UNORT-
KATASTER

AN URBAN EXPERIMENT TOWARDS PARTICIPATORY MEDIA DEVELOPMENT

Georg Trogemann,
Stefan Göllner,
Lasse Scherffig

1. Introduction

While in the past the predominant goal of software engineering was the efficient development of robust, reliable and easy to use systems, the state of affairs seems to change radically at the moment. On the basis of gained experience and acquired adulthood, a significant number of users ask for empowerment. They not only want to be asked afterwards what they like or dislike about an application, but actively participate and get a say at all stages of development, right up to the questions of profit-sharing. Therefore, the recently checked out software design methodologies primarily try to come up with strategies for participation on all levels of the software process. Packaging and shelving of software seems to be a discontinued model, the new software development processes try not to end with the first deployment of a system but stay open to the requirements of an ever-changing context and the continuously upcoming needs of users. To achieve such systems we need open software architectures on the one hand, and the active involvement and the energy of participating people on the other.

The development of the *Unortkataster*, which will be discussed below, is driven by the intention to create an online tool in collaboration with users that facilitates controversial thinking about critical places (*Unorte)* in an urban environment (the area of the City of Cologne). An “Unort” is a marked place or area on a map, added by text-descriptions and other illustrating media by one author or a group of authors. The term “Unort” refers to any urban place or situation, which is criticised for a special lack of quality – based on individual settings. The tool is intended to moderate discussions about these kinds of places and organise them by temporal and spatial aspects. In the following we analyse what potential the approach of ‘technology probes’ delivers for participative software development in the context of big cities. This entails the question, which possible map-based, community applications may be applicable in the interleaved domain of physically localised public space and a globally networked public sphere?

In the following, we also try to illuminate another quite important – normally underestimated – aspect regarding participatory software systems. Once developed and implemented, software acts independently from their creators. It becomes an active player in the social communication game. Algorithmic processes are self-operating entities that actively change the domain in which they perform. This viewpoint does not focus on the cultural memory and/or its well-investigated, modern realisation as digital archive. Rather it is about the underlying algorithms that select, copy, transform, compare, visualise, and permanently reorganise the digital information and its connected communication processes. Thus, due to their invisibility and despite their omnipresence, algorithms are probably the most underrated artefacts of our days.

It seems to be time to ask a basic question: How do software algorithms actively shape society, its knowledge, and communication patterns?

2. Societies and *their* technologies

In his much-cited book *Myth of the Machine*, Lewis Mumford convincingly demonstrated that human culture is not – as often stressed – mainly a result of man's ability to build and command tools, but that tools could only develop so far because of a significant series of inventions in ritual, language, and social organisation.[1] In fact, the first complex machines in history were not mechanical entities but composed of living humans, each assigned to his special office, role, and task, that led to – even under contemporary criteria – tremendous work performances (e.g. the pyramids, palaces, town walls, etc.). Because the components of the machine were working in separated locations, although their interplay implemented an overall functionality, Mumford called them *invisible machines*. The labour machines were merged with the destructive military machine and the controlling administrative machine to the *megamachine* of the early totalitarian states. Within this system we encounter the very first, really powerful machine operator in history. In this image, the whole society is the machine and the god-king has complete control over the machinery. "But only Kings, aided by the discipline of astronomical science and supported by the sanctions of religion, had the capability of assembling and directing the megamachine." So the myth of the machine and the cult of divine kingship rose together, as Mumford puts it.

The invisible machine of the ancient divine kingship was only manageable because of its strict hierarchical organisation. And hierarchy is still the most powerful and important principle we have at our disposal to realise reliable and controllable complex machinery. Thus it is not surprising that we also find this principle within the most complex (and maybe most powerful) machinery of our times, the computer. From an engineering point of view, the computer is only manageable because of its consequent hierarchical organisation. But what has been adopted from past totalitarian states and still is valid on a technological level has completely changed on the social level where the computer becomes effective. The organisation of society, the whole socio-economic and political structure as well as the accompanying power relations, is today based on the (more or less) free decision of participating humans and therefore completely different to the times of divine kingship. But also freedom can be considered as part of a computable function, as it was attempted by Stafford Beer in his failed experiment of the seventies to

1 Mumford 1966

transform the whole Chilean political economy into a self-organising, cybernetic machine.[2] The substitution of hierarchy by self-organisation was an important paradigm shift in the modelling of societies.

According to the sociologist Manuel Castells, we now live in the Network Society, which also only marks a transient state in an ongoing process of a renewed radical change of the power relations. "Cultural battles are the power battles of the Information Age. They are primarily fought in and by the media, but the media are not the power-holders. Power as the capacity to impose behaviour, lies in the networks of information exchange and symbol manipulation, which relate to social actors, institutions, and cultural movements, through icons, spokespersons, and intellectual amplifiers."[3] This means that in the upcoming society, power does not simply disappear but becomes inscribed in the cultural codes through which people and institutions represent their interests and arrive at decisions. The most important cultural code in which power will be inscribed is probably software code.

What we should learn from Mumford, Castells, and others and what is still not sufficiently considered within the software engineering faction, is the insight that major technological progress always goes hand in hand with social dynamics. We can no longer restrict the scope of our software development projects to the characteristics and technicalities of the intended applications, but have to consider the whole social universe in which the application will function. The necessity to consult social aspects and the needs, life-styles and desires of people during software design processes is also supported by another novelty of the software design problem. This has to do with missing experience in our dealing with computational objects. "When designing a classic object such as a chair, there are long traditions embedded in the practices of designing and using chairs that are not easily escaped. When designing a computational thing, however, not only the object but frequently the entire object category will be new to us. The effect is that our understanding of the objects we set about designing is extremely limited. Lacking such a fundamental understanding of the object, studies of possible situations of use, the needs and desires of potential users, and methods from other domains of practice have become tools for navigating an unfamiliar design space."[4] Thus engineers who participate in the design and development of the new technologies will constantly – consciously or unconsciously – have to construct hypotheses and expectations about their users and the social context in which the technology will be used. Involuntarily, software

2 Pias 2004

3 Castells 1998, p. 335

4 Mazé/Redström 2004

engineers thereby become sociologists, or engineer-sociologists as Michael Callon calls them.

All these arguments lead to the conclusion, that, if we want do develop participatory media where people take over substantial parts of the responsibilities for the vividness and progress of the whole application, we have to find answers to some fundamental questions. Some of these questions are: What are the important contemporary rituals, the rules of communication, and the organisational forms of the involved communities and institutions? How does the social fabric of modern communities work and what is the role of software in this network? How can software show its own social conditionality and the wide influence of the subjective perspectives of the participants? How does software become effective in reality and change the communication and knowledge of the social environment in which it is operating (e.g. the Google page rank algorithm)? How can software remain open to the variations of the context in which it is performing and to the changing and proceeding needs of the users?

3. Urban investigations and participatory processes in urban design

In order to find answers to some of these questions the city appears as the condensed laboratory of society where changes become evident and future developments can be anticipated. The city still represents the most successful form of human co-existence and has resisted many announcements of its crisis. Contrary to former assumptions, communication technology can be regarded as a stabilising factor for the physical structure of the city. "The physical synergies between telecommunications and physical networks mean that both tend to concentrate in the same city centres and the same corridors between them."[5]

Condensation and the related diversification is an old motive for the appeal of the city and has a crucial impact on the social structure. Cities do not provide a homogenous environment where people share the same attitudes at the same location. Instead, people of different ethnic and cultural backgrounds coexist and are able to live in close proximity to each other without being confronted directly. The guarantee to maintain differences in spite of the lack of geographic distance seems to have become a particular phenomenon for the modern city society.[6] Moreover, the city allows the individual or community to erode the mechanisms of communicative constraints that are character-

5 Graham/Marvin 1996

6 Siebel 2004

istic for rural societies.[7] The city in this sense should be regarded as space where people *might* start talking to each other but might also refrain from doing so, as Dirk Baecker puts it.[8] The containment of the citizen contradicts efforts to change city culture by communication systems in order to enforce understanding among citizens and create a feeling of togetherness. Normative theories of urbanity therefore polarise communications into a "for or against" and try to enforce a consensus for one of the alternatives.[9] However the city as a physical structure is characterised by the production of conflicts that touch questions which go beyond the individual sphere. Wikipedia describes a community as "a social group of organisms sharing an environment, normally with shared interests." While in online communities shared interests are indeed the dominant connector, in cities the shared physical location becomes the other dominant condition for community building. Civic communities in this aspect are not primarily connected by shared similarities but by *shared problems*.

However, the process of community building in real space evolves much differently from how it happens in an online environment. Online communities have begun to conquer the former, exclusive territories of the city: portals like MySpace or Facebook have started to compete with the street or the mall as the preferred place to "see, be seen, and connect". However, community building in the city may emerge deliberately as much as accidentally. Communities therefore evolve because people share the same housing areas, the same thoroughfares, or the same recreation spaces. In addition the relations of communities are changing over time or may even be time-based: people joining a traffic jam, people sharing the experience of a noisy street, people being involved in the effects of a natural catastrophe etc. Environmental aspects become connecting points as well as infrastructural connexions. Participative processes allow a better understanding of how the character of those communities is changed by the confrontation of physically localised space with a globally networked space.

An early forerunner of participatory city-design, Kevin Lynch, tried to involve people into city planning processes. For that purpose he used simple but effective instruments: He asked citizens for sketched mental maps and did interviews at the same time to better understand the relation between the participants maps and the explanations. By using these very simple and communicative methods he tried to understand how the individual image of an environment is constructed. He aimed at a better understanding for the "collective image", which should finally lead to better city-design methodologies.

7 Luhmann 1997

8 Baecker 1994

9 Schroer 2006

In his book *The Image Of The City* he summarises the requirements for an improved *image*: "The image should preferably be open-ended, adaptable to change, allowing the individual to continue to investigate and organize reality. There should be blank spaces where he can extend the drawing for himself. Finally, it should in some measure be communicable to other individuals."[10]

Lynch's concepts have remained a challenge because the act of designing a city from the very beginning is a rare possibility. Architecture proceeds in a *dictatorial*[11] way because it forces the city into a structure of temporal persistence. The built structure is static and resists being adjusted according

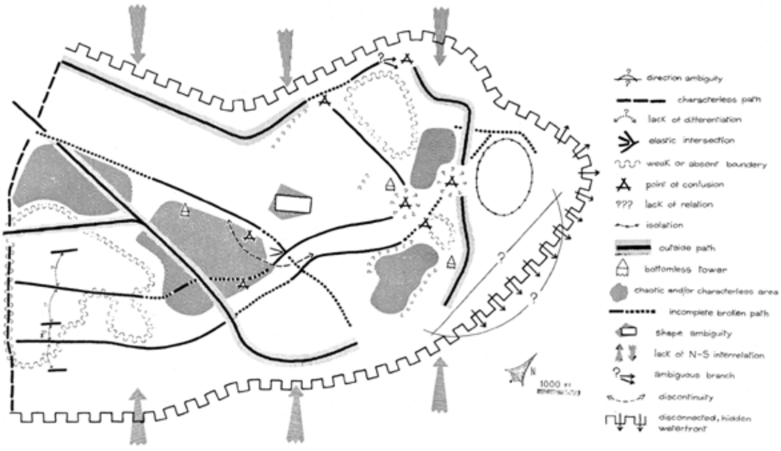


Fig. 1. Kevin Lynch's analysis of »problems of the boston image« resulting in a mental map *(Kevin Lynch (1960): The Image of the City, Cambridge: MIT Press, p. 24, Fig. 8)*

to changes in society. New architectural and city planning concepts do not lead to flexible structures that allow people to modify environments related to changing life conceptions.

But the requirements Lynch verbalised for the rebuilding of cities can be read as a rationale of how networked communication systems are used by citizens today and which change the perspective of how openness of the city-image might be established by *building into software*[12]. Communication networks are working on top of built city structures and allow interaction inde-

10 Lynch 1995, p. 9

11 Mersch 2000

12 Rötzer 1998

pendently of physical barriers. Cell phones give the most obvious example of how this is influencing the behaviour of the citizens: making phone calls without depending on temporal and spatial constraints changes the relations between participants as well as the relations between public and private territories. Shopping by the Internet allows not having to go to real shops and leads to new business models. Positioning systems suggest alternative routes for road users and changes traffic. Openness under these conditions means that the built structure loses its character as a communicative barrier: People's activities and communication are liberated from the conditions of real space by the flexibility of the network. But to get an impression of how these changes influence today's citizens' *image of the city,* a proper analysis of impressions and opinions that are related to it would still be necessary.

The mental maps that Kevin Lynch used to collect input from citizens during his investigations are becoming effective again in the context of city-related, online community tools. Transformed to software, they can become a powerful tool for the involvement of citizens in the process of finding common requirements for future developments.

4. Software in social context

This section will roughly describe our approach to software design that we pursue in the development of the *Unortkataster*. As a process of software development, this approach must partially rely on the history of designing both software systems and their interfaces. From this history we borrow ideas and methods and apply them within the context of participatory media and therewith the perspective of the engineer-sociologist.

Firstly, to build such a participatory medium we need a strong *engineer* approach that not only leads to robust, reliable, and easy to use software architectures, but also allows for an easy adaptation to ever changing social contexts and the accompanying drift of user needs and expectations. Here we have to fight a serious paradox. We know that the context of the application will change, but we do not know how it will change. Nevertheless, our software has to be prepared for it. To encounter this problem we try to transform and apply laws from Software Evolution. Secondly, we need transparent strategies of communication, moderation, documentation, and decision, which allow user *participation* during all stages of development. This entails the problem that the processes organised by the system must be understood in their changing subjective and social dimension and context. Thirdly, this involvement has to start before the first ideas are brought down to paper or any decisions about the application are made. This usually generates another paradox: Potential users (this term is already difficult) in most cases have no clue how novel and innovative participatory media could look like, but that is exactly what we ask them for. Designers and users likewise need *inspiration* that is driven by technology and its possibilities as well as the needs and practices of its situated users.

We may roughly cluster these problems into three categories: 1) *engineering*, 2) *participation*, and 3) *inspiration*. The following brief history of interface design is oriented along these three terms.

4.1 A brief history of interface design

For software to become effective in reality in an immediate sense, an important technological change had to occur: the introduction of the interrupt. The interrupt constitutes a computer's ability to react to new input during program execution. Although it has accompanied computing machinery since the late 1940s, computer science still struggles with the problems it poses for the theory of computation, as the theory of Turing Machines does not generally account for such interventions.[13] For the application of computing machinery, the interrupt established for the first time a real-time coupling between computing machinery and the world. Another change taking place around the same period similarly shaped what interactive computing remains until today: with the introduction of digital computing unlike analog (computing and other) machines, the relation of digital data to its representations became arbitrary, or in terms of semiotics: not longer indexical but symbolic.[14] Since then, the symbols through which users may perceive and act upon an interruptible computer have a two-sided nature: they are sign and signal – open to interpretation but at the same time causally effective in a machine.[15]

Following these changes, *operators* operating computing machinery became *users* wielding tools. With that, a science of users and interfaces emerged that tried to solve the problems raised by the changes, a science that from the 1980s on was called *human-computer interaction* or HCI.

4.1.1 Early HCI: Engineer and test

As the construction of computing machinery was an engineering activity, HCI was strongly influenced by engineering disciplines. But since the interrupt allowed the incorporation of the user into computations, it was

13 Wegner 1997

14 Pias 2000, p. 45

15 Nake 2000

also, from its very beginning, a science of people. It hence co-developed with another young, scientific discipline trying to understand the human mind: cognitive science (and artificial intelligence). And while cognitive science was largely influenced by work on logic and hierarchical problem solving, human use of interfaces was seen as a process of goal directed problem solving, too – a process that could be modelled and calculated in order to build ideal interfaces for the "human processor".[16]

For the engineer, who since the days of divine kingship has been working with the principle of hierarchy, the process of creating software could likewise be organised as a process of hierarchical problem solving, starting from requirements defined in the beginning, and ending with a product that fulfils these requirements. The model that, until today, is associated with this process for its hierarchical, top-down structure was named the "waterfall model"[17]. Since the requirements of an interface can only be fulfilled in conjunction with a user, soon usability tests with users were introduced – a method later named *usability engineering.*

Because of the close connection of HCI, cognitive science and artificial intelligence, it is not surprising that the first influential criticisms of the way user and interface were conceptualised stem from books that were written as contributions to the artificial intelligence discourse. The book *Understanding Computers and Cognition*[18] attacked the assumptions much work in artificial intelligence rested upon by confronting this "rationalistic tradition" with Heideggerian hermeneutics and speech act theory. Thus the authors stressed the problems implied by context dependency and situatedness. The subtitle of the book already suggested the role it should later play, outside the artificial intelligence discourse, when it was adopted by HCI researchers: "*A New Foundation for Design*".

Similarly Lucy Suchman in her book *Plans and Situated Action* also used Heidegger's terminology to show that human actions and goal-directed problem solving rarely coincide.[19] Her theoretical considerations were backed up by a detailed empirical study of the interaction of users and an intelligent machine. This analysis was based on the methods of ethno-methodology. Here, too, what was introduced to the artificial intelligence field as a means of criticism was soon taken up by the HCI community and used as a design method.

16 Card et al. 1983

17 Denning/Dargan 1996, p. 109

18 Winograd/Flores 1986

19 Suchman 1987

4.1.2 Understanding situated use: Participatory design

The critique formulated by Winograd and Flores or Suchman with its focus on understanding the situated human was paralleled by a development in Scandinavia: Here during the 1970s the participatory design tradition was born.

Participatory design emerged from a special historical condition: during the 1970s, 90% of the Scandinavian workforce were organised in unions. Consequently, the unions were granted influence on large parts of what determined their members' working conditions. In Norway, a co-determination agreement was signed allowing worker participation in the development of new workplace technology.[20] Central to the early projects on co-determination was the Marxist notion of ongoing conflict between capital and labour, as well as approaches that tried to understand computer systems as socio-technical systems.[21] Against the observation that "democracy stops at the factory gates"[22] the vision of workplace or industry democracy was developed.[23] The power relations within the factories were to be rebalanced and this had to include incorporating the workforce into the overall design of their workplace.

In a series of projects in Norway, Sweden and Denmark, computer scientists and their prospective users developed new technology for several areas of industry. These projects first of all led to new methods of how to enable users to participate in the design of new technology as it soon became clear that just facilitating conversation between designers and prospective users in order to establish requirements was not enough.[24] Tools were needed that would bridge the gaps between designers and users, allowing each to access the tacit and contextual knowledge of the other. Therefore, methods such as role-playing with mockups – *non-functional,* low-fidelity prototypes made, for instance, of cardboard – became an important design method.[25] Prototypes may here be seen as artefacts that provide a common language for designers and users, enabling and structuring discourse across cultural boundaries.

This "Scandinavian challenge" yielded "a US response"[26] that unlike the Scandinavian tradition was not fueled by Marxist ideas but by market

20 Kuhn/Winograd 1996, p. 290
21 Kuhn 1996, p. 284
22 Spinuzzi 2002
23 Kuhn 1996, p. 284
24 Kuhn/Winograd 1996, p. 291
25 Kuhn 1996; Muller 2002
26 Spinuzzi 2002

demands, where “the result of a good design is a satisfied customer.”[27] The methods developed within participatory design projects here proved to satisfy customers (or management representatives) just like workers’ unions in Scandinavia. Participatory design hence became one major set of tools within the toolbox of HCI. However, as Clay Spinuzzi argues, corporate participatory design reinforced a division of labour between designers and users that contradicted the emancipatory ideas of workplace democracy:[28] One attempt, for instance, to treat user participation from an engineering perspective is the use of “personas”.[29] These are abstract archetypical members of the target group based on observation but defined by the designers. Personas, according to Kari Rönkkö, include the user’s perspective into the development by at the same time excluding the user from major parts of the work.[30]

At this point Peter Denning and Pamela Dargan draw a general line through the HCI field dividing it into two parts: On the one hand they see engineering approaches, including software engineering and the cognitive science inspired work for which design leads to products that fulfil specifications. On the other hand there are “user-centred approaches” focusing on situated action and yielding satisfied customers.[31]

4.1.3 Getting inspiration: Artistic practices

While one may follow Denning and Dargan and argue that the design of computer systems evolved between engineering and user-centred design, other ideas developed, too. HCI has repeatedly been seen as a field in which spaces of action and articulation are defined because design defines “the space of what can be said.”[32] With the interface seen as a social and therewith relational space, the discourse opened for inputs from a new field: artistic practices were taken up, and in opposition to the “engineer-designer” the idea of the “artist-designer” was promoted.[33] At the same time, large parts of the HCI community attempted to change the name of the field into “interaction design”[34].

In a discourse that already treated interfaces as spaces for communication, ideas from art forms that had already concentrated on space and

27 Denning/Dargan 1996, p. 110
28 Spinuzzi 2002
29 Cooper 1999
30 Rönkkö 2005
31 Denning/Dargan 1996, p. 108
32 Winograd/Flores 1986, p. 78
33 Smith/Tabor 1996
34 see e.g. Preece et al. 2002

communication could have a major influence. The filter through which these ideas entered the HCI discourse again was space: within a project that aimed to develop interfaces for urban and rural public space in Italy, Norway and the Netherlands, interventionist strategies were applied to the interface design problem.[35] Artistic maps, disposable photo cameras, albums and postcards were given to the target group (elderly people in this case) and understood as "cultural probes" that were to create visual and textual response. From such subjectively sampled material, prototypes and situations were constructed and again confronted with the audience. Finally, interventions that artistically treated the results of the process were suggested.

The official genealogy of Gaver and Dunner positions this approach in a line with situationism (especially, of course, psychogeography) answering the questions about the situated use of new artefacts with situationist strategies. However, cultural probes implicitly recalled some participatory design projects in which already methods such as end-user photography and storytelling had been applied.[36]

For the HCI field the probes promised to be able to answer a fundamental question that neither engineering nor user-centred design approaches could handle. Namely, what applications should be developed for novel technologies? This is a question that historically proved hard to handle because on the one hand users tend to use new technologies in an unintended way, and on the other hand users in participatory design sessions are likely to suggest applications they already are familiar with. The HCI practice hence depends on methods to "identify needs" of prospective users[37] and to forecast their use of what is offered them. One method to deal with this problem by studying such "possible situations of use"[38] has been developed in marketing research. It consists of a marketing analysis focusing on "lead users" – avant-garde users that are already familiar with new technologies and that therefore may serve as a "need-forecasting laboratory."[39]

With similar goals the cultural probes approach was taken up by other researchers from the HCI field. In order to not only probe the context of the intended users but to directly involve them in the design of new technologies, the concept was extended creating "technology probes." These, as opposed to non-functional mockups, are functional prototypes using new technology and are deployed in real-world situations where they observe their own use.[40] Hence 'technology probes' not only provide a common language enabling par-

35 Gaver/Dunner 1999

36 Muller 2002

37 Preece et al. 2002, p. 201

38 Mazé/Redström 2004

39 von Hippel 1986

40 Hutchinson et al. 2003

ticipation, but also become active players in the language game they are part of. Remarkably, all the projects working with probes resulted in designs that primarily wanted to create possibilities of communication.

4.2 How software makes a difference

Software which operates in social contexts needs continual management during its whole life cycle. Software engineers have simultaneously to track and reconstruct the social context upon which the software is acting. That is the only way to make sure it keeps its validity and does not progressively lose acceptance (see *Laws of Software Evolution* below). This requirement is due to the fact that software changes its own operation domain and causes a growing mismatch between the model of its domain, which is implemented in the software, and the domain itself. From a general perspective we have different agents acting in the same space, such as individuals, communities, institutions, texts, concepts, programs, buildings, and other sorts of artefacts that mutually influence each other. From the Actor-Network-Theory of Bruno Latour (and others) we know that in such networks humans are not the only ones who act. According to its theory, the Actor-Network links together elements (and all the elements of successive links) to a network of mutual influence that performs as a whole. The Actor-Network-Theory, which treats humans and artefacts symmetrically, tries thereby to explain how material-semiotic networks generate certain results and behaviour as an integral entity. One interesting aspect about this theory is that it rejects the naive view of technical artefacts or humans existing prior and independently of their participation in the social and semiotic network of interactions. Accepting this line of argumentation, we also have to regard software as an acting entity within a complex network. In general, software is linked to many other agents, thus it becomes effective on mainly two levels:[41]

- *mental*: programs objectify abstract ideas. They implement concepts that become part of daily life. The users are forced, whether they want or not, to deal with this new reality, interpret it and integrate it in their own subjective way into their view of the world;
- *auto operational*: as self-executing entities, programs have an instantaneous effect on reality. For example, robots, via their effectors, can directly manipulate their own environment. But such self-operating systems in general, if they are working in a social context, change the conditions of their operation since they open and close options for

41 Floyd/Klischewski 1998

alternative choices. They are directly connected to the action space of the participating users and actively alter it.

4.2.1 Models and E-type applications

Models are of central importance in the whole field of computer science. Consequently, during the common software development process different sorts of models are also developed and deployed. To achieve a deeper understanding of the complex connections between software and its application domain, we have to briefly inspect modelling theory. The key properties of the general concept of a model are according to:[42]

Mapping property
Models are always models of something, namely mappings, i.e. representations of natural or artificial originals, which themselves might be models. The concept of a mapping coincides with the concept of assigning model attributes to original attributes. That means mapping is based on the mathematical definition of mappings;

Reduction property
Models in general never capture all attributes of the original they represent, but only those which seem to be relevant for the creator or user of the model;

Pragmatic property
Models are not per se unambiguously assigned to their originals. They fulfil their function of substitution a) for specific – recognising, and/or acting, model-using – subjects, b) within a certain time interval c) under the restriction to certain mental or actual operations.

According to Meir M. Lehman, under the perspective of modelling, a program "is a model of a model within a theory of a model of an abstraction of a portion of the world or of some universe of discourse."[43] This quote points out that in software engineering at least three models are at work: the application model (including the domain and context), the formal model (specification), and finally the program itself.[44] Lehman's early work on software engineering and the growth dynamics of programs started from the conviction that Software Evolution is intrinsic to *large systems*. During his long-standing

42 Stachowiak 1973
43 Lehman 1980
44 See also Floyd/Klischewski 1998.

work it became more and more clear that the concept of *largeness* could not provide the suitable basis for the study of Software Evolution. To solve this problem a new software classification scheme was proposed that was not based on size. Programs were divided into types S, P, and E, whereby the most important class in the present context are the E-type applications.

A program is called an S-type application, if the necessary and sufficient condition for the acceptance and proof of success of the whole development is its correctness in the full mathematical sense, relative to a given formal specification. It is assumed that the specification can be fully predetermined before the development of the software begins. The computation of mathematical functions and/or all sorts of formally definable transformations (e.g. compilers or proof procedures), are typical examples of S-type applications.

The E-type applications are closely related to the concept called *Software Evolution*. In contrast to the S-Type, E-programs are applications which become part of the domain they are modelling. That means that they are not only embedded into the real world but change the reality they are living in through their usage and activity. These programs become effective in reality, they modify what they model. Thus, E-Type programs are the ones we have to deal with in participatory media development.

Somehow halfway between S-type and E-type programs the P-type is located. These programs address problems that are fully specifiable, but the results of the execution has to be checked against the application domain and not its specification model. A typical example might be weather forecasting systems. They are mathematically correctly describable but their performance has to be demonstrated in reality, i.e. in comparison to the actual weather. On the other hand, the execution of the program does not change its domain, e.g. influence the weather, like E-type programs do.

4.2.2 Software Evolution and some of its laws

In its most general meaning the term evolution tries to capture the phenomenon of progressive change of the attributes of a system. From this point of view, not only nature evolves over time but also cities, societies, ideas, theories and also software. Of course, for all these different types of evolution we have to say what progress means in a particular field of evolution. The term evolution in software engineering is used to describe the cyclic and continual structure of maintenance, after the initial development phase. But it is important to mention that the meaning of maintenance in software engineering is incompatible with its common usage. Lehman and Fernández-Ramil point out that over the years it has been recognised that the term has to be carefully used in the context of software development. Normally, the term describes the efforts of people to soften or neutralise the processes of ageing, wear and tear or other forms of deterioration, which are typical and ongoing in every material artefact. For software maintenance it is more about the rectification of assumptions made about the application. We do not have to care about the ageing material but about the ageing of our thoughts, expectations, and models. “What happens with software is that it is changed or adapted to maintain it satisfactorily in changed domains and under new circumstances as judged by stakeholders such as users. Software is evolved to maintain embedded assumptions and its compatibility valid with respect to the world as it is now. Only in this sense, is the use of the term ‘maintenance’ appropriate in the software context.”[45]

The research on Software Evolution came up with a set of behaviour that is now known as Lehman’s laws. Although these laws mainly apply to monolithic, proprietary software systems, some of the most important laws are obviously also valid for participatory design approaches or open source software developments. In the following, we do not present a complete list of all laws of Software Evolution but just a few that seem most appropriate within our participatory *Unortkataster* development:[46]

> *Continuing Change:* An E-type program that is used must be continually adapted or else it becomes progressively less satisfactory;
> *Continuing Growth:* Functional content of a program must be continually increased to maintain user satisfaction over its lifetime;
> *Declining Quality:* E-type programs will be perceived as of declining quality unless rigorously maintained and adapted to a changing operational environment;
> *Feedback System:* E-type programming processes constitute Multi-loop, Multi-level Feedback systems and must be treated as such to be successfully modified or improved.

5. Unortkataster: An urban experiment

The development of the *Unortkataster* Köln is based on cooperation with a group of lead users. The working group *Bild der Stadt* of the initiative *Leitbild 2020* formulated the idea of creating a technological platform for the citizens of Cologne to work on the city, from within the city. The continual talks with members of the community are the basis for an iterative software development process.

45 Lehman/Fernández-Ramil 2006

46 Lehman 1996

The *Unortkataster* application is an outcome of this collaboration and provides a community platform for the citizens of Cologne, based on a shared city-map. Adding state-of-the-art community functionalities to a digital map extends it to a multi-user interface with user administration capabilities and a database structure for storing media content. The map thereby allows people to mark areas or places on the map that are considered to be an *Unort,* and to add opinions and media content to justify the personal decision for the marking. What happens if maps become software interfaces? What does the term *Unort* suggest and how is the added content related to the meaning and function of the administrative *Kataster* maps?

On the *Unortkataster,* marking space becomes equal to starting and locating a discussion about that special location. The setting of a marker creates an *Unort* and is the start of a discussion about the place. Other people may, in succession, add comments or media content to express affirmation or disaffirmation. The *Unortkataster* is intended to facilitate processes, therefore time becomes the second principle of organisation. Since each action on the map is recorded by time, the development of the discussion can be regarded relative to space. It hence constitutes a spatio-temporal interface through which audiovisual communication is made possible. Conversations are connected with places, hence places become blogs. This allows subscribing to the information feed of a place and the tracking of the ongoing discussion. Places have a start and possible end in time. They may appear and vanish. If citizens decide to take part in the discussion about places in Cologne, the time layer of the map is updated and thereby the discussion is documented.

5.1 Maps and software

The originator of the General Semantics discipline, Alfred Korzybski stated, "A map is not the territory it represents, but if correct, it has a similar structure to the territory, which accounts for its usefulness." This points to what we have said about models and modelling. Our perception of reality is not reality itself but just a subjective version, or our own *map* that fulfils a certain function within a certain time interval and under the restriction of certain mental or actual operations.

Maps are figurative representations of relations or objects in the physical or logical world. They represent a certain selection that is related to the observation of an author. Maps contain hierarchies that influence how we see the world and are often based on arbitrary conventions. In the history of mapmaking, the setting of conventions was part of the legitimisation of power. If map users become mapmakers, they are empowered to set the conventions themselves and define their own territories. Software driven mapping systems extend maps into dynamic structures. When maps become digital, they cut the line between the database and the image. The map becomes an interface that allows communication with the database. It thereby structures the circulation of information as an interaction of command, addressing, dating, storing and feedback.

Maps have to be upgraded constantly if they relate to a dynamic content. In the *Unortkataster*, the updating of the map is conceded to the participants. They start building the dynamic map upon the static map delivered by Google. Thus, the map's view evolves from a single-sided perspective to a many-sided perspective on the city. The significance of observation by users acquires another reputation.

5.2 Orte and Unorte

In order to analyse the phrasing of *Unort*, the meaning of *Ort* (place) is to be considered. Martina Löw states that places become visible as the target and result of the placing of social goods or people. Löw notes that places emerge by placements. In order to enable things to be *placed,* to be able to place something, space has to *exist.* On the other hand, the process of placing leads to spacing. The characteristics of places in physical space can be compared to those in an online environment: places appear by placings but are not identical with them.[47] On an online map, a marked place may relate to a real place in the city but, at the same time, generate a different kind of place in the online setting. *Marking an Unort* in this context leads to another paradox: when people mark places on the *Unortkataster-map* this becomes an *Unort* but at the same time it is becoming an *Ort*. The indissoluble contradiction of the two terms is productive as it maintains controversy about the place – being an *Unort* or not – alive. The *Unort* may thus become a catalyst for social controversies about places in the city. It is thereby to be distinguished from the *Non-Place* that Marc Augé describes as a space where neither identities nor relations to history are legible.[48]

Marking on the application happens in a virtual environment. The question of which resulting changes may happen in physical space is, in contrast, related to administrative structures and political processes. But the *Unortkataster* delivers the possibility for a controversy in a setting where a single person is no longer authorised to change an environment, i.e. in public space. Public space, in this sense, is defined as the area that is exposed to the criticism of the public audience. It stands in opposition to private places and exclusive territories that allow an owner or a group of owners to exclude criti-

47 Löw 2001, pp. 198-200

48 Augé 1994

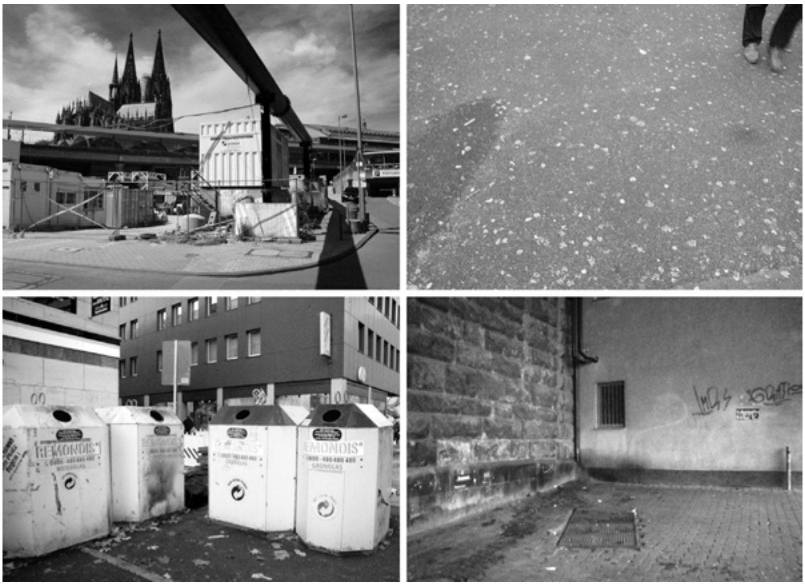


Fig. 2. Unorte in Cologne (Top left to bottom right): Building site close to the Dom, chewing gum, messy glass containers, dirty corner behind the station *(Authors from top left to bottom right: Stefan Göllner, Keiko Takahashi, Jan Hopmann, Oliver Salkic)*

cism to a certain degree. Thus, questions which are related to the relations of public and private properties become of major import for the platform.

Individual observation, marking places and the interlinking of different opinions, are the basic actions of participating in the *Unortkataster*. The tool thereby acts as a platform for organising this communication process. As the object of observation happens to be the public sphere, any favoured change will target the exponents of the public, which are represented firstly by the city administration. The *Unortkataster* thus provides insight into how citizens observe the city. The administration is excluded from this process but will be put in the position to follow the ongoing discussion and might draw conclusions out of this. Conflicts among users, property owners and city administration may arise because of activities on the platform. On that score, the setup of effective moderation mechanisms will be a crucial part of the research of the project.

5.3 Kataster leads to Unortkataster

The administrative *Kataster*-map mainly intends to deliver an up-to-date data basis for the land register and to protect private property rights. Although the *Kataster* is used for a completely different purpose than the *Unortkataster*, there are some crucial similarities between the two different kinds of maps. The *Unortkataster* does not target on archiving of facts and conditions. The content of the application therefore is *soft*: contributions and related media articulate *opinions* of participants that might be put into perspective by any kind of reply or addition of another user. The dynamics of an *Unort* lies in the impossibility of its final definition. The negative rating that distinguishes the *Unort* from the *Ort* refers to the personal opinion of its author and all conclusions drawn from it by related critics and supporters. The value of an *Unort* for the application is measured by the participants' interest in adding their own contributions. Both *Kataster*-maps have in common open-endedness and dependence on the permanent input of citizens. The *Kataster* documents evolution of private property and is thereby related to architecture and the built structure. The *Unortkataster* documents social dynamics that proceed in relation to the public city stage. When input stops, both applications become historic documents that demonstrate how the social sphere and built structure changed over time.

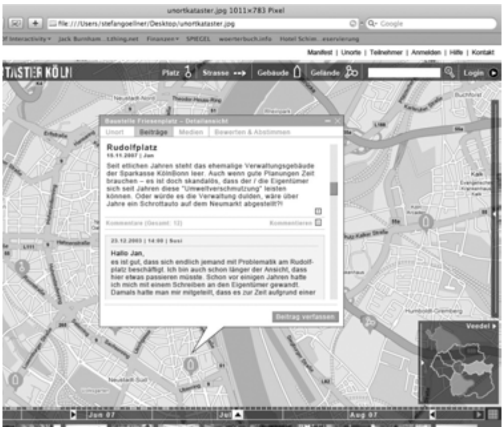
5.4 Unortkataster as a web-probe

Web 2.0 applications have recently shifted the focus away from the user and toward the community. With that shift, methods such as the *persona* that centre around the single user, its needs and its coupling to the interface, have become much less appropriate. Instead here, early probing implicitly developed towards a standard design method. Within this setting (and without explicit reference to the participatory design tradition), this methodology has been called "perpetual beta"[49]: Applications are already opened for use during development, while their development cycle is possibly never declared finished: a praxis that, in the famous example of *flickr*, yielded a successful photo sharing application that had originally been developed (now abandoned) as an online game.

Methodologically, the *Unortkataster* developed through a crossover of practices described above. It started from a lead user approach while the *Unortkataster* itself may be seen as a *Web 2.0 probe*. The concept of the *Unort* was (and continues to be) discussed by probing the city through various media.

49 O'Reilly 2005

As an E-type application it borrows various ideas and methods from the history of software development, while at the same time neither a product nor a satisfied customer are its intended outcome. At best, it will generate discourse, structured by a dynamical (software based) map of the city, acting back upon the city and its real places through the discourse it generates. Similar to the prototypes of the participatory design tradition, it is intended to create a common language for discussing the city seen as a space of shared problems. As a technology probe it is structuring this discourse, at the same time it is open to being shaped by it. Moreover, the database on which the *Unortkataster* is built may also have mobile access. How do *Unorte* change when they can be created and discussed in situ? How does perception of the city change, viewed through a mobile representation of the *Unorte*?



Rudolfplatz
15.11.2007 | Jan

Seit etlichen Jahren steht das ehemalige Verwaltungsgebäude der Sparkasse KölnBonn leer. Auch wenn gute Planungen Zeit brauchen – es ist doch skandalös, dass der / die Eigentümer sich seit Jahren diese "Umweltverschmutzung" leisten können. Oder würde es die Verwaltung dulden, wäre über Jahre ein Schrottauto auf dem Neumarkt abgestellt?!

23.12.2003 | 14:00 | Susi

Hallo Jan,
es ist gut, dass sich endlich jemand mit Problematik am Rudolfplatz beschäftigt. Ich bin auch schon länger der Ansicht, dass hier etwas passieren müsste. Schon vor einigen Jahren hatte ich mich mit einem Schreiben an den Eigentümer gewandt. Damals hatte man mir mitgeteilt, dass es zur Zeit aufgrund einer

Fig. 3. Unortkataster application prototype developed together with lead users of the initiative »Leitbild Köln 2020«

6. Conclusion

Participatory media development is not least about balances of power. In the end the forces and interests of the people driving the development of a certain application have to aid one another instead of blocking each other. An advised motivation, moderation and guidance approach seems to be the key to a successful collaborative development. Direction of the development and at the same time the balancing of power between the participants is subject to the new engineer-sociologist, it is actually his major power. Today we have to accept that there is no longer any technical activity that is not, at the same time, a cultural activity, and vice versa.

According to *Wikipedia,* collaborative governance is a process and a form of governance in which participants (parties, agencies, stakeholders) representing different interests are collectively empowered to make a policy decision or make recommendations to a final decision-maker who will not substantially change consensus recommendations from the group. It is obvious that not all urban problems can be solved with participatory media on the basis of some sort of *communication in the wild*. For example, serious resource allocation conflicts (were does the money, manpower, engagement, etc. of the city go) also need intervention of a powerful government. Without that, the new forms of cooperation would mainly be “a new realm of creative expression and empowerment for the middle classes while the interests of the less powerful will continue to be represented (and distorted) by the devalued traditional welfare structures of (local) government.”[50]

50 Maloutas/Malouta 2004

References

Augé, Marc (1994): Orte und Nicht-Orte der Stadt. Vorüberlegungen zu einer Ethnologie der Einsamkeit, Frankfurt a.M.: Fischer Verlag.

Baecker, Dirk (1994): »Soziale Hilfe als Funktionssystem der Gesellschaft«. Zeitschrift für Soziologie 23, 93-110.

Card, Stuart K./Newell, Allen/Moran, Thomas P. (1983): The Psychology of Human-Computer Interaction, Mahwah: Lawrence Erlbaum Associates, Inc.

Castells, Manuel (1998): The Information Age: Economy, Society and Culture, Vol. 3: End of Millennium, Massachusetts: Blackwell Publishers.

Cooper, Alan (1999): The Inmates Are Running the Asylum: Why High-tech Products Drive Us Crazy and How to Restore the Sanity, Indianapolis: Sams.

Denning, Peter/Dargan, Pamela (1996): »Action-Centered Design«. In: Terry Winograd (Ed.), Bringing Design to Software, Boston: Addison-Wesley, 105-120.

Fawcett-Tang, Roger/Owen, William (2002): Mapping. An illustrated guide to graphic navigational systems, Beverly: Rockport.

Floyd, C./Klischewski, R. (1998): »Modellierung – ein Handgriff zur Wirklichkeit. Zur sozialen Konstruktion und Wirksamkeit von Informatik-Modellen«. In: Klaus Pohl/Andy Schürr/Gottfried Vossen (Eds.), Modellierung '98 – Proceedings des GI-Workshops in Münster, 11.-13. März 1998 (= CEUR Workshop Proceedings 9), CEUR-WS.org, 21-26.

Gaver, William/Dunne, Anthony/Pacenti, Elena (1999): »Design: Cultural Probes«. New Visions of Human-Computer Interaction (= Interactions 6(1)), 21-29.

Graham, Steve/Marvin, Simon (1996): Telecommunications and the City, London/New York: Routlegde.

Hippel, Eric von (1986): »Lead Users. A Source of Novel Product Concepts«. Management Science 32, 791-805.

Hutchinson, Hilary/Mackay, Wendy/Westerlund, Bosse/Bederson, Benjamin B./Druin, Allison/Plaisant, Catherine/Beaudouin-Lafon, Michel/Conversy, Stéphane/Evans, Helen/Hansen, Heiko/Roussel, Nicolas/Eiderbäck, Björn/ Lindquist, Sinna/Sundblad, Yngve (2003): »Technology Probes: Inspiring Design for and with Families«. In: Gilbert Cockton/Panu Kohonen (Eds.), Proc. ACM Conference on Human Factors in Computing Systems, CHI 2003, (= CHI Letters 5(1)), 17-24.

Kuhn, Sarah (1996): »Design for People at Work«. In: Terry Winograd (Ed.), Bringing Design to Software, Boston: Addison-Wesley, 273-289.

Kuhn, Sarah/Winograd, Terry (1996): »Profile: Participatory Design«. In: Terry Winograd (Ed.), Bringing Design to Software, Boston: Addison-Wesley, 290-294.

Löw, Martina (2001): Raumsoziologie, Frankfurt a.M.: Suhrkamp.

Lehman, Meir M. (1980): »Programs, Life Cycles and Laws of Software Evolution«. In: Special Issue on Software Engineering (= Proc. IEEE 68(9)), 1060-1076.

Lehman, Meir M. (1996): »Laws of Software Evolution Revisited«. In: Proceedings of the 5th European Workshop on Software Process Technology (= Lecture Notes in Computer Science 1149), London: Springer, 108-124.

Lehman, Meir M./Fernández-Ramil, Juan C. (2006): »Software Evolution«. In: Nazim H. Madhavji/Juan C. Fernández-Ramil/Dewayne E. Perry (Eds.), Software Evolution and Feedback: Theory and Practice, Chichester: John Wiley & Sons, 7-40.

Luhmann, Niklas (1997): Die Gesellschaft der Gesellschaft, Frankfurt a.M.: Suhrkamp.

Lynch, Kevin (1960): The Image of the City, Cambridge: MIT Press.

Maloutas, Thomas/Pantelidou Malouta, Maro (2004): »The glass menagerie of urban governance and social cohesion: concepts and stakes/concepts as stakes«. International Journal of Urban and Regional Research 28(2), 449-465.

Mersch, Dieter (2000): »Erotik der Stadt. Das Problem von Urbanität zwischen Kommunikation und Ereignis«. In: Helmut Bott/Christoph Hubig/Franz Pesch/ Gerhart Schröder (Eds.), Stadt und Kommunikation im digitalen Zeitalter, Frankfurt a.M.: Campus, 189-209.

Muller, Michael J. (2002): »Participatory design: the third space in HCI«. In: Julie A. Jacko/Andrew Sears (Eds.), The human-computer interaction handbook: fundamentals, evolving technologies and emerging applications, Mahwah: Lawrence Erlbaum Associates, 1051-1068.

Mumford, Lewis (1966): Myth of the Machine, New York: Harcourt, Brace & World.

Nake, Frieder (1989): »Das algorithmische Zeichen«. In: Kurt Bauknecht, Wilfried Brauer, Thomas A. Mück (Eds.), Informatik 2001 – Tagungsband der GI/OCG-Jahrestagung, Wien: Österr. Computer-Gesellschaft, 736-742.

O'Reilly, Tim (2005): »What is Web 2.0 – Design Patterns and Business Models for the Next Generation of Software«, O'Reilly Online. Online available: <http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html> (last access: October 2007).

Pias, Claus (2000): Computer Spiel Welten, Dissertation, Weimar: Bauhaus Universität.

Pias, Claus (2004): »Der Auftrag. Kybernetik und Revolution in Chile«. In: Daniel Gethmann/Markus Stauff (Eds.), Politiken der Medien, Zürich/Berlin: diaphanes, 131-154.

Preece, Jennifer/Rogers, Yvonne/Sharp, Helen (2002): Interaction Design – Beyond Human-Computer Interaction, New York: Wiley.

Rönkkö, Kari (2005): »An empirical study demonstrating how different design constraints, project organization, and contexts limited the utility of personas«. In: Proceedings of the 38th Hawaii International Conference on System Sciences, Hawaii: IEEE.

Rötzer, Florian (1998): »Die Baustelle als Lebenswelt«. In: Franz Pröfener (Ed.), Zeitzeichen Baustelle, Frankfurt a.M./New York: Campus, 224-232.

Schroer, Markus (2006): Räume, Orte, Grenzen. Auf dem Weg zu einer Soziologie des Raums, Frankfurt a.M.: Suhrkamp.

Siebel, Walter (2004): Die europäische Stadt, Frankfurt a.M.: Suhrkamp.

Smith, Gillian C./Tabor, Phillip (1996): »The Role of the Artist-Designer«. In: Terry Winograd (Ed.), Bringing Design to Software, Boston: Addison-Wesley, 37-56.

Spinnuzi, Clay (2002): »A Scandinavian challenge, a US response: methodological assumptions in Scandinavian and US prototyping approaches«. In: Kathy Haramundanis/ Michael Priestley (Eds.), Proceedings of the 20th Annual International Conference on Documentation, Toronto: ACM Press, 208-215.

Stachowiak, Herbert (1973): Allgemeine Modelltheorie, Wien: Springer.

Suchman, Lucy (1987): Plans and Situated Action, Cambridge: Cambridge University Press.

Wegner, Peter (1997): »Why interaction is more powerful than algorithms«. Communications of the ACM 40(5), 80-91.

Winograd, Terry/Flores, Fernando (1986): Understanding Computers and Cognition. A New Foundation for Design, Norwood: Ablex Publishing.

Acknowledgements

The *Unortkataster* is developed within the collaborative European research Project CITIZEN MEDIA, which is co-funded by the European Commission under the "Information Society Technologies" Sixth Framework Programme (Priority IST 2.41.6, Action Line: Networked Audio Visual Systems and Home Platforms).