TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

MASTER'S THESIS

Testing the Usability of OpenStreetMap's *iD* Tool

Institute of Geoinformation and Cartography

Vienna University of Technology

Supervisors:

Prof. Dr. Georg Gartner

Dr. Corné van Elzakker

and

Manuela Schmidt

Submitted by

Jan Behrens

jb3@tzi.de

Place, Date

Signature

Statement of Originality

This thesis titled "Testing the Usability of OpenStreetMap's *iD* Tool" is a presentation of my original research. I certify that this thesis contains no material which has been accepted for the award of any other degree or diploma in my name, in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text.

Acknowledgements

I would like to thank my supervisors Manuela Schmidt (TU Vienna) and Corné van Elzakker (ITC Enschede) for their patience and highly valuable support in accompanying and shaping this thesis in the best possible way.

I also want to thank my programme supervisor Stefan Peters (TU Munich) for his help and advice throughout the study programme, and my fellow students for being great company.

Finally I want to express my gratitude to all test persons who volunteered for this research and without whom this work would not have been possible, as well as my parents and Hans for their support.

Abstract

OpenStreetMap (OSM) is the biggest and best-known source for open geographic information gathered by crowds of volunteer enthusiasts worldwide. With the needs for open-source geodata increasing, OSM has taken steps to increase its visibility towards potential users as well as usability of its tools for new contributors. As the ease of using OSM editors is vital to attracting a greater amount of contributing members, usability evaluations of the tools are needed in order to provide a solid and growing basis of volunteers for the project. The *iD* editor for OSM is one of the most recently developed tools and, being the default online editor for OSM, it is also the most used.

The objective of this study is to investigate the usability of the *iD* editor. To this end usability tests have been conducted with users selected with the aid of an online survey. The participants were given mapping tasks to complete using *iD* and observed with the *thinking aloud* method as well as screen recording and mouse/keyboard logging. Additionally the test persons were interviewed after each test. The data gathered were analysed with regard to key usability criteria such as learnability, efficiency, error tolerance, and subjective user satisfaction. The outcome of this study is the identification of usability issues from which possible improvements of the tool have been derived. The study shows that *iD* is an overall usable tool for novice users, but still shows opportunities for improvement especially in terms of learnability and error handling.

Contents

List of Figures

List of Tables

1. Introduction

1.1. Background

Volunteered geographic information (VGI) has been in the focus of research since Michael F. Goodchild first coined the term in 2007, referring to it as "the widespread engagement of large numbers of private citizens, often with little in the way of formal qualifications, in the creation of geographic information" (Goodchild [2007], p. 2). The phenomenon has emerged from the Web 2.0 paradigm that web-based content production is not only done by experts any more, but also by amateurs—i.e. the ones who were previously only meant to consume content. Simultaneously, the lack of free and open sources of geographic information has led to the development of VGI platforms which will increasingly provide an alternative to (costly) authoritative and commercial geodata.

The relevance of volunteered and collaborative geodata that it has become has been proved by the nearly ubiquitous existence of map mash-ups (user-generated maps put together from available online maps as a base map and individual points of interests on top of them), the increasing use of open geodata by commercial enterprises, and by the great success and growth of collaborative geodata projects such as OpenStreetMap (OSM).

OSM (URL 1) is the best-known and biggest collaborative geodata project today, numbering more than 1.7 million registered contributors (URL 2) (of which, however, only a small fraction actually produce the majority of data). OSM's goal is to collect geographic data for the whole world, and this goal is pursued with the help of a crowd of volunteers. In this context the term 'crowdsourcing' is widely used, referring to any community-based activity aiming at content generation including, for example, Wikipedia (URL 3). Based on the increasing amount of crowdsourcing of geographic data the term "neogeography" has emerged, referring to any techniques and tools that have helped to free map making from the expert-driven domain of geographic information systems (GIS), i.e. to open the field up to any people interested in maps (Turner [2006]).

In OSM, as in Wikipedia, any person (provided they have registered an account) can edit and improve the map with their local knowledge. Raw OSM data is freely accessible, meaning that everyone is free to take it and use it for individual purposes, such as creating a custom map or setting up a location based service. From a scientific point of view, the role of the 'user' in neogeography projects like OSM has been reconceptualized in a VGI sense—data producers and consumers dissolving into a single group of producers/users. Terms like "produser" (Budhathoki [2010], p. 6) or "user-participant" (Eckert [2010], p. 5) underpin that the traditional roles in map making have largely been overthrown.

OSM has become one of the most popular resources for map production, enabling laypeople to put geographic information to their personal use. It is being increasingly adopted by the media and commercial suppliers of location based services, e.g. Nestoria (URL 4) or Flickr (URL 5), who have made the switch to using OSM on their websites, or the navigation system Skobbler (URL 6), which solely relies on OSM data. OSM has further proved to be the tool of choice for citizen mapping as a means of social intervention, such as humanitarian or community mapping (URL 7).

With the popularity of OSM rising constantly, an increasing number of studies investigate the phenomenon with respect to user characteristics, users' motivation to contribute, and data quality (Budhathoki [2010]; Nedović-Budić & Budhathoki [2010]; Lin [2011]; Haklay [2010]). Several studies among them have highlighted the need for better usability of OSM tools in order to lower the barriers to contributing especially for new users. Jones & Weber [2012] have conducted one study of this sort and pointed out that a "chasm" (p. 527) exists in the transition of OSM contribution rates from a small number of pioneer users to mass adoption. This is an indication that (among other things) flaws in the editing tools might prevent the project from gaining a greater amount of dedicated and actively contributing members.

Investigations into the tools that contributors work with in order to make their edits is therefore vital. If these tools are appealing and, most importantly, usable enough, a bigger fraction of potential users will adopt the system for themselves. The motivation for this thesis is to contribute knowledge to this endeavour by conducting a usability test of one of OSM's editing tools, *iD*.

iD is a recently developed online editor for OSM, integrated directly into the main OSM website. It has been designed primarily for beginners and casual contributors, but can sometimes also serve as an alternative for more advanced users. Programmed in JavaScript, it was intended as a replacement for the previous Adobe Flash-based online editor Potlatch 2. It has been launched on the OSM main page in May 2013 (URL 8) and set as the default online editor in August 2013 (URL 9). I have chosen the *iD* tool because it is currently the most used editor for OSM data (measured by changesets and distinct users; URL 10) and because so far—to the best of my knowledge—no scientific usability study exists about it.

1.2. Objective

The objective of this research is to investigate the usability of the *iD* editor for OSM. The outcome of the usability research is the identification of usability issues that are indicative for possible improvements of the tool.

Research questions related to the objective of testing the usability of *iD* are:

1. What is *iD* and what is its purpose?
2. What are the expectations of users using *iD* and how well does *iD* satisfy them?
3. What usability criteria apply to the testing of *iD*?
4. What is the usability of *iD*?

5. Which conclusions for improving *iD* can be drawn from the tests?

1.3. Outline of the thesis

The thesis is built around a case study on the usability of OSM's *iD* tool. Chapter 2 introduces OSM and explains how the editing works, and presents a literature review of usability testing in general and in the context of VGI interface evaluations.

In Chapter 3 the methodology used in the study is described, including a review of the *thinking aloud* method and other observation methods. The chapter also covers the conceptual planning of the test, e.g. the definition of user groups and the overall plan of attack.

Chapter 4 describes the case study itself in detail, from the pre-test survey and selection of participants to the pilot and actual tests. The terminology used in the further analysis of the results is introduced, and an overview of the results is given.

In Chapter 5 the results, i.e. the outcomes of the observations during the test and the interviews, are analysed, focusing on usability issues. Prior to this, the coding system which I have developed to structure the observations is presented. At the end a conclusion is derived from the information presented with regard to different usability criteria.

Finally, Chapter 6 concludes the thesis with a reflection on the research questions and the answers found as well as recommendations for future research.

2. OpenStreetMap and related user research

The motivation for this work derives from the fact that the *iD* editor for OSM is very new and has not been studied with regard to its usability, which may impede first-time users. This chapter further justifies the research by investigating what OSM is, how it works, and to what extent the usability of its tools is the key to the project's success and growth. As such, the concept of usability and the methods and criteria it entails are explained on the basis of publications on that topic. A more detailed overview of previous studies is given where they are directly related to OSM, because this study will have to take into account previous findings and recommendations for the usability testing of VGI interfaces. The chapter concludes with a summary of the scientific basis of the presented case study.

2.1. OpenStreetMap

According to its makers, OSM has been started "because most maps you think of as free actually have legal or technical restrictions on their use, holding back people from using them in creative, productive, or unexpected ways" (URL 11). It was started in 2004 and has since then grown to become the biggest open-source geodata project, counting more than 1.7 million registered users and offering detailed street-level coverage for many areas of the world that often matches up to commercial geodata (Ather [2009]; Kounadi [2009]; Zielstra & Zipf [2010]).

The licensing of the crowdsourced data (URL 12), which permits its free usage, modification, and distribution, lowers the legal barriers to using OSM and thus contributes to the system's openness.

2.1.1. Using OSM

The diversity of uses of OSM is huge. While the main OSM website (URL 1) can be used for browsing the map and finding places, plenty of third-party websites and tools based on OSM data are available to many different use cases, e.g. MapQuest Open (URL 13) for web maps and routing, the "Waymarked Trails" maps (URL 14) for hiking, biking and other leisure activities, or OpenSeaMap (URL 15) for nautical applications. At the same time users and developers are encouraged to take OSM maps or raw data to create new products from them. Among the products derived from OSM maps are so-called 'neogeography mash-ups'—maps created from available online maps and user-generated markers on top (Das & Kraak [2011])—, for which a variety of OSM map styles can be used. The more difficult, but also more powerful

way of putting OSM to use is to export raw data from it and work with it. The raw data can be downloaded either from the website itself or from external suppliers such as Geofabrik (URL 16) or the Overpass API (URL 17). OSM data is usually stored in either XML or binary format and can be loaded into a geographically enabled database. Plenty of open-source tools are available to process that data and/or to generate custom maps or applications from it, e.g. the desktop application Maperitive (URL 18) or the command-line tool Osmosis (URL 19).

Besides using digital products from OSM directly, an increasing number of initiatives is also beginning to lever the crowdsourcing potential of its contributors to their benefits or goals. An example is the Humanitarian OSM Team (URL 20), who have been using an OSM Tasking Manager (URL 21) to coordinate efforts of volunteers to map areas affected by natural disaster or humanitarian crisis.

The richness of OSM uses is, although not thoroughly researched yet, by all means an indicator to the relevance of OSM itself and to the necessity to test the software that enables contributions to it. In order to further investigate editor programs, however, it is first necessary to understand the OSM editing process as a whole.

2.1.2. Editing OSM

OSM data are collected by volunteer enthusiasts using techniques such as Global Positioning System (GPS), tracing from aerial imagery, or simply local knowledge. In the mapping process the gathered information is translated and integrated into the OSM raw data format, which comprises nodes (point-like features), ways (line and area features), and relations (semantic arrangements of arbitrary objects) as data primitives (Ramm, Topf, & Chilton [2011], p. 51ff.). This conversion is facilitated by editor programs that provide an interface between the geographic data collected and their representation within OSM.

The tagging system of OSM, which is based on free-text key/value tags that can be attributed to any geographic data object, allows for arbitrary classifications and attributions. Hence, OSM's tagging system is referred to as a main pillar of the system's openness (Haklay & Weber [2008]). Members of the OSM mapping community have developed and agreed upon a set of commonly used tags to classify and attribute map objects of all sorts, whereas the creative use of tags is not prohibited at all. However, as it is not always clear whether OSM beginners understand the concept easily, in the editors' user interfaces the actual tags are often hidden behind tagging presets (e.g. "hospital" is automatically translated to the tag "amenity=hospital"). While advanced editors like JOSM (URL 22) do not advertise the option to use tagging presets very much, editors targeted mainly at beginners, such as Potlatch 2 or *iD*, more or less directly confront users with them, effectively hiding the true nature of the OSM data format. Whether this affects the (novice and experienced) users' satisfaction while using the editor will be seen in this study of *iD*.

The OSM database is designed to host any variety of geographical features. However, the question arises what actually belongs in the OSM database and what does not, about which there is a loose consensus among the community. Objects in OSM will have to be verifiable "on the ground", meaning that someone who has not surveyed a feature him-/herself should still be able to go there and check that it is correctly

mapped (however there are exceptions to this rule, like country and district boundaries) (URL 23). Besides, any information in OSM should be of public interest and outside of private realms—unlike, for instance, personal addresses or phone numbers.

With the basics of OSM editing explained now, knowledge about the contribution patterns, intrinsic motivation and demography of OSM contributors is required too, before the usability of an OSM editing tool can be investigated.

2.1.3. Understanding OSM

The quality of the OSM dataset has been studied intensively (e.g., Haklay [2010]; Zielstra & Zipf [2010]). Although the dataset is created mostly by non-experts, it has been shown that the completeness and accuracy of OSM data competes with, and even outmatches, that of commercial geodata suppliers in densely populated areas in certain parts of the world—and it is steadily improving (shown for Germany by Neis, Zielstra, & Zipf [2012]). It is important to note here that the major amount of user-generated geodata in OSM is contributed by only a small fraction of its members. Nielsen's '90-9-1 rule' that states that "in most online communities, 90% of the users are lurkers who never contribute, 9% of users contribute a little, and 1% of users account for almost all the action" (Nielsen [2006]), while not applying precisely, helps shape a rough picture of OSM contributions as well. Statistical analyses of OSM distributions have indeed shown that 1 % of users account for approximately 80 % of the data (URL 24, section "Contribution percentage by user"). On the other hand, only 35 % of the users who create an OSM account actually make edits to the database (Jones & Weber [2012]), which is more than the 10 % suggested by Nielsen, but still a number that could be increased for OSM's benefit.

In order to learn about the reasons why people refrain from contributing, researchers try to understand users' motivation to contribute to OSM. According to Nedović-Budić & Budhathoki [2010] unique ethos, self-expression, personal enrichment, self-gratification, recreation, social and group accomplishment, monetary expectation, and altruism are—among others—distinct motivational factors of contributing to OSM. Investigations into the demographical structure of OSM contributors can also yield insight into the reasons for low contribution rates; studies have revealed that the majority of contributors are male (even around 96 %), that more than half are between 20 and 40 years old, and that a considerable number of them are highly educated, employed, working in the private sector, and relatively experienced with geographic information systems (GIS) (Nedović-Budić & Budhathoki [2010]). Attracting more diverse groups of users (by analysing the reasons why there is a lack of diversity and then taking measures) could not only change the overall "climate", but also increase the mere number of OSM contributors.

The questions why people withdraw from contributing and why certain users do not contribute at all, although they have some initial motivation for it, remains to be investigated more extensively. In research done by Schmidt, Klettner, & Steinmann [2013] users who do not contribute any more reveal that the time-consuming aspect of mapping and complexity in editing were some of the reasons why they withdrew from contributing. From this fact it follows that research on the usability of editors

is necessary in order to lower the barriers to contributing especially for novices and passive users.

2.2. Usability of OSM editors

2.2.1. Introduction to usability

Usability is a concept of human-computer interaction science widely referred to in testing, improving and validating software, websites, and web tools. According to Gould & Lewis [1985], a usable system should be "easy to learn (and remember), useful, that is, contain functions people really need in their work, and be easy and pleasant to use" (p. 300). Nielsen [1992] lists five usability criteria commonly referred to: "learnability, efficiency of use once the system has been learned, ability of infrequent users to return to the system without having to learn it all over, frequency and seriousness of user errors, and subjective user satisfaction" (p. 15). An evaluation of *iD* with a focus on these criteria is a main objective of this study.

According to Nielsen [1994b] there are, theoretically, four different ways of evaluating user interfaces: automatically, empirically, formally, and informally. Automatic methods involve a computer program assessing the user interface; empirical methods require testing with real users; and formal methods use exact formulas or models, whereas informal ones are based on "rules of thumb and the general skill and experience of the evaluators" (p. 413). Nielsen states that "empirical methods are the main way of evaluating user interfaces, with user testing probably being the most commonly used method" (p. 413). A typical example of informal usability testing is 'heuristic evaluation', which means that an expert walks through the program and evaluates it based on his/her own set of usability principles, or 'heuristics' Nielsen [1994b].

Empirical test methods include the *thinking aloud* method (see also section 3.1), field observation, and questionnaires (Holzinger [2005]). These are commonly applied methods, probably because their implementation is simple, but effective: As Dicks [2002] puts it, "the results of a typical usability test are 'good enough' to help us uncover problems with a product and to correct enough of those problems so that the testing more than pays for itself" (p. 27). This refers to usability testing that is less formal, i.e. that does not necessarily rely on the testing of hypotheses or the application of formulas and models (Dicks [2002]; Nielsen [1994b]).

2.2.2. Testing VGI interfaces

As usability testing can be applied to a wide range of usable things, it is no surprise that is has been used to test neogeography maps and VGI applications as well. For instance, Das & Kraak [2011] conducted a case study in which they tested a newly designed damage assessment map in comparison with an OSM map that had been developed during crisis mapping efforts, using participant observation and questionnaires. The research revealed that the test persons found the custom map, which made use of improved generalization of map symbols, was more readable.

Other usability studies have been performed on OSM tools, such as Moseme & van Elzakker [2012], who conducted a lab test in order to evaluate the usability of the OSM editing process, uploading GPS data, and extracting data. Using screen recording, eye tracking, and the *thinking aloud* method (see section 3.1 for a description), they showed that test persons had difficulties in finding and using required functions.

A study that deserves particular attention was done by Jones & Weber [2012]. They conducted a usability test of Potlatch 2, an OSM online editor that was integrated in the OSM website in November 2010 (URL 25), i.e. the direct predecessor of *iD*. Using screen recording, eye tracking, and *thinking aloud*, the researchers tested the user interface with ten test persons who had no prior experience with OSM. The analysis, focused on the usability criterion of learnability, showed that there were significant problems with the tool, e.g. misplaced interface elements, inconsistent interaction, insufficient warnings etc. Those shortcomings led to the users being unable to complete tasks or accidentally breaking data.

Furthermore, Jones & Weber [2012] developed a set of usability heuristics based on previous work in the field of usability science and tailored to the specifics of VGI interfaces. These measures are, in addition to Nielsen's criteria, used in the analysis of the results of this study. The case study differs from Jones & Weber's work in a way that it not only works with novices, but also with experienced OSM mappers (not ones experienced with *iD*), and that a broader spectrum of usability criteria is taken into account (not only learnability). Finally, a comparison of the two studies will allow for a comparison of the two editors, demonstrating to what extent *iD* has been designed in line with Jones & Weber's recommendations.

2.3. Conclusion

In this chapter I have given an overview of OSM—what it is, how it works, and how people contribute to it. In this context I have shown the richness of OSM uses, indicated the difficulties that contributors (and those who want to start contributing) need to surmount, and looked into the reasons why they would not. After all, a major interest for the project must be to lower the barriers to contributing by improving the usability of OSM editing tools.

I have shown that usability testing is a well-known and well-established method for evaluating user interfaces including VGI interfaces. The five pillars of usability according to Nielsen are learnability, memorability, efficiency, few errors, and subjective satisfaction; usability heuristics specifically for VGI interfaces have furthermore been developed by Jones & Weber [2012]. All of these criteria are measures against which the usability of *iD* is evaluated in this study.

This thesis is intended to complement previous usability studies of OSM by providing an analysis of a recent OSM editing tool that has not yet been subject of scientific usability research. The methodology that is used to attain this goal is the subject of the next chapter.

3. Methods used for testing the usability of *iD*

In this chapter I give an overview of the methods used in preparing and executing the usability test and in analysing the results. The preparation of the user test involved drafting a detailed test plan including a description of the user groups, phrasing of the online pre-test survey and test materials as well as technical considerations of recording the tests. With respect to the observation methods used, special emphasis is put on the *thinking aloud* method used to elicit information about the test persons' mental processes while working on the tasks. Additionally the techniques used to record oral expressions and interactions with the interface are described. Any methodological considerations are contrasted with alternatives and evaluated with respect to their respective advantages and disadvantages. The chapter concludes with a comparison with the methodology used in the aforementioned usability study by Jones & Weber [2012].

3.1. The *thinking aloud* method

3.1.1. Overview

In order to produce useful qualitative information on iD's usability, observation techniques are used that elicit unbiased data about the participant's interaction with the study object, i.e. thinking aloud, screen recording, and mouse and keyboard logging. Additionally, interviews are used to gain a clearer picture of the participants' subjective interpretations of the tests. Research on usability evaluation techniques has theorised a great variety of methods, all of which show advantages and disadvantages. Among the most popular ones are heuristic evaluation (Nielsen & Molich [1990]; see also section 2.2.1), cognitive walkthrough, a technique for evaluating "first-time use [of interfaces] without formal training" (Rieman, Franzke, & Redmiles [1995], p. 387) done by system designers or experts, questionnaires, and *thinking aloud*.

Usability evaluation methods are divided into two groups, inspection methods and test methods. While inspection methods (such as heuristic evaluation and cognitive walkthrough; see section 2.2.1) are conducted by usability experts alone, test methods (such as questionnaires and *thinking aloud*) require real users with whom the system is tested (Holzinger [2005]). Usability testing methods generally have the advantage of identifying the users' actual needs, which a usability expert might easily overlook, provided he is not a domain expert. On the other hand, *thinking aloud* is a time-consuming effort in terms of both execution and analysis. *Thinking aloud* is usually

employed in earlier phases of software design and therefore not necessarily appropriate for final testing.

Thinking aloud is the chosen method for this research, because it may be expected to produce the most valuable results; since testing with real users has shown to produce better results than heuristic methods (Gould & Lewis [1985]). On the other hand, it requires a long testing phase and a time-consuming analysis as well, but this is acceptable in the framing of this research, and it is also expected to pay out. While the fact that *thinking aloud* is most appropriate in early testing makes it seem less of a choice for testing *iD*, which has been out in the world for months, it can also be argued that *iD*, being a community-driven project, is still undergoing constant review and improvement and will thus benefit from user testing.

An implementation of the *thinking aloud* method can be based on a rather extensive amount of literature, which is reviewed in the following sections.

3.1.2. Theoretical foundations

Thinking aloud is a method that has evolved from cognitive science and has been widely adopted since by usability practitioners (e.g., Nielsen [1992]; Nielsen, Clemmensen, & Yssing [2002]). Essentially it describes a process in which the test person is asked to say out loud whatever he or she is thinking while working on a given task, whereas the test administrator adopts a passive listener's role. While the *thinking aloud* method has the disadvantage that it might perhaps increase the time that a participant needs for completing a task, it excels in eliciting a lot of useful data about what the participant is doing and thinking, i.e. about his/her interaction with a program's interface (Nielsen [1992]).

A theoretical foundation for the application of the *thinking aloud* method based on findings in cognitive science research has been established by Ericsson & Simon [1984]. They developed a classification of verbalizations with respect to the amount of cognitive processing that happens between the heeding of the information and its articulation through words. Level 1 articulations are those that require no processing at all, because the verbalization equals its cognitive representation, whereas Level 2 verbalizations (encoding of heeded information into language) and Level 3 verbalizations (transformation of the information beyond encoding, such as filtering, interpretation, or judgement) do require some amount of processing. While Level 1 and Level 2 verbalizations can be considered "hard", unbiased data, Ericsson & Simon argue that Level 3 verbalizations are not usable for interpretation due to their subjectivity.

Ericsson & Simon advise any test practitioner to be as invisible as possible during the *thinking aloud*, to the extent of creating the illusion that the test person is "alone in the room" (Boren & Ramey [2000], p. 263). The only type of interaction they allow is to issue a reminder, "Keep talking", whenever the test person falls silent.

One problem with applying the model to usability testing is that Ericsson & Simon were not usability practitioners themselves and thus fall short of dealing with particular situations that appear uniquely in usability testing—e.g. what to do if the system crashes and intervention of the practitioner is required. Adopting the model in usability testing makes it furthermore necessary to account for the idea that while in cognitive research the test person is the object of study, in usability testing it is the system.

According to Boren & Ramey [2000], gap exists between Ericsson & Simon's theory and the common practice of usability testing purportedly based on it; this will often find expression in usability tests being executed without any justified theoretical basis at all. While Boren & Ramey admit there are limitations to the model when it is applied to usability testing, they urge test administrators to either stick with the model and extend it in accordance with its original "spirit", or to employ another model based on speech communication theory. For such a speech communication-based framework, for which they lay the theoretical foundation, Boren & Ramey suggest the following guidelines:

- Accept that a pure monologue by the test person is not possible, as the practitioner always has to be present (so as to be able to issue reminders) and necessarily is part of the conversation, even when he remains silent;
- "set the stage" for the communication, i.e. make it clear that the test person is the domain expert, whereas the practitioner is the learner and listener; also make it clear that not the person, but the system is the object of study;
- instead of ignoring the participant's utterances that implicitly or explicitly ask for a response, use non-obtrusive and non-directive "acknowledgement tokens" such as *"mm-hmm"* or *"uh-huh"* as "back channels";
- also use those tokens as a more natural way of reminding the participant to think aloud whenever he/she falls silent;
- intervene in unexpected situations as necessary, e.g. fix the system after a breakdown or remind the participant of tasks not completed, but always do that in a way that reassures the defined roles;
- possibly ask for clarifications by repeating a participant's words with an interrogative intonation, or even give directive probes when a key part of a task is being overlooked or when the participant navigates around it.

With Boren & Ramey's proposal, however, there is (admittedly) the danger that interventions by the practitioner, as forbidden by Ericsson & Simon's theory, can influence the participant's further performance. Krahmer & Ummelen [2004] conducted a study in order to compare the two approaches with respect to the usability issues found, as well as the participant's performance on the tasks. While the detected issues were more or less the same with both methods, the number of tasks successfully completed by the participants was significantly higher in the group that was tested with Boren & Ramey's speech communication method. Test persons of that group also got lost in the interface less often. This is evidence that increased intervention indeed affects participants' performance, and in any more liberal approach towards Ericsson & Simon's theory this needs to be remembered. Considering though that the outcome of found issues is not affected considerably (as the mentioned research has shown), Boren & Ramey's approach seems legitimate.

3.1.3. The method applied

While most of the competing theorizations are justified, a decision for a model to implement in this research needed to be made. I favour the Boren & Ramey protocol for better taking into account the nature of human communication (according to Bakhtin [1986], Ericsson & Simon's imagination of the test practitioner being absent is called a "fiction" (as cited in Boren & Ramey [2000], p. 267)).

Specifically, I have developed the following set of rules for intervening in the tests of this case study:

- **Reminding the participant to think aloud.** Instead of the "Keep talking" proposed by Ericsson & Simon, more natural tokens are used (with an interrogative inflection), as suggested by Boren & Ramey (for instance, *Hmm? Thoughts?*).

- **Providing back-channels.** Each time the participant explicitly or implicitly asks for feedback, an acknowledgement token like, e.g., *"mm-hmm"* is returned, in order to keep up the participant's flow of work and a natural type of communication.

- **Putting the participant back on track when he or she is stuck.** This sort of intervention is more critical, as it poses a greater danger of interrupting the test person's flow of work. It is allowed only if his/her inability to work it out on his/her own is preventing them from continuing with crucial tasks that lie further down the road. For example, as soon as he/she is unable to add attributes to a map object, this obstacle needs to be overcome, because adding tags is key to several of the tasks. I must intervene only after the test person has been stuck for a minute or more. The guidance given must not be directed at a solution, but should rather be a suggestion to reread the task, a hint at the help sources provided, or—if necessary—clarification of the task. Lastly, the participant may be encouraged to remind him-/herself of certain steps that have been covered in the walkthrough (task 1; see section 3.4.3). When problems arise during any task that is not crucial enough to justify an intervention, I shall simply allow the participant to proceed.

- **Asking the participant to rework a task when the functionality of interest has been sidestepped.** If this happens, I must intervene only if the avoided functionality is crucial to the task. Otherwise the flow of work is to be given priority. (However, this will likely not happen often, as in OSM editing there is usually more than one correct way of achieving a particular goal.)

- **Acknowledging the participant after completion of a task.** If any of the top-level tasks has been completed by the participant, a lightly affirmative comment (like *"OK, this was Task 1"*) is allowed in order to give the participant some orientation that he/she is not being off-track and to induce a little motivation to go on with the remaining tasks. While this approach may be questionable for its talkativeness, it is expected to contribute to the participants' flow of work.

- **Stepping in when the system behaves unexpectedly, crashes, or hangs.** Such an intervention is required if otherwise the test cannot be continued. A comment may also be in order if the software shows a behaviour that was clearly not intended by its programmers (a "bug"). This sort of intervention shall be handled in a way of reinforcing the roles of the 'expert' and the 'learner', as suggested by Boren & Ramey (2000).

I did not answer participants' questions in other cases than the aforementioned, engage in conversation, or proactively probe participants for additional information, because the participant's concentration on the task was to be maintained and the risk of influencing his or her behaviour to be minimized.

The test persons' utterances during the test were recorded with a headset microphone, which was attached to the computer also running the *iD* application. To record the audio from the microphone input during the tests the command line tool "FFmpeg" was used (see section 4.3.3 for technical specifications). After the tests the *thinking aloud* data was analysed and relevant portions of them were tagged with certain codes and finally transcribed by the word, including a description of what the participant did. The coding system is presented in more detail in section 5.1.

3.2. Additional user research techniques

3.2.1. Screen recording

Through screen recording the computer's screen is captured and saved to a video file. With this method the interface the user is confronted with is recorded as well as cursor movements. The method is popular with usability practitioners as it allows "unobtrusive collecting of a rich record of actual computer work activity" (Tang et al. [2006], p. 479). In combination with *thinking aloud* it facilitates a very detailed analysis of the user's interactions, enabling the detection of misled behaviour resulting from possible usability issues of the system.

While screen recording makes the collection of lots of valuable information possible, it is a method that raises privacy concerns. Whenever this method is applied, participants must be informed about the practice in advance.

In the tests of this study the screen was recorded with the "FFmpeg" software, which was simultaneously used to record the audio stream.

3.2.2. Mouse and keyboard logging

Screen recording is a great method for capturing everything that happens on the screen—but it is not fully capable of taking a record of mouse clicks or keyboard strokes: Although one can spot occurrences of mouse clicks and keyboard strokes whenever there is a visible feedback to them, one cannot identify them when there is no visual feedback, e.g. a click on an area of the screen that is not a button or does not react otherwise. But luckily there are tools that can record all mouse and keyboard interactions. I used the "key-mon" tool, which displays a small panel on the screen showing the buttons

pressed on the mouse or keyboard in real time. As the panel is displayed on the screen, it is embedded within the screen recording videos.

3.2.3. Video recording

Originally I had intended to produce a video recording of the test person's face during each test, as has been done in numerous usability tests before (e.g., Arhippainen & Tähti [2003]). A web camera built into the computer just above the screen was to be used for this purpose. It is a useful method, because a test person's facial expressions and gestures are a useful supplement to the *thinking aloud* method. However, it has turned out in the pilot tests and the first three actual tests that the overall load on the computer created by the three simultaneous recordings was quite high, resulting in peaks of CPU usage and thus in loss of audio and video data (see section 4.3.4 for more information on technical issues). Therefore I decided to waive the video recording of participants in order to reduce load on the computer. I chose to drop the video rather than anything else because I considered that data less crucial for the analysis than the *thinking aloud* and the screen recording data. Whether the waiving of the video recordings has actually helped to improve the recording quality is difficult to say, because the frame dropping seemed to appear rather randomly.

3.3. User groups

The test was conducted with two user groups:

- users who have not edited OSM before except maybe one or two times (the 'novice' group),
- and users who have some more experience with OSM, but have not, or rarely, used the iD editor before (the 'mapper' group).

The distinction between these two user groups is important because those who have edited OSM before are certainly more familiar with the basic concepts of OSM editing—such as the tagging system, the ways how map objects should be drawn, or how aerial imagery can be interpreted. As users of the novice group usually lack this knowledge, they were expected to perform differently on the editing tasks and thus to reveal different usability issues with the editor. However, it must be remembered that with respect to the *iD* editor all test persons were (almost) novice users.

The conclusions that can be drawn for any improvement of *iD* will certainly depend on the developers' and the OSM community's opinion which user group the program is to be targeted at (who, as indicated in chapter 1, are mainly novices to OSM). The usefulness of the test results will, for that matter, be increased by the possibility to differentiate between the groups of users that have produced them.

For each of the two user groups I attempted to find between 6 and 10 test persons. I have chosen this number in consideration of previous research on what the recommended number of participants in a usability test is. There is no clear consensus about the adequate number (which will also depend on the methodology applied and whether

the research is qualitative or quantitative). The aim of this qualitative research project is not to derive statistically valid outcomes, but to identify usability problems to arrive at recommendations for the improvement of *iD*. Virzi [1992] found that in his experiments "approximately 80% of the usability problems identified would have been found after only five subjects" (p. 467), whereas any additional test person would not have increased the discovery rate significantly. On the other hand, Law & Hvannberg [2004] suggest that in a *thinking aloud* test 11 participants are needed in order to discover 80 % of the usability issues, and according to Hwang & Salvendy [2010], 9 participants are needed to reach the 80 % mark in a *thinking aloud* test. Thus, trying to attract up to 10 participants is well in line with the recommendations given in literature.

3.4. Overview of the test plan

3.4.1. Test setting

The tests were announced to take place in the region of Hamburg, Germany, in April 2014. As a location I offered my home and, alternatively, the possibility to do it elsewhere, at a place of the participant's choice. This was possible thanks to the fact that I could easily move the equipment, which was merely a notebook computer and a headset, and that nothing else was required except a quiet atmosphere and an internet connection. This approach had the benefits of putting the participants in a situation they were most likely comfortable with and of having less organizational constraints than the planning of lab visits would have imposed.

3.4.2. Online survey

The test persons were selected with the aid of an online survey (see section 4.1 for a more detailed description of the selection process, including the survey results, and appendix A for the complete survey). In the survey the candidates were asked questions used to assign them to the novice and mapper groups, as well as to collect some background information about their contributing characteristics and their experience with related technologies such as IT, GIS, and social media in order to better interpret the test results afterwards. All test persons were required to complete the survey prior to the actual test.

The online survey was announced on the OSM community's regional mailing list for Hamburg "OSM-HH" (URL 26), on the German-language OSM mailing list "Talk-de" (URL 27), and on the general discussion list "OSM-talk" (URL 28) on 22 March 2014 in order to attract potential test persons.

3.4.3. Description of the tasks

The actual test has been designed to simulate a real use case as authentically as possible. Solving the tasks required different activities typical of a beginner level OSM mapping session, with a more advanced extra task for the mapper group.

- The first task is intended to make the test person familiar with the OSM website and the *iD* tool. The test person is asked to navigate to a target map area by using the search function. Afterwards he/she is supposed to go through the *iD* walkthrough, which is a built-in introduction to the basic functionality of the editor. The reason why this is done is that the walkthrough is an integral part of the editor and therefore deserves to be tested with respect to its usability as well. If it proves to be a useful tutorial for novice users, it certainly contributes to the ease of learning to use *iD*. Besides, it is a rather easy task and therefore very suitable as a starter (also for test persons to practise thinking aloud).

- In the second task the test person is asked to add features to the map that are annotated on a printed "field paper". Field papers (URL 29) are automatically generated printouts of a chosen map area in OSM that can be taken out into the field, filled with notes, and afterwards scanned and loaded into the editor program. Though the scanning is optional, they are anyway a useful (and popular) way of taking notes during a mapping survey. The field paper provided for this task contains annotations of typical map objects, such as a restaurant, a car park, or a foot path. In this task the test persons are required to make use of the functionality they have become acquainted with during the walkthrough.

- In the third task the test person is asked to use the aerial imagery, which is displayed as a background layer in the editor, to make additional edits. In this task the test person is supposed to add two objects of his/her choice and to improve the geometries of two existing building outlines. This task may be more demanding, because editing of geometries is not covered in the walkthrough (but still a very common thing to do when editing OSM).

- A fourth task is given to the participants of the 'mapper' group only. They are asked to add a piece of information derived from fictitious local knowledge (a speed limit imposed on a particular section of a road) to the map. This task requires the test persons to make use of a function of *iD* not needed in any of the previous tasks (splitting a line). The reason why this task is not given to the novice participants is because this mapping technique is not very commonly used by first-time mappers and also because the first three tasks are likely to be time-consuming enough for participants of that group.

The set of instructions given to the participants, including the field paper, is presented in its original form in appendix B.

While the tasks given are supposed to imitate a typical mapping session, some compromises had to be made. For example, I have waived to send test persons out in the field in order to do actual mapping and fill a field paper by themselves. As the *iD* editor is not used in this part of the process (and because everything would have become much more time-consuming), this is not really required. Another reason for dismissing such an approach is that it would not produce comparable results, as different people would survey different things—unless told otherwise. Preparing one field paper for all test persons, on the other hand, has the advantage that the results will be comparable on the level of individual map objects.

While working on the tasks the test persons were allowed to use two external help sources, the OSM wiki (URL 11) and the iD introduction on the "LearnOSM" website (URL 30), in the case that they needed information on mapping recommendations or additional help with using the editor.

3.4.4. Debriefing

Immediately after the test the participants were asked for their subjective opinions on their performing in the tests. The questions were targeted at the test persons' pleasure and satisfaction in using the tool, and were intended to elicit retrospective comments on, e.g., situations in which they struggled or got lost. This is not exactly the retrospective *thinking aloud* Ericsson & Simon [1984] suggest, but it is nonetheless interesting to obtain subjective accounts of the participants' satisfaction and the tool's usability.

However, the participants' statements received after the test were not given priority over the data produced during the thinking aloud, for the latter are always expected to be less biased. Studies have shown that the participants' own perception of usability often does not correspond with the usability issues that have actually been observed (e.g., Krahmer & Ummelen [2004]).

3.5. Conclusion

In this chapter I have presented the methodology determining the planning and execution of the *iD* usability tests. The *thinking aloud* technique has been described in detail, and I have arrived at conclusions for its application in this case study, i.e. a detailed set of rules has been established on the basis of theoretical considerations. The choice of the method in preference over other usability evaluation methods has been justified. In this study the *thinking aloud* is furthermore supplemented with video recordings of the computer screen and mouse and keyboard logging.

Altogether, the chosen methodology is similar to the one applied in the Potlatch 2 usability study by Jones & Weber [2012]. However, their investigations were somewhat different in the test user profiles, the tasks, and the observation methods used:

- Participants in this research are divided in two groups, novice users and users experienced with OSM, whereas in Jones & Weber's study all test persons were new to OSM.
- Tasks given to the participants by Jones & Weber are different from mine, although they too represent typical beginner's work—except they are slightly more difficult perhaps. On the other hand, Jones & Weber did not make use of a field paper.
- As for the observation methods, Jones & Weber used eye tracking, which I did not, because the decision not to work in a lab setting was made in favour of greater flexibility in making appointments, thus widening the circle of potential participants.

- Jones & Weber's usability evaluation focused on the usability criterion of learnability (see section 2.2), whereas I evaluated *iD* with regard to an enlarged set of criteria.

The next chapter describes the case study in more detail, including the selection process and participants' characteristics, the execution of the pilot and actual tests as well as technical issues with them, terminology, and an overview of the results in terms of task completion.

4. Case study: Testing *iD*

With the methodology for the usability testing of *iD* discussed, this chapter reflects on the planning and execution of the actual tests. The description covers the whole process starting from the online survey that was used to select participants to the pilot and actual tests, and it is followed by a summary of the test results and an overall reflection.

4.1. Selecting test persons

4.1.1. Online survey

An online survey (see appendix A) was sent out to the OSM community, as described previously in section 3.4.2, mainly to attract test persons of the mapper group. In order to attract participants for the novice group—as far as they could not be found through the OSM community—, I also reached out to acquainted persons of mine. The goal was to find up to 10 test persons with diverse characteristics and backgrounds for each group.

The survey for one thing served as a means to distinguish the respondents between potential novice group and mapper group participants. For this purpose I asked the questions "Have you ever contributed data to OpenStreetMap (OSM)?" (possible answers: "Yes" and "No") and "How often do you edit OSM (approximately)?" (possible answers: "several times a week", "once a week", "several times a month", "once a month", "several times a year", "once a year", and "I have edited no more than one or two times"). To the novice group I assigned those respondents who answered either "No" to the first question (in which case the second question was not displayed at all), or "Yes" to the first question *and* "I have edited no more than one or two times" to the second question. All other responses were categorized as of the mapper type.

For another thing, the survey also helped to collect some basic background information about the respondents, to enable me to assemble groups of test persons with the most diverse characteristics possible. This included questions about the ways the respondents contributed data to OSM, when they contributed to OSM for the last time, how frequently they contributed to OSM, their preferred OSM editor programs, and their overall experience with IT, GIS, and user-generated content on the web. Additionally personal information such as the respondents' age, highest education degree, and occupation was collected.

Lastly, the survey included a field for an e-mail address or phone number, in order to allow me to contact the respondents for the purpose of arranging a time and date for the actual test.

4.1.2. Survey results

Altogether I received 13 valid responses (i.e. completely filled forms with valid contact information) of the novice type, and 26 valid responses of the mapper type. This includes responses not only from those persons who found the survey through announcements in the community (all mapper and 8 novice respondents), but also from those who I pointed at it personally (the remaining 5 novices). I invited respondents successively, each time only as many as I needed. When any of the invited candidates cancelled or did not reply at all, I started another round of invitations until a sufficient number of test persons were found. Whenever I selected respondents for an invitation, I took care to choose the ones whose characteristics were most underrepresented among the ones I had already invited, to make sure the actual test persons' characteristics would be as diverse as possible—with respect to OSM editing experience as well as literacy in IT, GIS, and social media.

Table 4.1 lists the answers given by all (valid) respondents to the online survey. Eventually I managed to select nine test persons from each group, novices and mappers. Individual responses to the survey questions by the selected test persons are given in Tables 4.2 and 4.3.

Moreover, graphical representations of the responses of the 18 accepted test persons, separated by novice and mapper groups, are listed in Figures 4.1 through 4.10.

Apart from the respondents' OSM and IT/GIS/social media experience, some basic demographic information has also been collected, such as age, education, and occupation:

- The test persons' age ranged between 19 and 65, with an average age of 40 years.
- The test persons' education included various school qualifications (5 persons) and university degrees (13 persons).
- The test persons held occupations in various fields; 8 persons stated that they worked in a computer science-related field.
- 2 out of 18 test persons were female, 16 were male.

Survey item	*responses (of 39 total)*	*responses (of 18 selected test persons)*
Q1: Have you ever contributed data to OpenStreetMap (OSM)?		
– Yes	28	11
– No	11	7
Q2: How did you contribute?* (multiple choice)		
– (a) edited the map in my neighbourhood	27	10
– (b) edited the map in places I went (e.g. during vacation)	21	8
– (c) edited the map in places I didn't go (e.g. using aerial imagery)	18	6
– (d) uploaded GPS tracks	17	8
Q3: When did you contribute to OSM for the last time (roughly)?*		
– less than a week ago	15	3
– less than a month ago	3	2
– less than six months ago	6	3
– more than six months ago	4	3
Q4: Which editor do you use mostly when you are editing OSM?*		
– iD	0	0
– Potlatch 2	4	2
– JOSM	22	8
– Merkaartor	1	0
– I don't know	1	1
Q5: How often do you edit OSM (approximately)?*		
– several times a week	10	1
– once a week	4	1
– several times a month	1	1
– once a month	2	1
– several times a year	8	4
– once a year	1	1
– I have edited no more than one or two times	2	2
Q6: How is your experience with information technologies in general?		
– I can accomplish basic computer tasks	4	2
– I can accomplish advanced computer tasks	17	6
– I am an IT (quasi-)professional	18	10
Q7: How is your experience with geographic information systems (GIS) other than OSM?		
– I have no idea	19	10
– I have some experience with GIS software	13	8
– I have lots of experience with GIS software (3 years or more)	7	0
Q8: How is your experience with user-generated content on the web? (multiple choice)		
– (a) I use social networks	26	10
– (b) I upload photos or videos	26	11
– (c) I have a blog	16	6
– (d) I contribute to crowdsourcing projects other than OSM	18	8

Table 4.1.: Responses to the survey by all respondents and by the 18 selected test persons (* displayed only when the answer to the first question was "Yes")

	Q1	Q2a	Q2b	Q2c	Q2d	Q3	Q4	Q5
Novice 1	No							
Novice 2	No							
Novice 3	No							
Novice 4	No							
Novice 5	Yes	X			X	> 6 months	Potlatch 2	1 or 2 times
Novice 6	Yes	X				< 6 months	I don't know	1 or 2 times
Novice 7	No							
Novice 8	No							
Novice 9	No							
Mapper 1	Yes	X				< 6 months	Potlatch 2	once a year
Mapper 2	Yes		X		X	> 6 months	JOSM	several a year
Mapper 3	Yes	X	X	X	X	< 6 months	JOSM	several a year
Mapper 4	Yes	X	X	X		< 1 month	JOSM	once a month
Mapper 5	Yes	X	X	X	X	< 1 week	JOSM	once a week
Mapper 6	Yes	X	X	X	X	< 1 week	JOSM	several a month
Mapper 7	Yes	X	X	X	X	< 1 week	JOSM	several a week
Mapper 8	Yes	X	X		X	> 6 months	JOSM	several a year
Mapper 9	Yes	X	X	X	X	< 1 month	JOSM	several a year

Table 4.2.: Individual responses to the survey by the 18 selected test persons, questions 1–5 (see Table 4.1 for question codes)

	Q6	Q7	Q8a	Q8b	Q8c	Q8d
Novice 1	advanced	no idea		X		X
Novice 2	basic	no idea	X	X	X	
Novice 3	professional	no idea				X
Novice 4	professional	some exp.	X			
Novice 5	professional	some exp.	X	X	X	X
Novice 6	professional	no idea	X			
Novice 7	basic	no idea		X		
Novice 8	advanced	no idea				
Novice 9	advanced	no idea	X	X		
Mapper 1	professional	some exp.	X		X	
Mapper 2	professional	some exp.				
Mapper 3	advanced	some exp.		X		
Mapper 4	advanced	some exp.	X	X	X	X
Mapper 5	professional	some exp.	X	X	X	X
Mapper 6	professional	no idea		X		X
Mapper 7	advanced	no idea				
Mapper 8	professional	no idea	X	X		X
Mapper 9	professional	some exp.	X	X	X	X

Table 4.3.: Individual responses to the survey by the 18 selected test persons, questions 6–8 (see Table 4.1 for question codes)

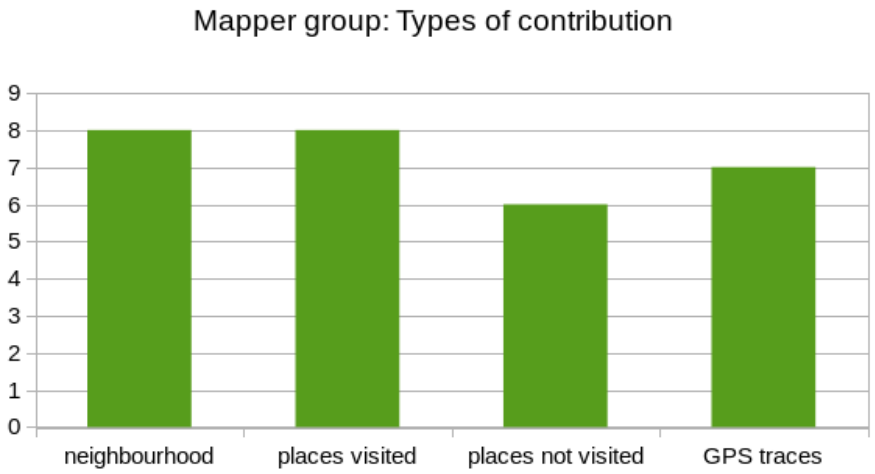


Figure 4.1.: Mapper responses to the question “How did you contribute?” (multiple replies possible), n=9

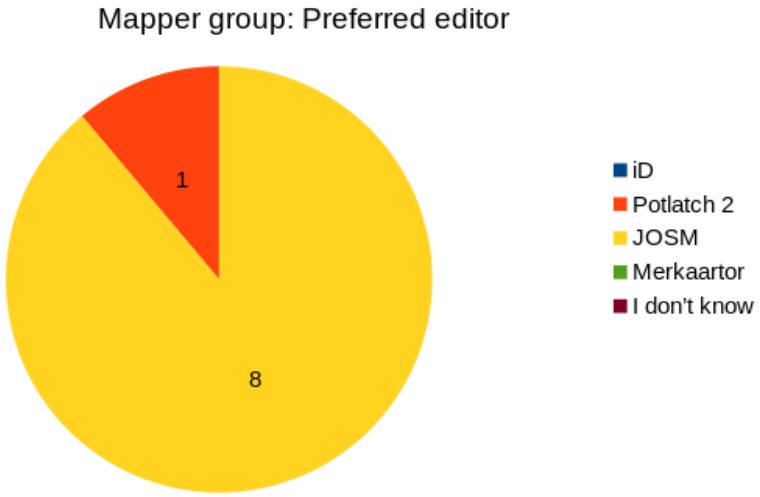


Figure 4.2.: Mapper responses to the question “Which editor do you use mostly when you are editing OSM?”, n=9

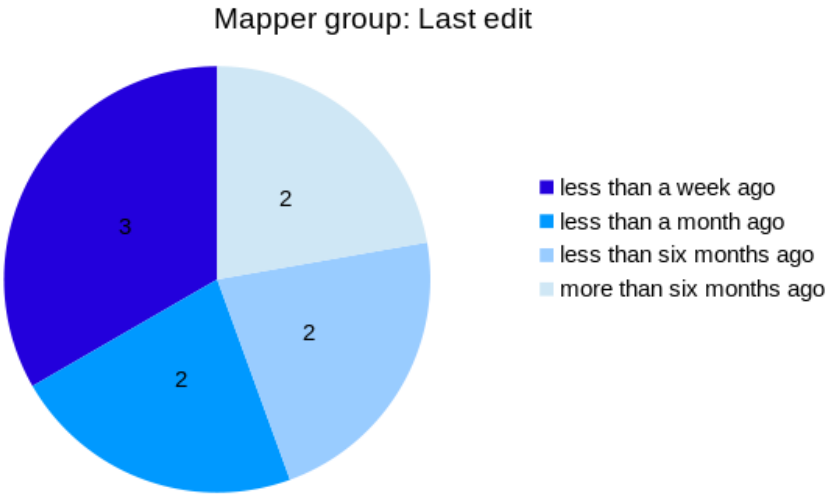


Figure 4.3.: Mapper responses to the question “When did you contribute to OSM for the last time (roughly)?”, n=9

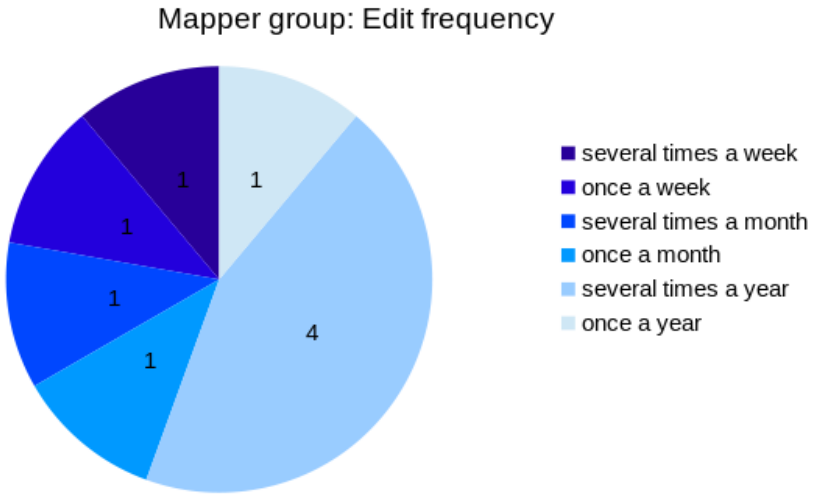


Figure 4.4.: Mapper responses to the question “How often do you edit OSM (approximately)?”, n=9

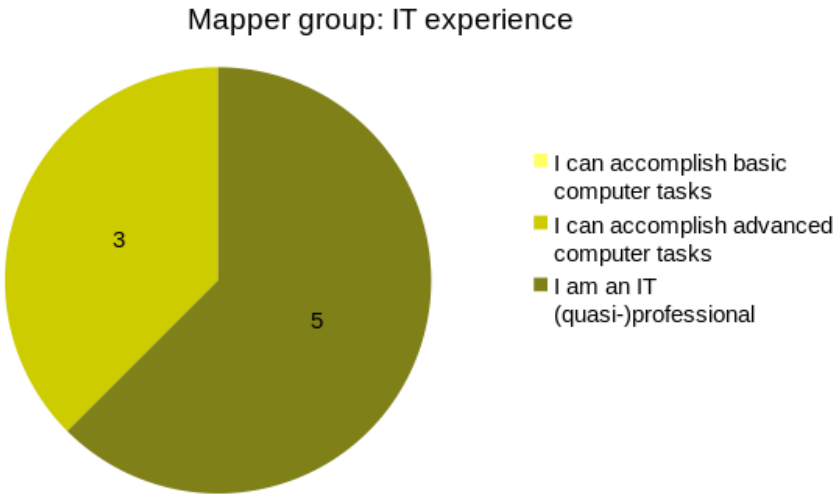


Figure 4.5.: Mapper responses to the question “How is your experience with information technologies in general?”, n=9

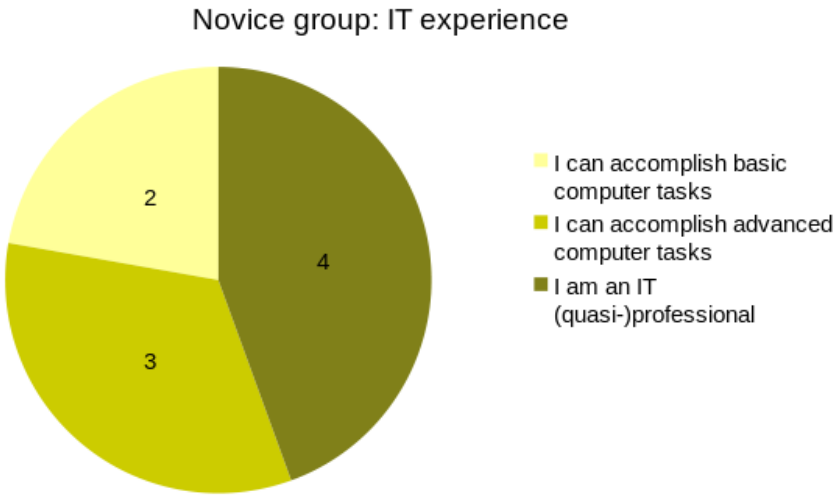


Figure 4.6.: Novice responses to the question “How is your experience with information technologies in general?”, n=9

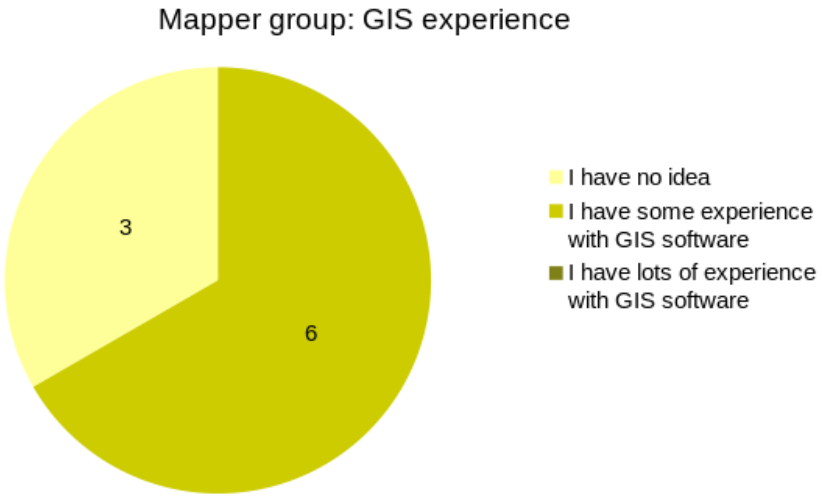


Figure 4.7.: Mapper responses to the question “How is your experience with geographic information systems (GIS) other than OSM?”, n=9

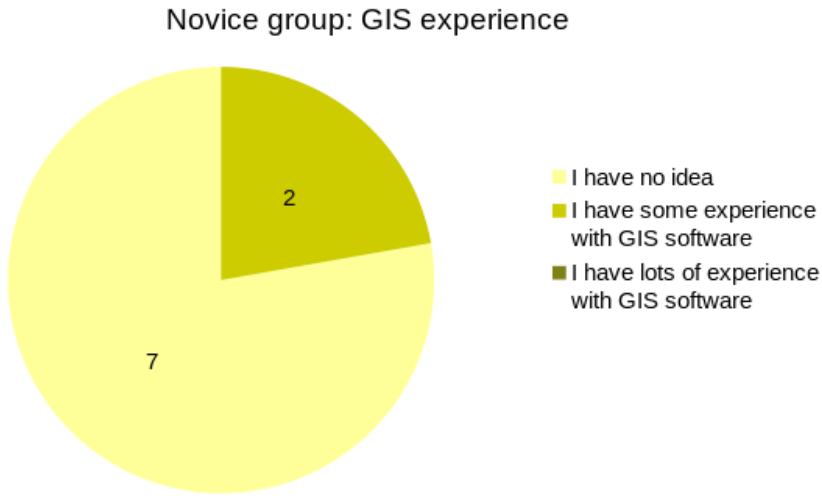


Figure 4.8.: Novice responses to the question “How is your experience with geographic information systems (GIS) other than OSM?”, n=9

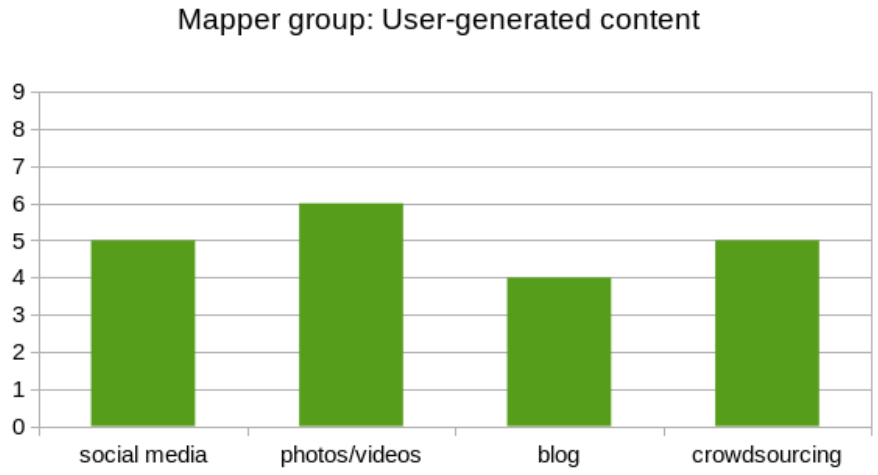


Figure 4.9.: Mapper responses to the question "How is your experience with user-generated content on the web?" (multiple replies possible), n=9

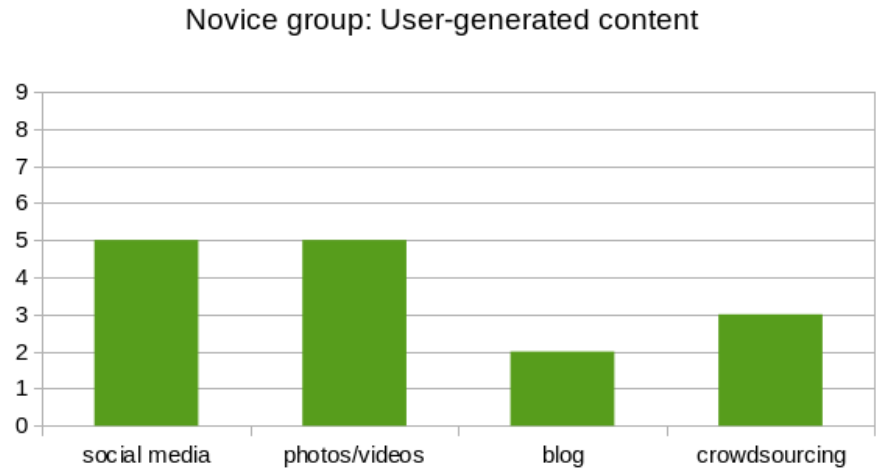


Figure 4.10.: Novice responses to the question "How is your experience with user-generated content on the web?" (multiple replies possible), n=9

4.2. Pilot testing

The user tests were preceded by two pilot tests, one for each group, to find out whether the test was designed properly, in terms of suitability for the participants and in terms of technical procedure.

The first pilot test was conducted with a person who had never contributed to OSM before. It turned out that task 3 was not formulated clearly enough, as the pilot tester did not understand the task, even after reading it several times. Consequently, I modified the task wording so that the assignment became more understandable. I also added some buildings to the map data which did not exist in the original OSM data in order to create better opportunities for the task, which required modifying existing buildings.

The second pilot test with an experienced mapper user went smoothly. Both pilot tests took around 45 minutes, which I considered to be in an acceptable range.

4.3. The test sessions

4.3.1. Organisational aspects

The actual user tests were conducted between 1 April and 25 April 2014. While most of the tests were done at my home, some tests were done at places chosen by the test persons, i.e. either in a public place, or at the test persons' homes or offices: out of 18 tests 5 took place somewhere else, other than my home.

This flexible organisation was possible thanks to the fact that both the editor program and the recording tools could run simultaneously on a moveable notebook computer. Additional equipment used during the tests included an optical mouse (as the mapping requires high cursor precision and some people are not used to touch devices in notebooks) and a headset microphone for recording the test person's voice.

When a test was executed at my home (with the exception of the first one), I took advantage of the possibility to run the OpenStreetMap server, which also hosts the *iD* application, on a different machine. In that case, the browser on the notebook computer on which the test person worked accessed the local OSM server running on the other machine over the local network. The reason for this approach was that the computational load on the notebook was reduced, which I expected to lower the risk of gaps in the audio and screen recordings. With respect to the validity of the results, there is no concern about this issue, as the same versions of *iD* were running on both machines. On the contrary, when a test was not executed at my home, the OpenStreetMap server ran on the notebook, i.e. the same machine that the participants were working on. The dropout rate of the video and audio materials, however, was not affected very much by this choice, and all materials were at least partially usable.

All except one test were conducted in German, because the participants were mostly German speakers. The transcripts of the think aloud data presented in the analysis section are my translations of the original transcripts into English. (For reference, the original audio recordings are included on the DVD supplemented to this thesis; see appendix C.)

4.3.2. Post-test interviews

After each test I interviewed the test persons and recorded their answers on the same audio stream. In the interviews I asked them about their own impression of the editor program, their performance in the tasks, their suggestions for improving *iD* (if any), and some extra questions referring to their responses given in the online survey (e.g., whether and how they have used OSM before). Depending on the issues the test person have had using *iD*, I sometimes also asked them to comment on situations they found themselves unable to handle.

The interviews showed that the majority of test persons were overall satisfied with the editor. However, they often also made remarks about particular situations in which they experienced difficulties. Interestingly, the issues they referred to did not always include the ones actually observed. A more detailed description of the participants' subjective accounts given in the interviews is presented in section 5.3. The wording of the prepared interview questions is given in appendix B.

4.3.3. Technical specifications

Hardware:

- Lenovo X130e notebook (1.65 GHz dual-core processor, 3.6 GB memory)
- customized desktop computer (4.6 GHz dual-core processor, 4 GB memory)
- Logitech H330 headset
- Havit optical mouse

Software:

- Operating system: Ubuntu 13.10
- Browser: Chromium 34.0.1847.116-0ubuntu
- OpenStreetMap website (the "Rails Port"): downloaded from GitHub (URL 31) on 17 March 2014 (commit 4cad19), containing *iD* development version 1.3.7
- Audio/video recording software: "FFmpeg" version 2.2, downloaded on 10 March 2014 (URL 32)
- Keyboard logging software: "key-mon" version 1.16-1ubuntu1

4.3.4. Technical issues

Due to multiple applications running simultaneously on the computer, some technical difficulties could not be avoided. Despite careful experimenting with and testing of the recording application, frame dropping occurred rather frequently in the produced audio and video files (i.e. failure in the recording of data due to computational overload), sometimes more and sometimes less. As far as the audio is concerned, this could result

in gaps of silence within the recording, and in the video files this resulted in duplication of frames, which would look as if the image was frozen. This is unfortunate insofar that parts of the recorded video and audio are practically unusable. The frame dropping did not seem to follow a predictable pattern and could result in data loss of up to 40 % in the audio recordings, albeit not more than an estimated 20 % in the screen recordings. Though it must be conceded that the validity of the research is, in fact, flawed by this recording issue, due to the fact that there were nine test persons per group (an amount somewhat greater than the minimal recommendations given in literature, as described in section 3.3), it can still be argued that a sufficient amount of qualitative data has been gathered in order to justify a usability evaluation.

Another problem was the synchronisation of the screen video and audio data. Although the recording of both the video and audio streams happened within a single process and thus should have remained synchronised, the aforementioned frame dropping led to the streams falling out of synchronisation. Therefore the streams had to be reconstructed, for which I used the video editing software "Cinelerra" (URL 33). I resynchronised them—as far as possible—by rearranging parts of the audio and video streams based on noticeable correlations between the two streams (e.g., mouse clicks seen in the video and heard in the audio). The synchronised streams from each test were finally exported to a single video file.

Supplemented to this thesis is a DVD (see appendix C) containing the videos produced, where permission for publication has been granted by the respective participants (16 out of 18).

4.4. *iD* terminology

In order to better understand the upcoming presentation of results and analysis of the *thinking aloud* protocols, it is necessary first to provide a detailed description of the *iD* editor and its functionality and to introduce the terminology used in this thesis to name its elements.

Figure 4.11 shows the OSM main page (URL 1). The *iD* editor is opened by clicking the "Edit" button and replaces the map view on the OSM main page as soon as it has started up. The editor itself (Fig. 4.12) consists of the *main pane*, the *map window*, the *tool bar*, the *map panel*, and the *information panel* (see also URL 30 for a detailed description of iD's user interface).

The main pane is always visible. As long as nothing is selected, it shows the *search form* (Fig. 4.12). When a feature is selected or hovered over with the cursor, it shows the *feature editor* instead (Fig. 4.13a). The feature editor contains a *header* that indicates the selected feature's *type*. Below that, different attribute fields are available to let the user change the feature's attributes (in Figure 4.13a: Name, ATM, Address, and Hours). Below the fields there is a row of *attribute icons* one can click to open new attribute fields (such as phone, website, etc.). Below the icon row there is an expandable section "All tags" that lets the user edit tags directly, and the "All relations" section, where a feature's membership in relations can be inspected (these two are not visible in Fig. 4.13a). The "All relations" section is only mentioned for the sake of completeness;

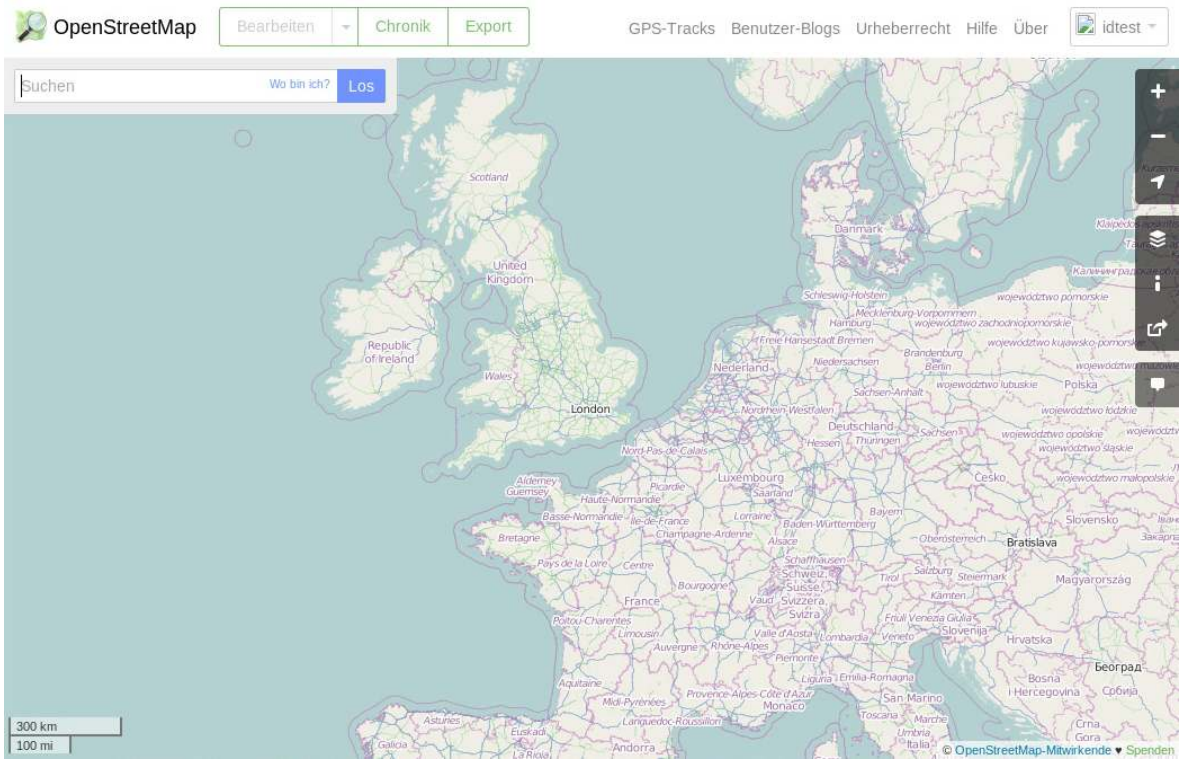


Figure 4.11.: The OSM main page

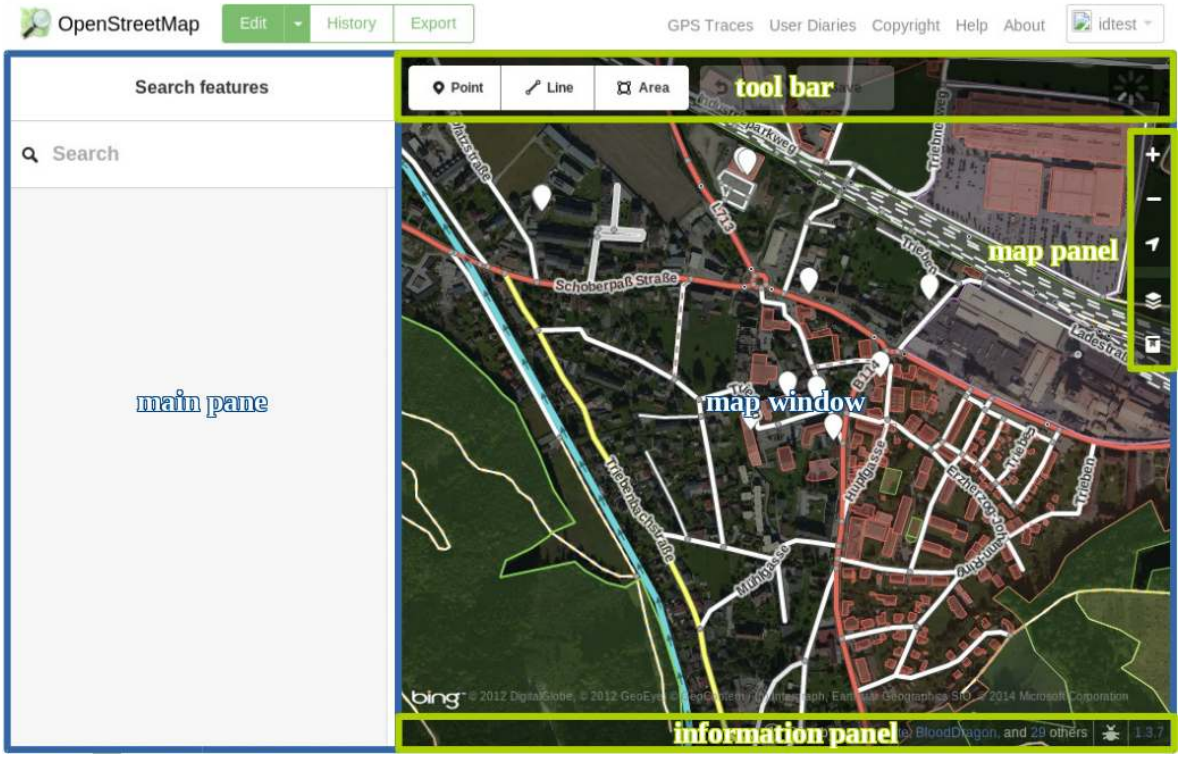
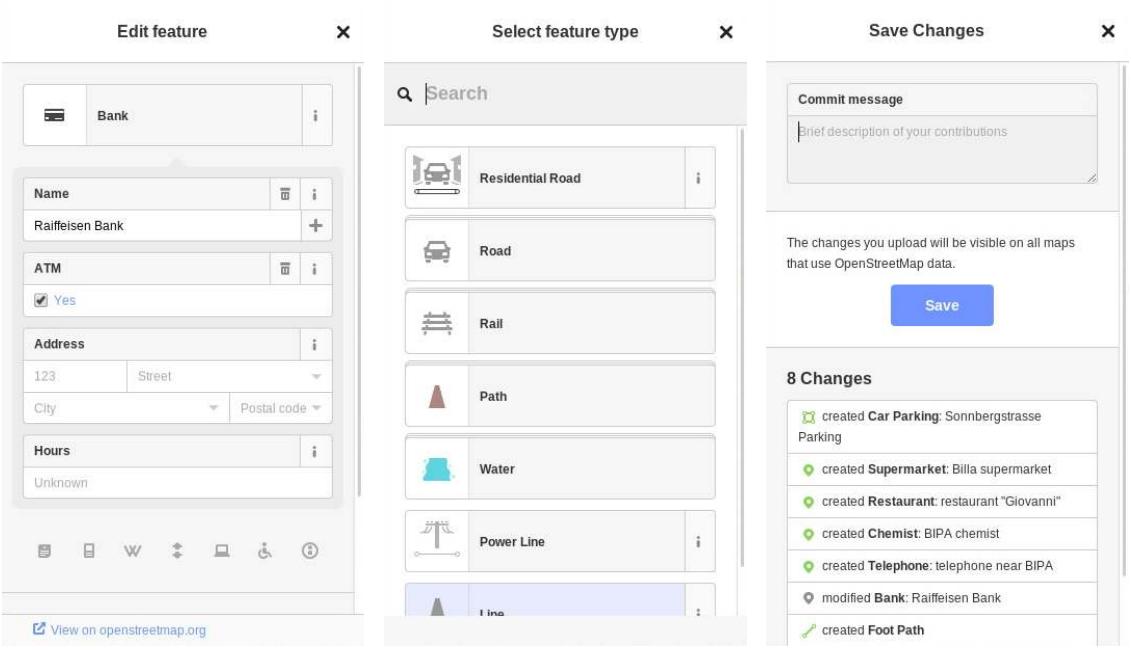


Figure 4.12.: The *iD* editor and its components



(a) The feature editor (b) The feature type search (c) The save dialogue

Figure 4.13.: The main pane showing different kinds of content

it is not relevant to this research, as the tasks given to the test persons did not require relation editing.

When the user has finished placing a point or drawing a line or area, the main pane shows the *feature type search* (Fig. 4.13b). At the top there is a search form that helps to find the desired feature type. Below it there are several feature types, suggested by the editor, that can be selected by clicking on them. Among them there are *singular feature types* and *categories*. The latter are, in principle, expandable lists of singular feature types and are recognisable by their "multi-layered" appearance (e.g. the *Road* category in Fig. 4.13b). Upon selecting a feature type the feature editor shows up.

Whenever a user clicks on the "Save" button in the header bar, the *save dialogue* shows up (Fig. 4.13c). It contains a text field for the *commit message*, which is stored alongside the map changes in the OSM database, the save button, and a *list of changes*.

When a feature is selected by clicking on it, the *circular menu* pops up, centred at the location of the cursor. It contains different tools, which are symbolized by different icons. The available tools depend on what kind of object is selected (point, line or area, and how it is related to other objects), e.g. while a point's circular menu only has a delete function (Fig. 4.14a), an area's circular menu also has functions for rotating it, squaring its corners, moving it, and making it circular (in clockwise order) (Fig. 4.14b).

(a) The circular menu at a point object

(b) The circular menu at an area object

Figure 4.14.: The circular menu

4.5. Completion of the tasks

4.5.1. Success in the tasks

As an overview, this section presents statistical information about the participants' performance in the given tasks (see section 3.4.3 for a detailed description and appendix B for the wording of the instructions).

Task 1. The first task was to complete the walkthrough and was successfully completed by all test persons, as they only needed to follow the instructions given on the screen. (A detailed analysis of the observations made during this task is given in section 5.2.1.)

Task 2. The second task entailed entering map information that was given on a field paper in analogue form. Success in this task can be measured by whether the test persons achieved to add the information that the task wanted them to add. There were nine objects to be added or modified (Table 4.4).

Notation on field paper	*Correct feature type (with actual tag) and/or additional attributes*
Billa supermarket (point)	Type: Supermarket (shop=supermarket) Name: Billa
car parking (area)	Type: Car parking (amenity=parking)
restaurant "Giovanni" (point)	Type: Restaurant (amenity=restaurant) Name: Giovanni
BIPA chemist (point)	Type: Chemist (shop=chemist) Name: BIPA
telephone (point)	Type: Telephone (amenity=telephone)
Raiffeisenbank (point, existing) address: Hauptplatz 2 phone: 03615 22170	Housenumber: 2 Street: Hauptplatz Phone: 03615 22170
foot path (line)	Type: Foot Path (highway=footway)
Triebener Bundesstraße (line, existing)	Name: Triebener Bundesstraße
hotel "Triebenerhof" (point)	Type: Hotel (tourism=hotel) Name: Triebenerhof

Table 4.4.: Individual map items of the field paper task and their solutions

Some tolerance is given with respect to the attributes added, e.g., the name entered for the hotel could be one of Triebenerhof, "Triebenerhof", Hotel "Triebenerhof", or even other variations. Similarly, different formatting was accepted for the address and phone number fields. For the phone number also two different tags were accepted, namely "phone" and "contact:phone" (the latter could be applied when the tag was added directly from the "All tags" editor and not from the available *Phone* field).

More variety was observed in the mapping of both the chemist and the hotel: While some users created a new point for these, others decided to edit the building outlines

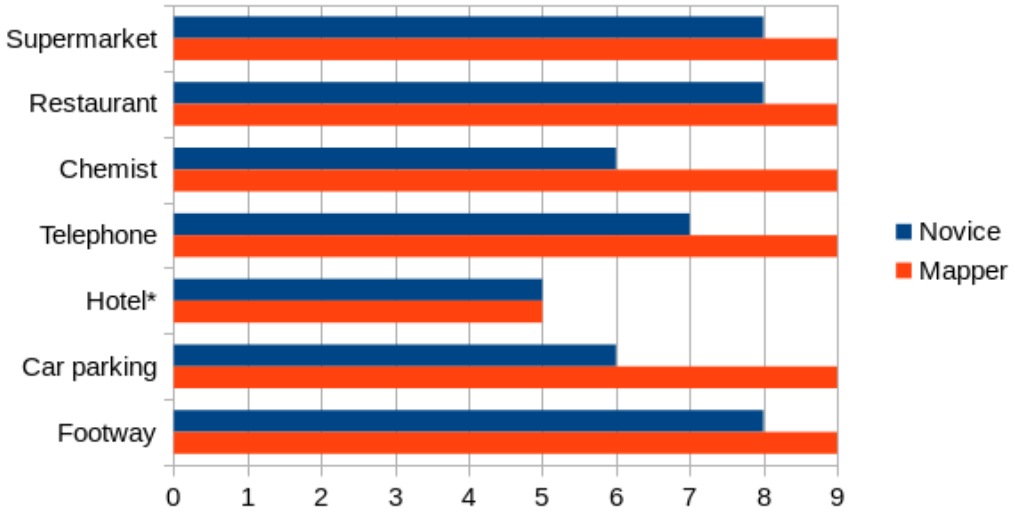


Figure 4.15.: Success in using the correct feature types in task 2, n=9 (* Hotel/Novice: n=8)

located at the marked spots and to change their attributes to reflect what is inside the building. Both approaches were accepted and regarded as correct (as long as the type and attributes were correct).

Figure 4.15 shows how many of the test persons of each group used the correct feature types, and Figure 4.16 shows how many of them added the correct attributes. As the hotel was erroneously missing on the field paper for the first novice test person, the total number of novice users adding a hotel was 8. A special case is the task of adding the name of the primary road: The segment that was marked on the field paper was represented in the map data by two line objects; Fig. 4.16 shows success rates both for adding the name to at least one segment (partially correct), and for adding it to both segments (fully correct).

Task 3. Task 3 asked the test persons to add two objects of their choice to the map using the aerial imagery in the editor's background layer. As the test persons were given a lot of freedom in completing this task, one can only measure if any meaningful feature types have been used at all. This is in fact true for all features that have been added except two that have been added by two novice users: they added area objects using the generic feature type Area and only added a name to it.

Task 4. The extra task, given only to test persons of the mapper group, asked the test persons to add attributes to segments of existing lines, requiring them to use the split functionality. This task was successfully completed by all 9 mapper users.

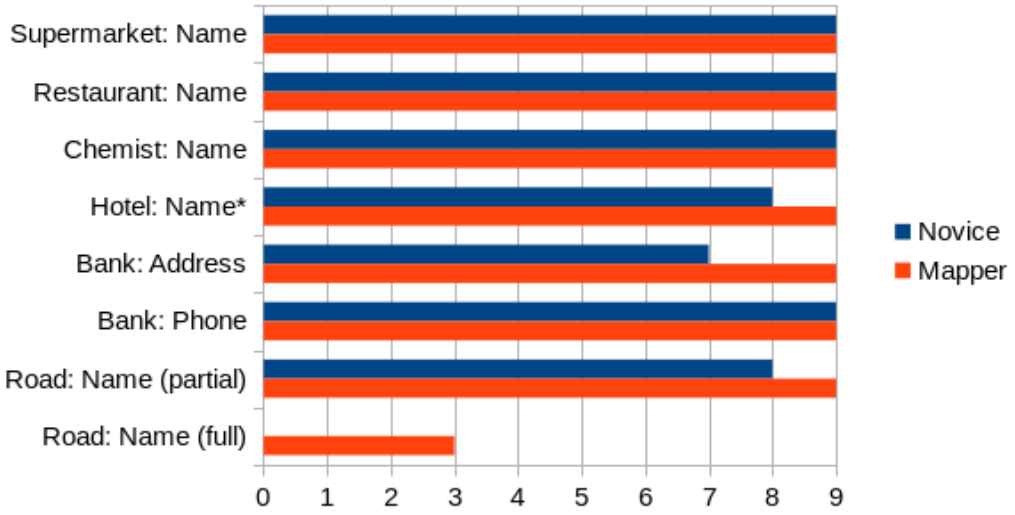


Figure 4.16.: Success in adding correct attributes in task 2, n=9 (* Hotel/Novice: n=8)

4.5.2. Task completion times

Apart from the success rates, the average time test persons of each group needed to complete each task is given in Table 4.5. Although the test persons had been told that they could take all the time they needed, it is still a good indication of how efficiently they worked.

Task	*Average time*		*Minimum time*		*Maximum time*	
	novice	*mapper*	*novice*	*mapper*	*novice*	*mapper*
Task 1	9 min	6 min	5 min	2 min	18 min	12 min
Task 2	17 min	15 min	7 min	6 min	29 min	22 min
Task 3	15 min	8 min	8 min	4 min	26 min	11 min
Task 4	–	5 min	–	3 min	–	10 min

Table 4.5.: Average, minimum, and maximum task completion times by user group

4.6. Reflection

In this chapter I have presented the procedural and technical aspects of the user test, terminology that will be used in the analysis, and an evaluation of the overall test results.

The test persons taking part in the test fulfilled the criteria that had been defined before. Not only has a sufficient number of test persons been found, but also were they diverse in terms of mapping experience and computer literacy, so as to allow for a distinguished analysis of the results. It must be noted, however, that the distinction between the novice and the mapper group was not always very clear, as several novice

users indeed had some minor experience with OSM, while some mapper users were still only very occasional contributors or had not contributed for a longer period of time. This, however, is not surprising as in reality there is usually no clear line between two arbitrary groups of users; on the other hand, the distinction worked well enough to allow for different conclusions regarding usability for the respective groups.

It is also obvious that not all test persons would have been motivated to use the *iD* editor if they had not taken part in the test—either because they had little interest in OSM in general, or because they were too advanced to consider using an online editor primarily designed for beginners in the first place. However, the usability issues these users have had are still relevant for a usability evaluation of *iD*.

The next chapter presents an analysis of the data produced by the test persons, including precise descriptions of the issues as well as further conclusions regarding the usability of *iD*, measured by various usability criteria.

5. Analysis of the outcomes

The previous chapter has given a summary of the organisational aspects of the user tests and presented the terminology used to describe the observations made during the tests as well as an overview of the results.

In this chapter, the observations elicited from the participants' *thinking aloud* data is presented in all detail. Relevant sections of the recordings that exhibit issues with the editor or other interesting remarks by the participants have been identified and coded depending on the type of interaction they were related to. In the first section of this chapter the coding scheme with which the selected segments have been coded is presented. In the following section the most interesting excerpts from the *thinking aloud* are quoted in transcripts and discussed. The chapter concludes with a review of the post-test interviews and an overall evaluation of the editor's usability in terms of learnability, efficiency, error tolerance and subjective user satisfaction.

5.1. The coding system

The first step in analysing the recordings was to identify the parts relevant to answering the research questions guiding this study—particularly the question how usable the *iD* editor is. For this purpose a coding system was used. The coding of the *thinking aloud* data first requires a selection of the particular segments that seem relevant for the analysis. Second, the selected segments are assigned different codes that relate to specific situations or subtasks that appear during the test persons' interactions (Ericsson & Simon [1984], p. 5f.). The coding system used in this study had not been defined *a priori*, but was determined by the contents of the data itself, because this study is not based on a particular theoretical model, but rather by what actually happened during the users' interactions with the tool.

The qualitative analysis software ATLAS.ti 7, trial version (URL 34) was used to assign codes to individual sections of the videos.

Table 5.1 contains a list of the codes, grouped in categories used to describe interesting fragments of the recordings. The number of appearances in the novice and mapper groups, respectively, is not an indicator of the success of the two groups in individual tasks, as both problematic and particularly successful interactions could be coded. Also, non-coding can either mean that the interaction was flawless or uninteresting, or that it has not happened at all.

The categorization of the codes is not strictly organised by the tasks given, but rather by smaller subtasks. The reason is that subtasks, such as selecting a feature type or adding an attribute, do often appear across different tasks and should not be separated. An exception to this approach is the walkthrough, which presents unique situations as in it the interaction is guided by a structured set of instructions.

Table 5.1.: Codes used in the analysis of the thinking aloud data

Code	*Description*	*Appearances (novice)*	*Appearances (mapper)*
Walkthrough			
w_selectpoint	selecting a point on the map	3	5
w_closefeatureeditor	task of closing the feature editor after inspecting the townhall	1	0
w_point	clicking the point button	0	1
w_selectfeaturetype	choosing a feature type (cafe, playground, or road)	4	5
w_addname	task of adding a name (cafe, playground, or road)	18	13
w_deletepoint	task of deleting the cafe	5	4
w_drawarea	task of drawing the playground	5	3
w_drawline	task of drawing the road	4	2
w_startmapping	tooltips at the end of the walkthrough	0	2
w_contexthelp	the context help (tooltips) appearing next to the cursor	3	0
w_bug	any software bug that appeared	8	5
w_remark	any other remark	1	1
Navigation			
pan	panning (moving) the map	2	0
zoom	zooming the map	1	3
Save			
save	saving the changes	4	8
Search			
search	using the search function for places	4	8
Visual elements			
circularmenu	the circular menu appearing on top of map objects	0	1
highlight	the visual highlight of selected features	0	1
mainpane	the “main pane” (on the left)	0	1
maplabels	map labels	1	0
translations	translations of user interface elements	1	0
Selecting objects			
selectpoint	selecting a point on the map	1	0
selectmultiple	selecting multiple features at the same time	0	5

Table 5.1 (continued from last page)

Selecting a feature type			
selectfeaturetype	selecting the feature type of a newly created object	21	14
changefeaturetype	changing the feature type of an object	8	10
Basic editing			
closefeatureeditor	closing the feature editor ("main pane")	2	1
editattributes	editing attributes other than the feature type	0	2
address	adding an address (bank on the field paper)	5	5
phonenumber	adding a phone number (bank on the field paper)	6	6
alltags	use of direct tag editing	3	6
inspectfeature	inspecting the geometry (validity) of a feature	0	1
residentialarea	problems with the residential area covering a large part of the town	0	1
Editing of lines and areas			
drawline	drawing a line	4	2
drawarea	drawing an area	7	6
movewaynode	move a node of an existing line or area	12	10
addwaynode	adding a node to an existing line or area	11	10
rotate	rotating an area object	0	1
squarecorners	squaring the corners of an area	4	5
splitline	splitting a line into two	0	5
mergelines	merging two existing lines in order to create a single object	0	2
disconnectpoint	disconnecting two objects from each other that share a node	1	0
Use of help sources			
learnosm	use of the "LearnOSM" page	3	4
wiki	use of the OSM wiki	9	4
Other			
closeeditor	closing the editor	0	1
keyboardshortcut	use of keyboard shortcuts	2	0
bug	any software bug that appeared	1	0
posttest_interview	the post-test interview	9	9
remark	any remark not related to any of the previous codes	1	1

5.2. Analysis of the thinking aloud protocols

The analysis of the observations, which is presented in this section, is structured by the codes and their categories listed in Table 5.1. Quotations from the *thinking aloud* protocols are given in the following form:

> Test person [`code`]: description of what the user does; "quote from thinking aloud"

5.2.1. Walkthrough

The "walkthrough" is a built-in feature of the *iD* editor that acts as a tutorial for first-time users. It guides the user through basic tasks of navigation, point editing, and area and line drawing. While all test persons managed to complete the walkthrough, in some instances there was some confusion about the meaning of interaction elements, or the procedure.

In this section only issues appearing specifically in the walkthrough are discussed. Anything notable that happened during the walkthrough, but could have happened at any other time as well (e.g., problems selecting a point) is discussed in the respective sections below. This analysis is structured by the logical sequence of the steps in the walkthrough, which is divided in five parts.

Part 1: Navigation In the navigation step the user is asked to drag the map. This is the easiest task, but often at this point the program hung because of a software bug. See under "Other issues" in this section for a discussion of bugs appearing in the walkthrough.

Next, the user is asked to select a point and get familiar with the so-called feature editor. Apart from problems selecting the node (see below in section 5.2.6), there was another issue: The task of closing the feature editor after the town hall had been examined required the user to find the "close button in the top right". This was unclear to one user, as the top right of the main pane, not of the editor window, was meant.

> Novice 7 [`w_closefeatureeditor`]: "Close the feature editor with the close button in the top right"; searches for the close button in the upper right corner of the editor window

Part 2: Points The task of pressing the *Point* button, introducing the section about adding and modifying points, created confusion with one user.

> Mapper 6 [`w_point`]: "OK, other symbols than I know from JOSM, but the same content... and now I can select a fuel station here, for example..."; clicks on the fuel icon in the instructions window; "probably I'll have to double-click again"; double-clicks; "... or not"; "oh, I have to... ah, not click in the help field, but on the actual... ah, yes"; clicks *Point* button

The task to place a node on top of a building was easily completed by all test persons. However, there was an issue with the subsequent selection of the feature type *Cafe*: some tried to click on the *Cafe* button, although the highlight was on the search form, and the button thus not clickable.

> Novice 6 [`w_selectfeaturetype`]: "Search for Cafe"; clicks on *Cafe* while only the search form is highlighted (several times); "but it's there... do I have to type it in the search up there, although it is being shown?"
>
> Mapper 4 [`w_selectfeaturetype`]: clicks on *Cafe* while only the search form is highlighted; "Why should I search when it's already there?"
>
> Mapper 7 [`w_selectfeaturetype`]: clicks on *Cafe* while only the search form is highlighted; "Search..."; clicks in search form; "ah, Cafe, OK"
>
> Mapper 8 [`w_selectfeaturetype`]: clicks on *Cafe* while only the search form is highlighted; "When I already see 'Cafe' I should be able to select it right away"

Other remarks users made about the feature type selection:

> Novice 3 [`w_selectfeaturetype`]: "I'm supposed to search for Cafe"; clicks in search form; "so a kind of feature type window had popped up here, I suppose it appears whenever I add a new point and it's not only something special from the tutorial... I just hope so, otherwise it would make the editing more difficult"
>
> Novice 4 [`w_selectfeaturetype`]: "Oh, already after the first letter? That's great."

A common issue users got stuck with in this task was adding a name to the cafe. The walkthrough did not react when a name had been entered, so many did not know how to move forward. The situation appears four times during the walkthrough: after adding the name for the cafe, after changing the cafe's name, and after adding the names to the playground and the residential road. However, the first instance was the most problematic case, because no indication was given by the tutorial that the feature editor needed to be closed until the user clicked somewhere outside of the name field.

> Novice 1 [`w_addname`]: presses Enter; "I press enter, it doesn't react to my command... so I'm gonna try with the plus sign"; clicks '+' next to the name field; reads: "The feature editor can be closed by clicking on the close button. Close the feature editor"; clicks close button
>
> Novice 1 [`w_addname`]: presses '+' button; "I pressed the plus sign, but I'm not sure if it's saved"; clicks in name field; clicks close button
>
> Novice 2 [`w_addname`]: presses Enter; "yeah, I did that"; clicks in the map; clicks close button

Novice 3 [w_addname]: “Well... obviously I’ve got to click on the plus button... no, it says ‘Translate’ when I click the plus... I’m not quite getting it now how I can apply the name. The plus doesn’t do it, so I’ll try Enter”; presses Enter; “no, doesn’t work”; “I’ll just click somewhere else”; clicks in another field; “that works, OK”

Novice 3 [w_addname]: “I just assume that the name is being applied even when I close the editor without doing anything else”

Novice 4 [w_addname]: clicks in another field; “... ‘can be closed by clicking on the close button’, that’s quite, err, that’s the way it is with many things”

Novice 5 [w_addname]: “I’ve done that... why doesn’t it notice?”; clicks in another field; “you have to go out of the field once”

Novice 6 [w_addname]: enters name and presses Enter; hovers over the ‘+’ button; clicks in the name field; presses Enter; clicks in the name field again; presses Enter again; “Nothing happens. I have entered a name...”; changes the name and presses Enter; hovers over ‘+’; “Translate is stupid”; scrolls down and inspects the icons; “so I’m looking for some kind of a button that is like ‘Save’ or something”; “next to Name there are three icons, the first I’d say is a wastebin, the arrow is undo, and the ‘i’ is info. Let’s see what comes with the ‘i’...”; then reads new assigment and closes the feature editor; “I was looking for save, because the ‘x’ can often also mean something like cancel without saving”

Novice 7 [w_addname]: enters name and presses Enter; “Add name, done... Cuisine?”; clicks on the instruction text and continues

Novice 7 [w_addname]: enters name; hovers over ‘+’; “Translate?”; then closes feature editor

Novice 9 [w_addname]: “I’d rather have looked for a save button instead of the close button... if you look for it intuitively, I wouldn’t necessarily have pressed the close button”

Mapper 2 [w_addname]: “Add a name - done... Translate? I don’t want that”; clicks on ‘+’ sign; reads instructions and closes feature editor

Mapper 3 [w_addname]: “Add a name - done. Translate? No. What? Or just press Enter?”; presses Enter; “Now I don’t know how to continue”; clicks ‘+’ sign; “ah, multilingual name, that is not what I want”; reads instructions and closes feature editor

Mapper 3 [w_addname]: “Probably I can just close it without pressing Enter or anything”

Mapper 5 [w_addname]: enters name; “Well... Translate? No...”; clicks outside of the name field and continues

Mapper 6 [w_addname]: enters name; clicks on *Cuisine*; “OK, out of that field by simply clicking elsewhere, obviously”

Mapper 7 [w_addname]: enters name; “Add a name, done that”; clicks in instructions window and closes the feature editor; “ooh-hoooh... uuuhh-huh”

Mapper 7 [w_addname]: “I dislike it with the close button there”

Mapper 8 [w_addname]: enters name, clicks in another field; “It knows that I have added it only after I’ve changed the field... I don’t like that very much, it’s confusing”

Mapper 9 [w_addname]: “Here this is weird, so I think I’d have to press save or something, that’s kind of... not intuitive”

The task of deleting the cafe created some confusion with the circular menu, first because it had not been noticed after selecting the point before, and second because the meaning of the delete button was not always obvious.

Novice 2 [w_deletepoint]: “Delete the point”; “The menu around the point...”; clicks delete button; “I don’t know what happened now . . . but I didn’t click on Delete”

Novice 3 [w_deletepoint]: “Oh, what was that? Now something different happened than last time I clicked on the point, now there’s a wastebin or something here . . . I’m wondering now why that wasn’t there before, in the step before”

Novice 6 [w_deletepoint]: “A wastebin is showing, but the help text doesn’t tell me to delete it” (indeed the assignment reads, ‘Click on the point you created’); double-clicks (the delete assignment appears)

Novice 8 [w_deletepoint]: right-clicks on the point; clicks browser menu away; hovers over the delete button; “well, ‘around the point’, so I’m expecting now that in the area of the selected object some kind of menu pops up, oh, wait... it isn’t revealing to me right now”

Mapper 2 [w_deletepoint]: instructions read “Delete the point”; clicks on the point again; instructions turn back to ‘Click on the point you created’; clicks on node and then on the delete button

Mapper 6 [w_deletepoint]: “Ah, so now we have a menu here around the point... OK”; clicks somewhere in the map; instruction goes back to the previous step; selects the node; “I guess that was wrong... or is that the wastebin? Well, there the text is covered unfortunately”; deletes the point

Mapper 7 [w_deletepoint]: “Delete the point”; hovers over the delete button; “... and how do I get there, to that delete button, if I want to control it myself? So, delete the point”

Mapper 9 [w_deletepoint]: clicks on the node; “Ah yes, here... well, symbols... they could have written ‘delete’ on it. It could as well have been a wastebin on the map, for instance”

Part 3: Areas Most users were successful adding an area, but some did not identify the correct area that was meant to be drawn.

Novice 5 [w_drawarea]: “It wasn’t clear to me which of those areas was the playground... I just clicked in the centre, and then that window showed me where I was supposed to...”

Novice 6 [w_drawarea]: “Well if I knew now what the playground was exactly, it would be easy, but... can I zoom in? No. OK, I suppose that all this would be the playground”; clicks in the upper left of the highlighted area, after the change of focus the point is outside of it; places nodes around the playground; struggles to finish the area as the first node is not clickable any more (Fig. 5.1)

Figure 5.1.: Novice 6 having problems finishing the area for the playground

Some expressed satisfaction about the area drawing function.

> Novice 8 [`w_drawarea`]: “OK, this reminds me of the polygon creation in graphics software”; “it absolutely conforms to my expectations, this function... also finishing the process by clicking on the first node again”
>
> Mapper 8 [`w_drawarea`]: “Had some problems starting the area... I find it intelligent that you have to click on the first node to finish the area, that’s well done. I would have expected that I had to use the right mouse button or press Escape or something. So I like it that way. You just have to know it, because it doesn’t conform to your expectations”

As with the cafe a few steps earlier, users complained that they could not select the playground (German: *Spielplatz*) feature type until they typed something in the search form.

> Novice 6 [`w_selectfeaturetype`]: looking for playground, types ‘sp’; clicks on *Spielplatz* while only the search form is highlighted; “I find it silly that I can search for things here, but the moment it appears, I’m not allowed to click it”
>
> Mapper 4 [`w_selectfeaturetype`]: searches for ‘SP’; clicks on *Spielplatz* while still only the search form is highlighted; “That’s stupid too. Usually two letters are enough and I can click on it, I don’t have to type the whole word ‘Spielplatz’ now, do I?”

Again, the instruction to close the feature editor after the name had been added raised concerns with several test persons.

> Novice 5 [`w_addname`]: “A bit unusual that you don’t have to click on Save, but just close the window”
>
> Novice 7 [`w_addname`]: enters name; clicks ‘+’ button; clicks in name field and presses Enter; “ But it’s done! Why wouldn’t it take it?”
>
> Mapper 6 [`w_addname`]: “What confuses me a bit here is that there is no OK button, that I obviously save it automatically when I close it, that’s odd”

However, some test persons had learned the procedure until then and expressed they were, in fact, comfortable with it.

> Novice 3 [`w_addname`]: “Honestly I like it that the attributes are applied by closing it, because it somehow saves you another key press or click, they just have to explain it better somewhere that it works that way”
>
> Mapper 3 [`w_addname`]: “Now I learned that you don’t have to press Enter or plus or anything, but just close it and it’s fine”

Part 4: Lines As with drawing areas, most users succeeded at the task of drawing a road, but several did not find the correct place to start drawing. Some clicked on existing roads until they realized that the road was supposed to be traced from the imagery. In a few cases they got stuck with this task.

> Novice 1 [`w_drawline`]: "Start the line by clicking on the end of the road... so I guess... here"; clicks on Flower Street at the bottom, creating a node; "ah, so I pressed on the wrong side of the road, now I'm supposed to click Flower Street... I would like to go back, but I can't..."; clicks on Flower Street next to the previous node
>
> Novice 2 [`w_drawline`]: "clicking on the end of the road..."; clicks on the intersection of Flower Street and Elm Street; moves cursor in highlight area and reads tooltip; "but there is no road here"
>
> Novice 7 [`w_drawline`]: "Start the line by clicking on the end of the road... somehow that's a bit hard to recognize"; clicks on the eastern end of Flower Street; then clicks on the northern end of Elm Street; "I'm not getting it"; clicks on the intersection of the two streets; "Start the line by clicking on the end of the road. I'm not seeing an end of a road"; moves cursor into focus area, reads tooltip and clicks somewhere; "Oh!"
>
> Novice 8 [`w_drawline`]: Moves cursor to the eastern end of Flower Street and the northern end of Elm Street; "The end of the road, which one could that be?"; clicks on Flower Street, creating a node; reads new instructions; "obviously I've put the first node wrong, so I'm going to have a problem creating an intersection"; creates another node on Flower Street and continues
>
> Mapper 1 [`w_drawline`]: "Start the line..."; clicks on Flower Street (but nothing happens); "... by clicking on the end of the road. There's nothing happening now... maybe I have to stay in this box"
>
> Mapper 2 [`w_drawline`]: "Start the line by clicking on the end of the road. That would be here..."; clicks on the northern end of Elm Street, then on the intersection; "But there is no road here... yes, that could be one"; first clicks on Flower Street (outside of the highlight), then on the correct end of the road
>
> Mapper 9 [`w_remark`]: points at the instruction window; "I find it a bit strange, I mean, here there's always the small arrow that points from... it's talking to me and when it says, 'Start the line by clicking on the end of the road', I think there is a road where the arrow points to"

Again, users commented on the add-name-and-close-feature-editor workflow.

> Novice 4 [`w_addname`]: "From the Mac I'm somehow used to pressing Enter and then it's gone, but here I still have to close it"
>
> Mapper 4 [`w_addname`]: "I'd really like to always just press Enter there"

Part 5: Start Editing In the conclusion of the walkthrough two text boxes show up for a few seconds, explaining the help and save buttons. For some, the text boxes disappeared too quickly, so they could not finish reading them.

> Mapper 5 [`w_startmapping`]: "More documentation and this walkthrough... now it's gone"

> Mapper 6 [`w_startmapping`]: "More docu... oops! There just was some text that I didn't see"

Other issues A problem that occurred frequently during the walkthrough was that the tooltip floating next to the cursor was often overlaid by the instructions text and therefore not readable.

> Novice 3 [`w_contexthelp`]: clicks *Point* button; "It's a pity that as I hovered over the Point button with the mouse, some pop-up went up and it was under the instructions of the tutorial, I don't know if that can be changed"

> Novice 3 [`w_contexthelp`]: "Unfortunately I can't read what that is"; "I'm confused because that pop-up is under the instructions" (Fig. 5.2)

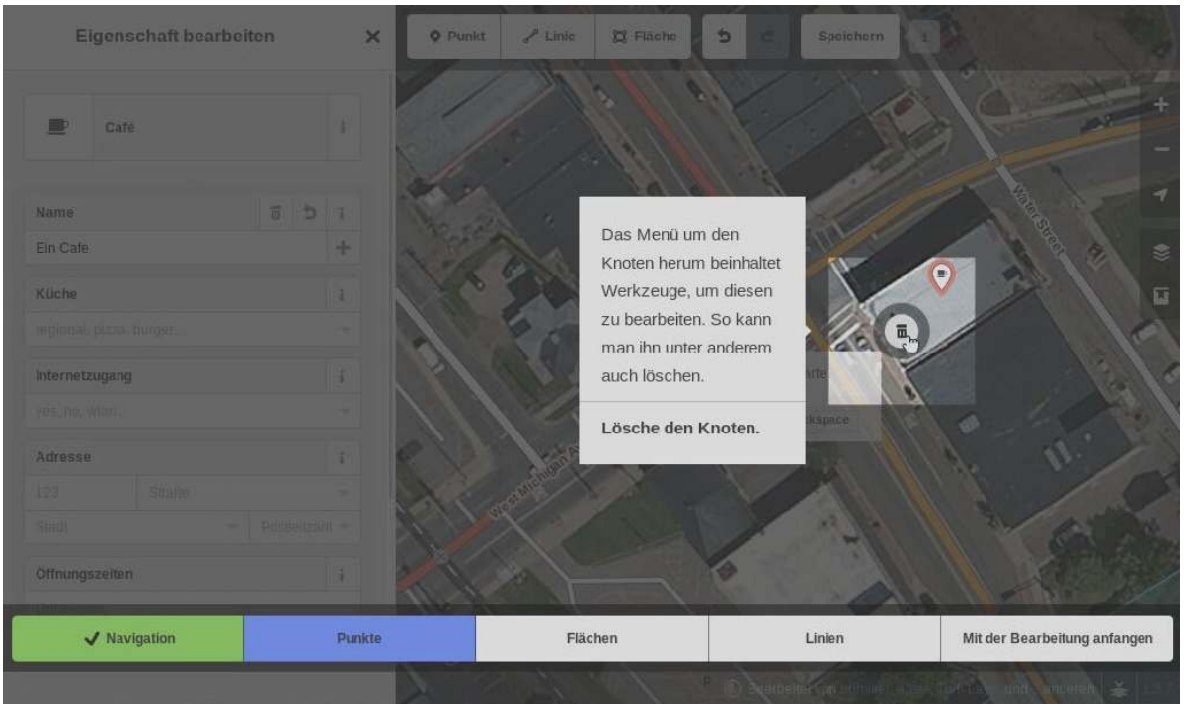


Figure 5.2.: Novice 3 complains that the context help is not readable

> Novice 4 [`w_contexthelp`]: "Now the task somewhat overlays the tooltip"

Frequently the walkthrough task was interrupted by a software bug, often in the form of the highlight window not moving to the place it belonged, or the instructions not appearing. This often prevented the user from proceeding. Altogether, it happened with 8 different users: Novice 2, Novice 6, Novice 7, Novice 8, Novice 9, Mapper 1, Mapper 4, Mapper 9. In this case I interrupted the test in order to reset the walkthrough to the beginning of the previous step so that the task could be continued.

5.2.2. Navigation

Apart from the walkthrough task, which has now been covered, various issues arose during the remaining tasks. As these issues appeared across tasks, the corresponding observations are structured by their codes given in Table 5.1, starting with map navigation.

Panning the map could result in moving nodes accidentally (which is achieved by dragging and dropping—unlike moving lines and areas, which is a function accessible from the circular menu). Also, users were not always aware of when and how the map could be panned.

> Novice 6 [`pan`]: accidentally moves a node in an attempt to drag the map; "What have I done now? Did I... I've pulled that marker away"

One user wondered how the map could be panned, and whether it worked in drawing mode (although this had been covered in the walkthrough).

> Novice 4 [`pan`]: "I'm wondering if I can scroll with the keys..."; pans the map with the arrow buttons; "what happens when I'm in area mode and try to scroll by dragging and dropping? It should actually make the lines then... no, it scrolls! Good."

Zooming the map with the mouse wheel was too slow for several test persons:

> Mapper 6 [`zoom`]: "With the scroll wheel, is the computer somewhat lagging or isn't the scroll wheel as I'm used to it?"

> Mapper 8 [`zoom`]: "The plus button enlarged the frame much more than the scroll wheel does... I don't know if it was coincidence, but it's just the size I wanted"

> Mapper 9 [`zoom`]: zooms with mouse wheel; "The zoom is being much too slow for me"

Some users discovered that it was also possible to zoom the map by double-clicking (not mentioned in the walkthrough).

> Novice 4 [`zoom`]: double-clicks; "Oh, zooming by double-click works too. Nice."

5.2.3. Save

There was mixed feedback about the editor's save functionality. Some were pleased by the indication of what has been changed during the mapping session.

> Novice 3 [save]: "Oh, that's good: it shows me the changes and obviously I can also hover over it with the mouse and then it shows me... yeah, that is nice..."

> Mapper 8 [save]: points at the number next to the save button; "I like that it highlights the changes, that I can see I have eleven changes"; clicks *Save*; enters commit message; "oh, there is a list of changes. I'll have a look at them"

> Mapper 8 [save]: "Field paper exercise... nice that it already applied the comment. I'll check my list of changes again... I added the area, removed a point... it's great that there are symbols there, I miss that in some places"

While Mapper 8 liked that the previous commit message was preserved, Mapper 4 did not.

> Mapper 4 [save]: wants to change the commit message; "I would like it if the commit message was removed each time"

One complained that the grey text in the empty commit message field was not selectable.

> Novice 6 [save]: "Here, in the fields in which it's supposed to be typed in, there's always such a grey background. I would expect, because elsewhere it works that way too, that I can mark or delete or cut and paste it or whatever, that I can simply write over it isn't familiar to me at least from other software products... so that's what I tend to do usually, first go to the start and drag over it so I can select it"

One mapper had recurring problems using the keyboard shortcut for save, Ctrl-S.

> Mapper 4 [save]: types commit message and presses Ctrl-S, browser's save dialogue pops up; "No, I didn't want that"

> Mapper 4 [save]: reads tooltip of the top right save button, presses Ctrl-S, enters commit message; "That's bugging me too that Control-S doesn't work here either. That I really have to click on it, that's silly"

> Mapper 4 [save]: presses Ctrl-S while the cursor is placed in the feature editor, the browser's save dialogue pops up, tries again; "It wouldn't save there again... for whatever reason"

> Mapper 4 [save]: changes a feature's name and hovers over the save button, but it is greyed out; "Huh? Why no changes to save?"; clicks on the button anyway and gets to the commit message (Fig. 5.3)

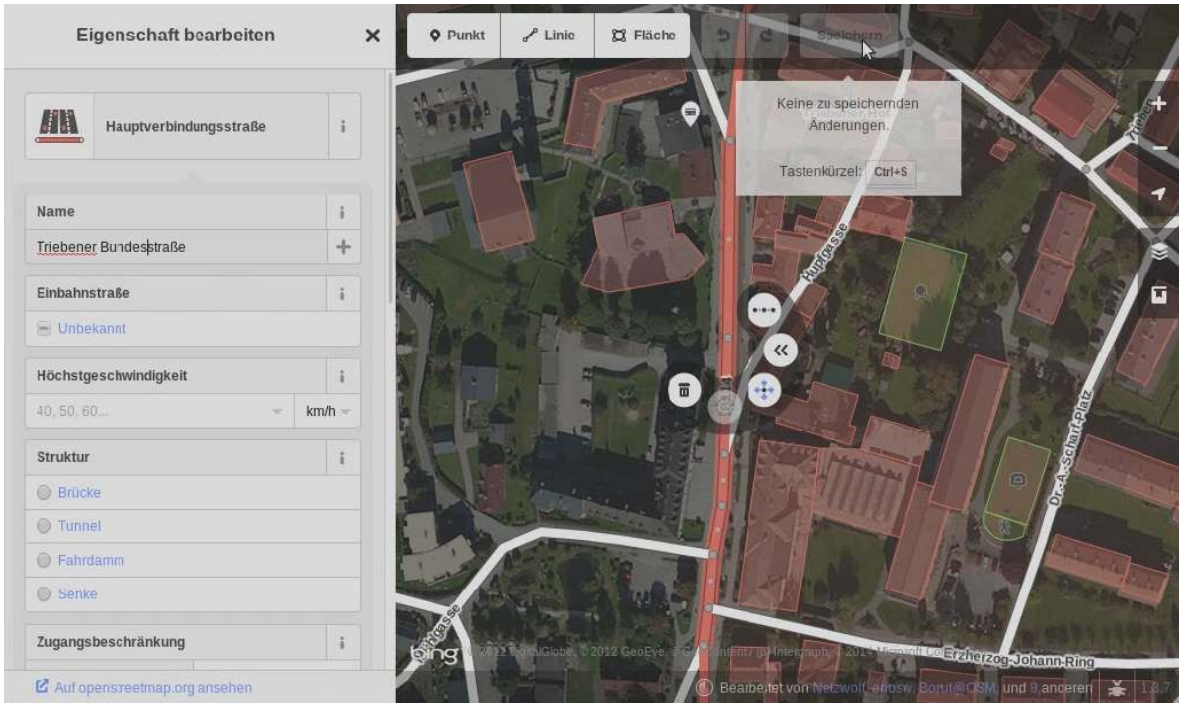


Figure 5.3.: Mapper 4 wants to save the edits, but the editor says there is nothing to save

5.2.4. Search

The search function that is built into the *iD* editor has not been found by all users, so some of them closed the editor and used the main page's search function instead.

> Novice 3 [`search`]: wants to search for Trieben and uses the browser's back button to leave the editor and use the search function of the OSM main page.
>
> Mapper 7 [`search`]: goes back to the main page and uses the search function there
>
> Mapper 8 [`search`]: "So then I'll search... the question is how. I don't see a magnifier"; leaves the editor and uses the search function on the main page

If users did use the editor's search function, they often had trouble spotting the searched feature on the map, because it was not always clearly visible.

> Novice 3 [`search`]: uses *iD* search to search for 'Hauptplatz'; "I'm going to search, maybe Hauptplatz is included... No results in visible map area"; zooms out; finds Hauptplatz; "There is Hauptplatz... OK, it's been marked nicely, has a nice red edging, I'd find it nice if it would also move a little bit to the centre of the map so you can find it directly"
>
> Mapper 3 [`search`]: searches for Martin-Luther-Platz and clicks on the result, but doesn't find the highlighted object, which appears very small and not centred

Mapper 4 [`search`]: searches for Martin-Luther-Platz and clicks on the result; “And now where is it? Hello? Why don’t I see it now? Wouldn’t it be nice if it showed up in some loud colour now... I mean, it must be there somewhere, but again there a those stupid things” [referring to the circular menu]

Mapper 5 [`search`]: searches for Martin-Luther-Platz, zooms out, and clicks on the search result; the highlighted street is located at the very edge of the map area; “So where is it?”; finds it; “That could have been centred” (Fig. 5.4)

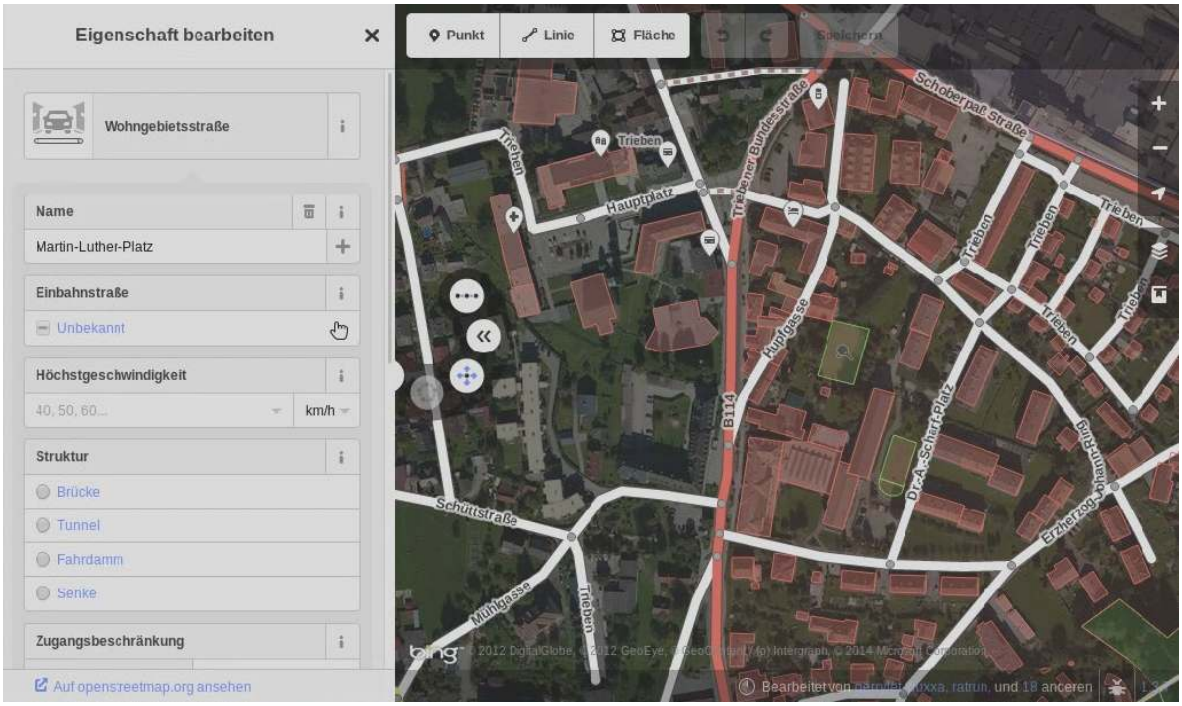


Figure 5.4.: Mapper 5 has selected Martin-Luther-Platz from the search, but it is hardly visible (on the left edge of the map window)

Mapper 9 [`search`]: “Earlier I had that... oh, wait, there you can search”; searches for Martin-Luther-Platz; “I assume now that it searches automatically because down there it says: ‘No results in visible map area’”; clicks on ‘Search worldwide’ and finds it

In one instance, the search function simply did not work as expected:

Novice 2 [`search`]: uses *iD* search function to find the town Trieben, although the editor already showed that area; search for ‘trieben österreich’ does not yield a result.

One user noticed that the search results were not translated, although the interface language was set to German.

> Novice 6 [search]: searches for 'Trieben Österreich', but no result shows up, clicks on 'Search worldwide'; "Now I've got into the English version somehow... Styria, Austria, town"

5.2.5. Visual elements

This section covers comments by the participants about various visual elements of the map, such as the circular menu, object highlighting, or map labels.

One mapper expressed dissatisfaction with the circular menu, because it showed options that he found useless in the given context.

> Mapper 7 [circularmenu]: "I find it adverse that the options don't fit the object. You'll most rarely want to make a building circular. So in that case the editor could already say, at least I don't put it directly on top of the object, but maybe to the side, just for the case that you do have a circular building"

One mapper made a remark about the red highlighting of selecting objects while a primary road was selected.

> Mapper 5 [highlight]: "That red marking around the red road is a bit hard to recognize..."

One user complained that the main pane could not be closed to enlarge the map view.

> Mapper 6 [mainpane]: closes the save dialogue on the main pane; "Now I hoped that I'd get more picture here on the left side, but that doesn't work. (...) So this window on the left obviously always stays open"

One user was missing map labels on newly created objects.

> Novice 7 [maplabels]: "It's mean that it doesn't read 'Billa' now"

One test person was confused that some tags were not translated into German.

> Novice 4 [translations]: inspects dropdown menu of the surface tag; "It is somewhat confusing that some things are translated and others aren't"

5.2.6. Selecting objects

Any object can be selected by clicking on it in the map window (except when the editor is in drawing mode). However, some users had difficulties selecting a point by clicking on it. They often moved the point instead of selecting it because they moved the mouse a little bit while they pressed the left mouse button—possibly caused by unfamiliarity with the hardware.

Novice 2 [w_selectpoint]: clicks point, moving it (three times); "And what now?"; double-clicks point

Novice 3 [selectpoint]: clicks on point, moving it; "Oops, didn't take it, so... click..."; selects point

Novice 6 [w_selectpoint]: clicks point, moving it; "Now I have moved it"; moves it again (two times); "do I have to double-click?"; double-clicks with success

Novice 8 [w_selectpoint]: clicks point, moving it; then selects it successfully

Mapper 2 [w_selectpoint]: clicks point, moving it; "Doesn't take it... now tell me, why does nothing happen?"

Mapper 2 [w_selectpoint]: clicks point, moving it; in the third attempt moves it outside of the focus area, making it impossible to select it

Mapper 4 [w_selectpoint]: clicks point, moving it (three times)

Mapper 6 [w_selectpoint]: clicks point, moving it (three times); "Ah, now I seem to move something here already"; succeeds with double-click

Several mapper users were dissatisfied that although multiple objects could be selected by holding the Shift button, they could not be edited in a single step.

Mapper 3 [selectmultiple]: "I'll try with Shift now and the second line... now I have two... OK, good, but I can't edit the two objects at the same time now"

Mapper 9 [selectmultiple]: selects two parts of a road with Shift-click; inspects the list of objects in the main pane; "I'd actually like to edit both of them. I'll click on the upper one..."

They also had to guess how multiple objects could be selected, as this had not been covered by the walkthrough.

Mapper 8 [selectmultiple]: "I have the impression that the two are identical and I'd try to make them one now... Control-click... doesn't work, to select both... Shift... yeah, Shift works"

5.2.7. Selecting a feature type

A major issue test persons complained about was the procedure of selecting the feature type of a newly created object, as well as changing an existing object's feature type. First, some users often did not find the correct feature types because they simply did not use the search function.

> Novice 2 [`selectfeaturetype`]: scrolls the list of feature types, not using the search function; "well, which one could this come under"; selects *Geschäftshaus* (commercial building)
>
> Novice 2 [`selectfeaturetype`]: "Hmm... then I'll just call it a path"; clicks on *Path*, then on *Footway*
>
> Novice 6 [`selectfeaturetype`]: "Well, now how can I add attributes to this area? Because I want to describe somehow that it's used as car parking"; clicks *Land Use* and scrolls down; selects Area; "Area... no, that's silly"
>
> Novice 6 [`selectfeaturetype`]: selects *Point* and adds name 'Telefonzelle' (phone booth); "It doesn't really satisfy me, but..."
>
> Novice 6 [`selectfeaturetype`]: clicks in search form; "Ah, I could have searched for it there maybe"; finds *Chemist*; "Ah, one learns something new"
>
> Novice 7 [`selectfeaturetype`]: doesn't use the search function; expands Land Use but ignores it; scrolls down the list and selects *Area*
>
> Novice 7 [`selectfeaturetype`]: selects *Line* (for footway)

Second, they sometimes did not find the feature type they were looking for, because they did not search for the correct terms, or simply because the feature type they had in mind did not exist.

> Novice 1 [`selectfeaturetype`]: "This is a square"; types 'square'; "not there, hmm"; types 'platz'; "not there..."; scrolls down the list; "it's not a place of worship... maybe it's land use..."; clicks on *Land Use*; "not residential, not industrial, not a farm, not a cemetery... so it's not land use"; collapses *Land Use*; scrolls up and down; selects *Area*
>
> Novice 4 [`selectfeaturetype`]: types 'verkehrs' in feature type search; "Oh, there is no traffic island, what a pity"; "OK, how could we... it's a little green area"; searches for 'grün'; "Green tree, grass, garden, ditch, gravel... that isn't it really..."; "no, that's bad, if I can't name it, I don't want it"
>
> Novice 8 [`selectfeaturetype`]: searches for 'Acker'; searches for 'Land'; "I was thinking of agricultural land. Land use sounds similar, or is that something different?"; clicks the 'i' next to *Land Use* and goes to the wiki page

Third, several users were frustrated because the search function did not always work very well.

> Mapper 2 [`selectfeaturetype`]: "Let's see if there's parking *(Parkplatz)*"; searches for 'parkp'; "no, doesn't exist..."; "Why is there no parking?"; searches for 'auto' and selects *Autoparkplatz (Car Parking)*
>
> Mapper 3 [`selectfeaturetype`]: types 'park'; "It's not there..."; types 'parkp'; then types 'auto' and finds *Autoparkplatz*; "OK, 'Parkplatz' doesn't work, but 'Autoparkplatz' does"
>
> Mapper 4 [`selectfeaturetype`]: types 'pa'; types 'par' and scrolls down the list; types 'park'; "Do I really always have to type the whole word, that's annoying"; types 'parkplatz' and selects *Autoparkplatz*
>
> Mapper 4 [`selectfeaturetype`]: types 'sup' (*Supermarket* appears at top of the list); "Look, now it works after three letters and for the parking it didn't work, isn't that silly"
>
> Mapper 8 [`selectfeaturetype`]: searches for 'park'; "'park' doesn't find it"; types 'auto'; "with 'auto'... Autoparkplatz"

Fourth, several users did not understand the feature type category buttons and were confused by them.

> Novice 2 [`selectfeaturetype`]: "Well, what does a parking come under... place of worship..."; scrolls up and down the list; "under Land Use..."; clicks on *Land Use*; clicks again (two times); double-clicks; clicks (two times); "hmm, well..."; clicks (four times); "I'd say I have selected it... but nothing happens"; clicks many more times, clicks in the map, then goes back to the feature type selection; "... or is Land Use some kind of super-category?"; scrolls down and selects *Area*
>
> Novice 3 [`selectfeaturetype`]: clicks on *Building* (category); "Now obviously something has opened up there... it had been hinted at that something would open up, because this is sort of a stack... and then I click on House"
>
> Novice 6 [`selectfeaturetype`]: "Just as a building, right?"; clicks on *Building* (category), but doesn't see that it expands; clicks on *Hotel*, goes back to feature type selection; "I don't get this again: The hotel was accepted when I clicked it, the building is not accepted. Ah, maybe because there's more behind it, is that another selection?"; searches for building and selects *Building* (singular); "Well, when I click on Building now, it is accepted anyway, so that's really weird"
>
> Novice 8 [`selectfeaturetype`]: "Right now I'm wondering why it isn't accepted or doesn't call the editor to action, when I click on Land Use..."

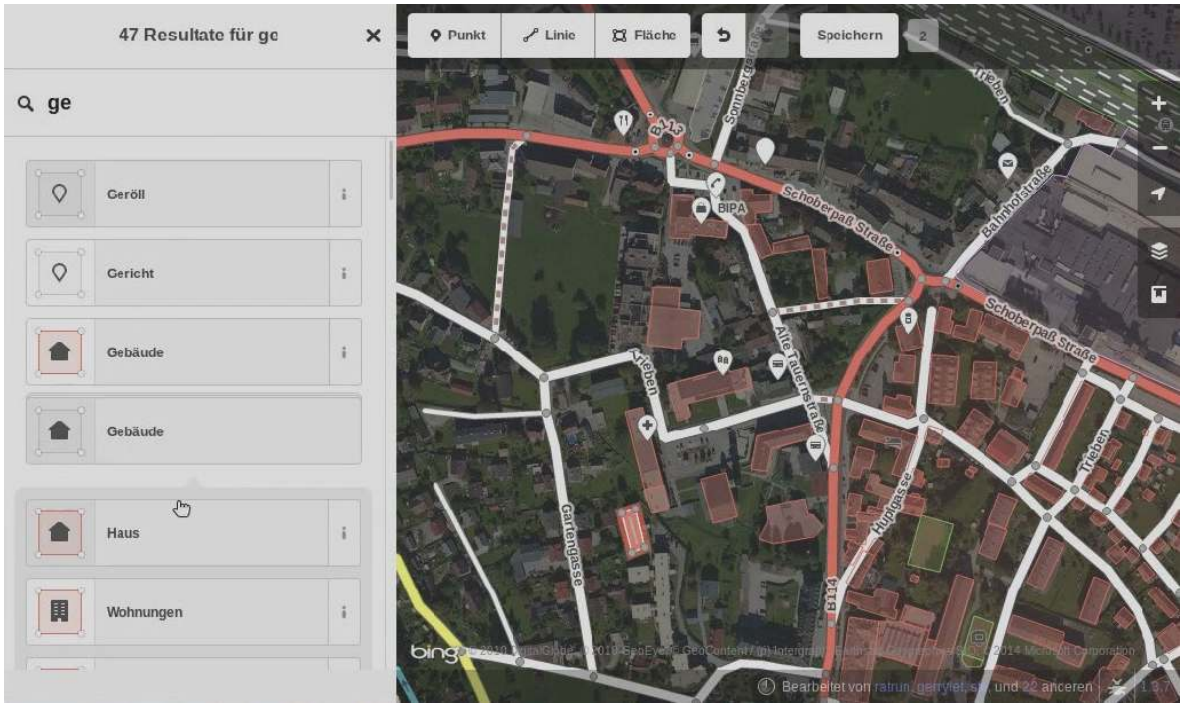


Figure 5.5.: Mapper 2 takes a look at the *Building* feature types (singular and category)

> Mapper 2 [`selectfeaturetype`]: draws building; clicks on the *Building* category (several times), then clicks on the singular *Building* type (Fig. 5.5)

> Mapper 5 [`selectfeaturetype`]: searches for building, the *Building* feature type appears once as a category and once as a singular type; "Why are there several suggestions now? One is Building and... that looks like a collection, looks as if there were several cards on top of each other, what happens if I click it?"; clicks on Building category; "ah, a House, and so on... I can't tell that, it's just a building"

> Mapper 9 [`selectfeaturetype`]: searches for building; "Now there are two different... apparently there are several in there, I'll just click there"; clicks on *Building* category; "That's not so intuitive that this is the list although up here there is, very vaguely, a grey... arrow "

Lastly, a few users were looking for a feature type for which they had the actual tags in mind. They missed a possibility to select the feature type directly by its tag.

> Mapper 7 [`selectfeaturetype`]: "Highway=service... now of course I'd have to know the German equivalent that somebody has defined..."

> Mapper 7 [`selectfeaturetype`]: "This is highway=service, service=parking_aisle... in that case... can I, directly on an object... Road. It is a road anyway"; clicks *Road* category and scrolls down the list; "Service road..."

Mapper 9 [`selectfeaturetype`]: searches for 'service'; "Tag won't work, because it's in German... I'll check if I find it, driveway or something... drive... driveway"; clicks on the 'i' and checks the wiki

As a final remark on the feature type selection, one user was pleased that previously used feature types were shown on top of the list.

Novice 3 [`selectfeaturetype`]: "Ah, at least it remembers what I've selected last time, I like that"

When test persons needed to change the feature type of an existing object, they did not always find an easy way to do it (which is by clicking on the feature type header in the feature editor while the object is selected).

Novice 3 [`changefeaturetype`]: "... I'll just try if I can somehow assign a category to the area... so this is a building, I select it by clicking on it..."; "err, probably this is not going to be so easy to assign that building the attribute 'supermarket'. That's a pity"

Novice 3 [`changefeaturetype`]: "Well, there... that should work like with the supermarket before I suppose"; enters name; clicks in *Building* field; "Building, yes... house, residential, I'll type in 'hotel'"; types 'hotel'

Novice 4 [`changefeaturetype`]: "So presumably this building is the BIPA chemist"; clicks on the building; adds name; clicks on feature type and changes it to *Chemist*

Novice 4 [`changefeaturetype`]: "This building should be the hotel Triebenerhof... that is, it's name is Triebenerhof..."; adds name 'Triebenerhof'; clicks on *Building* feature type; "and the building is a hotel"; searches for hotel; "OK, now what's the difference between a hotel and a hotel building?"

Novice 6 [`changefeaturetype`]: "So I've realized now that using the search function I can find additional categories that are not there in the list, because those apparently are just the most recently used or whatever. Hence, I now want to change the tagging of the point that I previously added as a point with the name 'Telefonzelle', but I currently don't know how I can get back to that search thing for building types and point types"; deletes the point and creates a new one

Mapper 1 [`changefeaturetype`]: wants to tag a building as chemist; scrolls down feature editor; "Can you add something here?"; clicks on the '+' below All relations; "no, that's wrong"; scrolls up and clicks on the 'i' next to the feature type header; "OK, I'll stick with point"; places a point on top of the building

Mapper 1 [changefeaturetype]: selects building to tag it as a hotel; "Just see how it was done with the bank"; selects the bank; goes back to the building, clicks on the feature type header and changes it to *Hotel*

Mapper 2 [changefeaturetype]: selects building to tag it as a hotel; inspects the circular menu; clicks on the 'i' next to the feature type header; then clicks on the header itself and changes the feature type

Mapper 3 [changefeaturetype]: selects building; "From my experience I know that you can mark such an object as a hotel, too"; inspects main pane; "but somehow that isn't being offered. I'm stuck there now, so I'll rather put a point"

Mapper 4 [changefeaturetype]: selects building to tag it as a hotel; enters name; clicks in the *Building* field and inspects the drop-down; "That's what I hate about tagging buildings, there never is anything that makes sense..."; chooses *Commercial building*

Mapper 5 [changefeaturetype]: "Now, can I change the feature type of this building? It is a building, but I want to put it as chemist"; scrolls down the main pane; clicks on feature type; "if I click 'Change feature' now..."; changes it to *Chemist* and checks if the building tag has been preserved

Mapper 7 [changefeaturetype]: "Building... Name, Building: yes, house, residential"; clicks in *Building* field and reads suggestions; "Are those now tags it suggests or are they the tags that it has? No, those things are suggestions. Because 'yes' plus 'house' plus 'residential' would be too much"; types 'hotel', which is not suggested, aborts and checks the wiki

One mapper was also uncomfortable with the fact that each feature had to have a "primary" type, while, in fact, any object in OSM has multiple, equivalent tags.

Mapper 9 [changefeaturetype]: selects building to tag it as a hotel; clicks on the feature type header and searches for hotel; inspects feature editor again; "Yes, it's a building... strange, earlier there was no 'yes' here, now there is a 'yes', although I've actually just changed it from Building to Hotel, but Hotel seems to imply that it is a building. It's done right, but I still find it weird that up here... I mean that there is a main tag for everything"

5.2.8. Basic editing

The fact that a 'Save' or 'OK' button was missing in the feature editor (see also remarks about the walkthrough in section 5.2.1) was mentioned again:

Novice 1 [closefeatureeditor]: "Now again I don't know if it has been saved or not..."

Novice 4 [`closefeatureeditor`]: "That you have to close it in order to save is a bit unusual"

Mapper 2 [`closefeatureeditor`]: "What's always annoying me so much, to be honest, is that there is no confirm button. So I don't know, if I click here, is that right or not"

The editing of feature attributes within the feature editor raised various issues.

Mapper 2 [`editattributes`]: clicks in the *Capacity* field of the car parking; clicks on the up arrow to increase the number; "This is boring"; goes on clicking; "do I have to click up to 100 now? I'm too lazy for that, I'll only put 50. Or can you type it in here?"; types the number in the field

Mapper 9 [`editattributes`]: edits driveway; "Access, General, OK... suppose it means that anyone can walk there"; reads suggestions for *General*; "now a list has come up, 'yes' was already there, probably as default value... but I can't see what 'yes' means, I can only see, what the other mean... I'll just choose 'no'"; clicks 'no'; "is there now 'yes' in there? No... because 'yes' isn't in the list, I'll put it back in"; types 'yes'; "oh, now all are 'yes', oh well. That's shit. (...) I guess I'll leave it there, because I don't know it... oh, I can delete it too"; deletes tag by clicking the Delete button; "okay... OK, it was all grey, that's probably why the value wasn't filled in, it shows whatever the person who wrote the editor considered as the defaults..." (Fig. 5.6)

Adding an address to the bank (in task 2), users frequently commented on the arrangement of the street name and house number fields (house number first), which is uncommon in Germany.

Novice 4 [`address`]: "First the house number? That's unusual"

Novice 6 [`address`]: "Argh, I hate those American notations"

Mapper 2 [`address`]: "The house number, that's silly..."

Mapper 5 [`address`]: "That's not so nice of course that, for German circumstances, the house number comes first"

Mapper 6 [`address`]: "Ah, very American, the house number being first, oh well"

Satisfaction was expressed about the fact that drop-down menus were offered for the address fields.

Novice 3 [`address`]: "Town... there is a drop-down menu, that's good"

Mapper 8 [`address`]: "That seems to be the house number, the first one"; clicks in street field; "it's nice that I can choose it, I totally like that"

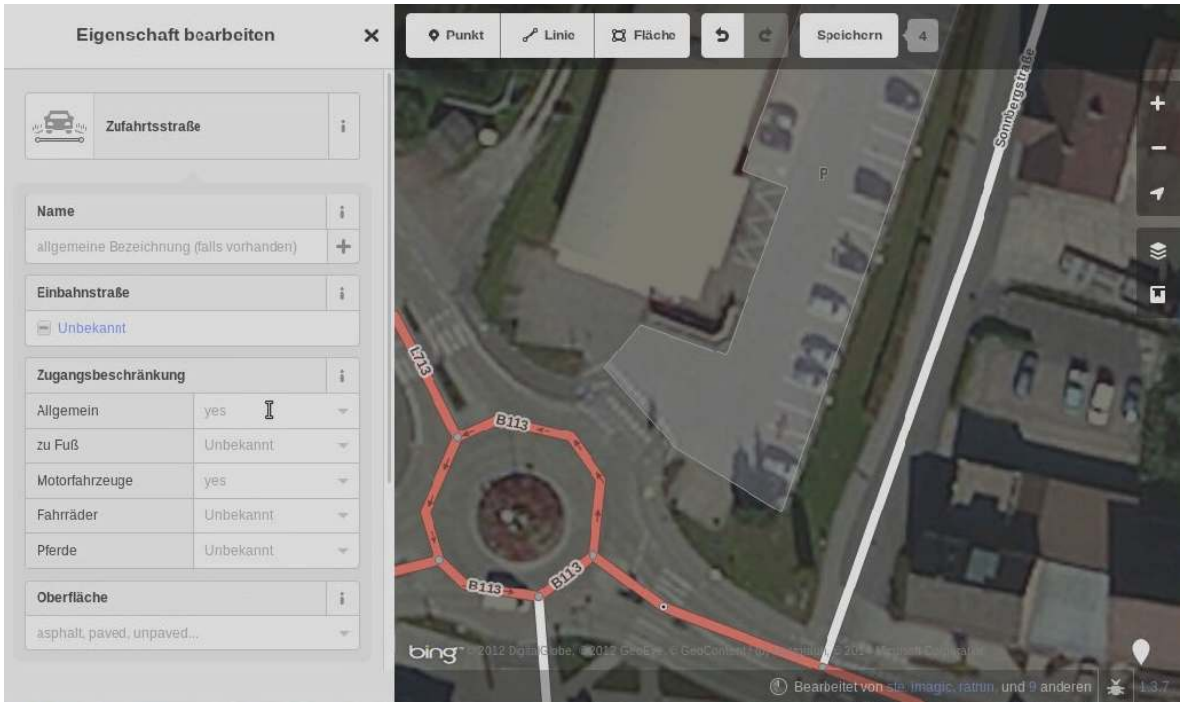


Figure 5.6.: Mapper 9 explores the access tags

> Mapper 9 [`address`]: "There, American notation, that's a bit weird of course"; goes to street field; "so, house number is 2, street is... cool, it suggest where it would be. That's awesome, really good"

While most users eventually succeeded in adding a phone number, not all found the easy way through the phone icon in the attribute icons panel.

> Novice 3 [`phonenumber`]: "... then I guess we're going to do it just like before, because there is no available field for phone..."; checks the OSM wiki about tagging phone numbers; then enters "contact:phone=03615 22170"

> Novice 6 [`phonenumber`]: "Well then, how can I enter the phone number there?"; clicks on *All tags*; closes *All tags*; finds the phone symbol

> Mapper 4 [`phonenumber`]: clicks on *All tags*; opens a new field, types 'te' and checks the suggestions; types 'add' and checks suggestions; "Is there no phone?"; scrolls up; "Is there a general plus anywhere?"; finds phone icon

> Mapper 6 [`phonenumber`]: "Phone... I don't know how to go on now... can you scroll here? Ah, yes, you can scroll here, you just don't see it. But still there's no phone. Yes, there"

> Mapper 8 [`phonenumber`]: "Phone. That means I have to add another field..."; searches; "ah, OK. I was actually looking for kind of a plus sign in order to add another field to the list, but if there extra symbols for it down there, that's fine. I'm just wondering about the ones that aren't there"

Mapper 9 [`phonenumber`]: "Down here we have... it is still open, still expanded from earlier, where the tags are displayed... I'll just check if the phone number could be entered anywhere up there... no. Then I'll have to enter a new one"; types 'phone' and checks wiki to clarify usage of 'phone' and 'contact:phone'

Some users, mostly mappers, made use of the 'All tags' section of the feature editor, either to check the tagging or to change it.

Novice 3 [`alltags`]: uses the *All tags* editor to enter a phone number: "... so I'll enter it manually, and do not choose any one from the list, that's called contact:phone... it suggests it to me, I don't know if someone else has done it or if it's some sort of autocomplete functionality, which I'd like very much, of course..."

Mapper 1 [`alltags`]: checks *All tags* (on supermarket); "What tags do we have there? Retail, that's fine"

Mapper 5 [`alltags`]: checks All tags (on driveway): "Let's see how it does that... service, service=driveway, fine"

Mapper 7 [`alltags`]: "So now I'm looking for a way to define a new tag freely... at this point I would usually resort to... I have a plus sign here, but it sits under Name, I'll click on it... multilingual, no, I didn't want that... err... All tags... All relations..."; clicks on *All tags*; "Ha, got it"

Mapper 9 [`alltags`]: "So I'm looking for... I just check how it's called... operator I guess... yeah, operator... so auppose I also wanted to add BIPA as an operator, then I'm a bit clueless, where that would be"; clicks '+' next to Name; "I can add another name here... All tags... OK"

One mapper wanted to analyse the geometry of a road he had drawn.

Mapper 8 [`inspectfeature`]: draws footway; in the feature type search, clicks on 'i' next to Line; "I click on the 'i' in the hope that it's going to help in some way... no documentation"; clicks on Line; "well, I've got the feeling that I can't find out of how many elements this line consists... I'd probably now try to save the whole thing and look at it in JOSM"

One mapper made a remark about the residential area whose shading affected the visibility of the background imagery.

Mapper 5 [`residentialarea`]: "What's somehow bad is that seemingly there is sort of a grey layer on top of everything, probably from the residential area - I always have this kind of thing deactivated in JOSM, it's annoying because you can recognize so little on the aerial photo"

5.2.9. Editing of lines and areas

The line drawing functionality was picked up quickly by the majority of test persons. In a few cases, however, it was not clear to the test persons how the line could be finished (although this had been covered in the walkthrough).

> Novice 3 [`drawline`]: "How to end it, with double-click, I'll try with double-click... obviously works with double"

> Novice 7 [`drawline`]: creates first and second node of the footway, but doesn't finish it; clicks on first node without success; clicks on last node, finishing it

> Mapper 8 [`drawline`]: tries to draw a line by dragging; "First click"; places two nodes and clicks on the first node again; "Hm, it needs getting used to. I didn't know how to finish the line. Now I went back to the beginning, because it's simliar to the field, but that doesn't work. Apparently I have to go on the end"

Drawing areas was, like drawing lines, convenient for most users. One problem that occurred with areas was that clicking on the node previous to the last one (or similar cases where continuing the line would not make sense) was prohibited by the editor. However, no feedback was given about this limitation, which created confusion with some test persons.

> Novice 7 [`drawarea`]: tries to finish an area by clicking on a point which is not the first one (nor the last); "Take it! ... It's being mean to me! ... I don't understand why it doesn't put the point here now" (Fig. 5.7)

What affected the users' workflow negatively in some cases was that when the undo button was used to revert the last node, the editor cancelled the drawing mode, making it impossible to resume drawing the area.

> Novice 7 [`drawarea`]: "Now I don't know how to undo it"; finds and clicks *Undo*; fails to resume drawing and starts all over again

> Novice 8 [`drawarea`]: aborts drawing an area and deletes the last node in order to correct a mistake; fails to resume drawing, deletes all nodes and starts over again

Some users wondered whether areas would be automatically connected with adjacent lines and areas.

> Mapper 4 [`drawarea`]: draws area, placing a node on a road; "And why does it automatically connect with the road that I haven't clicked?"

> Mapper 5 [`drawarea`]: draws an area adjacent to another one; "Now I'd like to draw the supermarket here without connecting the lines... that's not going to work like that"; connects areas anyway

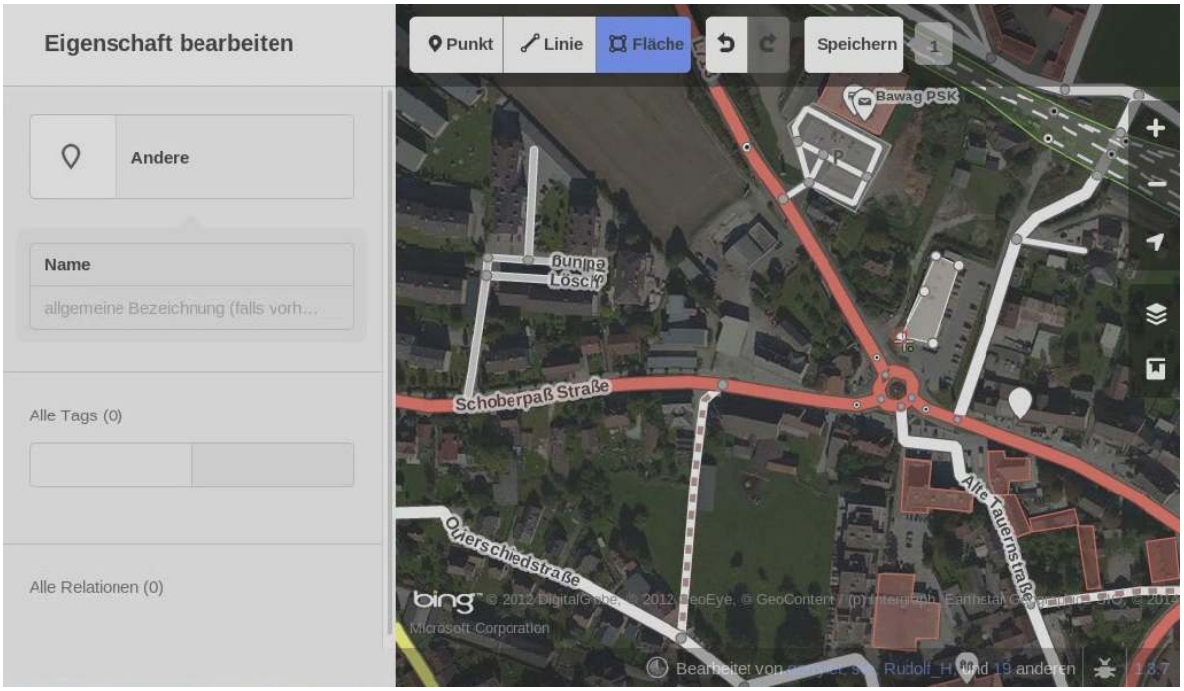


Figure 5.7.: Novice 7 wants to finish an area, but it does not work

In one case a test user attempted to place a node in a place covered by the header bar on the top of the editor window, which was not possible.

> Mapper 6 [`drawarea`]: starts drawing area, tries to place a node in the space behind the header bar (Point/Line/Area buttons etc.); "OK, now I've got something where I don't understand what happens. Well, now I guess I've found a problem here, right?"; places a node somehere else; "OK, that one was wrong"; finishes area and clicks Undo; starts again, having the same problem; "in JOSM I could now drag and move it out, but that doesn't seem to work here"; undoes the area and starts again; "ah, there's the menu up there, that's why I can't go there. It's not easy to recognize that apparently I'm not allowed to click there" (Fig. 5.8)

While creating lines and areas did not trouble the test persons very much, many had significant issues with modifying existing lines and areas, most notably moving and adding nodes. In an attempt to move a node of an existing line or area, some users looked for a button or other type of interaction than simply dragging the node.

> Novice 1 [`movewaynode`]: "I'm gonna make the building bigger. So I select it... and now I have a menu. I don't want it circular, I don't want to move it, I don't want to square the corners..."; clicks the Move button and moves the building; "oh, no, the whole thing is moving, I just wanted to move a point"; drops building; "I click on the corner again... now I have a better chance. I now have another option... to split, to cut, disconnect... no, I just want to move it"; succeeds by dragging and dropping the node

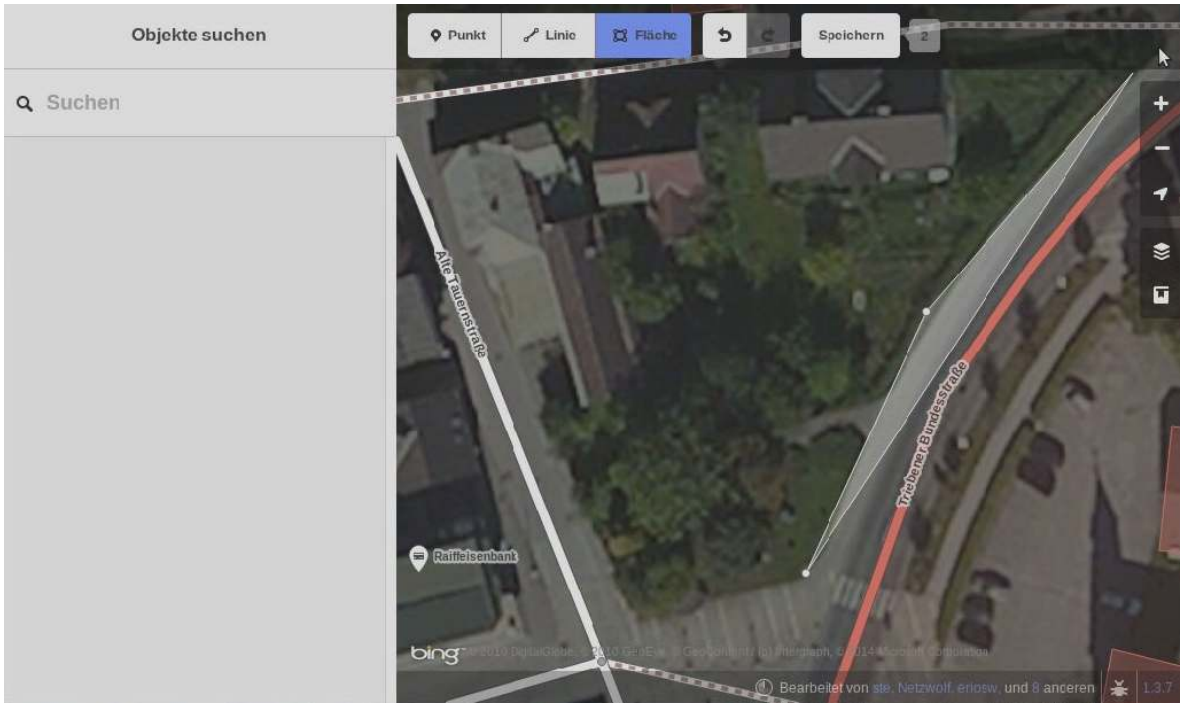


Figure 5.8.: Mapper 6 trying to draw an area where it is not possible

Novice 2 [`movewaynode`]: selects building; inspects circular menu; switches to area drawing mode, no success; inspects circular menu again; considers deleting and redrawing the object; eventually succeeds dragging nodes

Novice 9 [`movewaynode`]: clicks on node and then on 'Cut'; moves node, but ignores that the area has been cut in two

Mapper 1 [`movewaynode`]: "How can I grab it?"; clicks the *Area* button and tries to move a node in drawing mode

Mapper 2 [`movewaynode`]: tries to move a node while the area is selected; misses the node, which results in the map being dragged; then tries again while the area is not selected any more, not working either; inspects the circular menu (of the area)

Mapper 9 [`movewaynode`]: selects node; inspects menu; then moves node by dragging it

Sometimes users simply lacked cursor precision and missed the node they wanted to move.

Novice 7 [`movewaynode`]: "Why can't I just grab the point and move it?"

Mapper 6 [`movewaynode`]: "Alright, now make this longer, don't know if that is going to work the way I think"; tries to drag the node but misses it and moves the map

One mapper wanted to move an edge of an area rather than individual nodes.

> Mapper 7 [`movewaynode`]: "I'd like to grab the flank now and just pull it away"; right-clicks; "right mouse button? Has nothing to do with the editor"

Not only moving nodes raised some issues, also adding nodes presented a major challenge to several test persons. There are two ways to add nodes to lines or areas: by double-clicking on an edge, and by dragging and dropping the "virtual nodes" signified by grey dots and located in the middle of two existing nodes. But these options were not always obvious; instead the circular menu was often (and erroneously) believed to be useful.

> Novice 1 [`addwaynode`]: "Here there should be another point, so I right-click... no, nothing happend with right-click... I'm gonna try to disconnect... I click the node and... hm, split the boundary or disconnect? Disconnect. So, now that it's supposed to be disconnected, I move it"; moves the disconnected node away from its original place, leaving a gap in the area; "... and now I want to close it again, so..."; clicks end point; clicks on "Continue this line" and closes the area; "oops, right-click doesn't cancel... and now we have a line that I don't want"; clicks to continue the line, cancels it by pressing Backspace; selects the superfluous point and deletes it

> Novice 1 [`addwaynode`]: "and now I'm gonna add a point... I click on a node in the hopes of adding another point, but nothing happens. I click on the line..."; inspects menu; double-clicks on line creating a node; "oh, double-click works, good" (Fig. 5.9)

> Novice 5 [`addwaynode`]: clicks on the edge of an area and inspects menu; "No, I don't want to make it circular, I don't want to make it rectangular... double-click"; adds node by double-clicking

> Novice 6 [`addwaynode`]: "I clicked on a point here and then on the scissors... but the effect of it I don't understand"

> Mapper 2 [`addwaynode`]: enters area drawing mode and clicks on an area's edge; draws a new, adjacent area instead of extruding the other

> Mapper 3 [`addwaynode`]: "Here I'd actually like to add another point in order to extend this one"; clicks on edge and inspects the circular menu; "Rotate... square corners... move... no, that is not it... I'm clueless now how I could place another point in that line"

> Mapper 4 [`addwaynode`]: clicks the *Point* button and places a node on the way, a point marker appears; "No, that was not the point"; clicks *Undo*; then uses virtual nodes to correct the way

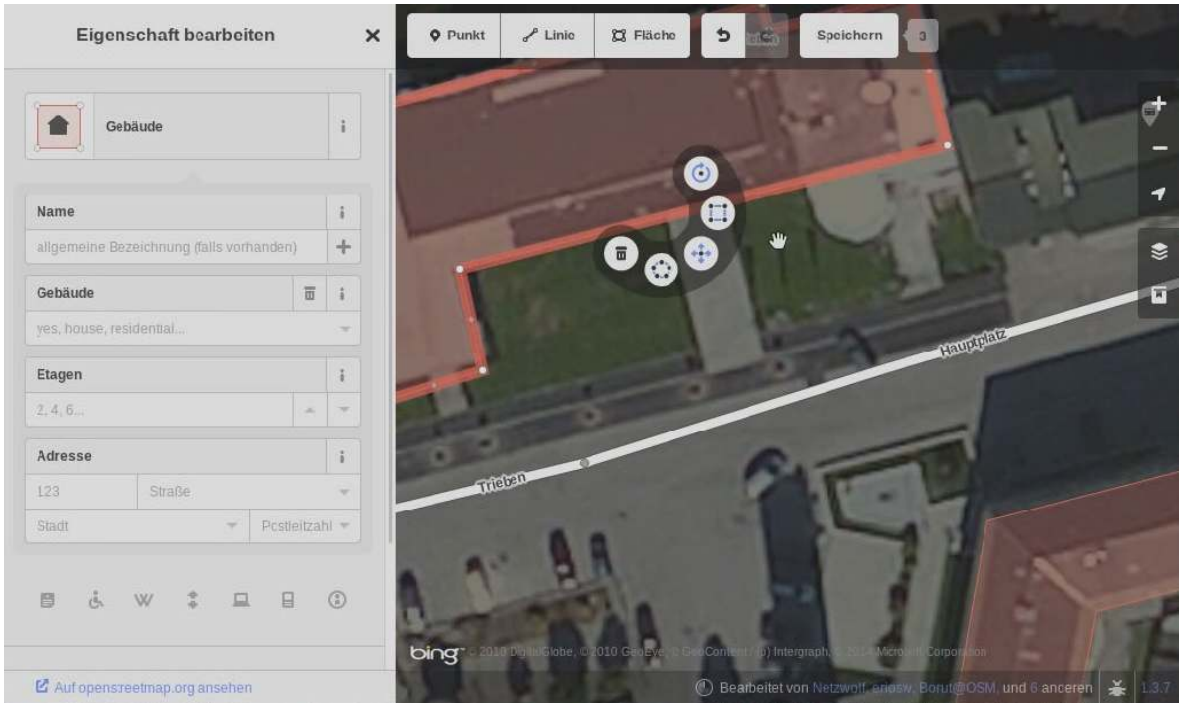


Figure 5.9.: Novice 1 inspects the circular menu during task 3

Some mappers who used the virtual nodes commented on that functionality, sometimes comparing it with the JOSM editor, which has a similar functionality.

> Mapper 6 [`addwaynode`]: clicks on an area's edge; "Ah, now there's such a point here in the middle, as I know it from JOSM... but... it flees when I click it... well, so that, in principle, works the same way as in JOSM"

> Mapper 7 [`addwaynode`]: "Well... for example, I like this better than in JOSM, that I don't have to insert another point"

> Mapper 8 [`addwaynode`]: "That's nice that it turns it into a plus (...) OK, so the point in the middle is the one where you can pull the line. If you know it, it's good"

Users often expressed the wish to make an area rectangular (i.e., squaring its corners). Unfortunately they did not always find the correct way to do it.

> Mapper 2 [`squarecorners`]: draws an area; area is selected, but menu not showing; "Earlier I had the function 'square corners'... why can't I find it now? ... OK, so it isn't rectangular"

> Mapper 4 [`squarecorners`]: draws area; "Yeah, I miss that there's no rectangular function"

> Mapper 5 [`squarecorners`]: "Wonder if there is a rectangular function? I rather don't try"

In one particular case a user had cut an area at one of its nodes, using the 'cut' tool. After this operation the square corners function was not available any more.

> Novice 6 [`squarecorners`]: wants to square the corners of an area that he had cut into two before, so the option does not appear any more; "... I believe last time I clicked on a corner point I had that palette with 'square corners'. Now, when I click on a line something different comes up, when I click on the building something different comes up, and when I click on a corner point something different comes up too. I can't find it"

Other users showed their satisfaction with the 'square corners' function.

> Novice 4 [`squarecorners`]: "So let's see if that works, make rectangular"; clicks 'Square corners' button; "like it"

> Novice 6 [`squarecorners`]: "Square corners... that is nice, I didn't even know that before"

Splitting a line into two parts (required in task 4) was no big challenge to the mapper users. However, there was not always sufficient clarity about the difference between the 'cut' and 'disconnect' functions.

> Mapper 3 [`splitline`]: "So I have to click here"; clicks on the node at which the road is to be split and inspects the menu; "ah, so here we have some scissors. Split, yes. That looks good"; splits the line

> Mapper 5 [`splitline`]: clicks on the node at which the road is to be split and inspects the menu; "Disconnect these lines/areas from each other... Split the line into two at this node... this seems to make more sense, although it is not unambiguous, at least for beginners"; splits the line

> Mapper 8 [`splitline`]: selects line and inspects menu; "That means I have to use 'split'... the question is how"; selects the node and hovers over the 'Disconnect' button; "Okay. Not by clicking on the line, but by clicking on the node at which you want to split the line. The question is only, what else am I going to split when I click that? I'll just click that..."; selects line again; "OK, anyway I haven't split the road... maybe with Shift again?"; selects line and node; "yes, I like that"; clicks on the 'cut' button

5.2.10. Use of help sources

The "LearnOSM" help page (URL 30) was used by three novice and three mapper users: Novice 2, Novice 3, Novice 4, Mapper 1, Mapper 4, and Mapper 8. The OSM wiki (URL 11) was used by five novice and two mapper users: Novice 3, Novice 4, Novice 5, Novice 6, Novice 8, Mapper 7, and Mapper 9.

5.2.11. Other observations

To one mapper it was not obvious how to close the editor.

> Mapper 9 [`closeeditor`]: "I am tempted to close the editor for a minute to make it zoom faster... I'm zoomed in very far right now. I'll click Edit, the normal map view will probably come then... no. (...) I'm a bit disappointed, I'd simply like to close the editor"

One user expressed his satisfaction with the existence of keyboard shortcuts.

> Novice 3 [`keyboardshortcut`]: hovers over *Area* button; "Oh, keyboard shortcuts, I love keyboard shortcuts, that's very good"; presses '3'

The same user also expressed his satisfaction with the overall workflow (during task 2).

> Novice 3 [`remark`]: "I'm currently a bit surprised how quickly you can add things, but until now I haven't noticed anything that I have forgotten, that's great"

The following remark demonstrates a mapper's dissatisfaction with the feature types and standards imposed by the editor.

> Mapper 7 [`remark`]: "What I'm noticing here, on a side note—I do not learn the tags this way. And of course I also don't learn to properly search the wiki, but I'm always depending on the German translation. And if I give this to my colleague in China, he will have to download the Chinese language file. (...) So if the mask doesn't fit, there is no chance for the user, unless he has more knowledge, to stick to the rules. And this is a dependency that I don't like."

5.3. Post-test interviews

After each test the test persons were asked a number of questions (see appendix B) about their personal opinion on their experiences while working with the *iD* editor.

While several users reported they were very satisfied with the overall experience except minor difficulties, some users expressed more mixed feelings about it. Two novice and three mapper users (Novice 3, Novice 4, Mapper 2, Mapper 3, Mapper 8) said they were very pleased by the editing experience and would recommend the program to others.

Five novice and five mapper users gave an overall positive feedback, but also expressed some dissatisfaction (Novice 1, Novice 5, Novice 7, Novice 8, Novice 9, Mapper 2, Mapper 4, Mapper 5, Mapper 6, Mapper 9).

More negative responses were given by two novice and one mapper user: Novice 2 and Novice 6 reported a range of issues they had with the program, preventing them

from developing a satisfying workflow, whereas Mapper 7 explained that he did not like working with predefined categories, which bothered him as an advanced user.

Apart from a general evaluation of the editor, the test persons also stated in the interviews which particular problems they had during the tasks and often also how, in their opinion, these could be mitigated.

- Novice 1 complained that the program did not always react as expected, e.g., when after adding a name the editor would not react to pressing Enter. He also mentioned that the circular menu did not offer an option for adding a node and that he could not find an appropriate feature type in one case. Having experience editing Wikipedia, Novice 1 also suggested that it would be useful if one could comment on individual map features or start a discussion around them, similar to the talk pages in Wikipedia.

- Novice 2 said that using the program was not very comfortable for him, especially because the navigation was not smooth and because he had difficulties interpreting the aerial photos. As it often bothered him he suggested to move the circular menu to another place where it does not obstruct the map view so much.

- Novice 3 was generally satisfied with the tool and stated that he was able to solve all tasks using the OSM wiki as a help source.

- Novice 4 expressed overall satisfaction, too, except that the program was sometimes slow. He wished to have a function that allows the user to move not only a single node of an area (and neither the whole), but an edge.

- Novice 5 was satisfied with the editor and said that he would prefer it over JOSM and Potlatch, which he had tried before. He mentioned he was confused once when he accidentally connected an area and a line object with each other and that a notification by the editor about it would have been helpful.

- Novice 6 complained that it was not always clear to him what was going to happen as a result from a particular interaction in a given context, e.g. when does a mouse click select an object, when does it move the map, when does it drag a point etc. He also said that he sometimes lost the overview and suggested to add a small window within the editor that shows the current map content within a larger geographical context.

- Novice 7 said she took long to adapt to the interface, but enjoyed it more as soon as it went more smoothly.

- Novice 8 said he enjoyed working with the program, but also remembered how he was confused by the *Land Use* feature type category until he understood that more items were lying behind it.

- Novice 9 found the program mostly intuitive, but too slow at times. He did not remember being lost at any point except after the walkthrough when he did not know which map area was shown.

- Mapper 1 said he had no difficulties using the editor and that it was intuitive to use. He only suggested to add more help sources to the editor.
- Mapper 2 expressed overall satisfaction, but said he missed a confirm button in the feature editor.
- Mapper 3 mentioned that he did not manage to add a point to an area and that the search was difficult use, i.e. that he had to zoom out in order to find the object he was looking for.
- Mapper 4 said that the editor was much too slow and that she was irritated by the circular menu. She also wished to have more advanced mapping tools like, for example, the "terracer" tool in JOSM with which you can easily create rows of equally shaped buildings.
- Mapper 5 complained about speed and smoothness and that he had to zoom out to find objects with the search function. He suggested to increase the search area beyond the current map view.
- Mapper 6 mentioned difficulties he had while zooming with the mouse wheel, but otherwise stated that the program was usable.
- Mapper 7 said that he did not like to be guided through lists of predefined categories and that he would rather have an option for entering tags directly. He also reported a difficulty with the search function.
- Mapper 8 expressed his wish for more help sources, or links to the OSM wiki, within the editor as well as better use of symbols and speed optimisation.
- Mapper 9 found the editor mostly intuitive (while slow), but he said he was confused that for moving buildings he had to click a button from the menu. He expressed doubt whether it was a good idea to assign a primary feature type to each object, because they could have multiple uses as well. He also often did not know which tags were behind the various categories and would have liked them to be indicated in the editor more clearly, e.g. in an "advanced view". Furthermore he was disappointed that it was not possible to edit multiple objects at the same time.

From these comments some things become clear regarding both the novice and the mapper group. As far as the novices are concerned, several examples of disorientation or misunderstanding are noticeable—such as difficulties locating functionality, and getting lost in the map view. Mappers, on the other hand, almost always complained about speed, and sometimes about the absence of more advanced features and shortcomings in the search function (which was needed only in the extra task for mappers).

The test persons also answered the question if they would continue to use the *iD* editor in the future. An interesting trend is noticeable in this question. Among the novice users, those who had somehow been acquainted with OSM before (but not edited actively) mostly stated that they would use the editor and make some edits. On

the other hand, those novice users who did not have a strong interest in OSM in the first place were usually not "convinced" by *iD* and said they would rather not start contributing to OSM. Among the mapper group, especially those who were not "heavy" mappers, i.e. who did not contribute to OSM more than once a month, indicated that they would probably use *iD* more often, whereas all the more active contributors said they would not use *iD* in preference to JOSM. This shows that *iD* is, more likely, most suitable for novice users and occasional mappers, but still lacks the potential to attract users who are less motivated to start contributing.

5.4. Conclusions regarding the usability of *iD*

In terms of usability, the presented excerpts of the *thinking aloud* and the interviews only give an impression of what the most crucial issues were. The purpose of this section is to further analyse the results with respect to the specific usability measures of learnability, efficiency of use, frequency and seriousness of errors, and subjective user satisfaction (Nielsen [1992]) as well as the usability heuristics for VGI interfaces formulated by Jones & Weber [2012] (as discussed in section 2.2.2). Nielsen also lists memorability as a key usability goal ("ability of infrequent users to return to the system without having to learn it all over" (Nielsen [1992], p. 15)), but as this was not a long term study, no conclusions can be drawn with respect to memorability aspects of *iD*.

5.4.1. Evaluating learnability

Learnability is defined as the extent to which "the system [is] easy to learn so that the user can rapidly start getting some work done with the system" (Nielsen [1994a], p. 26). It is a key usability concept, as for any program it is desirable that users can come to grips with it quickly. In the case of OSM, which depends on a large amount of active contributors who do not necessarily need to have great expertise, this is especially important: The easier it is to learn, the more likely it is that interested users will adopt the system and start editing.

iD has primarily been designed as a program for novice users, so it needs to be easily learnable even for users with little experience with online mapping tools, or web technologies in general. The evaluation of the tool in this thesis shows that it does a fairly good job at that, but the fact that all test persons had to complete the walkthrough before working on the actual tasks may have contributed to their success. The decision to include the walkthrough in the tests was influenced by the idea that it was part of the program and thus deserved a usability evaluation of its own. In this way, however, one cannot make a judgement about the learnability of the tool if users had tried it without completing the walkthrough first.

But even with the walkthrough included some particular interactions were significantly harder to learn than others. The most striking example turned out to be the procedure of editing feature attributes, such as a name, and having the changes accepted (see codes `w_addname` and `closefeatureeditor`). Almost all test persons expected to find a button reading like "OK", "Save", or "Confirm" and often were uncomfortable when simply closing the feature editor (of which one user said it reminded him of a

"Cancel" interaction). On the other hand, this peculiarity was carefully introduced during the walkthrough by asking the user to practise this step four times (add a name to the cafe, change its name, add a name to the playground, and add a name to the road). While some test persons said that once they were taught the concept, it was acceptable for them, others remained in discomfort with it all the way through the test. Based on these observations it can be argued that first-time users who decide to skip the walkthrough are very likely to be confused by the way attributes are saved.

Another thing that seems relevant in terms of learnability and that was observed with several test persons (especially novice users) was their failure in adding nodes to existing lines or areas, or moving nodes (codes `addwaynode` and `movewaynode`). While the circular menu provided opportunities for all sorts of editing tasks, adding and moving nodes was not included in it. In other words, the functionality was hidden—and not introduced by the walkthrough either. This posed an obstacle to novice users: Neither did they find anything in the circular menu, nor did a right-click help, which some test persons tried. The confusion may have been increased by the fact that moving line and area objects as a whole *was* an option from the circular menu, whereas moving nodes was not.

Furthermore, the circular menu offers some tools that can be considered rather advanced, for instance, the 'cut' and 'disconnect' tools (see code `splitline`). Test persons of the mapper group were mostly able to use these tools appropriately, although they sometimes hesitated which of the two was the better choice. Novice users, on the other hand, often did not seem to understand these tools, but sometimes tried them nevertheless in an attempt to perform a completely different action. Then it was not clear to them what the tool had done (as this was not explicitly indicated by the editor), and they were left astray with the obscure results they had produced (e.g. in one case a novice user split an area and then wondered why he could not find the 'square corners' tool any more). This demonstrates that the learnability of the program, with respect to the tools mentioned can be improved.

Comparing this study with the usability study of the Potlatch 2 editor in Jones & Weber [2012], it also becomes clear that sometimes the same mistakes have been made in the design of the two editors, whereas other issues of Potlatch 2 have been resolved in the design of *iD*. Referring to the usability heuristics that Jones & Weber have developed specifically for learnability issues of VGI interfaces, another perspective on the learnability of *iD* is given.

- *Provide visibility and feedback of VGI edits status.* Jones & Weber noted that for Potlatch 2 there was a need for more system feedback, e.g. when users were able to start editing or when their edits were finished and recognized by the system. Similar issues are apparent with *iD*. For example many test users were not always sure whether the information they had entered in the feature editor was immediately saved or not; there was also confusion with some novice users not being aware of whether they were in drawing mode or not.

- *Match languages of VGI object attributes to the real world, and embed meaning for the users.* A well-designed system takes care that it confronts the user with terms and concepts that he/she knows, rather than using terms only the developers

have in mind. This is especially relevant in the context of OSM, of which the data format is not easily comprehensible and therefore should not necessarily be required to know for using an editor interface. With regard to this *iD* is well on the right path, having replaced the node/way terminology with more understandable 'points', 'lines', and 'areas'. However, the plain tags still show where perhaps they should not, e.g. in the different options for attributes like access or surface type, for the understanding of which one might have to consult an external help source like the OSM wiki.

- *Ensure clarity and consistency of the editing process for different map objects.* Jones & Weber have criticized major inconsistencies in Potlatch 2, e.g. that it takes different approaches to adding point, line, and area features, respectively. *iD*'s concept of adding things is much more consistent, always following the same pattern of clicking the *Point/Line/Area* button, drawing it, and finally editing its attributes. One thing that is inconsistent in *iD*, however, is the way objects are moved: While points (and nodes that are part of lines or areas, for that matter) are moved by dragging and dropping, lines and areas can only be moved by selecting the appropriate option from the circular menu. This has in fact confused several of the test persons, as discussed in section 5.2.9.

- *Present editing options for specific objects only.* Whereas Potlatch 2 was criticized for offering the same feature types for both point and line objects, *iD* offers typical feature types for the respective geometry types (e.g., it offers such feature types as restaurant or supermarket for point objects, but road and river types for line objects). It also presents fields for appropriate feature attributes depending on the feature type (e.g., the "operator" tag for supermarkets or the "capacity" tag for car parks). However, one user has commented that *iD* could also be more context-sensitive with respect to the circular menu: The 'make circular' tool could be removed at objects that are already approximately rectangular, and buttons that are deactivated, for whichever reason, need not be shown greyed out, but could be removed altogether.

- *Have consistent and standardized map controls and clear interaction elements for both editing and viewing different windows.* As *iD* has a dedicated map panel on the right side that is even consistent with the map panel on the OSM main page, one can hardly complain about the placement of control elements in *iD*. The only thing that was apparently hard to find for some test users was the search function within the editor: it shows up only when the main pane is not occupied by anything else, which is, of course, not always the case; for example, when the cursor is placed above any map object, the main pane shows the feature editor. Thus it would be better to have a search button or form that is always visible.

- *Minimize editing errors by preventing similar interaction actions for different VGI edits.* Jones & Weber observed that with Potlatch 2 test persons often inadvertently moved map objects when they intended to pan the map. The fact that the same mouse interaction (moving the mouse while pressing down the left mouse button) was functional for different kinds of interaction made the test

persons do things they did not want to do. Sadly, the developers of *iD* have not learned this lesson: moving points and panning the map are still done the same way. While this may make working with the tool more efficient for users who are aware of it and thus work carefully enough, to inexperienced users it poses a conceptual difficulty as well as a danger of making mistakes, which may have negative effects on their confidence.

- *Understand the editing task from a user's perspective and their behaviour and ensure interface visibility of common editing interactions.* Making interactions and options visible to the user is a key to usable software. While *iD* does not feature as many shortcomings as Potlatch 2 in this regard, it does have hidden functionality too. The most apparent problem during the tests was the issue of adding and moving nodes of lines or areas; this option is not available from the menus (as most other tools are) and, therefore, novice users often struggled to find it. What is done well in *iD* are the keyboard shortcuts, which several of the test persons adopted happily—meanwhile, using them is never required, but always optional.

- *Enable users to escape easily from editing errors.* Preventing and mitigating errors is vital to the usability of VGI editors as much as to any other kind of software. What has been observed is *iD*'s unforgiving behaviour when the undo button is pressed while a user is drawing a line or area. It takes control away from the user and when it happens, there is also no clear indication that he/she is not in drawing mode any more.

- *Provide obvious and visible links to editing tutorials, videos and help from the editing interface.* *iD* already has a good set of help sources: first the 'i' button next to each individual feature type and attribute field embedding information from the OSM wiki if available, and second the help menu that is accessible from the map panel. However, many test persons still had to use the external help sources made available to them (OSM wiki and LearnOSM). Some were disappointed because the 'i' button often yielded nothing but a message that no help was available—sometimes even although the OSM wiki actually had information about the object in question. Besides, it is worth noting that almost no test person had actually found the help menu in the map panel. While this may be due to the fact that external help sources had been advertised before the test, another reason could be that the meaning of the icon from which the help is accessed was not easily understood.

To sum up, the case study revealed a number of learnability issues of *iD*, most notably the absence of a confirm interaction in the feature editor, difficulties with adding and moving nodes of lines and areas, and an inconsistency in the way map objects are moved.

Apart from the various aspects of the program's learnability that have now been discussed, this study also draws conclusions for *iD*'s efficiency of use, error tolerance, and user satisfaction.

5.4.2. Evaluating efficiency

According to Nielsen, a "system should be efficient to use, so that once the user has learned the system, a high level of productivity is possible" (Nielsen [1994a], p. 26). Of course, it is not always clear whether the participants in a test like this have already "learned" the system (or parts of it) at some point during the test, but nevertheless it can be observed whether a user has developed a smooth workflow with respect to a specific subtask. For example, many test persons quickly adopted the procedure of adding simple features in *iD*: Click the Point, Line, or Area button, draw the object, select a feature type, and edit the attributes. Most test persons internalized this procedure during task 2, in which nine map objects were to be added from information given on a field paper. A few users even expressed their satisfaction about the ease of use, e.g., "I'm currently a bit surprised how quickly you can add things" (Novice 3).

With regard to the editing of existing features (task 3), several test persons, especially novices, had significant problems: whereas the average completion time for task 3 among mappers was 8 minutes, among the novices it was 15 minutes—a difference more significant than in any of the other tasks (see section 4.5.2). This may be due to the fact that in some ways, specifically in the way objects are modified, *iD* makes use of functionality known from other editors, giving advantage to users who are experienced editing OSM. This is, of course, essentially a good thing as long as novice users are given proper guidance to learning this functionality (unfortunately though, the walkthrough does not cover the essentials of modifying the geometries of existing map objects).

Test persons especially of the mapper group complained about more inconveniences hindering their workflow. One of the most frequent complaints was about the program's speed, a problem also due to *iD*'s nature as an in-browser editor. Three test persons also complained about the low speed of the zoom when using the mouse wheel. Another, minor flaw was, according to some mapper users, that they could not edit multiple objects at the same time (although they could *select* multiple objects). This is clearly a constraint to power users who wish to finish their work as quickly as possible.

Other than that, an overall evaluation of the editor's efficiency should highlight the ease with which most users managed to add new objects to the map, and in part their success in using advanced functionality.

5.4.3. Evaluating errors

Nielsen suggests that "the system should have a low error rate, so that users make few errors during the use of the system, and so that if they do make errors they can easily recover from them" (Nielsen [1994a], p. 26). This has basically two aspects: errors within the system, and the way the system deals with user mistakes.

Software errors in the system ("bugs") have appeared frequently: during the walkthrough task the system hung 13 times (see code `w_bug`). This should of course be fixed, for "catastrophic errors must not occur" (Nielsen [1994a], p. 26). A more thorough analysis is needed with respect to the way the editor deals with the users' erroneous behaviours. The following is a list of the most frequently occurring user errors that could not always be resolved easily.

- At least eight test persons had problems selecting points on the map, i.e. either they clicked near but not exactly on the point they wanted to select or they moved the mouse while pressing the mouse button down, which resulted in panning of the map (see code `selectpoint`). Of course, it is not their fault, but more likely due to difficulties they had with the hardware they were not comfortable with, or a lack of motoric precision. Either way, the editor program should be more forgiving about the user's precision, e.g. allow cursor movement and accept mouse clicks within a defined buffer around the point.

- Another recurring issue users had involved drawing a line or area object (codes `drawline` and `drawarea`): While drawing, users sometimes realized that they had made a mistake and wanted to undo the last node, so they clicked the undo button. Then, however, the program finished the object, leaving the drawing mode so that it was not possible to resume the drawing. Most test persons who found themselves in this situation deleted the object they had started drawing and then started to draw the object from scratch again. This behaviour of the editor is rather unfortunate, as users may find themselves in such a situation quite often. After all, one cannot expect them to get every drawing right in the first place.

- Several times it happened that test persons accidentally connected an object they were drawing with another object on the map by placing a node on the other object's shape (see also `drawarea`). From a mapper's perspective this may be the right thing to do—but it may as well not. In the cases that were observed the users did not intend to connect the objects and thus looked for a way to revert it. Unless they used the undo function (which comes with its own caveats as I have shown before), they needed to find the 'disconnect' tool which is hidden in the circular menu of the node that connects the two features. This is a very cumbersome way of correcting an error especially for less experienced users.

- Finally, another issue with areas was the editor's reluctance to react when a user clicked on the wrong node, e.g. in an attempt to close the area (it would only close the area properly when the resulting shape did not have overlapping edges). When the program already recognizes the user's error—otherwise it would not behave differently—, it should also tell the user why nothing happens.

Altogether, besides the critical bug that has been observed many times during the walkthrough, *iD* is too sparing with feedback when users make mistakes. While too many system messages can slow down the user's flow of work, some more feedback would probably be helpful, especially in the situations described.

5.4.4. Evaluating subjective user satisfaction

Of all usability measures, subjective user satisfaction is probably the easiest to evaluate. I have already given an overview of the statements the test persons have made in the debriefings after each test (see section 5.3). Given that 7 out of 9 test persons from

each group have given mostly positive feedback about their subjective satisfaction, *iD*'s user experience can be regarded as a mostly pleasant one.

However, 2 in 3 novice users mentioned at least one situation in which they got stuck and which took them rather long to resolve, e.g. while selecting a feature type, moving and adding nodes, etc. The mapper users, on the other hand, complained less about difficulties they had using the editor, but more frequently about speed issues or functionality they missed; generally the mapper users seemed to have learned the tool more easily, perhaps due to their prior experience with OSM.

5.4.5. Summary

In this section many of the individual issues found through the thinking aloud observations have been summarized and evaluated in the light of the four usability criteria of learnability, efficiency, errors, and user satisfaction. In addition, the learnability heuristics by Jones & Weber [2012] have been used to investigate whether the design of *iD* conforms to their recommendations for VGI editor interfaces.

It has been shown that, although *iD* is no overall failure in any of those aspects, there are clear opportunities for improving the tool. This investigation has demonstrated that, especially in terms of learnability and error tolerance, there are considerable obstacles for novice users and that easier recovery from user errors is needed.

The following and last chapter will conclude the thesis by summarising the answers to the research questions this study has found and by giving recommendations for improving *iD* as well as for further research on this topic.

6. Conclusion and recommendations for future research

6.1. Findings

In the introduction to this thesis (chapter 1) I have framed a number of research questions which have now been answered by the thesis work in order to achieve the objective of investigating the usability of *iD*. Summaries of the findings are given below, followed by an overall reflection as well as recommendations for future research.

1. What is *iD* and what is its purpose?

I have explained in chapter 1 how *iD* is one of the most recent and important developments in the realm of OSM editor tools. Its purpose is to make editing easier and more accessible to infrequent and new contributors to OSM. This is evident from several facts: (a) designed as an in-browser editor, *iD* does not require the user to download and install anything before he/she can make edits; (b) written in JavaScript, it makes use of a widespread technology supported by all popular web browsers; (c) abstracting the user interface from the OSM data format (i.e., introducing predefined feature types and point/line/area types) in order to achieve a better match between the system and the real world, *iD* ostensibly targets a novice audience that is not necessarily familiar with the tagging of objects in OSM; and (d) having an appealing and clean design, it advertises itself to users who appreciate visually appealing interfaces, thus lowering the barrier they might face when testing the program.

2. What are the expectations of users using *iD* and how well does *iD* satisfy them?

The test persons' expectations have become clear during the tests at least partially, and they varied between the novice and mapper groups. While novice participants with no prior OSM experience went into the test with little or no expectations, those who had already tried to make some edits hoped for a tool that would be easier to use than the ones they had previously tried. The mapper test persons, on the other hand, mostly knew other existing editors well and usually had their own preference towards one of them—they might have expected *iD* to be a usable alternative for small and quick edits in the browser, but not necessarily as a replacement for another editor. Additionally, several mappers expressed their motivation to contribute to a study that will lead to the improvement of OSM tools.

As the evaluation of the post-test interviews has shown, most of the participants were satisfied with the editor; some novices and infrequent mappers stated that they would continue to use *iD* in the future, and some of the more dedicated mappers said

they might embrace it as an alternative for smaller edits, whereas they would stick to other editors (such as JOSM) for bigger tasks.

3. What usability criteria apply to the testing of *iD*?

In section 2.2 I have given an overview of the field of usability evaluation and the criteria commonly used. In principle, the same usability criteria that are applied to any web tool or stand-alone program can be applied to VGI editing tools including *iD*, too.

In research done by Jones & Weber [2012] *iD*'s predecessor Potlatch 2 was subject of a usability case study. The study focused on an evaluation of learnability as one important usability criterion. Jones & Weber have furthermore expanded upon existing and widely adopted measures of learnability and reformulated them in order to provide a framework for evaluating learnability in VGI tools specifically. With reference to their work, I have used their heuristics for answering the research question of how usable *iD* is in section 5.4.1.

4. What is the usability of *iD*?

This study has highlighted a number of issues that were evident in the thinking aloud data as well as in the subjective user feedback (see section 5.4). While it is difficult to summarize the usability of *iD* in a few sentences, the responses given by the participants paint a mostly positive picture. On the other hand, the observations from the tests reveal several issues in terms of learnability, efficiency, and errors.

Learnability affects the way how easily and quickly novice users get used to using the system. The analysis has shown that *iD* is in fact easily learnable, as the majority of test persons have picked up its basic functionality quickly. The walkthrough certainly contributes to that and is itself an easy thing to do as well. However, the tests have revealed a number of issues that should be mitigated in order to make the program even more attractive to novice users. Especially more feedback about the system's status, clarity and consistency of editing options, and more help options are needed.

Based on the results presented, *iD* can also be regarded as an efficient tool. Although it suffers from speed issues, most of the test persons got used to the program quickly, at least in the first editing task, task 2. During task 3, however, especially novice users had greater difficulties. More forgivingness with respect to cursor precision is also welcomed in order to make sure users will not struggle with the most basic actions such as selecting points.

In terms of errors and error prevention, first the walkthrough needs to be fixed to show no more bugs; other than that it was often observed that users had to spend too much effort into correcting their own mistakes, e.g. misplaced nodes or inadvertently connected objects. As this can be very frustrating for first-time users, it is important that *iD* deals with user errors much better.

The test persons' satisfaction with the tool, as evaluated from the post-test interviews, was found to be mostly positive, possibly with three exceptions. The novice users often appreciated how easy to use the program was and considered it a possible alternative to other editors, and the mapper users, although they would not always switch to using it themselves, made positive and critical remarks alike.

The various issues found (and presented in chapter 5), less and more severe, need to be analysed individually in order to derive an agenda for improving the *iD* editor. From the findings presented I give some recommendations below.

5. Which conclusions for improving *iD* can be drawn from the tests?

Recommendations for improving the *iD* editor have been given by some of the test persons themselves and can also be derived directly from the analysis part of this thesis. While some changes to the tool could probably be implemented easily, others will require more effort with regard to programming and will also need to be considered more carefully. The following list has suggestions derived from the participant observations and interviews, loosely ordered by the intricacy that the changes will entail.

- **Fix the bugs.** The most frequent bug appeared on occasions during the walkthrough. Hasty dragging and/or clicking by the users often led to elements being misplaced, e.g. the next instructions window showing in the wrong place, or not at all. This clearly needs to be fixed in order to not upset first-time users, who are most likely to start with the walkthrough. Another bug appeared with the keyboard shortcut for "save" (see section 5.2.3).

- **Fix the name-adding part of the walkthrough.** The point almost all users got stuck at in the walkthrough was when they were supposed to add a name to the cafe. As the instructions do not change as soon as something has been typed in, the users were confused, not having a clue what was to be done next. While this should be an easy fix, it will increase the users' confidence significantly.

- **Explain what the feature type categories are.** The feature type categories (as opposed to singular feature types) that are shown like a stack of feature type buttons were often not understood. Even after repeated expanding and collapsing some users did not realize that those buttons actually opened up lists of more specific feature types. Maybe a help text could be added or better symbolisation applied, and it could be covered explicitly in the walkthrough.

- **Explain how an object's feature type can be changed.** In order to change an object's feature type one has to click on the feature type button at the top of the feature editor. This may be intuitive to some, but test persons have also been observed to overlook this functionality. It would probably be good to cover this in the walkthrough.

- **Improve the search function.** Many test persons spent too much time with the search function, mostly because the selected search result was not highlighted properly (i.e., the map was not centred at it). Also the search should be multilingual where possible and could be more context-sensitive (e.g. search not only in the current map view, but also around it).

- **Inform the user about problems while drawing.** In several instances users made mistakes that were not very obvious and yet they received no feedback by the editor that something went wrong (e.g. attempting to close an area at a node

where it is not permitted, or trying to place a node in an inactive portion of the map window). In such a case the editor should clearly indicate that what the user is intending to do is not allowed. Otherwise the user gets stuck and frustrated.

- **Let users edit multiple objects at a time.** Several of the mapper users were disappointed that editing of multiple objects was not possible. Adding this functionality would perhaps make *iD* more appealing to experienced users.

- **Provide more help.** Although some help is already available, test persons often had to check one of the external help sources. The help hidden behind the 'i' buttons can be improved, and the concept could be extended to other interaction elements (e.g., the items of the circular menu and the map panel). Useful external help sources other than the OSM wiki could be embedded or linked to from within the editor.

- **Be more clear about the system's status.** Especially novice users were sometimes not aware which mode they were currently in, i.e. select mode versus drawing mode. Moreover, many users were unsure, especially at the beginning, whether the feature attributes they had entered were immediately accepted by the system, as there was neither a confirm button nor a notification. In a more specific case a user was confused that the contents of the circular menu had changed after he had performed a cut operation on an object, obviously not realizing the causality. All these things should be signalled more clearly to the user.

- **Create consistency.** There is some inconsistency in the editor's interactions that has apparently confused some users. For example, moving a node is done in a completely different way than moving a line or area (and can also be irritating if done inadvertently).

These are some of the most obvious suggestions that can be derived from the observations made during the user tests in this study. However, this list is not exhaustive and more issues (possibly for further research) can be found through a deeper investigation of the quotations presented in section 5.2, or directly from the original recordings attached on DVD.

6.2. Reflection and further research

Usability testing can never be completed by a single evaluation, but should rather be part of an iterative design cycle that includes testing with users at every stage of development. Already by the time this thesis is published, *iD* has developed further away from the version that has been evaluated in this study. Therefore further investigations into the tool will be needed.

One issue with this research being that it is quickly outdated by *iD*'s development, there are more limitations to this study. For one thing I have described in section 4.3.4 the technical issues I had with the recordings. This is, in large part, due to my decision to use consumer equipment in order to allow for greater flexibility in the planning of the

tests. Usability tests conducted with a high-end laboratory equipment, on the other hand, would likely have produced better results in terms of technical quality. With better equipment one might also use eye tracking and/or video recording as additional observation methods, which could help to discover even more usability issues with the study object.

For another thing, this study is not a long-term study, but instead involved one test per participant, usually not lasting longer than an hour. Under such circumstances it is not possible to extend the usability evaluation to the aspect of memorability, as discussed in section 5.4. Finally, while a large amount of data have been collected in this study, this thesis may fall short of discussing all possible analyses and conclusions that the data might have allowed. Especially conclusions with regard to the test persons' unique characteristics and interview responses could be extended further.

Any future research on the topic that will help to overcome the shortcomings mentioned will be welcomed by the scientific and the OSM community. After all, for the sake of iterative design, usability testing of more recent and future versions of *iD*—preferably ones in which the findings of this study have been taken into account—is necessary and thus strongly recommended.

As I have stated in chapter 2 of this thesis, usability testing is important to OSM and its future development as the primary alternative to commercial and authoritative geodata sources, because more contributors need to be attracted to and kept engaged with OSM. Lowering the barriers for contributing needs to remain a main goal of the project and it requires successive investigations into the usability of the tools.

Research on VGI software is furthermore not only relevant to OSM, but also to the overall field of cartography. More and more web mapping tools embrace the possibility of allowing the creation of user-generated geographic content, as its potential to elicit interesting and valuable information has been proved to be great. As much as the market leaders of online mapping have developed de facto standards for digital maps, OSM and other VGI applications can and should take the lead in the design of online map editing tools. As these tools are becoming more sophisticated, they also need to be tailored to different target audiences—as *iD* is a tool mainly targeted at beginners. Additional research into the different applications and design possibilities of map editors will be needed in order to engage more people in generating geographic content online, worldwide.

A. Online survey

Dear participant,

Welcome to the preparatory survey for my iD usability test. My name is Jan Behrens and I am a contributor to OpenStreetMap (OSM), the free wiki world map. Meanwhile, I'm doing research about one of the OSM tools, the 'iD' editor, for my thesis in the Master of Cartography programme at TU Vienna. My objective is to test the usability of the iD tool, that is to determine how useful and usable it is. Thus I am looking for test persons who are willing to participate in a user test. You could be a suitable candidate for the user test if one of the following applies to you:

- You haven't ever contributed to OSM, or
- you have contributed to OSM, but aren't familiar with the iD editor yet.

I will invite a number of respondents to participate in it (that is, it may happen that I will not invite you even after you completed this survey, but I will notify you in any case). The test will be done with a computer provided by me, in or around Hamburg. It could be at your place, at my place, or in any public place of your choice. It will take around 30 minutes, and you will receive a little compensation for your effort. The date for the test should be in April 2014.

This survey will help me plan the test and interpret the results. It will take a few minutes to complete. When you're done, I will get in touch with you in order to arrange the actual user test with you (please note that it may take me a couple of weeks to reply).

Most answers are voluntary, but the more information you provide, the more valuable it will be for my research. Also you must leave your contact details. I will respect your privacy and will not give any personal information to third parties. In my thesis I will only publish aggregated, anonymized data.

1. *[mandatory]* Have you ever contributed data to OpenStreetMap (OSM)?

   ☐ Yes

   ☐ No

2. How did you contribute? *[multiple answers possible] [*]*

   ☐ edited the map in my neighbourhood

   ☐ edited the map in places I went (e.g. during vacation)

   ☐ edited the map in places I didn't go (e.g. using aerial imagery)

☐ uploaded GPS tracks

☐ Other: _____

3. *[mandatory]* When did you contribute to OSM for the last time (roughly)? *[*]*

   ☐ less than a week ago

   ☐ less than a month ago

   ☐ less than six months ago

   ☐ more than six months ago

4. *[mandatory]* Which editor do you use mostly when editing OSM? *[*]*

   ☐ iD (currently the default editor)

   ☐ Potlatch 2 (the default editor until August 2013)

   ☐ JOSM

   ☐ Merkaartor

   ☐ I don't know

   ☐ Other: _____

5. How often do you edit OSM (approximately)? *[*]*

   ☐ several times a week

   ☐ once a week

   ☐ several times a month

   ☐ once a month

   ☐ several times a year

   ☐ once a year

   ☐ I have edited no more than one or two times

   ☐ No answer

6. How is your experience with information technologies in general?

   ☐ I can accomplish basic computer tasks

   ☐ I can accomplish advanced computer tasks

   ☐ I am an IT (quasi-)professional

   ☐ No answer

7. How is your experience with geographic information systems (GIS) other than OSM?

   ☐ I have no idea

☐ I have some experience with GIS software

☐ I have lots of experience with GIS software (3 years or more)

☐ No answer

8. How is your experience with user-generated content on the web? *[multiple answers possible]*

   ☐ I use social networks

   ☐ I upload photos or videos

   ☐ I have a blog

   ☐ I contribute to crowdsourcing projects other than OSM

   ☐ Other: _____

9. Your age: _____
10. Your highest education degree: _____
11. Your occupation (and field): _____
12. Your OSM username: _____ *[*]*
13. *[mandatory]* Your email: _____

    (I will need to contact you somehow. Alternatively you may enter a phone number.)

[*] Only displayed when the answer to question 1 was "Yes"

Thank you for taking the survey! I will contact you soon.

B. Test materials

iD user test – Instructions

This test contains a set of tasks that I am asking you to complete. From the way how you interact with the editor program I will draw conclusions about how usable it is. So it is important that you try seriously to complete the tasks. Remember though that this is not to test you, but the software that you are using.

I want to find out as much as possible about potential problems with the OpenStreetMap editor that you are going to use. To help me detect such problems, I am asking you to *think aloud* during the test. That means, you should always *say out loud* what you are thinking while you are working on the tasks.

As you are the one I want to learn something from, I will only be watching you passively.

I am hereby informing you that I am going to produce a *screen capture* and an *audio recording of you*—if you do give me permission to do so.

Before you start, have a look at the tabs that are open in the browser window. There is an OpenStreetMap tab, in which you will do all the work. On the remaining tabs you will find resources that you may use to get help in case you are stuck at some point: the OpenStreetMap wiki containing a general documentation of OpenStreetMap, and a "LearnOSM" cheat-sheet for using the iD editor. Remember that I am not going to help you, so please use those resources.

You do not have to solve the tasks fast, you may take yourself all the time you need.

Task 1 – Getting started

a) In OpenStreetMap, find the town **Trieben** (Styria, Austria). Use the search form as well as the pan and zoom functionality. Have a look at the 'field paper' (see attachment). Compare the map in the browser with the field paper and adjust it so that both maps show roughly the same area.

b) Click the "Edit" button. The iD editor will start up. Click "Start the Walkthrough". Go through the introduction one by one and follow the instructions on the screen. Read and think aloud.

Task 2 – Using a field paper

Having learned the basics of iD, you should now be able to make use of the field paper. Obviously, somebody has used it to survey some places in Trieben. On the field paper there are hand-written annotations of features that I wanted to add to the map. Unfortunately, I have not found the time to do it yet, so you will take over.

a) Add the annotated features to the map. Make sure that you place them at their correct locations and add all available information such as the type of the feature, its name (if any), and other attributes. (There is a total of nine features to be mapped.)

Hint: In a few cases it will be necessary that you edit objects that are already on the map.

b) When you are finished, upload the changes you have made by clicking on "Save". Enter "field paper exercise" as a commit message.

Task 3 – Using aerial imagery

We are out of field papers now, but no worries. We will resort to aerial imagery, which you have already noticed in the background of the map.

a) Use the aerial imagery to add more features to the map. Add **two** different objects that you can recognize in the imagery (for example: a driveway, a garden).

b) Is the map looking good now? Take a closer look (use the zoom). Try to notice how some of the building outlines (especially near Hauptplatz) do not match the background imagery very exactly. Improve the geometries of at least **two** buildings. Make sure that the objects you edit will match the imagery with the best precision.

Hint: You can achieve higher precision by adding more nodes to the outline.

c) When you are done, save your changes.

Task 4 – Using local knowledge

This task was given only to participants of the mapper group.

There is one more little thing that requires our attention. Find the street that is named "Martin-Luther-Platz". It intersects with another street named "Trieben", and further south it continues as "Gartengasse". From a local person we know that Martin-Luther-Platz and Gartengasse have a speed limit of 30 km/h, starting from the intersection. Put this information on the map.

Hint: You will have to split a road in order to solve this task.

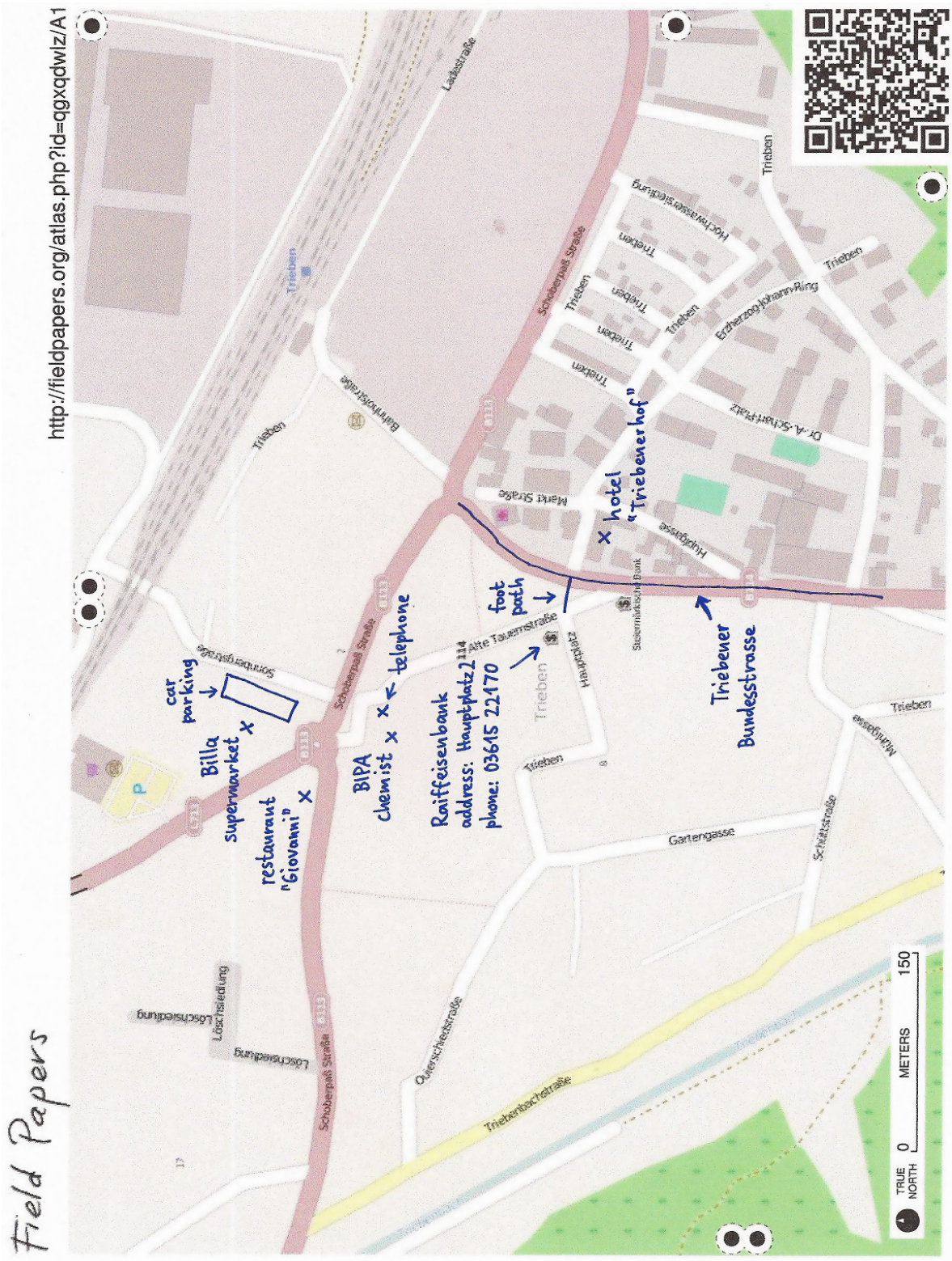


Figure B.1.: The field paper (attachment to task 2)

Post-test questions

Questions about the test:

1. How do you feel you have performed in the test?
2. Did you find the tasks too easy? Too hard?
3. Was the editor pleasant to use?
4. Were you sometimes distracted or confused while editing?
5. Was there a problem you would not have solved without the help provided?
6. How could the editor be improved in your opinion?
7. Would you recommend editing OSM/iD? Why?/Why not?
8. Will you contribute more to OSM? If yes, what motivates you?/Will you use the iD editor more often?

Questions on background (depending on the test person's answers given in the pre-survey):

1. Have you ever used OSM before? If yes, what for?/For what purposes have you been using OSM?
2. How do you usually get help while editing OSM?
3. Which GIS programs have you used?
4. Which social media platforms do you use?
5. Which other crowdsourcing projects do you contribute to?

C. Recordings

The DVD-ROM attached to this thesis contains the following files and folders:

`README.txt`	General information
`videos`	(folder)
`mapper1.mp4`	Recording: Mapper 1
`mapper2.mp4`	Recording: Mapper 2
`mapper3.mp4`	Recording: Mapper 3
`mapper5.mp4`	Recording: Mapper 5
`mapper6.mp4`	Recording: Mapper 6
`mapper8.mp4`	Recording: Mapper 8
`mapper9.mp4`	Recording: Mapper 9
`novice1.mp4`	Recording: Novice 1
`novice2.mp4`	Recording: Novice 2
`novice3.mp4`	Recording: Novice 3
`novice4.mp4`	Recording: Novice 4
`novice5.mp4`	Recording: Novice 5
`novice6.mp4`	Recording: Novice 6
`novice7.mp4`	Recording: Novice 7
`novice8.mp4`	Recording: Novice 8
`novice9.mp4`	Recording: Novice 9

Bibliography

Arhippainen, L., & Tähti, M. (2003). Empirical evaluation of user experience in two adaptive mobile application prototypes. In *Proceedings of the 2nd international conference on mobile and ubiquitous multimedia* (pp. 27-34).
Retrieved from http://www.ep.liu.se/ecp/011/007/ecp011007.pdf

Ather, A. (2009). *A quality analysis of OpenStreetMap data.* ME Thesis, University College London, London, UK.
Retrieved from ftp://s-she-junco.cits.rncan.gc.ca/pub/cartonat/Reference/VGI/Dissertation-OpenStreepMap-Quality-Aather-2009.pdf

Bakhtin, M. M. (1986). *Speech genres and other late essays* (No. 8). University of Texas Press.

Boren, T., & Ramey, J. (2000). Thinking aloud: Reconciling theory and practice. *IEEE Transactions on Professional Communication, 43*(3), 261-278.
Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=867942

Budhathoki, N. R. (2010). *Participants' Motivations to Contribute Geographic Information in an Online Community* (Dissertation, University of Illinois).
Retrieved from https://www.ideals.illinois.edu/bitstream/handle/2142/16956/1_Budhathoki_Nama.pdf

Das, T., & Kraak, M. J. (2011). Does neogeography need designed maps. In *Proceedings of the 25th international cartographic conference and the 15th general assembly of the international cartographic association, International Cartographic Association (ICA), Paris* (pp. 1-6).
Retrieved from http://hostmaster.icaci.org/files/documents/ICC_proceedings/ICC2011/Oral%20Presentations%20PDF/B3-Volunteered%20geographic%20information,%20crowdsourcing/CO-123.pdf

Dicks, R. S. (2002). Mis-usability: on the uses and misuses of usability testing. In *Proceedings of the 20th annual international conference on Computer documentation* (pp. 26-30). ACM.
Retrieved from http://dl.acm.org/citation.cfm?id=584960

Eckert, J. R. (2010). *Tropes 2.0: Strategic Mobilizations of the Geoweb* (Master's thesis, University of Washington).
Retrieved from http://students.washington.edu/jeckert1/eckert_masters_thesis.pdf

Ericsson, K. A., & Simon, H. A. (1984). *Protocol analysis*. MIT press.
Retrieved from: http://ammonwiemers.com/IdetPortfolio/articles/Assessment/Protocol%20Analysis%20--%20Verbal%20Reports%20as%20Data.pdf

Goodchild, M. F. (2007). Citizens as sensors: the world of volunteered geography. *GeoJournal, 69*(4), 211–221.
Retrieved from http://link.springer.com/article/10.1007%2Fs10708-007-9111-y

Gould, J. D., & Lewis, C. (1985). Designing for usability: key principles and what designers think. *Communications of the ACM, 28*(3), 300-311.
Retrieved from http://dl.acm.org/citation.cfm?id=3170

Haklay, M., & Weber, P. (2008). OpenStreetMap: User-Generated Street Maps. *Pervasive Computing, IEEE, 7*(4), 12-18.
Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4653466

Haklay, M. (2010). How good is volunteered geographical information? A comparative study of OpenStreetMap and Ordnance Survey datasets. *Environment and Planning B: Planning and Design, 37*(4), 682-703.
Retrieved from http://kfrichter.org/crowdsourcing-material/day1/haklay10.pdf

Holzinger, A. (2005). Usability engineering methods for software developers. *Communications of the ACM, 48*(1), 71-74.
Retrieved from http://dl.acm.org/citation.cfm?id=1039541

Hwang, W., & Salvendy, G. (2010). Number of people required for usability evaluation: the 10±2 rule. *Communications of the ACM, 53*(5), 130-133.
Retrieved from http://dl.acm.org/citation.cfm?id=1735255

Jones, C., & Weber, P. (2012). Towards Usability Engineering for Online Editors of Volunteered Geographic Information: A Perspective on Learnability. *Transactions in GIS, 16*(4), 523–544.
Retrieved from http://onlinelibrary.wiley.com/doi/10.1111/j.1467-9671.2012.01319.x/full

Kounadi, O. (2009). *Assessing the quality of OpenStreetMap data.* MSc geographical information science, University College of London Department of Civil, Environmental And Geomatic Engineering.
Retrieved from ftp://ftp.cits.nrcan.gc.ca/pub/cartonat/Reference/VGI/Rania_OSM_dissertation.pdf

Krahmer, E., & Ummelen, N. (2004). Thinking about thinking aloud: A comparison of two verbal protocols for usability testing. *IEEE Transactions on Professional Communication, 47*(2), 105-117.

Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1303808

Law, E. L. C., & Hvannberg, E. T. (2004). Analysis of combinatorial user effect in international usability tests. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 9-16). ACM.
Retrieved from http://dl.acm.org/citation.cfm?id=985694

Lin, Yu-Wei (2011). A qualitative enquiry into OpenStreetMap making. *New Review of Hypermedia and Multimedia, 17*(1), 53-71.
Retrieved from http://dl.acm.org/citation.cfm?id=1970253

Moseme, M. T., & van Elzakker, C. P. J. M. (2012). *Neogeography Map Users and Uses*. Paper presented at AutoCarto 2012.
Retrieved from http://www.cartogis.org/docs/proceedings/2012/Moseme_VanElzakker_AutoCarto2012.pdf

Nedović-Budić, Z., & Budhathoki, N. R. (2010). *Motives for VGI Participants* (presentation). Paper presented at VGI for SDI Workshop, Netherlands Geodetic Commission/Wageningen University.
Retrieved from http://www.ncg.knaw.nl/Studiedagen/10VGIforSDI/presentaties/Zorica_Nama_VGI_for_SDI.pdf (accessed 2013-04-03)

Neis, P., Zielstra, D., & Zipf, A. (2012). The street network evolution of crowdsourced maps: OpenStreetMap in Germany 2007–2011. *Future Internet, 4*(1), 1-21.
Retrieved from http://www.mdpi.com/1999-5903/4/1/1

Nielsen, J. (1992). The usability engineering life cycle. *Computer, 25*(3), 12-22.
Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=121503

Nielsen, J. (1994a). *Usability engineering*. Elsevier.
Retrieved from http://books.google.de/books/about/Usability_Engineering.html?id=95As2OF67f0C

Nielsen, J. (1994b). Usability inspection methods. In *Conference companion on Human factors in computing systems* (pp. 413-414). ACM.
Retrieved from http://dl.acm.org/citation.cfm?id=260531

Nielsen, J. (2006). *Participation Inequality: Encouraging More Users to Contribute*.
Retrieved from http://www.nngroup.com/articles/participation-inequality/

Nielsen, J., Clemmensen, T., & Yssing, C. (2002). Getting access to what goes on in people's heads?: reflections on the think-aloud technique. In *Proceedings of the second Nordic conference on Human-computer interaction* (pp. 101-110). ACM.
Retrieved from http://dl.acm.org/citation.cfm?id=572033

Nielsen, J., & Molich, R. (1990). Heuristic evaluation of user interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 249-256). ACM.
Retrieved from http://dl.acm.org/citation.cfm?id=97281

Ramm, F., Topf, J., & Chilton, S. (2011). *OpenStreetMap. Using and Enhancing the Free Map of the World*. Cambridge, England: UIT Cambridge.

Rieman, J., Franzke, M., & Redmiles, D. (1995). Usability evaluation with the cognitive walkthrough. In *Conference companion on Human factors in computing systems* (pp. 387-388). ACM.
Retrieved from http://dl.acm.org/citation.cfm?id=223735

Schmidt, M., Klettner, S., & Steinmann, R. (2013). *Barriers for Contributing to VGI Projects*. Paper presented at the 26th International Cartographic Conference.
Retrieved from http://publik.tuwien.ac.at/files/PubDat_218906.pdf

Tang, J. C., Liu, S. B., Muller, M., Lin, J., & Drews, C. (2006). Unobtrusive but invasive: using screen recording to collect field data on computer-mediated interaction. In *Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work* (pp. 479-482). ACM.
Retrieved from http://dl.acm.org/citation.cfm?id=1180948

Turner, A. (2006). *Introduction to neogeography*. O'Reilly Media, Inc.
Retrieved from http://pcmlp.socleg.ox.ac.uk/sites/pcmlp.socleg.ox.ac.uk/files/Introduction_to_Neogeography.pdf

Virzi, R. A. (1992). Refining the test phase of usability evaluation: how many subjects is enough?. *Human Factors: The Journal of the Human Factors and Ergonomics Society, 34*(4), 457-468.
Retrieved from http://coursesite.uhcl.edu/hsh/peresSC/Classes/PSYC6419seminar/Refining%20the%20test%20phase.pdf (accessed 2014-03-31)

Zielstra, D., & Zipf, A. (2010). A comparative study of proprietary geodata and volunteered geographic information for Germany. In *13th AGILE international conference on geographic information science* (Vol. 2010).
Retrieved from http://agile2010.dsi.uminho.pt/pen/shortpapers_pdf/142_doc.pdf

All URLs were accessed and checked on 2014-08-06 (unless indicated otherwise).

URLs

URL 1 OpenStreetMap. http://www.openstreetmap.org/

URL 2 OpenStreetMap stats report. http://www.openstreetmap.org/stats/data_stats.html

URL 3 Wikipedia. http://www.wikipedia.org/

URL 4 Why (and how) we've switched away from Google Maps – Nestoria Blog. http://blog.nestoria.co.uk/post/43883369968/why-and-how-weve-switched-away-from-google-ma

URL 5 Around the world and back again – Flickr Blog. http://blog.flickr.net/en/2008/08/12/around-the-world-and-back-again/

URL 6 Skobbler. http://www.skobbler.com/

URL 7 Uncharted Territory: The Power of Amateur Cartographers. http://www.wired.com/wiredscience/2013/08/power-of-amateur-cartographers/

URL 8 OpenStreetMap: The Maps In Your Apps Are About To Get A Lot Better. http://readwrite.com/2013/05/07/openstreetmaps-the-maps-in-your-apps-are-about-to-get-a-lot-better

URL 9 iD In-Browser Editor Now Default on OpenStreetMap. http://blog.openstreetmap.org/2013/08/23/id-in-browser-editor-now-default-on-openstreetmap/

URL 10 Editor usage stats – OpenStreetMap Wiki. http://wiki.openstreetmap.org/wiki/Editor_usage_stats

URL 11 Main Page – OpenStreetMap Wiki. http://wiki.openstreetmap.org/wiki/Main_Page

URL 12 Copyright and License. http://www.openstreetmap.org/copyright

URL 13 MapQuest Open. http://open.mapquest.com/

URL 14 Waymarked Trails: Hiking. http://waymarkedtrails.org/en/

URL 15 OpenSeaMap. http://openseamap.org/

URL 16 GEOFABRIK Downloads. http://www.geofabrik.de/data/download.html

URL 17 Overpass API. http://overpass-api.de/

URL 18 Maperitive. http://maperitive.net/

URL 19 Osmosis – OpenStreetMap Wiki. http://wiki.openstreetmap.org/wiki/Osmosis

URL 20 Humanitarian OpenStreetMap Team. http://hot.openstreetmap.org/

URL 21 OSM Tasking Manager. http://tasks.hotosm.org/

URL 22 JOSM. https://josm.openstreetmap.de/

URL 23 Verifiability – OpenStreetMap Wiki. http://wiki.openstreetmap.org/wiki/Verifiability

URL 24 Stats – OpenStreetMap Wiki. http://wiki.openstreetmap.org/wiki/Stats

URL 25 OpenStreetMap editor Potlatch 2 launched. http://blog.openstreetmap.org/2010/11/29/openstreetmap-editor-potlatch-2-launched/

URL 26 Hamburg – Hamburger OSM Interessierte. http://lists.openstreetmap.de/mailman/listinfo/hamburg

URL 27 Talk-de – Openstreetmap allgemeines in Deutsch. https://lists.openstreetmap.org/listinfo/talk-de

URL 28 talk – OpenStreetMap user discussion. https://lists.openstreetmap.org/listinfo/talk

URL 29 Field Papers. http://fieldpapers.org/

URL 30 iD editor – LearnOSM. http://learnosm.org/en/editing/id-editor/

URL 31 openstreetmap-website – GitHub. https://github.com/openstreetmap/openstreetmap-website

URL 32 FFmpeg snapshot. http://ffmpeg.org/releases/ffmpeg-snapshot.tar.bz2

URL 33 Cinelerra package archive. https://launchpad.net/~cinelerra-ppa/+archive

URL 34 ATLAS.ti. http://www.atlasti.com/index.html

All URLs were accessed and checked on 2014-08-06.