RETHINKING THE ROLE OF USERS IN ICT DESIGN: REFLECTIONS FOR THE INTERNET

Article · January 2009

CITATIONS
0

READS
80

3 authors, including:

Aphra Kerr
National University of Ireland, Maynooth
62 PUBLICATIONS **713** CITATIONS
SEE PROFILE

Stefano De Paoli
Abertay University
55 PUBLICATIONS **159** CITATIONS
SEE PROFILE

RETHINKING THE ROLE OF USERS IN ICT DESIGN: REFLECTIONS FOR THE INTERNET

Aphra Kerr*, Stefano De Paoli *, Cristiano Storni,**
Department of Sociology, National University of Ireland, Maynooth, Republic of Ireland*
Interaction Design Centre, University of Limerick, Republic of Ireland**
Aphra.Kerr@nuim.ie ; Stefano.DePaoli@nuim.ie ; cristiano.storni@ul.ie

Abstract

This paper reports on work from an interdisciplinary project exploring the design of future telecommunications services, networks and applications, particularly focusing on the Internet1. A starting point for this work is our contention that users and technology are co-constructed in the design process and that in some cases it is difficult to distinguish between designers and users since users may play a variety of roles. Some academics argue that the future of the Internet depends on controlling and limiting what users do. Others argue that the future of the Internet will depend on maintaining its openness and enabling users. In our project we seek to critically explore these assertions and empirically examine the role of the user in internet design. This paper outlines some key concepts and presents two case studies to support our approach.

Keywords

User, designer, co-construction, innovation, technology, Internet

1 INTRODUCTION

A variety of literature now argues that the concept of the 'user' is problematic and that for any ICT service or application there are a diversity of users/actors who exercise their agency in different ways. From the digital artisans, to expert users, hackers, players and spectators users can play a range of roles at different stages of the design/innovation process. Users can be disruptive and dangerous but can also be a source of value and innovation. Non users and implicated users may also play a role, often indirectly (Cowan, 1987). This literature moves us beyond simple statements about passive and active users, about diffusion and impacts, and focuses attention on different types of users, different contexts of use and differences in power and agency.

According to Steve Woolgar every process of innovation, design and development can be characterized as a struggle between competing conceptions of the user. Designers have indeed developed a range of methodologies to try and include users in the design process: from user-centered design to participatory design, from the development of flexible technology to forms of co-design and peer to peer production [28]. These methodologies often supplement approaches based on the imagined user, and feedback gathered through formal and informal market research mechanisms. Some have attempted to incorporate this feedback into formalized models of user behaviour, others have attempted to develop more flexible design processes which respond to actual rather than ideal user behaviour. Some have even radically criticized system design arguing for a new style of reflection [8]. Similarly for social sciences a range of models of innovation exist from the more traditional linear approach to 'open innovation' and 'whirlwind innovation' [5]. All attempt to incorporate feedback loops and multiple stages at which a variety of user groups can input into the process. The literature in many disciplines now talks about the designer-user relationship in terms of a community or hybrid collective [5], a network [24][29], a space of flows [6] or an ecosystem. Within this network or community users may play, or may be invited and allowed to play a range of roles, including that of the designer. This network of course is shaped by a larger regulatory, social, economic and political context.

In section two of this paper we outline some theories and concepts coming from sociology of technology, innovation studies, interaction design and media studies which we have found useful in attempting to grapple with the potential roles which users can play in ICT and Internet design. This is followed by two case studies, one historical and one contemporary, which we believe help to illuminate the contingent nature of user roles. We conclude with some comments and ideas for future studies specifically relating to the Internet.

[1] The authors would like to acknowledge funding from the HEA Ireland, under the PRTLI 4 programme.

2 CO-CONSTRUCTION OF DESIGN AND USE

Within social science, Science and Technology Studies (STS) have played an important role in opening up the process of technological design and of use to academic scrutiny. STS have questioned received orthodoxies of the neutral engineer, creative individual or technologist working to develop new products and services which are later diffused into society as stable products and proceed along a rather standardized s-shaped diffusion curve. STS have challenged technologically determinist accounts of social change and demonstrated that technical design is a thoroughly social process influenced not only by the technologist, or design team, but also by the wider context of design and use.

Early work by Bijker, W. [4] under the umbrella of the Social Construction of Technology (SCOT) noted that there were a number of relevant social groups involved in the development of a technology and indeed they may attach different meanings and play different roles in the development process. Later this work started to argue that technologies and users were co-constructed or formed a socio-technical ensemble. Thus it was not solely a process of users shaping the technology or technology shaping users but rather an iterative process of co-shaping. Clearly influenced by Actor-Network Theory (ANT) [1], [5], the need to pay attention to the role of the technology as actor, or 'actant' is for us crucial in any study of users. Also emerging from ANT is the concept of 'delegation' or the fact that roles can be delegated to actors or technologies. Both ANT and SCOT researchers have also been interested in how technologies become embedded in society, stabilize and become locked-in.

Studies of designers and technologists at work have highlighted the role that the imagined user plays or indeed the fact that many designers rely on their own knowledge and expertise, what Akrich [1] calls the I-methodology, to design new goods and services rather than an understanding of user needs or use contexts. For Akrich it is clear that designers develop a script which they inscribe into a technology and this script constrains how that technology may be used and ascribes certain tasks and roles to certain actors. A study by Kerr [16] of the design of a digital game also found that the developers were adopting an I-Methodology and designing for themselves, i.e. young white affluent and technologically literate males. De Paoli at al. [11] observed that software licenses are themselves scripts that act as boundaries for the inclusion and exclusion of users in Free Software projects. These more semiotic approaches to the design of a technology again highlight the fact that certain roles can be delegated to a technology and that technologies can exclude certain uses as well as include certain uses. Thus technologies are socially shaped but may also in turn shape their use and users. Users are active and may behave in ways which conflict with the designers intentions (antiprogram), or act in line with the prescribed behaviour (subscription) or indeed renegotiate it (de-inscription).

Early STS and SCOT work tended to focus on those highly visible social groups who were involved in the production/design process and for some closed off the role of users too early in the development process. More recent work has attempted to rectify this to some degree. Pinch and Trocco [21] highlighted how users can contribute to the refinement of a technology through their study of the synthesizer. Feminist scholars have contributed to our understanding of the barriers that certain designs or scripts can create for certain users, particularly women [10]. Pioneering work by Cockburn and Ormrod [9] followed a technology from production into consumption contexts. Such work started to question the boundaries between design and use and noted that technologies could be redesigned, reshaped and used in very different ways that originally intended by the designers. Histories of technologies like automation systems, the typewriter, the computer and the telephone illustrate how the original designs may have excluded certain user groups but how these technologies were reshaped in a variety of contexts of use. Thus the telephone which was originally envisaged as a type of broadcast mechanism for theatres and solely for business users has taken on many different roles in society. Work by Jamie Fleck [13] on complex computer systems in organizations coined the term 'innofusion' to illustrate that users struggle to make technology work and match local needs. Innovation studies more generally have demonstrated that the efficient incorporation of technologies into the workplace is a process of learning by doing and new practices may emerge from a process of learning by using [2], [22], [19]. In both studies of work and domestic contexts it was noted that if the learning is not captured by the producers or designers then the learning may be localized and the technology may fail. The key message of this work for us is that users and use may be prefigured in the technological design process but nevertheless technologies may change considerably over time, in many cases due to feedback and challenges from users. Indeed innovations maybe be reinvented or substantially changed through use and negotiation.

Using a technology is a much more complex process than merely purchasing the technology and learning how to use it. The learning process has variously been called 'appropriation' [12], domestication [27], [17] or 'social learning' [34]. For Silverstone domestication is about the 'taming' of a new technology and he argued that a new technology goes through various stages before it is incorporated into everyday life including appropriation, objectification, incorporation and conversion. Appropriation is defined as the process of purchasing a product, objectification is the process of placing an object in the consumption context, incorporation is the process of

using the new product and finally conversion is the process of shaping the relationship between people within and outside the context of use. The focus in this approach is that technologies must be adapted to pre-existing social settings, routines and norms and enter into, shape and are shaped by a range of social factors. Users must symbolically, cognitively and practical grapple with a technology in order for it to be incorporated successfully into their lives. For Eglash [12] there are various types of appropriation from reinterpretation to adaptation and re-invention. This mirrors concepts in media studies where Hall [15] notes that users and audiences may accept, renegotiate or reject a media message. Work by Williams et al [34] argues that the process of technological design and use is one of 'social learning' and that certain groups or 'intermediaries' play an important role in translating a technology into everyday life. These intermediaries may be public bodies but may also be private commercial shops, websites or magazines which allow users to experiment and get more information about a new technology. An important part of the material and symbolic process of appropriation is played by advertising and marketing and these actors may develop new user groups or 'represented users' which may or may not actually coincide with real or imagined users [25]. Another group, which may play an important role, are non-users. Sally Wyatt [35] would suggest that non users, whether they are voluntary non-users or involuntary non-users, may still contribute important knowledge to the development of a technology.

In terms of our research it is important to realize that users may struggle to understand what use a new technology might have for them, and indeed having purchased a technology they may still struggle to use and incorporate it into their everyday life. Early work on the diffusion or impact of technologies in society focused on the user and non-users as a problem. The user did not understand or could not use the technology in the ways anticipated. STS would argue that we need to examine the co-construction of the users and the technology; the ideas, biases and constraints which are designed into a technology and the struggles and processes at work in the marketing and use of a technology. Indeed in certain contexts the user may become a designer or what von Hippel calls a 'self-manufacturer' and the designer may also be a user [32]. The complexity of the process of appropriation/domestication/social learning contributes to the uncertainly of outcomes and ultimately to the potential for reinvention and innovation. Capturing and effectively using this feedback is an important part of the innovation process and a far from trivial or straightforward process. Nevertheless, it is now recognized that 'learning by using' and 'lay knowledge' is at least as important a source of innovation as 'expert knowledge'. Instead of focusing on upstream processes it is important to analyse the network of social actors and intermediaries/mediators involved in producing and consuming the technology over time. This process is what Callon and Latour might call 'translation' or Williams et al 'social learning'. This process changes both the technology and the user/contexts of use.

The key concepts emerging from this review which we feel will be important in our studies of the Internet include: the concept of free association and treating human and non-human actors as equally important in our study, the concept of delegation, the concept of power and inequality, the concept of co-construction of technologies and users, the concept of script and inscription, the concept of appropriation and domestication and finally the concept of hybrid communities/networks of innovation. Overall it will be important to identify moments of closure and stabilization while also allowing for dynamism. Lievrouw [18] notes that one can 'Characterize the development and use of new media technologies as a process that involves a constant tension between determination and contingency, that is, between the imposition of order and uncertainty'. She argues that at different moments in the development process the prevailing tendency may change from one of determination to one of contingency.

3 CASE STUDIES

In the following two cases – the protection mechanisms of the system Multics and the Open Hardware project Arduino - we can see how the design of ICTs can regulate/control or enable user behaviour and generate or prefigure different user groups. In the first, historical, case we see how the behaviour of users was prefigured for 'security purposes' and this task was delegated to three simple bits of technology which in turn effectively created a division of labour between two user groups: normal users and administrators. In the second, contemporary, case we explore an open hardware platform system which while placing some 'creative commons restrictions' on users opens up a range of possibilities for user innovation by different user groups.

3.1 Multics case[2]

During the 1950s and the 1960s a typical computer installation – a mainframe computer - consisted of a rather large machine, occupying entire floors of a building. These mainframes were used mainly for research purposes

[2] This case is based on secondary data analysis of research reports (Van Vleck, 1973) and design papers (Saltzer, 1974; McCarthy, 1992) and manuals related to the Multics and to operating systems (Watson, 1970; Silberschatz, 2004).

with a restricted group of programmers/researchers working on them. With the growth of users a data processing method known as Timesharing was introduced[3]. In Timesharing Operating Systems (OS) the CPU time is equally divided among the users, hence creating a centralized network (fig. 1). Using "stupid terminals" users have access to computer resources (memory storage and CPU cycles), hence permitting each user "*to behave as though he were in sole control of a computer*" [20].

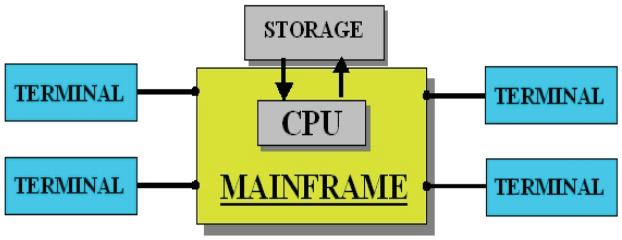

Figure 1 – Timesharing system: centralized mainframe used via a series of remote stupid terminals

The creation of this technology raised a new series of concerns about how to "regulate" dozens or even hundreds of users using the Timesharing OS and the need for security and protection measures. The following passages, taken from a book on Timesharing System design [33], describe some design principles to be followed in the creation of protection for Timesharing:

1. "*Protection of the system from user processes. It is imperative that users not be able to take actions which would stop the system from running or destroy information essential to the system.*

2. *Protection of the users from each other. A user must not be allowed to take an action which would harm the operation of another user's process.*"

These forms of protection were performed using carefully designed software solutions. In an operating system protection refers "*to a mechanism of controlling the access of programs, processes or users to the resources defined by the computer system*" [26]. It is this mechanism that is interesting for us because, as we shall see, it embodies a division of labor between users and administrators of computer systems. We can observe this briefly describing the protection mechanism of a well known Timesharing OS: the Multics system.

The Multics system was developed at MIT (from late '60 till late '80) as part of project MAC (see ref. wikipedia). For several reasons[4] protection and security were considered as fundamental features of the system (see ref.). What is important for us is to observe that the memory space (storage in figure 1) of the Multics system was segmented. Each segment is what we today know as a computer file, where the segment can be considered as the «*cataloging unit of the storage system, and it is also the unit of separate protection*» [23]. Each segment of the system therefore embodied a single protection mechanism. Associated with each segment there was an Access Control List: a list of each user of the system and the related permission to access the file/segments[5].

Once a user accessed the system (via the Username/Password exactly as in today's system) the user then had access to all the segments belonging to her/him[6]. The access control list can be modeled using a matrix (table 1), in which we have the users and the segments. For each file there were different modes of access and in particular: None (access denied), R (read only), W (write only) and E (execute).

In the table 1 example the User_1 can only read the file 1 & 3, while she is allowed to write over the file2. The User_2 can instead also write on File1&3, while she has not rights at all on file 2. So, we can imagine that User_2 gave to User_1 the right to read but not to write the files 1 & 3. User_1 instead gave no right at all to User_2 for using file2.

The figure 2 represents a segment of the Multics system. While the section 1, 3 and 4 of the segment represent different part of the Multics computer file (such as for example the specification of the physical address of the file, section 1.), it is the section 2. which is of particular interest to us. What we have in section 2 are three bits that independently control the access to the segment. These 3 simple bits constitute the core element of the

[3] This was meant to substitute a pre-existing data processing known as Batch (see on this Silberschatz, 2004).
[4] Included the military participation in the development of this system.
[5] Due to limited space, we provide here a simplified account of Multics protection mechanisms.
[6] This is logically true also for the segments belonging to the groups of which the user is member.

control mechanism of Multics: it is because of the setting of these bits that we have an overall division of labor.

	File1	**File2**	**File3**
User_1	Read	Read & Write	Read
User_2	Read & Write	None	Read & Write

Table 1 – Example of Access Control Matrix

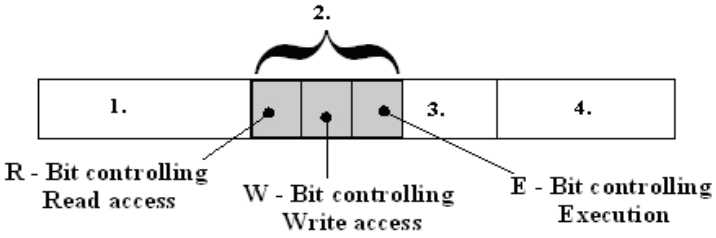

Figure 2 – Control bits of the Multics segments, adapted from Saltzer [23]

In order to understand how these control bits work we can observe the following examples (figure 3). The first one graphically describes the situation of User_2 & File1 of the access control matrix (read and write access to file1), while the second describes the situation of the User_1 & File 1 (only read access). These examples illustrate the protection design principle number **2.** and how the inner working protection mechanism worked. The file protection bits (according to appropriate setting) were able to prevent the users from damaging other user's file: User_1 is not allowed to write over file1. However this form of protection does not characterize the design principle number **1.**.

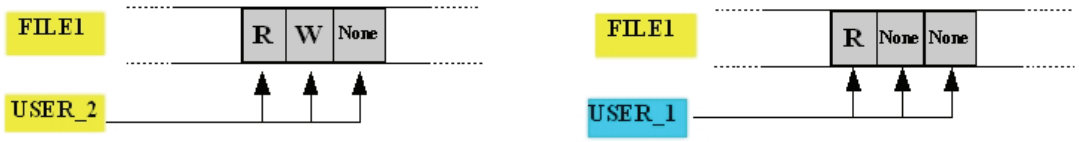

Figure 3 – Examples of Multics segment access control and control bits

What is worth noting is that within the Multics system there was a functional division between the "normal" users and the "administrator". While both were considered users by the system (i.e. both had their username and password), the administrators had particular privileges over the use of the system [31]:

- *"they have access to certain segments which regular users do not.*
- *the system will grant certain requests for them which it will not grant for normal users;*
- *some special abilities, such as the privilege of being able to patch the system, are available to system administrators."*

Some of these privileges, again, were determined by the three control bits. The examples in figure 4 clarify this, where the normal users (in this specific case) do not have the right to write over the OS files or to install a computer application on top of the operating system. Normal users had rights only to their specific segments and only administrators were allowed to install new programs and to modify and patch the OS segments .

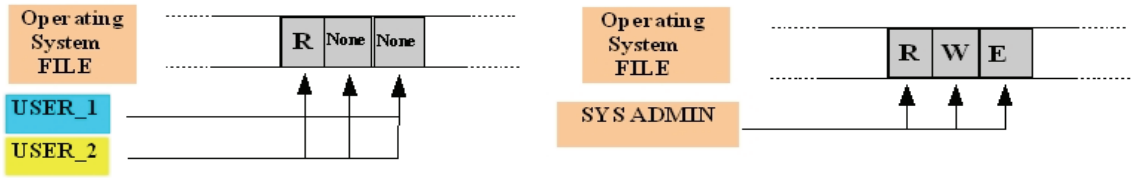

Figure 4 – Control bits and division of labor between users and admins

The control bits clearly limit what the users can do with the OS and constitute a concrete implementation of the protection design principle **1**. While the protection design principles had an acceptable goal (i.e. impede the user to harm other user's file or the whole system), this also leads to a very specific user definition: the user which is not allowed to modify the Operating System files, and therefore is not trusted to do that.

3.2 Arduino case study[7]

The **case of Open Hardware (OH)** and the recent popularity of some projects in this area represent an interesting illustration of new ways of problematizing the relation between designers and users. Here users are not intended just as creators of contents - within well established information infrastructures - but also as active participants in the design process by developing uses and applications that were not originally prefigured.

Wikipedia defines OH as: "*computer and electronic hardware that is designed in the same fashion as free and open source software (FOSS). Open source hardware is part of the open source culture that takes the open source ideas to fields other than software…The term has primarily been used to reflect the free release of information about the hardware design, such as schematics, bill of materials and PCB layout data, often with the use of FOSS to drive the hardware.*". OH opens up these design resources to active communities of hobbyists, invited to do whatever they want with the original design: build new things on top of it, modify it to adapt to new contexts and exigencies, or radically redesign it. Certainly, there are differences with FOSS: the replication of written code is definitely less problematic than that of hardware physical components. At the same time, given that electronic components are becoming cheaper and powerful computers are now widely available to run the software to program OH, the problem with cost is reducing. Among the many examples of OH project, the Arduino board represents a successful instance that merits analysis as it challenges our traditional understanding of the separation between users and designers. According to the official Arduino community website (www.arduino.cc), the Arduino is:*"an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments" ... "It's an open-source physical computing platform based on a simple microcontroller board, and a development environment for writing software for the board"*.

Arduino can be used to develop interactive objects, taking inputs from a variety of switches or sensors, and controlling a variety of lights, motors, and other physical outputs - all electronic material that are widely available, inexpensive and easy to use. Essentially the board is constituted by: a series of ports for inputs that come from whatever sensor is used (motion, light, proximity sensors, etc), a series of output ports connected with whatever actuator is used (motors, lights, computer devices) and a central processor (a micro-controller chip) with a flash memory where written code to process inputs onto outputs can be stored. Code is written with a specific FOSS programming language called "Processing" that allows the user to write instructions for the processor.

The Arduino board was initially designed for teaching purpose within design education. As one of the main developers of the board recalls: *"We already work with Processing a lot. At that time Processing was limited to graphical animations. When dealing with tangible and real-time interaction we had to use another language. One day we asked ourselves: why not to have Processing to generate programs for our hardware too?*"

A specific module for the Processing language was then implemented so that students would not have to learn another language to program hardware in their tangible real-time interactions design works. Initially, the programming language was used with Wiring, a more expensive, bigger and less open board than the Arduino. The idea of developing a more agile and open board and an integrated development environment then emerged. In a couple of months due to a series of collaborations among some volunteers in a design institute Southern Europe, the first Arduino Board was ready. Along with it, a first series of workshops - where the board were actually given to students and interested teachers – were arranged. Interviewees recall a big design workshop in Madrid in 2005 following by one in London and in Copenhagen. In a few months, the Arduino board was popular in design institutes all over Europe and - due to word-of-mouth – within many DIY communities.

While in the Multics system roles are prefigured, they and their interdependencies are inscribed in the system, this OH product – and open hardware in general – prefigure roles only partially. The board is intended to be the central core for the implementation and prototyping of interactive products or environments designed by the users themselves. Essentially, the board does things only if it is connected (i.e. soldered) with input sensors, output actuators and programmed. But the design possibilities for users here are endless, also because Arduino

[7] Documents analysed include reviews of the Arduino Board from on-line magazines (from Wired, Slashdot.com and the official Arduino Web site), an analysis of some of the on-line discussions around the Arduino board on DIY technology sites (Make.com or Ponoko.com). This was supplemented with non structured interviews with the original three developers.

developers have developed very weak forms of control over future uses. The negotiation of the role is not anticipated in the board design but it remains equally open through a series of strategies that we will now list.

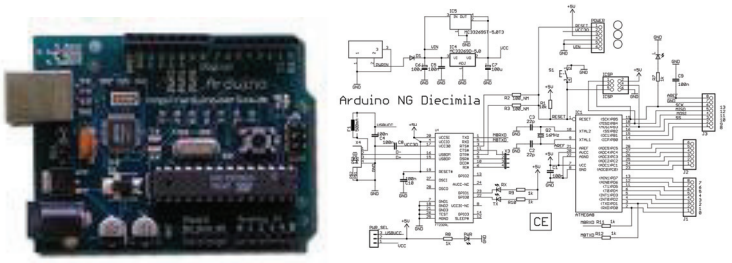

Figure 5 - The Arduino Board and one of the schematics used to illustrate its circuit

On the Arduino web site, all the schematics, design files and software are posted for anyone to access. Anyone can download them and manufacture their own Arduino. In an extensive article on open hardware boards, Wired wrote about the Arduino: "*You can send the plans off to a Chinese factory, mass-produce the circuit boards, and sell them yourself — pocketing the profit without paying the creator a penny in royalties. Arduino developers won't sue you. Actually, they are sort of hoping you'll do it.*" The board schematics and design files are in fact released under the 'Attribution share alike 3.0' Creative Commons' license. Under this license, anyone is allowed to produce copies of the board, to redesign it, or even to sell boards that copy the design. You do not need to pay a license fee to the Arduino team or even ask permission. However, if you republish the reference design, you have to credit the original Arduino group (this is the attribution). And if you tweak or change the board, your new design must use the same or a similar Creative Commons license to ensure that new versions of the Arduino board will be equally without fees and open to future modification and redesign.

The language used to program the microcontroller is borrowed by Processing (Proce55ing.org), an easy FOSS programming language originally intended for graphic design that has been extended - by the Arduino team - with a particular module in order to deal with microcontroller physical boards. The Arduino integrated development environment to write the code and flash it into the board is a piece of software released under the GNU GPL license. This license gives the user the power to change and distribute the software source code, provided that new enhancements are released under the same license terms (i.e. the copyleft clause). This makes user and designer of software as one and the same [5]. The Arduino Web site, where a collection of library of code examples from the user community is growing on a daily basis, is also released under creative commons so that you can freely make use of all the scripts, code and tricks posted by users. This will make the appropriation of the Arduino even easier because inexperienced users can take advantage of solutions and tutorials by peers. The only thing that is registered as trademark is the name Arduino. An interviewee stated: "*The only protection we have in play regard the name of the board that is trademarked. If you want to make a board and called it Gino, it is ok with me and I do not care. But if you make a board and you called it Arduino you cannot. We want to prevent the diffusion of low quality copies. Arduino for us means that the design respects certain quality as the easy of use, the quality of the components and of their assemblage.*"

Within months, hobbyists from all around the world suggested changes and improvements to the programming language, to the software and also to the physical board. Companies also offered to act as distributors. People used Arduino to build their own robots, amateur UAVs (Unmanned Aerial Vehicles), music electronic gadgets and interactive systems. Expert users publish their projects while inexperienced users – even those who do not know anything about electronics – take advantage of the many tutorials available over the web. For those who were already familiar with the Processing programming language the game is even easier. Several quirky companies have emerged. A firm called Botanicals developed an Arduino-powered device that monitors house plants and phones you when they need to be watered: you can buy it online. The web-site Makezine.com inaugurated an Arduino section (An Arduino gift guide) by introducing the board as *the best all-around centerpiece to a modern electronics project* and listing a huge series of step-by-step tutorials on how to build nice gadgets with the Arduino. Also Ponoko.com - one of the biggest DIY technology web sites – sells Arduino based products along with a series of add-ons that extend the possibility of the board.

However, the Arduino board also divides users. Some, for instance, are not happy with the schematics provided by the web site as making a compatible board is 'not easy' (enough) and users must reverse engineer the Arduino. Others ask for more computational power, or to have PIC processor instead of AVR ones to program for. Thus projects like the Freeduino, Saguino and Pinguino have emerged even if the controversies that

generated them are still unsolved in many on-line forums (e.g. http://hardware.slashdot.org /hardware /08/10/24/0343244.shtml).

4 IMPLICATIONS FOR FUTURE WORK AND THE INTERNET

The Internet should be seen as a cluster of technologies and as many have pointed out, the Internet is still in development and has far from stabilized. Gillespie [14] points out that we are currently witnessing the development of an 'Internet culture' of institutions, conventions and practices. The Internet involves an underlying network, protocols and service level information and applications all developed by a range of users. The complexity and openness of the system makes it an ideal context to study the construction of the user. To date discussions of the internet, and in particular of Web 2.0, tend to hyperbolize, acclaim the 'singular' user, and overstate or take for granted the creative agency of production as well as consumption. The reality in many Web 2.0 applications is that the user has relatively little agency to create new forms of content and that much of the value they create lies in the collective metadata collected behind the scenes. The user, while afforded certain limited productive roles, has their agency largely circumvented by the producer/designer/technology of the commercial application. Studies of web 2.0 users signal that many users play relatively inactive roles preferring to lurk, spectate and rate others rather than create or design themselves [30]. More interesting cases exist in relation to crowd sourcing, smart mobs, hackers, cheaters, open source software, and hardware developments and hacktivism [1] that provide us with examples of new user roles, some of which enable the user to contribute to the design/innovation process in a more equitable way [28].

The Multics case study can be seen as a basic example of what we identify as trusted systems today (we need to refer to something here). Trusted systems are quite pervasive since their role is that of creating security in computer networks such as the Internet by enforcing rules known as security policies. The Multics example, in its simplicity, gives some clues about the possible role of users in Internet innovation as related to trusted systems. It is clear that the role played by (certain) users may be that they are prevented from doing things (such as modifying the operating systems) by the technical infrastructure. It is interesting to observe that in order to limit users' actions you do not need large technical solutions, 3 simple bits are enough. They create a separation between administrators and normal users. The first has full control over the system and basically can do whatever he/she wants; the second has prescribed limits. So, we can conclude that while Trusted systems have an important goal – that of creating and enabling protection and security over the network (I.e. ensuring privacy or preventing unwanted computer intrusions), sometimes this goal and its implementation has a side effect: that of preventing users from doing things. Further, it is worth noting that small parts of system infrastructures can have a great impact on users actions.

The Arduino case is an example of multiple user roles where the separation between users and designers is problematic. What is interesting here is the way actors and their tools co-evolve within a socio-technical systems that have very little pre-inscription of actors' roles. This already teaches us – from a methodological point of view – that the role of user in ICT can be understood only as a part of a larger socio-technical system that is constituted by human and non-human elements that mutually shape one another. In this second case, licenses and the Internet plays a key role. Together they afford and invite the formation of creative communities of actors whose role move from that of designer to that of users. But they are nothing without the actual formation of communities where different people participate, learn from each other, sometimes fight and separate. By developing a series of strategies that regulate a collective creation, we can see how the agency of users of the Arduino are nurtured and invited to appropriate, often in unpredictable ways. The original developers seem to have developed an open-ended design process. There is not side effect for them as they delegated the creation of Arduino applications and any change to the board's schematics or the programming environment to anyone who is interested. And in this case (as in many other), the strategy pays back as communities – where use and design mix as well as means and ends – start to feed in. Again, learning from our first example, a change in a single element of the whole assemblage and the mechanism can degenerate into something completely different.

In terms of planning future research on users and the design of the internet it is clear that we need to problematize the concept of the user and in studying a case over time realize that roles and the overall system are dynamic. These findings have larger implications also. As the current framework program develops research related to the future of the internet and in particular trusted and secure networks we have to be careful about how we are changing the internet.

The internet is not stable and given. We have to be attentive that even small technical decisions are not neutral, and that they might have social implications (e.g. create unnecessary social fragmentation, deskilling divisions of labor as well as age, racial or gender discriminations and so on). This is true especially when speaking about computer protection and security something that, by definition, aims at greatly limiting users actions. Our point

is definitely not that of suggesting a best or singular approach to internet innovation but to see that we have different possibilities where some actors or aspects are privileged (mass participation or control) to the detriment of others (access privileges or appropriation). Here decisions, (even small ones), when fixed in the technology can have great consequences for design and innovation.

5 CONCLUSION

The two cases proposed in this paper illustrate two different approaches to design and two different types of what Akrich called scripts. The case of Multics illustrates the establishment of barriers and control systems that limits user's actions for security reasons. The Arduino case shows the melting of these barriers in the process of defining the user as the lead, or at least another innovator. The second case opens up channels of communication and feedback between designers and users, the former closes down the channels. The first case clearly defines roles for user groups while the second leaves these roles more open. Both these case might represent ideal types of users' scripting that can be conceptualized as opposite, where on one side we have a more managerial, top down approach to ICT design, while on the other we have a decentralized, peer to peer approach. The Italian ICTs' theorist Claudio Ciborra [8] would probably have argued that this is an example of "From Control to Drift".

Conceptualizing the relationships between these two models and exploring other models is one of the challenges for our research project. These relationships cannot be taken for granted, risking to fall into subtle forms of reductionism, determinism and essentialism. The next step for our study of the future of Internet innovation will be to better understand and theorize these and other cases while trying to avoid falling into sterile celebrations of one model over another.

These models are not just out there as matters of facts, rather their *matters of factuality* are the result of the same socio-technical processes in which even their relationships are implied. Therefore we need to assess if we are really witnessing the growth of a drift or more open approach or are there forces acting to restrict such approaches. Is this really a new change of paradigm? Will this radically change social relations of production? Will this free us from what the German philosopher Martin Heidegger called the enframing of technology (*das Gestell*)?. It is clear that it would be easy to assign positive values to just one side hence making any effort to describe power relations or even political struggles between various socio-technical collectives a difficult task. For us and for our agenda, further empirical research on the role of users in Internet design is needed.

6 REFERENCES

[1] AKRICH, M.: User Representations: Practices, Methods and Sociology. Managing Technology in Society. Rinter, 1995, pp. 167 — 184.

[2] ARROW, K.: The Economic Implications of Learning by Doing. Review of Economic Studies, 1962, vol 29, pp. 155 — 173.

[3] AUTY, C.: Political Hacktivism: tool of the underdog or scourge of cyberspace?. Aslib proc: new information perspectives, 2004, vol. 56, no. 4, pp. 212 — 221.

[4] BIJKER, W.: On Bicycles, Bakelites, and Bulbs. Cambridge, MIT Press, 1995.

[5] CALLON, M.: The role of hybrid communities and socio-technical arrangements in the participatory design. Journal of the centre for information studies, 2004, vol. 5, no. 3, pp. 3 —10

[6] CASTELLS, M.: The Rise of the Network Society. Blackwell, 2000.

[7] CIBORRA, C.U.: The Labyrinths of Information: Challenging the Wisdom of Systems. Oxford University Press, Oxford, 2002.

[8] CIBORRA, C., U, AND ASSOCIATES: From Control to Drift: The Dynamics of Corporate Information Infrastructures, Oxford University Press, Oxford, 2000.

[9] COCKBURN, C., ORMORD S.: Gender and Technology in the Making. Sage Publications, 1993.

[10] COWAN, R.: The consumption junction: A proposal for research strategies in the sociology of technology. The Social Construction of Technological Systems, MIT Press, 1987, pp. 261 — 278.

[11] DE PAOLI S., TELI M., D'ANDREA V.: Free and Open Source Licenses in Community Life: two empirical cases. FirstMonday, 2008, vol. 13, no. 10.

[12] EGLASH, R. et al.: Appropriating Technology. Vernacular science and social power. University of Minnesota Press, 2004

[13] FLECK, J.: Innofusion or Diffusation? The Nature of Technological Development in Robotics. Edinburgh University, 1988.

[14] GILLESPIE, T.: Engineering a Principle: 'End-to-end' in the Design of the Internet. Social Studies of Science, 2006, vol. 36, no.3, pp. 427— 457.

[15] HALL, S.: Encoding and Decoding in the Television Discourse. Centre for Cultural Studies, University of Birmingham, 1973.

[16] KERR, A.: Representing Users in the Design of Digital Games. Computer Games and Digital Cultures, University of Tampere, 2002.

[17] LIE, M., SORENSEN K.: Making Technology Our Own? Domesticating Technology into Everyday Life, Scandinavian University Press North, 1996.

[18] LIEVROUW, L. A., LIVINGSTONE S.: The Social Shaping and Consequences of ICTs. Handbook of New Media, Sage Publications, 2002, pp. 1 — 15.

[19] LUNDVALL, B.-Å., JOHNSON B.: The Learning Economy. Journal of Industry Studies, vol. 1, no. 2, 1994, pp. 23 — 42.

[20] MCCARTHY, J.: Reminiscences on the History of Time-Sharing. Annals of the History of Computing, 1992, vol. 14, no. 1, pp. 19 — 24.

[21] PINCH, T., TROCCO F., et al.:. Analog Days: the Invention and Impact of the Moog Synthesizer. Harvard University Press, 2002.

[22] ROSENBERG, N.: Inside the Black Box: Technology and Economics. Cambridge University Press, 1982.

[23] SALTZER, J.H.: Protection and the control of information sharing in multics. Communications of the ACM, 1974, vol. 17, no. 7, pp. 388 — 402.

[24] SAXENIAN, A.: The origins and dynamics of production networks in Silicon Valley. Research Policy, 1991, vol. 20, pp. 423 — 437.

[25] SCHOT, J. and de la BRUHEZE A.: The Mediated Design of Products, Consumption, and Consumers in the Twentieth Century'. How Users Matter. The Co-Construction of Users and Technology, MIT Press, 2005, pp. 229 — 245.

[26] SILBERSCHATZ, A.: Operating Systems Concepts with Java, Addison-Wesley, 2004.

[27] SILVERSTONE, R., HIRSCH E. EDS.: Consuming Technologies. Media and information in domestic spaces. Routledge, 1992.

[28] STORNI, C.: Rethinking the role of designer in future ICT, Proceedings of the Second Irish Human Computer Interaction, University College Cork, 2008.

[29] STORPER, M.: The Regional World:.The Guilford Press, 1997

[30] VAN DIJCK, J.: Users like you? Theorizing agency in user-generated content. Media, Culture and Society, 2009, vol. 31, no. 41, pp. 41 — 58.

[31] VAN VLECK, T.: Multics System Administrators' Manual, 1973. Available from, http://www.bitsavers.org/pdf/honeywell/multics/AK50-0_sysAdmin_Feb73.pdf

[32] VON HIPPEL, E.: Democratising Innovation. Mit Press, 2005

[33] WATSON, R.: Timesharing System Design Concepts. McGraw-Hill Company New York, New York, 1973.

[34] WILLIAMS, R., STEWART J., et al.: Social Learning in Technological Innovation. Experimenting with Information and Communication Technologies. Edward Elgar, 2005.

[35] WYATT, S., THOMAS G., et al.: They came, they surfed, they went back to the beach. Technology, Cyberpole, Reality. Oxford University Press, 2002, pp. 23 — 40.

[36] WOOLGAR S.: Rethinking the Dissemination of Science and Technology. CRICT Discussion Paper. No. 44. May. 1994.