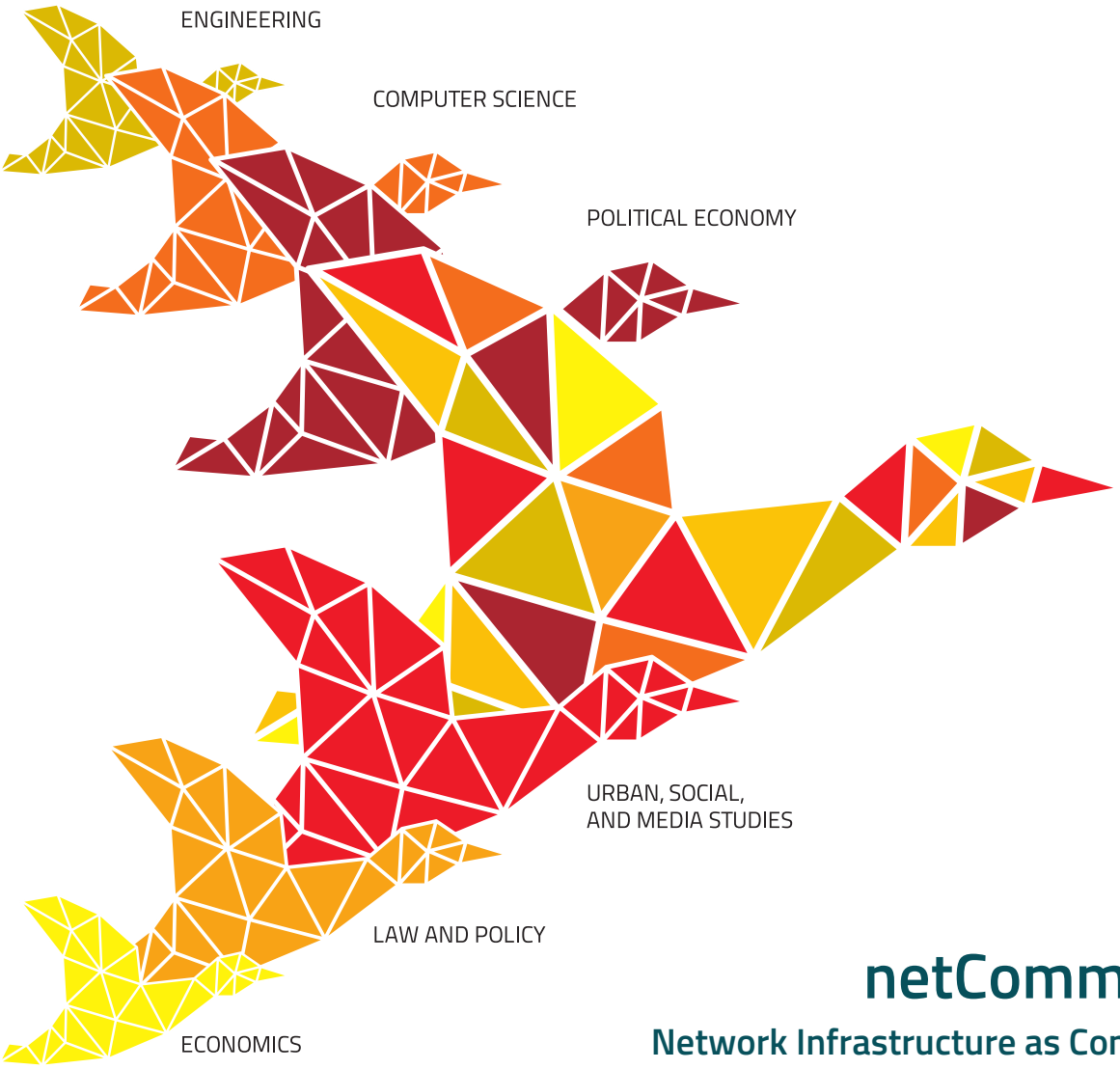


netCommons

Network Infrastructure as Commons

Deployment experiences with a Multi-Disciplinary approach

Deliverable Number D3.6
Version 1.0
October 8, 2018

Co-Funded by the Horizon 2020 programme of the European Union
Grant Number 688768

netCommons.eu

Project Acronym: netCommons
Project Full Title: Network Infrastructure as Commons.
Call: H2020-ICT-2015
Topic: ICT-10-2015
Type of Action: RIA
Grant Number: 688768
Project URL: http://netcommons.eu

Editors:	Alexandros Papageorgiou, NetHood Renato Lo Cigno, UniTN
Deliverable nature:	Report (R)
Dissemination level:	Public (PU)
Contractual Delivery Date:	July 31, 2018
Actual Delivery Date	October 8, 2018
Number of pages:	47
Keywords:	community networks, participatory design
Authors:	Alexandros Papageorgiou, NetHood Panayotis Antoniadis, NetHood Merkouris Karaliopoulos, AUEB George Klissiaris, AUEB Vassilis Chryssos, AUEB Leonardo Maccari, UniTN Renato Lo Cigno, UniTN Felix Freitag, UPC Leandro Navarro, UPC
Peer review:	Luca Baldesi, UniTN

History of Revisions

Rev.	Date	Author	Description
v0.1	10/07/2018	Alexandros Papageorgiou	First interviews summaries
v0.2	30/07/2018	Renato Lo Cigno	Introduction and structure of the document
v0.3	05/08/2018	Leonardo Maccari	ninux.org methodology application
v0.5	10/08/2018	Alexandros Papageorgiou	Sarantaporo.gr methodology application
v0.6	18/08/2018	Alexandros Papageorgiou	Co-creation chapter
v0.7	20/08/2018	Panayotis Antoniadis	Various edits and additions
v0.8	20/08/2018	Leandro Navarro	UPC methodology application with Cloudy
v0.9	22/08/2018	Panayotis Antoniadis	Final draft and executive summary
v1.0	1/10/2018	Renato Lo Cigno	Final editing and inclusion of feedback from authors and peer-review

Executive summary

The participatory design (PD) methodology presented in Deliverable 3.3 ”Multi-Disciplinary Methodology for Applications Design for CNs, including Design Guidelines and Adoption Facilitation” consists of two main parts:

1. A list of recommended actions that might need to be taken by the project team, organized in different Processes, and Threads, together with evaluation metrics. This list has been compiled based on our own experience in the context of the Sarantaporo.gr case study, and more specifically the design of a smart farming application called AppLea, but also other experiences around the world, as described in Part I of D3.3.
2. A recommended process for team work inspired partly by popular agile methodologies like SCRUM, but also by less structured activities like Jazz improvisation. The process, which draws an explicit analogy with music, has been experimented by netCommons researchers in on-line activities with Community Networks (CNs).

This deliverable reports the results of the experiments and the feedback obtained from the three software teams leading the three corresponding software development tasks of the project. Furthermore, it describes the steps undertaken to improve both building blocks of the methodology as a result of the the feedback and experiments as well as the activities planned to increase the impact of this work beyond the end of the project.

This feedback has been collected in two different ways: 1) through personal interviews by Alexandros Papageorgiou, the mediator of the PD process that was applied in the case of Sarantaporo.gr, and 2) through detailed reviews received by the mediators of the three PD processes corresponding to the three different software development tasks.

The outcome has been very productive leading to various suggestions for the improvement of the list of recommended actions, but also toward the simplification of the overall process, or at least the offering of different alternatives with different levels of complexity to match the needs and available resources of different teams. Nevertheless, the methodology itself is conceived as a product of co-creation and this evaluation and refinement process will continue until the end of the project and beyond it, the methodology itself a participatory design effort we hope many communities will undertake. By the end of the project, we will release and print the final version of an easier to read and use booklet, and an on-line platform to support the adoption in scenarios where frequent meetings in person are difficult.

Contents

List of Figures

List of Acronyms

CC	Community Cloud
CN	Community Network
DIY	Do It Yourself
FLOSS	Free/Libre Open Source Software
ICT	Information and Communication Technology
ISOC	Internet Society
PD	Participatory Design
PS-ng	PeerStreamer-ng
RIPE-NCC	Réseaux IP Européens – Network Coordination Center

1. Introduction

This deliverable documents the review and co-creation process that has been carried out between the authors of the netCommons Participatory Design (PD) methodology, described in detail in D3.3 [1], and the software development groups of WP3 based on their experience in working together with CNs developers. This process has evolved through interviews and detailed documentation of the steps taken in applying the methodology, in collaboration with the local communities and software development groups.

After the first review meeting, netCommons was explicitly asked to invest additional effort to expand as much as possible the experimentation with the participatory design methodologies developed in D3.1 [2] and D3.3 [1], and already applied, during the methodology development phase, to the consolidation of the Sarantaporo.gr network itself and to the initial design and implementation of the AppLea farming application (see D3.2 [3], and 3.4 [4], for its complete description).

Part 2 of D3.3 formalized a broad, omni-comprehensive methodology for the continuous involvement of a local community in the design and, possibly, implementation of innovative projects. These projects do not necessarily focus exclusively on Information and Communication Technology (ICT), but bear as common feature that ICT normally plays an important role, as information and the right to communications and free access to global resources are main motivators of the methodology.

The application of such a broad methodology, however, needs a customization to the local environment and the project/application itself. It is thus of the utmost importance for its final success and applicability to understand how this process of customization and simplification can be implemented on field. Especially for PeerStreamer and Cloudy[1], which are complex software projects that have been under development for years. Thus their further development and deployment can only be marginally framed in the holistic methodology described in Part II of D3.3, and the steps taken by netCommons developers to involve the communities in the design and deployment has been necessarily biased by the existing applications as well as by technical and other constraints. For example, in the case of Cloudy the methodology needs to adapt to a local, already established, more informal, PD process, and even after this adaptation it is possible that the local community will prefer to remain with the old one.

Different interventions and different local projects ended up in different levels and types of engagement of the different software development teams with local communities, subject to available resources on both sides. Nevertheless, the individual feedback from all the different PD process mediators seems to converge towards two important points, expressing the natural trade-off that every methodology has to face:

1. The presented Processes, Threads, Actions and Guidelines, provide useful insights on often neglected aspects of a software development process aiming to serve the needs of local communities;
2. The full-fledged methodology is rather complex to grasp and rather demanding in terms of human resources to implement, increasing significantly the entry barrier.

During the last months of the project we will integrate this feedback into the production of a user-friendly booklet with both a printed and online version (to be continuously improved also after the end of the project).

[1] PeerStreamer and Cloudy are the other two software projects, besides AppLea, that netCommons is customizing and optimizing for CNs deployment, exploiting P2P technologies and relaying as little as possible on the presence of a global Internet connection. They are thoroughly described in D3.2 and D3.4, while results on their use in CNs will be reported in the final deliverable D3.5 at the end of the project.

1.1. Organization of the Deliverable

The deliverable is organized in three main chapters. The first two chapters present feedback from the software development teams, as obtained through personal interviews with the process mediators of each team (Chapter 2) and through the self-documentation of the effort to apply the methodology to the respective case study (Chapter 3). Next, Chapter 4 summarizes several directions toward the continuous co-creation of the methodology, indeed in a participatory way.

Chapter 2 and Chapter 4 are coordinated by NetHood, the methodology's author, while Chapter 3 is coordinated by UniTN that has involved the participatory design mediators of each initiative as appropriate.

More specifically, each of Chapter 3 sections is devoted to one of the three software development processes of netCommons that include "local" considerations on the additional development effort. Of course, as stated above, the customization of the software and its initial deployment were done before the methodology was completed, thus we have to map the methods and terminology used there on the specific instances and phases of D3.4 methodology, rather than applying this methodology top-down. Also the timing and phases used may differ for this reason. Sec. 3.1 is devoted to the further participatory design events and the corresponding application refinements of AppLea done after the delivery of D3.3 [1] Sec. 3.2 reports the actions taken by the developers and deployers to adapt the P2P streaming application to CN environments, and specifically to the ninux and guifi.net architectures (both in term of networking and in term of addresses communities of people). Sec. 3.3 reports on the methodology followed in the collaborative evolutionary design of a Community Cloud (CC) commons for community networks along several years in guifi.net and its Cloudy software implementation, with more details about community clouds reported in [5]. The netCommons methodology has been introduced in the last months of activity and we report on the experience and lessons learned.

Finally, Chapter 5 concludes the deliverable summarizing the experiences, and providing useful feedback to be integrated with D3.4 for anyone wishing to adapt applications to specific community environments.

2. Feedback through interviews

In January 2017, Panayotis Antoniadis presented to the other members of the netCommons consortium the participatory design methodology developed by NetHood, regarding the design and development of local applications in community networks. In the context of the evaluation and improvement of the methodology and with the explicit goal that its content and form become a product of continuous co-creation, it was deemed necessary to get systematic feedback regarding its utility from other members of the consortium who are involved in designing and implementing local apps for community networks, namely guifi.net, ninux.org and Sarantaporo.gr.

Therefore, Alexandros Papageorgiou (NetHood) conducted interviews to discuss the methodology with Felix Freitag and Leandro Navarro of UPC (responsible for the design and development of the Cloudy software and members of Guifi), Leonardo Maccari of UniTn (responsible for PeerStreamer software, and member of ninux.org), Merkouris Karaliopoulos and Aris Pilichos of AUEB (responsible for the development of the AppLea farming app) and George Klisiaris of Sarantaporo.gr. A first round of interviews took place in the end of January and beginning of February 2018 (with L. Navarro in March), in the course of which interviewees shared their initial impressions. In July 2018, Leonardo, Merkouris and Aris gave additional feedback on the methodology, based on further elaboration that they had made. Leonardo also provided feedback that he got from other members of the ninux.org team, as described in detail in Chapter 3. George was interviewed for the first time in July 2018. Merkouris, Aris and Alexandros also worked collaboratively on the methodology to create the 'Project Score' for Sarantaporo.gr's AppLea.

In the following, the most important conclusions from the interviews are summarized in the following categories: Overall evaluation, Critique-Comments, Suggestions, and Open questions.

2.1. Overall evaluation

Leonardo pinpointed the specificity of the methodology as being a toolbox that guides you to focus on certain things rather than do things in a certain way. In this sense he compared it to the Method Kit.

Merkouris summed up the contribution and value of the methodology as a set of tools that:

- Reminds you everything you should take into account;
- Enables monitoring through visualization, self-control and communication;
- Helps you to proceed to evaluation and decision-making based on contextualized measurements.

For Merkouris, monitoring is a minimum. In addition, the establishment of valid metrics can allow the enhancement of the methodology's utility.

As Leonardo argued, when executing a project, you need to evaluate based on the goals you had set. Otherwise, it is very hard to measure and evaluate the impact of Actions. Nonetheless, setting specific goals can be difficult for an inexperienced team walking into an uncharted field.

All interviewees view most Threads-of-Actions and Actions as pertinent and consider it positive that a new user has access to suggested Actions. However, these should be easy to see, understand and select, both visually and conceptually.

For Leandro, the methodology should help one to 'see' the steps to follow in order to solve one's problems. It should help one to find and extract the tool one needs (and fits one's project) directly, without having to go through and understand the whole toolbox. One should not have to absorb, adopt and embrace the methodology

in its entirety before starting to use it, as an imposed set of tools that could even be irrelevant. While, if one can extract the elements one needs quickly and easily, the methodology would be successful.

As Felix argued, in theory it's nice and useful, but what happens in practice is that things are done in a non-organized, haphazard way: "*Practically, we cannot have regular face-to-face meetings and we don't do them, neither can everyone be informed about everything. Usually what happens is that those who are hierarchically superior say how things should be done. From time to time there is some feedback, but generally it's not organized. Some members of the team might have an idea and they decide to do it, but processes are not participatory.*" When asked what he would expect from such a methodology, Felix answered: "*a recipe*".

Leonardo affirmed that, in the participatory design process for the PeerStreamer-ng, the methodology actually helped him having some concepts in mind: "*I had in mind that we had to take care of some issues, like physical space, or the long-term sustainability, or to not focus only on the technology but to also have informal meetings and discussions to reach points of agreement. [...] to be more inclusive, diverse, to listen more. [...], I did it also because I had this frame in mind.*" However, he didn't want people to think he was pushing his agenda, using them as an experiment: "*I didn't make it explicit for them, but I used it as my agenda, as a checklist that I have to take care of, because I think it's beneficial. Not just because I had to. The end result was good. I would do it again.*"

2.2. Weaknesses

All interviewees acknowledged that the methodology has many positive elements, as it includes everything that one should have in mind when doing participatory design, and could thus be a very useful toolbox to accompany the work of a design and development team. However, there were several points raised with regard to weaknesses and shortcomings that could be improved. Focusing this deliverable on the latter is not meant to overshadow the virtues of the methodology. Rather, we believe that emphasizing the points to amend is highlighting our conviction that this methodology has the potential to become a helpful toolbox for the purpose it has been designed for.

Another point to be taken into consideration is that, while getting familiarized with the methodology and assessing it, all interviewees drew on their prior experience with similar sets of tools. In the following we summarize the main current weaknesses that should be considered carefully during the further development of the methodology.

2.2.1. Complexity

The methodology can be intimidating because it looks complex. There are too many layers of conceptualization and it asks from users to *understand it adequately before they use it*. Also, in thinking about each Action one needs to see it through the prism of the Thread it belongs to and in thinking about each Thread one takes the Process into consideration.

The effect of this on users is a feeling of 'heaviness', indeed, complexity may impede the passage to action. As Leonardo stated in January 2018:

> "*I think that's too complex. Knowing myself, I would spend much more time to analyze how I can optimize everything —what I need what I don't— rather than start doing things. So, I think either you drastically reduce the number of roles, processes, etc., or, you characterize specific scenarios and you say "you are here, you don't need this" and finally you keep in your mind ten things and you proceed from there. [...] The thing that scares me more is the complexity of the thing.*
> *[...] I'll simplify it a lot and people might use it in the end. But if I weren't in the project and in the ninux community I would consider it really hard to convince someone to use it, given the complexity of starting. This is the main difficulty in real life. So, one feedback you are going to have from us for sure is to simplify things down, so that people can start with a very easy and basic thing.* "

And in July, after getting some direct, non-mediated feedback from other ninux.org members (see Sec. 3.2 for more details), Leonardo reported that

> "*The general feeling is that it's over-complicated. One person said that it would be nice to use it in corporate or work environment, where people would be obliged to do this, with somebody to guide people. If not, the methodology asks you to keep in mind way too many things. [...] The other feedbacks I received were all more all less along the same line. The document, at this stage, was not clear, so they could not understand what was the problem to solve, they did not understand who should use it and in which context, and generally, they thought it was an over-complicating thing. Instead of helping you to achieve a goal, it has a learning curve that is really high, really steep. So, before you can understand how it works, you have to read it all and at the end you are not motivated anymore. You need to read 15 pages just to bootstrap the thing. This is the initial feedback. Maybe next versions will be more digestible, friendlier.*"

After listening to Leonardo's and other interviewees' comments on complexity (explained in more detail by Leonardo in Sec. 3.2), the question arises: Is there a solution that would facilitate action without sacrificing essential layers of conceptualization? Sec. 2.5 on Suggestions provides some guidelines that we will take into account in the last phase of the project to transfer the methodology into a usable booklet published on-line hoping to leave a meaningful impact of this work beyond the end of the project.

2.2.2. Concept overlapping

Different concepts introduced in the methodology seem to overlap across Processes and sometimes across Threads of Actions. Additionally, the hierarchical relationship between categories is often unclear.

Quoting again Leonardo Maccari from the January interview:

> "*There were too many concepts and many of them were overlapping. Too many layers of conceptualization: Threads, Processes, Metrics, going up and down, and in some cases I couldn't realize what was the hierarchical relationship between things. I was asking myself, e.g.: 'Should this, instead of being a sub-category of something else, be another category?'* "

Also, interviewees had difficulty at trying to register some Actions, for instance Merkouris and Aris in the Functionality Thread in Software Development, in relation to the right term to use, the duration of the action and its possible repetition.

A characteristic part from the July interview is reported as Interview Extract 2.1 to exemplify the difficulty to map high level participatory design principles on real software and system development actions. Another example of concepts overlapping is that it could be unclear if Actions contributing to Community Participation are about the development of a local app or the development of the CN. Of course, this is caused also by the fact that they are interdependent: what helps the CN can help the app indirectly and vice-versa.

Interview Extract 2.1: Extract from the interview of Merkouris Karaliopoulos by Alexandros Papageorgiou in Sarantaporo in July 2018.

...
Merkouris: *Around this period the Functionality of the app started to take form. How do we illustrate this? There was an iteration in Software Design about the content of the app, not Functionality.*
Alexandros: *Perhaps we can add Design Actions under Functionality.*
Me: *But Design is a Process. Isn't it overlapping? Maybe we could rename the Process to Software Design and Development.*
Al: *Could it help if we merged Threads? We're still experimenting.*
Aris: *Design and Development alternate, with intermediate iterations.*
Me: *I think that in Functionality there is an Action in the course of the Development.*
Al: *In terms of Design, could we find an Action that lasted? Instead of three different Actions? In the context of the initial design?*
Me: *We were already developing by this point.*
Ar: *The code was already on GitHub, since December 2016.*
Me: *So, when the workshop happened, there was uploaded code already. This means that in Functionality we have something with duration when developing code and another process here where we develop code.*
Al: *Ok, how could we illustrate this?*
Me: *Let's illustrate these as Implement Functionality Actions, regardless whether we include Design as part of the Process. And these are phases of the development.*
Al: *Let's put brackets to illustrate duration.*
[...]
Me: *How can we capture two Actions that have duration and are progressing in parallel?*
Al: *Perhaps with different colors* [he does it, adding the Beta testing Action under Phasing, as a continuation of Alpha testing]
[...]
Me: *We should clarify which Functionality we are talking about each time.*
Al: *This is also a decision regarding what you want to be recording in the Project Document that you have on the side. If you have, let's say, four times Implement Functionality, you can put a, b, c, d and note on the side what these Actions were, specifically.*
Ar: *They're different Functionalities each time. You'll note down the main ones.*
Al: *Are these registered somewhere anyway?*
Me: *In commercial software you do it, but there, there are many programmers working, so you document everything. And GitHub has helped a lot.*

2.3. Hybrid space design: a non-clear concept

For all interviewees the Hybrid Space Design process was not clear from a conceptual point of view: "*The Hybrid Space Design is the less clear Process*" (Merkouris and Felix in January-February) and "*Hybrid Space is still an unclear concept to me. [...] What goes into Hybrid Space? If I upload a video on YouTube, is that Hybrid Space, is that feedback, is it promotion?*" (Merkouris in July). As a result, they experienced difficulty in reflecting on how to implement relevant Actions.

2.4. Notation overload

The Notation can capture dependencies that do exist, but over-using it could become unmanageable, both conceptually and visually.

Quoting Leonardo in January:

> "*If you want to make all the relationships explicit you're building a waterfall project. Everything influences or is connected to many other things. So, after a while it might not be working for a reason. So I go back and think that 'this didn't work, maybe I put too much effort there, I should have put effort somewhere else'. But then I have a whole range of dependencies, how do I attach them to these new dots? All this Notation creates dependencies between Actions, which is reasonable. You can do this, but it can get unmanageable. [...] Still I don't see the use of the Notation. Sometimes I see an arrow, sometimes the dots, for instance in Slide 1, there is this arrow pointing downwards, there is a connection. There are two or three connections. One has an arrow and the other one doesn't. Is it because there is a semantic difference or is it because there is an arrow missing? [...] The Relationship should have another symbol.*"

And in July:

> "*The Notation is pretty confusing. [...] Too many things that are not very different one from another semantically. It's too time-consuming to place the Notation, compared to the result that you'll have once you've done. I would remove some stuff. Maybe in a short two-page version you could limit it to basic things, to hide the rest and introduce it afterwards. [...] if you look at project management books, they tell you that every time that you introduce a dependency between actions in a project, you are making it more complex. So, if something fails and all the tasks are more or less independent you can recover quickly. But if you have a waterfall of dependencies and something fails, you have to re-project everything. So, it's not clear to me what happens if one of these tasks fails or gets delayed or whatever. Do the tasks have to be moved, are the others somehow impacted, is it just for showing that there is this dependency? I think that something like this is needed, but it should be limited to the minimum. And the other thing I think I told you, I don't see much use in them. The fact that things diverge or converge, they are in the fact itself. I don't have to annotate the project to show this. I think that they add to the level of complexity.*"

2.5. Suggestions

These interviews and discussions lead to a list of possible suggestions to improve the methodology that are condensed in the following subsections, each one addressing one possible topic.

2.5.1. Templates

Templates would help users understand quickly what the methodology is about and which elements and which path is more suitable for their case, through the typification of cases (typifying different motivations and levels of experience).

2.5.2. Customization

Merging or striking out categories (Threads of Actions) should be possible, based on the statutory customizability of the methodology, with the help of templates. Also ranking elements could help users understand better and organize their own case.

2.5.3. Hierarchy

The methodology could improve in presentation and clarity, namely in visualization, and concision, providing examples and splitting up in two or three levels of detail. It could include a quick start guide and, of course,

a more detailed 'manual', also in order to attend to the different levels of experience. A user could start using the methodology from the Process of their choice, possibly the one they feel more comfortable with, while applying basic Notation. Leonardo commented on this matter with a 'hide the complexity' suggestion. In the interview with Alexandros he agreed that this could be done either by means of a kind of quick start guide, as described above, or with the mediation of a trainer who would hide the complexity from and for new users. Another possibility would be for a user to remain faithful to the methodology's suggestions in the beginning and, once they are experienced users, they could create their own categories and Actions, merge Actions, etc. But this way one would need to go through the difficult phase of studying and understanding everything.

2.5.4. Careful use of notation

With regard to Notation, perhaps only the most important dependencies should be illustrated, to avoid overburdening the Score. A guideline could be included that Notation should be used retrospectively, at the evaluation phase, when users can be sure about the effect of Actions.

2.5.5. Online tool

All interviewees commented that an online tool could prove very helpful. They all know, however, that the production of such a tool exceeds the time and resources of the project. Besides, it is a principle of the methodology to insist on the physical dimension of teamwork since it values face-to-face interaction. This is not to say that the development of an online version would not undoubtedly be helpful as a complementary tool to meetings in person. Felix Freitag mentioned that the development of a video tutorial would also be useful.

George Klissiaris mentioned the idea of a common pool, the methodology as a platform where you can pick out elements from and drop new elements in. This requires the existence of an online tool.

2.5.6. Project Score editing

It is important that the Actions can be moved in and out of the Score. This would either require a digital version of the methodology and stickers and/or cards and/or pieces for a physical version. Users should be able to move 'pieces' (elements) around, to change their minds without having to re-print the score and rewrite everything. So, an idea would be for stickers to be provided, both with specific Actions on them and blank. They could come in (fixed or customizable) colors. Also, blank stickers would be needed to cover the position from which tasks should be erased.

A question about history arises. Could changes be traceable? In a digital form this would be easy, but what about the physical? Perhaps it could be done with small sticker-flags which can be attached to the Action sticker. Notation would also need to be easy to change.

Perhaps different colors can symbolize a combination of temporality and degree of completion of Actions, that is: planned, aborted/transferred, completed, not completed/delayed, so that, with one look at the Score, one can know at which point in time we currently are.

2.6. Open questions

After several iterations with between developers and the team that designed the methodology, there are still many open questions that remain without a definite answer and that we will not be able to answer completely before the end of the project nor probably in the following months as they relate to fairly complex interactions between technical development and societal involvement. The following list summarizes some of them highlighting the reasons why they remain open and unanswered.

- As mentioned earlier, the use of templates was suggested during the interviews. **What is the right balance between the simplification of concepts and empowering users to adjust to very different conditions?**. This issues rises the clear conflict between the risk of doing wrong (because of oversimplification) and not doing (because the complexity of the framework block development.
- **Should everyone in a team know or at least understand everything or should tasks and responsibilities be distributed?** If each individual or team monitors one specific part/Process, they probably need to entrust someone who has the overview of all Processes, like the Scrum master[1]. But if so, can the collective use of the methodology and, consequently, the collective creation of the Score, still be ensured, meaning collectively establishing correlations, assessing actions, planning the strategy, etc.? This is an important question, since the methodology is aimed to encourage participation in the relationship with the community, but also internally inside the team of designers-experts.
- **Is the goal to create guidelines for a fully Do It Yourself (DIY) implementation, or should the role of experts be maintained and valued?** Perhaps we should not dispose of human intervention. Perhaps a guide is needed that helps the trainers, the people who have experience or who want to obtain experience, to pass on their expertise and facilitate others to adapt the methodology to their needs. Indeed, here we also rise the question of how much DIY approaches can finally get rid of experts, both at the participatory design level and at the technological level, remaining aware that technology experts often need a more agile and fast process than the one imposed by participatory design to successfully complete a project.
- **How should one deal with different levels of experience of users?** The main hypothesis is that users, the candidate implementors of the methodology, might be knowledgeable in one process (e.g., the software development), but might not have experience in participatory design with communities, to take an example relevant for the netCommons project. However, the Processes are not ordered hierarchically in order to not prioritize one Process over another, but, this way, less guidance is provided to the users. Once more, we highlight a sort of a conflict we have perceived between technology experts and developers and the experts and theoreticians of participatory design. We just "register" this perception and leave its solution for future research projects, as it clearly goes beyond the scope and resources of netCommons.
- **On which level of detail should Actions be recorded?** During the interview, Felix Freitag started enumerating specific types of Actions, whereas the methodology proposes to record Actions under more generic categories, and places the stress on the 'action' part, not the content. Among the proposed Actions, the methodology lists verbs, not nouns. One could conclude that the labeling of Actions should be neither too detailed nor too vague. When reviewing the Score, the user should immediately remember and understand what the Action that they did was, but loading the Score with much text should also be avoided. Perhaps a limit in characters can be proposed (especially in an online version). The level of information included on the Project Score is also related to the use of the Project Document.
- **Is time relevant?** Especially in George's opinion, capturing the dimension of time is not that important for the project's planning. It's the status of the Actions that a designer team should care about.[2] "*The time aspect is confusing, both visually and conceptually. The spans are not measured somewhere in a way that would offer value to you.*" What would be important, regarding the relevancy of time, is the level of completion of different Actions, combined with the ways they are interrelated with each other.
- A critical question remains and affects the answers to all the questions above: **To what degree should this methodology compromise completeness in order to become 'light', flexible and accessible?** Should it be able to adapt and be tailored 'unconditionally' to any level of experience and skills, any context and different needs, or should it draw some lines and be more rigid, as if to say that when designing and developing a local app together with a community there are things one should not escape from and work that one should not avoid. The methodology does not argue why one should involve the

[1] As members of guifi.net and ninux.org commented, in their groups one individual would probably have to assume the implementation and monitoring of the methodology.

[2] Enumerated in the previous paragraph: planned, aborted/transported, completed, not completed/delayed

community, it supposes that the user already knows why and is looking for a way to do it. Of course, the counter-argument could be that a potential user could see value in the methodology and would want to use it, but would also ask for help in order to understand and implement it. Furthermore we do not have a definite answer to the additional effort that the methodology imposes on developers. Developers are very often short in time, and there is a widespread literature in engineering and software engineering in particular that shows that projects that do not complete in reasonably short time-frames normally never complete, draining resources without generating appreciable results. This seems to be a profound conflict between the fast pace of ICT evolution and the slow pace of societal adaptation, a conflict that deserves attention and future studies, but cannot be disregarded if participatory design principles are to be integrated into ICT projects and development.

Concluding, at this point it seems there is a need to find ways to make the methodology more user-friendly, to work on the guidelines, with more visualization, with examples of concrete cases, perhaps creating some sort of templates, but without sacrificing or jeopardizing the layers of complexity that are necessarily entailed in projects where trust-building and human relationships are central.

3. Feedback from the application of the methodology

In D3.3 ([1] Sects. 6 and 7) we describe a simplification of the methodology and three different examples of implementing the proposed simplified methodology. These corresponded to the CNs strongly connected to the project (Sarantaporo.gr/AppLea, ninux.org/PeerStreamer, and guifi.net/Cloudy) in order to help the software developing teams of the project to understand better the rationale behind the specific methodology.

In this chapter, the different developing teams summarize their experience in using the methodology during their interactions with the target communities, which adds to the explicit feedback provided through the interviews summarized in Chapter 2.

Notice that the suggested actions in D3.3 were heavily adapted to the actual conditions and opportunities that appeared in the field, and also bended and twisted to already existing local practices; one more evidence that successful methodologies for inclusion and participation should be open-ended and flexible.

3.1. Sarantaporo.gr CN – AppLea

In the case of the Sarantaporo.gr CN, the methodology was put in place even before its final version was documented since the methodology development team (NetHood) worked very closely with the software development team (AUEB) from the beginning of the process. Many of the proposed actions in the methodology have been devised or tried out during this process, documented in detail in Deliverables 3.1 & 3.3.

In this section we continue documenting how the methodology was applied in the design of the AppLea application developed by AUEB, during its most important phase: the contact with the real community of users, i.e., farmers in the Sarantaporo area who were more or less familiar with the Sarantaporo.gr CN.

So, after the initial feedback that the farming app designer team got through the participatory process that was initiated with farmers in the West Olympus region in December 2017, Aris Pilichos, the team's programmer, started to work on the incorporation of farmers' comments in the app, in view of its first presentation to the interviewees who would become the beta testers for the app's further development. The beginning of this process coincided with a period that was marked by the CN's substantial upgrade and growth, which provided a positive conjuncture of increased participation and diffused optimism.

In the following, we present the most important meetings and corresponding turning points in the process.

3.1.1. Presentation of the app and initiation of the beta testing phase

In March 2018 the first version of the revised CommonTasker app was launched (CommonTasker was the previous name of AppLea, the change of its name is described later on in this section) . The designer team traveled to Flambouro village to present it to the farmers that had shown interest in this project and had given feedback a few months before (see D3.6, Sect. 2.5). On the same day, a training seminar on the CN took place in Flambouro, facilitated by Sarantaporo.gr Coreteam members and NetHood's Panayotis Antoniadis.

Following the seminar, the Coreteam, consisting of George, Vasilis and Achilleas of Sarantaporo.gr, Aris and Merkouris of AUEB, and Panayotis and Alexandros of NetHood, presented the context and aspiration around the app, as well as its first version, to Dimitris Ntallas, Thedoros Minas and Sakis Katis. Sakis hadn't given feedback last December, but agreed to participate this time.

The design team had agreed to be transparent toward the community of future users with regard to their goals and the context of the project. First, Alexandros presented the team, "the Greek branch of netCommons ", as

Figure 3.1: The training seminar for Sarantaporo.gr CN node owners in Flambouro village

well as the netCommons project in general, its goals and limitations. He assured participants that the design team is interested in building something that could prove useful for the community in the long run, however, he also mentioned that we cannot have too high expectations due to the project's limited resources and approaching conclusion.

George intervened to present what open source software is and to explain that there is the possibility for anyone (a commercial entity, the farmers themselves or us again, if we find additional funding) to pick up from the point we will have reached by the end of the year and to continue developing the app. Alexandros paralleled the app to the Sarantaporo.gr CN, how it started 'small' and one thing led to another, and how we hope for a similar evolution for the app. Next, the idea of participatory design was presented to participants. Panayotis said that we need to create a common language in order to 'translate' the needs of farmers to a digital tool.

Then Merkouris started to present the idea behind the app and its current status, using slides. He made reference to the gamification component, although he admitted that it might not be relevant to local needs. A short discussion followed on how the gamification aspect could be helpful (e.g. using subgroups and templates of behavior), but it was soon interrupted because participants acknowledged that all this cannot be implemented any time soon and that there are many intermediate steps to take.

The presentation continued and after a while we had a distinctive moment of participatory design. Panayotis proposed that, as the CN is named after the village of Sarantaporo, each village could 'own something'. In this sense, the app could be named Milea app (two out of three locals come from Milea and it's been an active village of the CN since the beginning of the CN's operation). Sakis suggested 'Applee' or 'Mileapp'. In a climate of euphoria and enthusiasm we settled on AppLea. The app had just been baptized in a participatory way.

Several important issues were discussed afterwards:

- Merkouris asked if all local farmers are familiar with the process of creating an account in an app. Participants commented that many locals don't know their emails, so we concluded that it would be

Figure 3.2: Introductory presentation of the AppLea application

better to have the double option to create an account either with one's email or phone number.

- With regard to sharing, Sakis asked if there's the possibility to choose with whom to share data, to create groups. The response was affirmative.
- Participants asked whether depending on the profession and the crops that each user registers, the corresponding options for actions should automatically appear.
- The question was raised whether place names should be default or if they should be customizable by each farmer. When discussing the possibility of voice recognition for selecting places, Dimitris joked that given the local accent "the machine will go crazy".
- Contrary to what Theodoros and Dimitris had stated during their interviews last December, Sakis proposed that the quantity of water used for irrigation should be registered and not the duration. This way, data would be more 'scientific'. Regarding the more or less precise registering of frequency of tractor use, in order for farmers to plan its lubrication, Theodoros commented: "*To know roughly is good enough. The tractor forgives.*"
- There was a long discussion around the creation of a history of actions and of the weather, possibly in combination, and also around the option for users to select the history for a period of their choice.
- Concerning weather stations, it had to be clarified by Vasilis that they do not forecast the weather, instead they give a report of its current status. Participants said that they would ask for the data collected by the weather stations of GAIA Epicheirein. "Do you have access?", Vasilis asked. "We should, it's our data anyway", replied Dimitris and Theodoros. Theodoros stressed the importance of creating personalized statistics about the use of resources and applied practices by farmers.

At some point the conversation diverged to more fundamental issues such as data privacy and sovereignty. This was sparked by the observation that what GAIA Epicheirein is lacking is precisely the information that this application tries to gather, and so it could become very useful in order to 'fit' with GAIA's services, i.e., it can complement standard GAIA services. Sakis reacted, saying that GAIA has a different agenda, they are

a company who wants the data for free to then sell them back to the farmers. Merkouris pointed out that it all depends on the service delivered back to farmers. GAIA, but indeed any company that implements such a service, normally has the goal to produce some valuable knowledge out of the data. If this knowledge is returned to farmers who can benefit from it and it helps them improve their practices and output, then this becomes a win-win case. Panayotis suggested that the community of farmers could negotiate with GAIA the use and return of their data. "*Like a cooperative of data?*", Sakis asked, now more interested on the subject. The discussion was no more about the technical-functional aspects of the app. Apparently participants were drawn to reflect on the prospect of this project, on its meaning and possible trajectories. Panayotis and Vasilis pointed out that it's important that the app becomes useful. Then we'll see what could happen. This was a good moment to reveal the long-term vision of the team: to have the data stored locally. But we didn't insist on this point, and left it for later when the data collected will be made more concrete and tangible through the AppLea app.

In any case, the fact that since the very first meeting the political dimension of data ownership was brought in the table and the complexity and inherent trade-offs were quickly revealed was very encouraging for us, a truly positive surprise. From a methodological point of view, this discussion represents a good example on why participatory design methodologies shouldn't be "over designed" and allow room for open discussions.

At the end of the discussion the newly 'appointed' beta testers installed the app in their phones (see Fig. 3.3) and a Telegram group was created. The event was considered very successful by everyone who was present.

Figure 3.3: Installing and testing the application

The team had not used the methodology explicitly before this presentation, by filling out a Score. Nevertheless, in the process of planning it, and especially since the interviews with local farmers in December 2017, the team has consciously tried to apply ideas and concepts fostered by the methodology, mainly regarding Community Participation, toward establishing the foundation of a solid relationship with the community.

Following this meeting, several Actions were registered later in the Methodology Project Score.

- In the Community Participation Process: Reinforcement of Collaboration in the Trust Thread, Minutes in the Document Thread, Participatory Workshop in the Engagement Thread and Comments in the Listening Thread.
- In the Hybrid Space Design Process: Installation and Presentation of App in the Hybrid Elements Thread and Projection in the Physical Presence Thread.
- In the Software Development Process: Beta Testing (initiation) in the Phasing Thread and Telegram Group in the Feedback Thread.

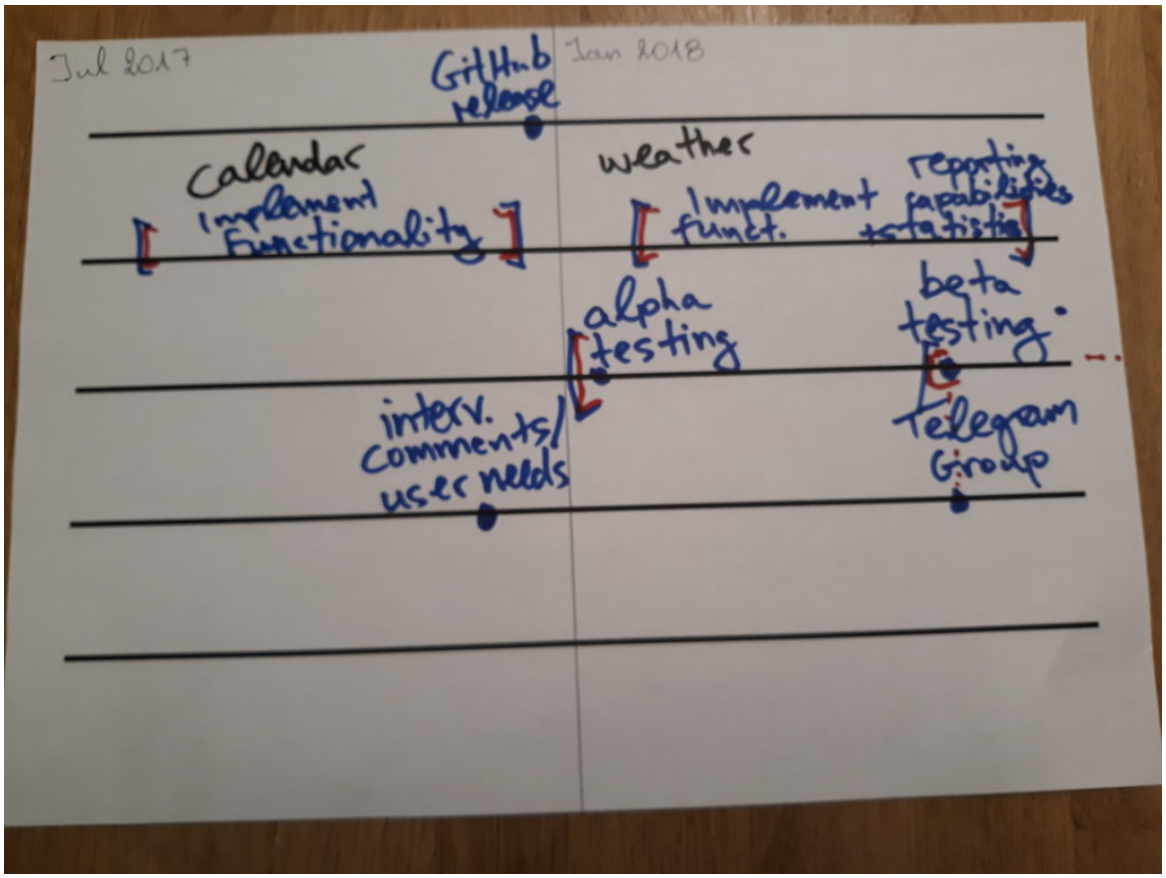


Figure 3.4: Software Development, 3rd and 4th period - recorded Actions

3.1.2. Second meeting with beta testers

In July 2018, on the weekend before the netCommons plenary meeting, the netCommons consortium together with some advisory board members and other visitors from abroad visited the area West of Olympus as guests of the Sarantaporo.gr CN to get to know the region and the progress of the Network[1]. The AppLea designer team seized this opportunity to organize a meeting with the app's beta testers, in order for Aris to discuss with them the latest version of the app and to request additional feedback for subsequent iterations.

We met in Pythio village, in the afternoon of July 8th. Participants were Aris, Alexandros, Dimitris Ntallas and Sakis Katis. The app's third beta tester was not able to attend. In the course of the conversation George Kleisiaris of Sarantaporo.gr made a short intervention and, what was a pleasant surprise, a new beta tester joined our team, Vasilis, a local farmer who was present at the presentation in Flambouro last March, but was

[1] https://www.netcommons.eu/?q=content/sarantaporo-conference-building-community-community-networks

hesitant to participate back then. It seems that in July the timing was right and perhaps the setting was more familiar and less intimidating for him. He was added to the AppLea Telegram group.

Since last March, Aris had made plenty of improvements and had incorporated comments from beta testing. He had published another iteration between meetings, but the Telegram group had remained inactive. The designer team thought that it was best that we sat down with beta testers and discuss developments on this occasion. The most central and critical points of the conversation are summarized in the following list.

- Participants had an interesting, detailed discussion about the *default lists* of products that farmers use for crop-dusting and top-dressing. Aris had done research on the names of the products and asked the testers if these names were accurate. The question arose whether it would be correct to include lists of brand names in the app or, instead, lists of products' active ingredients. On the one hand, the community of users shouldn't think that the AppLea team is promoting commercial products, but, on the other hand, the list of active ingredients might be unintelligible and harder to use. Participants argued that the list should be as default as possible, alleviating the users from having to insert elements themselves. Even though the brand name list was nearly 'voted' for for reasons of simplicity, Sakis insisted that both lists be included, leaving the option to the users to choose. It was an exemplary participatory process that resulted in a unanimous, informed decision.
- Sakis made much reference to *the community*. He demonstrated that he is not only focused on the use that he personally will do and about issues that he faces, but that he also reflects on what future users from the community would think when they'll use the app. Specifically, Sakis mentioned the community in relation to several items and in articular:
  - *Irrigation and herbicide*: "an individual farmer doesn't need this info, but it could be useful data sharing for the community";
  - *List of brand names*: "it shouldn't be imposed, people shouldn't think we are promoting products";
  - *Control on the form and content* : "let's leave it like this now and perhaps later other users ask for different things".
- Dimitris in particular showed an increased understanding of the issues related to the app being easy and friendly to use, namely the necessity for accurate and concise default options to ensure 'lightness' and subtle content, albeit combined with the possibility to add notes and elements. Dimitris got a smartphone for the first time last March, especially for the app, however he was been using his new phone quite a lot since then, as he said.
- Sakis was worried about privacy. He explicitly asked if all notes and photos that a user takes are private and specified that it should remain optional to share anything. He also made a remark, half-serious, half-joking, that the tax service shouldn't have access to the data.
- Sakis in particular was very impressed with the improvements on the weather component that Aris had done and with the level of detail on local weather conditions to which the app gives access.
- George brought up the importance of the possibility of users to work on the app offline, when they are in their fields. This is already possible and when they are online again, the actions that they registered are saved on the server.
- According to the feedback that Aris got, he was asked to adjust the default lists of options for top-dressing and crop-dusting, as well as the relevant quantity options, to enable access to the filtered history of a user's actions and to the history of rainfall in a specific location.

Participants discussed matters and made choices taking into account technical limitations, the conceptual and visual facilitation of users, and political parameters with regard to the app's reception and adoption. Beta testers are aware that the app is far from being ready or perfect, but acknowledged that it is on the right track.

Once more, it was interesting to see that different elements, which were considered as self-evident for the farmers, were not intuitive for non-farmers and, thus, needed to be expressed explicitly in order to be incorporated

in the app's logical structure.

At the same time, while AppLea's beta testers agreed on many points, there is a divergence of opinions between them on several issues, according to their crops and individual needs, their technical skills and experience, their political views and their vision. This was ascertained also by the interviews in December 2017 and the app's presentation in March 2018. For the purpose of being faithful to its initial principles of participation and inclusiveness, the design team needs to ensure the modularity and adjustability of the app in order to cater to users' varied needs.

Regarding the application of the participatory design methodology in practice, the Hybrid Space Design and Project Sustainability[2] concepts still seem to be either little relevant or premature. In contrast, most concepts discussed in the Community Participation and Software Development Processes are relevant to the process that has started with Sarantaporo.gr members in December 2017.

Namely, during the meeting presented in Sec. 3.1.3 in the Project Score the following Actions were added:

- In the Community Participation Process: Strengthening of Collaboration in the Trust Thread, Summary in the Document Thread, Face-to-Face Discussion in the Training Thread (there were doubts whether to be included in the Engagement Thread) and Brainstorming Process in the Listening Thread.
- In the Hybrid Space Design Process: Decisions on Privacy and Usability in the Needs Thread.
- In the Software Development Process: Establishing Prioritization in the Functionality Thread and Beta Testing in the Feedback Thread.

3.1.3. Global synchronization point

On July 13th 2018 Merkouris, Aris and Alexandros met on the premises of AUEB to enact the first "Global Synchronization point", as described in the methodology focusing on the progress of the AppLea development and to draw up the methodology Project Score.

Soon after the discussion started, Merkouris and Aris suggested to rename a Process. They though it might make sense to change 'Software Development' to 'Software Design and Development' because, they argued, the app's design is also part of the participatory process.

Everyone agreed that it would make sense, considering the density of Actions or the app, to divide each sheet of the Project Score into two 6-month periods. Next, participants wondered about and discussed the duration of some Actions, whether they could be represented as starting in a point in time and having duration instead of being represented as three separate Actions, or if the name of the Action itself should show duration (e.g. 'Informal Communication' instead of 'Meeting'), or how users could capture two Actions that have duration and are progressing in parallel (participants proposed different colors for this last one).

At several points of the brainstorming process to draw up the AppLea project Score, participants had doubts about how to 'translate' an event to an Action, how to name the Action, in which Thread and Process to capture it, what duration to give it, how to illustrate the effect that it had and if it was related to other Actions, and whether the sequence of Actions was accurate the way they illustrated it. Another conclusion that Merkouris, Aris and Alexandros reached is that it's hard to suggest Actions before one uses the methodology sufficiently. Different doubts and dilemmas they had were regarded as useful feedback for the reviewing of the methodology.

The team tried to have *conceptual consistency* of their Entries. In order to make sure that, firstly, Actions have a minimum degree of meaning and impact, secondly, that they fall under existing Threads and, thirdly, that they can be pinpointed in a moment in time. However, there was a case of an Action where participants considered it necessary to record an event linked to the CN as an Action ('Positive parallel developments') because they thought that it influenced the app, even though it wasn't an Action of the designer team.

[2]However, without made explicitly to the local community, several key actions have been taken to ensure the project sustainability like the invitation of external experts and the collaboration with Sarantaporo.gr for securing additional funding.

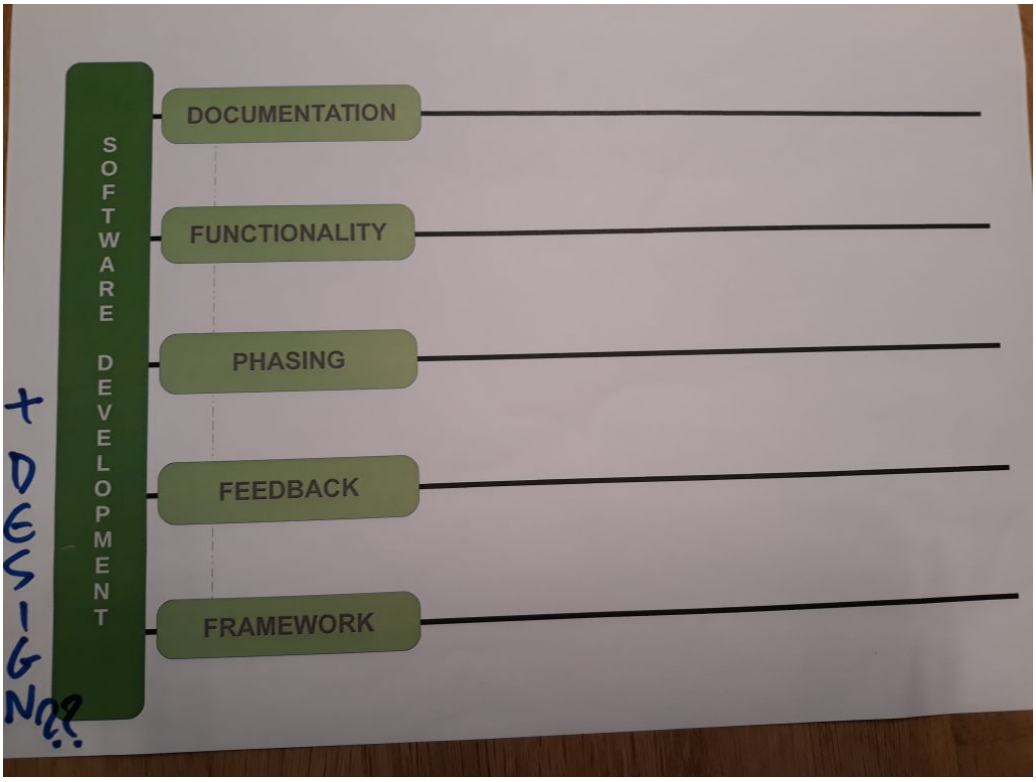


Figure 3.5: Software Development Process –Should it be renamed to include Design?

Also, Merkouris suggested that the CN Coreteam and Local stakeholders be viewed as part of the Community. Participants considered that Actions that contributed to the Sustainability of the CN are not directly related to the app's development and, since AppLea is supported thus far by the netCommons project, they concluded that there are no Actions yet linked to Project Sustainability. Finally, it was deemed necessary that a designer team should reduce dependencies to a minimum, only to illustrate the most important ones.

In the end, participants tried to prospect the team's future Actions and to capture them on the Score.

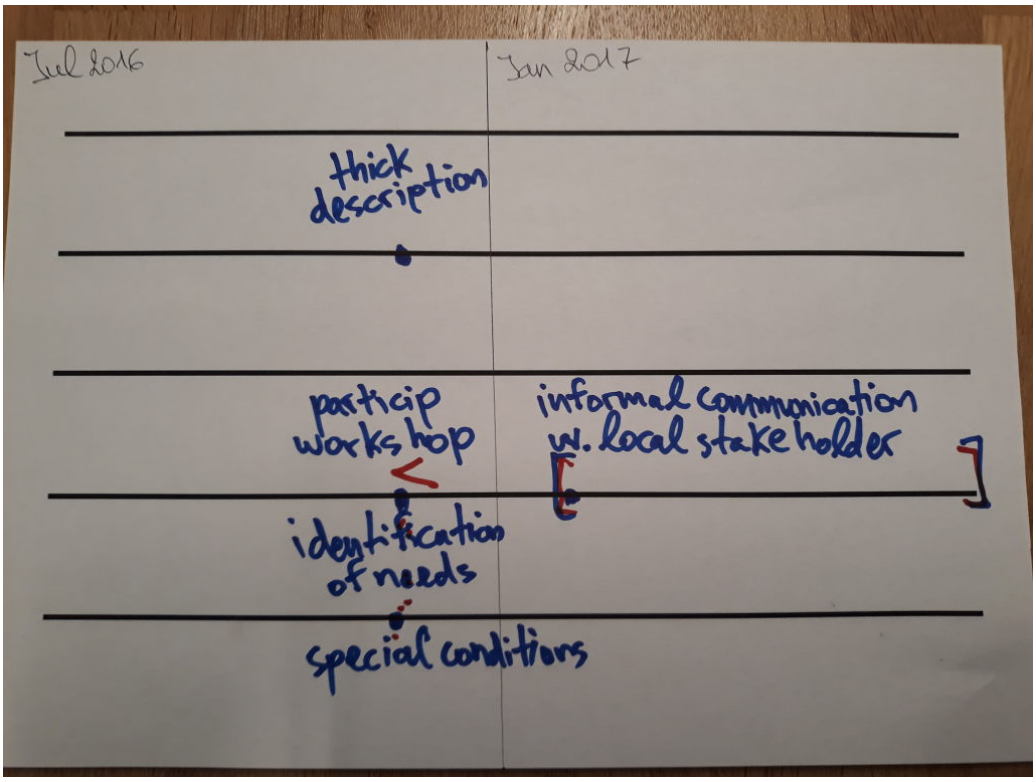


Figure 3.6: Community Participation Process, 1st and 2nd period - many repeating Actions or one Action with duration?

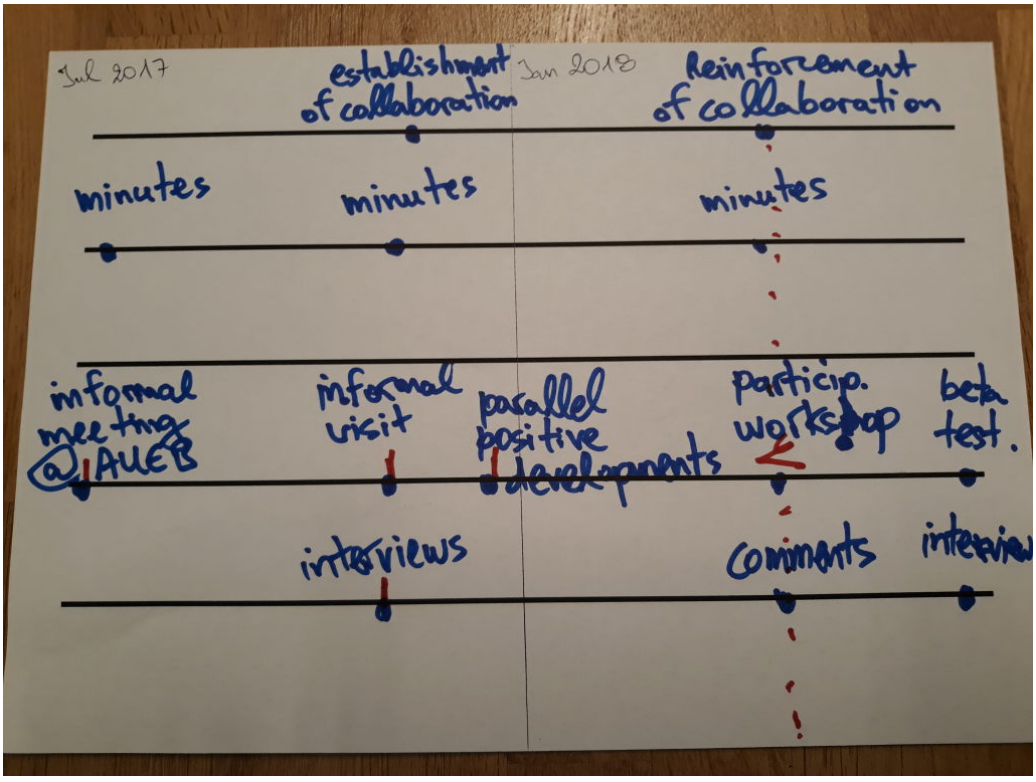


Figure 3.7: Community Participation Process, 3rd and 4th period - recorded Actions

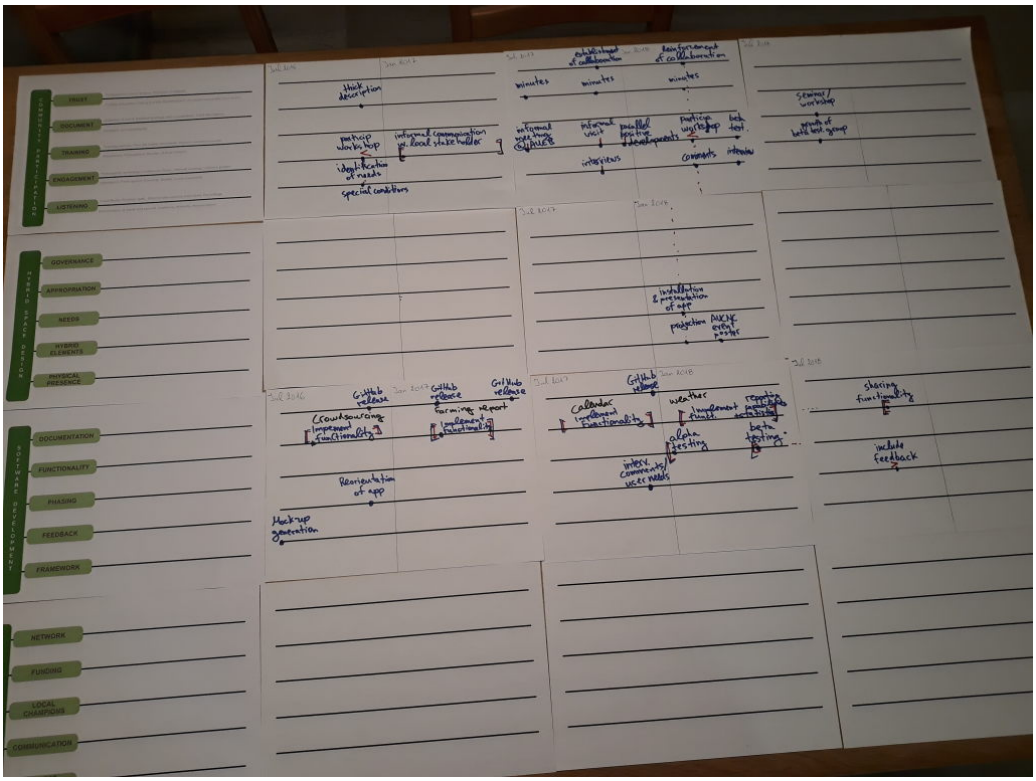


Figure 3.8: Overview of the Project Score

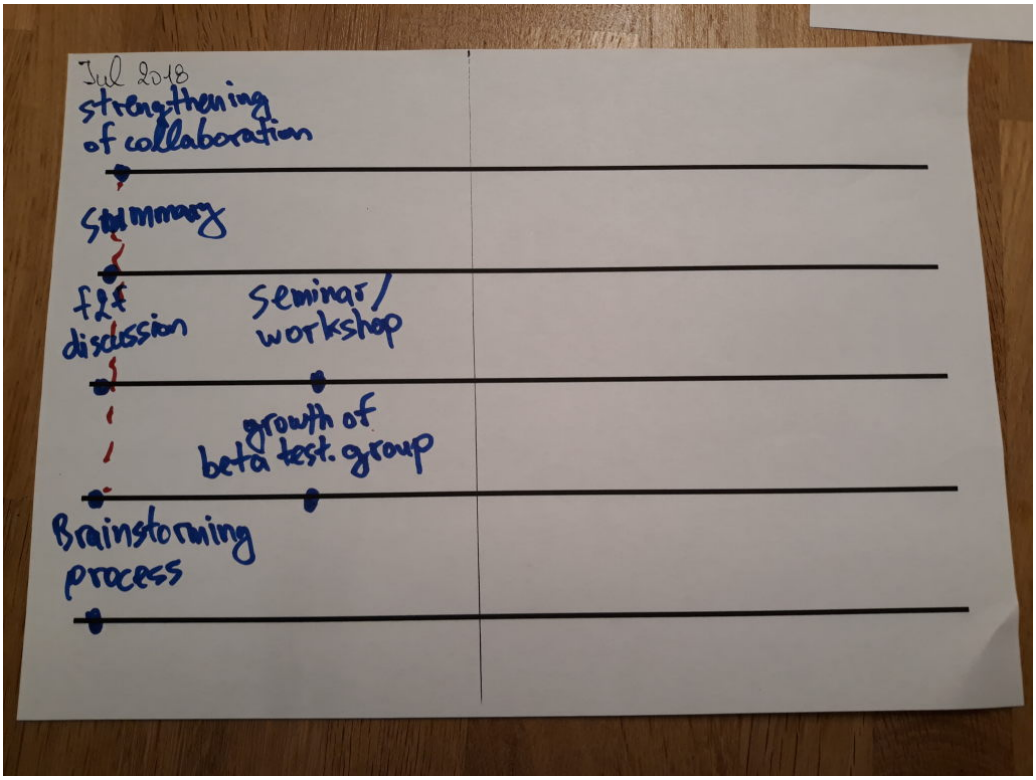


Figure 3.9: Community Participation Process, 5th period - recent and prospected Actions

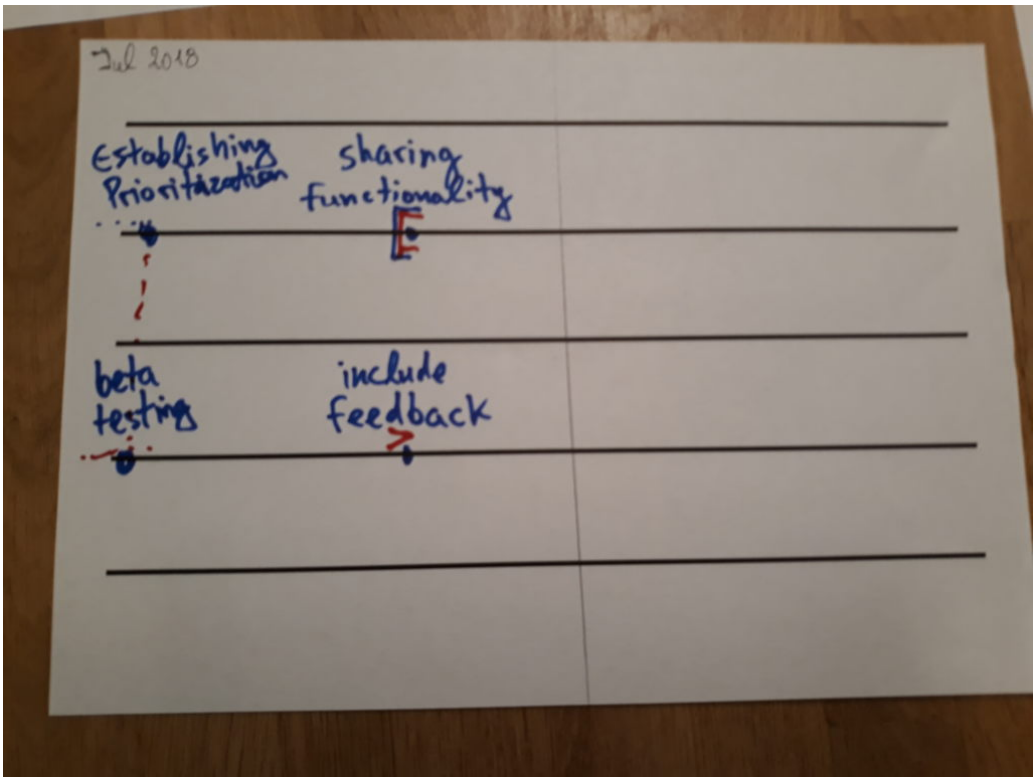


Figure 3.10: Software Development, 5th period - recent and prospected Actions

3.2. Ninux.org – PeerStreamer-ng

The methodology was presented and partly tested in the ninux community of Florence in May and June 2018 (M29–M30). Leonardo Maccari is an active member of ninux Florence, making this interaction possible. The application of the methodology followed two paths, the first was aimed at the use and deployment of PeerStreamer-ng (PS-ng) in ninux, and possibly to further PS-ng development; the second was instead aimed at the direct collection of raw feedback from the community. In the former case, Leonardo engaged the community in the effort to organize new activities that would lead to the use of PS-ng to support the video streaming of live events in Florence. This activity was lead by a member of netCommons, and thus it was mostly imagined as a guided attempt to apply the methodology. In the latter case instead the methodology was proposed to some of the the ninux members that are most active in developments and some direct feedback was collected on how this community see the application of the methodology at stake. The long-lasting cooperation between ninux and netCommons was instrumental to the successful set-up op of this activity.

3.2.1. Application of the methodology for PS-ng deployment in ninux

In spring 2018 ninux teamed-up with three more organizations from Florence: Cirkoloco, Libera Informatica[3] and Restarters Firenze. The first organization is a voluntary organization with the goal of re-inserting in society people that have been marginalized for being psychologically "border-line". The second one is an organization known in the city for supporting Free Software, and the third one focuses on re-use and repair of hardware. The link between these organizations came from the fact that the place where ninux physically meet (a public space managed by a pool of no-profit organizations named "ex-fila") is now managed by Cirkoloco, which uses it to organize public events and took the management of a bar.

Somehow informally and ironically a link emerged between the context of Cirkoloco (psychologically border-line people) and the other organizations, primarily made of computer geeks (calling and in some sense seeing themselves border-line too). This cooperation was set up with the goal of finding common ground, and the organizations decided that a way to cooperate was to organize, in ex-fila, a set of hands-on hacknights on themes related to open source and new technologies. Leonardo participated to the organization of these events, with the final goal of contributing to the growth of the community, and as a chance to test PS-ng. In this deliverable the initial process of setting-up the context and getting to the decision of using PS-ng is described. The full feedback on PS-ng will be described in more details in D3.5 at M36.

Without getting into too many details, the process included several threads, reported in Fig. 3.11. Key to the success was the trust building obtained with informal meetings and on-line participation. This contributed to realizing 5 nights, in which someone of the three organizations organized a public event on some IT topic (from bitcoin to 3D printing). Following the guideline of the methodology, focus was put on the creation of a physical shared space. The ex-fila bar was chosen for the hacknights. While formally the management of the bar is delegated to Cirkoloco, the bar was adapted (with projector and screen) to be able to host these events. This was a collective effort that later on justified ex-fila to become the de-facto meeting place for the group, which is particularly important for ninux, because in ex-fila there is a ninux network node. In parallel the organizations started to discuss how to intensify their activities together, for instance, through creating an umbrella organization that could be used to participate to projects and grants, among them Google Summer of Code grants, Internet Society (ISOC) funding or Réseaux IP Européens – Network Coordination Center (RIPE-NCC)[4] projects. This discussion is still on-going but is a key point for ninux, as, so far, ninux people never decided to make the step of formalizing (in a legal sense) what ninux community is.

At the end of the "season" (June 2018, before the summer vacations) another meeting was held to program next year's activities. The organizations decided to continue, try to enlarge their audience, and experiment with

[3] The name of these organizations translates more or less to "*mad circus*", and "*free informatics*".
[4] https://www.ripe.net/

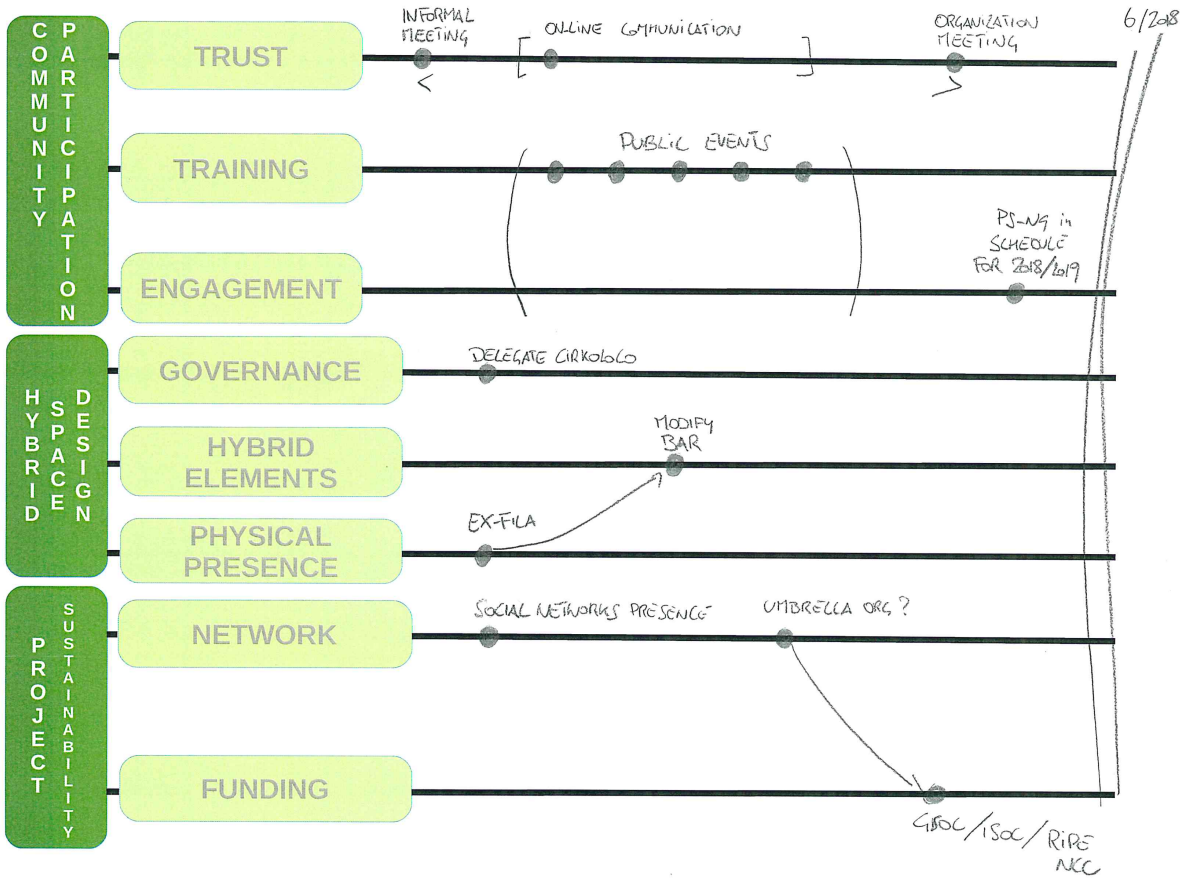


Figure 3.11: A description of the process using the netCommons methodology.

PS-ng. In general, the feeling of this experience was largely positive. Among the positive feedback we can put the large participation to the events (about 20 people per night in average, which is the maximum number the bar can handle). In particular, compared to a similar set of events ninux organized in 2017, Tecnolokia was much more participated, due to the mix with other organizations that led to a hybridization and opening up of the audience.

So far, the methodology was used basically to organize the premises of software deployment and development. This process took months, but created a core of people that are now involved in experimentation and interest in both the concept of community networks, and the potential impact of local applications.

In Autumn 2018, PS-ng will be put in the loop. As sketched in Fig. 3.12 this will consist, first, in sharing documentation among the participants, who will have to install PS-ng. Then an internal test will be done, separated by a real event, in order to test how the network support the streaming. This will feed back on the development process. Finally, a loop made of streaming live events, improving the code, and documenting the streaming session will be repeated for some of the hacknights, depending on the availability of people and on the number of hacknights that the organizations will produce.

3.2.2. Direct Feedback from ninux Activists

Sec. 3.2.1 documented the application of the methodology by one of the netCommons researchers in the ninux context. In parallel, we asked for direct feedback from ninux activists, without the guidance of somebody from the project. We sent them the available material, and asked for a feedback on their understanding and on their opinion on the applicability of the methodology. Three people were asked for feedback: we collected and

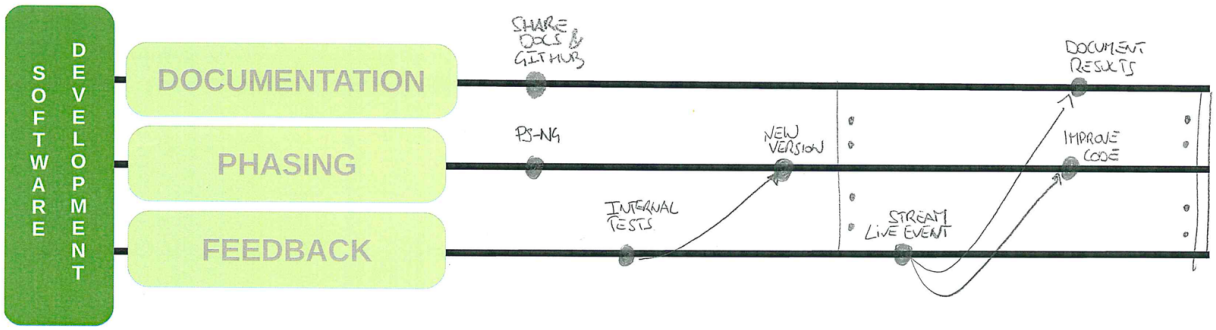


Figure 3.12: Planning of the software activities after summer break.

reorganized their informal report merging it with the report from the application on the Tecnolokia experience, obtaining the summary reported hereinafter.

In general ninux activists considered the methodology interesting but unclear and hard to understand. The current version[5] introduces too many concepts and the notation is too "rich", confusing the non-expert. The common feeling was that it is too complex to get enough information to start using it, the learning curve is too flat and slow, and indeed has a very high entry barrier. Another general feedback was that the methodology recalls some other existing ones ("Design thinking" was explicitly mentioned), which are generally applied in professional environments. In that context, people are forced to follow a schedule (whether waterfall, agile, or any other model) so that enforcing a methodology is easier. In a community network, instead, people do things for a mix of passion and fun, and thus it might be hard to enforce timing and dependencies. Some kind of "master", like in the SCRUM[6] methodology would help to keep the pace.

More detailed feedback was given on some formal details contained in the methodology, the following paragraphs summarise them and include tables with detailed feedback.

Notation. The notation is too rich and it is not easy to distinguish one item from another. Examples are:

- it is not clear how to use divergence/convergence signs. Should they be used as a retrospective when evaluating past actions, or should they be imagined as a way to plan future ones?
- the relationship and duration signs are similar (parenthesis), it is not straightforward to understand that one is intended to collect multiple related actions while the other expresses the span of time of a single action;
- the "o" symbol beside the duration symbol is not explained;
- A "loop" symbol would be beneficial, for instance, using the "refrain" symbol in musical notation as used in Fig. 3.12.

Placement of components. It is not clear where to place some of the items. For instance, it is not clear if metrics need to be put on the score and where. Also, it is not clear if one can have more than one line representing the same thread. This is necessary when two actions of the same kind go in parallel, for instance, one has regular meetings and on-line activity. Since the '[' and ']' symbols seem to be intended to comprehend

[5] The feedback was collected on the methodology as described in D3.3 and a simplified version; we expect that the publication of the user-friendly booklet at the end of the project, whose design is based also on these experiences will improve the situation.

[6] http://scrummethodology.com/

Issue	Suggestion
Too rich notation	Remove some pieces of notation and ensure that for all remaining ones there is one example showing how to use it
Unclear notation	Ensure that for each piece of notation it is stated clearly if it is needed to plan ahead or to use as a retrospective

Table 3.1: Feedback on the notation.

only one item (as in the case of on-line communications in Fig. 3.11), it is not clear how to represent two of these actions happening in parallel.

Issue	Suggestion
Unclear placement	Give examples of placement and use of several metrics
Parallel actions	Give examples of actions that happen in parallel

Table 3.2: Feedback on the component placement.

Overall Complexity. There are too many concepts defined in the methodology. While it is stated that it is modular and only some parts can be applied to a specific situation, before one is able to make this selection, he/she should actually read it all and understand what are the pieces that are needed. This is intimidating for someone that does not have a strong motivation to use it. Also, some concepts seem to be overlapping (TRUST/LISTENING, NETWORK/ENGAGEMENT etc.) and may be merged. Finally, as the number of terms is very high, it would be beneficial to associate icons or graphics to the important ones, to help the reader follow the stream.

Issue	Suggestion
Too many concepts	Reduce the number of threads and merge the ones that are overlapping
Flat organization	Make the threads hierarchical so that the reader can read at two levels of detail, one that is quick and introductory, and another that can zoom into specific concepts. Having at least a basic understanding of the methodology without going through tens of definitions is key
Graphically distinguish concepts	Use icons and/or side notes to single out key concepts and recall them in the following of the text

Table 3.3: Feedback on the complexity.

Text Description. The methodology was delivered as a project deliverable with all its formalization and complexity, but it is also meant to be condensed as a booklet for practitioners that are not generally keen to read a one-hundred pages deliverable. At the time of writing, the booklet was not fully ready, so the material that passed to ninux was the deliverable and a stripped down version with less details. Even though ninuxers are

practitioner and should not be exposed to the full, complex document, we collected useful feedback that can be useful for the next iterations and to finalize the booklet in particular:

- Improve the description of the methodology: one ninuxer said that after a first read it was not clear what the methodology was for. He had to discuss with us to understand its use. In general, the methodology should have a one-pager that gives the whole story to the reader, while right now it is an introductory text that explains the genesis, but it is not direct to the point. In the introduction do not rely on external citations (like the methodkit, which is not well known). The relationship with the "Yes to the mess" book and the music score is inspiring, but takes too much space and subtracts potentially more useful text from the introduction.
- There is a long list of concepts that is hard to digest. Whether one details all of them (and in this case it becomes too long), or else a long bullet list of keywords is not readable. This is strongly connected with the overall complexity of the methodology.

3.2.3. Final Remarks

The application of the methodology in ninux gave two key insights. The first one confirms that when used as an abstract organization of the work and of the principles that one must follow, it is a valuable instrument. Since hackers tend to have a very techy approach and focus only on the technical details that are more important to them, having a palette of principles that they must keep under control during the process is very useful. This helps organizing the work in a way that maintains at the same level several kinds of actions and achievements, including the group conviviality and organization, which may be unrelated with the success of software development at first sight, but are key in situations in which the participants are all volunteers.

The second one is on the formal structure of the methodology itself (notation, usability etc.) and it goes in the direction of simplification. Activists found it too complex in general and suggested several ways of improving/modifying it.

3.3. Guifi.net – Cloudy

The proposed participatory design methodology has been tested by the UPC team involved in netCommons as participants and part of the Barcelona community of guifi.net. This activity, from the guifi.net perspective, is part of the participatory design, development and evolution of the Cloudy community cloud software system, the community cloud experimentation in guifi.net, and the design and development of its commons governance and sustainability model. Along the way, we estimate about 60 individuals were involved with different roles and different levels of involvement, not all of them being necessarily exposed to the methodology developed in D3.3. The UPC team has led the process, typically with an active team of 2-4 part-time people on average, with diverse smaller contributions from approximately a dozen people over the last five years. This is a long-running activity that already started in 2014, with two main iterations, one prior to the netCommons project, and another along the project, mainly in 2016-2017, with limited activity of maintenance and small refinements in 2018 where the netCommons methodology was considered and partially introduced.

We discuss first how the (recent) development of the netCommons methodology has influenced our design process, and next we mention ongoing activities, represented by the redesign of the Cloudy user interface.

3.3.1. The guifi.net community

There are five main stakeholder groups according to their roles in the ecosystem and their motivations for participating in it: the *volunteers*, the *governing bodies*, the *professionals*, the *customers*, and the *public administrations*. While most of our participants are volunteers, we have also worked with a few professionals, mainly to develop and evaluate the feasibility aspects.

Internal communication in guifi.net is structured around mailing lists with global, territorial, and thematic scope (open by default); social media (open by default with a few exceptions to protect sensitive information); and face-to-face meetings that play a key social role in community engagement and knowledge sharing, some of them local and frequent, the guifi-labs (weekly in the case of Barcelona), and global (yearly, hosted by different local communities every year).

Guifi.net is a citizen-driven initiative, characterized to be bottom-up and decentralized. The majority of interactions and contributions in the different communication activities involve volunteers. Professionals have their own lists, with low traffic, and most of their interactions happen face-to-face, related or as part of the economic compensation tables.

The core activity of guifi.net revolves around the expansion of the network infrastructure. For that activity, and after many iterations, the community of that practice has consolidated a pattern of working together, so called the "wheel", a repeatable process pattern that has not changed significantly for the last decade. The process has not been defined formally by the community, but the original description by Ramon Roca, extracted from the guifi.net web site[7], translated into English (the original is in Catalan) and reported in Appendix A for easy reference, describes quite well its "philosophy" and its initial definition centered on the problem around network deployment, and related network services and applications.

We can say that guifi.net has developed its own participatory design process based on its own incremental process pattern, the "wheel", combined with other features inspired by design thinking, lean principles and agile development methods. This process does not have a formal base to deserve being called called a methodology, and is applied to some degree to a wide range of activities in the community, for network deployments, but also for the development of software and services. The process develops around a combination of informal face-to-face meetings, the guifi-labs, even more informal bar discussions, the birra-labs (beer-labs), online tools for synchronous (e.g., Matrix chats) and asynchronous communication (mailing lists mainly). Face-to-face meetings have rough schedules (people come and go over a period of a couple of hours) with an open agenda and topics that are discussed by everyone or in smaller groups. As other mostly techy communities (e.g., IETF.org), the community is based on many small and concrete contributions (as IETF says "we believe in rough consensus and running code", guifi could say "we believe in action"), coming (nearly always) from volunteers. After some previous experimentation, the design thinking phase, done individually or in small groups, these meetings are the equivalent to project inception meetings in software development, where requirements, features and decisions are made for a software application or a new network deployment, roughly following the lean design principles, with the members of the community involved or interested acting as product owners. Processes evolve organically, as in a lean startup, get defined and redefined by variable and short-term interest and experimentation from these volunteers. Software gets developed in sprints, and deliver results, fail or stall according to short-term objectives, and usually no dedicated "driver" (except for governance-related tasks) but always involving members of the community that collectively represent the role of product owner. We can find similarities in other self-organized, participant-driven and evolutionary processes or practices such as Open Space Technology[8] or Unconference[9]. Activities that succeed in delivering a result tend to be agile, relatively short and simple, comparable to the idea of sprint (or iteration) in the SCRUM software development framework, but more loosely defined.

As with other social communities, participation in the guifi.net community tends to match the power law pattern of participation[10] where a healthy activity is one with many participants with low level of engagement (e.g., they just subscribe, read, comment), the collective intelligence, and a few with high level of engagement (e.g.,

[7] https://guifi.net/node/7934

[8] https://en.wikipedia.org/wiki/Open_Space_Technology

[9] https://en.wikipedia.org/wiki/Unconference. Although guifi.net may not be characterized as a software development community, there are similarities with Free/Libre Open Source Software (FLOSS) communities and lessons described in https://en.wikipedia.org/wiki/The_Cathedral_and_the_Bazaar

[10] A "low threshold participation amounts to collective intelligence and high engagement provides a different form of collaborative intelligence" http://ross.typepad.com/blog/2006/04/power_law_of_pa.html

one or few leaders, proposers, designers, implementers, decision makers), the collaborative intelligence. Activities with the right balance of many low and few high engagement participants are those that tend to succeed in guifi.net. Therefore, activities in the community are characterized by being open to comments by many (participation), with the online and face-to-face meetings used for that purpose, but typically developed by a few (e.g., proposed by one, maybe designed by one or two implemented by a handful of people, while considered useful and used by many).

3.3.2. The research and design group

The method adopted by UPC researchers and other developers to interact and work with the community is inspired by participatory action research [6] combined with iterative experimentally driven research [7], evolutionary design and agile software development ideas[11].

A group of researchers (professors, research staff, developers, PhD and master students) from UPC has been part of a team with a group of volunteers, with good knowledge of guifi.net and programming skills, to design together proposals, conduct experiments, and discuss results and implications. This course of action is going on since nearly 10 years, as soon as UPC members got involved in guifi.net, thus it is a well established methodology, and it is difficult to imagine a "revolution" in the guifi.net way of working and developing.

In general, the interactions with the community took place in two mailing lists for discussion and mutual support, combined with focused groups in several weekly community meetings, both in the guifi-labs around Barcelona on Thursdays, and regularly at UPC research lab around lunch time on Wednesdays. Group discussions were around the planning, co-design, transformation, and result phases of the action research, with a focus on community cloud deployment and the use of the software system, services, and applications. We used a combination of larger meetings to collect diverse opinions, and small group meetings for analysis and decision making. These interactions were useful to better understand the needs and preferences, to obtain feedback about new features and changes as the software evolved, and to refine the community cloud model and its governance instruments.

The overall direction was driven by the joint interest on designing and developing a community cloud infrastructure for sharing application services and content, under the same fundamental principles of guifi.net, that we found also apply to a CC: be fully inclusive, that is, ensure the openness of access (usage) to the infrastructure, and the openness of participation (construction, operation, and governance) in the development of such an infrastructure and its community. The working cycle had a periodicity of a few months, without strict deadlines. At every cycle we jointly defined short term projects as Cloudy components according to the interest of the community and the research agenda (academic research activities and projects). Development happened over short term, typically two week sprints. For every phase or type of activities we considered different methods and tools to coordinate the design and implementation.

As mentioned, all of this happened since well before netCommons started or the theoretic formalization of D3.3 methodology was defined, but our course of action has been influenced by netCommons research and we deem that also D3.3 has been positively influenced by the interaction as activities developed and evolved at different levels.

3.3.3. The use of the methodology and feedback

As described before, guifi.net follows a bottom-up, purely decentralized process that evolves organically. When the methodology proposed by netCommons was discussed with several community members, they found useful to have a palette of principles and tools, but also identified three main points of mismatch on its focus compared with what they consider the best practices of the guifi.net community:

1. Initial design, e.g., the initial plan of the activity;

[11] https://en.wikipedia.org/wiki/Agile_software_development

2. Guidance, e.g., the need for appropriate counsel and inputs through structured activities;
3. Shared state, e.g., how the project is documented and how this documentations is maintained and updated.

It appears that the methodology developed by netCommons has a structure that is difficult to reconcile with the shared values and common processes in the guifi.net community culture, rooted in informal interaction, experimentation, lean principles, the power law of participation, decentralization, and informality.

Volunteers have many interests and short-term focus. For an external, the way guifi.net development groups work and evolve may appear a bit like a 'collective attention deficit hyperactivity disorder'. Community volunteers want to be aware about what's going on, they can give opinions and make small contributions, develop a few features in short sprints, but soon they may move to another topic or activity, and this is accepted as customary behaviour, and it is not to be blamed. Instead of extensive backlogs of features and long software development cycles, short term activities are preferred, like planning link deployments following the "the wheel" process, adding a new feature or correcting bugs for a software application. Therefore, for larger projects, it is ideal to divide them in small pieces, where one or a few volunteers can set-up a short-lived open team to focus, independently from others. The global outcome will emerge as the juxtaposition of the pieces, again without the need of any central coordinator, but the involvement of a variable set of developers. Probably this works sometimes since most volunteers have long experience and are used to bridge gaps.

In a couple of laboratory meeting, we explained the main points of the netCommons methodology. People recognized a lot of interesting ideas, ingredients, recommendations, but at the same time they found it too complex or too structured, with excessive overhead to apply in the lightweight iterations of an evolutionary design. They discussed about alternative models, such as agile development, partially tried in the development of software tools, but they didn't find the time or the motivation to adopt a new methodology, especially the formalization of steps and the sort of rigid structure they see it, in comparison to something that works for them without planned and detailed design, central coordination, and structured activities. It is nice to have proposals, but the community prefers lightness over formalization (and overhead), and experimentally-driven design over a detailed design. The developers' community of guifi.net thinks that the lighter, lean process used till now that "just works". That process flows naturally in the community and does not need any explicit coordination, as people find it a better match and choice to stick to, instead of trying to switch (self-impose?) a different, maybe better process, which however limits the spontaneous evolution of ideas and is felt to guide developers too much, in ways that are not perceived as necessary. The process in the guifi.net community is the result of an evolutionary process, and no one can decide or impose the others to adopt another one, furthermore the current way of working is considered fun, and effective enough for them. Fun, informal social interaction, learning through experimentation, and reputation based on small projects that work, are all important, probably elementary ingredients of the way the community works. Efficiency, measurable outcomes, or even the definition of what is socially good are not a main concern for this community.

This is not to be perceived as a conflict or a failure for netCommons, as our goal was not to impose a 'better' design methodology, but it is the acknowledgement that 'old-growth' communities have probably already evolved into a stasis, finding an ecological equilibrium, just like well-balanced ecosystems. The methodology studied in netCommons instead, may be more suitable to boost participatory design in new communities or to implement a cooperative development process in those communities that do not have one or are still evolving trying to find an equilibrium. People recognized the topics mentioned in the methodology, and it seemed to help to consider and not forget some aspects, as the documentation refers to them as "*things to think of*". However, the general attitude was "Why changing what already works?" Changing methods in a well established community seems too daunting, unless in a crisis that identify the main features of a methodology as their solution, especially in one that is decentralized and believes in it. In contrast to this observation, indeed, the guifi.net participants in our development group finds that the proposed methodology may be useful for newly created groups or to groups created for a specific purpose. New groups often lack a working methodology and gets stuck into non-productive initial phases, so they should benefit from guidelines, framing of work cycles and phases, and tools that suggest structures and processes. They may also find pleasant such a framing that confirms their work

and intuition, and likewise have fun following it. On the other hand, unless facing any crisis, well established groups may not feel the need for such guidance, or feel that a formal framing drains the fun or the freedom out of the activity.

Activity in the guifi.net community is not driven towards efficiency, but it is about openly exploring, learning and experimenting as the network infrastructure evolves, participants try out FLOSS services, or develop their own services or applications. In this context, we find short interactions, lightweight projects and activities, unstructured or lightly structured, with no formal coordination team, but ideas that develop into experiments, that sometimes have results, but are always exciting[12]. Work is performed by a small focused team with members that freely come and go, and activities that get reshaped by the participants in each design cycle or development sprint. This way of working diverges from what the methodology studied in netCommons assumes: "*The core idea behind the participatory design methodology is the description of the overall structure of a potential project and a set of key elements among which the leading team should choose from, according to the context and available skills and resources*". This framing appears to value formal, prior, global design, reminding the "waterfall" model, over a decentralized, minimally planned, diverse (power law of participation), reactive and evolutionary, lightweight and agile process the guifi.net community is used to follow and feel comfortable with. Once again, this is not necessarily a negative characteristic, it's just that it does not map easily with the main culture in the guifi.net community. However, it can be an added value in projects where the local community is engaged into projects they are not familiar with, and where results are fundamental to bootstrap the community itself, or to introduce in the community (r)evolutionary concepts, infrastructures, or services.

Given the characteristics of the community, we had to decompose the community cloud project into small topics and tasks and introduce these for discussion in the context of guifi.net, and the UPC research group that acted as part of the Barcelona guifi.net community. In contrast to the assumption of proactiveness and a-priori formulation of the design activity (formulation of strategy, framing design process, guidelines for events org, support tools, metrics and evaluation), we had to follow a reactive model, adapting and reformulating to what the volunteers wanted or did, driven by action and experimentally-driven research methodology and the timing and expected outcomes of the supporting projects (netCommons in the last 2.5 years). Therefore, our activities, instead of being "*... carefully planned, documented and reflected upon based on a set of available Methodological Elements: Actions, Evaluation Metrics, Guidelines for supporting the implementation of the different actions, and (optionally) expected Input & Output from / to other Actions*", were characterized instead as lightly planned (evolutionary: evident high-level long-term directions, clear and sharp short-term detailed actions), little documented (focus on interaction), less formal (little focus on methodology and more on flexible experimentation and learning). Therefore, we had to act in ways such as carefully react, adapt, and focus more on explaining and replying on demand rather than documenting (the community does not "believe" in detailed documentation but on interaction and action), decide and reformulate mainly the next step, discuss again, define metrics and actions as we go, learn from mistakes and not from upfront design. In other words, we had to adjust to the values of a hacker culture, instead of the office, software development, or the academic culture we are used to in our research group. As one member put it with a grin: "*we are here to try and learn, not to work as in an office.*"

We learned that even though the new methodology can be useful for a starting community, particularly when involved in collaboration with external agents (like researchers, funding agencies, governmental actors) that require formality and predictability, this methodology embeds many ways and values that may or may not fit with those from established communities. For these reasons, in our experience, replacing the methodology in our community appeared as artificial (not developed evolutionary in the community, adapted to "its local needs and conditions" in E. Ostrom terms) and foreign (external). Furthermore, when a community has a sense of crisis, they may be willing to adopt a new methodology or explore new paths, but otherwise the one in use, specially if felt as "their own" seems the best fit, even when not particularly designed for a specific purpose. In

[12] As Richard Feynmann said: "Physics is like sex: sure, it may give some practical results, but that's not why we do it."

addition, it is very hard or nearly impossible to change methodologies for a large and decentralized community, unless you are one of its leaders, and not just “normal” members or external agents. Anyway, the netCommons methodology was useful to think about the guifi.net community process, and help the group be aware, reflect and consider enhancements for its own methodology.

Despite the netCommons methodology could not be adopted and applied entirely, many of the elements were considered, enriched our process, and some elements can be recognized in our activity, documented in [8]. In concluding this part, it is honest to say that the methodology used by guifi.net does not have the same goals and formality as the D3.3 methodology, definitely specific for the participatory design of local applications. The guifi.net methods are more generic, oriented to engineering goals and applications related to the infrastructure, support for its governance and platform applications, experimentally-driven, while the netCommons methodology aims at long-term societal goals, to be obtained via capacity building within a community. The guifi.net community is strongly influenced by the engineering and software development culture, which in some sense implies the use of participation models based more on design thinking, lean design principles, agile design and development methodologies, instead of social science based design methodologies.

4. Co-creation and Impact of the Methodology

We can now, before proceeding to proper conclusions of this deliverable, sketch what has been the path in developing the methodology, what has been learned for the future, and what we think its lasting impact on CNs and communities in general can be.

The study and design of the proposed methodology has posed significant conceptual challenges to devise the proper ways of stimulating participation and intervention into highly technical development and deployment activities. Software and ICT developers in general have shown and declared a preference for agile, sequential, quick methods that lead to results in short time, even if starting from these results, they often repeat the cycle. In general this approach clashes with the more complex, structured, reflexive, ans slow approach of social sciences, that favors the process itself as evolution of the community compared to the short-term results achieved.

In addition to the conceptual challenges regarding the further development of our participatory design methodology, there are also practical issues regarding its co-creation by multiple parties. In other words how to foster the interaction and the development of the work when the interacting parties cannot meet frequently enough. For this, the publication of the participatory design methodology booklet will be complemented by an online platform (a wiki or a github/gitlab repository) for encouraging the contribution of comments and further suggestions for improvements, such as Actions that should be added or removed, case studies of real implementations, etc.

The remaining part of this chapter reports the personal perspective of Alexandros Papageorgiou as well as reflections and considerations on the simplification and improvement of the field work based on the methodology as described in D3.3.

4.1. A personal perspective on mediating the co-creation of the methodology

Alexandros Papageorgiou, besides being the co-editor of this deliverable, is the person that both tried to implement the methodology for the AppLea application and performed interviews with other netCommons researchers that experimented with the methodology. He is thus the perfect candidate to offer a personal perspective and comments that may prove useful for people that want to act as mediators and leaders in participatory design processes following this methodology.

"[. . .]
I was asked to conduct interviews with farmers in order to get feedback on the design of the app, and to create the narrative of the whole process. My academic training is mostly about observing and theorizing. Of course, doing anthropology involves 'deep' listening and trying to understand what people say and do in their own terms. But in ethnographic fieldwork you are not supposed to intervene, propose or question, whereas in this process I am not a neutral bystander. Although I am charged with the task to record events as objectively as possible, I am an actor of co-creation. Mediating between developers and farmers, I had to enable the communication between two languages which represent two worlds with different mindsets, at the same time as I was learning about them myself. In doing this, my experience in language translation probably helped. And translation involves intervention. Was I the right person for the job? It is not up to me to answer, although I would affirm that it is helpful, for reasons of explicit impartiality and fairness, to appoint as mediator someone who can establish unhindered communication, but does not 'belong' to either 'language group'. However, this was not precisely my case, since, in the minds of local farmers in West Olympus, I clearly came from the world of experts, from the universities, from the city. The only thing that differentiated me was that I admitted from the start that I am not a 'computer person' and, in this sense, I was 'one of them'. On the other hand, when interviewing netCommons members in the process of brainstorming on the methodology, what differentiated me again was my lack of technical expertise. What we had in common was the commitment to the project and systematic, academic way of thinking.
Therefore, being the person among those participating in the project who combined necessary skills and degree of neutrality at an adequate level, I was assigned the task of mediation. My effort focused on enabling the unconstrained utterance of participants' opinions by using (mostly social) techniques to minimize the degree of censorship that they would impose on themselves, and on faithfully capturing their comments in text. Nevertheless, my personal agency in this process should not be downplayed. On the one hand, there is no way to know if participants expressed their views impulsively. Additionally, my personal understanding surely influenced the discussions, as I did not refrain from expressing my opinion on various aspects of the methodology, or from suggesting solutions for concerns that arose. On the other hand, the reports of the feedback from interviews and of the participatory events, albeit faithful to the wording of participants, were categorized, summarized and interpreted based on what I considered meaningful for the improvement of the methodology and the requirements of the deliverable.
The above is yet another example that mediators are never un-involved agents. Quite the contrary, they always play a crucial role in the 'making of reality' and on its representation. My personal, non-neutral view that follows is founded on this premise and confirms it at the same time. Hopefully, these thoughts on mediation which are informed by the feedback received during the process of co-creation can contribute to the refinement of the methodology. "

4.2. Practice makes better

All interviewees agreed that there needs to be a sort of 'quick start guide' that would allow users to start experimenting with the basic concepts of the methodology relatively quickly and easily. Later they may consider to venture more into complicated concepts or less familiar processes. This progressive scaling of the relationship between requirements and benefits when using the methodology could help a user learn it empirically, 'by doing'. As a user's experience would grow, they could gradually incorporate more and more concepts and elements, specially if they become sort of 'professional' facilitators of participatory design principles. This process could be paralleled to the learning of a language or of music, where one needs to practice on every 'batch' of newly acquired knowledge in order to assimilate it and pass on to the next stage.

What is described above could concern the reception of the methodology by individuals and not necessarily by groups. Nonetheless, in the core of this participatory design 'language' is the idea of designing and monitoring

a project through the co-creation of a collective mind map. As in language or music, it is not enough to 'talk by oneself': One needs to be able to communicate with others to co-create meaning. And, as when learning a language or a musical instrument, users would need to practice in order to gain experience. The inspiration for the metaphor that symbolize the concepts used by the methodology, namely jazz improvisation, requires that musicians master their instruments. However, even experienced players would have to practice improvisation with each other to achieve satisfactory results.

4.3. Learning to trust the process

Trust is based on interpersonal relationships that are built over time and, in the case of community projects, on the proven moral integrity and technical expertise of entrusted individuals. All the more so with the participatory design methodology for local apps, since its complexity and the lack of time of individual members of activist teams do not allow an exhaustive 'digestion' of all its concepts and aspects before field work. For instance, Leonardo Maccari in ninux functions as a broker of and for the methodology, presenting it, explaining it, discussing with his peers about its utility. He is someone with a role in the group, technical knowledge, and experience working with communities. Therefore, it is safe to presume that trust in the mediating actor is crucial in persuading a team to adopt such a toolbox.

In this particular case, trust can be attributed to the individual agency of a member in both the CN core team and the research project developing the methodology. However, as Leonardo stated, were he not part of the netCommons project and of the community network, he would have had a really hard time convincing someone to use the methodology, especially given the required effort to understand it in all its complexity before one can start using it.

So, what happens when there is no trust-enjoying broker? When the methodology will be available online for people to download and use without 'expert intervention'? Before adopting it, a prospective user would need to be persuaded of its efficiency as a product. Apart from the accompanying text describing what the methodology could offer to a CN that aspires to design, develop and install a local app, it would be helpful to show either that it has had successful results in other CNs or that it is advocated by people with experience in the field, or both.

The first condition is still lacking. The second is met. However, the experienced advocates cannot be physically present for every team interested to try the methodology in order to break down its concepts and facilitate its use. Another way would have to be found to make the methodology 'trustworthy' and accessible. One could argue that this way should mainly be supported by a clear presentation and 'friendly' instructions on how to go about it in order to gradually familiarize a team with its use. In other words, by helping a team trust the process through the easy adaptation of the methodology's tools to the needs of their project and through the concurrent enhancement of teamwork.

At this point we need to ask: can this learning process be enabled exclusively by a guide in the form of written instructions, or should it be also facilitated by an individual, whether outsider or insider to the team, who would assume the task of 'hiding' (as Leonardo commented in his interview) or 'untangling' the complexity? Should this methodology (and its presentation, initially in the form of the upcoming booklet) aim to be 'self-sufficient', meaning that a team could learn it and implement it just by following the guidelines? Or should it be mediated by a trainer who would instigate members of a design team to participate and to accommodate the toolbox to their needs? To use once again the analogies of language and music, is a handbook or app to learn without a teacher equally effective as learning with a teacher?

Regardless of the answer to the questions above, the collective process of co-creation should be stressed even more. Even if we opt for facilitation, according all the power of the project's overview to one individual, as well as burdening the same person with the all the work of implementing the methodology is against participatory design principles. The task of monitoring and updating the methodology should not be entrusted to one individual (like a SCRUM Master), instead, as many members as possible should be involved in the process.

Though assuming that one individual will be initially in charge of introducing the methodology to the members

of a team, this condition of power and workload imbalance should not be extended for long. The methodology should explicitly stand against the consolidation of one person in the position of mediator or 'master'. This condition should be temporary, limited to the phase of introduction and familiarization.

Next, the alternation of individuals in the position of coordinator could be suggested. Perhaps this can be achieved through guidelines to include members in the participatory process, taking into account everyone's limitations in time and resources. Surely, given the possible difficulty of a team to meet up often in physical space, after the first introductory sessions an online tool could help to give everyone access to an updated version. Since, for the moment this tool cannot be available, this could be achieved with having a person responsible to aggregate the notes and comments that all members make on ongoing developments, until the next face-to-face meeting.

4.4. Distinguishing two phases of appropriation

We could distinguish two phases of insertion of the methodology into a team's practices: the introduction and familiarization phase and the improvisation phase. For the first phase a mediator could assume the role of training the team on the methodology and of facilitating the transmission of its logic, preferably by focusing on the involvement of members in order to tackle issues that the team faces at present. In the second phase the team members, having attained a basic level of understanding through practice, would be urged to experiment on their own, either collectively or alternating the person coordinating. Ideally, members would learn and evolve together as they go along, according to individual resources and interest.

Summarizing, we could conclude that the methodology's flexibility is not only about its accessibility to end users, or about accommodating the specificities of any given project, but also about facilitating the person or persons who will assume the task of mediating the process of the methodology's customization, whether it is:

In the beginning, training others on the use of the toolbox, focusing on how the proposed guidelines could become meaningful for a specific project as quickly and directly as possible through the addressing of needs on the basis of urgency, while at the same time shedding light to the different ways the methodology could be useful in the future, in other words revealing the logical structure of this new language as well as the potential benefits of learning it.

At a later stage, coordinating the team appropriately, maintaining the Score, interpreting and adjusting the methodology in accordance to the project's needs, goals and capabilities, 'translating' from the Score to real life and vice-versa, rendering categories meaningful. All this in a participatory way, synchronously or asynchronously, in person or online, investing in the appropriation of the methodology by the team, making its advantage evident to achieve the community goals.

In Sec. 4.2, reference was made to the importance of learning through practice. If we take this premise for granted, the main task of a mediator would be to facilitate practicing on and with the methodology and, accordingly, guidelines on how to practice could be given. Therefore, both the mediator in the beginning and the entire team later should be enabled to practice together, preferably with enjoyable elements. Emphasis should be given on including all members of the team, however, as stated before, respecting the individual inability of members to participate.

5. Conclusions

The effort around the participatory design methodology has been about developing a toolbox that can be used as a language to allow its speakers to communicate with each other and with others, in an interpersonal, interdisciplinary and inter-community exchange and creation of knowledge. Besides, any language is a dynamic system whose meanings are constantly performed, negotiated and re-defined. Similarly, the proposed methodology represents a codification of expertise and experience that aims to foster interactions for the creation of new knowledge, sometimes sacrificing the achievement of practical results in the short term.

As some people say, "experience is not transmitted, it is gained". However, people also believe that not all experiences should be lived. In the case of methodologies, authors share their experience with users through guidelines and shortcuts, so that the latter don't have to re-discover a process from the start when there are others who have went down that path before. For that purpose, 'pioneers' have to draw a map of their journey, and aspiring travelers should to be able to read that map.

In the context of the netCommons project, the focus of the work has been how to make this codification as intuitive and efficient as possible, since good intentions would not suffice for its adoption. The process of co-creating the methodology among netCommons partners took place precisely in view of reviewing, not only its relevance with regard to local apps design and development in CNs, but also its intelligibility for users. And while the former was affirmed practically by everyone, the latter has proven very difficult to achieve, as the feedback from ninux and specially from guifi.net in Chapter 3 has shown.

The suggestions included in Chapter 4, which take into consideration the results from the feedback received from netCommons partners, but most of all by communities themselves, seek the simplification of the methodology while preserving its richness.

A very critical point regarding the applicability and relevance of the proposed methodology, are the technical (or non-technical) goals and the target audience. These are dependent on the specific CN and must be carefully accounted for to succeed in the methodology application and adaptation. At the same time, it is necessary that somebody in the local community adopts the societal point of view and declines the methodology in the correct terms given the community goals, which may as limited as managing a commons infrastructure for Internet access or as ambitious as offering alternative services or even alternative Internets. Clearly the definition of the goals is not part of the methodology proposed, though we may well predict that its application may influence in time the goals of the community.

To facilitate such processes, the lasting impact of this research work is being sought for by producing a more agile (compared to the 'academic' methodology) booklet and possibly a collaborative on-line platform that takes into account all the feedback received from the software development teams and the local communities exposed to different parts of the methodology. This process will take place in three phases:

- Main content and presentation: A first draft of the booklet will be made available to the netCommons partners at mid October 2018 for final feedback;
- The draft booklet will be disseminated in PDF format to other CNs and interested groups, through various e-mailing lists, and further refined until early November.
- A first printed version of the booklet will be distributed in IGF 2018 in Paris (November 11-13) and a collaborative online platform (wiki or gitlab) for the further refinement of the methodology beyond the end of the project will be set-up.

Bibliography

[1] P. Antoniadis, I. Apostol, A. Papageorgiou, G. Klissiaris, V. Chryssos, and M. Karaliopoulos, "Multi-Disciplinary Methodology for Applications Design for CNs, including Design Guidelines and Adoption Facilitation (v2)," netCommons Deliverable D3.3, 2018. https://netcommons.eu/?q=content/multi-disciplinary-methodology-applications-design-cns-including-design-guidelines-and-0

[2] P. Antoniadis, I. Apostol, P. Micholia, G. Klissiaris, V. Chryssos, and M. Karaliopoulos, "Multi-Disciplinary Methodology for Applications Design for CNs, including Design Guidelines and Adoption Facilitation (v1)," netCommons Deliverable D3.1, Jan. 2017. http://netcommons.eu/?q=content/multi-disciplinary-methodology-applications-design-cns-including-design-guidelines-and

[3] N. Facchi, F. Freitag, L. Maccari, P. Micholia, and F. Zanini, "Release of All Open Source Software for all Applications (v1)," netCommons Deliverable D3.2, Dec. 2016. http://netcommons.eu/?q=content/release-new-open-source-software-all-applications-v1

[4] L. Maccari, L. Baldesi, N. Facchi, R. Lo Cigno, F. Freitag, L. Navarro, R. Messeguer, M. Karaliopoulos, P. Micholia, and A. Pilichos, "Release of All Open Source Software for all Applications (v2)," netCommons Deliverable D3.4, Jan. 2018. http://netcommons.eu/?q=content/release-new-open-source-software-all-applications-v2

[5] L. Navarro, R. Baig, and F. Freitag, "Report on the Governance Instruments and their Application to CNs (v2)," netCommons Deliverable D1.4, Dec. 2017. https://netcommons.eu/?q=content/report-governance-instruments-and-their-application-cns-v2

[6] J. A. Tacchi, D. Slater, and G. N. Hearn, *Ethnographic Action Research: A User's Handbook*. New Delhi, India: UNESCO, 2003. https://eprints.qut.edu.au/4399/

[7] A. Gavras, A. Karila, S. Fdida, M. May, and M. Potts, "Future internet research and experimentation: The fire initiative," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 3, pp. 89–92, Jul. 2007. http://doi.acm.org/10.1145/1273445.1273460

[8] R. Baig, F. Freitag, and L. Navarro, "Cloudy in guifi.net: Establishing and sustaining a community cloud as open commons," *Future Generation Computer Systems, Elsevier*, vol. 87, pp. 868–887, Oct. 2018. https://doi.org/10.1016/j.future.2017.12.017

A. Original guifi.net wheel procedure

What follows is the original description (freely translated from Catalan) of the "wheel" participatory methodology adopted to deploy a network in giufi.net. It was meant not really as a formal participatory design methodology, but as a best practice for the community to bootstrap networks; however it is important to keep track of it to understand the feedback given by guifi.net community to the participatory methodology of D3.3.

The "wheel": What is the typical process for a deployment?

`[Created: Mon, 26/03/2007 - rroca, updated: 22/10/2008]`

Deploying a free network in an area or territory is an ongoing process, not a one-time event. It is important to know what it is like and in what phase we are at each moment, and which are the following ones otherwise we can get stuck.

By repeating it in several places, we can describe this system as a process wheel, which must turn completely and then start over and repeat itself to infinity. If the wheel doesn't turn completely, that's when we get stuck at some point. The secret is to make this wheel spin as many times as possible.

Since people are the most important part of networking, we want the wheel to spin fast: to complete a circle, i.e. to pick up some results, it should not take more than 6 months if we do not want to run the risk of people becoming discouraged.

The wheel, broadly speaking, is this one:

1. **Meeting / Presentation**

   It is when a presentation meeting is held explaining the project and its nature. It is very important in this phase to describe what is and how does guifi.net work and the character of free and citizen network under the Wireless Commons. When someone is interested in networking, this meeting should be held as soon as possible. The objectives of this meeting will basically be:

   a) To understand the nature of the free web. There's no point in continuing if there's no agreement on this.

   b) Identify the drivers of development in the area and potential sponsors.

   c) Define the scope of the project (coverage to be achieved, to be done)

2. **Drafting the project**

   Once the meeting has taken place, if you decide to go ahead, and know what you want to do, you will describe as quickly as possible a technical project that you think you will need to do (cover a village, make a link . . .). To do this, the points to be connected are marked and the type of equipment to be used is described in order to achieve maximum performance. If it does not already exist before, the project must necessarily include:

   a) A "proxy" or Internet access of any kind for users.

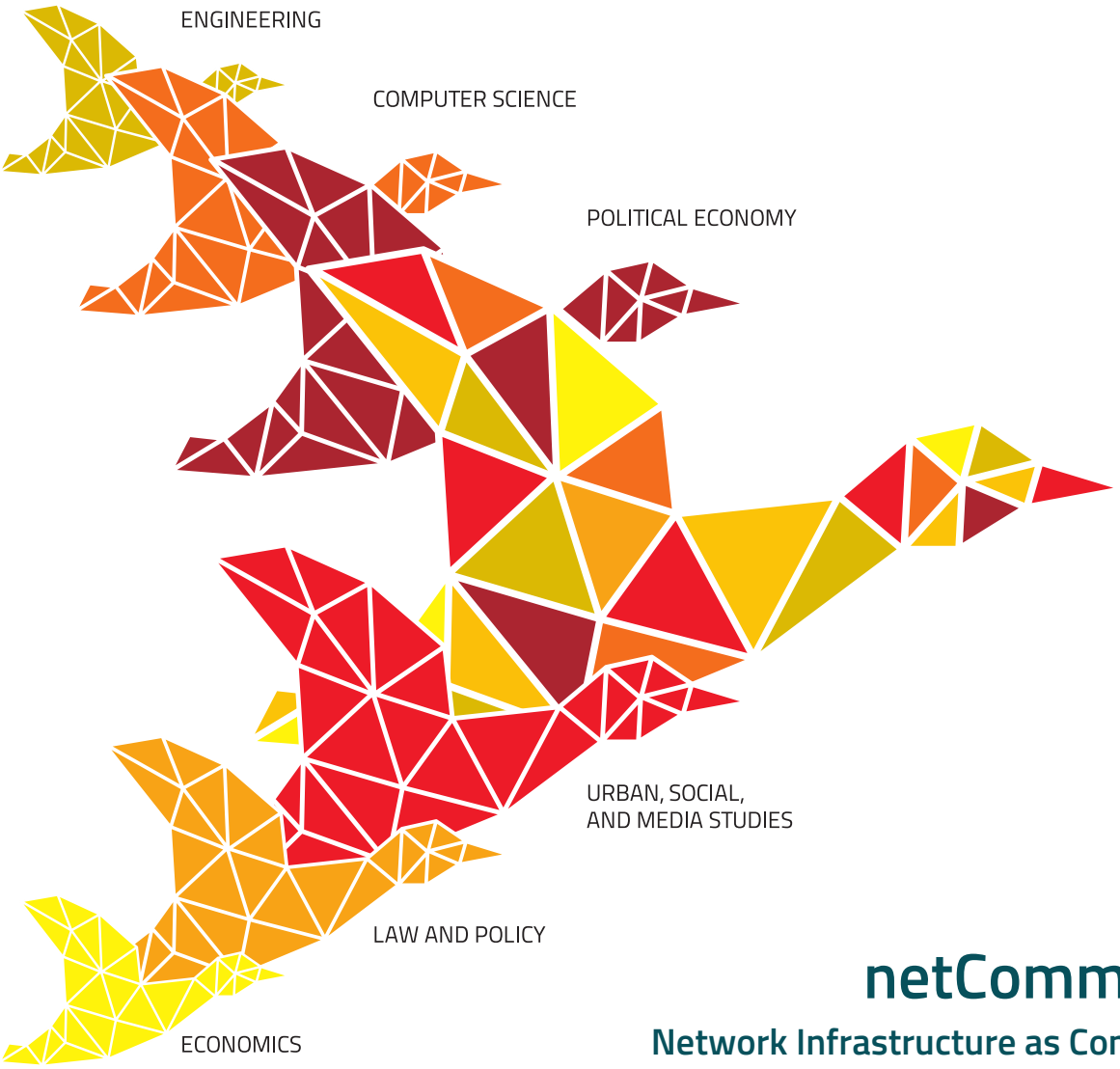
   b) If technically possible, 2 trunk links to other areas where there is already a network, including the equipment to be installed to make the connection to each band. If two cannot be done, only one is foreseen, and if only when none can be done, remote access by ssh is foreseen.

   c) Plan how the network will be monitored, preferably from a local server located near the nodes.

If done without any of these things, you have to be aware that you are losing interest and that there is a structural deficit that can easily lead to failure.

It is also necessary to foresee who and how the work will be done: Ideally, it is possible to include in the assessment the hiring of a company to do the work, and if this is not possible, it should be clear that it will be necessary to resort to voluntary work.

3. **Patronage** Once the project has been written and evaluated, we already know that we have to spend. It is therefore necessary to know how it will be financed (who pays for it). If the budget is exceeded, we go back to cut back on the project, keeping the previous one as a frame of reference if you like, but executing a smaller iteration trying not to leave aside the essential aspects mentioned. It is important not to entertain yourself in this phase for many weeks.
4. **Execution** Once the funding has been resolved, implement it as soon as possible. Obtaining the equipment and installing it may take a few months in some cases.
5. **Start the wheel again** Once something is up and running, we have another meeting, but in this case it serves as a presentation of the one that has already been executed, and many times it can be used to start the wheel again, to extend our area, network in another or renew the equipment that may have become obsolete.



netCommons

Network Infrastructure as Commons

Deployment experiences with a Multi-Disciplinary approach

Deliverable Number D3.6
Version 1.0
October 8, 2018

netCommons.eu