an analytical vocabulary:

> The third component in the new geography of power is the growing importance of electronic space. There is much to be said on this issue. Here, I can isolate one particular matter: the distinctive challenge that the virtualization of a growing number of economic activities presents not only to the existing state regulatory apparatus, but also to private-sector institutions increasingly dependent on the new technologies. Taken to its extreme, this may signal a control crisis in the making, one for which we lack an analytical vocabulary. (Sassen, 1996)

What are the parameters for a qualitative evaluation of the way private, public and common spaces are governed by algorithms? The analysis of legal texts and regulations here shifts into an entirely new domain; it has to refer to conditions that only algorithms can help build or destroy. Thus, referring to the initial part of this theoretical framework, the research on open source economies, such configurations for sovereignty are established through the process of *Creation*, *Appropriation*, *Sharing* and *Distribution* of algorithms. The only assumption made in starting this exploration is now that such an observation of a live, complex and influential literature can only start from practice.

2 The Bitcoin experience

Table 3: Information on the research about the Bitcoin project

Project name	Bitcoin
Home online	https://bitcoin.it
Started in	2010
Nature	Research
Advancement	Context

Late during the year 2010 and following up some more superficial interaction with the early developers of the Bitcoin project, I've started getting more involved with its community and its development process. My participation in Bitcoin has been mostly tangential and my direct code contributions rather limited, but I've taken care to study in-depth its code since its early releases, advise some of its core developers and most importantly follow the interaction and decision making process behind its development choices.

Considering what Bitcoin became later, the size and importance it grew to, this has been a very interesting experience, providing enough research materials to move well conscious steps in the context of the D-CENT, described in a later chapter of this thesis. After a few code contributions and reviews, my agency in the Bitcoin project has mostly focused on assessing the solidity of its cryptographic implementation (Roio et al., 2015), as well advocating for reliable improvements, especially to solve the malleability issue with the segwit implementation (Sward et al., 2017). Most importantly I've helped through the years to consolidate and document the OP_RETURN protocol implementation, a feature that leaves room for experimentation on Bitcoin without flattening its function to the mere financial and speculative use that is made of what I consider to be the biggest distributed trustless network in the world.

My intention here is not that of concentrating too much on the Bitcoin project, but to highlight some often overlooked historical and socio-political aspects in the history of this embarassingly popular project: aspects I've documented and experienced in first person and are relevant to complete my research on algorithms and sovereignty of this thesis.

I will start from the basic assumption that Bitcoin, first and foremost, constitutes an incredibly popular proof of the possibility to establish sovereignty by the means of an algorithm. The opportunity offered by Bitcoin for the emergence and sustainability of new constituency has had an enormous political impact on projects which were excluded by the mainstream financial system and, as I will argue, it started growing

popular exactly because of this aspect. Seen in these terms, this project shows how a passionate yet transversal critical mass, an idea of algorithmic neutrality and the strong demand for social justice can all coalesce in a Καιρος for a multitude (Hardt and Negri, 2005) that unfolds as a very generative and ungovernable mass phenomenon. It is however not given for granted that the project can stay faithful to its initial premises, especially when developed in the highly speculative and manipulated context of algorithmic finance, yet its political underpinning are remarkable as well the strong ties that have been kept to them through all this time.

To trace the element of sovereignty in the Bitcoin project is a basic assumption since the act of minting an authenticated medium for value circulation is indeed the most iconic and basic sign of sovereignty in history, yet not the most important one I'd argue. Bitcoin became so incredibly, virally popular on the wave of a "violation of neutrality" operated at the time another controversial project was coming into international attention: Wikileaks. Briefly: at the end of 2011 Wikileaks was in trouble because of payment processors Visa, Mastercard and Paypal shut down its accounts for receiving donations. Wikileaks was then in the middle of an increase of attention for its "Cable Gates" release[9], yet the oligopoly of payment processors acted unilaterally and without any legal mandate against one of their customers.

This episode is documented in a declaration released by Wikileaks at that time:

> Since 7th December 2010 an arbitrary and unlawful financial blockade has been imposed by Bank of America, VISA, MasterCard, PayPal and Western Union. The attack has destroyed 95% of our revenue. [...] The blockade is outside of any accountable, public process. It is without democratic oversight or transparency. The US government itself found that there were no lawful grounds to add WikiLeaks to a US financial blockade. [...] The UN High Commissioner for Human Rights has openly criticized the financial blockade against WikiLeaks. [...] The blockade erects a wall between us and our supporters, preventing them from affiliating with and defending the cause of their choice. It violates the competition laws and trade practice legislation of numerous states. It arbitrarily singles out an organization that has not committed any illegal act in any country and cuts it off from its financial lifeline in every country. [...]
>
> In the US, our publishing is protected by the First Amendment, as has been repeatedly demonstrated by a wide variety of respected legal experts on the US Constitution. In January 2011 the U.S. Secretary of the Treasury, Timothy C. Geithner, announced that there were no grounds to blacklist WikiLeaks. There are no judgements, or even charges, against WikiLeaks

[9]The "Cablegates" are a list of notable content from the United States diplomatic cables leak that shows the United States' opinion of related affairs. Beginning on 28 November 2010, WikiLeaks had been publishing classified documents of detailed correspondence—diplomatic cables—between the United States Department of State and its diplomatic missions around the world. On 1 September 2011, it released all of the Cablegate documents in its possession without redaction.

or its staff anywhere in the world. (Wikileaks, 2011)

This episode has broken many people's expectations for the neutrality of payment networks (Turenhout, 2012), while Bitcoin seems to satisfy most of these expectations by virtue of its algorithm (Weber, 2014). In 2011 Wikileaks started asking its donors to use Bitcoin in order to wire donations to it and the huge increase in popularity of Bitcoin started from that episode, as shown by figure 13.

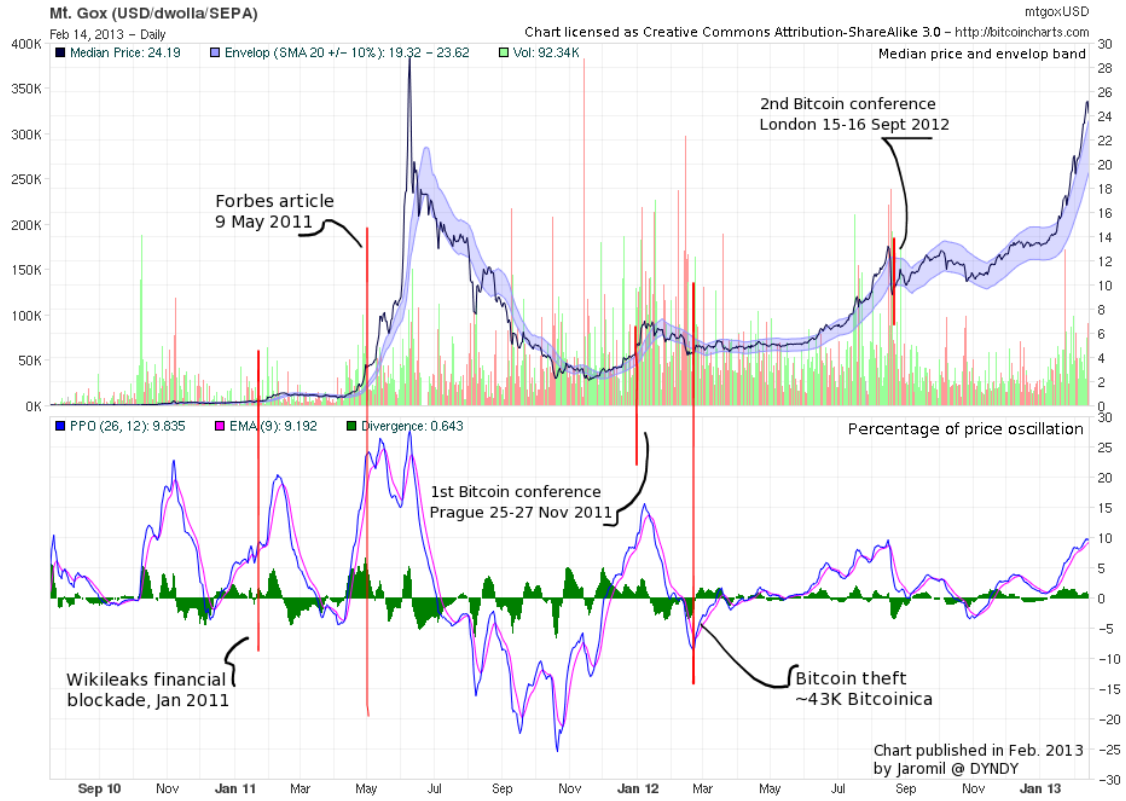

Figure 13: Price oscillation and volume transaction chart for Bitcoin between September 2011 and 2013

While engaging with the project in the past 6 years I've published two slanted, popular texts which have been widely circulated. While I've met the approval of most members of the community at describing what was happening, my approach has not been just that of commenting on facts, nor that of ventilating my own opinions. Instead I've challenged my opinions with a semi-public interaction on different fora[10] and then published a more public account on what I thought was a good interpretation of most views. The response to such public accounts is even more interesting that the process leading to them.

The first is an article published in April 2011 to comment on the first presentation of Bitcoin to the financial sector, which I've witnessed in person, made by core developer Amir Taaki also known as Genjix. Taaki is arguably the most interesting figure of the Bitcoin community, not just from a technical standpoint, but from a

[10]For instance on the original bitcointalk forum and on the nettime.org mailinglist, here what later became the "Bitcoin Manifesto" for the earlier community https://bitcointalk.org/index.php?topic=5671.0

social perspective: he is a passionate social justice activist, provocative, opinionated and very knowledgeable. The article I've published about this episode is publicly available on-line with the title Bitcoin presented to the Old-World.

The second is an essay named "Bitcoin, the End of the Taboo on Money" inspired by even more conversations that included also worldwide renown currency experts like the German architect Margrit Kennedy (Kennedy, 2016) or the Belgian economist Bernard Lietaer (Lietaer, 2001). In this article I left behind most topics connected to finance, to explore the socio-political dynamics of Bitcoin, but also to give an account on its "liturgy", the apparatus of glory behind its sovereign(Agamben, 2011), the folklore that bond its original community, a symbology that I have challenged with various interactions before.

The full article is published in the participatory research publication "Autopsy of an Island Currency" in Finland(Nold et al., 2014) and translated in Italian by the Effimera collective for the publication in the book "Moneta del Comune"(Braga and Fumagalli, 2015). Here below an important excerpt that focuses on aspects well beyond the most debated financial and technical considerations on Bitcoin.

2.1 The glory of algorithms

> Glory, in theology as much as in politics, is what takes the place of the inconceivable void that is the idleness of power; nevertheless, is this very inconceivable emptyness that nourishes and feeds the power (or, better said, what the apparatus of power transforms in nourishment) (Agamben, 2011)

Every form of currency, since the very beginning of its earliest forms, has dealt with the grammar of power. It is the establishment of a sovereign and its glory that justifies the shared trust into a symbolic form of value circulation. The investment of power into currency, especially when its not backed by mineral values, is codified in mystery and glory.

Bitcoin is not exempted from such dynamics: it innovates the way the digital becomes tangible, a role with highly disruptive potential. Hence, even when choosing the iconography for its own currency, the Bitcoin community shows a political rupture.

The intriguing mystery of the identity of its disappearing author Satoshi Nakamoto, might seem a detail, but not for our analysis: it is of central importance to the Bitcoin myth and that of future crypto-currencies. Bitcoin has no single monetary authority, but a shared pact and the underlying rationality of a mathematical algorithm - the intangible dream of neutrality. Being deflationary, Bitcoins exist within a finite range of possibilities, a quantity of value that is increasingly difficult to mine. No one can create more Bitcoins than those established to be created in the first place, to the

great horror of modern economists that regard fiat currency as a necessary tool to move within the troubled waters of contemporaneity, with good reason indeed. But there is no hierarchy in Bitcoin: meaning literally that there is no sacred origin, no written fate, no single ruler, no second thought on its essence.

Bitcoin promises to be the neutral medium for an economy based on participation, not the edict of a king, a central bank, or their authorized intermediaries - nevertheless, it must be said, Bitcoin did create new riches, those who believed earlier than others in the promise of this algorithm. The rupture offered by this new perspective on money is not dealing with equality or welfare, it might not benefit society or help us get out of the crisis: it is a protest for network neutrality.

Such a medium, we must also admit, will likely incarnate the market freedom of the Austrian school of economics. The European Central Bank has produced an analysis of the Bitcoin scheme in October 2012 reciting:

> The theoretical roots of Bitcoin can be found in the Austrian school of economics and its criticism of the current fiat money system and interventions undertaken by governments and other agencies, which, in their view, result in exacerbated business cycles and massive inflation.

This insight should be handled carefully: it might overstate on the ambitions of Bitcoin, which first and foremost is a successful implementation of a system for value transactions in the digital domain, whose success is due to the biopolitical dynamics we are exploring in this Chapter. Nevertheless, the interpretation of its ethos in fieri is not far from reality. It is paradoxical how, in a time in which we face the failure of most Austrian economic theories, we are confronted with narratives that mystify and popularize them on the wave of technical innovation and functional transformation. But this is a reductionist way to describe Bitcoin and it strictly depends from the adoption of universal categories: I am convinced such a method of analysis can't lead the quest for comprehension we are engaging here. So lets take a step back from this dead end and look into Bitcoin's symbology.

If we look back in the history of icons used to mint money, we'll find a long stream of symbols of leadership: heads or bodies of humans or animals that address or signify the power of scientists, rulers, educators, judges or that of a nation-state. Many are the symbols of hierarchy that govern the minting and authentication of the currency, as well symbols of wealth and geographical maps. I'll refrain now from engaging an analysis of such symbols used in the past, but observe that Bitcoin has and will have a different symbology to glorify it.

The iconography of Bitcoin reflects the shared values of the community behind it. If there would be a person representative of it, this would be its mysterious creator Satoshi Nakamoto, but the fact that he doesn't really exist makes things much more interesting. One of the early symbols of Bitcoin was alpaca, for instance the mockup

Figure 14: The Alpaca Coin, an early symbol in Bitcoin, still present in the works of Satoshigallery.com

presented here comes from an old forum's thread and in its own way it is meant to celebrate the first artisans that ever sold their creations on the Bitcoin market.

As an experiment, in a previous article for the Bitcoin community I've suggested the use of the empty throne as a bridge symbol across classical, modern and post-human iconography. The image of an empty prepared throne (ἑτοιμασίᾳ τοῦ θρόνου) is an icon found in the Old Testament and in books comprising the Upanishad, a sacred icon whose value "..is never so powerful as when the throne is empty" (Trell, 1988). The empty throne was used on minted currency in the Augustan era and sculpted exemplars of it are found in Knossos and Rome.

But the response of the Bitcoin community to such an old symbol of power, despite the fact it could represent the absence of Satoshi Nakamoto, has been negative. Someone commented that "perhaps a broken empty throne would be even better, symbolizing the breaking of the old power", someone else suggested that "a physical Bitcoin should have a mirror in the middle. Bitcoin is all about the individual" and again another suggestion "Bitcoin is mercurial – it's quicksilver. It's the fool of the tarot and a touchstone. It turns base electrons into gold. It subverts and debases all norms and conventions. The fool is the perfect symbol for bitcoin". Many also acclaimed the use of the Guy Fawkes mask, already adopted by Anonymous, from the V for Vendetta comics and movie.

The glory behind Bitcoin is mostly shrouded in mystery, revolt against tyrannical injustice, the reclamation of individual rights, power distribution and the disintermediation and self-determination. But also, I strongly argue, by the transverse presence of a community feeling and the joyous consciousness that a powerful process is unfolding in history: those participating have the possibility to express themselves in their diversity, rather than the uniformed, sterile and omnipresent corporate language of economics.

> After the phase in which the Multitude has built its body inside the language, the next opening cycle of conflicts will see the Multitude engaged in the construction of its body beyond language. (Marazzi, 2000)

2.2 The activist sacrifice

In more recent events and keeping my interaction with Taaki and other more anonymous members of the community, I believe it is even more evident the proof that Bitcoin was not a project born to satisfy the needs of banks or the dreams of techno-utopians. This is somehow controversial, following the enormous hype created around the project and the polarisation of mainstream media.

Following anthropological values (Graeber, 2001) that are deeply ingrained in the Bitcoin project, Taaki spent about two years in Rojava, Kobanî, at the northern border of Siria, surrounded by war to resist the advancement of ISIS (Federici, 2015). During this time we kept in contact, with him and some colleagues, about the projects they were working on there, also in cooperation with the University of Kobanî and its municipality.

On 9 January 2015 Taaki wrote me his intention to go:

> I've been preparing for last few months. the kurds are good people. I'm really happy to help them. I hope I don't get captured by isis. but I must help them. I cannot sit by. I'm very useful. I know they need my skills, despite what others might imply to me. I can save lives defending a good cause. I think caedes is a little upset I'm abandoning him, but the project is ready. There's plenty of opportunity for him to continue. I hope in time he can realise the magnitude of what's happening (and that rojava at becomes a success and isn't betrayed by the KRG). maybe you can visit him sometime? everyday I keep searching for more info, I'm very excited by the things they're trying. it's also an experience, but it's not like I'm jumping to the next cool thing. I feel a genuine sense of duty for a revolutionary cause. It's maybe a bit hard for caedes to comprehend since we're doing a lot together already and I'm jumping ship but it's also an opportunity for us all too, that I make friends in this new anarchist republic. I must do it or it will keep tormenting my mind. I find it difficult to work because I want to be gone. if I die, well I will die someday. better to die once standing than a thousand times on your knees.

> if this works, then we can bring peace to the middle east. this is the only way I see for that to happen. and a new tangible example of how an anarchist society looks like when people say it never existed. implant something new and wonderful in people's minds to look forward and strive towards for how we work together. and I need to ensure the success of this political experiment, to make it work because I think the ideals are beautiful but I'm also scared of treachery from within rather than outside threats.

Taaki's interest was not in war, far from being a "foreign-fighter", a popular media stereotype on these matters. He was also helping liaising with press, but never disclosing his real identity even after having been a televised Bitcoin expert on BBC and Al-Jazeera. While in Kobanî he was often lamenting that journalists there were only interested in covering airstrikes and armed fights. On 12 August 2015 he wrote me airing his frustration:

> Man some journalists from NY times come here [...] Come to do a report on US airstrikes and danger of IS. I tried to explain to them that there's a revolution here with women's rights, direct democracy and cooperatives but they couldn't even understand. The women asked me "what's a cooperative?". How can you be so clueless about a place you're visiting, they didn't even read the wikipedia article or what. total disinterest.

Taaki was mostly dedicating his time there, about a year and a half, to the setup of a media-center for ICT education and independent journalism within the premises of the University of Kobanî. After coming back and still at the time of writing, Taaki is held at house arrests waiting for the British police to declare his imputations, a special condition allowed by anti-terrorist laws which apparently apply to any British resident returning from Kobanî.

I'll conclude here this rather eccentric excursus on the Bitcoin saga, conscious that the facts I've reported in this thesis are not documented anywhere else, while worth to be acknowledged as they provide good hints on the contexts where algorithmic sovereignty can be individuated and recognised in the ethos of programmers. In fact, this sort of genealogy for a techno-political consciousness goes beyond the importance of the Bitcoin project itself, while risking to be completely obscured by the overwhelming presence of a literature that, to quote Nathaniel Popper defending such omissions in his popular romance about Bitcoin, "tells the story of the winners".

2.3 The utopia of neutrality

This historical account serves in fact to clear how in different contexts and ways the Bitcoin experiment is born out of needs, ideas and practices that wanted to and in some degrees de-facto have established a new sovereignty based on the existance of an algorithm and of a community that knows it well and considers it reliable.

Both the ethos of some the activists behind it, as well the commercial interest of some of the big industrial and financial players involved, relate to the possiblity of sovereignty offered by the project and its independence from currently existing state jurisdictions. In relation to industrial interests it must be noted that the Bitcoin sword is double-edged, as it can also be envisioned to bypass existing sovereignty boundaries and conceal relations established between or outside of them. The most accellerated financial uses of Bitcoin and related technologies indeed look into that

direction, but its debatable if it will ever be really useful to financial industries.

3 Decentralized Citizen Engagement Technologies

Table 4: Information about the D-CENT project

Project name	D-CENT
Home online	https://dcentproject.eu
Started in	2013
Nature	Research and development
Advancement	Methodology
Media footage	https://www.youtube.com/watch?v=bK2QdViY5PU
Source code	https://github.com/d-cent
Secrets (demo)	https://secrets.dyne.org

In 2012, together with colleagues of various European organisation in a process lead by Dr. Francesca Bria (NESTA UK) we have converged in writing down past experiences and future plans for a project called "Decentralised Citizen Engagement Technologies" (D-CENT). The project drew on ideas and assumptions about new possibilities of political engagement for citizens offered by algorithms and their use for communication, deliberation and bottom-up organisation. In particular, with D-CENT we focused on new emergent constituencies in Europe which had plans to institutionalise (DiMaggio, 1988) their agency - or partially already did.

D-CENT focused on three concrete pilots across Europe:

- the Pirate Party and related movements in Iceland
- the Open Data movement working in Finland with close relations to its Ministry of Justice and city of Helsinki
- the Podemos party and related grass-root movement 15M in Spain, following its progression from the streets to the institutions

The D-CENT project was funded by the European Commission (FP7/CAPS grant nr. 610349) and ran for 32 months involving 9 different partner organisations:

- The National Endowment for Science, Technology and Arts in UK (NESTA)
- The centre for economic studies of the Sorbonne University (CES)
- The Open University and Technology Centre of Catalonia (UOC / EURECAT)
- The Icelandic Modern Media Initiative and the Citizen Foundation in Reykjavik (IMMI)
- Forum Virium, the innovation company of the City of Helsinki (FVH)
- The Open Knowledge Foundation advocating open-data standards (OKF)
- The software company Thoughtworks, international excellence in Agile methodology (TW)

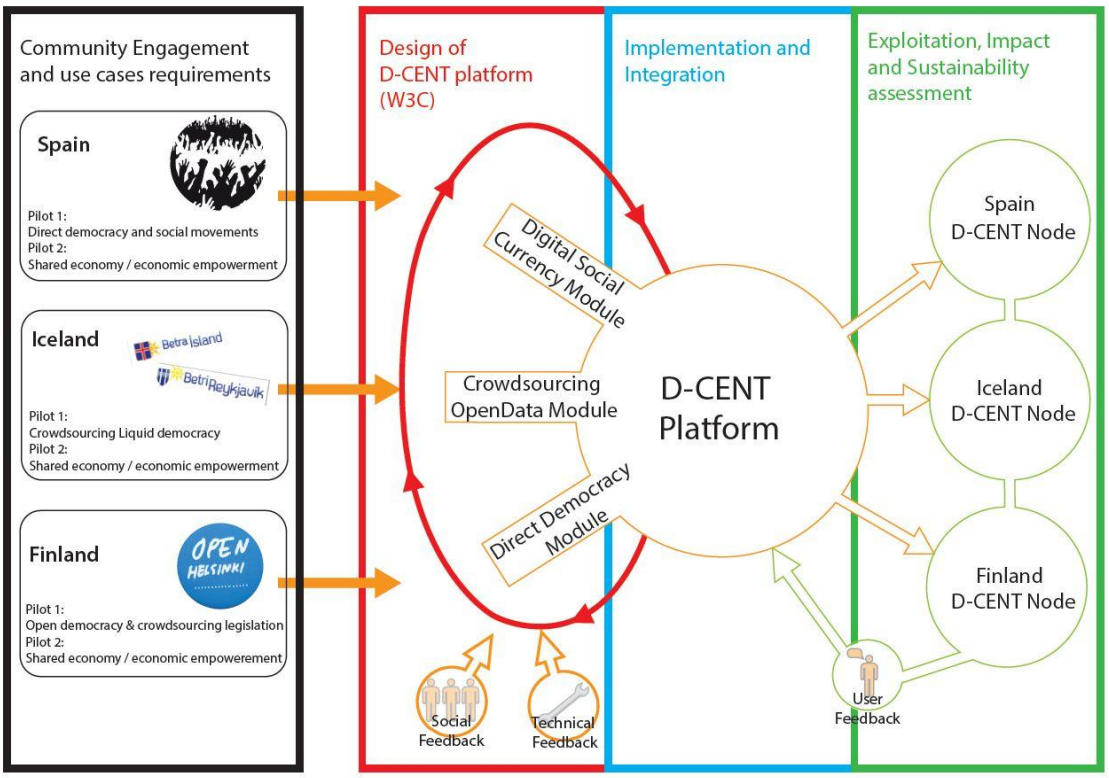

Figure 15: D-CENT pilots

- The World Wide Web consortium for open standards (W3C)
- The Dyne.org foundation, Think/do tank and software development organisation (DYNE)

Here below a quick overview of the work-package and task distribution across the project:

- WP1 - Methodology, use cases requirements and impact assessment
  - T1.1 User-centric lean methodology
  - T1.2 Communities requirements and social design
  - T1.3 Indicators, impact assessment and sustainability models
- WP2 - Network driven data analysis, modelling and visualisation
  - T2.1 Socio-technical framework on mobile collective action to solve societal challenges
  - T2.2 Data Analysis and data visualisation of citizens movements' networks
  - T2.3 Data modelling for collective awareness and self-governance
- WP3 - Economic analysis on new Commons and sustainable Economic cultures
  - T3.1 Theoretical framework on future knowledge-based economy
  - T3.2 Managing the commons in the context of the knowledge economy
  - T3.3 Producing the subject analysis of machine-mediated identity systems
  - T3.4 Action research on alternative economic cultures and currencies
- WP4 - Design of platform architecture and pilots
  - T4.1 State of the art of social network systems, identity ecosystem and

social data stores
  - T4.2 Technical requirements and specifications based on users' requirements
  - T4.3 Pilot1 Design of open social web for crowdsourced democracy
  - T4.4 Pilot2 Design of social digital currency
- WP5 - Lean implementation of open social wed for crowdsourced democracy
  - T5.1 Pilot1 Sociotechnical implementation of open social web for crowdsourced democracy
  - T5.2 Pilot2 Sociotechnical implementation of social data currencies
  - T5.3 Technical interoperability and interaction amongst pilots
- WP6 - Promotion and Dissemination
  - T6.1 Dissemination, Exploitation and Standardisation
  - T6.2.1 Develop online engagement tools
  - T6.2.2 Civil society and developer community engagement
  - T6.3 Events and outreach
- WP7 - Project Management
  - T7.1 Administrative and financial management
  - T7.2 Quality management, coordination and reporting
  - T7.3 Consortium and review meetings

As already visible in this overview, D-CENT focused on software instruments for participatory democracy, which is a somehow more established field of research. In addition to that it also included a research on digital blockchain systems and complementary currency. This was the part of the research that involved my organisation (DYNE) the most, together with my colleague Marco Sachy who took care of the economic analysis, and among other organisations: TW for joint analysis and development of the software platforms and CES to assess the economic analysis conducted on pilots as well on the theoretical models on sustainability, knowledge economies and Commons based policy strategies.

What makes D-CENT extremely relevant for this research is not only its direct pertinence to the topic of technologies adopted by emergent political scenarios and bottom-up organisations (and institutionalisation). While this is an interesting aspect in D-CENT, other projects engaged in this research may provide a clearer look into the very topic of algorithmic sovereignty. As ambitious as it is, D-CENT has produced rather complex analyses on many software products, mostly focusing on aspects bound to their execution (usability, use cases, range of adoption, standard compliance and modularity) rather than on the algorithms and on the languages they are written in.

3.1 Research across fields and disciplines

A very interesting aspect of D-CENT for this research is the methodology we adopted, especially in the research conducted between CES, DYNE and TW. While the starting point for the software design process in D-CENT is the LEAN UX methodology (Liikkanen et al., 2014), such a methodology cannot cover all of the steps necessary to build an interdisciplinary bridge across economic analysis and software prototyping.

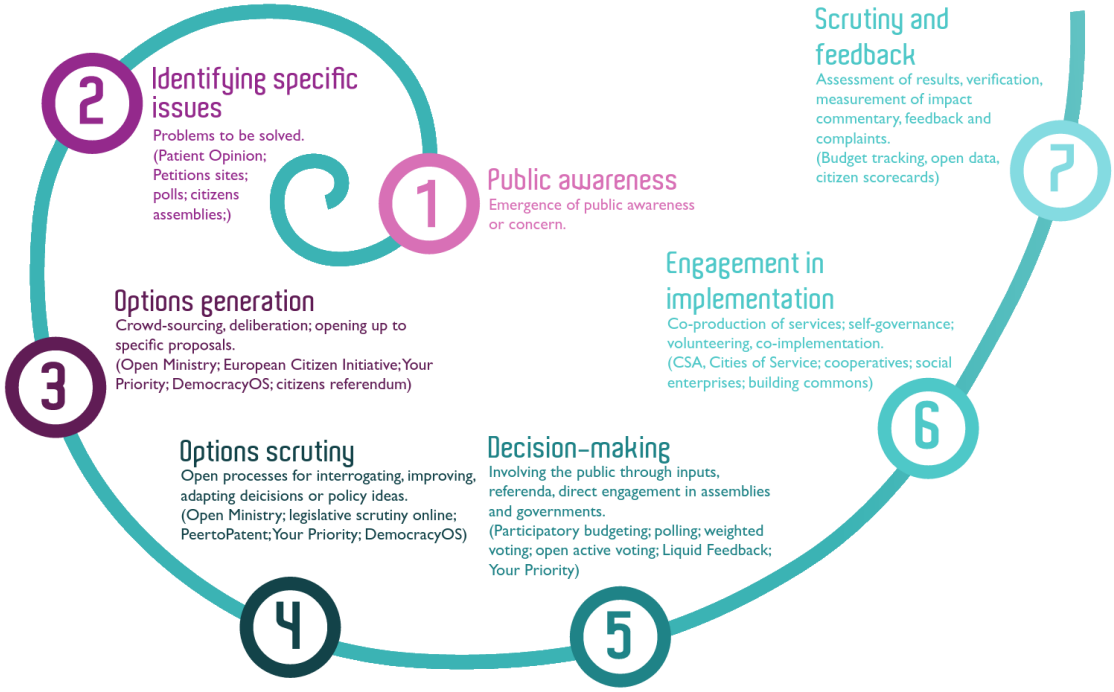

Figure 16: Spiral representation of LEAN UX methodology in D-CENT

The LEAN UX methodology mostly works by quick inception, lowering the latency between prototyping and testing and arguably lowering the risks attached to wrong assumptions carried far into the implementation. By running interviews and quick tests on potential users, one proceeds then abstracting "Personas" to sample the most common characteristics and typologies found. Scenarios and "user journeys" are then drawn to inform programmers on the work that needs to be done both in terms of interface and back-end engine (Guo, 2016).

The advantages for adopting such a methodology are clear especially from the points of view of productivity and risk-management, as a sort of contemporary "just-in-time" mode of production (Swamidass, 2000) extended to the digital dimension. Seen as a design practice and from my empirical observation of the work done in a large company as TW, it is also possible to drop reserves on the quality of results and of labor conditions (Valeyre, 2004) since the LEAN UX is mostly applied to narrow down design challenges rather than fine-tune the volume of production. Results are often lean - in the literal sense of the term - maintainable and elegant, with a higher chance of matching initial commitments and enriching them with real-world insights.

The short-coming of the LEAN UX methodology in the context of an interdisciplinary project like D-CENT becomes clear when interacting with an articulated body of theory that is often too abstract to be related to concrete use-cases and personas. In our case the economic analysis conducted by CES, while establishing an excellent framework with deliverable 3.1 "Theoretical Framework on future knowledge-based economy" (Vercellone et al., 2014) and D3.2 "Managing the commons in the knowledge economy" (Vercellone et al., 2015) was difficult to conjugate with the extremely pragmatic approach of a software company like TW focused on productivity of agile processes. Both LEAN and Agile approaches are extremely pragmatic and process focused, while an academic research coming from the field of macro-economics and policy analysis can be extremely abstract. On such premises, the ultimate goal of producing a tool to solve the problems analysed can be very far from understanding the theoretical underpinning of its existence. What is most troubling then in such contexts is the emphasis on technology: the creeping assumption that a particular technology - in our case the "blockchain" technology - will be the solution, because it already worked as an impacting innovation on large industrial scale. One can well argue that the emphasis itself on technology as a solution is troubling, as it often goes into the direction of reductionism (Farrel et al., 2012), yet the ambition of D-CENT was since the beginning to produce not just research, but technical tools to solve the problems identified.

Our immediate approach to such a challenge was that of focusing on details, preferring the micro to the macro aspects, keeping closer to humans rather than machines, on variety and diversity rather than performance, on adaptability and a less rigid architecture. Our most important deliverable on the subject recites:

> To engage this challenge we must keep faith that a minimalist approach will be the best and most generative. We suggest to consider the field of analysis as microscopic: we aren't analysing the economy of the big numbers, but only its microscopically developed medium, yet the most used medium in the world, the one closest to humans. This is a context-shift. [...] Interconnectivity among different possible currencies substantiates a system more resilient than one presenting a unique type of currency. A Social Blockchain design must focus on synchronicity, reusability and resilience. A Social Blockchain is neutral by design, but not enough to allow peers to meet each other, off the record, optimizing transport over distances, handshaking over any form of value, privately or communally. The biomass is neutral, the unknown its complimentary to it. People should be able to negotiate in serendipity the space between their desires and the possibilities to realize them.

> We should inform design choices at the grass-root level, rather than prospectively state a design. Indeed, a municipal currency can be designed with an architecture that resembles clearing mechanisms and circulated as a

> reward for either political or working engagements, but will not create an environment that will make the innovation we are focusing on as virtuous.

> While we are dealing with innovation, we have to deal with the fact that the notion of Blockchain implies completely rethinking the landscape of currency and payments, at the structural level. Our intervention is not that to trace a map of a new landscape, rather than practicing a walk through its socially oriented possibilities, mostly marginalized by the financialisation of value markets. We intend to start from the street-level implications and how they reflect in the digital space. The concept of money we are trying to put in focus has been left underdeveloped by the growth of capitalism into a financial empire detached from his fundamental dynamics of production. At the margins of this defeating situation of abstraction between the body and soul of capitalism, we intend to adopt a quantum point of view and free ourselves from the universal narrative of finance. The decroissance analysis of capitalism is complementary to most of the theoretical assumptions made so far and we will accompany the necessary deceleration by zooming in the texture of money as medium.

in D-CENT we have put in place a process of transformation for the sort of consciousness produced by theoretical research, through a series of passages that are both technical and ethnographical. Our team at DYNE has analysed first and foremost the needs and desires of communities involved, then proceeded to compare the personas abstractions with the actual features of the technologies which were envisioned as solutions. While other pilots were fairly evolved and literate about the solutions they needed, what is most interesting for this research is the interaction with the Spanish pilot and the grass-root community of entrepreneurs (EUROCAT) lead by economist Susana Belmonte.

> In case of Eurocat in Spain the need to withdraw endorsements is expressed clearly, but that collides with the inherent features of blockchain based credits which mostly consist of "digital assets" that cannot be controlled by a central authority. In such a case it is recommendable that decentralized technologies and architectures are deployed for the resilience of the data storage (both of transactions and individual wallets), but the Eurocat system itself appears to be designed to be best operated in a centralized fashion, based on a central database.

On the other side of the spectrum for this research we have analysed the dynamics of innovation and what can be considered a political interaction on development choices, which is mostly obscure to policy analysis because deeply inscribed in technical considerations.

3.2 Scaling down technological expectations

The lesson learned from D-CENT is a very important one: when developing, never start from the technology, always start from people's needs, dreams, desires and perhaps fears. It may sound somehow obvious, yet in most of the "disruptive innovation" research it happens that the technology is considered innovative per-se, even before considering its real role for those who interact with it and within the society it is deployed.

While LEAN UX helps to frame such an approach within a clear methodology which also well integrates into the productivity patterns of industry, it is not enough to envision what is the best possible intervention on a technological level. When the vision for a design is interpreted from a "user perspective" it is easy to have an image of the final product, but harder to choose the technological path leading to it. In fact such a choice is mostly left to developers and engineers, in a somehow separate process that breaks down "user journeys" into technical specifications.

The passage between a well defined user experience to technical specification is then problematic, as much as it cannot be considered neutral, to the contrary it still embeds values and propositions affecting the socio-political aspects of the implementation. This is clear in the work done in D-CENT on the definition of the software Freecoin.

3.3 Freecoin's readable and sharable keys

First of all, we envisioned Freecoin to be a toolkit, rather than a final product:

> While conducting our research on D-CENT pilot communities and projects we did realize that, especially when dealing with value transactions, the needs are very diverse and the software implementations that can satisfy them turn out to be very different from each other, especially on the level of human-machine interaction. Therefore for our implementation task we aim at developing Freecoin as a toolkit that can deal with a common backend: distributed and authenticated ledgers, like cryptographic blockchains that can be easily adapted to draw interaction patterns that adhere to particular situations. It is important that those projects willing to adopt it are able to familiarize with the toolkit, install and control it directly. Freecoin does not configures itself as a final product, but as means to build final products that can satisfy the needs of specific deployements. All free and open source.

Then observing the abundance of (sometimes competing) blockchain implementations, it was decided to make Freecoin a "middleware", a software layer sitting between specific implementation and a more generic and uniform REST API that would serve common needs.

> Other than fostering the bottom-up appropriation and comprehension of technologies, another problem we are trying to address by developing a toolkit is that of the rapidly changing panorama for the backend we are adopting, that of blockchain technologies. The increasing development activities in this field, also previously described in D4.4, end up inhibiting developments and generating a techno-political instability around foundational choices advocated by different factions, with even spectacular clashes (for instance in case of the Bitcoin implementation) between developers, governmental and industrial interests. Therefore a toolkit like Freecoin aims to be mostly a "middle-ware" placing itself between an ad-hoc customizable layer of interaction and a process of standardization for backends that is still in-flux and should contemplate multiple options.

This is important to understand the ethical posture taken, going beyond the principles of free and open source software, by designing and developing a software component that encourages appropriation, distribution, sharing and inclusion as part of different creations:

> In short, the Freecoin toolkit is free and open source software that anyone can use to design an application using patterns for individual or collective access to distributed authenticated ledgers, recording value transactions within small or mid-sized groups of participants. Ledgers exist not only on databases, but also on peer to peer blockchains, with a sort of peer-validation of authenticated records that are tamper proof and exist beyond the actual Freecoin installation.

This is particularly important since Freecoin deals with community dynamics that have much to do with the governance choices and have therefore to be verifiable and in complete control of the communities themselves. In some cases such a need for local governance (and sovereignty on the system) goes as far as requiring the complete control on the transaction and the possibility to override systemic features as previously mentioned for the Eurocat pilot.

Furthermore, at the core of the development done in Freecoin is a portable implementation of the "Secret Sharing"[11] algorithm, allowing also the transmission and storage of the codes by analogue means (pen and paper). We have individuated a limit to the deployment of blockchain technologies in real world scenarios: the fact that once the password to access an account is lost, there is no third party that can help recuperate it. This is very important as we can easily recall many cases in which usage is sporadic and password saving is not reliable: many users today resort to the "password recovery" function and it is hard to overcome this behaviour at large. The FXC protocol in Freecoin is a proposal for a string based format to create, store and recompose secret codes that can be used to access a blockchain. Such codes can be broken in pieces and stored in different places, allowing the effective recovery of the final secret code by reconnecting at least 5 out of 9 "slices".

[11] An online demo of this software is available on https://secrets.dyne.org

The FXC protocol aims at marshalling fairly large integer numbers into strings that humans can easily note down on paper and communicate in voice. The encryption scheme still requires a machine to run software and with Freecoin we struggle to make such software free and open source, well readable and easy to re-implement.

Freecoin's approach to protect the access to blockchain operation is that of splitting the wallet key in 3 parts distributed among participants, the minting organization and an auditor.

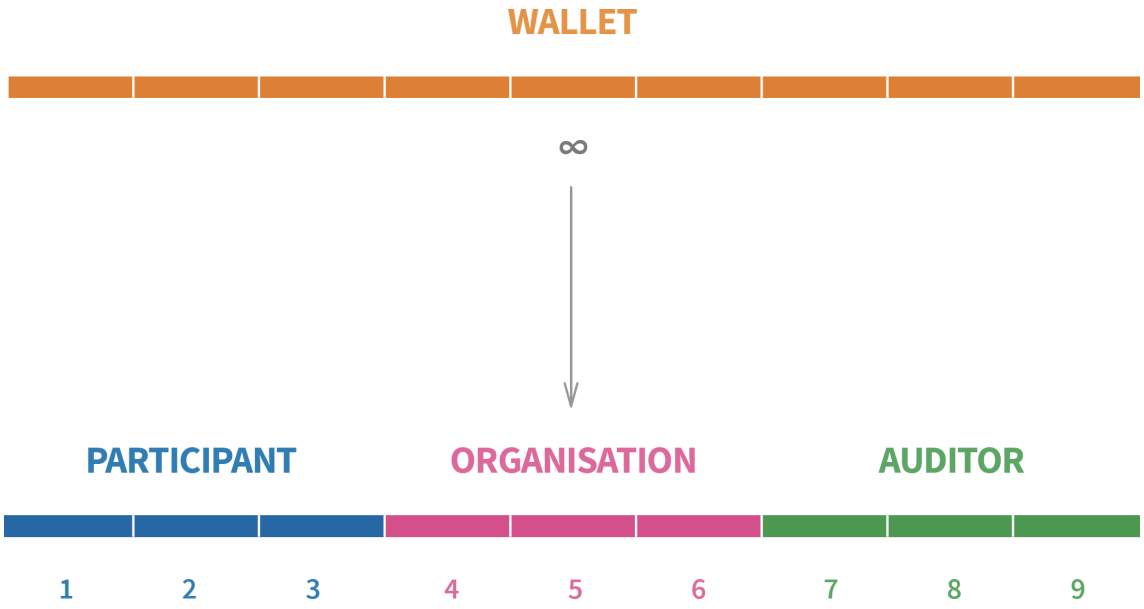

Figure 17: A sketch of the FXC Secret Sharing encryption scheme as adopted in Freecoin

More in detail, each part is constituted of three splices for a total of 9 different splices, consisting in numeric sequences of integers generated by processing the wallet key through a "Shamir secret sharing" (SSS) algorithm1. The SSS takes a secret integer number and splits it in 9 new integer numbers of which only 5 are needed to "unlock" and retrieve the secret. In our case the secret wallet key is composed by two numbers, to increase the cryptographic strenght with larger number sequences. These two numbers are marshalled into the FXC protocol.

While descending in more technical aspects of the implementation, this approach marks also some important points concerning the space for algorithmic sovereignty to take place. It allows to have a shared custody of the secrets, enabling a sort of stewardship among members of a community, yet avoiding the total centralisation for the storage and recovery of lost secrets. In these regards, FXC can be considered a concrete implementation of a cryptographic scheme that encourages algorithmic sovereignty.

Going back to a real-world scenario, the FXC implementation allows a group of people to decentralise the responsibility in keeping accesses and secrets even

without the need for a central server installation. This design pattern is different from the sort of decentralisation provided by blockchain technology, because it socialises responsibility and access rather than distributing it among "monadic" nodes of a network of individuals. Ultimately, this sort of distributed and socialised toolkit to manage a network of trust is an example implementation of the sovereignty concept applied to software architectures.

4 Dowse: making visible the invisible

Table 5: Information about the Dowse project

Project name	Dowse
Home online	http://dowse.eu
Started in	2013
Nature	Software
Advancement	Practice based
Media footage	https://www.youtube.com/watch?v=vquh3IXcduc
Source code	https://github.com/dyne/dowse

To dowse means to search for underground sources of water, metal, and other elements, using a divining rod. The Oxford English Corpus describes the word "dowsing" as:

> A technique for searching for underground water, minerals, ley lines, or anything invisible, by observing the motion of a pointer (traditionally a forked stick, now often paired bent wires) or the changes in direction of a pendulum (or dowsing rod) supposedly in response to unseen influences.

Dowse is the name given to a free software project to explore the local network and reveal automated events that are normally hidden from humans. The name was chosen to explicitly avoid commonly used metaphors in network language and develop new narratives and patterns of interaction. Dowse was created in the context of the Internet of Things (IoT), as a tool to create awareness, demystify network computing, and enable reflection and action on algorithmic sovereignty.

As the technogical race to "smarten" cities and the environment rages, the Internet of Things (IoT) is becoming the next big thing, lauded everywhere as a fantastic solution to rationalize human activity globally and help reduce uncertainty and related risks. While the possibilities for cost reduction for food distribution and insurance policy seem obvious, this analysis comes at a severe price: not only it often confuses a map with the territory, more immediately it poses tremendous risks for privacy (blowing away the constitutional right to privacy and the sanctity of the home), and exposes the problematics of aggregate information from many "smart devices", information asymmetry, and social control.

In an article from 1951, General Semantics author Alfred Korzybski explores the role of language in the perceptual process and declares:

> An awareness of the processes of abstracting clarifies the *structure* of a great many of our interpersonal, professional, etc., difficulties, which may become trivial or nonexistent if we become conscious of the identifications

> involved. Self-made problems often turn out to be no problems. (Korzybski, 1951)

Limited perception (or contextual blindness) in a highly technological environment such as posited by the Internet of Things greatly amplifies information asymmetry and the power of the technical vendor over its users. For example, a "smart fridge" would be able to resupply automatically according to your consumption patterns, and why not derive medical risk factors from the foods inside it, warning your insurance company of the presence of alcohol, sugary foods, and other health-damaging products that would allow the company to assess your risk more precisely and adjust your health insurance fees accordingly. Since the device operates from home, it can easily access private information, infer personal behaviors and habits, and influence them directly, with complete disregard for ethical, deontological, and legal considerations.

The Internet of Things is based on the assumption that more data and more precise measurements provide more accurate models of governance (Gluhak and Kranenburg, 2012). But while we see a growth of the possibilities to harvest data from "smart environments", while sensors proliferate in public and private spaces, there is limited knowledge on how to understand the algorithms that process such data, or even proprietary barriers from the very access to such algorithms.

Following a practice based approach on the topic I've conceptualised Dowse as a process leading to a well-defined product:

> Dowse is a smart digital network appliance for home based local area networks (LAN), but also small and medium business offices, that makes it possible to connect objects and people in a friendly, conscious and responsible manner.

Dowse is following a minimalist approach while studying the main nodes of "power" on a network, in this case purely intended as sovereign power: the ISO/OSI layer 2 and 3 along with the physical hub/gateway device:

> As the concentration of network-connected devices and applications increases, so does the volume and complexity of network activity. While these network actors communicate on ever greater scales, the central device which interconnects them has remained basically the same. The so-called gateway or router is usually provided and programmed by an ISP, and meant to be largely ignored by the 'user.'

> The router is opaque in terminology and an engineering of disempowerment in practice. By making the router an esoteric device, a closed device, a device which hides under the couch, opportunities to create, distribute, and use software which properly govern the small-scale network are lost.

> The centrality of the gateway device in the home/office puts it in a position of unique power and future opportunity. It is the locus of discovery, communication, and regulation between connected devices. It forms the fundamental structural matrix for the Internet of Things at the most basic scale.

> We see an opportunity to create a hub which is a part of the experience of the networked person, the networked household, the owner of devices, the Internet participant. While the term "hub" echoes the era of 10Base-T (Ethernet over twisted pair), it seems appropriate to revive the term now, as we seek a new set of generic non-authoritarian terminology to talk about the device which joins the other devices in our local network.

The Dowse project started first and foremost as a collective reflection on these matters, facilitated by the process of drafting a "whitepaper" document. I've ran debates and interviews with various practitioners from various fields: city planners, software developers, bankers and philosophers. All the input given has been organically coalescing into a peer-reviewed document which took shape in the arc of 1 year, then published around 2014. In its incipit, the Dowse whitepaper introduction recites:

> Running a network in the age of the Internet of Things means hosting the connectivity of multiple devices owned by a diversity of subjects. Often such devices have full access to private, common and public information about humans operating them. Furthermore, devices can talk to each other without humans being consulted, and such interactions are not even manifest. This situation raises issues that are not just technical, but socio-political, about the way connections happen without human consent, within local networks and towards the outside, to and from the Internet.

> The risks of unconscious abuse and exploitation of information asymmetry are growing tremendously. As things initiate on the behalf of users, we are making a major leap towards a world that provides us with contexts that we may not want at all. Getting insight on such situations is crucial for societies at large.

Bringing forward a concern that is mostly bound to consciousness, but also standing issues in terms of liability and responsibility for choices being made automatically or semi-automatically. With the advent of portable computing devices, the act of "opening a context" is extremely opaque to the subjects whose information is being handled and elaborated. It has been a while since we witnessed this silent transition to software that switches itself on and start conversating with other devices and on-line servers without even noticing us. This can be easily seen as a controversial aspect in IoT based computing, further described in Dowse:

> In the IoT paradigm, having a clear overview of what goes in and out of the

> network becomes of crucial importance for home users and professionals. The ultimate question of responsibility for whatever happens within a network cannot be easily answered, considering the way things can autonomously decide to initiate communications.

The risk for privacy isn't just a private matter. It is already possible (and actually performed) to detect how many people live in a household simply looking at the power consumption patterns. Having your fridge order milk when your stock is running low may sound great in certain contexts and indeed provide an effective way for food catering networks to gain actual knowledge of consumption, to adjust production and transport logistics. The IoT literature brings many examples of how electronic sensors, constant measurements, and 1:1-scale pattern recognition can save lives, time, money, reduce pollution, optimise production, etc.

What is lacking from all such literature is a reflection on societal risks of IoT. Most of this literature suggests that by harvesting more data from objects we interact with and running algorithms that are considered somehow smarter by having access to more information and "improve future decisions":

> Smart communication protocols and algorithms make use of several methods and techniques (such as machine learning techniques, decision making techniques, knowledge representation, network management, network optimization, problem solution techniques, and so on), to communicate the network devices to transfer data between them. They can be used to perceive the network conditions, or the user behavior, in order to dynamically plan, adapt, decide, take the appropriate actions, and learn from the consequences of its actions. The algorithms can make use of the information gathered from the protocol in order to sense the environment, plan actions according to the input, take consciousness of what is happening in the environment, and take the appropriate decisions using a reasoning engine. Goals such as decide which scenario fits best its end-to-end purpose, or environment prediction, can be achieved with smart protocols and algorithms. Moreover, they could learn from the past and use this knowledge to improve future decisions. (Rodrigues, 2012)

There are three interesting characteristics to be noted in the work done conceptualising and realising Dowse, which can be considered useful conclusions for design patterns and practices when developing a devices in such a socialised context as IoT.

4.1 Methodology: artist lead tech iterations

Among the lessons learned developing Dowse, the first is the methodology. We have mentioned initiating the project with the research and inclusion of various points of views by people that may need or be affected by the development of Dowse, that resulted in its whitepaper.

What followed up then was a pragmatic decision to include artists in the early

phases of the design process, by the implementation approach of exposing most of the network data and events gathered by Dowse, so that other projects can use them in various ways. By using open protocols widely adopted by digital art communities (primarily OSC and websockets, followed by MQTT) and running workshops on their use, we opened to "artist lead design"(Castagné, 2014) the early formulation of any algorithm dealing with such information. Thanks to this, the project has immediately involved artists (interestingly enough mostly from theatre and performance) into its development process, accelerating into the LEAN UX methodology. As of today and in the midst of its development Dowse already offers multiple solutions to signal network events in ways that are aesthetically more elaborate than the usual appliance interfaces and, what is most important, ways that break out of the screen to interface with different sensorial devices.

Figure 18: Dowse has found a place in the practice of a growing number of artists and was featured in a workshop at the Live Peformers Meeting in Amsterdam during 2016 (photo courtesy of Federico Bonelli)

Artist lead development can be an extremely valid approach even for projects of functional nature. The way to implement it at its best is that of studying which open protocols are most used by artists experimenting in fields as digital media, multi-media and synesthetic audio/visuals, and then implementing an easy to use bridge to the production of information in those formats, allowing customisation and development of algorithms to the maximum possible extent of *Freedom* and independence from the original project.

4.2 The Narrative Counts

In order to relate people to the comprehension of dynamics of privacy, defense from intrusion and comprehension of algorithms regulating the network space, the language of engineering often resorts to militaristic metaphores, in fact inheriting the origin of such technologies. Words as “firewall”, “shield”, “defense” and “guard” are overly present in network devices which are built mostly to enforce virtual walls around a local area network rather than consciousness for those who inhabit it. Dowse took its own way following a radically different posture on these matters, as the whitepaper states:

> Dowse is not only a functional tool, but a symbolic operation proposing a different linguistic approach to networking. [...] Privacy awareness (rather than protection) is envisioned and presented to its users not as a violent process, but as a responsible, natural act - one in search of harmony among those things connecting the inside and outside of a person’s private, common, and public aspects of life.

This paternalist approach consisting in keeping people in the dark and delivering ready-made black-box technologies favors the dismantling of the private and public spheres, and the impotence of users to express their ideas and share their concern without giving in to polarizing and antagonistic concepts, ultimately leading to the foundation of an alienating technosphere where humans become remote-controlled guinea pigs in a hyper-rational corporate experiment.

Reframing the language of networking away from military imagery contributes to the re-appropriation of technologies by citizens who are literaly flooded by “smart devices”, sometimes unknowingly. If Dowse grants citizens awareness of actual network events, a demilitarized narrative enables the collective questioning of the dubious premises and false promises of IoT; participants can reclaim their local network, the sanctuary of their home, the private spaces that would be controlled remotely by third-parties “for your own good”.

4.3 Functional transparency

A critical engineering project, Dowse strives for *transparency*. First published in Berlin in October 2011 by the “Critical Engineering Group”, composed of artists and engineers Julian Oliver, Gordan Savičić, and Danja Vasiliev, the “Critical Engineering Manifesto” enumerates guidelines for engineers to consider when embracing technological change from a critical perspective, and was translated so far into 17 languages. The full text of this manifesto can be found as an appendix of this thesis.

With a growing group of enthusiastic developers and contributors, the effervescence of ideas and work constrasts with the practical need of the project to keep things

simple: given the high responsibility commanded by its task, Dowse must keep its code base accessible and auditable; this is achieved by focusing on minimalism, and narrowing down functionality rather than extending it.

A simple ON/OFF switch empowers users to control which connected things can access the network. This simple concept is easily comprehended by most people approaching the project, especially considering it is increasingly difficult to switch off or disconnect mobile devices, notwithstanding the usual lack of useful feedback for such operations. Such a simple yet empowering tool changes the local network awareness from complete ignorance to instant control at your fingertip. Advanced functionality can extend this basic control, for example to filter access and authorize selected devices to connect to selected servers and domains without growing complexity nor challenging comprehension.

4.4 Status of the project

At the time of writing, Dowse has released its version 0.9 and is approaching the beta stage by the end of 2016. Its development has been progressively recognised by institutions and organisations concerned with privacy, security and the health of networks from a citizen perspective and it is currently funded by NLNET, in charge of scientific R&D on network protocols, and SIDNfonds, another Dutch organisation bound to the official entity tasked with the assignation of domain names for the `.nl` TLD.

Beyond the functional merits of the project, the success of Dowse is better understood by taking into account its critical approach on technological sovereignty. The industrial embrace of IoT is at times euphoric: a marketed narrative which is unquestioning and almost fanatical, it follows the growth of mobile communication and the Internet, omitting its critical aspects. The Dowse project, simply pointing to information asymmetry and the poverty of technological literacy, aims to enable the collective elaboration of an intentional discourse on the meaning of local network, data transmission and sovereignty.

Situational awareness is only the first step towards making informed decisions, but the lack of it signifies a complete submission to existing processes. Quoting Jacques Rancière:

> "Critical art is an art that aims to produce a new perception of the world, and therefore to create a commitment to its transformation. This schema, very simple in appearance, is actually the conjunction of three processes: first, the production of a sensory form of 'strangeness'; second, the development of an awareness of the reason for that strangeness and third, a mobilization of individuals as a result of that awareness." (Rancière, 2010)

An example of such collective emancipation process appears in the next section on the creation of Devuan GNU/Linux free software operating system.

Figure 19: Dowse pendulum photo by Anatole B. Shaw, 2014

5 Devuan: the anatomy of a fork

Table 6: Information about the Devuan project, fork of Debian

Project name	Devuan
Home online	https://devuan.org
Started in	2014
Nature	Software
Advancement	Participatory
Media footage	https://www.youtube.com/watch?v=wMvyOGawNwo
Source code	https://git.devuan.org
Mail forum	https://lists.dyne.org/lurker/list/dng.en.html
Web forum	https://dev1galaxy.org

This project is to further evolve my research question on the assumptions laid down so far. Let us ask now: what are the traits for a (huge conglomerate) of algorithms to acquire the dimension of a sovereignty controlled by its participants? How does a socialised truth looks like, beyond the impossible assumption of universal neutrality? Beyond its mere execution, what in the design of an operating system generates or erodes the trust people put into it?

In this chapter I will describe at length the technical and socio-political conditions for the birth of a new GNU/Linux operating system. The widespread resistance to the introduction inside Debian of a new central framework for system management, called "systemd", is an extremely interesting opportunity to explore such dynamics in details. I've engaged this project in first person and this chapter is mostly written in the form of participatory research, reporting various accounts contributed by people who involved themselves into the creation of a new system, a sort of independent exodus for half of the active Debian user population.

The analysis of this episode is important to provide an advanced answer to my research question. As demonstrated, the design and adoption of software algorithms has a close relation to the design and adoption of systems of governance. In the experience of Bitcoin, it is evident how the fundamental trait of such a governance is the concept of neutrality, which has much to share with the definition of truth.

In the experience of Devuan it becomes clear that, even having a "democratic" system that relies on majority voting for the main choices affecting the design of its algorithms, as its the case for Debian, is not a sufficient condition to define the participants to this system as exercising sovereignty on the algorithms. In this particular case not only the needs of a 49% minority were disregarded, but the backward compatibility of the system was broken (Ellison and Fudenberg, 2000). In

case of Debian this became a sufficient condition to motivate the enormous effort of a fork.

After taking part as a leading participant into this initiative, for my own need to avoid systemd in production-level and mission critical projects, it became evident how this experience opened a new useful direction to explore in practice what I want to be a more complete and detailed answer to my research question.

This chapter will briefly outline the Debian project, which is worldwide renown and has been covered by valuable research quoted here. Then will proceed documenting the decision that in 2014 has torn the Debian community (and other GNU/Linux communities at large) to the point of making people switch (especially sysadmins) to platforms like BSD. Such episode has more or less marginally started a process of erosion for the trust in the Debian distribution. The tearing decision that was taken regarded the adoption of systemd, a new daemon managing the whole OS in place of the historical SysV scripts. The resentment and frustration for the faction against that adoption justifies in the eyes of many the creation of the Devuan fork.

The topic is still extremely controversial and as usual the voices of those who have "lost the battle" are the less heard in the future of the Debian project, sometimes even derided. This chapter provides an historical archive of some of the wisest voices in the debate, with a selected series of witnesses providing headspace for the comprehension and interpretation of this literature in terms of sovereignty, subjectivity and self-determination in technology.

5.1 The Debian project

Debian is one of the most popular GNU/Linux distribution and a uniquely big volunteer driven effort to build a UNIX-like operating system that can run on most computers or electronic devices and its mostly made of free and open source software. Debian calls itself a "universal operating system" and covers an impressive quantity of target architectures and has been adopted as a base for many more distributions. It was first created in 1993 by its now defunct author Ian Murdock and later supported by the Free Software Foundation and the GNU project (Murdock, 1994). It gathered a fast-growing community of passionate people who work on it on a voluntary basis, with the main incentive of reputation. As of today, Debian has access to online repositories that contain over 50,000 software packages making it one of the largest software compilations in the world.

Debian and the fork of it into Devuan is an extremely relevant subject for this thesis because, as other researchers pointed out:

> Debian is one of the largest F/OSS projects, although it is also one that has received very little academic analysis to date. Debian is a non-commercial

> version of the GNU/Linux operating system maintained by over 900 volunteer developers who package a wide variety of software applications. With the possible exception of the Free Software Foundation's (FSF) GNU project, Debian demonstrates the most overt commitment to the principles of free software as originally expressed by the FSF and the GNU General Public License (GPL) of any large F/OSS project. Debian developers have modified and extended these principles in their social contract and Debian Free Software Guidelines (DFSG).3 While in some respects, Debian is unique in its explicit ethical codes, its uniqueness should not obscure its wider relevance. The type of moral cultivation that forms an important facet of the Debian social sphere is a component of many other F/OSS projects in less accentuated forms. Debian is ideal to analyze because it brings into clear relief the subtle yet significant processes of ethical socialization that occur in most F/OSS projects. Though we focus on the domain of F/OSS, this case study also serves as an example of the unique ways in which an ethical social order is built and sustained in the "immaterial" setting of the Internet. (Coleman and Hill, n.d.)

The strong commitment of Debian to software freedom has gathered through the years what we can consider a large population of ethically attuned users and developers. While there are other GNU/Linux and BSD software distributions that have refined their ethical commitment to freedom,[12] Debian is most notable because of its size and outreach, combined to the fact it is an ethically-oriented volunteer-based project.

> F/OSS developers' attitudes toward freedom, set in broad terms in the larger F/OSS community and reinforced through the broad hacker public sphere, are particularized and reinforced in the context of individual F/OSS projects. [...] Our data shows that over the course of participating in the Debian project, developers move toward a more vigorous and overt ethical stance toward the uniqueness of their project and the importance of free software than when first joining. (Coleman and Hill, n.d.)

At the base of Debian there is its "Social Contract" with guarantees which include:

- Ensuring that the operating system remains open and free.
- Giving improvements back to the community which made the operating system possible.
- Not hiding problems with the software or organization.
- Staying focused on the users and the software that started the phenomena.
- Making it possible for the software to be used with non-free software.
- A set of guidelines for distributing free software.

The Debian project ratified its social contract version 1.0 on July 5, 1997. This document served to establish a clear differentiation when compared to other big GNU/Linux distributions, most importantly Red Hat, which have a corporate gov-

[12] The Free Software Foundation maintains a list of 100% free GNU/Linux and BSD distributions at the webpage address https://www.gnu.org/distros/free-distros.en.html

ernance and are characterised as business ventures, arguably with the merit of demonstrating F/OSS can be commercially sustainable and socially rewarding.

Debian changed a lot since its inception and the episode we describe here is undoubtedly the biggest crisis this initiative has ever suffered in terms of technical choices and community reaction to them. It was a practical schism between supporters and detractors of the switch to systemd, that split the community in half.

[13] Debian is not considered a 100% free operating system because it still allows the installation of non-free software for certain tasks, according to the GNU project.

5.2 The legacy init system: sysvinit

For a long time, the services handler called System V Init (also known sysvinit) has been responsible for the initialization of services after a reboot. Sysvinit is considered a "style" of system startup configuration, it was derived from AT&T's UNIX System III and later adopted into its System V from which it takes the name. Sysvinit is based on runlevels: at any moment the running operating system is considered to be in a specific state, numbered from 0 (shutdown) to 6 (reboot). Switching from one runlevel to another causes a per-runlevel set of scripts to be run, which typically mount filesystems, start or stop daemons, start or stop the graphical desktop, etc.

With each new release of GNU/Linux and BSD distributions, this init system, comprised of many scripts that have been written by different people and has been stratified on top of each other through decades to solve various problems, started to look more like an antique to many people.

One of the fundamental issues mentioned as an argument against the continued use of sysvinit is that it runs in a linear fashion, which is hardly efficient when parallelisation is possible.

Until approximately 2010 sysvinit was the init system of choice for most distributions and it was supposed to cover only the init role in operating systems, packed as a distribution of readable scripts written in procedural programming style and contributed to the project by various people around the world. Such scripts would often take values from configuration files, or carry the configuration values within themselves.

The sysvinit package weights approximately 560 Kilobytes and is comprised of 75 files and 15.000 lines of code.

[13] The Free Software Foundation maintains a list of 100% free GNU/Linux and BSD distributions at the webpage address https://www.gnu.org/distros/free-distros.en.html

5.3 The controversial innovation: systemd

Systemd was born with the main goal of unifying basic GNU/Linux configurations and service behaviors across all distributions. It has the ambition of verticalizing the integration of calls and access rules in a single monolitic system that spans from the kernel layer (ISO/OSI layers 3, 4 and 5) to the presentation and application layer (ISO/OSI layers 6 and 7). Quoting authors on the intentions behind system development:

> We try to get rid of many of the more pointless differences of the various distributions in various areas of the core OS. As part of that we sometimes adopt schemes that were previously used by only one of the distributions and push it to a level where it's the default of systemd, trying to gently push everybody towards the same set of basic configuration.

While systemd is most known for substituting the sysvinit and other init systems across distributions, this is not what it simply is. It is comprised of approximately 69 binaries and covers several roles as a central point that offers command and control operations to other software running in application space. Its main goal is to unify the interaction interface, standardize the code that handles the tasks covered using C language (compiled, hence not directly readable in production environments) and ultimately handle several base and mission-critical tasks as network configuration, logging of operations, virtualization setup, user login authentication.

Systemd also interfaces new specific Linux Kernel features as cgroups and is de-facto the only alternative that is actively developed to cover such new features, mostly due to lack of investment from other companies in the labour needed to achieve such results.

What is also notable is that systemd records initialization instructions for each daemon (background running software) in a configuration file (referred to as a "unit file") that uses a declarative language, replacing the traditionally used startup shell scripts written in a procedural language that is not compiled and therefore can be easily read an modified directly even on a running system.

The systemd software is also bound to a US patent 20150040216-A1 "Systems and Methods for Restricting Application Binary Interfaces" filed by Paul Moore, Dan Walsh and Lennart Poettering on behalf of Red Hat inc.

Arguably, referring to the patent filed by its main author, systemd may adopt this approach as the underpinning of its security model: lock down the functioning of the system in a binary architecture that, despite systemd being opensource, cannot be changed (nor verified) in production environments.

This approach is regarded as naive by many and presents several flaws which may appear on the long term, at least this is the opinion I share with most of the

community of professionals that is trying to resist the pervasive adoption of systemd in most GNU/Linux distributions. Yet regardless of us being right or wrong about this assessment, what is really important in the eyes of many people is the freedom of choice.

In 2014 and after a much heated debate, the chosen replacement by Debian was systemd. Debian followed as one of the last distributions to make the switch, as the controversial move was already made by other distros back in 2012 and in every single episode this caused considerable grief between supporters and detractors.

At the time of writing, systemd depends from other two big software components, dbus and glib, and is made out of 900 files and 224.000 lines of code, for a total weight of 125Kb source-code written in C language.

5.4 Summary of arguments for sysvinit

The systemd software presents several arguments for its adoption, mentioning advantages based on its ease of use, integration with application space and uniform API. There are interesting arguments also by those trying to defent the old system, sysvinit, as they do not touch the rethoric of innovation and efficiency, but try to defend the diversity and integration of different systems.

Referring to the debate that has taken place within the Debian users and developers community[14], the perceived virtues of sysvinit can be summarised as follows:

- It is integrated in the Debian ecosystem and this integration is well tested.
- It works on all supported architectures and kernels
- Has a long history (System V R4, 1988?), and so is widely used, well documented and understood by users of UNIX-like systems; /etc/rc.d initscripts are still used today by FreeBSD (since at least 1999), NetBSD since v1.5, OpenBSD uses something similar since v4.9
- Is the smallest and simplest initsystem available
- Init scripts can be improved using abstraction and addtional features can be implemented as suplemental software that could be used from shell scripts.
- Many people using sysvinit know how to fix an unbootable system: boot with init=/bin/sh and you can edit or invoke the initscripts by hand.
- Init scripts are a standard interface for many other purposes. Things like heartbeat, pacemaker, ... can all call initscripts directly.
- Init scripts are most easy to understand and debug. All important logic is contained in them, code is very readable, intuitive and simple and debug outputs can be inserted everywhere.

The most prominent arguments against its change are summarized by the following

[14]The summary reported here is based on information published on the Debian wiki and only slightly edited for the purpose to give a general overview on the issue.

points:

- Any change at all involves work, learning something new, and therefore should be a choice; this decision feels somewhat forced upon us by some who insist it is better for us.
- If GNOME[15] requires logind which requires systemd, it is unreasonable to impose such a change on the whole distribution, and ought to find an alternative which doesn't.

The consequences of a change from sysvinit to systemd were assessed as follows:

- Estimates are that up to 1200 Debian packages may be required to adapt to, and continue to support, such a change.
- If packages remove their current initscripts, ports like kFreeBSD and Hurd may be unable to use software that was working before this. (Unless the new initsystem can be ported; this is most likely for OpenRC but completely ruled out for systemd).
- Much third-party software outside of the Debian archive, or in-house projects within business, may no longer work without taking effort to port them to a new initsystem.
- Some maintainers object to maintaining separate initscripts to support alternate systems.
- Declarative systems are nice to look at and might cause fewer problems, but if there is a problem, they are almost impossible to debug. Even more so if you only have a non-working system (as the problem would be in the init system).
- At least judging from their documentation they all support significant logging.
- Even given the disadvantages of editing init scripts (hard to merge with upgrades, ...) and the resulting reluctance of admins to edit them directly, almost every admin has used this ultima ratio at some time, which indicates the missing flexibility to do so might be a problem with other init systems.

Further, here a summary of arguments that were put forward against systemd:

- Not portable, by design; uses Linux-specific functionality not available on other platforms.
- Binds Debian to being a Linux distribution; even some Linux ports may have to be dropped if systemd stops supporting them.
- Violates traditional UNIX principles of simplicity and the separation of kernel/initsystem/userland duties such that components are interchangeable.
- Begins a precedent for disruptive change, which may happen again if systemd changes its APIs or deprecates some interfaces (has been described as resembling 'vendor lock-in'; really meaning we must continue to keep up with changes mandated by upstream).

[15] A graphical and widely integrated desktop framework comprising many user-facing functionalities for GNU/Linux desktops.

- Not all functionality can be controlled by configuration files and is instead contained in compiled code and if you need to change something it is difficult.
- Principle of all-mighty solutions is in its nature authoritarian. All-in-one "solutions" are a characteristic of big companies that want to seduce their users into a vicious circle of using their integrated software, while sacrificing other tools that would give them more benefit, thereby realizing abusive psychological advantage over the user.
- Init system supported by a big company may have better support but that support is influenced by that company's interests and may not align with the community values.
- They depend on DBus, which means that if DBus is not working system initialization cannot begin.
- Key feature of free software is forking, which saves the software from falling into hands of corrupted people. When a project is corrupted, good developers can fork it and make a new project. But if a software is too complex, then forked project is either also complex, and that could demotivate developers to work on such project.

5.5 The General Resolution vote

When the pressure built up towards the decision of renewing the init system in Debian, the debates became absolutely heated and in many instances the community gave the worst show of itself, from both factions. This was so because the decision is an extremely important one which will arguably change the course of life of Debian as well that of many other distributions.

To tame down the debate to reason and call for a vote among the responsible developers, on the 16th October 2014 a Debian developer, Ian Jackson, called for a "General Resolution" vote with the following proposal. His proposal did not aim to affect the choice of the new init, but mostly the way the change should be handled, respecting the freedom of users and not imposing the change on everyone:

```
0. Rationale

  Debian has decided (via the technical committee) to
     change its
  default init system for the next release. The
     technical committee
  decided not to decide about the question of "coupling"
     i.e. whether
  other packages in Debian may depend on a particular
     init system.
```

```
  This GR seeks to preserve the freedom of our users now
     to select an
  init system of their choice, and the project's freedom
     to select a
  different init system in the future. It will avoid
     Debian becoming
  accidentally locked in to a particular init system
     (for example,
  because so much unrelated software has ended up
     depending on a
  particular init system that the burden of effort
     required to change
  init system becomes too great). A number of init
     systems exist, and
  it is clear that there is not yet broad consensus as
     to what the
  best init system might look like.

  This GR does not make any comment on the relative
     merits of
  different init systems; the technical committee has
     decided upon the
  default init system for Linux for jessie.

1. Exercise of the TC's power to set policy

  For jessie and later releases, the TC's power to set
     technical
  policy (Constitution 6.1.1) is exercised as follows:

2. Loose coupling of init systems

  In general, software may not require a specific init
     system to be
  pid 1.  The exceptions to this are as follows:

   * alternative init system implementations
   * special-use packages such as managers for init
      systems
```

```
   * cooperating groups of packages intended for use
     with specific init
     systems

  provided that these are not themselves required by
     other software
  whose main purpose is not the operation of a specific
     init system.

  Degraded operation with some init systems is
     tolerable, so long as
  the degradation is no worse than what the Debian
     project would
  consider a tolerable (non-RC) bug even if it were
     affecting all
  users.  So the lack of support for a particular init
     system does not
  excuse a bug nor reduce its severity; but conversely,
     nor is a bug
  more serious simply because it is an incompatibility
     of some software
  with some init system(s).

  Maintainers are encouraged to accept technically sound
     patches
  to enable improved interoperation with various init
     systems.

3. Notes and rubric

  This resolution is a Position Statement about Issues
     of the Day
  (Constitution 4.1.5), triggering the General
     Resolution override
  clause in the TC's resolution of the 11th of February.

  The TC's decision on the default init system for Linux
     in jessie
  stands undisturbed.
```

```
  However, the TC resolution is altered to add the
     additional text
  in sections (1) and (2) above.
```

Now this General Resolution proposal is a portrait of what we define an issue of "Algorithmic Sovereignty", its importance is confirmed by the facts that followed it.

The vote involved all "Debian Developers" took place on in November 2014 and offerent 5 different options to vote upon. Its outcome was calculated using the Schwartz criterion (Schwartz, 1972): assuming each voter must furnish an ordered preference list on candidates, the winning choice is the one preferred by a majority over every other choice in pairwise comparisons.

```
Starting results calculation at Wed Nov 19 00:02:03 2014

Option 1 "Packages may not (in general) require a
   specific init system"
Option 2 "Support for other init systems is recommended,
   but not mandatory"
Option 3 "Packages may require specific init systems if
   maintainers decide"
Option 4 "General Resolution is not required"
Option 5 "Further Discussion"

In the following table, tally[row x][col y] represents
   the votes that
option x received over option y.

                Option
             1     2     3     4     5
           ===   ===   ===   ===   ===
Option 1           133   183   147   241
Option 2    313          251   180   366
Option 3    263   172          135   286
Option 4    323   280   308          358
Option 5    212    99   166    95


Looking at row 2, column 1, Support for other init
   systems is
recommended, but not mandatory received 313 votes over
```

```
   Packages may
not (in general) require a specific init system

Looking at row 1, column 2, Packages may not (in
   general) require a
specific init system received 133 votes over Support for
   other init
systems is recommended, but not mandatory.

Option 1 Reached quorum: 241 > 47.5762545814611
Option 2 Reached quorum: 366 > 47.5762545814611
Option 3 Reached quorum: 286 > 47.5762545814611
Option 4 Reached quorum: 358 > 47.5762545814611


Option 1 passes Majority.              1.137 (241/212)
   >= 1
Option 2 passes Majority.              3.697 (366/99)
   >= 1
Option 3 passes Majority.              1.723 (286/166)
   >= 1
Option 4 passes Majority.              3.768 (358/95)
   >= 1


  Option 2 defeats Option 1 by ( 313 -  133) =  180
     votes.
  Option 3 defeats Option 1 by ( 263 -  183) =   80
     votes.
  Option 4 defeats Option 1 by ( 323 -  147) =  176
     votes.
  Option 1 defeats Option 5 by ( 241 -  212) =   29
     votes.
  Option 2 defeats Option 3 by ( 251 -  172) =   79
     votes.
  Option 4 defeats Option 2 by ( 280 -  180) =  100
     votes.
  Option 2 defeats Option 5 by ( 366 -   99) =  267
     votes.
  Option 4 defeats Option 3 by ( 308 -  135) =  173
```

```
    votes.
  Option 3 defeats Option 5 by ( 286 -  166) =  120
    votes.
  Option 4 defeats Option 5 by ( 358 -   95) =  263
    votes.


The Schwartz Set contains:
     Option 4 "General Resolution is not required"



-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-


The winners are:
     Option 4 "General Resolution is not required"

-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
```

To this result has followed the resignation of Ian Jackson, proponent of the vote and reference figure for all those who wanted to limit the impact of the systemd init switch:

5.6 Response to the GR vote

The outcome of the GR vote has been harshly criticized: by a small percentage of vote the winning option has been that of not needing a general resolution on the issue. The effect it sorted was not just of ignoring the matter put forward by more than a few concerned developers, but practically showing the fact that a large group of Debian volunteers was not willing to acknowledge any concerns on the consequences caused by the change of init.

An active and well respected moderator of Debian's forum going with the moniker of "dasein" has also resigned to his position and ran an independent analysis on the vote which shows that, beyond the curtain of the 4th option to not consider the issue there was a majority of votes which have selected that option together with a negative view on what systemd would be.

Here the issue of power in using algorithms becomes recursive, as in the very count of this vote there is another prevarication denounced by the community, referring to the way the options were presented and the votes counted.

> The results of the GR vote were diluted and obfuscated by two non-resolution outcomes. Of the three technologically-relevant resolutions to the GR, one was unequivocally pro-systemd, the other two were contra-systemd, differing primarily in phrasing (essentially the difference between "must not" and "should not").

Dasein denounced that, if we ignore for a moment the 4th option which is formulated as "no GR required" and captivates the frustration of most people involved after more than a year of incendiary debates, then the winning option is the first: "Packages may not (in general) require a specific init system". This option basically meant that freedom is maintained with regards to init system dependency and that packages requiring the OS to run a particular init system shouldn't be deemed as valid and included until they can be independent from it.

This simple statement not only aligns with Debian's mandate and commitment for user freedom, but creates a condition of separation of dependencies between the different layers of the operating system, de-facto insuring that one single program does not propagates its dependencies everywhere as systemd is doing. Here below is reported dasein's analysis.

> The GR vote was structured as a rank-ordering of five choices. These are ordinal data and need to be treated as such. To be clear: I'm saying that any attempt to demote these data to nominal, or to promote them to interval, is an act of intellectual dishonesty, deliberate or otherwise.
>
> The five choices can be broken down into two discrete and non-overlapping subsets: three choices that take a technical position on systemd/gratuitous init coupling (S/GIC), represented as:
>
> 1) MUST not
> 2) SHOULD not
> 3) MAY
>
> And two choices that do not take a technical position, summarized as: 4) "I don't want to talk about this." 5) "Let's talk about this some more."
>
> Again, each vote was a ranking of all five options, which means that each DD's vote can be represented as a vector in 5-dimensional space.
>
> In case it isn't obvious, both #4 and #5 are identical in that they are talking about the individual's perceived need for dialog, be it further or any. But they do not affect that person's relative preferences for #1-#3. At all.

Thus #4 and #5 are said to be orthogonal to #1-#3. To build an analytical dataset, we separate out the individual's opinions about the need for dialog from his/her opinions about S/GIC (and thanks to their orthogonality, without altering the relative rankings of either subset in any way).

Now, the resulting data is going to be harder to read if we don't take a moment to re-rank the votes. Re-ranking also does not affect the data in anyway. Watch. Take two example vectors:

```
12345
14532
```

When we separate out the editorial opinions about dialog, we are left with:

```
123
145
```

In case it isn't obvious at first glance, these two votes express identical rankings of the three technical choices (that is: MUST not/SHOULD/not/-MAY). Because these data are ordinal, not interval, re-ranking the data doesn't alter the results of any appropriate analytical procedure. So our vectors now read:

```
123
123
```

A total of 483 GR votes were cast, but not all of those votes are analytically usable.

Examples of unusable votes include:

- Votes that "ranked" multiple technical choices identically. (A "vote" for everything is the same as a "vote" for nothing.)

- Votes that failed to assign an explicit rank to each of the three technical choices (rendering ordinal analysis impossible).

NOTE: Although some might argue that "no vote" is basically the same as "low vote," replicable analysis is not and cannot be based on individual claims of clairvoyance. If the person actually casting the vote can't be bothered to express a simple, clear preference among three distinct choices, then there's really no point in trying to guess what s/he "really" meant, much less in trying to compare, for instance, a 42413 vote to a -2213 vote.

In all, some 125 of the GR votes (26% of the total) were unusable under these criteria.

The easiest way to think about the re-ranked data is as a series of three-digit codes that tells us the individual voter's preference among the three technical options.

Each digit of the 3-digit code represents the individual's preference among the three alternatives. The first digit position is the relative ranking of "MUST not," the second digit is the relative ranking of "SHOULD not," and the third digit is the ranking for "MAY." So a code of 123 would correspond to an extreme contra-S/GIC position (MUST not/SHOULD not/MAY), and a code of 321 would represent the opposite pole (MAY/SHOULD not/MUST not).

Aside: In case it's not obvious, we've just seamlessly deobfuscated the data by reducing what was once a huge 5-D matrix down to a handful of discrete points. Dimensionality reduction is a fascinating topic for anyone who's actually enjoying reading this.

Okay, so there are six possible "states" for each vote, but some of those "states" seem nonsensical or contradictory on their face. For example, a vote of either 132 (MUST not/MAY/SHOULD not) or 231 (MAY/MUST not/SHOULD not) would seem to be best explained by a typo. Happily, these contradictions occur in only 8 votes (6 for 132 and 2 for 231); because they are so few in number, the 8 affected data points may be included in or excluded from the analysis without affecting the ultimate results. (For simplicity's sake, this analysis excludes these 8 points.)

Ultimately, the 350 votes in the analytical dataset can be represented using the remaining four points. Let's have a look:

Code	Vote	Count	Total	Percent
123	(MUST not/SHOULD not/MAY)	94		
213	(SHOULD not/MUST not/MAY)	42	136	39%
321	(MAY/SHOULD not/MUST not)	144	144	41%
312	(SHOULD not/MAY/MUST not)	70	70	20%
		Grand Total	350	

Figure 20: Summary of Debian's GR resolution votes dataset by dasein

As I've grown tired of repeating, the two "extremes" (the ??3s and the ??1s) are easily visible in the data, and they are nearly identical in size. They serve mostly just to cancel each other out. As is often the case in political

elections, it's the folks in the middle who end up making the decision.

In this instance, those "moderates" are the 312s: folks who think that S/GIC is a Genuinely Bad Idea, but who simultaneously think that a(ny) blanket prohibition is also a Genuinely Bad Idea. (In my current mood, I do not concede their point, not even grudgingly.)

But here's the thing and there's just no getting around it: the 312s chose SHOULD NOT as their first choice. They pointedly and explicitly say that S/GIC is a bad idea, their only quibble is over Just How Bad.

To recap the quote from above: "...a requirement for a non-default init system will mean the software will be unusable for most Debian users and should normally be avoided."

"This a bad idea and should be avoided in all but the most exceptional circumstances" is in no way a pro-S/GIC stance, nor is it even slightly equivocal on the question. And any attempt to suggest otherwise is nothing less than an egregious attempt to distort/misrepresent the historical facts. And that's exactly what this thread exists to combat.

On technical merits, S/GIC lost, 206-to-144. Really. Which is precisely why "we don't even need to talk about this" was pure cowardice. QE-friggin-D

Thus endeth the lesson.[16]

This fascinating and critical analysis of a GR vote whose importance stands hardly disputed in Debian's history is not only making justice to what ultimately appears being a majority of developers in Debian, but also renders a fascinating account on how algorithms can be interpreted far beyond the "neutrality" aura results are often painted in.

5.7 The aftermath of systemd in Debian

As a follow up to all these events, Debian has witnessed a dramatic loss of interest from some crucial developers.

```
To: debian-ctte@lists.debian.org
Subject: Resignation
From: Ian Jackson
Date: Wed, 19 Nov 2014 13:08:29 +0000
```

[16] last forum post by author dasein, retreived from http://forums.debian.net/viewtopic.php?f=20&t=120652&p=576562 on 10 July 2016

```
Message-id:
   <21612.38477.245453.248600@chiark.greenend.org.uk>
I am resigning from the Technical Committee with
   immediate effect.
```

> While it is important that the views of the 30-40% of the project who agree with me should continue to be represented on the TC, I myself am clearly too controversial a figure at this point to do so. I should step aside to try to reduce the extent to which conversations about the project's governance are personalised.
>
> And, speaking personally, I am exhausted.
>
> The majority of the project have voted to say that it was wrong of me to bring this GR at this time. Despite everything that's happened, I respectfully disagree. I hope that the next time a controversial issue arises, someone will step forward to advance what might be a minority view.
>
> Thanks to everyone who has served with me on the TC. I wish those who remain on the TC the best for the future and I hope that you'll find colleagues who are as good to work with as you have been to me.
>
> I now hope to spend more of my free software time doing programming. dgit is at the top of my Debian queue, but some of my GNU and SGO projects could do with attention too.
>
> Thanks, Ian.

Not just Ian Jackson, third Debian Project Leader in 1998, filed his resignation, but also Joey Hess, after 18 years of active core participation in the project:

```
To: debian-devel@lists.debian.org
Subject: so long and thanks for all the fish
From: Joey Hess <joeyh@debian.org>
Date: Fri, 7 Nov 2014 17:04:10 -0400
```

> It's become abundantly clear that this is no longer the project I originally joined in 1996. We've made some good things, and I wish everyone well, but I'm out.
>
> Note that this also constitutes an orphaning as upstream of debhelper, alien, dpkg-repack, and debmirror.

> I will be making final orphaning uploads of other packages that are not team maintained, over the next couple of days, as bandwidth allows.

> If I have one regret from my 18 years in Debian, it's that when the Debian constitution was originally proposed, despite seeing it as dubious, I neglected to speak out against it. It's clear to me now that it's a toxic document, that has slowly but surely led Debian in very unhealthy directions.

Hess has always been a mature and useful voice in the project and also in this occasion could assess well the problems he felt beyond the main controversy of a change of the init system. Of his resignation is worth nothing what he mentions in a blog-post follow up:

> Debian used to be a lot better at that than it is now. This seems to have less to do with the size of the project, and more to do with the project having aged, ossified, and become comfortable with increasing layers of complexity around how it makes decisions. To the point that I no longer feel I can understand the decision-making process at all ...

Hess kept far from polarizing the discussion around systemd, in fact resigning any anxiety about it as he and some others believe it is possible to opt-out of it later on. What he clearly suggests is that maintaining a set of socialised procedures, a cluster of algorithms of growing complexity, has a significant taxing effect on the possibility to innovate the system. The trade-off is clear, yet here the question we should be posing ourselves is not how to make algorithms more efficient, but how to make them appropriated by the vast majority of a community, so that they can be shared and redistributed further. On these regards Hess may be proven wrong: there will be hardly a way out of the systemd init change as inner processes get more entangled and less transparent for the Debian community.

Another important developer resignation is reported here below: Roger Leigh, whose efforts to maintain sysvinit through the years and this last message have provided great motivation for the fork to happen and for Devuan to be born:

```
Date: Thu, 27 Nov 2014 13:40:10 +0000
From: Roger Leigh <rleigh@codelibre.net>
To: VUA@debianfork.org
Subject: Debian fork and systemd
Received: from nagini.codelibre.net
   (nagini.codelibre.net [80.68.93.164])
Message-ID: <20141127134010.GF1657@codelibre.net>
Content-Type: text/plain; charset=us-ascii
Content-Disposition: inline
User-Agent: Mutt/1.5.21 (2010-09-15)
Content-Length: 2760
```

```
Lines: 54
```

Hi,

I'm a Debian developer, currently quite disillusioned with what's been going on with Debian over the last two years. I'd certainly be interested in getting involved with a fork.

If systemd had just been an interchangeable init system it wouldn't be so problematic. It's the scope creep and mess of poorly-defined interdependencies that are truly shocking. Take logind, for example. When looking at how to implement XDG_RUNTIME_DIR for non-systemd inits, I couldn't find any actual specification for how to do this. That's because there isn't one, just some loosely-worded descriptions; it only exists in the systemd implementation. And the semantics of it are very poor indeed; it hasn't been developed with safety, security or flexibility in mind. We'll come to regret adopting this since the poor design decisions are likely to become entrenched.

And more recently, there have been several reports of unbootable systems. That's unconscionable, and a serious break with Debian's traditionally solid support for backward compatibility. Here, existing supported systems have had that support dropped on the floor. With sysvinit great effort was taken never to break existing configurations, and that appears to have been lost. Introducing dependency-based boot took over two stable cycles; optional in one, default in the next, mandatory after that. That could have been reduced certainly, but the point is that time was taken to ensure its correctness and robustness (and in the beginning, it did need work, so the wait was worthwhile). This has not occurred with systemd, which has been made the default yet is still not ready for production use.

Debian is developed by hundreds of active developers and used by many times more people. People rely on Debian for their jobs and businesses, their research and their hobbies. It's not a playground for such radical experimentation. systemd support was forced in rapidly and didn't just cause breakage, it caused breakage with our own past, breaking the reliable upgrades which Debian has been renowned for. Personally, I'd like to see a much higher regard for stability and backward compatibility, rather than just ripping out the old in place of the new without any regard for its true value. It might not be bleeding edge, but we already have Fedora for people who value this over a solid and dependable system. It's possible to be up-to-date without being a Fedora; Debian unstable historically made a good job of this.

Kind regards, Roger

```
--
  .'''.  Roger Leigh
 : :' :  Debian GNU/Linux
   http://people.debian.org/~rleigh/
 `. `'   schroot and sbuild
   http://alioth.debian.org/projects/buildd-tools
   `-    GPG Public Key      F33D 281D 470A B443 6756
     147C 07B3 C8BC 4083 E800
```

5.8 The Debianfork declaration

The "Debianfork" declaration was published during the period of the GR vote in November 2014 by an anonymous group calling itself "Veteran Unix Admins", referring to a seminal article by Paul Venezia defining the traits of such a professional figure[17]. In this declaration, reported below, there was a call to fork Debian over the diatribe, constituting an attractor for all those whose concerns were liquidated by the GR Vote.

As some people found out, I wrote this declaration myself after having followed closely most debates, strong of the support and critical review of a larger private group of sysadmin and programmers (more than a thousand subscribers) which is indeed called VUA. Here below the text:

5.8.0.1 Who are you?!

We are **Veteran Unix Admins** and we are concerned about what is happening to Debian GNU/Linux to the point we decided to fork the project.

5.8.0.2 And why would you do that?

Some of us are upstream developers, some professional sysadmins: we are all concerned peers interacting with Debian and derivatives on a daily basis.

We don't want to be forced to use systemd in substitution to the traditional UNIX sysvinit init, because systemd betrays the UNIX philosophy.

We contemplate adopting more recent alternatives to sysvinit, but not those undermining the basic design principles of "do one thing and do it well" with a complex collection of dozens of tightly coupled binaries and opaque logs.

5.8.0.3 Are there better solutions than forking?

[17] "Nine traits of the veteran Unix admin" published on 14 Feb. 2011 on Infoworld.com http://www.infoworld.com/article/2623488/unix/nine-traits-of-the-veteran-unix-admin.html

At the moment no, unfortunately.

The default Init system in the next Debian v.8 “Jessie” release will be systemd, bringing along a deep web of dependencies.

We need to individuate those dependencies, clean them from all packages affected and provide an alternative repository where to get them. The stability of our fork is the main priority in this phase.

5.8.0.4 Why is this happening in your opinion?

The current leadership of the project is heavily influenced by GNOME developers and too much inclined to consider desktop needs as crucial to the project, despite the fact that the majority of Debian users are tech-savvy system administrators.

Moreover Debian today is haunted by the tendency to betray its own mandate, a base principle of the Free Software movement: put the user’s rights first. What is happening now instead is that through a so called “do-ocracy” developers and package maintainers are imposing their choices on users.

5.8.0.5 Can you articulate your critique to systemd?

To paraphrase Eric S. Raymond on the issue, we see systemd being very prone to mission creep and bloat and likely to turn into a nasty hairball over the longer term.

We like controlling the startup of the system with shell scripts that are readable, because readability grants a certain level of power and consciousness for those among us who are literate, and we believe that centralizing control services, sockets, devices, mounts, etc., all within one daemon is a slap in the face of the UNIX philosophy.

A timely reply by some people willing to use systemd is visible at forkfedora.org. This page is useful to highlight a fundamental difference: systemd may simplify the task of configuring init, but it does so by enforcing an increasingly opaque approach to the init procedure. In systemd sure it appears easier: one can tweak a few variables and have all the rest handled by a big binary system that is way bigger than sysvinit.

```
  ls -lH /sbin/init
  sysvinit: -rwxr-xr-x 1 root root 36992 Jul 14  2013
     /sbin/init
  systemd: -rwxr-xr-x 1 root root 1317632 Sep  1 14:41
     /sbin/init
# You may not be veteran enough, but you're already
   pretty fat.
```

It can be said that the security model of systemd relies much more on developers and package maintainers and much less on system administrators. As Debian users we are simply asking not to be forced into this model and, taking into account the CTTE vote on the issue was almost draw, we believe its execution should be way

more attentive in listening to what many users are asking: Init Freedom! and not being entangled by systemd and its omnipresent accessories.

5.8.0.6 How long are your beards?

This is not a beard contest, rest assured the furry ones among us are not sheep.

5.8.0.7 To sum it up?

We are forking the Debian project to create a new base distribution that promotes Init Freedom.

This will take time and we will proceed step by step.

First of all and concurrently with Debian 8 "Jessie" release, we aim at having a complete solution to which current Debian users can dist-upgrade smoothly.

If you need this too, then please help: with a donation or getting involved.

5.8.0.8 We need to talk.

Sure, write an email to VUA@debianfork.org.

Following our call some people gathered on **IRC Freenode channel #debian-fork**. Be welcome.

The mailinglist is open to subscriptions. Break the ice if you feel like, take care to do it for a reason.

5.8.0.9 Are you guys alone in this?

Not at all, there are more protests against the imposition of systemd on users.

This article is a good introduction to the issue at hand: Systemd: Harbinger of the Linux apocalypse.

There is the boycott systemd website providing several references.

Then there is the "systemd fork" called uselessd with some good points and lots of lulz.

An exit strategy is being elaborated at The World After Systemd

The wikipedia page lists also some critiques in its systemd reception section.

5.8.0.10 Thanks for doing this. How can I help?

A small core group of Veteran Unix Admins is actively producing a *proof of concept* for the fork and setting up some infrastructure that can be used for its development.

The response to this declaration, published on the website debianfork.org, was overwhelming. The IRC channel, ran by volunteers we have never met, grew to an approximated amount of 500 regular participants in a few days from the vote. Headlines

on Slashdot, The Register UK, Heise.de, YCombinator, Linux Journal, Opennet.RU and ZDNet.FR ran stories with hundreds of comments attached. To summarise, an archive of the press coverage is maintained at the address: devuan.org/os/press/in-the-news

As the project opened for donations to be accepted to support the declared plan, without any marketing campaign and only during the first year it received approximately 10.000 EUR. To many of us who were involved into the Debianfork declaration it became clear: we needed to follow up with our plans. I was extremely lucky to not be left alone on the lead of such a project and immediately aided by Franco "Nextime" Lanza who took on him more than half of the work needed in the early stages of development.

Through work done in 2015 a new independent distribution was forked from Debian. After a quick consultation with the growing community we gathered, we decided to call this new distribution "Devuan", pronounced "Dev One" in english, since it recalled the name of Debian as well the VUA acronym in it.

5.9 General response to Devuan

Devuan coalesced the participation of a vast portion of people concerned in the course of 2 years to build a new GNU+Linux base operating system project called Devuan. The Devuan GNU+Linux project is not only of enormous interest for my thesis, but ended up leveraging my professional activity as a developer and the future of our research organization Dyne.org.

The overwhelming quantity of reactions that this event have generated is far beyond the possibility to gather a full account of it in this thesis. Nevertheless in appendix to this thesis is provided a qualitative selection of messages received in support of the effort to fork. These messages are particularly interesting since they take the time to explain, in some cases from a user perspective and in some other cases from a system administrator perspective, why the adoption of systemd was problematic, revealing a diversity of aspects that were easily missed or dismissed by systemd supporters.

Beyond most theoretical formulations, it is reasonable to think that the Devuan project gives a poignant picture of what can be perceived as "sovereignty" by software practitioners and more specifically what "UNIX principles" and "UNIX philosophy" really mean to many people as they are often used in ICT and security contexts.

5.10 Devuan is born

We decided the first focus with Devuan was not that of releasing an operating system for general use, not even a base system to build derivatives. It was that of building

a “continuous integration infrastructure” (CI) that could streamline the builds of packages with our modifications and overlap them to the existing unmodified packages in Debian.

This architecture has later revealed to be the best possible foundation for Devuan, leading to a modernisation of Debian’s infrastructure and to the creation of a few new software patches and applications for the building and distribution of software package. This approach also helped to establish a trustworthy process of deployement of the algorithms into the actual platform distributed and accessed by participants, while completely appropriating (and in some cases rewriting) the production means necessary to produce the distribution.

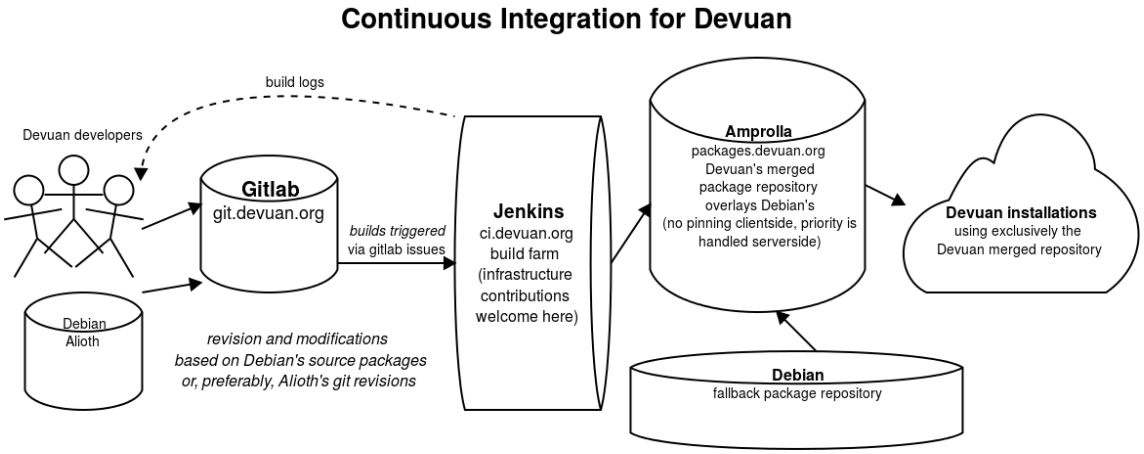

Figure 21: A graphical representation of the Continous Integration infrastructure planned for Devuan

As for the rest of the distribution, the rationale for such an approach has been that of minimalism. Assuming Debian as a trustworthy base at least until its version 7 release cycle, we needed to carefully document every single change we made to it. Our work then demonstrated how is well possible to do that with modern tools as Git and Jenkins and with an ad-hoc system for packaging which we called Amprolla.

The development continued touching the milestones of some alpha and beta releases until a stable release in mid 2017, only after the community behind Devuan considered the current beta to be stable enough to even use it in production. It is important to consider that the early focus on a solid CI for Devuan has also meant it took the role of a base distribution open to customization (appropriation) and that overall adapted to the sort of circular economy described in the first chapter of this thesis. With that in mind, it is easy to imagine how Devuan became a base distribution for other creations already in its early stage with up to 12 derivative operating systems at the time of writing, so called downstreams developments. The current development goes in the direction of curating the necessary changes in existing packages, welcoming a growing number of developers and establishing good practices for decision making, consolidating the on-line package availability under

heavy infrastructural load and developing from scratch a "Simple Development Kit" which allows people to build ad-hoc versions of their own operating system based on a very minimal base of Devuan. One of the first results of this possibility was the adoption of the system by co-developer Enzo "Katolaz" Nicosia (Queen Mary Univ. UK) to build a "Devuan Minimal Live" system which became extremely useful for the blind GNU/Linux user community for its simplicity and adherence to text interfaces.

In 2017 the Devuan GNU+Linux community is thriving and keeps growing, the distribution is listen in the top30 worldwide by Distrowatch, donations continue flowing in and development moves steadily towards a second stable release, while experimentations for future improvements (also to the init system) proceed at a slow pace and in good contact with the community. The chat channels have an approximate amount of 500 participants (#devuan and #debianfork on freenode.net) and the mailinglists count 1128 subscribers at the time of writing, with more than 500 daily unique visits on the websites and worldwide mirrors offered by multi-national ICT companies like Leaseweb, but also educational institutions as University Warwick and the Mathematics dept. in Princeton and historical cultural associations as NLUUG.

5.11 Conclusion

When I started studying the systemd phenomenon in 2013 I suspected it would be relevant for my Ph.D thesis, but I had not imagined it would have resolved to become an entirely new distribution. This project has an enormous positive impact not only on my part-time Ph.D research position, but also on my professional life, as it grew to consitute a base for many more projects and made Dyne.org double its size in terms of participation and donation budget. Arguably, the base for this success is not technical, but related to the very subject of algorithmic sovereignty.

What we came to call the systemd controversy revealed to be an extremely relevant issue in the software freedom movement, still unfolding its narrative. Software freedom is defined by the "four freedoms" devised by Richard Stallman and the Free Software Foundation. Two individual freedoms relate to *appropriation* (to use and to study how a program works, by reading its source code) and are completed by two collective freedoms related to *sharing*, *creation* an *distribution* (to share and modify the program's source code) that enable programmers in a community to create software on the behalf of others. This is a fundamental stance enabling social emancipation and appropriation of the technology.

The Debian Free Software Development Guidelines (Debian FSDG) was a foundational ethical document that grounded Debian as a promoter of software freedom, and inspired the Open Source Definition in 1997 (adapted by its primary author, and then Debian Project Leader: Bruce Perens). Although the four freedoms and

the GNU legacy are not mentioned in either of these documents, they form a solid base for people to understand the meaning of free software: “free software grants you freedom”.

Arguably, since systemd itself is free software, the four freedoms are a necessary but not sufficient condition to protect the interest of free software users. The four freedoms pose the question of who can participate, since programming requires special skills: if you don’t have these skills yourself, and your community doesn’t have these skills, you’re automatically excluded from controlling your own computing. With more software, lines of code and complexity, even programmers are not always able to interpret, understand, or modify software. The “avalanche” of systemd is a situation in which a new complex system is developed top-down and within a short amount of time and at the risk of security implications not being throughly verified. Such a complex system, characteristically based on CRUD[18] interaction, is immediately deployed to substitute a shared literature of algorithms in place since decades, a literature that through all those years has been *appropriated* and *shared* by communities, *distributed* also in professional and mission critical environments and on which several *creations* were based.

The systemd controversy illustrates how expectations can deceive the mind and lead to diverging, even antagonistic perspectives given the same facts. If younger developers who grew up with proprietary operating systems are mostly satisfied with a consistent set of interfaces to ease their work, or even abide to open source principles without conceiving the importance of democratic governance and diversity of needs and visions, people with experience of more UNIX-inspired systems and various deployments in largely different settings (from small companies with heterogeneous systems to large professional data centers) are more focused on portability and long-term focus on maintenance, continuity, and backwards compatibility that are definitely broken with a switch to systemd. As demonstrated by years of diatribes it is almost impossible to reconcile the two sides without a large effort to bridge the gap, that translates into a quantity of work that nobody wants to take up: on the one hand, systemd supporters want their way exactly to avoid having to rely on artesanal contextualization of their work, while systemd detractors prefer theirs to avoid having to rely on a large set of programs that require a completely alien mindset to cope with and a different governance on algorithms that completely substitutes the one in place (or arguably not in place) for decades.

Comparatively, the git program to manage source code, which shares some characteristics with systemd (a large set of very different binaries covering a large and extensible set of loosely related functionality, new interfaces from what was existing

[18]CRUD stands for “create, read, update and delete” and it indicates an interaction pattern in computer systems which allows such actions to be operated over variables in a dataset, implying the internal processes are hidden and in any case their design is not part of the interaction.

at the time of its introduction, and complex internals) didn't meet much resistance to become the most-used program in its field. This suggests that free software hackers and users (programmers, system administrators, advanced users) are willing to take up new systems as long as they choose to do so on their own terms. This is confirmed by the GR resolution vote where the option to create a hard dependency on a specific init system was marginal.

Well beyond the project and the community themselves, the Devuan experience provides new solid grounds for theoretical assumption of what algorithmic sovereignty means and what can be its potential. While this experience relates mostly to a sector of specialised professionals, is well imaginable that in a close future of pervasive computing scenarios analog situations will repeat.

After two years of working on Devuan, most decisions were taken by a handful of loosely-coordinated people who presented their results to the community. When attempting to take decisions affecting directly the community and without consultation (e.g., for the choice of a forum infrastructure that heavily differs from Debian's) there was a huge drawback on these choices, forcing resignation from roles and change of plans which were rather quickly taken up by new volunteers.

Radical decisions that affect the future of the project can be made as long as participants are given sufficient information and time to ponder it, already have positive experience of previous decisions by identified leaders, have a way to opt-in the change, or if the change is not perceived as disruptive of one's workflow. An initial effort can be made to adapt if the long-term effect is perceived as favorable.

On the level of algorithmic design, the lesson to be learned here is the difference between closed declarative designs and open scriptable ones. The difference is clear when considering an episode which was omitted so far. At the time of the debian-fork.org declaration there was a response given by the opposite camp, presumably anonymous developers of the systemd software or sympathizers, called forkfedora.org. On their website they have shown two different source code examples to start the popular UNIX mail daemon sendmail, one using sysvinit and the other using systemd, side by side. The main argument used against sysvinit was the length and simplicity of the source for the systemd example which indeed was improved. See the section "Sysvinit scripts vs. systemd service files" in Appendix for comparison.

More considerations on this project are left in the conclusion of this thesis, in particular about the role "forks" have in the context of algorithmic sovereignty.

6 Artistic interlude: entropy and design for living settlements

Table 7: Information about the Entropical artistic project

Project name	Entropical
Home online	http://entropical.org
Started in	2015
Nature	Artworks
Advancement	Context
Media footage	https://www.youtube.com/watch?v=tsTljDLXmUQ

The reason to put forward an artistic practice in this thesis lies in the belief that arts can bring resourceful insights in the process of creating value sensitive and trustworthy designs, and that dialogue among disciplines is the core of creativity (Bohm and Nichol, 2004). During the course of this research, my participation in the STARTS platform (Digital Agenda for Europe, Initiative for Science, Technology and the Arts) has encouraged me in the direction of artistic production to open a broader dialogue with society well beyond narrow areas of specialisation. It has been an enlightening and reassuring experience to hear the European Commissioner for Research, Science and Innovation state recently: "I think that more and more we all understand that innovation in the future will be on the intersection of the arts and sciences".

Entropical is an artistic research proposed as an interlude for this thesis. The Entropical research, conducted together with an artist and Permaculture expert, has been greatly inspiring for the methodology and practical analysis conducted throughout this thesis. This chapter intends to document the outcomes of Entropical that best relate to the thesis finding, as well to outline the analogies between an holistic methodology for the design of living ecosystems and a possible holistic methodology for the design of algorithms.

To widen the interdisciplinarity of my research I have moved to the far end of the design spectrum to consider practices and patterns of design in architecture, considering it the best discipline to describe the relation established between living beings and software algorithms. In particular I have been fascinated by the Permaculture design patterns, because of their strong holistic nature and because of my privileged access to the work and practices of my partner Debra Solomon, a Permaculture expert busy with the realisation of edible forests (Crawford, 2010), soil creation, phytoremediation techniques (Viljoen and Bohn, 2014) and continuous productive landscapes (Viljoen and Howe, 2012). This inspiration brought me to

draw a similitude between agile software design and permaculture design patterns, to the point this wandering across different fields became a renown lecture I have performed and I'm still performing worldwide, with the precise intention to open the field of contemplation when designing software and learning from a syncretic exercise of the mind, as recommended by our director of studies prof. Roy Ascott.

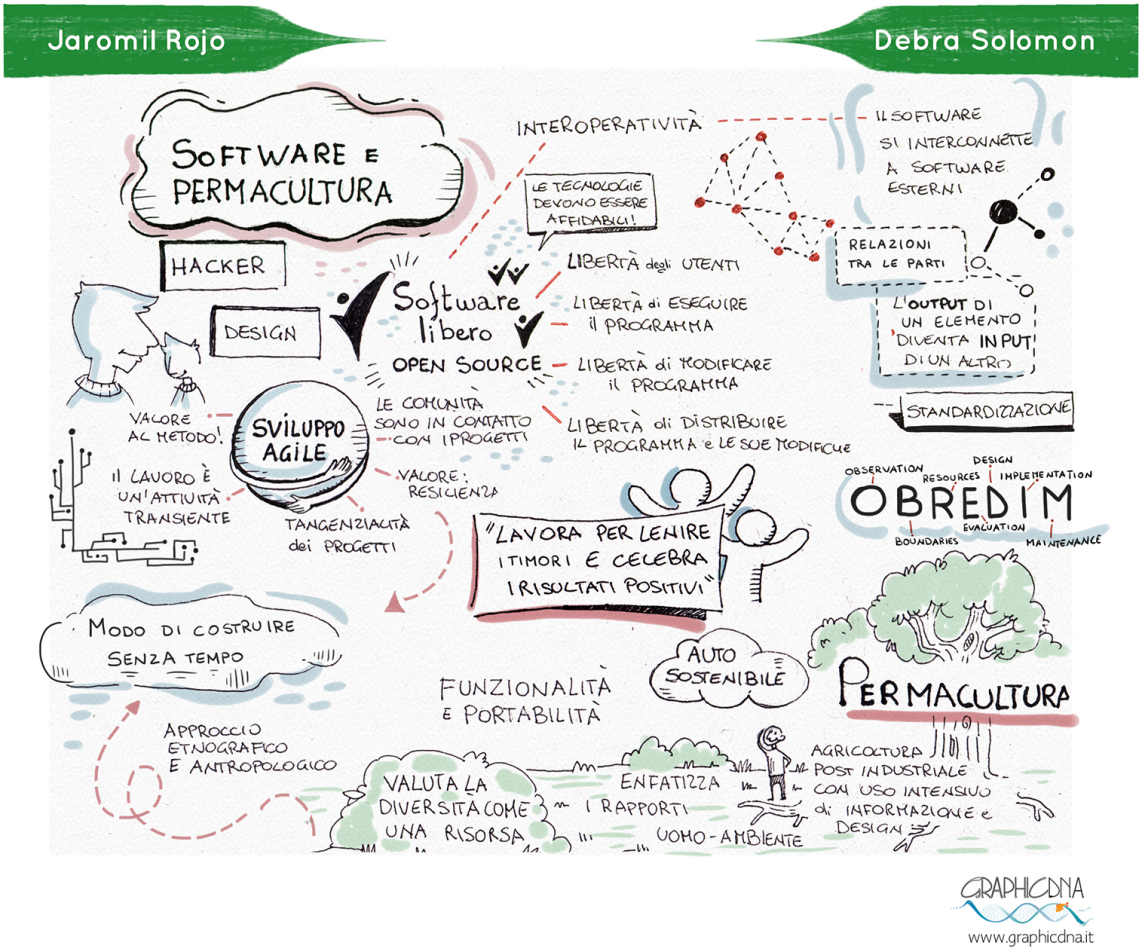

Figure 22: Graph of ideas and inspiration by the lecture "Design patterns between Free Software and Permaculture" contributed by an attendee to a workshop in held in 2012 on the premises of RuralHub

The discovery of differences and analogies between the design approaches of Permaculture (Mollison and Holmgren, 1978) and the Agile (Beck et al., 2001) software development methodology have revealed themselves to be extremely useful for my own practice and at the risk of making this thesis too eccentric I will outline the salient points and then proceed with a brief illustration of the Entropical research project. Further documentation of Entropical is contributed in the audio/visual material attached to this thesis, consisting in the filming of a debate that has seen the participation of international guests expert in the fields of arts and design.

In brief, Permaculture is an approach to the design of human settlements and agricultural systems focusing to:

- obtain a yield for human needs
- aim to create stable and productive systems
- mimic the relationships found in natural ecology
- harmoniously integrate the land with its inhabitants

It achieves its results by realising a system of sustainable land use that maximises effect and minimise work. In an historical perspective, Permaculture experts position themselves as a progression from the period of industrial agriculture and emphasise the use of design and information within their practices:

- Traditional (pre-industrial) agriculture was *labor* intensive
- Industrial agriculture is *fossil fuel* intensive
- Permaculture is *design and information* intensive

At last, according to the view proposed by Permaculture, elements in a system should be always interpreted as in relationship to other elements, where the outputs of one element become the inputs of another, a posture that has a striking analogy with the UNIX design principles well explored in the previous section about the Devuan project.

At the other side of the spectrum for this brief analysis is the Agile methodology, proposing to interpret the development process as "in transition" and always as a journey not a destination. It promotes the diversification of projects and focuses on resilience as much as on the usual emphasis put on efficiency. It establishes methods for ensuring success even before starting to organise work, in analogy with the LEAN methodology. Agile also aims at simplifying systems at every opportunity and stresses the importance of keeping contact with the communities around projects.

> Work to assuage fears, to celebrate positive results. Be patient if things get sticky. When all else fails, return to ways to create even more community. Do anything else you can to bring people closer. Celebrate every small success with food. (Schuh, 2004)

The posture of Agile development to give importance to the community adopting an algorithm resonates with the assumptions put forward by this thesis, while defining the path to feasibility to realise the sort of algorithmic sovereignty we are trying to imagine. Both Permaculture and Agile methodologies emphasise on the importance of long term sustainability. Also the *Interoperability* aspect is very strong in the design patterns of both disciplines, as well resonate in the ethical views proposed in F/OSS: Interoperability is defined as the capacity to perceives elements (especially the inner elements) of an ecotope in relationship to others, where the outputs of one element become the inputs of another and they are portable across systems. Quoting founding document of the Free Software Foundation:

> We make our software very portable and we make our software standardized

> so that other people can easily have portability, so we are aiding portability from every possible direction. Meanwhile, you see Microsoft deliberately introducing incompatibilities and deliberately blocking interoperability. (Stallman, 2004)

The introduction to this daring analogy concludes here, while the next section proceed to briefly illustrate a design methodology proposed by Permaculture that can be well adopted in the practice of designing algorithms.

6.1 OBREDIM: a design methodology going well beyond Permaculture

Within its literature Permaculture puts forward a fascinating proposal for the design practice, it is commonly referred as "OBREDIM" and consists in the formalisation of analytic steps to be taken when approaching the design of a new system without ignoring its relation to adjacent systems. The OBREDIM approach can be well related to other disciplines beyond the Permaculture focus on living settlements and helps activating a practice that keeps faith into most of the principles shared between the Permaculture and the Agile approach. OBREDIM is an acronym, mostly used as a mnemonic, and stands for:

1. **O**bservation: observation of the subject during all possible occurring circumstances and transformations of its context.
2. **B**oundaries: individuation of the boundaries touched by the subject, the surface of interaction with other subjects.
3. **R**esources: inventory of resources the subject can dispose of, what comes and goes from and to outside.
4. **E**valuation: analysis of how elements interact and define the subject, establishing the nature of the project.
5. **D**esign: definition and mapping of the subject, imagining it in all its possibly different aspects.
6. **I**mplementation: plan how to realise the subject, creating a plan of action for its full realisation.
7. **M**aintain: consideration of the maintenance aspects involved by the design and its sustainability.

In the explaination of the OBREDIM methodology offered above, the effort has been that of removing all linguistic references to a particular field of practice. The result presented well relates to both disciplines we are taking in consideration: that of designing a living settlement as well that of designing an algorithm that respects the sovereignty of participants.

Arguably the OBREDIM methodology can be applied as a guideline to envision all efforts dealing with a constituency of living participants, to include all possible

parts of such a process plus what already exists around it in the equation and final realisation. OBREDIM helps to put in relationship all salient aspects involved in the realisation of complex systems, also helping to simplify them. The most overlooked steps are the four "OBRE", points 1, 2, 3 and 4, which are evidently missing from many of the final designs for algorithms and complex software suites that tend to substitute what exist and impose themselves as an almost totalitarian solution that leaves no space for the combined agency of past and future participants.

6.2 Seven Layers and REALBOTANIK

As a final element for this interlude, an art research is presented as a practical experiment to realise an open commentary on the relationship between the abstraction of algorithms and the concreteness of natural processes. The research is also documented online on the website "Entropical.org" and in the attached audio/visual material "Entropical Finissage Conference Recording" provided in an easy to reproduce Mpeg4 file format.

Entropical is a research project started in 2015, the international year of the soil, so far consisting of four art works in which the value and dynamics of the exchange of materials in the biological world is set against the abstract value of algorithms and computer calculations. Entropical enquires into possible and imaginary ways to bring very different value systems into a direct productive relationship and it does so at a time in which intensive computation is valued far more than ecological regeneration.

In the art installation "REALBOTANIK", realised together with Debra Solomon, we presented cardboard mats inoculated with oyster mushrooms growing mycelia using the 'waste' heat released by a computer producing Bitcoin. The setup references an approach that is little know, but gaining traction within the industry: that of using heat as a byproduct of information industry, recycling it to grow nutrients (Sharma et al., 2010). The title 'REALBOTANIK' references the term 'Realpolitik', reflecting on the different value attributions for resource exchange between the different contexts of the soil organism and of financial networks, as in the notable differential between "use value" and "exchange value" in market evaluations.

REALBOTANIK elaborates on the almost poetical impossibility of a comparison between the abstract processes of value creation in finance and the material value creation of living processes. By promoting this perspective we focus our work on the contemporary production of "entropy": a word that resonates for its meaning both in the discipline of applied cryptography and in that of physics, but also in the body of works commonly referenced as "Bio-economics" (Georgescu-Roegen, 1971) which has the merit of re-contextualising the Second Law of Thermodynamics in the economic discourse.

At last, in the artwork "Seven Layers" we have operated a careful linguistic exercise

Figure 23: Photo of the REALBOTANIK installation at display in 2015 inside the "Glazenhuis" pavillon inside Amsterdam's Amstel Park

Seven Layers

	Function	*Permaculture*	*Software*	*Function*	
7	Surface area	Canopy	Application	Interprocess communication	*7*
6	Cropping	Understory	Presentation	Data en/coding	*6*
5	Zone exchange	Climbers	Session	Interhost communication	*5*
4	Coverage and Shelter	Shrubs	Transport	Reliability and Flow control	*4*
3	Pioneering and Cycling	Herbaceous	Network	Path determination	*3*
2	Remediation	Ground cover	Data link	Physical addressing	*2*
1	Nutrient exchange	Rhizo/Mycosphere	Physical	Media signal transmission	*1*

Figure 24: Reproduction of the artwork "Seven Layers", part of Entropical

around the analogy of the “seven layer principle” in Permaculture and the seven layers of the ISO/OSI engineering model. In Permaculture’s practice, contrary to mono-cultural intensive agriculture, it is recommended to conceive ground cover as a multi-layered system that has to integrate seven different layers of vegetation, an approach that improves both the botanical variety, the landscape and the nutrients and water sequestration of soil on the territory where it is applied. As an open commentary and an invite to interdisciplinarity to the audience, we have studied and put in an almost natural relationship the definitions of the two set of layered models, across very different disciplines, again highlighting the distance between them, leaving to the observer the role of connecting or imagining what is beyond this apparently lacerated condition in which we live while relating the living world with that of abstract algorithms.

7 sup: an alternative model for UNIX privilege escalation

Table 8: Information on sup UNIX privilege escalation project

Project name	sup
Home online	http://sup.dyne.org
Started in	2016
Nature	Software
Advancement	Practice based
Source code	https://github.com/dyne/sup

Intelligence communities use the concept of "need to know" (Desouza and Vanapalli, 2005) as a way to minimize information leaks and only grant access to secrets when it is absolutely necessary for the performance of an agent's action. In computer security, this principle somehow translates into the domain of "privilege escalation" (Vernotte et al., 2017) , which indicates the clearance for a program to execute actions not normally available to its calling user. Often used offensively to gain control of a target machine or extract information, privilege escalation has also legitimate use, for example to enable a desktop user to perform administrative actions such as rebooting a running system or upgrading its software packages.

The right to perform administrative actions is usually attributed to the role of the system administrator, often abbreviated in "sysadmin", who can access `root` status as well give access to privilege escalation to others for all programs present on a system or only a subset of them. Obviously this role deals with power and, despite it is kept in the technical domain of "administrative" functions, it can be of highly political nature: it is an executive power that executes decisions and grants privileges following policies, while the role of the sysadmin is highly disciplined and it is not difficult to find cultural and literary analogies between the code of conduct of a system administrator and that of an expert of martial arts.

Programs such as `su` or `sudo` are commonly used to perform operatins as `root` on a system. These programs are generic and often configured to grant unlimited administrator access for a specified command or a short period of time, given the right password. While `su` is the minimalist base implementation of a command accepting a password to turn the user into `root`, `sudo` has grown to support more complex authorization schemes (see `man 5 sudo` on a UNIX system) attracting also criticism for its growth in lines of code and arguably attack surface. On common systems such as Debian, `sudo` comes pre-configured to grant root privilege for a few minutes to

the default user upon entering their login password.

With the `sup` software development project my goal is that of shifting the power to grant privilege escalation into a completely different context, far from the reach of the sysadmin. Initially as a very ambitious and perhaps hypothetical effort, considering UNIX systems are designed to give full access to system administrators, I reached this project's goal by adopting a very minimalist approach with an implementation consisting of less than 500 lines of C code, where its simplicity is part itself of the security model proposed.

There is one immediately observable pattern into this implementation, which can be extended to many other software implementations dealing with security: that less code means also less possibility for mistakes and at the same time facilitates an open peer-review process obviously making it less demanding. This is an important point relating also to sovereignty. With such a short implementation I felt confident to adopt this system in production in public projects like Dowse or the upcoming DECODE implementation, where `sup` is a core part of the architecture. UNIX experts think twice before giving the `SUID` bit to an executable, meaning the process will run as `root` no matter who is the user invoking it, yet in `sup` there is virtually no risk in doing so because its design materially denies all known possibilities of configurability or runtime corruption of its logic, inscribing inside the program (at compile time) the privilege escalation model designed by the person building the program, with no possibility to change the model by the sysadmin operating on the system where it is deployed.

The design of `sup` is that of a simple procedural algorithm that handles a few conditions that authorize its execution and that are all "hard-coded" in a statically compiled binary. In addition to conditions that depend from external states (the identity of the caller, the command called and the location of the command) there is the powerful possibility to check the integrity of the commands invoked through `sup` by verifying their cryptographic signature. This way if a command that is granted privilege escalation by `sup` is changed by even one bit, or for instance is upgraded on the system, it will not be possible to execute it as `root` anymore. While adding very little on the Linux/BSD implementations of access control lists (ACL) it proposes a low-impact security model based on SHA256 cryptographic hashes pre-compiled inside a suid binary. This feature fundamentally allows to "freeze" the privilege escalation model of an entire operating system distribution to the exact state in which it is being distributed, making sure that no modification can occur to it if not through the same privilege escalation model. This way the process of privilege escalation is not re-negotiable and cannot be changed by means of system administration, unless desired at the moment of building the system.

The underpinning of this project is to provide an alternative to security models based on the mutability of the privilege escalation model, on continuous updates

of single components in the system and on the presence of system administration activities. To control the updates of a system is a fundamental trait defining algorithmic sovereignty: it also relates to the ability of changing the "rules of the game", or as is evident in case of Devuan to change a relevant number of conditions on the operation of a system that also affect participants.

The `sup` sourcecode deals with poweron a UNIX system, and does so in the most minimal way possible: the classical approach to UNIX system administration relies on a maintained set of packages, so that the decisions made on underlying software are delegated to distribution maintainers. This creates the situation of dependency from updates of the operating systems that need to be streamlined before the developers of software can use new features. The model proposed by `sup` is instead to empower the "bundle" approach, adopted for instance by Apple for OSX applications, where even a complex combination of different programs and libraries can run as a standalone application carrying its own privilege escalation model.

This approach also allows to produce "application bundles" that consist of customised and purpose oriented operating system distributions, a path choosen also for another project described in this thesis, DECODE. Such application bundles are de-facto complete systems made of different UNIX components that interact as separate processes rather than in the same process space. This way it is possible for an application developer to re-use existing components, blending the practice of programming with that of designing a system. Once the application bundle is ready, it can be distributed as a locked system whose update depends from the developer and cannot be interfered by any intermediary, allowing the possibility for decentralised trust models that are mostly common on popular desktop environments.

The possibilities brought forward by `sup` may or may not be desirable depending from the trust model in place between participants: it certainly does not substitutes the classic UNIX model of system maintainance and upstream updates, but allows a simple and efficient way to implement another model of appropriation, creation and distribution of software. The centrally administrated model of most GNU/Linux/BSD distributions often ends up in de/subjectivating participants (both developers and operators of software) because it prevents using updated libraries unless their "upstream" package is updated through an institutional process. It is important to consider that to operate changes on an upstream package often implies going through a rather rigid disciplinary iter that adds a considerable overhead compared to the simple effort needed for the initial upgrade intention.

7.1 Implementation details

Branded as "a small is beautiful tool for unix privilege escalation", `sup` was sourcecode originally written by Pancake in 2007, a hacker of the larger tribe who call itself

"Suckless". "Suckless Tools" are famous for their minimalism, expressed in terms of performing the minimal amount of steps, keeping the source code as small as possible, using a minimum of resources, and in the case of `sup`, granting the minimum set of privilege to specific programs in specific contexts.

I cloned the original `sup` source code written by Pancake from http://git.suckless.org/sup There was a compilation warning that I wanted to remove, so I produced the following patch:

```
[main] 87c9d0991867c29dd4c30cf672fd1cdd1277b323 - commit
   26 of 36

   72%
+     char cmd[MAXCMD];
      int i, uid, gid, ret;

      if (argc < 2 || !strcmp (argv[1], "-h"))
@@ -63,6 +65,8 @@ int main(int argc, char **argv) {
      uid = getuid ();
      gid = getgid ();

+     snprintf(cmd,MAXCMD,"%s",argv[1]);
+     fprintf(stderr,"execv: %s\n",cmd);
      for (i = 0; rules[i].cmd != NULL; i++) {
          if (*rules[i].cmd=='*' || !strcmp (argv[1],
             rules[i].cmd)) {
 #if ENFORCE
@@ -95,7 +99,7 @@ int main(int argc, char **argv) {
                  if (chdir (CHRDIR) == -1)
                      return die (1, "chdir", strerror
                         (errno));
 #endif
-             ret = execv (cmd, argv+1);
+             ret = execv (cmd, &argv[1]);
              return die (ret, "execv", strerror (errno));
          }
      }
```

Then added another patch for cosmetics. Then proceeded adding the most important feature for my own experiment: the SHA256 hashing of called binaries, in commit `c02c53bac1348302f190eb8d06be3e80edcc7692`: a feature that was also in the wishlist left by Pancake.

I proceeded further with documentation and a small refactoring of the logic to get as close as possible to literate programming (Van Wyk, 1987). This recent adoption of `sup` is consensual with the original author and the latest version and development can now be found on https://sup.dyne.org/.

`sup` is a special program in the world of free software in several aspects: it's self-contained, and like many Suckless Tools, configurable at compile time directly by editing the source code; that makes it a tool for programmers. As all privileges are defined at compile time and apply to specific versions of programs, guaranteed by their unique hash, `sup` itself is hardly packageable. It's not packageable in that it does not make sense to package such a program that only works in very specific conditions. `sup` sits between a reusable software component, and a generic software library, ready-made solution behaving like an abstract class in object-oriented programming that must be implemented to be useful at all. Thanks to its very small size and its lack of dependency on anything else than very basic C libraries available by default on POSIX-compliant systems, `sup` is uniquely composable and is best found in another program's source code repository where it serves as a controlled gateway to superuser action.

Because of its strong copyleft license (GNU General Public License version 3 or later), and its symbiotic nature, `sup` grants free software developers a competitive advantage, as the GPLv3+ practically prevents inclusion in proprietary software. It arguably encourages good licensing practice by promoting best practices for secure programming.

`sup` is thought to empower the bundle approach, where the su/sudo family of privilege escalation tools is made to empower the centrally administrated model. `sup` cannot even be packaged, since it hard-codes the privilege rules which are engraved into the package and activated at the time of installation.

As minimal as it is, the impact of the privilege escalation model of `sup` can be enormous since it enables software developers to distribute self-contained bundles and enable their access to a limited set of accepted functions engraved and approved by the user at installation time. After that, a SUID bit is set on `sup` and no more access will be asked, ultimately excluding from the process of establishing trust any intermediation by the operating system maintainers: it is a consensual agreement between the user and the developer.

`sup` doesn't suggests that the "bundle" approach is desirable above all and in any situation. But when a bundle is desirable, then the privilege escalation model proposed by `sup` is the minimal solution to a rather deep architectural change in how UNIX systems work and allow the distribution of packages, as well how hardware manufacturers can distribute their embedded device.

7.2 sup's sourcecode

```
// sup 1.2

//  (c) 2016 Dyne.org Foundation, Amsterdam

//  Written by:
//  - 2009-2011 pancake <nopcode.org>
//  - 2016      Denis Roio <jaromil@dyne.org>

// sup configuration file

#define HASH 1

#ifndef FLAGSONLY

#define USER 1000
#define GROUP -1

#define SETUID 0
#define SETGID 0

#define CHROOT ""
#define CHRDIR ""

static struct rule_t rules[] = {
    // allow user to run these programs when found in
       path location
    { USER, GROUP, "whoami",   "/usr/bin/whoami", "" },
    { USER, GROUP, "ifconfig", "/sbin/ifconfig",  "" },
    { USER, GROUP, "ls",       "/bin/ls",         "" },
    { USER, GROUP, "wifi",     "/root/wifi.sh",   "" },
    { USER, GROUP, "sleep",    "*",               "" },

    // allow to run this sha256 hashed binary when found
       in PATH */
    { USER, GROUP, "id", "*", "db533b77fc9e..." },
    /* { USER, GROUP, "*", "*"}, // allow to run any
       program found in PATH */
    { 0 },
};
```

```
#endif

// Headers
// -------

#include <errno.h>
#include <unistd.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <sys/ioctl.h>
#include <fcntl.h>
#include <libgen.h>
#include <pwd.h>

struct rule_t {
    int uid;
    int gid;
    const char *cmd;
    const char *path;
    const char *hash;
};

#include "config.h"

#ifdef HASH
#include "sha256.h"
#endif

// Help
// ----

static const char *HEADER = "sup " VERSION
    " - small and beautiful superuser tool\n";

static const char *COPYLEFT =
    "copyright (C) 2016 dyne.org foundation, license GNU
```

```
      GPL v3+\n"
    "this is free software: you are free to change and
      redistribute it\n"
    "for the latest sourcecode go to
      <https://git.devuan.org/jaromil/sup>\n";

static const char *LICENSE =
    "this source code is distributed in the hope that it
      will be useful,\n"
    "but without any warranty; without even the implied
      warranty of\n"
    "merchantability or fitness for a particular
      purpose.\n"
    "when in need please refer to
      <http://dyne.org/support>.\n";


static const char *HELP =
    "Syntax: sup [options] command [arguments...]\n"
    "\n"
    "Options:\n"
    " -l     list compiled-in authorizations and flags\n"
    " -u     set uid to this user name\n"
    " -g     set gid to this group name\n"
    " -d     fork command as background process
      (daemon)\n"
    " -p     saves pid of background process to file
      (daemon)\n"
    "\n"
    "Please report bugs to
      <https://git.devuan.org/jaromil/sup/issues>\n";

// maximum length of a command string
#define MAXCMD 512
// maximum length of a command full path string
#define MAXFILEPATH 4096

// Always return 1 on error, conforming to standard
   shell checks.
// Reason of error is described by stderr text before
```

```
   colon,
// extended reason can be provided or falls back to
   errno.
static int error(const char *code, const char *reason) {
    fprintf (stderr, "%s: %s\n",
             code? code : "",
             reason? reason : strerror (errno));
    exit(1);
}

// Check if binary is found in $PATH
// --------------------------------

static char *getpath(const char *str) {
    struct stat st;
    static char file[MAXFILEPATH];
    char *p, *path = getenv ("PATH");
    if (path)
        for (p = path; *p; p++) {
            if (*p==':' && (p>path&&*(p-1)!='\\')) {
                *p = 0;
                snprintf (file, sizeof (file)-1,
                          "%s/%s", path, str);
                if (!lstat (file, &st))
                    return file;
                *p = ':';
                path = p+1;
            }
        }
    return NULL;
}

// Compute SHA256 hash of binary
// -----------------------------

#ifdef HASH

#define CHUNK 1048576 // 1MiB
static uint32 getsha(const char *path, unsigned char
   *dest) {
```

```
static sha256_context sha;

unsigned char buf[CHUNK]; // 1 MiB
uint32 len, tot;
FILE *fd;

fd = fopen(path,"r");
if(!fd) error("fopen", "cannot read binary file");

sha256_starts(&sha);
clearerr(fd);
len=0;
tot=0;

do {
    // read chunk of data in binary file
    len = fread(buf, 1, CHUNK, fd);
    // stop if NULL read
    if(!len) break;
    tot+=len;
    // stop if EOF reached
    if(len<CHUNK) break;

    // compute sha256 of chunk
    sha256_update(&sha, buf, len);

} while(len>0);

// check file descriptor for errors
if(ferror(fd)) {
    fclose(fd);
    error("fread", "error reading binary file");
}
fclose(fd);

// compute last chunk
if(len>0) sha256_update(&sha, buf, len);

// finish and save result in *dest
sha256_finish(&sha, dest);
```

```
    return(tot);
}

#endif

// Main()
// ------

int main(int argc, char **argv) {

    static char fullcmd[MAXCMD];
    static char *cmd;

    struct passwd *pw;
    struct stat st;

    static int i, uid, gid;
    static int target_uid=0;
    static int target_gid=0;

#ifdef HASH
    unsigned char digest[32];
    char output[65];
#endif

#ifdef DAEMON
    int fork_daemon = 0;
    char pidfile[MAXFILEPATH] = "";
#endif

    // parse commandline options
    int opt;
    while((opt = getopt(argc, argv, "+hvdlu:g:p:")) !=
       -1) {

        switch(opt) {

#ifdef DAEMON
        case 'p':
```

```
            snprintf(pidfile,MAXFILEPATH,"%s",optarg);
            break;
#endif

        case 'u':
        {
            struct passwd *puid;
            errno=0;
            puid=getpwnam(optarg);
            if(!puid && errno)
               error("uid_getpwnam",NULL);
            if(puid) target_uid=puid->pw_uid;
        }
        break;

        case 'g':
        {
            struct passwd *pgid;
            errno=0;
            pgid=getpwnam(optarg);
            if(!pgid && errno)
               error("gid_getpwnam",NULL);
            if(pgid) target_gid=pgid->pw_gid;
        }
        break;

        case 'h':
            fprintf(stdout, "%s\n%s\n%s", HEADER,
               COPYLEFT, HELP);
            exit (0);

        case 'v':
            fprintf(stdout, "%s\n%s\n%s", HEADER,
               COPYLEFT, LICENSE);

            exit (0);

#ifdef DAEMON
        case 'd':
            fork_daemon=1;
```

```
            break;
#endif

        case 'l':
            fprintf(stdout,
                    "%s\n%s\nList of compiled in
                       authorizations:\n\n",
                    HEADER, COPYLEFT);
            fprintf(stdout,"User\tUID\tGID\t%s\t\t%s\n",
                    "Command","Forced PATH");
            for (i = 0; rules[i].cmd != NULL; i++) {
                // Using 'getpwuid' in statically linked
                   applications
                // requires at runtime the shared
                   libraries from the glibc
                // version used for linking. But not in
                   case of musl-libc.
                pw = getpwuid( rules[i].uid );
                fprintf (stdout, "%s\t%d\t%d\t%s\t%s\n",
                         pw?pw->pw_name:"",
                            rules[i].uid, rules[i].gid,
                         rules[i].cmd, rules[i].path);
#ifdef HASH
                fprintf(stdout, "sha256:
                   %s\n\n",rules[i].hash);
#endif
            }
            fprintf(stdout,"\nFlags: %s %s %s %s\n",
#ifdef HASH
                    HASH?"HASH":"",
#else
                    "",
#endif
#ifdef DAEMON
                    DAEMON?"DAEMON":"",
#else
                    "",
#endif
                    strlen(CHROOT)?"CHROOT":"",
                    strlen(CHRDIR)?"CHRDIR":"");
```

```
            exit (0);

        } // switch(opt)

    } // getopt

    // get the called UID and GID
    uid = getuid ();
    gid = getgid ();

    // copy the execv argument locally
    snprintf(fullcmd,MAXCMD,"%s",argv[optind]);
    // save a pointer to basename string in cmd
    cmd = basename(fullcmd);

    // get the username string from /etc/passwd
    pw = getpwuid( uid );
#ifdef DEBUG
    // one could maintain a log of calls here
    fprintf(stderr,"sup %s called by %s(%d) gid(%d)\n",
            cmd, pw?pw->pw_name:"", uid, gid);
#endif


    // Check that all rules match
    // --------------------------

    // loop over each rule
    for (i = 0; rules[i].cmd != NULL; i++) {

        // command is * or locked path matches
        if (*rules[i].cmd == '*' || !strcmp (cmd,
           rules[i].cmd)) {
            // if path is locked
            if (*rules[i].path != '*') {
                // and if path is specified
                if((fullcmd[0]=='.')||(fullcmd[0]=='/'))
                   {
                    // then check that path matches
                    if( strcmp(rules[i].path,fullcmd) )
```

```
                    return error("path","path not
                        matching");
                // or if path is not specified
            } else {
                // get the default path with our
                    getpath()
                snprintf(fullcmd,MAXCMD,"%s",getpath(cmd));
                // check if the default environment
                    path matches
                if( strcmp(rules[i].path,fullcmd) )
                    return error("path","path not
                        matching");
            }
            // or if path is not locked
        } else
            // and if path is not specified,
                getpath()
            if((fullcmd[0]!='.')&&(fullcmd[0]!='/'))
                snprintf(fullcmd,MAXCMD,"%s",getpath(cmd));

#ifdef DEBUG
        fprintf(stderr,"path check passed\n");
        fprintf(stderr,"fullcmd: %s\n",fullcmd);
        fprintf(stderr,"cmd: %s\n",cmd);
#endif

        // Command binary check
        // --------------------

        // command does not exist as binary on the
            filesystem
        if (lstat (fullcmd, &st) == -1)
            return error("lstat", "cannot stat
                program");

        if (st.st_mode & 0022)
            // command has wrong permissions
                (writable to others)
            return error("perm",
                        "cannot run binaries others
```

```
                                can write.");

            // user UID is not root
            if (uid != SETUID
                // and is not unlocked
                && rules[i].uid != -1
                // and is not the locked UID
                && rules[i].uid != uid)
                return error("uid", "user does not
                   match");

            // user GID is not root
            if (gid != SETGID
                // and is not unlocked
                && rules[i].gid != -1
                // and is not the locked GID
                && rules[i].gid != gid)
                return error("gid", "group id does not
                   match");


#ifdef HASH
            // Binary hash checksum
            // --------------------
            if( strlen(rules[i].hash) ) {
                int c;
                uint32 sizeread;

                sizeread = getsha(fullcmd, digest);
                if(sizeread != st.st_size)
                    error("getsha",
                          "binary file size differs from
                             size read");

                for(c = 0; c<32; c++)
                    sprintf(output + (c *
                       2),"%02x",digest[c]);
                output[64] = '\0';

                if(strncmp(rules[i].hash, output,
```

```
                    64)!=0) {
                        fprintf(stderr,"%s\n%s\n",
                                rules[i].hash, output);
                        return error("hash", "hash does not
                           match");
                    }
                }
#endif

                // Green light to privilege escalation
                // -----------------------------------
                if (setuid (target_uid) <0)
                    return error("setuid",NULL);
                if (setgid (target_gid) <0)
                    return error("setgid",NULL);
                if (seteuid (target_uid) <0)
                    return error("seteuid",NULL);
                if (setegid (target_gid) <0)
                    return error("setegid",NULL);

#ifdef CHROOT
                if (*CHROOT && (target_uid==0))
                    if (chdir (CHROOT) == -1 || chroot (".")
                       == -1)
                        return error("chroot", NULL);
                if (*CHRDIR)
                    if (chdir (CHRDIR) == -1)
                        return error("chdir", NULL);
#endif

                // Fork as daemon if desired
                // -------------------------

#ifdef DAEMON
                if(fork_daemon) {

                    pid_t pid;
                    pid = fork();
                    if(pid<0) return error("fork", NULL);
```

```
                // child
                else if(pid==0) {

                    int fd = open("/dev/tty", O_RDWR);
                    ioctl(fd, TIOCNOTTY, 0);
                    close(fd);
                    chdir("/");
                    // secure default
                    umask(022);
                    // process group
                    setpgid(0,0);
                    // stdin
                    fd=open("/dev/null", O_RDWR);
                    // stdout
                    dup(fd);
                    // stderr
                    dup(fd);

                    // Execute in foreground
                    // ---------------------
                } else {

                    // if pidfile is not an empty string
                       (-p is used)
                    if( strncmp(pidfile,"",MAXFILEPATH)
                       ) {
                        // save the pid of the forked
                           child. beware this
                        // does not work with some
                           daemons that follow up
                        // with more forks.
                        FILE *fpid = fopen(pidfile,"w");
                        if(!fpid) error("pidfile", NULL);
                        fprintf(fpid,"%u\n",pid);
                        fclose(fpid);
                    }

                    // leave us kids alone
                    _exit(0);
                }
```

```
            }
#endif

            // turn current process into the execution
               of command
            execv (fullcmd, &argv[optind]);
            // execv returns only on errors
            error("execv", NULL);

        }
    }

    // be polite
    fprintf(stderr,"Sorry.\n");
    exit(1);
}
```

8 Gufo Volante: live coding for data analysis

Table 9: Information on Gufo Volante live coding project

Project name	Gufo Volante
Home online	https://github.com/d-cent/gufovolante
Started in	2016
Nature	Software
Advancement	Practice based

"Gufo Volante", meaning "flying owl" in Italian language. The immediate application of this project is to operate spending analysis over large amounts of financial data on the national, regional and provincial budgeting of the Italian government. While this is a sort of pretext that is finding already some interesting uses and traction among the Italian open data community, the motivation to start writing Gufo Volante goes well beyond its immediate and most apparent function.

The main motivation to write Gufo Volante and include it in this thesis is that of demonstrating the practical implications and advantages of hybridating the live-coding approach with literate programming in the context of big data analysis. Live-coding (Collins, 2011) is an digital art performance practice, approximately 15 years old, where the code together with its result are contextualised and shown on stage (McLean, 2014): the code is adjusted by the performer and the results of any change are immediately visible or audible (Bovermann and Griffiths, 2014). The importance of live-coding for the comprehension and popularisation of algorithmic literacy is enormous, as it directly confronts the audience with computer programming by demonstrating the actions of performing programmers and the consequences of their actions. The cause-effect perception when learning programming is very important. Yet the function of live-programming is not only pedagogical, as even for seasoned programmers the definition of a "developer's user experience" mostly consists in the facilitation of rapid execution of changes and feedback on their consequences.

My own passion for this form of live performance after years of practicing with the theatre company Giardini Pensili (Monteverdi and Pino, 2011) has never really left me. While designing and developing new software, even when not for artistic purposes, I've noticed is always of great advantage the inclusion of a so called REPL (read-eval-print loop), a live console element common to LISP languages, which also facilitates the implementation of unit testing (Nørmark, 2010) and verification of code results.

Going across this research it became almost natural to think of the conjugation of live-coding with one of the biggest lessons learned, that of adopting various degrees of literate programming not just for its technical values, but also for the socio/political

importance of making code understandable and readable. The challenge at hand then was the last piece missing and, considering the importance and difficulty of big data analysis and the relevance of its results when taken into account for policy making, I took the initiative to write Gufo Volante the last and most recent software presented in this thesis. This software is a framework for of live-coding on big data which facilitates the rapid testing and the documentation of the algorithms implemented with it. It is written in Clojure (Emerick et al., 2012), a dialect of LISP which well implements most characteristics needed while being practical and complete of appropriate tooling. Taking advantage of the R framework for statistics, the live REPL implementation called Gorilla running in a browser and of the data published by the initiatives "SIOPE" and "Soldipubblici" supported by the Italian Government, I've realised a proof of concept of what may be a promising development to leverage the forensic quality of spending analysis.

Here I mention forensic quality concerned by how important is, in the field of big data analysis, the selection of indicators from different sources and the documentation of the processing they go through, in addition to the results. In fact, one may well argue, the documentation of the process is crucial to prove the results, or in other words the algorithm must be manifest and readable. Furthermore if an algorithm is presented in a setup of live-coding, so to say in a live data analysis framework, then it can also be adjusted and actualised, staying useful as a source of analysis.

This approach is fundamentally different from that of a spreadsheet, or a graphical presentation in the form of charts, because it shows the flow of transformations and allows to change them in any place with an intervention on the code which will show immediately the result of any adjustment. In addition to that, the functional, stateless and non imperative characteristics of Clojure encourage programmers to be concise and to break their code in clearly isolated steps, which can be then easily documented and recombined.

A last notable characteristic of Gufo Volante is the fact that its code is entirely translated to Italian, with all its functions, as a challenge to the malleability of LISP dialects and the possibility to easily redefine functions. Readability of code is an important aspect for the theoretical challenge behind the realisation of this software: on these grounds the barriers put by different human languages should not be overlooked, as they also imply different configurations for the sovereignty that algorithms offer.

The result is a very concise flow of code capable of retrieving data sets from CSV and REST APIs of a growing number of on-line open data services and cross them for spending analysis. While progressing in the analysis, at any the results can be saved into an HTML file or a formula to be replayed using the software. Here below a brief example session retrieving all the public expenses of the province where I was born, Pescara, ordering them by magnitude, excluding the voices below a million of

expenditure and rendering a readable bar chart.

```
(def dati (raccogli-dati "PRO" "000016324" "Comune di
   Pescara"))

(def rilievo (analizza-dati dati))

(let [interessante   (ordina-analisi {:2015 [> 100000]}
   rilievo)]
      (visualizza-tavola interessante))

(bar-chart :desc
           [:2016 :2015 :2014] {:flip? true :height 12}
           (where {:2015 [> 100000]}
                  (rest rilievo)))
```

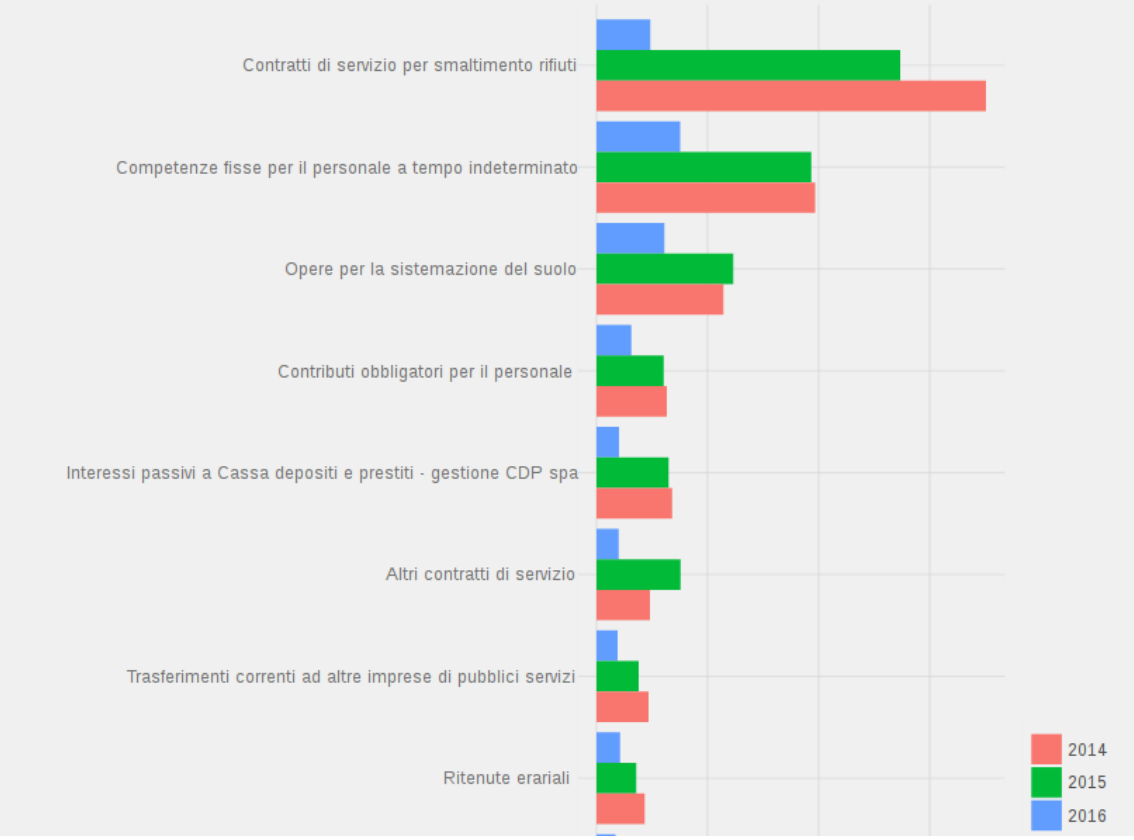

Figure 25: Bar graph resulting from the code above

Code and graphics appear in a cascade flow on the same page, allowing to explain with a graphic every single step of analysis and continuing to the next, with readable code in between. By exporting the whole result it is possible both to show and verify the computation done.

When we consider algorithms as contracts regulating relationships (between humans, between humans and nature and, nowadays more increasingly, between

different contexts of nature itself) then we should adopt a representation that is close to how the human mind works and that is directly connected to the language adopted. In this thesis I interpret algorithms as the systemic product of complex relationships between contracts and relevant choices made by standing actors (Monico, 2014). The ability to verify which algorithms are in place for a certain result to be visualised, to understand and communicate what these algorithms do, to describe and experiment their repercussions on reality is in fact conditioning the very choices standing actors will make. This ability is conditioned by the quality of computer languages, their possibility to provide a good "developer experience" when experimenting with them and their proximity to the way a human mind thinks.

At the time of writing the use of Gufo Volante has already triggered some attention for public sector use and in the field of data journalism. For this thesis, it serves to sketch a practical vector of experimentation and development that incorporates many of the lessons learned, directly showing the advantages of having code that is documented and in view for every resulting step, as well algorithms that can be adjusted live and whose value go well beyond the performative context where live-coding was initially born.

9 Decentralised Citizens Owned Data Ecosystem

Table 10: Information about the DECODE project

Project name	DECODE
Home online	https://decodeproject.eu
Started in	2016
Nature	Research and development
Advancement	Practice based

The DECODE project has been written during the early months of 2016 by a trans-disciplinary research and development group lead by senior project developer Dr. Francesca Bria, I've been involved in the role of technical architect together with Dr. George Denezis of the University College London and many other colleagues. In fact, DECODE brought together institutions regarded as centres of excellence for work on the Blockchain technology and it focused on the social perspectives of labour and so called sharing-economy patterns emerging. Here below the list or participating institutions:

- The National Endowment for Science, Technology and Arts in UK
- The Dyne.org foundation, Think/do tank and software development organisation
- The software company Thoughtworks, international excellence in Agile methodology
- The University College of London, department of blockchain studies
- The Catholic University Radboud in Nijmegen, The Netherlands
- The Technical University and NEXA centre for Internet & Society in Turin, Italy
- The Open University and Technology Centre of Catalonia
- The National Centre for Scientific Research of France
- The startup Thingful limited inc, specialising in IoT semantic analysis
- The EURECAT institute for statistical and data studies of Catalunya
- The Arduino company, makers of open popular hardware platforms
- The Municipality of Amsterdam
- The Municipality of Barcelona

Some of this research, as the partner composition shows, has inherited the outcomes and horizons offered by results of the D-CENT project, previously described in this thesis. Yet in DECODE the focus moved from engagement to data ownership, clearly addressing economic issues more than political participation. The abstract we wrote for the project recites:

Today's Internet is becoming increasingly centralised, slowing innovation

> and challenging its potential to revolutionise society and the economy in a pluralistic manner. DECODE will develop practical alternatives through the creation, evaluation and demonstration of a distributed and open architecture for managing online identity, personal and other data, and collective governance in a citizen-friendly and privacy-aware fashion. Strong digital rights that makes it possible for data subjects to determine access rights to their information through flexible entitlements and open standard-based agreements regarding data governance (on the model of Creative Commons licenses) will be woven into the technological architecture.

the DECODE project plan addressed techno-political concerns about the rapid expansion of companies of the so-called "sharing economy", like Uber or AirBnb, in ways that are directly and obviously affecting the administration, legal and economic framework of city-wide and state-wide sovereignty. In recent times one can witness a transformation of such power structures, in which the administrative, legal and economic frameworks do not originate nor they steer the changes operated by the algorithms and in fact are directly subjugated by them.

Within the process of DECODE's project definition I've contributed reasons and perspective outcomes for a research and development activity focusing on the importance of language and not just data, more in general bringing forward concerns on the possibilities to leverage algorithmic awareness and governance out of the small circles of expertise or the oligopolies of corporations that usually deal with them. While these concerns resonated in the interest of some of the partners, it has been fairly difficult to make it clear to the most, a process that made me aware of how unbalanced is the presence of literature on the topic of "data ownership" when compared to algorithmic sovereignty. That considered, the DECODE project results as a practical application of the findings of this research and awareness gained on the role of algorithms that should not be perceived as ancillary to data, but rather as a layer of inference and manipulation that is extremely more problematic and politicised than data itself and its ownership.

> The DECODE project develops a distributed and privacy aware architecture for decentralised data governance and federated identities. The platform employs state of the art technology that is intrinsically resistant to malware and hacking, and adopts Attribute Based Cryptography (ABC) to sensitive information, ideas from Certificate Transparency (CT) for identification, and Blockchain Technology for federated data access, distribution and resilience.

Here a brief explanation of terms: Attribute-Based Cryptography (Batina et al., 2010) is a technique that allows the extraction of a limited subset of data from a storage containing more of it, as for example extracting only the birth-date from a chip-on-card that contains more personal data like name, address and occupation. Certificate Transparency (Potzmader et al., 2013) (Toledo et al., 2016) refers to

a system where the identity of participants releasing a certificate of authenticity is known to everyone, based on a knowledge that is built structurally and to give confidence that the privacy protection applied on data shared using a particular certificate is readable only by the identified recipient. Both the ABC and the CT concepts belong to the domain of applied cryptography and deal with data protection and certification.

> The DECODE architecture is fully decentralised, and allows for a flexible and extensible data governance that can be applied to different regimes of data ownership and privacy. This flexibility is implemented through a set of algorithmic protocols called smart rules. Smart rules allow the data owners to define rules of management related to access, value attribution and other parameters. Smart rules will define access to subsets of data (for instance personal data) for specific use(s) granted to specific subject(s), according to a defined ontology.

> DECODE allows the consensual use of personal data in anonymous form for personalised services and applications that may be authorised to manage it. This mechanism defines the rules for data flow among any type of data sets or indexes provided by data aggregators. Once the access is granted the data can be processed by services subscribed to its feed and entitled to access it. It can be collected by any of these services or IoT applications. The DECODE platform aims to integrate the technical, economical and legal frameworks, managing common data, public data and personal data. The smart rules will be released under a Free and Open Source Software (FOSS) license (as will all the platform specifications, protocols, ontology, semantic specifications), and gradually extended with the participation of the community whenever new needs arise, aiming at becoming the standard language for managing data access and valorisation in distributed and decentralised architectures.

> Furthermore, DECODE targets external developers, researchers and social entrepreneurs to build applications on top of the distributed platform through bottom)up participatory mechanisms that aim to transform the current centralised data economy models, putting agency in the hands of citizens and innovators.

Governamentability requires the applying rules for the transmission and transformation of data, we can see how algorithms become an important part of the equation as "flexibility is implemented through a set of algorithmic protocols called smart rules". We can translate the term flexibility here with participation and interoperability. Ultimately in DECODE we call "Smart Rules" a computable (Sober, 1978) sociolect (Louwerse, 2004) that can be matched into a semantic model and executed by a distributed computing cluster. It is then of central importance to grant citizens the access to such a language according the circular economic phases of *Creation*, *Sharing*,

Distribution and *Appropriation* (according to the Semantic Square of Creation). It is important when we think of prominent phenomenons of "sharing economies" where in fact the rules are unknown, opaque and tyrannically adjusted by third parties who are not participating in the economy but in fact just profiting from it, situations that are ethically unsustainable and have no regard for the algorithmic sovereignty concepts this thesis makes explicit. The stated project vision in DECODE goes more into the specifics of these issues:

> The DECODE platform aims to create a level playing field for European innovators and social entrepreneurs in order to fully grasp the advantages of the digital transition associated with the Internet of Things, by developing a distributed, privacy-aware, and trusted architecture for decntralise data governance and identity management.
>
> European SMEs[19], developers and social innovators currently use open hardware, open source software, open knowledge, data storage and analytics to produce valuable data about people, the environment, and biometric and sensor data (see for instance, the emerging European DSI movement and the CAPS ecosystem of projects). But the resulting data is largely unavailable for the public good, since it rests in centralised commercial platforms, creating an oligopoly on who can mine and analyse it for profit. As a result, this data becomes inaccessible for a more innovative and inclusive economy.
>
> The lack of trusted solutions for decentralised data and identity management is putting severe limits on the capacity of small organisations to innovate. Sensitive and valuable data are either 'stolen' by applications, or not collected and used at all. By building and providing a technical solution and open standards that foster trust by design, SME's, social enterprises, industry, researchers, communities and individuals can start to negotiate a way to tap into the most sensitive and valuable data on fair terms, providing nearly unlimited numbers of new services. DECODE posits that decentralised architectures with open APIs are essential to build a smart Europe for all its citizens, intrinsically respectful of citizens' privacy and digital sovereignty.
>
> Citizen centric approaches to the digital economy still lack enough real added value for citizens. For example, in smart cities, the data of the connected car is not currently in the hands of the driver but in data silos controlled by manufacturers and insurance companies. There is no interoperability and lack of open protocols in the smart home space, undermining the vision of a democratic and open digital society. DECODE accelerates both

[19]Simple and Medium Enterprises, indicating small players as opposed to multi-national corporations, whose development and sustainability is high in the policy agenda of many countries for the resilience they provide to the economy.

the understanding, effective use, and adoption of large scale distributed architectures and standards to support citizen empowerment and citizen ownership of data, unlocking their potential by enabling any community platform to make the most of 'network effects', so that it can:

- Prevent the concentration of power in the hands of a few platform operators that aggregate big data on a global scale through a fully decentralised architecture for identity management, data storage, processing and governance.
- Represent and effectively use an extended range of data, coming from people, sensors, devices and the city; deploying privacy aware intelligent systems that help to make citizens more aware of issues and possible social solutions; fostering collective deliberation and bottom up decision making.
- Ensure that people are in full control of their data and identity, maintaining privacy and trust in the systems they use, reaping the benefits of big data aggregation, within a clear socio technical and legal framework, enhancing privacy and ownership of data.
- Create a level playing field enabling new entrants (social entrepreneurs, P2P developers, open source software and open hardware developers) to implement innovative approaches and applications, opening up new economic and social perspectives.
- Preserve the digital sovereignty of European (and worldwide) citizens, preventing unauthorised usage of their personal data, on clouds, social networks and the Internet of Things.

The next sections proceed presenting a deeper analysis of the DECODE project intentions and methodology, highlighting the relation to findings presented in this thesis. DECODE has been proposed to the European Commission for financing within the Horizon 2020 research framework for the program "Collective Awareness Platforms for Sustainability and Social Innovation" (CAPS) managed by the European Commission Directorate General for Communications Networks, Content & Technology (DG CONNECT). Right during the time of writing thesis news have reached me that DECODE has been successfully awarded a grant of approximately five million EUR to proceed on its intentions, works will start in 2016/Q4.

9.1 Concept and economic analysis in DECODE

The concerns we put forward in DECODE are clearly related to the theme of sovereignty which is in fact explicit in the use use of the term "digital sovereignty" which relates to a broader concept of data and algorithmic sovereignty. It is useful to distinguish between the nature of the two and in the text that follows with an explanation that goes into the economic, social and political details of the analysis behind the project all references to "data sovereignty and governance" can be interpreted as implying algorithmic sovereignty in the exercise of governmentality on data. The

following paragraphs are an important excerpt from the project plan to explain this concept:

> The current digital ecosystem and IoT landscape is highly fragmented, with a multitude of non interoperable vertical solutions, all offering their own set of devices, gateways and platforms, and means of data handling in data "silos". This fragmentation makes data unmanageable and end users ultimately lose control over it. This status quo arises because small SMEs, startups and other innovators cannot see a clear value proposition in offering open, horizontal, interoperable components and data driven solutions. The cost of engineering such solutions from scratch makes them unaffordable.
>
> [...]
>
> Yet, one key reason cities and municipalities have so far failed to foster local alternatives to Uber or Airbnb is missing access to raw data. One of the aims for DECODE is to demonstrate local open and decentralised data platforms, where people can use contextual data to guide meaningful decisions and actions. We aim to demonstrate how granular data generated in the context of, for example, your smart home, or using your smart watch, can be aggregated with the data of your neighbours and fellow citizens. DECODE will explore how to build a knowledge economy that is data centric but where data that is generated and gathered by citizens, IOT, sensors networks and open city level data, and is available for broader communal use - with appropriate privacy protections. As a result, a mass of innovators, startups, NGOs, cooperatives, and local communities can take advantage of that data to build apps and services that are most relevant to them and the wider community.

Itis important to note the analysis clears up the specific issue of certain companies, namely Uber and AirBnb, which have marketed a closed algorithm which is not only opaque and proprietary, but also violating territorial laws with its execution. This explains the early reference made to the toxicity of so called "sharing economies" which are in fact depriving the algorithmic sovereignty of established societies, with consequent impoverishment of their knowledge commons (Vercellone et al., 2015) and resources to maintain a shared infrastructure on which local societies rely. An easy example is given not only by the lack of tax payments made by such companies, but also by the fact that for instance a company like Uber, running a shadow network of taxi cabs through the city, hasn't even included into its own agenda any investment on the maintenance of streets. Officials of the cities with whom we have been in touch for the conceptualisation of DECODE, Amsterdam and Barcelona, very advanced European cities recently awarded with the price of "Cities of the Future", have explicitly put forward concerns and illustrated their difficulties with dealing with such situations without curbing the possibilities for innovation offered by a generic visions of sharing economy models.

The approach of DECODE is that of matching the concept of algorithmic sovereignty with that of territorial sovereignty in the framework of governance, with the intention of opening up neutral and fair grounds to curate, maintain and allow the responsible exploitation of immaterial commons.

9.2 Interdisciplinary methodology to bootstrap a privacy aware, decentralised innovation ecosystem

In DECODE we address directly the methodology adopted in this thesis, for which this research is both an advancement and a practice based experimentation. In this passage of the project we illustrate briefly the interdisciplinary relation to be established across multiple areas of expertise (most prominently in DECODE are legal, economic and technologic analysis) and we indicate also with a graphical representation the commitment to a LEAN cycle of experimentation that is iterated through the whole project. This is in fact a genuine application of the methodology described in this thesis.

> A interdisciplinary effort including technological, societal, business, and legal aspects will enable DECODE to unlock the full potential of the digital platform economy. The main premise of DECODE is to provide value for the data subjects and citizens themselves rather than the service provider only. DECODE addresses the challenge that user data is spread in silos, and that users have a hard time tracking and managing where it is and how it is used. DECODE focuses research and development effort on novel notions of trust and privacy (privacies) that can be operationalised in new governance frameworks, and innovative economic models based on data commons. Entitlements attached to the private data would be searchable in the public domain but will grant access only to those parties that have the entitlement to access it. This novel concept of data rights and entitlements also applies to data being sent to or consumed by connected IoT objects in order to perform actions on the real world, allowing citizens to manage and control their devices and the data they generate.

> Finally, an important part of DECODE will be public education and the creation of collective awareness around data ownership, digital identities, control and privacy. We think that the only way to put these ethics on the agenda is by bringing the subject of data collection from the proverbial, invisible cloud - as far away as possible from users' daily concerns - to their actual everyday. DECODE will experiment in a collective way to move away from the model of centralised data platforms towards a social and solidarity driven model that exploits technology driven changes for collective benefit. We are convinced that this can be done in a way that is fully respectful of European citizens' privacy and digital sovereignty - all in order to create the data reservoir on top of which all sorts of applications and innovative challengers, like Uber and Airbnb, can spring up and blossom

across Europe.

The work is divided into two major blocks: - Interdisciplinary research activities regarding the development of the DECODE architecture, including the legal, economic and policy frameworks on data commons, privacy and ethics; - Demonstration in four pilots that will bring added value to the partners and be coupled with participatory innovation mechanisms that allow new entrants to provide data services and innovate business models.

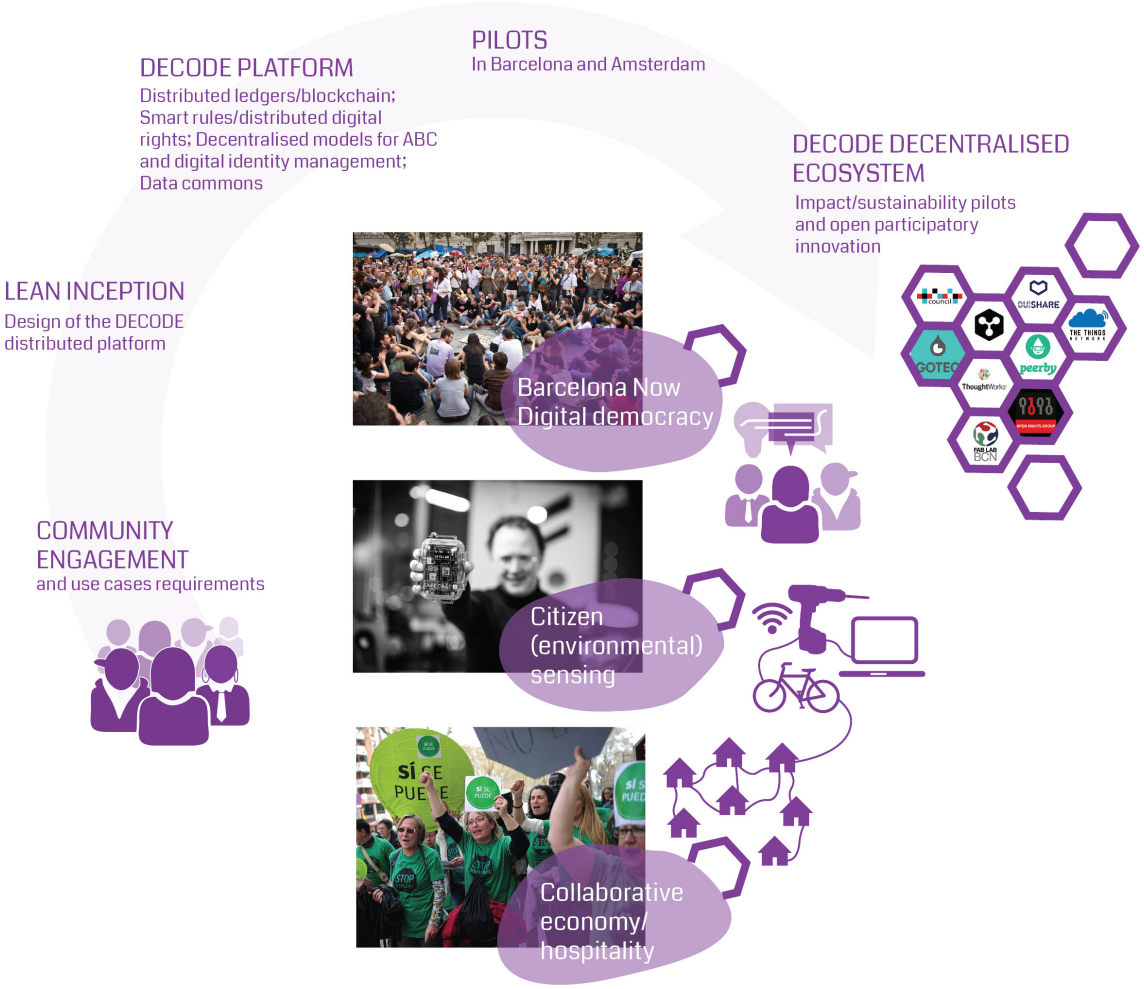

Figure 26: Illustration of DECODE methodology of LEAN development cycle in relationship to city pilots

The development of the DECODE components will be done in parallel with the development and implementation of the use cases, and open challenges follow a Lean and Agile methodology that allow for shorter iterative cycles, that are highly effective to manage high uncertainty. This way researchers may receive early feedback on their results, highlighting unanticipated issues earlier, so they can prototype and deliver solutions faster. A growing number of platform components will become available after the first iteration. Subsequently, the use case scope will be expanded, and the complete platform validated against complex scenarios. These will represent real life requirements, and will be open for external social entrepreneurs and

innovators to implement their ideas and develop disruptive applications.

The main focus points of DECODE methodology are:

Agile development (Beck et al., 2001) (Martin, 2003): The first overall design of the DECODE platform architecture is planned within the first year of the project. A second iteration is planned to take place after the inputs from the technical work packages and technical requirements from use cases. Two iterative phases of developments will then take place. First, the individual components will be implemented, and the first integrated prototype of the platform will be delivered. The platform will be evaluated through the deployment and demonstration of the use case scenarios which will provide an initial evaluation to inform the final version of the platform. At the end of the second phase, all components will become available and the platform will be validated upon real%life scenarios.

Bottom up demonstration activities and social impact experiments: Demonstration activities with strong social impact are a central part of the DECODE project. These are end to end technical component integration demonstrations and field trials based on the use cases allowing us to monitor and evaluate progress against key project indicators. Starting from the use case definition, the trials progress in three phases: prototype, large scale deployment and validation. To ensure the best results, the city of Barcelona and the Waag Society will lead the coordination of the trials, fully supporting the deployment phases.

Open standardisation activities: Partners will be active in a number of standardisation groups. During the first year, the project will identify new opportunities for promoting the DECODE results in upcoming activities, standards and other public reports. Partners will draft a standardisation road-map to guide and focus standardisation efforts. In the last two years we will focus on those particular actions..

End users (cities, citizens, SMEs, social entrepreneurs, P2P, open source, open hardware developers) co-design and involvement: Citizen engagement is based on bottom up participatory approaches to ecosystem creation, involving various stakeholders. Ecosystem building will use a variety of tools including focus groups, hackathons, co creation and co design workshops and gamification for service engagement, etc. Focusing on citizen driven innovation, supported by cities, local businesses and developer ecosystem, and the Digital Social Innovation Community (including HUBS, FabLabs, Hackerspaces, accelerators etc) forms the heart of the DECODE approach. This engagement will help us navigate privacy and ethical considerations - which can be loosely grouped under the heading of "trust". By building a trust framework into the design process, the project will ensure citizen engagement and digital sovereignty in the complete design process.

Disruptive (for example commons based) economic models, validation and new decentralised data governance models: Three Tasks in the project

are targeted at prototyping innovative and disruptive business models based on data commons, and focused to target the sustainability of the project, and the related disruptive business model of the ecosystem based on common, privacy and ecological sustainability. In particular, DECODE will: 1. Identify and track key disruptive business model hypotheses interlinked with the juridical framework of DECODE. 2. Test and validate the hypotheses with developers of platform and services. 3. Update DECODE commons based business model according to the feedback of citizens, developers, and cities. Enable new partners to build data driven and privacy preserving applications leveraging the DECODE platform.

Gender issues: Depending of the field trials, gender specific aspects could be one of the important contextual factors driving citizens' engagement. Therefore, during the project we will ensure that as many women as men are involved in the design process of use cases and also participate during the field trials and surveys analysis. The DECODE Consortium is also very aware of the need to balance the gender dimension and encourage women participation in Science and Technology research and activities.

9.3 Scientific and technological advances beyond the state of the art

Among its objectives, DECODE clearly lists the development of a language for "smart-rules" that is not conceived to stay behind the scenes, but to be understood and modified:

> DECODE will contribute to formulating and standardising a different approach to data management rather than CRUD-type interaction, necessitated by the distributed and write once immutable nature of blockchain technologies. It will propose a pragmatic and workable use of semantic standards for description, an innovative scheme for granular access to private information on blockchains, annotation and API[20] interfacing, including languages for expressing smart contracts relating to privacy and digital commons.

This language is conceived as a trust framework which is both usable (can be executed) and expressive (can be understood) to encode smart rules and contracts relating to the governance of personal and other data in the DECODE architecture. This is an important, practical formulation of objectives that resonates with the findings of this thesis. The possibility to understand and not only to execute a language is of paramount importance to allow algorithmic sovereignty. Algorithms are law in their own domain (Lessig, 1999) and just with a society of people, participants to

[20] Application Protocol Interface, the basic map for points of interaction with a software which is running to serve responses to requests.

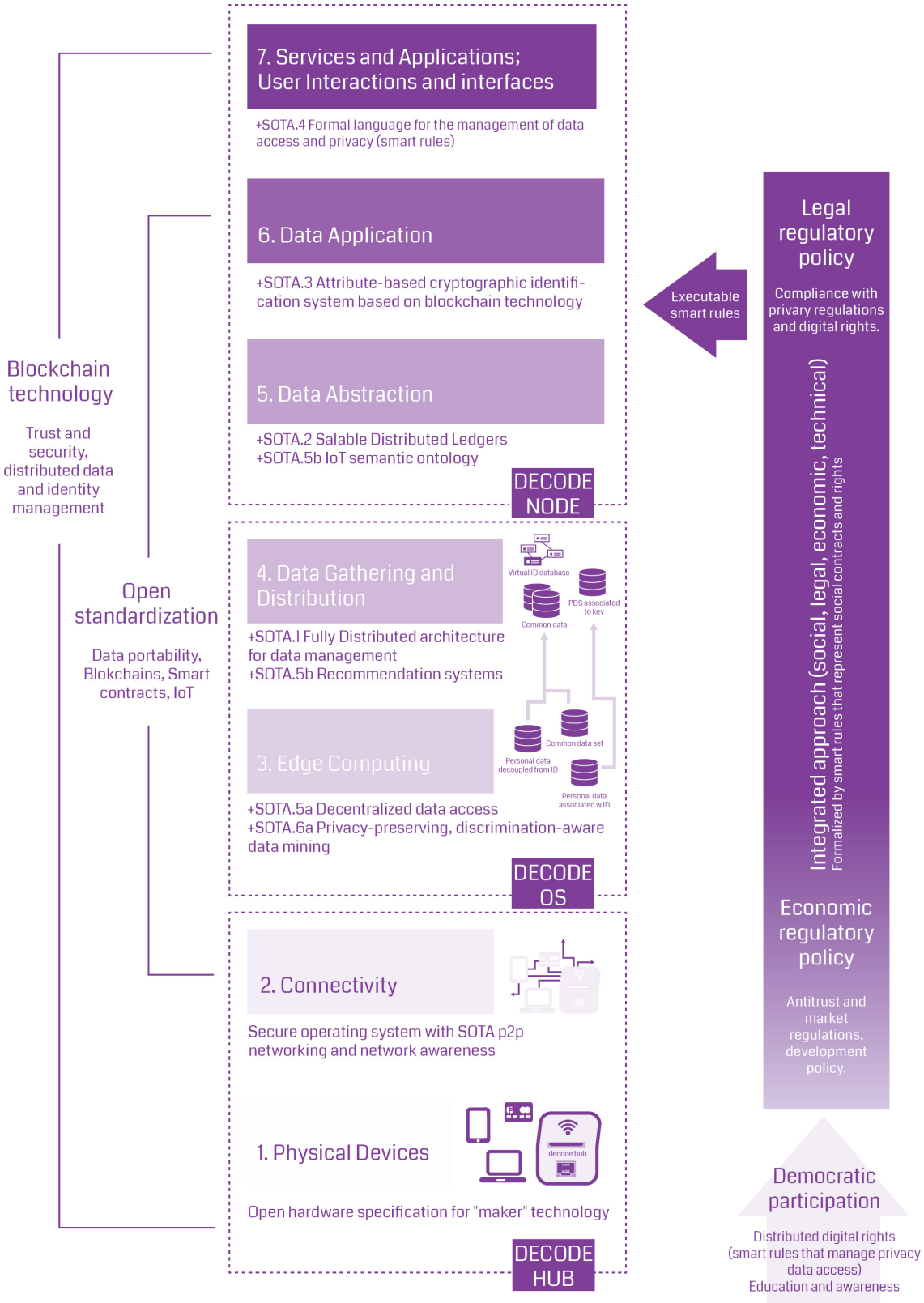

Figure 27: The technical architecture of DECODE explained on the ISO/OSI 7 layers model

an algorithmic system can never manage to establish sovereignty without the steps of *appropriation*, *creation*, *sharing* and *distribution* of the laws governing it. This is why in DECODE the concept of a language that can be a subject for critique and transformation according to societal needs is emphasised.

In the figure illustrating DECODE's technical architecture on the ISO/OSI 7 layers model this is made visible by the prominent presence of the right column whose point of integration is between layer 5 (data abstraction) and layer 6 (data application). The literature of both legal and economic regulatory policy is both usable and expressive, its source is "democratic participation" represented as distributed digital rights implemented via "smart rules that manage privacy data access" and, last but not least, "Education and Awareness" are indicated prominently as a foundation of this process. The compliance with privacy regulations and digital rights, the respect for antitrust, market regulations and development policy, all converge in an integrated approach (social, legal, economic, technical) which is formalised by smart rules that represent social contracts and rights.

Here becomes clear the central importance of a "Formal language for the management of data access and privacy (smart rules)": a language is needed to define, implement and enforce clear and transparent smart rules for data access, governance and identity management. Access rules should allow organisation to use data, depending on specific parameters; capture and enforce legal rules and constraints, as well as rules around anonymisation obligations and data deletion or retention. Language rules must be enforced by design, on either centralised or decentralised datasets. Their enforcement should be safe from platform attacks (through hacking or malware). Language design should promote clarity and avoid ambiguities; their syntax should be extended according to the need that will emerge. The rules should ensure the respect of the decisions of the data owner (for example, citizen or the city), but also of collective regulations and legal obligations on application providers.

The data management and policy enforcement according to rules is defined by the user interacting directly with rules or with a facilitate abstraction of them, graphically representing their flow. Only the data owner can change the rules, since they are stored on a distributed blockchain in the DECODE node network. When new needs from users gradually emerge, and the legislation becomes more complex, different kinds of rights will be addressed by extending the expressive power of the rules through a participatory process. The data management should respect the decisions of the data owner (for example citizen or the city) and also respect collective decisions, regulations and legal obligations, adding general rules, which include a collective dimension to protect communities and groups. This visionary approach will consolidate through the project and has to be considered a plan as it is sketched here, yet it is important how the design aspirations have already adopted the concept of algorithmic sovereignty as an underpinning for the scientific and technological

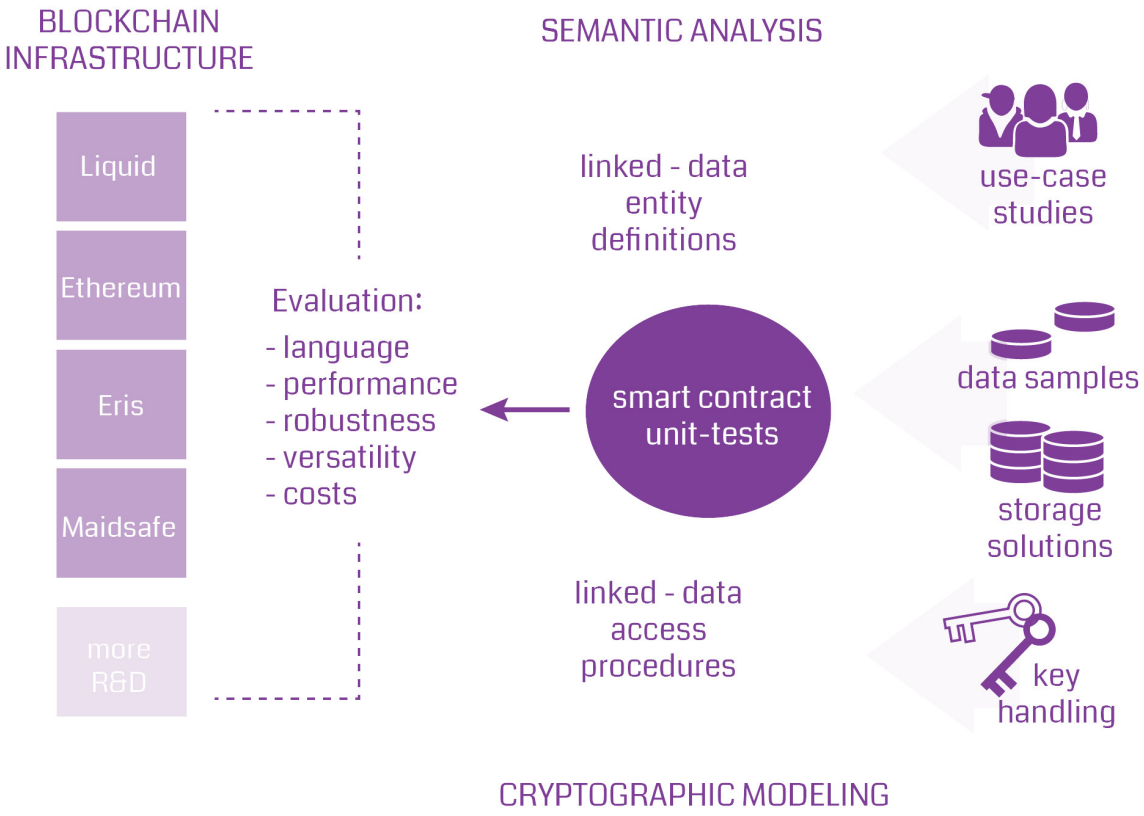

Figure 28: The research and development on distributed ledger technology in DECODE focuses not only around its cryptographic model, but also on the semantic analysis of language

advancements offered by DECODE.

9.4 DECODE, context and perspectives

To facilitate *Participation*, multiple implementations, wide adoption and *Interoperability*, DECODE will develop a smart-rule language and execution engine. All read and write operations affecting entitlements and accessing data can be expressed in this language, which we will design to become a robust open standard for authorisation around personal data. Such a language will aim to be data-centric(Qin et al., 2016) and functional, to naturally avoid complex constructions and define sets of transformations that can be then easily represented with visual metaphors.

The DECODE project has vast implications and its architectures goes obviously beyond the research and development of a language, yet in the context of this thesis is important to bring that into relevance, along with the methodology adopted which informs the language specification by a closer contact with real use-cases, situations, processes, desires and needs expressed by the living context.

The horizon of development of smart rules expressing the costrains and/or the requests of the users or of the data owners constitutes a perfect continuation for my quest to document even deeper the points of contact between governance and

algorithms. What I prospect already is that an ideal language for these smart-rules would be a non-imperative language like a dialect of LISP that should leverage the REPL approach in the direction of usability. LISP dialects hold some crucial characteristics for the design of such an executable language and makes it possible to encourage programming styles that are non-imperative[21], functional[22] and stateless. The language should be completely open to appropriation, translatable across human languages and particularly designed to facilitate the sort of publish/subscribe event driven architecture of blockchain technology.

This project is a direct outcome of the application of conclusions drawn in this thesis and as such it concretely represents an advancement informed by this thesis. The process of sharing reflections about algorithmic sovereignty with colleagues, as well conducting interviews with city government officials, was full of confirmations about the assumptions and conclusions presented in this thesis. This validity of such assumptions and conclusions is proven by the central role that language design plays in DECODE, as well by the recent expression of interest the European Commission Directorate General for Communications Networks, Content & Technology demonstrates by funding it.

[21]Non-imperative programming languages describe their desired results without explicitly listing commands or steps that must be performed. Functional and logical programming languages are characterized by a declarative programming style.

[22]In functional code, the output value of a function depends only on the arguments that are input to the function. Changes in state do not depend on the function inputs and make it easier to understand and predict the behaviour of the program.

10 Conclusion

This research has been a very engaging journey due to the active participation in the projects that have been presented, most of them some still ongoing. This thesis weaves a coherent thread across all these projects to demonstrate the role of sovereignty when considering the ethics of algorithms. In hindsight, the road was long and many of its steps were extremely useful to understand the past and present of my practice, as well as to adjust future aims.

The intention of this final chapter is summarise the lessons learned and in order to do that it will follow the same order in which the main projects were presented. The main research projects will be taken into reexamination: Bitcoin, D-CENT and Devuan. However, all of the projects have informed the conclusions presented and their features resonate across this section.

Ultimately this thesis suggests a holistic approach to determine the ethical profile of algorithms, to analyse their literature in detail and to comprehend the complex influences they exercise on and receive from the space in which they are written and executed. This holistic approach implies that the role of the observed cannot be considered abstract, to the contrary anyone gaining knowledge on socialised algorithms influences both the algorithms and the perception of those using them. For such an evaluation, algorithms cannot be studied in a lab environment, nor can they be abstracted from the context in which they are written and that in which they are executed. They cannot be evaluated as broken into parts, as the reductionist approach of input/output systems suggests. To state this is far from a rhetoric exercise, considering the growing importance that algorithms have in shaping physical and immaterial spaces.

The first main research subject, Bitcoin, and the rise of similar blockchain technologies around it, is by far the biggest in which I've been taking part. In this research I'm focusing on the societal and political dynamics behind Bitcoin conceived as a digital object (Hui, 2016), inspired by the collective action and the motivation of some prominent leaders and focusing less on the literature around its financial and technical implications. Through my participation to the project certain revelations became clear and recently also confirmed by episodes which are still unfolding: for instance the attempt to change its algorithm to favour the financial industry (Kokoris-Kogias et al., 2016) or barely documented blockchain anomalies (Natoli and Gramoli, 2016). Contrary to public perception, the feature of decentralisation does not relate well with all socio-political aspects connected to algorithms: because running an algorithm on a distributed cluster of nodes is only marginally connected to what the algorithm does. Distributing the computation of an algorithm guards it from unilateral influences, but also augments the complexity related to maintenance, to the point that changes may provoke unpredictable mutations to the results and, in

the worst case, community decisions may be lost in the folds of technicalities and their execution. Decentralisation is a feature, often an innovation against corruptible systems, but is not a usable category to analyse algorithmic sovereignty; it is rather a political aspiration often moving towards ideas of neutrality, disintermediation and resilience. Sovereignty relates to the way a community can influence an algorithm, *appropriate* it, *distribute* it, *share* it and *create* new ones: such negotiations take place within the development of the architecture and not at the level of the deployed platform.

With Bitcoin it becomes obvious that the influence on its sovereignty is not facilitated by its distributed characteristics, which is the product of negotiations among a centralised group of programmers with clearance to change its algorithms. This is largely demonstrated by the fact the most controversial debates in Bitcoin happen to focus on development choices and it is rather notable how mature is the interaction within the "Bitcoin Core" development group in light of the enormous peer pressure exercised on it. Even more than algorithms, with Bitcoin it is the protocol that has central importance: the conformance to a protocol, witnessed by all participants, is the point of access into its sovereignty, no matter what algorithm is used (and in fact many implementations of Bitcoin exist in various languages). While decentralising the algorithm, Bitcoin's sovereignty lies in its centrally defined protocol, which is neutral to all its participants.

The Bitcoin protocol is the milieu where its "consensus" algorithm is designed: a technical feature that balances the scalability, speed and fault resistance of the system. The protocol is declared and shared and measures all interactions to be true to the chain of events recorded in Bitcoin.

The most resonating concept motivating the communities behind Bitcoin is that of the neutrality of its protocol: it is brought to an extreme consequence with the implementation of a "trustless" currency, basically cash in a digital form. The fact that Bitcoin is a digital asset that can be transferred without intermediaries is perceived as the main feature of the project itself and at the same time its political constituent.

We can individuate close ties between this concept and the concept of truth, which is of course a much debated topic in philosophy and is echoed in many more projects addressing blockchain technologies. I believe it is important to individuate within this context a new process of de/subjectivation in the rise of algorithms for governance.

In the classical definition of de/subjectivation given by Foucault, the device (dispositif) is of disciplinary nature. Here we observe the digital object (Hui, 2016): a sort of new territory to negotiate governamentality, a milieu where algorithms are substantiated as new forms of interaction with the subject, a new nature for the process by which subjects acquire or loose their definition and agency, play their fears and desires. This is the path along which we can observe how violence is transferred

by a sort of biopolitical equation into a mechanic domain where the responsibility is not anymore a moral question for humans.

In its call to create a "trustless" space, where trust is not necessary among peers, Bitcoin relies on a "trustworthy" technology: a protocol, a digital object of extreme power as it mediates social dynamics and moral principles. The seminal work by Caroline Nevejan on design for trust is most likely the best way to explain this point:

> Information and Communication technologies facilitate a transcending of time and place by mediating presence and permit a different scale of tracking and tracing and a different scale in collecting and distributing of information and communication than natural presence facilitates. Doing so, information and communication technologies also facilitate the taking of a moral distance, because of the way presence is designed through these technologies. (Nevejan and others, 2007)

The efficient execution of a rule is not the objective for algorithms, but a collateral effect to their inception and their mechanic execution. The very nature of algorithms for which sovereignty is important is the mediation of moral principles and ultimately the mediation of violence. It is very important to understand this concept, because the discourse around algorithms is often misleading and can easily distract their evaluation with features and implementation details.

The reason for computational ideologies to exist is not that of speed of execution or better resource management: it is the answer to the ethical conundrum of governance. The egalitarian ethos of modernity claims this answer, the historical transition into an industrial does. The prevarication and privilege entitlement among humans and more in general all other carbon-based life forms raises the need for an articulated agreement, often represented by the naive view of a pre-established scheme that works *super-partes* and that captures as much complexity as possible. Bitcoin itself is not exempt from this naiveté.

Here we are observing the digital declination of an evolved concept of "neutrality" in governance that has much in common with the birth of the neo-liberal political tradition as described by Michel Foucalt in his lessons at College de France in 1978-79. The algorithmic descent of the concept of neutrality seems as powerful as its predecessors at providing a rationality for governance. It is still here, to paraphrase once again Foucault, that socialism failed to provide a consistent algorithm for governance: it provided only an approach that patches a generic algorithm at different phases and at every level of its implementation.

Foucault gives us a key to understand the predominance of neo-liberist models of governance in the contemporary world: it is the "rationality" of its model (Foucault, 2008). This concept is analogous to what most people refer to as "neutrality" of digital objects. This is where the axiomatic relationship established by capitalism

with the central role given to the algorithmic regime of financial instruments is sublimating into an ideology of modernity. According to Foucault's account of the birth and growth of liberist ideals (from ordo-liberism to the neo-liberism of today, an history which he considered ancestral to the birth of biopolitics) liberism is more fit for a modern governance because it provides a rationality for it. After what Deleuze and Guattari define as the "rupture of codes" in modern society (Deleuze and Guattari, 2004), which we can consider as a passage from the dimension of *Gemeinschaft* (Community) to that of *Gesellschaft* (Society), the axiomatic approach of mathematics becomes a central feature of capitalism, a feature inherited from the protestant ethic at its origin (Weber, 1904). The space for new codes, for an "universal operating system" in these regards, seemed to require an adaptation to the only universal left intact as of today: mathematics (Caronia and Bianchi, 2012). It is here that the universal of mathematics, what Antonio Caronia used to call "the last surviving universal", is naively extended to algorithms despite also belonging to the domain of logic, language and, arguably, the social sciences and the arts.

The debate around computationalism (Golumbia, 2009) as much as the driving forces behind financialisation (Fumagalli et al., 2009) and the informationalist paradigm of governance are all offering ways to approach this debate from different angles. On the tangential topic of financial crisis and in an open polemic with the notion of "wisdom of crowds" (Surowiecki, 2004), David Hakken writes:

> Notions like "the wisdom of crowds" are virtually pure statements of informationalism. In its naive presumption that, as more information is added to the model - that is, the more computable it becomes - its accuracy tends to increase, informationalism becomes an aspect of computationalism. In short, reliance on prestigious, information filled, but systematicity presuming and therefore blind to systemic crisis, and thus deeply flawed, computationalist computing also caused the crisis. In this way, the models contained too much information, on the one hand, while ignoring crucial information (i.e., regarding systemic risk) on the other, because it was not easily quantified. (Hakken, 2010)

While the views exposed by Hakken can be extended to constitute a fundamental criticism to most computational ideologies, the ambition of this thesis shifted towards outlining patterns where algorithms can contemplate systemic risks in the way they are designed, implemented, understood and communicated. Or, in other words: *created*, *shared*, *distributed* and *appropriated*. By doing so, it is not the intention of this thesis to dispute the critique of computationalist ideologies, rather accepting it by considering the concept of neutrality a mere utopia and proceed analysing the conditions of governamentality and sovereignty in algorithms taking into account the socio-political context in which their mediation takes place.

This research then reaches a firm conclusion: that there is nothing inherently

neutral in an algorithm. To the contrary, an algorithm is there to implement visions, ideas, beliefs and to satisfy needs and desires; contemplating this is a fundamental condition to understand the role of sovereignty and enable participants to relate to it. To perceive the presence of a decentralised algorithm as guarantee of neutrality, to consider its existence per-se as the basis of a constituency, is an error made by many. This error is also evident on the surface of Bitcoin, a project whose foundations are social, political and technical. Bitcoin is then better described as an hybrid object of algorithmic nature (Latour, 2012) focusing on the characteristics of replicability, transparency and freedom which are of paramount importance for its sovereignty and reflect in the ideals of the people inhabiting its territory.

Considering a complementary situation to Bitcoin's model of governance, it is well possible to conceive ethically sound algorithmic sovereignty under the control of a governance that takes into account their mediation. Decisions made by algorithms and changes operated on them should be clearly verified and explained to most people in the community and it should be possible to override the results of any mediation according to an "open world assumption" model (Keet, 2013). This is what we described in the D-CENT project as "Social Proof of Work" (SPOW). What is important to understand in these regards, the hard lesson many software projects go through, is that once a promise is made to a constituency, for instance once absolute decentralisation and the (naïve) idea of neutrality are cast as foundations, these foundations cannot be rejected anymore without an inclusive and democratic process. If a protocol does not assume "by design" that in the open world there may be new unknown conditions that need ad-hoc consideration, requiring it to apply unpredictable adjustments, it will always be controversial to contemplate such an adaptation at a later time.

When designing algorithms is always important to take an open world assumption (Keet, 2013) into account and a rational, negotiable way to apply changes that is transparent and whose process is reproducible. This is of central importance for the formulation of governance models that can keep focusing on living participants as the main constituency to be served and respected, but it is unfortunately not present in the vast majority of "blockchain" software implementations deriving from Bitcoin and, only in a controversial way, it is present inside Bitcoin as the political process of negotiation of code changes within the core development group.

The second main research and development subject in this thesis, D-CENT, represented my first involvement in an extremely diverse research consortium, has been pursuing rather ambitious goals and received praises by EU officials and reviewers. D-CENT has been focusing on techno-political strategies and solutions for participatory democracy: it analysed, specified, documented and even realised tools to help the bottom-up inclusion of societal movements in the policy making process. Lead by a interdisciplinary consortium, D-CENT brought attention to the importance of

interdisciplinarity.

The activity in D-CENT project is focused on facilitating the formation and consolidation of new networks of trust and new social institutions that are driven both by incentives and mutual interests. The project started from a theoretical approach bound to field research, with experts who contributed both a qualitative and quantitative analysis in the fields of anthropology, ethnography, economics, management and political science. Progressively more specialised observations have transformed the results gathered through iterations that became closer to technical specifications, through a process which may be presented as a research and development methodology by itself.

Following a linguistic approach that focuses on utterances (statements) can help by orientating through terminology, providing context through different discourses, carrying this expertise over into the namespace of algorithms, so that computational language can better reflect the features and needs identified by the diverse research materials. This also suggests that linguistic studies (and not necessarily computational linguistics) could be developed into a new interdisciplinary form of expertise, to complete the spectrum of specialisations needed to lead algorithmic research projects to success. Notwithstanding the achievement of establishing a interdisciplinary process in D-CENT, more awareness on the role of a linguistic approach can facilitate furthering the goals and improving the overall quality of the process.

At the onset of the D-CENT project, all researchers and practitioners inherently agreed to the fact that theory and practice must go hand in hand, though the temporal sequence for this synergy was not clearly established and the project was envisioned as a progression from theory to practice. As someone whose role was to be in charge of analysis close to technical development and related tasks, I believe now, in light of all what has been observed, that interdisciplinary processes should start as early as possible, as also suggested by the LEAN UX methodology, and inform theoretical and field research not only in terms of what is or not possible to do, but also in terms of verifying assumptions. In any case, and this became clear in D-CENT from an early point in time, technological choices should never come before theoretical ones, as mature experimentation with technology cannot transcend the social and economical implications of a theoretical research. Unfortunately and perhaps worryingly enough, this is how the contemporary phenomenon of the so called “startup economy” works. Startups too often decline the development of “disruptive innovations” from the starting point of new technologic promises, then proceed in ways that are often insensitive to the social, political and natural context in which they are operating, with the result of leaving most people (and prospective clients) insensitive to them. While some startups struggle to develop good “user experience” scenarios on top of technologies, the underpinning of this approach is to profit from the top-down provision of new technologies and the scaling of service and liability aspects connected

to them.

In the field of algorithmic development, a community based approach becomes very different, whilst opening the opportunity for what is commonly referred to as social entrepreneurship (Mair et al., 2006). The third main research and development subject, Devuan, is the clearest example this thesis presents to demonstrate the creative potential of communities willing to gain algorithmic sovereignty in contexts where a fork is permitted, namely projects adopting F/OSS licensing on their algorithms. A full blown GNU/Linux base distribution was born, Devuan, whose existence was of interest for specialised media outlets, among them Linux Journal[23] and Linux PRO Magazine which distributed it worldwide on DVD. Following the "Debianfork" declaration, resting on the basic claim of "init freedom", what can be defined "freedom of algorithmic choice", Devuan forked Debian after the massive switch to the systemd software. This project could immediately count on a large and active community and a group of highly skilled and motivated developers.

The nature of Devuan's constituency is not purely technical: most technical effort invested in the project went to setup its "build infrastructure", yet no major innovation had to be introduced in Devuan compared to the Debian system itself. To the contrary, Devuan's constituency mostly relates to the governance applied to its algorithms and their introduction, to the sovereignty that its participants wanted to have with regard to the software they use.

The role that a "fork" may have in algorithmic sovereignty is the most interesting aspect highlighted by this experience and in close relation to the research subject. To fork an algorithm (commonly indicated as software in these cases) means to copy its code and evolve it in a different direction than the original, basically splitting the development groups and, in some cases, the communities. It often also means re-deploying the software on a new platform, in case the software is operated as an on-line service. Not all forks succeed, yet some manage to be sustainable on the long-term: a popular example described in technical literature is "LibreOffice" (Hammouda et al., 2012) a mission critical software that had to be freed from the grip of a corporate acquisition putting at risk its ancestor project "OpenOffice". Forks occur in every software domain and have become more frequent in recent years, especially thanks to the evolution of tools aiding the maintenance of algorithms through integrated versioning environments and source-code meta-models (Antoniol et al., 2005). Very few forks merge with the original project (Robles and González-Barahona, 2012), but rather end up creating an "alternate future" for it, gathering the participation and consensus of the critical mass that initially motivated the fork

[23] "Devuan Beta Release" article by James Darvell, published on 3 March 2016, gives a balanced journalistic account of the origin of Devuan, but its aimed at specialists of the field and dives heads first into the topic without much introduction http://www.linuxjournal.com/content/devuan-beta-release it is advisable to refer to the full section on the Devuan project in this thesis to explain some of the passages in this and other related articles.

and that can well grow in time.

Forks can be classified to be of two different kinds: endogenous and exogenous (Fung et al., 2012). An endogenous fork is conceived as occurring "within a community of developers who work for the same software product line" and is basically a flavor of the same base product, a configuration common to "community" and "enterprise" versions of the same software, like with the Fedora and Red Hat distributions. Exogenous forks are splitting the development across different groups of their participants, which may or may not entertain a relationship over common components, often not about future choices. According to such a definition, Devuan is an exogenous fork born with the precise and agreed focus of maintaining the software architecture existent in Debian 6, integrating it with future updates while avoiding the introduction of systemd in Debian 7. The decision to fork was supported by a significant amount of people, many of which were part of the Debian community, with the formal endorsement of one Debian developer and the informal (private) endorsement of many more, though the development effort behind Devuan has been taken up by an entirely different group of developers.

Some Debian developers have perceived the Devuan fork as useful for Debian itself, because it maintains an alternative in case current choices in Debian's development prove to be wrong. But the trust is not mutual: those of us who forked to maintain status-quo believe that Debian has betrayed its social contract due to vote miscalculations and lack of mediation over the results. At the time of writing this thesis it's hard to predict if we can rebuild any trust in Debian's governance, as these negotiations require time and the completed fork gained us a certain advantage compared to the situation in which we were treated as mere "users". At this time, the Devuan development proceeds on its own path and maintains its own distribution without having to rely on a questionable democratic processes that include Debian developers to the exclusion of all others participants. But then thanks to the way Debian was conceived and built since the beginning following transparent, documented and reproducible processes, the trustworthiness of Debian's technical legacy can be maintained and integrated in Devuan, even inheriting all the work done in Debian packages to date.

Most free and open source licenses allow forking without a problem. The possibility to fork is considered to be protected by the four freedoms of free software: to study, modify and redistribute with modifications. Some licenses regulate the use of brands and trademarks in relation to forks, others allow the forks to deviate from the freedom clauses and build new software that don't even need to be free as their originals. From an ethical point of view, the best licenses contain a "viral" clause and are endorsed by the Free Software Foundation and the GNU project. These licenses deliberately protect the freedom of users also in case of forks: they extend this commitment to the release of all source-code, even that which is produced after the fork, under the

same license.

In terms of this research subject, it can be asserted that the possibility to fork is a controversial condition for algorithmic sovereignty: it is something that safeguards against planned obsolescence and misbehaviour by few leading actors (Ernst et al., 2010), but comes at a very high cost in terms of labour and resources. Forking is not simply granted by a licensing clause, but requires the presence of a critical mass of knowledgeable people who can at any moment and completely autonomously take another path for the project. This situation can be considered healthy for the project as it may be in the interest of all participants to have alternatives: it limits the hegemony of the current leadership while it clearly signals the fact that the project's algorithms are intelligible and can be mastered by people beyond the circle of current leaders. Forking is then interpreted as a gate that can always be opened from the inside once one has already accepted the free software license and its conditions for use, distribution and modification. Anyone stepping out can carry one's possessions (*Appropriation*) to be further *Shared* and built upon (*Creation*) and *Distributed*. The only condition is to keep intact the history of the process in all its details: if the common heritage is kept intact then the trustworthiness of algorithms can be traced back to its origins even beyond the point of the fork, allowing it to not waste what has been done before the fork occurred.

Yet the notion of fork is problematic and the mere fact that a project is "open to forks" does not grant sovereignty to its participants. The politics of open and closed systems cannot be reduced to a world of binary opposites (Tkacz, 2014), nor of legal or technical features. There are two important researches who have focused on the historical case of the Spanish Wikipedia fork, those of Nathaniel Tkacz and Robert W. Gehl, critically illustrating this rare episode of fork involving the exogenous duplication of a platform running the same software, provoking a change in the governance of the originating project and culminating in a final merge back into it.

I believe forking can be described as a re-appropriation of production means by a "labour force" or better defined a critical mass of participants who can afford the price of maintaining algorithms and in many cases re-deploying a platform on their own premises and with the ambition to match or surpass the scale of the originating process. This act comes at a high price as its well made clear in the documentation of the Spanish Wikipedia: a lot of work has been done just to steer clear of corruption - as arguably would be an act of corruption that of adopting commercial ads on the voluntary work of Wikipedia's editors. The price of a fork is not for anyone to be afforded and those of us who are involved in Devuan know it very well. We are paying a toll not only in terms of time and labour, since the entity of the effort affects also our private life and the time we can dedicate to family and friends, plus all the thoughts that occupy our minds and attention and that we feel debating with our peers. When caught in such a big amount of work is quite easy to loose balance

and burn out: an important part of the activity is then that of developing a sense of solidarity among core developers and even taking care of each other well-being.

In addition to this consideration, which is the most obvious drawback that anyone engaging a fork of substantial dimensions immediately faces, there is also a less evident yet very important aspect emerging from the analysis conducted by Tkacz: that the "binary politics" of open and closed worlds are actively hindering our ability to make sense of the political.

> The seeds of closure are always already present within the open, but the language of openness doesn't allow us to gain any traction on that closure. (Tkacz, 2014)

This passage may seem somehow obscure at first sight, but strikes some painful chords in my experience in Devuan: it echoes the "benevolent" argument of many Debian's developers in the moment they acknowledged our project was not a joke. The argument of Devuan "being good for Debian", namely because it creates competition and because it also demonstrate that "Debian is open and can be forked", completely denies the fact that the fork was made necessary because of a fundamental betrayal of Debian's mandate: it completely denies the political sense of this fork. This situation is not evident and difficult to describe, it will probably take more time to be understood and is only documented by the fact our collective explicitly refused Devuan to be listed as "derivative" in Debian documentation when asked. Furthermore, it shows the obvious, that an "open institution" may declare itself "open to be forked", but this way it can as well preserve itself from change and deflect criticism from affecting its governance while, as it was in case of Debian, even violate its own mandate. To achieve this awareness on the issue I believe the research methodology adopted by Tkacz while acting as a third person observer of such a phenomenon, emerges as incredibly successful: grafted on Austin's concept of utterance (Austin, 1975), echoing Foucault's notion of énoncé (Foucault, 1966) in one of his earlier works, this method of analysis demonstrates to reach deep into the subject and beyond what we could define as a rhetoric of openness. My biggest regret at the conclusion of this research is to not have adopted this methodology through all this thesis, yet that may have limited my reach into practice for a simple matter of time and focus.

The constructive challenge we are left with is indeed that of moving our attention on the "politics of organisation" and in my defence I believe that Dyne.org represents a practical attempt to create an organisation capable of engaging these issues at deep: the challenge of developing an holistic approach to algorithms that regards the sovereignty of all participants as the most important notion for their governmentality.

10.1 For an holistic analysis of algorithms

During the participation in the presented projects, it became increasingly clear that the categories proposed by the Free Software Foundation as well as the popular focus on the privacy implications on data storage and distribution, though necessary, are not sufficient to achieve a holistic understanding of algorithms. The methods adopted to design an algorithm, the participation gathered around such a process, the inclusion of affected communities, the governance that maintains and updates the algorithms, the techniques adopted to document them, the modes of production and the societal needs that justify them: all of these aspects must be taken into account, above and beyond the efficiency of algorithms themselves. I'm convinced of the need for interdisciplinary studies in the field of technological research because of the complex relationships that most technologies establish with their surroundings and the living world.

In light of all these examples and considerations it is very important to consider all algorithms in which living creatures participate as Commons (Ostrom, 2015), as belonging to all of them, or more precisely as Knowledge Commons (Vercellone et al., 2015). In order to do so, it is necessary to give particular attention to features and uses of algorithmic languages, well beyond the legal and economic considerations concerning the contexts in which their are interpreted and executed.

Recent and well informed studies in the field of governance, more specifically in the relatively new context of "Science and Technology Studies" (STS), clearly state that "algorithms are not only technologies for information production. By sorting data, people and behaviours out, they enact regimes of inclusion/exclusion, and thus act as full-blown decision makers" (Pelizza and Kuhlmann, 2017). Algorithms are responsible for increasingly influential implementations of norms and are used to consolidate digital architectures for societal interaction. Arguably, algorithms will increasingly mediate the physical space we are living as the industrial world progresses into "Internet of Things" scenarios and public spaces embrace sensor intensive approaches for all sorts of monitoring needs.

While the importance of algorithms keeps growing, their comprehension is still limited outside the field of computational science. It is important that the consequences for deploying an algorithm in a particular context are questioned in relation to its design and that those living in that context are brought into a position of algorithmic sovereignty.

To conclude with an even larger horizon beyond this proposition, while admitting the shortcomings of my own current perception of what an holistic understanding of algorithms can be, I believe Katherine Hayles goes far and still needs to be fully understood in her call to overcome the subject/object dichotomy:

> An essential component of coming to terms with the ethical implications of intelligent machines is recognising the mutuality of our interaction with them, the complex dynamics through which they create us even as we create them. (Hayles, 2010)

Far from being covered by most literature on the subject, this call responds to what we can really consider to be consciousness in cybernetics discourse (Ascott, 1999). In light of Hayles and Ascott's intuitions then this thesis is still half way, or a project in-fieri, to refuse to inscribe algorithmic interactions in structures of domination and instead to seek out understandings that recognise and enact the complex mutuality of the interactions.

> "What we make" and "what (we think) we are" coevolve together; emergence can operate as an ethical dynamic as well as a technological one. Realisation arrives not in an instant but in successive cycles of awareness, each building on what came before. So it has been for me in writing this book, and perhaps for you in reading it. As is always the case where dynamical emergences are involved, at the end we begin again. (Hayles, 2010)

10.2 New forms of rationality as new forms of liberation

As algorithms become increasingly important, there is the need for a new form of rationality that can interpret their role and objectives by engaging their literature, with the clear objective to see code not as a form of subjugation, but as an opportunity for new forms of liberation.

The intention of this thesis, underpinning for its practice based approach, was that of providing answers by engaging and documenting practical approaches. In light of a need to create an informed discourse on algorithmic sovereignty, the proposition is that this research can be exemplary for the development of a methodology, a sort of curriculum in "radical interface pedagogy" (Gehl, 2014) that, along the adoption of new organisational forms, is capable of studying algorithmic governance. The challenges of this proposition are beyond the scope of this research, yet this thesis advances the awareness about these needs and, by documenting concrete projects, demonstrate possible work-flows and a methodology. All the projects engaged start from a theoretical framework that embraces social, political and economical aspects bound to the act of designing, implementing or modifying old and new algorithms, then move their steps into the development of practical community based applications that can be governed by their own participants. This journey has been moving similar steps across different projects to answer the research question in all its complexity, avoiding to reduce the answer to a theoretical formulation or the application of a single discipline.

Essentially, this conclusion stands to oppose the looming perception of algorithms

as neutral and rational per-se. There is nothing neutral in an algorithm, to the contrary every algorithm has to be seen as the cultural product of a negotiation of power and its analysis requires the understanding of its language applied to the living world. It is not only the role of technology to develop a well informed debate on algorithms, while the pure analysis of data is not enough to describe the conditions in which samples have been produced and the ways they have been transformed. Ethical considerations on the social, economical and political conditions that algorithms produce should be conducted with great attention to the processes of *Creation*, *Sharing*, *Distribution* and *Appropriation* granted to participants. Just as a modern nation validates its sovereignty by entitling its own population to interact with governance by the principles of democracy, in the same way algorithms need to comply with notions of inclusiveness, accessibility and diversity to not threaten the freedom of their own participants. More precisely and according to the methodology adopted in this research, the *Purpose* and ethics of algorithms can be assessed by taking into consideration their reliability and interoperability, the collectivity they facilitate and their maintainability and elegance. A complete analysis of these aspects should not overlook an algorithm's capacity to facilitate the stewardship of the commons and the freedom it grants to participants to appropriate its language, as well its simplicity, transparency, replicability and accompanying documentation.

This research confirms the importance of understanding the purpose and ethical characterisation of new algorithmic forms of governmentality, especially in relationship to the sovereignty they grant to participants. However the observer must know well how to interpret the languages adopted and understand their use in detail. It is well known that an algorithm cannot reach excellence in its own role without being used by its own developers. Then the acts of observation and engagement presented in this thesis will turn the researchers into practitioners, the artists into hackers, the activists into political scientists and the programmers into poets.

11 Appendix

Documents attached to “Algorithmic Sovereignty” by Denis Roio

11.1 Quick reference of projects

Table 11: Reference table of all listed projects

Project	URL	Year	Nature	Advancement
D-CENT	http://dcentproject.eu	2013	R&D	Methodology
Bitcoin	http://bitcoin.it	2010	Research	Context
Devuan	https://devuan.org	2014	Software	Participatory
Dowse	https://dowse.eu	2013	Software	Practice based
Entropical	http://entropical.org	2015	Artworks	Context
sup	https://sup.dyne.org	2016	Software	Practice based
Gufo Volante	https://gufo.dyne.org	2016	Software	Practice based
DECODE	https://decodeproject.eu	2016	R&D	Practice based
Secrets	https://secrets.dyne.org	2017	Software	Practice based
Dyne.org	https://www.dyne.org	1999	SW house	Organisation

11.1.1 Git repositories

- https://git.devuan.org
- https://github.com/d-cent
- https://github.com/dyne/dowse
- https://github.com/dyne/sup
- https://githuc.com/pienews/secrets

11.1.2 Fora

- https://lists.dyne.org/lurker/list/dng.en.html
- https://talk.devuan.org

11.1.3 Audiovisuals

- D-CENT https://www.youtube.com/watch?v=bK2QdViY5PU
- Entropical https://www.youtube.com/watch?v=tsTljDLXmUQ
- Dowse https://www.youtube.com/watch?v=vquh3IXcduc
- Devuan https://www.youtube.com/watch?v=wMvyOGawNwo

11.2 Critical Engineering Manifesto

0. The Critical Engineer considers Engineering to be the most transformative language of our time, shaping the way we move, communicate and think. It is the work of the Critical Engineer to study and exploit this language, exposing its influence.
1. The Critical Engineer considers any technology depended upon to be both a challenge and a threat. The greater the dependence on a technology the greater the need to study and expose its inner workings, regardless of ownership or legal provision.
2. The Critical Engineer raises awareness that with each technological advance our techno-political literacy is challenged.
3. The Critical Engineer deconstructs and incites suspicion of rich user experiences.
4. The Critical Engineer looks beyond the "awe of implementation" to determine methods of influence and their specific effects.
5. The Critical Engineer recognises that each work of engineering engineers its user, proportional to that user's dependency upon it.
6. The Critical Engineer expands "machine" to describe interrelationships encompassing devices, bodies, agents, forces and networks.
7. The Critical Engineer observes the space between the production and consumption of technology. Acting rapidly to changes in this space, the Critical Engineer serves to expose moments of imbalance and deception.
8. The Critical Engineer looks to the history of art, architecture, activism, philosophy and invention and finds exemplary works of Critical Engineering. Strategies, ideas and agendas from these disciplines will be adopted, re-purposed and deployed.
9. The Critical Engineer notes that written code expands into social and psychological realms, regulating behaviour between people and the machines they interact with. By understanding this, the Critical Engineer seeks to reconstruct user-constraints and social action through means of digital excavation.
10. The Critical Engineer considers the exploit to be the most desirable form of exposure.

The Critical Engineering Working Group
Berlin, October 2011-2016

11.3 Dowse and the tactical withdrawal

This lyrical text surveys the techno-political context where Dowse was born, it was posted in the Bricolabs mailinglist by Rob van Kranenburg, worldwide renown IoT expert and Dowse developer.

11.3.1 I

In my dreamy moods I wonder and think there is only one purely rational explanation for the transition we call Internet of Things, #IoT: we are building a space ship; some organizational entity that is able to make contact with other spaceships out there. Hmm, I do admit that I have been watching all Star Wars episodes lately running up to the most recent one.

Still does anyone have a better explanation? The scope, size, speed and synergies fueled by #IoT, aka pervasive computing, aka ambient intelligence, aka ubicomp is like nothing we have ever seen on the planet. Making every object on the planet identifiable? (barcodes, QRcodes, RFID, NFC..) Ipv6 in any object that can hold a minimum of software? Is this not an attempt to turn the once unpredictable planet into a giant factory where every operation, every single everyday existence is subject to predictive maintenance?

The fact that most children in the Western world are growing up in light of sight of their parents leads to being fully surveilled in the cities is not perceived as problematic, but very 'normal'. Generations glued to screens and happy with that, are they not being groomed for a life in the ship? They might still want to walk in the woods (sometime sometimes), but what about their children? For these children the fact that an NFC tag triggers a movie has become a quality of the object, of the shirts. Objects trigger events, movies, other objects, of course! Was there a time this was not so? How boring!

Maybe I should say 'it' is building a space ship; some overtone in techné as we know it since the first tooling of chisels. Just look at worried Heidegger walking his hills, pondering what is happening to that 'extra' force that mechanical engineering brings to machines, adding force to human biceps. This overtone, this accumulation of extra, now picture that in software writing software and self learning algos in big data environments. Has this not always been our main worry as humans? Freud already identified that the 'unheimliche' was not something strange to us but the hyper-familiar, our home becoming 'smart' maybe?

According to Antonio Gramsci: "The crisis consists precisely in the fact that the old is dying and the new cannot be born; in this interregnum a great variety of morbid symptoms appear. The old world is dying away, and the new world struggles to come forth: Now is the time of monsters."

https://www.goodreads.com/work/quotes/855705-quaderni-del-carcere

Indeed, we see the monsters all around. Large corporations building intranets of things accumulating value just for their shareholders (for a while until they break). Decision makers tuned to the 20th century, politicians in 'political parties' keep chatting while existing only in the 8 o'clock news, fully void of any operational agency. Board members and CEOs functioning out of ego and superficialities clinging to what seems to define them, their 'job' description. No, it is difficult to find some style, some dignity these days. Now if old can not be defended, it is the new we should support.

Yet if the real goal of IoT is bringing radical (Luciferian) transparency, then paradoxically we have to support the Borg; the drive towards a oneness of global protocol, identifiers, database, and supported algos. Are we then on the side of Anakin, or Darth Vader? Can our minds synthesize a new order in which there is no dichotomy between the Empire and the Rebellion? Can we, in other words, build a federated yet global system in which radical transparency reigns as the new caste of rulers that we have seen on this planet from hunters to farmer-tribes to warrior-rulers, sages, priests and citizen-politicians?

After all, it is just a small planet, one of many in the universe. We should be able to manage that without heating it out of orbit?

11.3.2 II

"Truth is ugly... for its eventual"objectivity" – the latter understood not as "disinterested contemplation" (which is incoherent and nonsense) but as the ability to hinge and unhinge and to hold sway over its pro and con, so that one knows how to make the very diversity of perspectives and affective interpretations useful for knowledge" - Nietzsche, on the Genealogy of Morals

"We would certainly be happy if we could all get along well together and unite all the forces of anarchism in a strong movement; but we do not believe in the solidity of organizations which are built on concessions and assumptions and in which there is no real agreement and sympathy between members. Better disunited than badly united. But we would wish that each individual joined their friends and that there should be no isolated forces, or lost forces." – Errico Malatesta

The key to #IoT is identity management. However, before you can manage identity, it needs to be made manageable. In this brief post I will explain how this has come to pass, or actually how the fact that there was no fixed identity ever has been hidden in a historical narrative of the individual vs the collective.

You can call it a ship. You can also call it a 'smart object':

"The idea is turning the world into a smart object that can be continuously improved, and we couldn't be more excited," said Matthew Wood, the general

manager of product strategy at Amazon Web Services, or A.W.S., the retailer's giant cloud computing business."

looking-beyond-the-internet-of-things

Ted Hughes poke of the grin hat was trying out faces.

https://en.wikipedia.org/wiki/Crow_(poetry)

In our case it is fear attempting to become integral. Fear is trying out all human capabilities. This has come to pass as generation after generation of corrupt, spineless and greedy people have been in the drivers seat. We cannot call it leadership. These people, let's say Bilderberg, Fortune 500 and MBA globally have facilitated anything 'easy', outsourcing anything 'hard' to places that could be exploited. So now there is no more place to exploit. Hence the fight for the internal space, the very notion of what it means to be human. If people can not be exploited and enslaved but would wake up the shame some would feel of having been deceived for so long in what it means to be living, hmm hara-kiri would be their only option.

Most of them cannot wake up, however, as there is no longer a self to wake up to. They are zombies. No it is not a coincidence that we have them on our tv. The writers feel they are real. I too see them every day. They have been among us for quite some time. I just never thought there would be so many. So it is up to us to make things find the hard road again. This is not a mass movement. We will see you when you see us.

11.3.3 III

So there I was one night.

"Historical 'evidence' itself tells us that hegemony has always been a process of conflict and struggle, and that this conflict often took place at the level of the subjective. Human nature, the human self, has always been the terrain of conflict because it is first and foremost human beings who constitute social relations – these relations are not made by some invisible hand of god, or even of capitalists, without either the consensus or coercion of people themselves." – Kylie Smith

The heart of the matter is that empathy is necessary for the functioning of any group or society. Without it, it breaks down. Every generation negotiates the meaning and scope of empathy with the tools at hand. Among the tools of any given period is the notion of the individual and identity itself. It is a tool with which to construct the space where empathy can work and be negotiated. Identity itself is a tool. That is something to ponder for some time.

If there is a mismatch then every action in any given arena - political, social, economic, personal - feels 'off', not really matching. Hence the short but fundamental success of periods that feel well rounded and functioning well with each and everyone and thing to its "proper" place.

The notion of a single consciousness in individuals as a proper sense of undivided self can no longer be strived for but also no longer perceived as operational. It is a historic definition tuned to the scope of the tools and tooling at hand.

Current evidence from all kinds of sources and a wide variety of thought points at a plurality in the very heart of what it means to be an individual. This plurality is a given and a capability, not as a drawback of the 'original' concept of being undivided as there is no original concept, just a series of actualizations of consciousness reacting to and consequently shaping its environment.

The environment that we have to operate in is defined by a successful combination of the results of the outsourcing of meaning in the Shannon paradigm of communication (allowing the engineers to 'innovate' only on nodes that they can identify - thus not including social structures, other forms of intelligence, concretely: electronic and other waste, job loss, changes in power structures...) and the radical cheapness of hardware, software, database storage and analytics facilitating a world in which every object can (and will) be connected either passive (RFID, NFC, barcodes, QR codes) or actively through IPv6 in any object that can be embedded with a chip and software (Raspberry Pi Zero cost 5 dollars).

This environment foregrounds monitoring, classification, optimizing the classified, and operational efficiency in feedback and response loops favoring communication that is immediate, well identifiable as communication and direct. As such it is tailored to the functional qualities of a particular type of intelligence, not to all known forms of how humans have been known to act and feel.

At the same time does this operational technology allow for an immense variation in idiosyncratic and iconic language as every particular event is filmed, every particular first step of any particular baby is captured and transmitted and shared and commented upon by others that consequently share their stories the lead to new quite different yet similar but very different to the people doing them.

Our paradox then is that the sharing that we see forces individuals to be more unique and thus create as many different but similar stories still trying to tie these to some kind of Uber-ich, some kind of I that is no longer there. It is no longer living at home, as there is no more home to live in.

So that what was once seen as a dangerous way of living or thinking - to let go, to lose yourself, to be ego less - from an institutional point of view of a structured society becomes the default for a world in which connectivity is 'integral'. This means that ways of thinking that were deemed 'sick' or 'ill' now seem to be the ways that a 'modern' individual should try to incorporate in order to survive in a world of less and less certainties.

Among these is the notion that an individual is a series of interactions. This notion then becomes a tool for which we can begin to negotiate empathy in private

and public space.

As there is no 'self' we also have no more 'ethics' to engage as a tool, for this very notion assumes that there is a relatively stable 'self' that can be addressed and more important 'grow' and learn.

If we cannot have ethics, if we can not build 'hegemony as there are no actors to build it with, we need rules. We not just need rules, we ask for rules as we feel 'lost'. And we 'are'. With rules comes a system.

Full circle.

Our very self has become or is again more determined by the actual daily interactions that we have. Our daily interactions are fully immersed in brief burst of continuous short focused engagements in which 'we' fully lose ourselves.

In those we feel vulnerable and seek protection. What else is that but a social contract?

And what else can that be but the world as a 'smart object' or the world as a space-ship?

11.3.4 IV

```
Are they blind, the lords of Gaza In their strong towers,
Who declare Samson pillow-smothered
And stripped of his powers?
O stolid Philistines,
Stare now in amaze
At my foxes running in your cornfields
With their tails ablaze,
At swung jaw-bone, at bees swarming
In the stark lion's hide,
At these, the gates of well-walled Gaza
A-clank to my stride."
   - Angry Samson, by Robert Graves (1895 - 1985)
```

So our very self has become or is again more determined by the actual daily interactions that we have.

Our daily interactions are fully immersed in brief burst of continuous short focused engagements in which 'we' fully lose ourselves. In those we feel vulnerable and seek protection. What else is that but a social contract? So who can protect? Who can guide, shelter and comfort our individual attempts at being in this world?

No longer the warrior-soldier, as the warrior can protect inner territory only after the fact. The warrior function in the state has become drenched in moments of mourning and remembering. With every new attack or event we see bigger spectacles

of mourning. As is it not real territory that can be won or is under attack, we cannot turn to the warrior for help.

The merchant-warrior has build a global order in which half of the global wealth is held by the 1%. It has run out of clear and believable arguments of why this should be so, even within its own framework.

https://www.theguardian.com/business/2015/jan/19/global-wealth-oxfam-inequality-davos-economic-summit-switzerland

The sage-priest is lost in reconciling stories of old (mostly build on rituals around food) with meaningful stories about life and death in the current circumstances and thus focused on detail, turning every tale into a puzzle. I have borrowed the terms from Merchant, Soldier, Sage, A New History of Power from David Priestland and his model is very helpful.

https://www.amazon.co.uk/Merchant-Soldier-Sage-History-Power/dp/0241955211

We can now see what is at work in our current daily world. All the above categories of power are helpless in the face of the growing new caste that they all have helped to build: the technocrat. This category does have qualities of the merchant, the warrior, the sage, but as it is also this accumulation of extra's it has grasped at the potentials for an autonomous trajectory and is now actualizing itself.

At its core it is empty in values, economics, wisdom, morals or ethics as it purely the optimization of the things-at-hand. It therefore as Heidegger predicted will turn every thing and object into 'something at hand', 'some thing in reserve' hence the very notion of the 'smart' city that should consist only of sensor-readable objects.

The reaction of the old categories of power are predictable; their main qualities become hyper foregrounded. Their supporting qualities disappear. Examples are the very appearance and success of Donald Trump on the political scene as the Uber merchant, the actual occupation of territory by Putin at a time where we all thought that this was so un-digital and the full disappearance of any intellectual global sage- elite into entertainment and 'stardom'. The sage-merchant fully becomes a techno-bureaucrat as we can see in its best example: the European Union.

When giants fight and clash and finally die, it is best to stay out of their way.

What can simple people like us do anyway?

Under certain circumstances when the rule of a particular caste is clear we have clear ways of acting. For example, if you were living in the thirties in Germany you could go into Innere Emigration as a sage, you could rebel and oppose as a worker-warrior (communist) or you could fully support the growing fascist camp. You could even try to influence key actions directly by talking to one man. On 17 February 1940 Manstein traveled to Berlin to breakfast with Adolf Hitler.

"Lieutenant Colonel Henning von Tresckow, one of his staff officers, had invited his old friend Schmundt, Hitler's chief adjutant and personal staff officer, to Koblenz.

Schmundt reported the observations of Headquarters Army Group A's thinking to Hitler, and came up with the idea of a breakfast meeting so as not to incur the suspicion of OKH. In Inside the Nazi Warmachine we read:"Reported to the Fuhrer with the others. Breakfast followed. [He displayed] amazing knowledge over military-technical innovations in all states. Afterwards I was detained for an hour to discuss operations. I presented the essentials of our memorandum to OKH. Had full agreement. Indeed an astonishing convergence of thinking from the same points of view that we had represented right from the beginning."

http://bevinalexander.com/books/inside-the-nazi-war-machine.htm

The rest is history.

So this is the simple observation that explains the current situation. We as simple people, as citizens, we see all the categories of power of old overreact in our eyes and lose credibility. They have supported the view of life as convenient and consumerist. We accumulate things and the more we have the happier and 'full' we should be. Once there were still other values that framed this view of the world. Now, it is framed no longer. It is in the drivers seat.

It has no vision of where to go. It does not really know what to do. So it starts connecting the other cars in the hope that the granularity of that very action brings new use cases.

The first and foremost duty that we have is to stare reality in the face and confront it.

This then, is the situation. It cannot be changed by any action of us. We have to sweat it out and see what happens.

As Sheryl Crow sings in Leaving Las Vegas: "No joker, no jack, no king can take this loser hand and make it win."

It is clear that the situation is serious. The old forms of power will not give in and will keep pushing their core qualities into more wars, more inequality and more spectacles. That much is certain.

So, tell me, how difficult is it to unlearn literally every thing you have been taught and realize the truth and then lean back against the wall of a three-store apartment balcony?

In a moment like this it does not make sense to try to be part of any of the processes that any of the giants is leading. We can not change the drivers behind the actualizations of these struggles. So let those who still profit from the hegemonic fictions that they have build around their core qualities; as if there still is a society, as if there still is peace, as if there still is rationality behind political action, well, let them stand on their own.

I think it is best to withdraw all support and focus only on what you can change, namely: yourself and your immediate surroundings.

This is a tactical withdrawal. It is not a surrender, nor an act of selfishness. “I am a north wind to ripe figs” Nietzsche said. Until that wind blows it is best to stay put and become ripe.

11.3.5 Concretely?

Download the latest release of Dowse, dowse.eu http://dowse.eu/ and let’s start to work with real building blocks, step by step, (free from the ‘guilt’ energy we lose as we were made ‘responsible’ for these current realities, for which we are clearly not,

So let’s kick it and get lighter (and lighter...)

Salut, Rob

Founder of Council, theinternetofthings.eu http://theinternetofthings.eu/ @rob-vank

11.4 Community reactions to Devuan

```
Date: Sun, 19 Oct 2014 10:29:16 +0200 (CEST)
From: Emile ‘iMil’ Heitor
User-Agent: Alpine 2.11 (NEB 23 2013-08-11)
```

Hi, I don’t know who is behind that email, but sincerely, thank you for doing this. I’m an UNIX/Linux sysadmin for nearly 20 years, I am nowadays dealing with a 5k servers which consists of nearny 90% debian systems. I’ve been a long time opponent to systemd, first because I read the code (thing that too few of that crap’s zealots do), and ultimately because I tried it. That thing is a desktop toy, and even then, it has failed me on 50% of the cases. Its very nature is an abomination to UNIX principles and me and my team, colleagues, friends on the sysadmin field are *VERY* worried (to say the least) on what’s coming. Please keep that movement going, make it strong, and if you want a hand, count me in.

Again, thank you.

```
-------------------------------------------------------------
Emile ‘iMil’ Heitor *
   <imil@{home.imil.net,NetBSD.org,gcu.info}>

                                                            -
            | http://imil.net         | ASCII ribbon
              campaign ( )
            | http://www.NetBSD.org   |  - against HTML
              email  X
            | http://gcu.info         |              &
              vCards / \
```

```
From: "Tuschen, Alexander"
Subject: Do the right thing
Content-Type: text/plain; charset=US-ASCII; format=flowed
Organization: Zieger-Tuschen GbR (Hallenbeck.de)
Message-ID:
   <6ed75871bf7b3b024b0cd2eee4c56927@hallenbeck.de>
Content-Length: 716
Lines: 13
```

I'm on your side. I got in touch with Linux as sysadmin in 1997 and personally switched completely to Debian in 2005. I'm a child of the 80's, programming since the C64 and CPC464. Debian, through it's very open structure, has always satisfied my needs. But systemd is horrible.

I understand that there's a need for a modern parallelized init system, at last for phones and other mobile equipment. I understand these devices need to boot in seconds. But as Debian itself is mostly used on servers, boot-times are irrelavant. How often a thoroughly maintained Debian-server needs reboots? Only on Kernel-Updates or serious hardware problems (at least that's my experience). I want to keep it that way!

Alex

```
Date: Sun, 19 Oct 2014 07:13:07 -0500
From: Vince Mulhollon
```

I'm in, at least in general.

"because readability grants a certain level of power and consciousness for those among us who are literate"

I'd edit that to add that systemd is for software devs, but sysvinit is for sysadmins. Sysadmins can debug shell scripts, but not necessarily debug and recompile systemd, which will of course be necessary in the course of admin duties. Excessive product tying and complication for the sake of complication takes away the ability of admins to administer, which is highly counter productive. [...]

You missed an excellent lack of freedom argument. There will be only one way to do it and it will be mandatory and all else will be forbidden and only one guy gets to decide. Intentional incompatibility is a pretty screwed up way to go thru life. It does not help that most of the new features and abilities of systemd appear totally useless. [...]

And the embrace extend extinguish issue. If submarine software patents sink apache (unlikely, but possible) then I don't really care although the emergency

conversion to nginx might be a PITA for a day. When the one and only init system to bind them all is subverted either by submarine patent or security holes then the temporary work around will be to install the xyz package to switch to ... well I guess it'll be impossible and I'd have to switch to freebsd.

There is a cost - benefit ratio issue. Most of what systemd is capable of doing is unfortunately completely useless and irrelevant in comparison to what is being lost WRT to ease of debugging and reliability and security and freedom. If the costs were low/zero, the useless features it provides would likely still not be worth it, but at least the ratio would make it less awful. To some extent the whole situation is a farce. You've got software devs deciding what desktop users that don't exist want, while ignoring actual current desktop users, and they're re-applying that "successful" (LOL) development model to screw over sysadmins, and we're not subject to being told what we want by non-admins. Its a very arrogant business model.

```
Date: Sun, 19 Oct 2014 14:26:05 +0200
```

From: Henrik Sigurðsson

Thanks for setting up the Debianfork website. It summarizes my general feelings towards the SystemD adoption that has taken the distro world by storm. It is nice to know that I am not the only one who thinks this way.

I don't want SystemD for the same reason that I don't want Windows on my computer. I switched to Linux from Windows because I didn't like the thought of not being able to trust the OS, or not knowing what it was doing under the hood. SystemD has grown into such an incredible amount of modules with very complex interactions that it's being harder and harder to being able to understand how it works, which in turn makes me not being able to trust it.

It seems that the Red Hat developers are more interested in making a init system for the enterprise market and don't care if they break compatability with older or competing software, which makes it seem to me that SystemD is tailored to Red Hat distributions and should not have been adopted so rapidly by the distro world.

That's why I think SystemD shouldn't have been adopted as the default init system in Debian. I would rather go back to the old init system which is easily understood. Besides, SysV-init was never broken.

```
Date: Sun, 19 Oct 2014 23:37:54 +1100
From: Dan Fruehauf
```

I'm Dan Fruehauf, feel free to quote me if needed. I've been on Linux for over 15 years, I'm also a contributor in Fedora and author of NetworkManager-ssh (SSH plugin for Network manager). I'm a sort of a cross between a sysadmin and developer,

currently doing both.

I'm predominantly a Fedora user (and contributor) and am thinking recently about forsaking Fedora in favour of something without systemd (was thinking PCBSD). I've been around Linux for a while and understand the core philosophy behind it and how systemd just betrays all of that. I actually had an argument with Mr. Pottering on the Fedora-dev list about having /var/log/messages having a binary format. When log files in linux will not be plain text - it'll be the end in my opinion. needless to say Pottering didn't agree with me and tried to shut me up.

Pottering somehow managed to push systemd quite far in Fedora and at the moment I'm in absorption mode trying to digest WTF happened to my Fedora with systemd. Horrow show.

If this fork goes forward, I promise to try and do my best to contriube as much as possible. I'm good with packaging and shell, I believe I will be able to help. I think you guys hit the nail on the head with your web page and you have my full support.

Lets not let our voice be silenced,

```
Date: Sun, 19 Oct 2014 08:45:26 -0400
From: Steve van der Burg
```

As a veteran (20+ years; Solaris, HP-UX, RHEL, Debian) sysadmin who prefers Debian over everything for use on about 40 GUIless servers, it's either this (a fork away from systemd) or I move away from Debian altogether.

```
Date: Sun, 19 Oct 2014 16:21:30 +0300
From: WubTheCaptain
```

Thanks for doing this. I run four Debian servers in production, three of which are connected to an IRC network. Gnome remains the default DE on Debian for accessibility reasons, so it's obvious they have the monopoly of votes in favor of systemd. Large companies like Red Hat are also backing destroying the UNIX-philosophies.

Once I heard Debian would be making the switch to systemd, I've been slowly migrating all Debian servers over to OpenBSD over the impending death of Debian. There's still much that leaves me missing Debian, most importantly the large amount of different packages that are not available in OpenBSD and would take lots of effort to port over.

However, creating a Debian fork isn't just so easy. It requires massive amounts of dedication, not just from the developers like you but also from the community. It also seems GNU/Linux has headed a way that's no longer UNIX-like of "doing one thing and doing it well". Take a look at the GNU project, for example. Why do we

need 274-lines of code for an echo program, including the license header?

It's just absurd how things have changed in the past ten years, most things for the worse. Niche users are ignored as everything is being made more mainstream for the non-savvy users.

```
Date: Mon, 20 Oct 2014 09:06:05 +1100
From: Duncan Bayne
```

I like the idea of Debian not being tied to systemd - whether that comes around through voting or forking seems immaterial.

That said it's too late for me; the very fact that systemd is a likely prospect has motivated me to jump ship to FreeBSD. I'm already using it on my personal laptop, and will be migrating all my machines over to it shortly.

So, something to consider: some long-time (since 1995, in my case!) Linux users have already voted with their feet. The Debian team shouldn't necessarily take silence as consent, as the most deeply disaffected users may already have left.

In fact, a long-term FreeBSD user I know says there's been a spike of interest on the BSD lists since systemd was announced. I suspect I'm far from alone in having already departed.

```
Date: Mon, 20 Oct 2014 21:59:48 -0500
From: (Anonymous law attorney)
```

Please do fork it if necessary. I'm a longtime Linux user (1997) and feel screwed by the anti-user arrogance these Red Hat guys are showing.

I'm using Mint Mate 17 right now because of the Gnome guys and the KDE guys went crazy.

It's not a legal problem so I cannot sue them for you:)

Best wishes,

```
Date: Wed, 22 Oct 2014 21:53:57 -0400
From: Lee Hansen
```

There was a time, during the beginnings of my Linux adventures when any nasty complexity could be reduced to lucidity simply by reading for a few minutes. I remember spending days compiling and crashing and trying again and ending up with Gentoo. I remember stability in 9's. Throughout this entire evolution I have used Linux as my desktop of choice and to this day I refuse to use anything else. A few years ago I noticed the beginnings of the "adverse desktop mutation". When dev's began chucking out stability for a more groovy desktop experience, fashion over

function so to speak. Getting older and having less time I came to a crossroads, build another Gentoo box and get a divorce(not enough time) or move to Kubuntu - I'm still married but I miss the days of bomb proof stability, speed and easily overcome complexities. Enter systemd, seriously, wtf? Either I have burned more brain cells than I realize or it is the FSM in a kaleidoscope. Oh brilliant devs of OSS please spare us lowly sys admins from the epic pain of systemd, if you continue to make us use it we will have to relinquish our secret smiles as we will no longer be getting paid to play. At the very least give us a choice, as we chose to participate in this community specifically because of them.

```
Date: Mon, 20 Oct 2014 11:33:27 +0200
```

From: Anna Christina Naß

Hi,

Thank you for your intention to fork Debian when systemd becomes mandatory and the current poll is lost.

I'm using Debian as a server system for about 15 years now and I'm very much used to administering sysvinit-scripts and I like them and I like the robustness of this system.

You have my support for your idea!

AnnaChristina - acn128

(you may publish the names above, if you like)

Kind regards Anna Christina Naß, Germany

```
Date: Tue, 21 Oct 2014 07:43:07 +0200
```

From: Martin Waschbüsch

Hi there and kudos for you effort!

I do not feel strongly about either sysvinit or systemd or any of the alternatives. However, I do feel *very* strongly about being able to setup the systems I am an admin for in the way I w ant it. I *really* like that I am able to choose the shell, mta, editor, web server, database, etc. that I prefer working with. Why should the init system be different? Sometimes the software I chose turned out to be a bad idea. But in my book, the only thing worse than choosing the wrong software for a given task is to not have a c hoice at all.

```
Date: Mon, 20 Oct 2014 10:41:52 +0100
From: Michael Fyles
```

I am a programmer and part-time system administrator. I, too, have been considering whether or not a fork (given enough support) would be possible if Debian choose systemd exclusively. It is reassuring to see that others are thinking along the same lines. If the fork does need to go ahead, I would be very willing to assist.

```
Date: Mon, 20 Oct 2014 16:31:01 +0200 (CEST)
From: commandline@telenet.be
```

Thank you for this initiative, though i've only cross-read the counter-arguments they do appear valid and sufficiently substantiated by personal experience when it comes to such convoluted "one-in-all solutions" It is hard to believe an established company such as Red Hat is actually advocating it's use. Recently i've also learned about musl-libc which does seem to be a viable alternative for future system growth. At least in spirit and to some extent in proof already, not a perfect drop-in but the potential does seem obvious.

Debian is a META-OS, i was there on fidonet when it took shape and the idea has proven visionary in many ways. Choosing for a mono-init-system approach would make it just another distro. Though by now it could probably need some trimming or reorganisation in some places, overall quality seems to have degraded a bit here and there.

It is due time for a wiki dedicated to the design of a new init system, a massive community influenced design of such importance would prove significant in many ways. For non-critical systems systemd will have it's benefits but the "feel" is "awkward" even from an operational perspective. Let's just not fall into another flamewar based on fear and prejudices on both sides, maybe the authors of systemd are forthcomming in adapting to the community criticism and a win-win can be achieved.

Thank you for your consideration, Joris

```
Date: Mon, 20 Oct 2014 16:26:59 +0100
From: david
```

A wonderful idea. It would send a message that few could ignore that systemd is a serious power grab and an unwelcome one at that. I'm not a coder, or a sysadmin, just a happy Linux user of some 12 years. It was the philosophy behind Linux that attracted me and I see that philosophy now almost entirely sidelined and ridiculed, partially by the "our convenience is more import than your stupid philosophy" distros and also by the systemd bullies. So yes, as a user and long time fan - please fork Debian and give us back the Linux we love so much.

```
Date: Mon, 20 Oct 2014 16:28:04 +0100
```

From: tuxd3v

Hi folks,

I want to thank you in the first place, for what you are doing!! I am in the same situation like you are, and almost all the professionals out there!!

I am a sysadmin for some time now(before I was a programmer only), and I thought I was alone on this problem... I run a small park of servers(more or less 400 servers), for high availability... I prefere debian off course, but now I am in a difficult situation... because systemd will be banned in my systems, I DOENS'T ALLOW SYSTEMD HERE!!!!

Systemd doesn't allow us to control our process's in the same way that sysvinit does!!! I want control, and a simple control, not some binaries that I don't know exactly how they work underneath!

Systemd seem to be simple at the beginning, but when you want to adapt some software, to automate him, or to change the way it work on the system... ITS a NIGHTMARE!!!

Systemd is only good for desktop's ... NOT for mine, thanks!!

I doesn't care of speed in the boot process of my systems, because I boot them ONE Time in at least 2-5 years!!!!!!!!!!!!!!!!! And my desktop/laptop's I boot them almost of the time, one time only a day!!!

RUNLEVELS... what??

ABOMINATION what they have created , ABOMINATION!!!!!

RunLEVELS are one of the tons of beautiful things of Unix like systems.

Runlevels should be preserved at ANY cost!!! When I saw it I realized that the guys beguind systemd seem's to be kids... that's the only answer... sorry for my conclusion, but I think that not even 1 person with a minimal knowledge an respect for the Unix like culture, exist's there!!

How a programmer has the guts to kill Runlevels, or at least remove from the user the chance to use them explicitly!?

yes I Agree, when a attacker gains access to root, its a problem, because the logs are exposed... and in this I agree that are some work to do, but what they have done is so bad that I just can't accept it!!

Well, I write this email, that ends up being a rant, but the goal is to supportyour actions towards this degrading situation, and if the Debian committee doesn't care, encourage you guys to continue fighting this problem that seems to stole the, freedom and power, we have on our beloved Debian!

Cheers from Portugal and stay strong! tuxd3v

Date: Mon, 20 Oct 2014 12:02:53 -0500

```
From: Dan Saint-Andre
```

Colleagues, he recent discussions surrounding ‘systemd’ and ‘sysvinit’ have brought the “one thing done well” philosophy under renewed consideration. I would like us to consider the value of this philosophy in another situation – Linux™ distributions. Each week I see new distributions, each packaged to suit the preferences of yet another end-user community. Someone prefers a specific Desktop Environment. Someone else wants something different. Someone wants a specific initialization suite. Someone else wants something different. This diversity of preferences is a major strength of the Linux universe.

First, my humble definition of a Linux distribution. “A distribution is a packaged collection of software components: a kernel, drivers and modules, libraries, utilities, user interface, applications and so on.”

Why would anyone create a packaged collection of components? One motive might be a belief that their collection has benefits not offered by existing collections. Sadly, I don’t hear much discussion about the “one thing done well” aspects of these collections.

```
As a community, there is a tendency to treat a
   distribution as a
```

monolith, but that is far from the case. Consider “the kernel,” “drivers” and “modules.” [I call this the kernel suite.] As a collection, it is difficult to describe “one thing” that the suite needs to “do well.” I’ll try:

```
The kernel suite has the task of enabling the
   workstation hardware such
that multiple utilities and applications can be
   installed, configured
and run to deliver their designed benefits to end-users.
```

Notice that I use very few words to describe the “one thing done well.” Such a brief statement cannot be a specification or even a statement of requirements. However, these brief statements are crucial to any “one thing done well” evaluation.

```
When I look at software suites and seek their
  description of "one
```

thing done well" I usually find large documents. What would you write as a description for a software suite such as ‘systemd’ ‘upstart’ or ‘sysvinit’? Let’s consider the problem space in parts.

```
*Once the kernel is loaded and running with a suitable
   set of drivers
  and modules, we are able to load and start a
     process-one executable to
```

```
  take over the activation of utilities, applications
     and such. CHECK
  -- We can do this and do it well.
*Process-one needs to read configuration details and
   activate those
  utilities, applications and such. CHECK -- We can do
     this and do it
  well.
*'sysvinit' accomplishes the above in a handy way. This
   is especially
  true for workstations with a mostly static complement
     of utilities and
  applications. For workstations with a dynamic
     complement, there are
  some difficulties. Clearly beneficial for servers ...
     not so much for
  laptops.
*'systemd' does the job of handling the dynamic
   deployment of utilities
  and applications by responding to discovery events.
     Many of these
  same events happen during a cold start and so systemd
     also handles the
  initial deployment. Herein lies one possible source
     of confusion and
  conflict.Clearly beneficial for laptops ... not so
     much for servers.
```

Given that "the static case" and "the dynamic case" are two things, this appears to violate the "one thing done well" philosophy. When coupled with a large list of dependencies on supporting libraries and utilities, and we have a large and complex subsystem.

The Linux universe has other large, complex subsystems – web servers, email servers, database managers, etc. These don't generate the sort of divisive language found in the 'systemd' discussion. I seem to remember that they used to until enough time and effort allowed the benefits and liabilities to shake out and sound software engineering to refactor and otherwise address complexities of the early editions.

```
  Respectfully,
  ~~~ 0;-Dan
  Daniel M. St.Andre'
```

```
   Austin, TX USA


   --------------------
   [1] I use "universe" in the astronomical and
      cosmological sense rather
   than as an indicator of linux package availability
      made popular by certain
   distribution publishers.
```

```
Date: Mon, 20 Oct 2014 19:26:02 +0200
From: ch1qu1ta.lives
```

Hey,

people still focus on the wrong thing with this systemd debate. All the arguments are just minor things too me. This is what is particularly wrong with the whole thing:

There is a debate whether to replace legacy init-systems. It is a good debate, and imho a new init system is very due.

What should have been done (*):

1.) define interfaces/apis for a new init system by the linux community/process 2.) standardaize these interface 3.) have somebody provide a reference implementation and reference-test-suite (an init-system is missing critical, I cannot debug umteenth servers when they fail initing)

what has been done:

1.) a reference implementation has been pooped into existence with interfaces/apis 'designed' on the fly 2.) this mix of standards/implemention has then been pushed and force-fed to the community 3.) now the community is pissed

The discussion about whether or not systemd must be used is moot. If standards exist, systemd can be replaced. If not, like we have now, it cannot.

So, stop the stupid debate, and really get down to solving this problem (see *).

```
Date: Mon, 20 Oct 2014 14:03:06 -0500
From: Matthew Alton
```

Hi. I'm a professional Unix and Linux systems admin. I work at Magellan Healthcare in the operations department. My title is Senior System Administrator. I've been doing this job at various places of employment for 19 years. I've been involved with Unix since 1988. I have a beard. It is quite gray. Please keep me in the loop on the Debian fork. I will help preserve our way of life if I can.

Thanks.

```
   Matthew Alton
   UNIX Systems Programming & Administration
   "Beware of bugs in the above code; I have only proved
      it correct, not
   tried it." -- Donald Knuth
```

```
Date: Mon, 20 Oct 2014 16:38:20 -0400
From: Peter Fedorow
```

THANK YOU

I am Systems Manager for an installed base of over 2000 kiosks running an OS built on Ubuntu. I've ran Debian and Ubuntu on my home and work desktops, laptops, and servers since I ditched OS/2 in the late 90s.

I have the Unix philosophy pasted on my wall. Most notable here is: - Make each program do one thing well. - Store data in flat text files. - Use shell scripts to increase leverage and portability.

The developer across from me runs Fedora and deals with his workstation breaking every day or three due to systemd bugs. He's had enough too.

My advise – open up the collection jar to finance the birth of DebianFork.

```
Date: Mon, 20 Oct 2014 22:59:33 +0200
From: Alfredo Mungo
```

Hi,

I've been a GNU/Linux user and developer since I was a kid, when the kernel crashed now and again. I grew up with open source and the community-oriented feeling that distinguishes us from the rest of the cyberspace and I find absurd that "someone" is pushing systemd so hard, that many distributions have chosen to forsake that spirit that makes us what we are. We should not forget that we ARE the alternative, but without freedom of choice, what alternative can we be?

For long time I've been an Archlinux user and now that they (and many others) enforced systemd, I'm working on my own GNU/Linux system to have my freedom back. With your reaction you are showing the entire community that the spirit of choice is not dead.

Hope your project succeeds, you have all my support.

```
Date: Wed, 22 Oct 2014 06:22:35 -0600
From: John Nanney
```

Hi guys,

I'm a software engineer and sometime system admin, and I'd love to help out.

Linux has always been full of choice, from window manager to editor and even down to shell. Freedom of choice makes Linux distros great, and making software depend on the init system is simply bad design.

```
Date: Thu, 23 Oct 2014 15:41:31 -0500
From: Jim
```

I've used Debian on my desktop and internet servers since the very start of Debian. What I liked is that Debian always took a strong engineering-based approach to the OS, with package management done right, continuous upgrades between versions, and lots of choice. However, if they don't maintain a free choice of init systems, I will drop Debian for some other distribution or OS. Poettering's attitude is appalling and contrary to everything I care about in an OS. Somebody has to take a stand to avoid lock-in, and I sincerely hope that Debian will do that since it has a history of caring about good engineering and freedom of choice.

```
Date: Mon, 20 Oct 2014 18:01:06 -0400
From: Rich Kulawiec
```

I've been using 'nix since Research Unix v6 in the late 70's. My code's in Berkeley Unix, SunOS/Solaris et.al., and various Linux distributions. I run OpenBSD, Mint Linux, FreeBSD, CentOS Linux, Solaris, and half a dozen other distributions at the moment.

The rush to make systemd the end-all be-all is quite disturbing to me, not only because it flies in the face of the Unix and "Software Tools" philosophy – a tool should do one thing and do it well – but because there's seems to be quite a rush to shove it into distributions without carefully considering its long-term impact – including the security implications. A powerful tool is useful; but it's useful for *everyone*, including adversaries, and the more functions that are crammed into systemd, the more attractive a target it will become.

I'm not saying that the functionality offered by systemd is a bad thing: it's not. I'm saying that architecturally superior ways of providing that functionality [mostly] exist, and those that don't should be and can be implemented by far less complex, opaque and unwieldly tools.

It would be a *serious* strategic error to force its use in Debian – or in any other distribution for that matter. I don't see a problem offering it as alternative, but it should NOT be the default and it should NOT be required.

Those with contrary views are strongly encouraged to read "Software Tools" by

Kernighan and Plauger, which is arguably – still – one of the very best books for anyone engaged in any form of software development, and should be mandatory reading for anyone doing so in the 'nix world.

```
---rsk
```

```
Date: Tue, 21 Oct 2014 14:42:51 -0500
From: Michael Faurot
```

I've been working with and administering Unix systems since System III (yes, that's not a typo :). I have always appreciated the ethos of the founding fathers of Unix:

```
...do one thing and do it well...
```

I use Linux for servers–no pointy clicky–just the command line. Let's keep init as it has always been, and leave systemd as an option for those that actually want it.

```
Date: Tue, 21 Oct 2014 00:46:57 +0100
From: Richard Neill
```

I think what you are doing is quite important.

I originally did like systemd, because normal initscripts are really hard to write well and portably (for example, try running a second instance of $daemon_of_choice and see whether "service daemon stop" kills your personal copy as well as the main one). Also, the "service $daemon status" messages under systemd are much more helpful.

I find systemd inelegant because it is so badly cross-coupled. For example, why does it need to replace cron? (there may be a separate case to be made for a new version of cron, but not one that belongs in the init system). The whole thing doesn't feel very Unixy.

Even desktop users have no real benefit from a few second reduction in boot (I only reboot for kernel updates and power failures), and I don't mind too much if postgres and apache aren't running for the first 5 seconds after I log in! Likewise, laptop users mostly need good suspend, rather than good reboot.

HTH, Richard

```
Date: Tue, 21 Oct 2014 03:16:00 +0200 (CEST)
From: Manfred Wassmann
```

Hello,

I switched to Debian with the release of 0.93R5 and I kept loyal regardless of its degradation through ubuntification. But a switch to systemd is a no go. If a fork should prove necessary, I'm with you.

```
Date: Mon, 20 Oct 2014 21:36:12 -0400
From: Lee Winter
```

First, the topic of systemd is not one for discussion IMHO because I noticed that systemd is now showing up in the crypto packages. For me that's a third-rail type of non-starter. I will not ever use a distribution that has such a massive code base to review as part of crypto acceptability. So talking about it is a complete waste of bits.

Note that the above issue is *not* philosophical support for "the unix way". It is a real-life matter of practicality. Small tools represent incremental additions that need to be reviewed. Massive code bases cannot be effectively reviewed, so people don't even bother. See Micros$_{1\ \text{Window}}$1 for a good example.

Secondly, since managing dependencies is key to an effective startup sequence, I suggest that people consider DJB's redo system as a model. It is very simple. And very effective.

Why should anyone care what I have to say? They shouldn't. Instead they should care about what they think after they read what I have to say. I've been a developer for over 40 years. I delivered my first high-performance, high-tech software system over 30 years ago (pointing lasers out of airplanes for NASA). I've worked at all levels of scale and all levels of development from nanocode to horizontal and vertical microcode to dozens of assemblers to C/C++ to more than a dozen "high level" programming languages and several languages so high their level isn't named that I know of. I administer several hundred machines ranging from 16KB Z80s, up to multi-CPU, many-core, NUMA machines with steaming heaps of main memory. Hint: I buy memory by the /pound/.

But I am not a debian maintainer or developer. Just an avid user. I don't expect that to change, but the future is a mystery to me, so anything is possible.

I wish you success (luck is for losers),

Lee Winter Nashua, New Hampshire United States of America

```
Date: Tue, 21 Oct 2014 13:44:27 +1100
From: Owen Genat
```

Hi. Just wanted to add my support to those behind this website and those others gathering in opposition to systemd.

I am noone of significance (home user). I came to GNU/Linux about 15 years ago and have tried several flavours, with Slackware perhaps being most memorable. I

now (past 4-5 years) run Debian on a small home server and on the couple of clients on the same network. My customisations and CLI-fu are moderate.

I remember at first feeling awkward with the manner/approach of GNU/Linux, but over time I have come to appreciate the work done by those have gone before tremendously. I tend to work a fair bit with text files and find the ubiquity of handling this format constantly impressive. I also love the UNIX philosophy of small tools that can be combined to produce custom solutions. While I could probably transition to the upcoming systemd-based version, I believe, among other things, it impairs my ability to learn how GNU/Linux works.

I have watched the unfolding systemd saga with interest and a growing sense of dis-ease. It appears corporate ideals (in particular, homogeneity) have usurped the diversity that sits at the heart of the UNIX philosophy. The cavalier attitude towards existing installations (and the work of those that have gone before) is also repugnant. I remain hopeful that the Free Software community will route around this problem, but it appears something of a David vs. Goliath situation.

Best wishes, Owen.

```
Date: Tue, 21 Oct 2014 01:31:40 -0400
From: Ryan McGuigan
```

I don't actually pray, but if I did, I would be praying for a group of people willing and able to fork debian to prevent the total abortion that is systemd from absolutely destroying the entire future of Linux. systemd really is THAT bad. This current situation is the stuff of nightmares.

[...]

Replacing simple scripts with C code, which cannot easily be reviewed or altered by sysadmins in unique situations that may arise. I have had so many occasions over 20 years of *nix work where being able to make adjustments to init scripts was a huge life saver in unique situations and configurations. I've even had an occasion where I built a complete init system using busybox(IIRC) and wrote a complete set of startup scripts in one day for a series of Linux-based appliances for a small company. It was mostly compatible with SysV-style startup scripts - in fact I used the unaltered SysV startup scripts of all the services I used for the project. I put together the whole init system in one day, and it was simple and easy. That was a very fun project, and it all started with the init system that was designed precisely for the unique custom solution I had to develop. It would be impossible to implement with systemd. Not only would the custom init system be impossible to implement, but the logging facilities of systemd wouldn't have worked either. With the way things are being made to depend on systemd, should that fuck Poettering have his way, things like that may no longer be easy or even possible in the future.

[...]

```
Date: Tue, 08 Dec 2015 22:59:53 +0100
From: root@tehtrb
```

Thank you for all that you are doing. My greatest wishes and hopes are with you guys. You are doing extremely important work. Please do take note of my gratefulness, and please do remind yourselves that for each person speaking up, thousands of others think the same thing to themselves. Whatever deity you may pray to, you are doing its work. Thank you.

```
Date: Tue, 21 Oct 2014 09:43:21 +0200
From: "Arne Babenhauserheide (IMK)"
```

Hi,

I just wrote an update to my systemd troubles article with a link to your site and the general resolution. Thank you for taking this up! → http://draketo.de/light/english/top-5-systemd-troubles

(besides: that systemd is controversial shows clearly in my access stats: The top 5 systemd troubles article has been the most accessed article on my site for weeks, if not months)

```
Best wishes,
Arne
--
Doktorand
Gruppe: GHG

Karlsruher Institut fuer Technologie
```

```
Date: Tue, 21 Oct 2014 10:21:04 +0200
From: Eckard Brauer
```

Hello there,

thanks a lot for your engagement! I've been a UNIX sysadmin for the last 25 years now for different companies and on different flavours. That whole discussion about changing the init system is IMHO really about (further) changing the UNIX system to somthing fundamentally (in more than one single sense, indeed) different, what I don't see the need for. So as already written, let them check options for people who like or really need that, but let the system remain as we all know it – don't build a Babel tower.

```
Date: Tue, 21 Oct 2014 11:47:06 +0300
From: Johan
```

Hi,

I think the approach you are taking is great. While I may still even consider a distro that defaults to systemd, I want the choice to be mine. Maybe it could be managed as a package set like the choice of desktop for many distros (*buntu, mint).

Right now, I am planning to build a cloud. The current generation of stable releases are not bound to systemd, but choosing one that will not be bound to a specific init in the future either, is key, in order avoid the burden of a huge migration if that specific init go south.

Please keep me posted about how things move on. Please feel free to quote anonymously.

```
Date: Tue, 21 Oct 2014 12:08:52 +0200
From: Florian Lohoff
```

Hi, i am DD and one of the supporters of Ian Jacksons proposal. I am professional sysadmin and software developer for more than 20 years and i am using Debian from the very early relases. systemd is unacceptable for me because it doesnt restrict itself for beeing a init replacement but rather gobbles up myriads of basic unix system functionality up to the point in offering a “hostname” DBUS functionality. Think about it - Using thousands of LOC to request the hostname via DBUS instead of opening a file or using a glibc hostname(2) call. I am not even asking what the init has to deal with a hostname or a hosts icon (see systemd-hostnamed). I dont care about gnome depending on certain functionality in systemd to be operable. We (as Debian) have had issues in the past where we decided not to follow upstreams decision (e.g. the glibc vs eglibc case). Debian is not only using upstream but influencing upstream. Telling the GNOME ecosystem we dont like the architecture of systemd would send a strong signal.

I am willing to support and work for a Debian fork.

```
Date: Tue, 21 Oct 2014 09:24:07 -0400
From: David
```

I’m been solely using Debian for over 10 years. Both on my home desktop and on my web server. From the very beginning, I wrote my own simple init system. All it does is run a script at boot up. I just want my computer to run a list of commands when it boots up and nothing else. I don’t need any more features than that.

What I love about the sysvinit scripts that come with Debian packages is that I can easily just call them from my script if I want any of them to run at boot time. And if something goes wrong, I can examine those scripts to see the problem, or make changes to them.

I just came across your website today, and after learning about systemd from the links on your site, it is clear that there is no way I am ever going to use it. The boycott systemd website was the most influential and convinced me completely that systemd is incredibly awful.

Since I'm running Debian testing on my desktop, I decided to look at the packages, and there it was, systemd was installed automatically somehow when upgrading my packages! I've installed Debian once, 10 years ago, and was just upgrading the packages occasionally ever since then, so package upgrades was the only way systemd was able to sneak into my computer. It feels like my computer was infected by a bad virus. I quickly removed the systemd package and all packages that depended on it without issues. However, I noticed libsystemd0 was still installed and doing a reverse dependency check showed that there are lots of important packages that indirectly depended on it because of dbus depending on it. That is clearly a bug, and I hope Debian can fix this issue. Normally, I wouldn't object to having some library installed, but from what I read about systemd from the links on your site, it's so awful that I don't even want its library installed on my computer.

I really hope Debian doesn't create more dependencies on systemd or its library, because I will NEVER run systemd. If, in the future, Debian has so many dependencies on systemd that I can no longer run a Debian system without running systemd, I will be forced to switch away from Debian after more than 10 years.

- Humanoid

```
Date: Tue, 21 Oct 2014 16:42:12 +0200
From: Jakub 'Eremiell' Marek
```

Hi!

I just today learned about your initiative to fork Debian and I can't agree more.

I haven't been using Debian for so long, as some other people who already wrote you, only about 5 years, but it grew to my heart for it's efficiency, pureness and power. No other distro I tried made me feel so home.

Unfortunately, some late changes made be really uncomfortable. Every day, I feel I have less and less control over my system up to point, where I decided switching a distro. I tried, but none felt half as effective as what I was used Debian to be.

Honestly, I was thinking about just the same thing. What if we just fork the whole thing? Start anew. Draft rules, which reflect real needs of real users. Start by

packaging just the things, we need and invite anyone to package what they need. But I wasn't sure, if the we wasn't just I. I'm more than excited to know, that I don't feel that way alone.

For me, using free software is about freedom. Freedom to choose my tools at the most granular level. Don't have to swallow, what someone else decided to be best for me. Even hack, modify or write my own tools when necessary. Have control over my computer, not let computer have control over me.

This abomination, systemd, broke about every essential freedom, we used to have. I'm not saying to ban it. It's everyone's free choice and those who want it should have the option. But there should be all the other options for those, who don't. The arrogance of some people connected to systemd is unprecedented.

With all of that said, I believe, there might be an easier solution than a complete fork. Better support for OpenRC, Upstart, all the alternatives, even old good SysVinit, if anyone wants to use that. But I see less and less possibility for such movement. Reading some threads and seeing the approach of many people makes me sad and disgusted.

I'd be happy to accept any solution. But if you decide to go to Mordor, you've got my keyboard. Drop me a note and I'll see what I can do.

Yours sincerely,

Jakub 'Eremiell' Marek

```
Date: Wed, 22 Oct 2014 10:16:56 +0200
From: Lada 'Ray' Lostak
```

Just one small +1

Thank you keeping word a bit better :)

```
--
Best regards,
Lada 'Ray' Lostak
Unreal64 Develop group
http://www.unreal64.net
http://www.orcave.com



-----------------------------------------------------------------
In the 1960s you needed the power of two C64s to get a
   rocket
to the moon. Now you need a machine which is a vast
   number
```

```
of times more powerful just to run the most popular GUI.

                Imagination is more important than
                   knowledge...
```

```
Date: Thu, 23 Oct 2014 13:11:49 -0400
From: (Anonymous, business intelligence professional)
```

I'm with you guys. I've often been asked why I prefer administrating Unix and its derivatives vs Windows and I usually explain it this way. Any idiot can install Windows Server but it takes a genius to debug it if you have a problem. Unix requires a reasonable about of technical knowhow to install but that same level of knowledge is normally sufficient to solve problems. KISS over obscuration. We don't need Winux.

```
Date: Wed, 29 Oct 2014 10:03:30 -0400
From: Kene Meniru
```

Hello people: I am writing in support of your effort to retain sysvinit as the default in debian instead of systemd. I am a programmer and user of debian. I am watching and listening and will lend a hand if there is a need, to fork debian. Thank you. Sincerely: Kene Meniru, Ph.D., Assoc. AIA

```
Date: Mon, 10 Nov 2014 12:45:36 +0100
From: Alexandre
```

Hello guys,

You rock ! I am appalled to see how the crowd is sheepishly embrassing that nightmare of systemd, and how one needs to state the obvious regarding the unix philosophy, as you do very well... If/when it sees the light, I will promote the fork with all my might in my 150k employee company.

11.4.1 Sysvinit scripts vs. systemd service files

Here's a comparison of sendmail sysvinit init script and the corresponding systemd service file, taken from the pro-systemd fedorafork.org parody site.

11.4.1.1 sendmail sysvinit unit

This code below, known to many professionals, which has been used in the past 20 years to start the UNIX mail daemon sendmail on SysV unix and derivates. It is several lines long and it demonstrates every single step from the loading of clear text

files with configuration directives to their processing. It is shown here with syntax highlight to facilitate the syntactical recognition of different elements.

```
#!/bin/bash
#
# sendmail      This shell script takes care of starting
   and stopping
#               sendmail.
#
# chkconfig: 2345 80 30
# description: Sendmail is a Mail Transport Agent, which
   is the
#              program that moves mail from one machine
   to another.
# processname: sendmail
# config: /etc/mail/sendmail.cf
# pidfile: /var/run/sendmail.pid

# Source function library.
. /etc/rc.d/init.d/functions

# Source networking configuration.
[ -f /etc/sysconfig/network ] && . /etc/sysconfig/network

# Source sendmail configureation.
if [ -f /etc/sysconfig/sendmail ] ; then
    . /etc/sysconfig/sendmail
else
    DAEMON=no
    QUEUE=1h
fi
[ -z "$SMQUEUE" ] && SMQUEUE="$QUEUE"
[ -z "$SMQUEUE" ] && SMQUEUE=1h

# Check that networking is up.
[ "${NETWORKING}" = "no" ] && exit 0

[ -f /usr/sbin/sendmail ] || exit 0

RETVAL=0
```

```
prog="sendmail"

start() {
    # Start daemons.

    echo -n $"Starting $prog: "
    if test -x /usr/bin/make -a -f /etc/mail/Makefile ;
       then
      make all -C /etc/mail -s &rt; /dev/null
    else
      for i in virtusertable access domaintable
         mailertable ; do
        if [ -f /etc/mail/$i ] ; then
        makemap hash /etc/mail/$i < /etc/mail/$i
        fi
      done
    fi
    /usr/bin/newaliases &rt; /dev/null 2&rt;&1
    daemon /usr/sbin/sendmail $([ "x$DAEMON" = xyes ] &&
       echo -bd) \
            $([ -n "$QUEUE" ] && echo -q$QUEUE)
               $SENDMAIL_OPTARG
    RETVAL=$?
    echo
    [ $RETVAL -eq 0 ] && touch /var/lock/subsys/sendmail

    if ! test -f /var/run/sm-client.pid ; then
    echo -n $"Starting sm-client: "
    touch /var/run/sm-client.pid
    chown smmsp:smmsp /var/run/sm-client.pid
    if [ -x /usr/sbin/selinuxenabled ] &&
       /usr/sbin/selinuxenabled; then
        /sbin/restorecon /var/run/sm-client.pid
    fi
    daemon --check sm-client /usr/sbin/sendmail -L
       sm-msp-queue -Ac \
            -q$SMQUEUE $SENDMAIL_OPTARG
    RETVAL=$?
        echo
        [ $RETVAL -eq 0 ] && touch
```

```
          /var/lock/subsys/sm-client
        fi

    return $RETVAL
}

reload() {
    # Stop daemons.
        echo -n $"reloading $prog: "
    /usr/bin/newaliases &rt; /dev/null 2&rt;&1
    if [ -x /usr/bin/make -a -f /etc/mail/Makefile ];
       then
      make all -C /etc/mail -s &rt; /dev/null
    else
      for i in virtusertable access domaintable
         mailertable ; do
        if [ -f /etc/mail/$i ] ; then
        makemap hash /etc/mail/$i < /etc/mail/$i
        fi
      done
    fi
    daemon /usr/sbin/sendmail $([ "x$DAEMON" = xyes ] &&
       echo -bd) \
        $([ -n "$QUEUE" ] && echo -q$QUEUE)
    RETVAL=$?
    killproc sendmail -HUP
    RETVAL=$?
    echo
    if [ $RETVAL -eq 0 -a -f /var/run/sm-client.pid ];
       then
        echo -n $"reloading sm-client: "
        killproc sm-client -HUP
        RETVAL=$?
        echo
    fi
    return $RETVAL
}

stop() {
    # Stop daemons.
```

```
    if test -f /var/run/sm-client.pid ; then
        echo -n $"Shutting down sm-client: "
        killproc sm-client
        RETVAL=$?
        echo
        [ $RETVAL -eq 0 ] && rm -f /var/run/sm-client.pid
        [ $RETVAL -eq 0 ] && rm -f
           /var/lock/subsys/sm-client
    fi
    echo -n $"Shutting down $prog: "
    killproc sendmail
    RETVAL=$?
    echo
    [ $RETVAL -eq 0 ] && rm -f /var/lock/subsys/sendmail
    return $RETVAL
}

# See how we were called.
case "$1" in
  start)
    start
    ;;
  stop)
    stop
    ;;
  reload)
    reload
    RETVAL=$?
    ;;
  restart)
    stop
    start
    RETVAL=$?
    ;;
  condrestart)
    if [ -f /var/lock/subsys/sendmail ]; then
        stop
        start
        RETVAL=$?
    fi
```

```
    ;;
  status)
    status sendmail
    RETVAL=$?
    ;;
  *)
    echo $"Usage: $0
       {start|stop|restart|condrestart|status}"
    exit 1
esac

exit $RETVAL
```

11.4.1.2 sendmail systemd unit

This is the "systemd unit" to start sendmail in the new system advertised by forkfedora.org. It does not consists of shorter code, but of a short listing of assignments. In fact this unit is made only of variable assignments, while the code processing them is the compiled systemd software which is stored as a binary object on the system. To read and modify the startup logics sendmail as operated by systemd, one should download its source (stored separately and significantly bigger) and eventually recompile it for deployement.

```
[Unit]
Description=Sendmail Mail Transport Agent
After=syslog.target network.target
Conflicts=postfix.service exim.service
Wants=sm-client.service

[Service]
Type=forking
PIDFile=/run/sendmail.pid
Environment=SENDMAIL_OPTS=-q1h
EnvironmentFile=-/etc/sysconfig/sendmail
ExecStartPre=-/etc/mail/make
ExecStartPre=-/etc/mail/make aliases
ExecStart=/usr/sbin/sendmail -bd $SENDMAIL_OPTS
   $SENDMAIL_OPTARG

[Install]
WantedBy=multi-user.target
Also=sm-client.service
```

Bibliography

Agamben, G. (2011) *The kingdom and the glory: For a theological genealogy of economy and government*. Stanford University Press.

Agamben, G. (2005) The state of exception. *Politics, Metaphysics, and Death*. 284–297.

Ahn, L. von (2006) Games with a purpose. *IEEE Computer Magazine*. 96–98.

Alexander, C. (1977) *A pattern language: Towns, buildings, construction*. Oxford University Press.

Amoore, L. (2011) Data derivatives: On the emergence of a security risk calculus for our times. *Theory, Culture & Society*. 28 (6), 24–43.

Angwin, J. et al. (2016) *Machine bias: There's software used across the country to predict future criminals. and it's biased against blacks*.

Antoniol, G. et al. (2005) Towards the integration of versioning systems, bug reports and source code meta-models. *Electr. Notes Theor. Comput. Sci.* 127 (3), 87–99.

Arikan, B. & Erdogan, E. (2008) *User labor: A framework for sustaining user labor across the web*.

Arvidsson, A. (2016) Facebook and finance: On the social logic of the derivative. *Theory, Culture & Society*. 21 (1),.

Ascott, R. (2001) Arts education @ the edge of the net: The future will be moist! *Arts Education Policy Review*. 102 (3), 9–10.

Ascott, R. (1990) Is there love in the telematic embrace? *Art Journal*. 49 (3), 241.

Ascott, R. (1999) *Reframing consciousness*. Intellect Books.

Austin, J. L. (1975) *How to do things with words*. Oxford university press.

Batina, L. et al. (2010) 'Developing efficient blinded attribute certificates on smart cards via pairings', in *International conference on smart card research and advanced applications*. 2010 Springer. pp. 209–222.

Beck, K. et al. (2001) *Manifesto for agile software development*.

Benjamin, W. (2008) *The work of art in the age of mechanical reproduction*. Penguin UK.

Benkler, Y. (2007) *The wealth of networks: How social production transforms markets and freedom*.

Bianchi, A. & Roio, D. J. (2011) Frames from the life and death of jean charles de menezes. *Computers, Privacy and Data Protection: an Element of Choice*. 101–109.

Bohm, D. & Nichol, L. (2004) *On creativity*. Vol. 13. Psychology Press.

Bovermann, T. & Griffiths, D. (2014) Computation as material in live coding.

Computer Music Journal. 38 (1), 40–53.

Braga, E. & Fumagalli, A. (2015) *La moneta del comune*. Alfabeta edizioni e Derive Approdi.

Bratton, B. (2016) *The stack: On software and sovereignty*. computer software studies. MIT Press.

Brock, K. (2012) One hundred thousand billion processes: Oulipian computation and the composition of digital cybertexts. *Technocu ltur e*. 2.

Cabello, F. et al. (2013) Towards a free federated social web: Lorea takes the networks. *Unlike Us Reader: Social Media Monopolies and Their Alternatives*. 338–346.

Campesina, V. & others (2003) What is food sovereignty. *Via Campesina*.

Carettoni, L. & Laniado, D. (2005) Etica hacker: L'imperativo e' hands-on. *Testo reperibile*.

Caronia, A. & Bianchi, A. (eds.) (2012) *Michel foucault, per una genealogia del soggetto*. Nuova Accademia di Belle Arti, Milano. [online]. Available from: https://archive.org/details/MichelFoucault_PerUnaGenealogiaDelSoggetto.

Castagné, N. (2014) Software Technologies of Multisensory and Instrumental Modelling, Simulation and Interaction for Creation. Accreditation to supervise research thesis. Grenoble INP, Université de Grenoble.

Coleman, E. & Hill, B. (n.d.) The social production of ethics in debian and free software communities. *Free/Open Source Software Development*. 273–295.

Coleman, G. (2009) Code is speech: Legal tinkering, expertise, and protest among free and open source software developers. *Cultural Anthropology*. 24 (3), 420–454.

Collins, N. (2011) Live coding of consequence. *Leonardo*. 44 (3), 207–211.

Crawford, M. (2010) *Choice Reviews Online*. 48 (04),.

Da Rimini, F. (2011) Networking the container project: A radical approach to digital literacy, creativity and social change'. *Acoustic Space*. Special issue on 'Networks and Sustainability' (10), 157–172.

De Souza, P. (2010) Rethinking the dissension between software and generative art. *The International Journal of Technology, Knowledge and Society*. 6 (5), 13–26.

Dekker, A. (ed.) (2011) *The greater cloud exhibition*. 6. Metropolism magazine.

Deleuze, G. & Guattari, F. (2004) *Anti-oedipus*. A&C Black.

Dencker, K. (2011) *Optische poesie: Von den prähistorischen schriftzeichen bis zu den digitalen experimenten der gegenwart*. De Gruyter.

Desmarais, A. A. (2003) The via campesina: Peasant women on the frontiers of food sovereignty. *Canadian Woman Studies*. 23 (1), 140.

Desouza, K. C. & Vanapalli, G. K. (2005) Securing knowledge in organizations: Lessons from the defense and intelligence sectors. *International Journal of Information*

Management. 25 (1), 85–98.

Diakopoulos, N. (2016) Accountability in algorithmic decision making. *Commun. ACM*. 59 (2), 56–62.

DiMaggio, P. J. (1988) Interest and agency in institutional theory. *Institutional patterns and organizations: Culture and environment*. 13–22.

Ek, R. et al. (2007) 'Revisiting foucault through reading agamben: Implications for workplace subjectification, desubjectification and the dark side of organizations', in *5th critical management studies conference, machester*. 2007 pp. 11–13.

Ellison, G. & Fudenberg, D. (2000) The neo-luddite's lament: Excessive upgrades in the software industry. *The RAND Journal of Economics*. 253–272.

Emerick, C. et al. (2012) *Clojure programming - practical LISP for the java world*. O'Reilly.

Ernst, N. A. et al. (2010) Code forking in open-source software: A requirements perspective. *CoRR*. abs/1004.2889.

Farrel, K. et al. (2012) *Beyond reductionism. a passion for interdisciplinarity*.

Federici, V. (2015) The rise of rojava: Kurdish autonomy in the syrian conflict. *SAIS Review of International Affairs*. 35 (2), 81–90.

Flores, A. W. et al. (2016) *False positives, false negatives, and false analyses: A rejoinder to 'machine bias: There's software used across the country to predict future criminals. and it's biased against blacks.'*

Foucault, M. (1966) *Les mots et les choses: Une archéologie des sciences humaines: Une archéologie des sciences humaines*. Gallimard.

Foucault, M. (2008) Edited by Michel Senellart ; translated by Graham Burchell. ISBN 978-1-403-98654-2. *The birth of biopolitics*. Lectures at the Collège de France, 1978-1979. Palgrave Macmillan.

Foucault, M. (1991) *The foucault effect: Studies in governmentality*. University of Chicago Press.

Fumagalli, A. et al. (2009) *La gran crisis de la economía global. Mercados financieros, luchas sociales y nuevos escenarios politicos*. Traficante de Sueños (Madrid).

Fung, K. H. et al. (2012) 'Social forking in open source software: An empirical study', in Marite Kirikova & Janis Stirna (eds.) *Proceedings of the caise'12 forum at the 24th international conference on advanced information systems engineering (caise), gdansk, poland, june 28, 2012*. CEUR workshop proceedings. 2012 CEUR-WS.org. pp. 50–57.

Gehl, R. W. (2014) *Reverse engineering social media: Software, culture, and political economy in new media capitalism*. Temple University Press.

Georgescu-Roegen, N. (1971) *The entropy law and the economic process*. Harvard

University Press.

Ghose, J. R. (2004) *Saving seeds or saving face? Seed acquisition mechanisms among farmers in jharkhand, india and sui generis protection*.

Gluhak, A. & Kranenburg, R. V. (2012) 'New instruments of governance for our societies', in *Tenth annual IEEE international conference on pervasive computing and communications, percom 2012, march 19-23, 2012, lugano, switzerland, workshop proceedings*. 2012 IEEE Computer Society. pp. 191–196.

Golumbia, D. (2009) *The cultural logic of computation*.

Goriunova, O. (2016) Participatory platforms and the emergence of art. *A Companion to Digital Art*. 297–309.

Graeber, D. (2001) *Toward an anthropological theory of value*.

Greimas, A. J. (1983) *Structural semantics: An attempt at a method*.

Grover, A. (2006) Curatorial statement. *Phantom captain: Art and crowdsourcing*.

Guo, H. (2016) Lean but not mean ux: Towards a spiral ux design model. *Lecture Notes in Computer Science*. 25–33.

Hakken, D. (2010) Computing and the current crisis: The significant role of new information technologies in our socio-economic meltdown. *Journal for a Global Sustainable Information Society*. 205–220.

Hammouda, I. et al. (eds.) (2012) *Open source systems: Long-term sustainability - 8th IFIP WG 2.13 international conference, OSS 2012, hammamet, tunisia, september 10-13, 2012. proceedings*. IFIP advances in information and communication technology. Vol. 378. Springer.

Hardt, M. & Negri, A. (2009) *Commonwealth. harvard*.

Hardt, M. & Negri, A. (2005) *Multitude: War and democracy in the age of empire*. Penguin.

Hayles, N. K. (2010) *My mother was a computer: Digital subjects and literary texts*. University of Chicago Press.

Horst, P. (2008) Code poetry's aesthetic (ww) web: Examples of contemporary intermedial experiments. *Another Language: Poetic Experiments in Britain and North America*. 93.

Howe, J. (ed.) (2006) *The rise of crowdsourcing*. Wired.com.

Hui, Y. (2016) *On the existence of digital objects*. University of Minnesota Press.

Keet, C. M. (2013) 'Open world assumption', in *Encyclopedia of systems biology*. Springer. pp. 1567–1567.

Kennedy, M. (2016) Neues geld. über eine strukturelle problematik unseres geldwesens. *Die Zwischengesellschaft*. 201–212.

Kokoris-Kogias, E. et al. (2016) Enhancing bitcoin security and performance with

strong consistency via collective signing. *arXiv preprint arXiv:1602.06997*.

Korzybski, A. (1951) 'The role of language in the perceptual process', in *Perception. an approach to personality*. Ronald Press.

Lampo, L. et al. (2005) *Connessioni leggendarie. net. art 1995-2005*. Domenico Quaranta.

Latour, B. (2012) *We have never been modern*. Harvard University Press.

Lazzarato, M. (1997) *Lavoro immateriale: Forme di vita e produzione di soggettività*. Ombre corte.

Lessig, L. (1999) Code is law. *The Industry Standard*. 18.

Lietaer, B. (2001) The future of money: Towards new wealth, work and a wiser world. *European Business Review*. 13 (2),.

Liikkanen, L. A. et al. (2014) Lean ux. *Proceedings of the 8th Nordic Conference on Human-Computer Interaction Fun, Fast, Foundational - NordiCHI '14*.

Louwerse, M. M. (2004) Semantic variation in idiolect and sociolect: Corpus linguistic evidence from literary texts. *Computers and the Humanities*. 38 (2), 207–221.

Lovink, G. et al. (2013) *Unlike us reader: Social media monopolies and their alternatives*. 8. Institute of Network Cultures.

Mair, J. et al. (2006) *Social entrepreneurship*. Springer.

Marazzi, C. (2000) La révolution dérivée. *Multitudes*. 2 (2), 48.

Martin, R. C. (2003) *Agile software development: Principles, patterns, and practices*. Prentice Hall PTR.

McLean, A. (2014) 'Making programming languages to dance to: Live coding with tidal', in Alex McLean et al. (eds.) *Proceedings of the 2nd ACM SIGPLAN international workshop on functional art, music, modeling & design, farm@ICFP 2014, gothenburg, sweden, september 1-3, 2014*. 2014 ACM. pp. 63–70.

Mendieta, E. H. (2013) Arte y maquinas. *Arte y Políticas de Identidad*. 9103–112.

Mollison, B. & Holmgren, D. (1978) *Permaculture*. Lesmurdie Progress Association.

Monico, F. (2014) Premesse per una costituzione ibrida.: La macchina, la bambina automatica e il bosco. *Aut/Aut, La condizione postumana*.

Monteverdi, A. M. & Pino, O. P. di (2011) *Nuovi media, nuovo teatro: Teorie e pratiche tra teatro e digitalità*. F. Angeli.

Murdock, I. (1994) Overview of the debian gnu/linux system. *Linux Journal*. 1994 (6es), 15.

Natoli, C. & Gramoli, V. (2016) The blockchain anomaly. *arXiv preprint*

arXiv:1605.05438.

Negroponte, N. (1996) *Being digital*. Vintage.

Nevejan, C. I. M. & others (2007) *Presence and the design of trust*.

Nikolopoulou, K. et al. (2000) Homo sacer: Sovereign power and bare life. *SubStance*. 29 (3), 124.

Nold, C. et al. (2014) *Autopsy of an island currency*.

Nørmark, K. (2010) Systematic unit testing in a read-eval-print loop. *J. UCS*. 16 (2), 296–314.

Ostrom, E. (2015) *Governing the commons*. Canto classics. Cambridge University Press.

Pallant, C. (2011) *Demystifying disney: A history of disney feature animation*. Bloomsbury.

Pasquale, F. (2015) *The black box society: The secret algorithms that control money and information*. Harvard University Press.

Pelizza, A. & Kuhlmann, S. (2017) Mining governance mechanisms. innovation policy, practice and theory facing algorithmic decision-making. *Handbook of Cyber-Development, Cyber-Democracy, and Cyber-Defense*.

Poe, E. A. (1836) Maelzel2019s chess-player. *Southern Literary Messenger*. 2 (5), 318–326.

Potzmader, K. et al. (2013) 'STUNT: A simple, transparent, user-centered network of trust', in *European public key infrastructure workshop*. 2013 Springer. pp. 65–82.

Qin, Y. et al. (2016) When things matter: A survey on data-centric internet of things. *J. Network and Computer Applications*. 64137–153.

Raley, R. (2014) Outsourced poetics. *American Book Review*. 35 (2), 5–6.

Rancière, J. (2010) *Dissensus. on politics and aesthetics*. Bloomsbury.

Rancière, J. (1995) *On the shores of politics*. Verso, London.

Ridgway, R. (2014) Crowdfunding: Monetizing the crowd? *North East West South*.

Ridgway, R. (2015) Personalisation as currency. *A Peer-Reviewed Journal About (APRJA)*. 4 (1),.

Robles, G. & González-Barahona, J. M. (2012) 'A comprehensive study of software forks: Dates, reasons and outcomes', in Imed Hammouda et al. (eds.) *Open source systems: Long-term sustainability - 8th IFIP WG 2.13 international conference, OSS 2012, hammamet, tunisia, september 10-13, 2012. proceedings*. IFIP advances in information and communication technology. 2012 Springer. pp. 1–14.

Rodrigues, J. J. P. C. (2012) Smart communication protocols & algorithms.

Network Protocols & Algorithms. 4 (2), 1–4.

Roio, D. et al. (2015) *Design of social digital currency*.

Romero, D. M. (2009) Crowdsourcing the complexities of electronic design automation. *IEEE*.

Sacks, D. (2011) The sharing economy. *Fast company*. 15588–93.

Sassen, S. (1996) *Losing control? Sovereignty in an age of globalization*. Columbia University Press.

Scholz, T. (2016) *Uberworked and underpaid: How workers are disrupting the digital economy*. John Wiley & Sons.

Schuh, P. (2004) *Integrating agile development in the real world*. Charles River Media, Inc.

Schumacher, E. F. (2011) *Small is beautiful: A study of economics as if people mattered*. Random House.

Schwartz, T. (1972) Rationality and the myth of the maximum. *Noûs*. 6 (2), 97.

Serino, M. et al. (2016) Pokemon go and augmented virtual reality games: A cautionary commentary for parents and pediatricians. *Current Opinion in Pediatrics*.

Sharma, R. et al. (2010) 'Design of farm waste-driven supply side infrastructure for data centers', in *ASME 2010 4th international conference on energy sustainability*. 2010 American Society of Mechanical Engineers. pp. 523–530.

Shiga, J. (2007) Copy-and-persist: The logic of mash-up culture. *Critical Studies in Media Communication*. 24 (2),.

Shiva, V. (1993) *Monocultures of the mind: Perspectives on biodiversity and biotechnology*. Palgrave Macmillan.

Sober, E. (1978) Computability and cognition. *Synthese*. 39 (3), 383–399.

Stallman, R. (2010) Copyleft: Pragmatic idealism. *Free software, free society : selected essays of Richard Stallman*. 129–131.

Stallman, R. & others (1985) *The gnu manifesto*.

Stallman, R. M. (2004) *GNU and the free software foundation*.

Standing, G. (2011) *The precariat: The new dangerous class*. A&C Black.

Stiegler, B. (2009) Teleologics of the snail: The errant self wired to a wimax network. *Theory, Culture & Society*. 26 (2-3), 33–45.

Surowiecki, J. (2004) The wisdom of crowds: Why the many are smarter than the few and how collective wisdom shapes business. *Economies, Societies and Nations*. 296.

Swamidass, P. (2000) *Encyclopedia of Production and Manufacturing Management*.

812–812.

Sward, A. et al. (2017) *Data insertion in bitcoin's blockchain*.

Terranova, T. (2000) Free labor: Producing culture for the digital economy. *Social Text*. 18.233–58.

Tkacz, N. (2014) *Wikipedia and the politics of openness*. University of Chicago Press.

Toledo, R. R. et al. (2016) Lower-cost private information retrieval. *CoRR*. abs/1604.00223.

Towse, R. (2005) Economics and copyright reform: Aspects of the ec directive. *Telematics and Informatics*. 22 (1), 11–24.

Trell, B. L. (1988) Ancient coins as evidence for the history of art. *Bulletin of the Asia Institute*. 253–65.

Turenhout, R. (2012) Wikileaks & het naïeve idee over wat het toegankelijk maken van informatie teweeg zal brengen. Master's thesis thesis. Utrecht University, MA nieuwe media & digitale cultuur.

Valeyre, A. (2004) Les nouvelles formes d'intensification du travail industriel : logiques technologiques, organisationnelles et économiques. *Economies et sociétés, série Socio-économie du travail*. (n24), p.1993–2027.

Van Wyk, C. J. (1987) Literate programming. *Commun. ACM*. 30 (12), 1000–1010.

Vercellone, C. et al. (2015) *Managing the commons in the knowledge economy*. D-CENT (FP7/CAPS 610349). [online]. Available from: http://dcentproject.eu.

Vercellone, C. et al. (2014) *Theoretical framework on future knowledge-based economy*. D-CENT (FP7/CAPS 610349). [online]. Available from: http://dcentproject.eu.

Vernotte, A. et al. (2017) 'In-depth modeling of the unix operating system for architectural cyber security analysis', in *Enterprise distributed object computing workshop (edocw), 2017 ieee 21st international*. 2017 IEEE. pp. 127–136.

Viljoen, A. & Bohn, K. (2014) *Second nature urban agriculture: Designing productive cities*. Taylor & Francis.

Viljoen, A. & Howe, J. (2012) *Continuous productive urban landscapes*. Routledge.

Weber, B. (2014) Bitcoin and the legitimacy crisis of money. *CAMECO*. 40 (1), 17–41.

Weber, M. (1904) *Die protestantische ethik und der geist des kapitalismus*. JCB Mohr (Paul Siebeck).

Wikileaks (2011) *Banking blockade*.