Using Spatial and Temporal Editing Patterns for Evaluation of Open Street Map Data

by

Simon Gröchenig

Master's Thesis

Submitted in partial fulfillment of the requirements of the degree
Master of Science in Engineering

Spatial Information Management

Carinthia University of Applied Sciences

School of Engineering & IT

Department of Geoinformation & Environmental Technologies

Supervisors:

FH-Prof. Mag. Dr. MSc., MAS Gernot Paulus
Carinthia University of Applied Sciences

Dr. Karl Rehrl und Dipl.-Ing. Andreas Wagner
Mobile und Web-basierte Informationssysteme
Salzburg Research Forschungsgesellschaft m.b.H.

Prof. Mag. Dipl.-Ing. Dr. techn. Hartwig Hochmair
Fort Lauderdale Research and Education Center (FLREC)
University of Florida

Salzburg, 08/30/2012

Science Pledge

By my signature below, I certify that my thesis is entirely the result of my own work. I have cited all sources I have used in my thesis and I have always indicated their origin.

Salzburg, 08/30/2012 *Simon Gröchenig*

Place, Date Signature

Abstract

Volunteered Geographic Information (VGI) (Goodchild, 2007) has become a widely used alternative to commercial datasets for a variety of geo-applications. The OpenStreetMap (OSM) project, which has the goal to create a detailed map of the world based on VGI in vector data format, is one of the most prominent web projects. The OSM database is collected by voluntary data contributors and therefore not governed by an authoritative agency. Due to the heterogeneity of contributors, data validation is crucial to warrant usability of OSM data for spatial applications. Previous research assessed OSM data quality primarily through comparison with commercial or governmental reference data sets (Hacklay 2010; Zielstra and Hochmair 2011). This work presents a novel intrinsic approach for quality evaluation, which means that only OSM data itself, more specifically, the editing history of features, is used for the evaluation.

The evaluation method uses an activity-action-operation hierarchy for describing edits, which is based on the activity theory defined by Kuuttii (1996). After downloading the OSM features and their history files for a selected area basic mapping operations for features are identified. Basic editing operations include (1) node coordinate changes, (2) modifications of the node list for ways, (3) the member list of relations, and (4) updated tags. Also special operations, such as “Way Split” or “Feature Recreation” are supported. Next, the basic operations are sorted by time and mapper and aggregated to actions based on a set of action rules. For example, the “Create Line” action must have (1) a “Create Way”, (2) at least two “Add Node”, (3) and some “Add Attribute” operations. Each action is linked to an activity which is based on the ISO 19157 quality parameter along with a quality value. Activities are Improvement of Positional Accuracy, Improvement of Thematic Accuracy, Improvement of Completeness and Improvement of Logical Consistency. It is assumed that a heavily edited feature has a better quality in comparison to a poorly editing feature. Thus, summing up the quality values of all editing actions of all features could give some indication on the general quality level of the selected area.

The evaluation method produces an output which shows how many actions are found for which feature type (e.g., highway, amenity, landuse, or building). Since each action is linked to a quality parameter, the quality of different feature types, measured along the five quality parameters, can be assessed for the selected area. The algorithm uses a set of time ranges for analyzed edits, which is used to identify temporal patterns in quality improvement for different feature types.

Keywords

Volunteered Geographic Information, Open Street Map, Data Quality, Editing Action

Acknowledgements

I would like to use this page to say thank you to the people which have been involved in the work for this thesis.

At first I would like to thank my partner company Salzburg Research and especially Dr. Karl Rehrl for their continuous support throughout all phases of my thesis from June 2011, when I met him for an initial talk, until now.

Next, I want to thank my university-internal supervisor Dr. Gernot Paulus for the feedback and support. He also established the link to Dr. Hartwig Hochmair at the Fort Lauderdale Research and Education Center (University of Florida). I went to Florida between February and May and enjoyed a huge amount of valuable input for my thesis by Dr. Hochmair, but I also enjoyed the possibility to learn another culture and new friends.

In this context, I also want to express my gratitude to the Austrian Marshall Plan Foundation, who provided the generous financial support for my research stay in the USA.

Finally, I want to say thank you to my family for all their help during the last five years.

Table of contents

List of Abbreviations

Ac	Action
Acc	Accuracy
API	Application Programming Interface
CC-by-SA	Creative Commons Attribution-ShareAlike 2.0
DB	Database
GPS	Global Positioning System
HTTP	Hypertext Transfer Protocol
ISO	International Organization for Standardization
JOSM-Editor	Java-OpenStreetMap-Editor
km	Kilometer
m²	square meters
Op	Operation
ODbL	Open Database License
OSM	Open Street Map
POI	Point Of Interest
REST	Representational State Transfer
VGI	Volunteered Geographical Information
XML	Extensible Markup Language

1 Introduction

The first section gives an overview of this thesis. It consists of motivation, goals, overview of methods, expected results, audience and document structure.

1.1 Motivation

The OpenStreetMap (OSM) project aims to develop a complete map of the world (Bennett, 2010). Since the project start in 2004, geographic features have been recorded and integrated into the OSM database. Volunteers collect streets, buildings, forests and water bodies, power lines, post-boxes and more. Also abstract features which are not visible (e.g. bus routes, administrative boundaries, land use) are documented. OSM data is available in the internet[1] and is distributed via the Creative Commons Attribution-ShareAlike 2.0 (CC-by-SA) (until summer 2012) and the Open Database License (ODbL) (since summer 2012) (OSM Foundation, 2012a). As the coverage and feature quality improves, the map is a freely available and a world-wide alternative to commercial maps (Mondzech and Sester, 2011).

Everybody is able to add or update features, so the data is coming from many different sources. Also non-expert users are allowed to record data sets (Bennett, 2010). Therefore, data quality is an issue. Most previous work compared OSM data to commercial or third-party datasets for assessing quality. But as the quality of those external datasets is not known, this method provides only limited insight into OSM data quality.

Examples of application areas of OSM are route planning and accessibility analysis. As many foot paths and bicycle facilities are included in the dataset, OSM is also considered as a data source for pedestrian and cyclist related projects (Zielstra and Hochmair, 2011). High data quality is essential in order to provide a user with the correct directions.

Many map-producing organizations lost motivation in updating their maps during the last decade (Goodchild, 2007). Reasons are the rising cost and the availability of remote sensing. However, maps are still indispensable in everyday life; therefore even commercial data providers rely of data input from individual volunteers (Goodchild, 2007). For example, Google Maps added a new service called Map Maker[2] which allows everyone to improve their maps. OSM is a more appropriate example as the maps are built by volunteers exclusively.

Some companies make decisions in favor of OSM. Microsoft's Bing Maps can be used to digitize features. Lately, ESRI developed an add-on for ArcGIS, which provides tools to edit OSM data (ESRI, 2012). Apple has been using OSM data within their application iPhoto since March 2012 (OSM Foundation, 2012b). In addition to this, companies restrict the usage of

[1] http://www.openstreetmap.org
[2] http://www.google.com/mapmaker?output=html

competing products, which makes OSM even more attractive. Google changed the license agreement for their application Google Maps. They introduced a daily limit of 25,000 map loads of the free license (Google, 2012). As a result many customers change from Google Maps towards OSM. As the number of OSM users is increasing, data quality is becoming even more important.

1.2 Goals & Research Problems

The goal is to find a method which evaluates the OSM data quality for a defined area. In order to avoid dependencies on third-party datasets, a direct evaluation method should be developed which assesses the quality by analyzing the features within the dataset (ISO, 2011). In this work, this so-called intrinsic approach should extract editing histories of the features. The idea is to find editing actions which manipulate the OSM data set and attribute quality parameters to these actions. The primary research question of this work is to find out if the editing history can be used to assess data quality of OSM features. In order to prove the appropriateness of the method, data evaluation will be conducted for OSM datasets from different geographical regions and time spans.

1.3 Methods of Solution

The method to extract editing actions from the OSM dataset is based on the Conceptual Framework of Activity theory (Kuutti, 1996). This framework introduces an operation-action-activity hierarchy. Operations represent basic data changes which can be found in the OSM history (e.g. changing the coordinates of a node, setting a new attribute). Associated operations are aggregated to actions (e.g. creating a way and adding the way points). Third, the actions are linked to activities. Activities represent action sets which have been executed with the goal to improve certain quality parameters defined by the ISO 19157 standard (completeness, logical consistency, positional-, thematic-, and temporal accuracy). Activity representations are prepared for quality evaluation. A quality matrix which allows filtering by actions, activities or features types (e.g. highway, building, and natural) is created along with some statistical characteristics of the dataset, such as node density. These results will be used to determine the usability of the algorithm in order to prove research questions.

Alternative intrinsic methods which have been described by third-party publications are explained in the following list:

(a) Analyze the current map without comparison to previous versions (e.g. Mooney, 2010). Suggested quality indicators are among others the node density in lines/polygons, the general node density, or the number of tags.
(b) Maps of the same area and different time stamps can be compared (e.g. Neis et al, 2012). An example is the set of dates 2010-01-01, 2011-01-01 and 2012-01-01. Changes

in-between are identified and analyzed in a similar way to this work. For example the number of new features between two dates can be determined.

The difference between this work and method (b) is the better accuracy of the results since also the editing actions between two timestamps are considered. Additionally the total number of all editing actions, mappers and feature modifications can be determined.

1.4 Expected Results

The literature review reveals a comparison between data quality evaluation methodologies in the context of VGI. It contains (a) the procedure, (b) the quality criteria used, (c) the region, and (d) the year of the evaluation.

Based on this review, a criteria catalogue which includes the quality attributes for this work is derived. The quality attributes are the editing actions which are extracted from the OSM data. A three-tiered hierarchy (operation-action-activity) ensures a transparent and modular design, which can be adjusted easily in order to meet the requirements for any data evaluation project.

A prototypical workflow of the algorithm describes the process of evaluating OSM data. This workflow covers extracting and interpreting editing actions and the preparation of the output. The results are prepared in a way that a filter can be applied on actions, activities and/or feature types. Additionally, some statistical numbers reveal further information (e.g. number of features, node density).

The algorithm is tested using quality indicators. The test areas are selected based on third-party evaluations and different node densities. It is assumed that areas with a high node density and feature type variety have a better quality. In order to produce comparable results, city centers with a known high feature density in four different towns (urban and rural environments) are chosen. The evaluation results for these areas will be analyzed, which confirms or rejects the correct functionality and purpose of the algorithm.

1.5 Audience

Everyone who wants to know if OSM data is useable for a specific region and purpose can use the algorithm.

1.6 Structure of the Thesis

The next section gives the problem definition and the research questions. Section 4 gives an insight into the current situation in the field of VGI quality evaluation. Together with theoretical and technical background information, a state of the art analysis is done. Afterwards, the algorithm is discussed in detail. The approach, concept and workflow explain how the algo-

rithm is implemented. Section 6 introduces the evaluation protocol. Section 7 validates the algorithm and provides conclusions, which is followed by aspects of future work.

2 Problem Definition and Research Questions

VGI datasets can are modified by everyone, therefore their dataset is updated permanently. This special case requires innovative methods in order to evaluate the data. There are extrinsic and intrinsic data evaluation methods (ISO, 2011). While research studies compare the VGI data with third-party datasets (extrinsic approach), this work follows an intrinsic approach to determine the data quality of OSM data based on the editing history of the features. Until now, no other work performed a data quality evaluation based on the features' editing history (see 4.4 Related Work).

Research questions

- Which quality criteria can be considered for an intrinsic evaluation of VGI?
- Which methods can be applied to analyze the relationship between achieved quality and typical editing patterns?
- Can these algorithms be applied to determine data quality in different spatial and/or temporal dimension? If yes, which quality indicators can be identified?

Objectives

These research questions will be addressed through the following objectives:

- Development of a methodology which extracts the editing actions from the OSM dataset of a defined area and maps those editing actions to quality criteria.
- Selection of similar evaluation areas (e.g. urban areas) with different levels of OSM data completeness to quantify data quality differences using the developed methodology.

3 Methodology

The work on the evaluation method is separated in four main steps, which are illustrated in Figure 1.

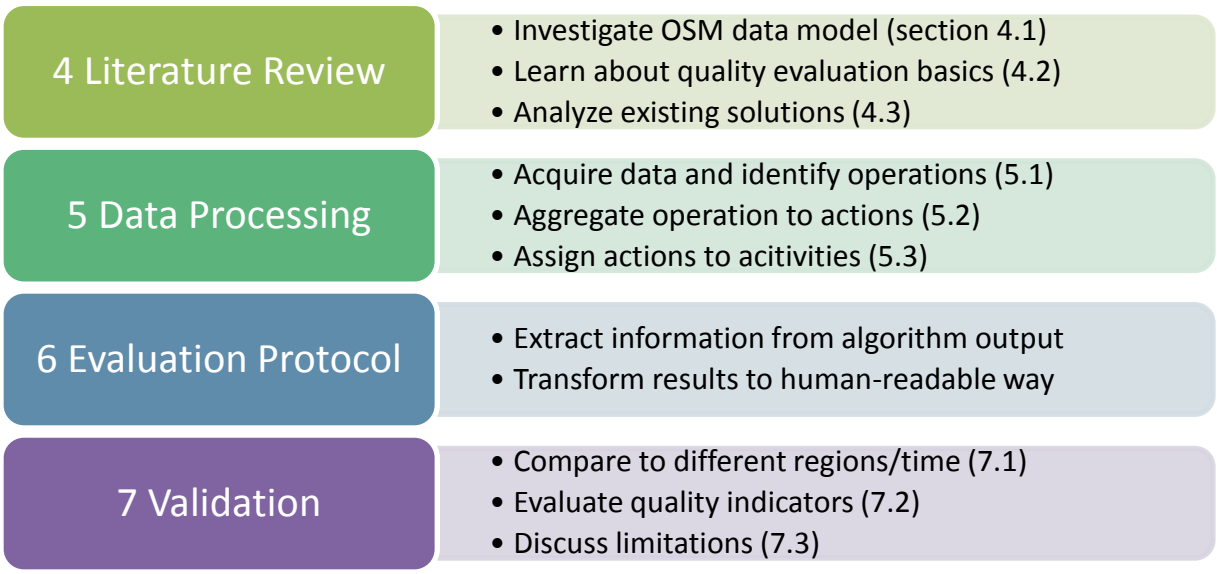


Figure 1: Work steps

The literature review collects state-of-the-art concepts for the subject of quality evaluation of VGI. This review results in a list of regions, which have been tested in regard to the data quality, are listed along with the applied methodology. Later, the OSM data evaluations found in the survey are used to find regions with a good data quality. These regions can be used for further rating comparisons. In parallel, methods to detect all OSM editing actions (create, modify or delete feature) and other quality criteria (like point density) are investigated. This work step is covered by section 4.

The data processing task explains how the editing actions are identified. This process is based on the activity theory introduced in section 4.3.3. The questions which OSM data is necessary and which operations can be extracted from it are answered. Next, rules for aggregating operations to actions are defined. Third, the actions are linked to activities by assigning them to an ISO 19157 quality parameter, which includes (1) positional accuracy, (2) thematic accuracy, (3) completeness, and (4) logical consistency (International Organization for Standardization, 2011). Activities involve the improvement of the respective quality parameters. Apart from introducing the process, the related workflow will be described. It reveals how to download data from OSM, how to extract operations, how to aggregate operations to actions and how to assign actions to activities with a calculated quality value.

The evaluation protocol step prepares the outcome of the data preprocessing for the evaluation. Information which is necessary to evaluate the data quality is extracted and arranged. The quality values can be filtered by actions, activities and/or feature types.

OSM datasets for selected urban test regions in Austria (see section 7.1 Evaluation test area) are evaluated using the previously developed evaluation module. The algorithm is calibrated and validated in order to provide well-grounded results. Based on a set of quality indicators, the selected areas are evaluated. Quality differences between the regions and time spans are identified. This algorithm validation can use reference datasets (contrary to dataset evaluation) to learn about expected results.

4 Literature Review and State of the Art

This section gives an overview of VGI, especially on OpenStreetMap as well as theoretical and technical background knowledge related to the subject of this thesis. It also provides a review of related work.

4.1 Volunteered Geographic Information

This section explains the term Volunteered Geographic Information and gives the details about the technical background of OSM.

4.1.1 Volunteered Geographic Information

During the last decade many new services were developed which allowed every person to collect and share spatial data. Professionals and also many more semi-skilled contributors produce geographical data voluntarily, which leads to good and bad results. Both producers and users work on the maps (Coleman et al, 2009). Goodchild (2007) introduced the term Volunteered Geographic Information (VGI) for this kind of data. This trend has a massive influence on GIS in general (Goodchild, 2007). Projects like OpenStreetMap, Wikimapia[3], OpenAddresses[4] or Flickr[5] follow the primary example of the encyclopedia Wikipedia.

The change to Web 2.0 which allows internet users to edit the content of a website was the essential factor in favor of VGI. By using devices which can access the Global Positioning System (GPS) geographical features can be recorded on site and uploaded to the database later. Another data acquisition method is using an existing dataset like a satellite image and tracing the features (Goodchild, 2007).

4.2 OpenStreetMap

This section gives information about the OSM data model, the OSM API and the OSM data format.

4.2.1 OSM data model

The data model describes how the features are stored in the OSM database (Ramm and Topf, 2010). It is also the basis for the OSM data format which is explained later. The model provides tools to describe real-world objects. Figure 2 shows the OSM data model (Ramm and Topf, 2010).

[3] http://wikimapia.org/
[4] http://openaddresses.org/
[5] http://www.flickr.com/

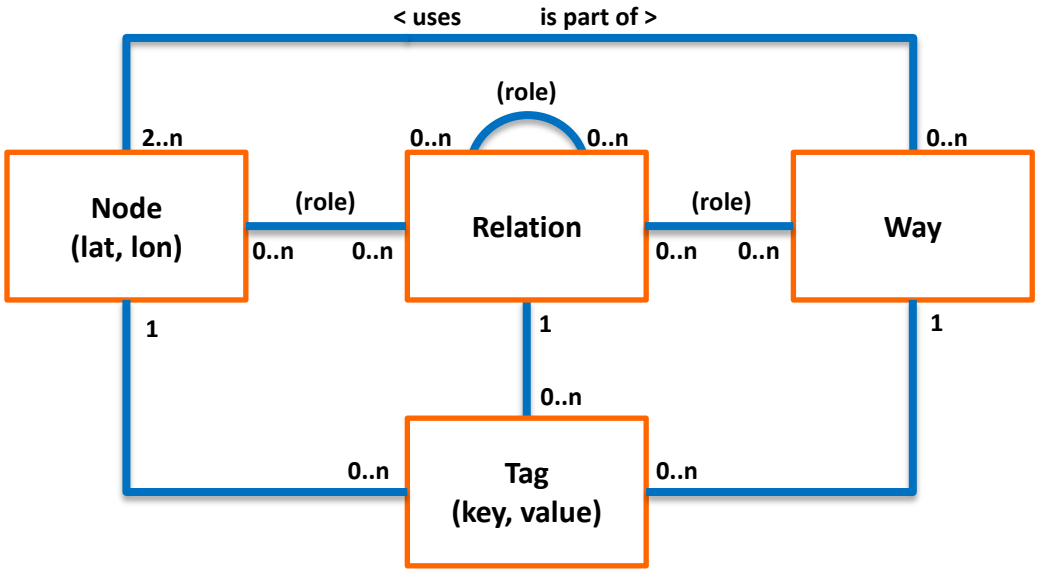


Figure 2: OSM data model (Ramm and Topf, 2010, simplified)

The basic element for all OSM geometries is the node. It possesses a coordinate-pair, a set of tags and information about the last change (user, timestamp, changeset). A tag consists of a key and a value and is used to store information about the geometry. Nodes can be used for Point Of Interests (POI) and as a vertex for a line (way). Two or more nodes can build a way. A way possesses an ordered list of nodes, a set of tags and information about the last change. If the first node in the list equals the last node and if the tags are suitable, the way is interpreted as a polygon. As the name suggests, relations create relations between members (nodes, ways and/or relations). A relation consists of an ordered list of members, a set of tags and information about the last change. Each member has a role (Ramm and Topf, 2010).

Node

A node is a point with a coordinate. It is the only OSM element which contains geographic information. Lines and polygons consist of nodes. A node contains the following information:

- Longitude and latitude
- Information about the last change (user, user id, changeset id, timestamp, version)
- Visibility (is feature deleted?)
- Set of tags

Nodes are used for the following purposes:

- POI feature (node must contain a primary tag for defining the type of the feature)
- Way node (node must not contain any tags); node is part of the node list
- POI and way node (node contains a primary tag and is part of a way)
- Unconnected and untagged node (node is no member of way or relation and has no tags; therefore it is useless for the map)

Way

Ways are used to form lines and polygons. An ordered list of nodes is used to build the feature. If the first node equals the last node, the way is closed, otherwise it is open. Ways have a direction. It depends on the tags if the direction is meaningful (e.g. one-way-streets versus normal streets). A way contains the following information:

- Ordered list of nodes
- Information about the last change (user, user id, changeset id, timestamp, version)
- Visibility (is feature deleted?)
- Set of tags

A way can be used to form the following kinds of features:

- Open polyline (e.g. road)
- Closed polyline (e.g. roundabout)
- Area (e.g. building)
- Closed-polyline and area (e.g. roundabout with grass within the circle)

Relation

Relations represent relations between features. Nodes, ways and other relations can act as members. Every member also has a role in order to specify its position within the relation. The members are ordered. A relation contains the following information:

- Ordered list of members (nodes/ways/relations); each member has a role
- Information about the last change (user, user id, changeset id, timestamp, version)
- Visibility (is feature deleted?)
- Set of tags

Every relation also has a type tag which determines its purpose. Table 1 lists all existing relation types and explains their usage (Ramm and Topf, 2010; OpenStreetMap Wiki, 2012d).

Table 1: Relation types

OSM relation type	**Relation description**
multipolygon	Complex polygon and multi-polygons
boundary	Connection of multiple way features (administrative boundaries, land usage zones)
associatedStreet relatedStreet street	Routes (streets, roads, bus lines) which use multiple ways
route public_transport	Network information (rail and bus lines)
restriction	Turn restrictions (e.g. only_left_turn, only_straight_on, no_right_turn)
destination_sign	Destination signs (e.g. Millstatt, 15km) [2]

OSM relation type	Relation description
bridge tunnel	Define bridge/tunnel combinations
dual_carriageway [1]	Connect both motorway directions
junction [1]	All junctions elements can be merged
enforcement	Traffic enforcement elements like speed traps [2]
site	Merge elements like a school [2]
waterway	Merge waterway elements [2]

[1] Ramm and Topf, 2010, proposed types; other types: OpenStreetMap Wiki, 2012d
[2] OpenStreetMap Wiki, 2012d; other descriptions: Ramm and Topf, 2010

Tag

Tags store information about the features. Nodes, ways and relations can have an arbitrary number of tags. A tag consists of a key-value pair. A feature cannot have two tags with the same key. Keys and values always have the data type string. Primary tags define the feature type, while all other tags give additional information about the feature. Primary tags are tags with a special key. The following list contains keys, which can be combined with different values to primary tags (OpenStreetMap Wiki, 2012c):

- Aerialway
- Aeroway
- Amenity
- Barrier
- Boundary
- Building
- Craft
- Emergency
- Geological
- Highway
- Historic
- Landuse
- Leisure
- Man Made
- Military
- Natural
- Office
- Places
- Power
- Public Transport
- Railway
- Route
- Shop
- Sport [1]
- Tourism
- Waterway

[1] use with landuse or leisure

Every feature must have a tag with a key from this list along with a value which specifies the feature type. Other tags which are linked to a feature just give additional information about it.

Feature Versions

When a feature is created, it has version 1. This number is increased by 1 as soon as a mapper modifies and uploads the feature (Ramm and Topf, 2010). It is possible that there are no changes between two consecutive versions. The reason is the behavior of some editors which add unchanged features to a changeset.

OSM changesets

Another key element which is necessary during this work is the changeset. A changeset stores all data modifications done by one user during one session and transports them to the OSM DB (Ramm and Topf, 2010). The OSM map data itself only contains information about the last change. Changesets help to find previous versions of the geometries.

4.2.2 OpenStreetMap API and Data Format

The OSM Application Programming Interface (API) provides access to the OSM data via a Hypertext Transfer Protocol (HTTP) based Representational State Transfer (REST) architecture (Ramm and Topf, 2010). The data format for the transfers is the OSM Extensible Markup Language (XML) format. Everyone is allowed to read data, but in order to write data user authentication is required (Ramm and Topf, 2010).

The following API calls are used in this work:

Map

The map returns the current versions of all features within a specified area. The input parameter is a bounding box (left, bottom, right, and top). The map consists of the following information (Ramm and Topf, 2010):

- All nodes within the bounding box
- All ways which use at least one of those nodes
- All nodes which are used by the ways and which are outside the bounding box
- All relations which use at least one of the nodes or ways

The following XML file gives an example of a map file (shortened version). It includes two nodes and one ways (with two tags).

```
 1  <?xml version="1.0" encoding="UTF-8"?>
 2  <osm version="0.6" generator="CGImap 0.0.2">
 3   <bounds minlat="46.8171545" minlon="13.4949875" maxlat="46.8179584"
          maxlon="13.4960979"/>
 4   <node id="1850144542" lat="46.8206416" lon="13.4964625" user="sharansp90cc"
          uid="501083" visible="true" version="1" changeset="12600241"
          timestamp="2012-08-03T16:31:11Z"/>
 4   <node id="1859776555" lat="46.7978379" lon="13.5007218" user="Twix73"
          uid="478912" visible="true" version="1" changeset="12676080"
          timestamp="2012-08-10T06:55:57Z"/>
 5   <way id="61740817" user="Twix73" uid="478912" visible="true" version="2"
          changeset="9109180" timestamp="2011-08-24T04:44:54Z">
 6    <nd ref="769293746"/>
 7    <nd ref="418304083"/>
 8    <tag k="highway" v="primary"/>
 9    <tag k="name" v="Hauptstraße"/>
10    <tag k="ref" v="B98"/>
11   </way>
12  </osm>
```

Each OSM feature in the file includes the complete information which belongs to the features (see section 4.2.1 OSM data model). Each feature has an id, a changeset id, a user, a user id (uid), a visible attribute, the timestamp and the version. Nodes also have the lat and lon attributes. Ways have an ordered list of nodes and relations have an ordered list of members along with their respective role. Both are stored as child elements of the features. Similarly, tags are also stored as child elements and contain a key (k) and a value (v).

History

The history call is used to get all versions of one feature. Parameters include the element type (node, way or relation) and the feature ID. The returned file lists all feature versions in ascending order starting with the creation of the feature and ending with the current version (Ramm and Topf, 2010). The following XML file is an example of a history file of the node with the id 21099755.

```
1  <osm version="0.6" generator="OpenStreetMap server">
2  <node id="21099755" lat="48.0035327" lon="13.6490052" changeset="156068" us-
       er="cdaller" uid="924" visible="true" timestamp="2006-11-20T07:39:10Z"
       version="1">
3        <tag k="created_by" v="JOSM"/>
4  </node>
5  <node id="21099755" lat="48.0035327" lon="13.6490052" changeset="535999" us-
       er="curmet" uid="40444" visible="true" timestamp="2008-10-23T06:49:33Z"
       version="2"/>
6  <node id="21099755" lat="48.0045354" lon="13.6501235" changeset="535999" us-
       er="curmet" uid="40444" visible="true" timestamp="2008-10-23T07:31:37Z"
       version="3"/>
7  </osm>
```

Changeset

The third API call retrieves changesets. All changes which are performed by one user within a short period of time can be downloaded after sending the changeset ID. Within the XML file, all changes are organized into Create, Modify and Delete blocks. Example: If a mapper removed a feature and replaced it with a new one instantly, both actions will be in the same changeset. Otherwise, if the removal and creation have been done in different changesets, it is not possible to find the old feature. A changeset is created as soon as a mapper starts editing the map, and will be closed if (1) it has 50,000 changes, (2) it has been open for 24 hours, (3) no more changes have been added for one hour, or (4) the mapper closes it manually. Afterwards a new changeset will be created (Ramm and Topf, 2010). The following XML file is an example of a changeset file with the id 294490 (shortened file).

```
<?xml version="1.0" encoding="UTF-8"?>
<osmChange version="0.6" generator="OpenStreetMap server">
  <create>
    <node id="260772163" lat="46.7816787" lon="13.5404679" changeset="294490"
      user="sumpfralle" uid="18325" visible="true" timestamp="2008-04-
      30T16:49:20Z" version="1">
      <tag k="created_by" v="JOSM"/>
    </node>
  </create>
  <delete>
    <node id="190321463" changeset="294490" user="sumpfralle" uid="18325" vis-
      ible="false" timestamp="2008-04-30T16:49:33Z" version="2"/>
  </delete>
  <modify>
    <way id="24045896" visible="true" timestamp="2008-04-30T17:24:13Z" us-
      er="sumpfralle" uid="18325" version="2" changeset="294490">
      <nd ref="260772180"/>
      <nd ref="190322134"/>
      <nd ref="260772179"/>
      <tag k="created_by" v="Potlatch 0.6"/>
      <tag k="highway" v="motorway_link"/>
    </way>
  </modify>
</osmChange>
```

4.3 Theoretical and technical background

In this section, the topics (1) intrinsic data evaluation, (2) Quality Standard ISO 19157 and (3) Conceptual Framework of Activity Theory are discussed.

4.3.1 Intrinsic data evaluation approach

ISO gives the following definition for the direct (intrinsic) evaluation method: "*… method of evaluating the quality of a dataset based on inspection of the items within the dataset*" (ISO, 2011). This means that no external data is acquired. The following three approaches can be used:

- The current state of the map can be analyzed by analyzing the node density or the number of tags.
- Maps of the same area and different timestamps can be compared in order to find differences between them. This static method does not consider editing actions between two timestamps; only the overall changes are analyzed.
- Maps of different areas and/or different time ranges can be compared in order to find differences between them. This dynamic method considers all editing actions between two dates. The evaluation method introduced by this thesis uses this approach.

As opposed to intrinsic method extrinsic evaluation methods compare the own with third-party data. As the quality of the other data set is not always known, only limited results can be achieved.

4.3.2 ISO 19157: Standard for Geographic Data Quality

Before using a set of spatial data, its fitness to use has to be determined. If there are two or more similar data sets involved, the user has to know which one is the most appropriate for the given task. Among others, the International Standardization Organization (ISO) publishes quality standards for spatial data. The current version of the geographic data quality standard is ISO 19157 (ISO, 2011). This version replaces ISO 19113:2002, ISO 19114:2003 and ISO/TS 19138:2006 and merges them. The standard defines a standardized frame for quality assessments. Therefore it introduces five data quality elements: (1) completeness, (2) logical consistency, (3) positional accuracy, (4) thematic accuracy, and (5) temporal accuracy (ISO, 2011).

Completeness

Completeness analyzes the existence of features, attributes and their relationships. There are two elements: (a) Commission is about data, which is in the data set but should not be there; (b) Omission is about data, which is missing in the data set.

Logical consistency

Logical consistency is about the features' compliance with logical rules. The features itself and their attributes and relationships are taken into account.

Positional Accuracy

Positional accuracy defines accuracy of the location of the feature in the dataset compared to the real-world position.

Thematic Accuracy

Thematic accuracy is about the attributes of the features. Quantitative ones should be accurate, qualitative attributes should be correct and the values should be classified correctly.

Temporal Accuracy

Here, the accuracy of the time measurement, the temporal consistency (correct chronological order of events) and temporal validity (e.g. April, 31st) are considered.

4.3.3 Conceptual Framework of Activity Theory

The proposed method for describing edits is based on the conceptual framework of the activity theory (Kuutti, 1996). The idea of this theory is to hierarchically structure tasks into the following concepts:

- Operation: Atomic changes (e.g. adding an attribute, deleting a way point)
- Actions: aggregated operations: a goal (e.g. creating a way)
- Activity: a set of actions which follow a certain motive (e.g. improvement of quality)

Figure 3 shows the basic idea behind this three-tiered framework.

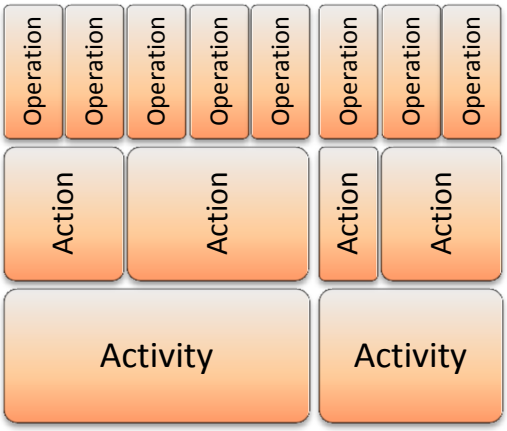


Figure 3: Conceptual framework of activity theory

4.4 Related Work

This section reviews related work in the area of VGI data quality evaluation. Both intrinsic and extrinsic approaches are considered.

Until 2011, the majority of related work used data from third-party datasets to validate the VGI data sets. Results of the introduced works are included in Table 2. Lately, Zielstra and Hochmair (in press) performed shortest path analyses of pedestrian routes and compared the results to those from the commercial datasets TomTom, NAVTEQ, and ATKIS. Also Mondzech and Sester (2011) use a routing algorithm to compare OSM data and the ATKIS data set. Zielstra and Hochmair (2011) compare pedestrian-related network data between free (OSM, Tiger) and commercial (Tele Atlas, NAVTEQ) maps. Haklay (2008) and Ather (2009) compare OSM data with the ordinance survey dataset from Great Britain. Both use a buffer analysis in order to evaluate positional accuracy. Kounadi (2009) also used the buffer method to assess the data, but additionally analyzed the issues completeness and thematic accuracy. Stark (2011) accesses open web mapping services like Google Maps, Bing maps or Yahoo maps in order to compare their data sets with geocoded addresses stored in OpenAdresses.

During the last year, some intrinsic approaches were published. Rehrl et al. (2012) analyzed accessibility between metropolitan areas in Central Europe. Mooney and Corcoran (2012a, 2012b) analyzed characteristics of heavily edited features and collaboration behavior of mappers. A similar mapper exploration was done by Neis and Zipf (2012) and Neis (2012), who also compared OSM maps between Germany and USA. Mooney (2010) calculated the ratio between the length of a way and the number of its nodes and analyzed the annotation tags (data source, description…) of the features.

4.4.1 Comparison

Table 2 shows a comparison of all related research studies. Tested regions, used methods and the results are listed next to the author names. This information is used as a starting point for the analysis in this thesis.

Table 2: Comparison of related work of VGI data quality evaluation (ordered by year)

Author(s)	Region	Method ([I] … intrinsic; [E] … extrinsic)
Rehrl et al. (2012)	Austria, Czech Republic, Slovakia, Hungary	[I] Accessibility analysis between metropolitan areas Result: OSM traffic data is suitable for accessibility analysis
Zielstra and Hochmair (in press)	Germany, USA	[E] Shortest path analysis of pedestrian routes; comparison between OSM and TomTom/NAVTEQ/ATKIS Result: OSM has a better coverage of pedestrian segments compared to commercial dataset for some cities.
Mooney and Corcoran (2012a)	London (Great Britain)	[I] Based on the OSM history of London Mooney and Corcoran try find out if the mappers are collaborating and interacting to each other. Result: Limited collaboration as 35 % of all features have been modified once or twice.
Mooney and Corcoran (2012b)	United Kingdom, Ireland	[I] Features with at least 15 versions are analyzed Result: 79 % of edits involve adding nodes (to ways); no relationship between many edits and many attributes.
Neis et al. (2012)	Germany	[I, E] Analysis of German features and mappers (2007 – 2011) and comparison of street networks with TomTom (TeleAtlas) Result: Quality of OSM data in Germany is very high; estimations that perfect dataset (in comparison to TomTom) can be ready by the end of 2012
Neis and Zipf (2012)	World	[I] Analysis of mapper; how active are they? Where do they come from? Result: 38 % of all mappers have performed at least one OSM edit; and 72 % are located in Europe
Mondzech and Sester (2011)	Urban and rural areas in and around Hannover (Germany)	[E] A routing and accessibility algorithm compares OSM data with ATKIS data. Especially pedestrian related paths are considered Result: Completeness in OSM is worse in rural area, but better than ATKIS in urban areas.

Author(s)	Region	Method ([I] … intrinsic; [E] … extrinsic)
Stark (2011)	Solothurn, (Switzerland)	[E] Comparison of OpenAddresses addresses with Google Maps, Bing maps, and Yahoo maps Result: While the thematic accuracy is good (77 % of all features), positional accuracy bad (less than 50 %)
Zielstra and Hochmair (2011)	USA, Germany	[E] Pedestrian related data, street length comparison with TomTom (TeleAtlas) and NAVTEQ maps Result: In Germany, the overall street length of OSM is better than TeleAtlas whereas the street length in the USA is worse
Mooney (2010)	Wales, Bretagne (France), Ireland, Latvia, Switzerland, Denmark, Estonia, Iceland, Austria, Scotia, Spain, Lower Saxony (Germany)	[I, E] spacing between way nodes of water and forest polygons; analysis of annotation tags (data source, description); extrinsic: shape similarity test (comparison with OSi lakes dataset) Result: Spacing is good in Central Europe, Features in Austria often have a source tag
Hakley et al (2010)	London (city center and suburban)	[E] Positional Accuracy comparison with the Ordinance Surcey dataset; is there a relation between number of users and positional accuracy Result: If there are 15 or more mappers in one km², positional accuracy is good; if there are not more than five, accuracy can be improved
Ather (2009)	Great Britian	[E] Comparison with Ordinance Survey data; positional accuracy with buffer analysis Result: Over 80 % overlap between the two datasets; high thematic accuracy in areas with many mappers
Kounadi (2009)	Greece	[E] Comparison with HMGS dataset (official cartographic service in Greece); positional accuracy with buffer analysis; also completeness and thematic accuracy was assessed Result: Overlap between both datasets (89 %); highway type correctness and name attribute completeness is poor (33 and 26 %)
Haklay (2008)	London (as starting location of OSM), England	[E] Comparison with Ordinance Survey data; positional accuracy with buffer analysis Result: 80 % overlap of highway features; 6 meter positional accuracy on average

4.4.2 Summary

Over the course of the years a shift from extrinsic to intrinsic methods can be observed. Until 2011 OSM data has been compared to (commercial) third-party datasets. In 2012, intrinsic methods to evaluate VGI data sets were established. Both the features and users are analyzed in order to gain information about data quality. In a geographic sense, OSM data quality for some features (e.g. pedestrian-related data) is better in Central Europe (especially Germany) than in USA as there is a large OSM community in Germany and United Kingdom (origin of OSM). OSM has a good data quality in urban areas and in the area of pedestrian- and bicycle-related features.

5 Data Processing

This section describes the second part of the OSM data quality evaluation procedure, which is data processing. After an explanation of the general concept, the single steps are explained in detail. Additionally, the workflow gives an insight into the technical realization. Eventually, the results are presented and explained.

An overview of the data processing is shown in Figure 4. After the user selects a set of OSM objects, the data is downloaded from the OSM DB. All previous versions (the history) of these features are necessary in order to create a complete editing profile. After the data transfer is finished, all basic operations are extracted and ordered chronologically and by user. Next, operations which belong together are aggregated to actions. Now every action is classified to an activity (improvement of one of the ISO quality parameters) and a quality value is calculated. Eventually a quality matrix reveals the evaluation outcome and comparisons between the regions and/or time spans can be done.

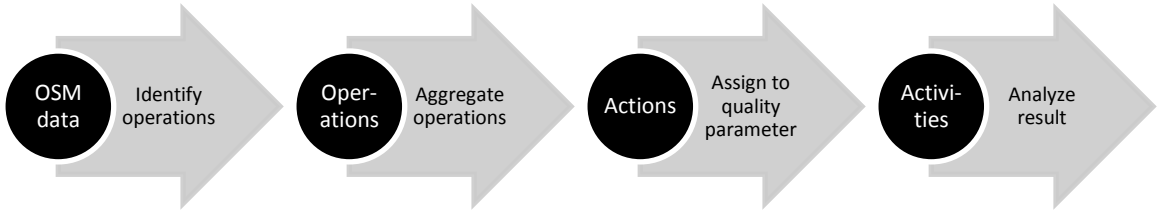


Figure 4: Methodology

Figure 5 shows the relationships between operations, actions, activities and the related items (Rehrl et al., 2012). Operations have a timestamp, are executed by a single person (mapper) and can create, delete or update OSM data. A sequence of operations builds actions, and a set of actions forms activities. Operations, actions and activities are attached to one feature. The results of all features can be merged in order to reveal the overall evaluation outcome.

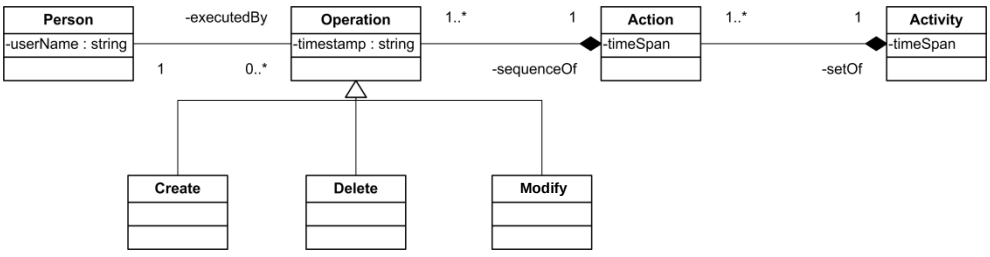


Figure 5: Operations-Action-Activity relationship (Rehrl et al, 2012)

5.1 Identifying Operations

First, the maps and history files (see section 4.2.2 OpenStreetMap API) which contain the editing actions are imported. To identify the operations, the following input parameters have to be defined (see Figure 6): (1a) bounding box of the region or (1b) specific feature, (2)

a set of time ranges, a title (for an easier identification during evaluation), and a time buffer which will be used during action generation.

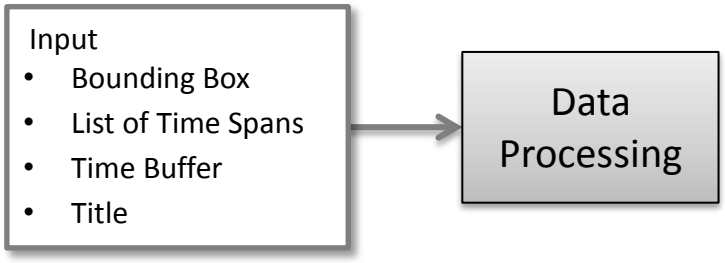


Figure 6: Input parameters

The algorithm is looking for differences between successive versions in order to extract operations. Table 3 gives an overview about the basic operations. They are related to the special characteristics of the elements of the data model. Nodes have coordinates, ways have a node list, relations have a member list and every feature has tags. The basic operations are (1) creating and deleting features, (2) updating node coordinates, (3) adding and removing way nodes, (4) adding and removing relation members, and (5) adding, updating and removing (primary) tags. Italic operations in the table are special operations and are derived from a more complex search method (see section 13.1.5 Other operations).

There are three basic operations which also occur in OSM changeset files: *CREATE*, *MODIFY* and *DELETE* (Ramm and Topf, 2010). These can be applied to every OSM feature. Additionally, Rehrl et al (2012) introduced the Add/Update/Remove classification in order to further detail modify operations.

Create

A new feature with a new feature ID is created. The version of this feature is 1.

Modify

Between creating and deleting a feature, one or more modifications can be performed. These are separated into Add, Update and Remove operations. Modified features have the version 2 or higher.

Delete

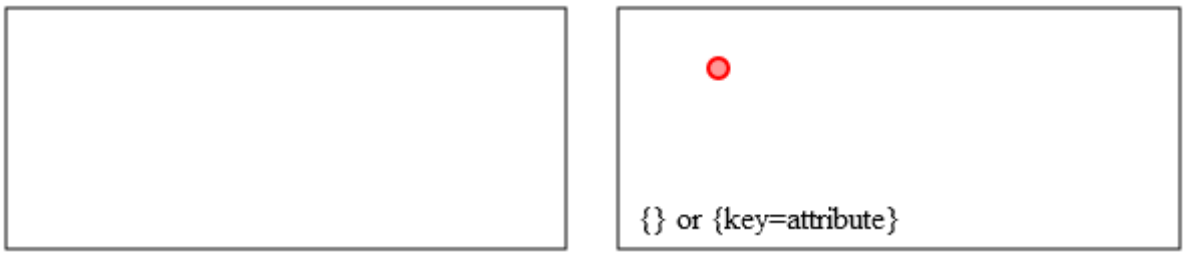
A feature is deleted. This means that the “visible”-attribute is set to false. The feature definition remains in the OSM database. Attributes of the feature like the node’s coordinate, the way’s node list, the relation’s member list or the tags are not included in this feature definition, but can be found by accessing the feature’s previous version. It is possible to recreate the feature by adding a new “visible” version. Deleted features have the version 2 or higher.

Table 3: Overview of basic operations

	Node	Way	Relation	Feature (Node, Way, Relation)
Create	Create Node	Create Way	Create Relation	*Replace Feature*
Modify	Update Coordinate	Add Node Remove Node *Split Way* *Merge Way* *Reverse Way*	Add Member Remove Member	Add Primary Tag Add Tag Update Primary Tag Update Tag Remove Primary Tag Remove Tag *Recreate Feature*
Delete	Delete Node	Delete Way	Delete Relation	

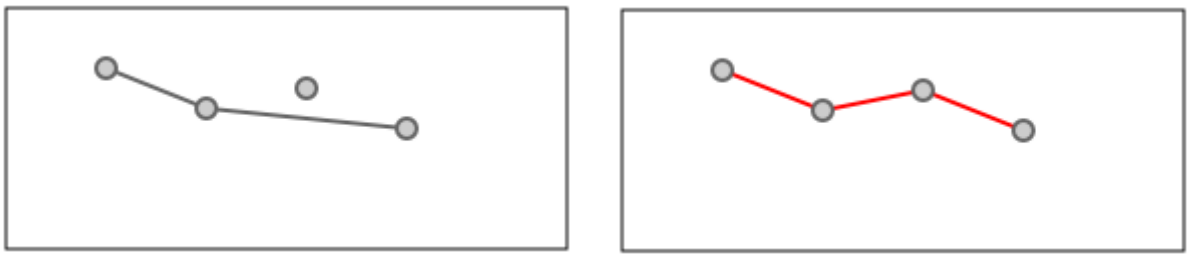
5.1.1 Operations

A complete and detailed description of the operations can be found in the appendix (section 13.1). In general, two consecutive versions of one feature are compared. The differences between them are identified as operations. Here, an excerpt of two operations is given. The first one shows the creation of a node, the second one adds a node to an existing way.



(1) (2)

Figure 7: Create node example



(1) (2)

Figure 8: Modify way (Add node) example

5.1.2 Summary

Figure 9 gives an overview all operations in a class diagram. Blue classes in the figure indicate basic operations, while yellow classes are the other operations. In the reminder of this work, all operations have an “Op”-prefix in order to differentiate them to actions, which have similar names. They are introduced in the next section 5.2.

The diagram classifies all operations to (1) general feature operations, (2) node operations, (3) way operations, and (4) relation operations). Each operation has some attributes (see attributes within class Operation in Figure 9). The feature points to the OSM feature which is affected by the operation. E.g. if the coordinate of a node is updated, this node is the feature. The *FEATURERELATED* attribute saves the OSM feature, which is related to the operation. E.g. if a node is added to a way, the way is the feature, and the node is the related feature. The other two attributes (hierarchy and aggregated) are both linked to actions and are explained in the next section.

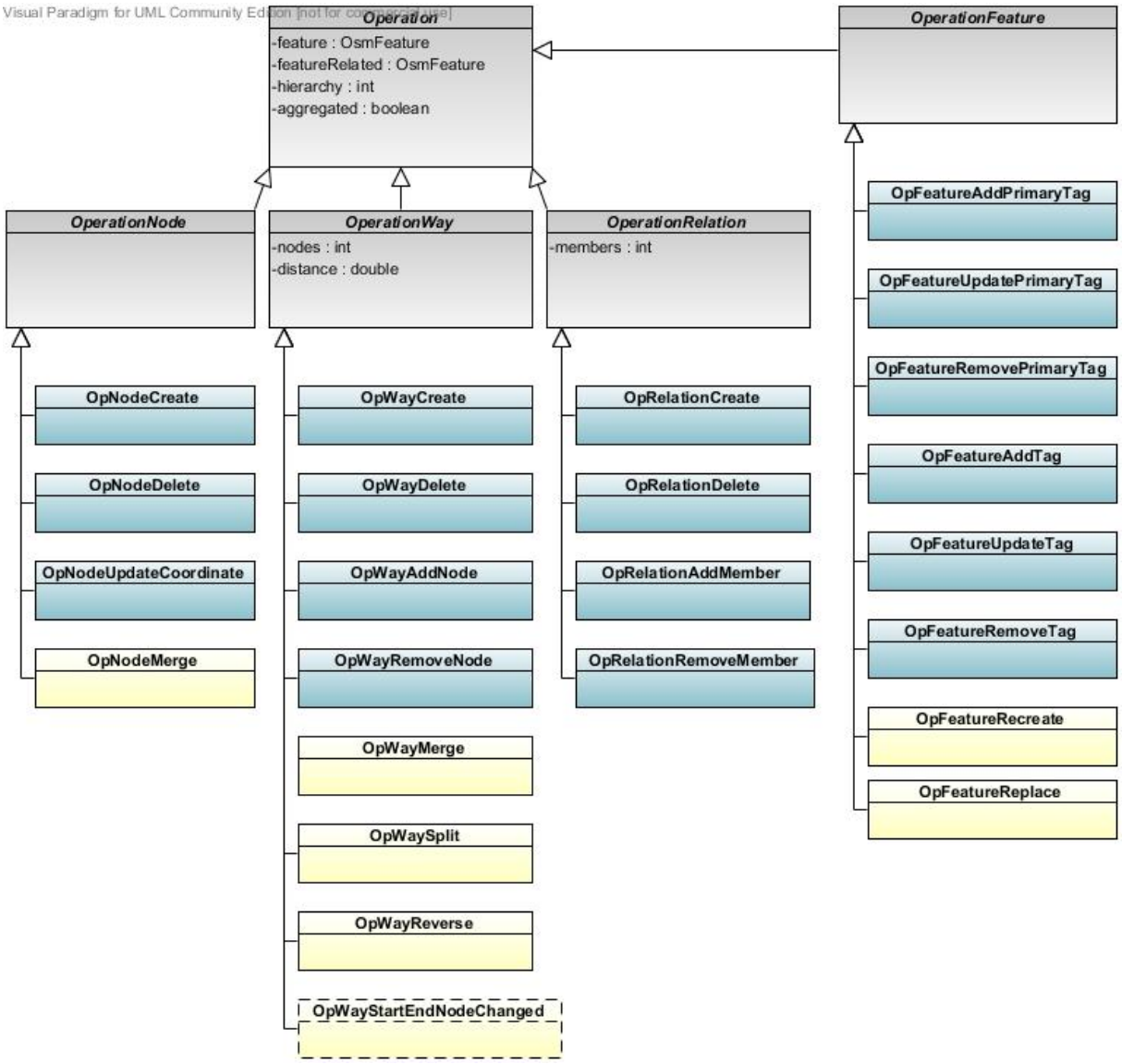


Figure 9: Operations

5.1.3 Examples

The first example (see Table 4) extracts operations from a node feature (ID = `140426086`). The first column contains the OSM features, the second one the extracted type of operations. The first version creates the node with its initial coordinate. The second version updates the

coordinate as the coordinate changes. The third version does not contain any operations; the coordinate remains the same and the feature is still visible. The reason is that some OSM editors add unmodified features to the changeset.

Table 4: Node example: Operation extracting:

OSM feature	Operations
<node id="140426086" lat="46.6160879" lon="13.8435813" changeset="641684" user="wapi001" uid="11034" visible="true" timestamp="2007-11-28T21:07:33Z" version="1"/>	Op Node Create Node
<node id="140426086" lat="46.6160667" lon="13.8435839" changeset="478448" user="pronti" uid="51987" visible="true" timestamp="2008-08-31T21:02:14Z" version="2"/>	Op Node Update Coordinate
<node id="140426086" lat="46.6160667" lon="13.8435839" changeset="6994959" user="pronti" uid="51987" visible="true" timestamp="2011-01-16T22:53:51Z" version="3"/>	no operations

The second example (Table 5) shows the way with the ID 4475775. The displayed feature is simplified (fewer nodes, fewer attributes) in order to improve clarity. The first version creates a way and adds three nodes and two tags. The second row shows another version of the way with an updated node list and more tags.

Table 5: Way example: Operation extracting:

OSM feature	Operations
<way id="4475775" visible="true" timestamp="2007-04-17T22:49:31Z" user="cdaller" uid="924" version="1" changeset="18104"> <nd ref="27435283"/> <nd ref="27435284"/> <nd ref="27435304"/> <tag k="highway" v="unclassified"/> <tag k="name" v="Bahnhofsstraße"/> </way>	Op Way Create (4475775) Op Way Add Node (27435283) Op Way Add Node (27435284) Op Way Add Node (27435304) Op Feature Add Primary Tag (highway) Op Feature Add Tag (name)
<way id="4475775" visible="true" timestamp="2011-01-30T15:43:48Z" user="pronti" uid="51987" version="2" changeset="7135552"> <nd ref="445066391"/> <nd ref="445066399"/> <nd ref="27435304"/> <tag k="addr:city" v="Villach"/> <tag k="addr:country" v="AT"/> <tag k="highway" v="unclassified"/> <tag k="name" v="Bahnhofstraße"/> </way>	Op Way Remove Node (27435283) Op Way Remove Node (27435284) Op Way Add Node (445066391) Op Way Add Node (445066399) Op Feature Add Tag (addr:city) Op Feature Add Tag (addr:country)

5.1.4 Workflow for identifying operations

This section introduces an algorithm which retrieves the necessary OSM data sets and extracts operations from them. Figure 10 shows the associated activity diagram. The process is sub-divided into five groups. The *INITIALIZATION* group selects the evaluation area, downloads the OSM data and prepares the feature lists. The *PREPARATION* section iterates through all features and prepares them. The *EXTRACTION* part, as a sub-group of the *PREPARATION* one, identifies the operations for all versions of the feature. The *EVALUATION* part evaluates the retrieved operations and returns the results.

INITIALIZATION - The first step is to set the input parameters. The first input parameter is the element type. It is possible to select a map region which is defined by a bounding box, or only a feature which is identified by the feature type along with the feature ID. It is possible to select single features in order to prove the evaluation process. Only maps are chosen for the evaluation of regions, which is the ultimate goal of this work. The second parameter a set of time ranges in order to compare the results for multiple time ranges. The third parameter is a time buffer for operation aggregation. If a bounding box has been selected, a map with all features in this region is downloaded, otherwise the selected feature with all associated elements (e.g. nodes for a way, members for a relation) are downloaded. The server sends the selected features in the OSM file format. This file includes all nodes, ways, and relations in this order (Ramm and Topf, 2010). Based on this file, a feature list is generated containing all nodes, ways and relations. Next, all features are copied into a queue, which will be processed in the next steps.

The *PREPARATION* group iterates through all features from the queue. After calling the next element, the version number is analyzed. Only the current version of the feature has been downloaded in the previous *INITIALIZATION* section. If it has version 1, all information is available, otherwise a feature history has to be downloaded from the OSM database. A message is written if the download fails. The versions are ordered descending, because the operation *EXTRACTION* algorithm starts with the current version and iterates through all versions until the first version which marks the creation of the feature is reached. Next, the active way segment is extracted. This affects all versions except the current one. If the way has been longer in an earlier version, the exceeding segment is cut (see Figure 11 for a more detailed explanation). If the current feature is a way, its nodes are linked to the way; similarly members are linked to a relation. The feature list which was generated in the first part only contains features which are not deleted (“visible”-attribute is true) in its latest version. But it is possible that a way in an earlier version has possessed nodes which have been deleted later. In the process of linking nodes to the way, these deleted nodes are found and as a result, added to the feature list and the queue.

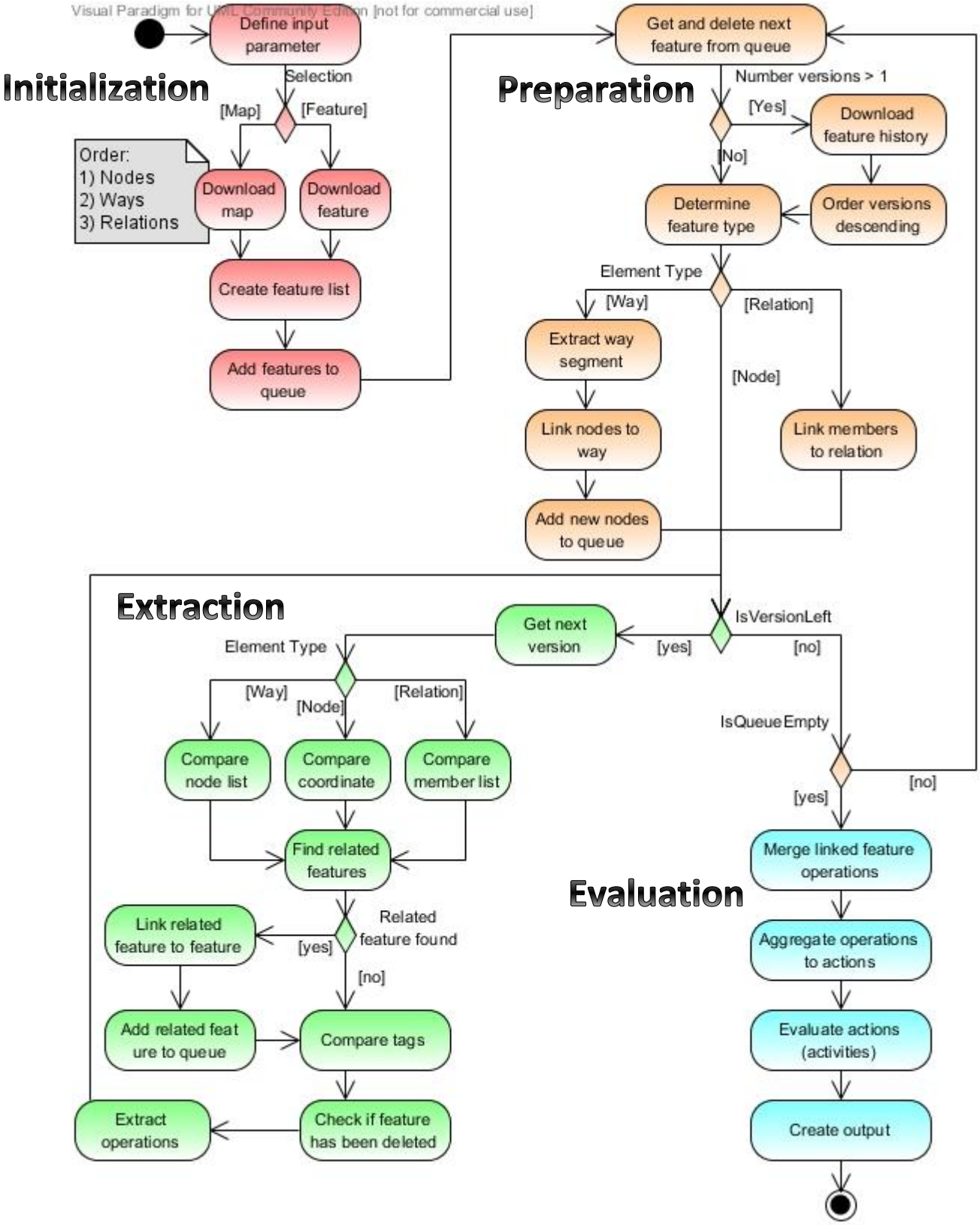


Figure 10: Workflow which shows the data retrieval and operation extracting progress

As a sub-group of the *PREPARATION* part, the *EXTRACTION* part extracts the operations for the current feature. A loop iterates through all versions of the feature, starting with the current one. If the feature is a node, the coordinate is compared, if it is a way, the node list is examined for added and removed nodes and if the feature is a relation, the member list is checked for added and removed members. The next step is to find related features. This includes (a) merged ways, (b) split ways, and (c) replaced features. If a related feature is found, it will be linked the feature and added to the queue for further investigations. Now the tags are compared between two feature versions. The last operation extracting task is to check if the feature has been deleted (i.e., if the "visible"-attribute is set to false). Now, all operations are extracted and collected within the feature. If there are no differences between two consecutive feature versions, no operations are found. This *EXTRACTION* part is repeated until every version is checked for operations. The outer loop (*PREPARATION*) is repeated until the queue is empty, which means that all features have been processed.

The *EVALUATION* part finalizes the evaluation. First, the operations merge linked features' operations. This is about copying the way's nodes *OP NODE UPDATE COORDINATE* operation to the way. If the node is moved, it also affects the way. The next steps in the diagram "Aggregate Operations to Actions" and "Evaluate actions (Activities)" are discussed in the upcoming sections of this work. Finally, the output is created and displayed.

The remaining section gives further details for selected workflow steps:

Link nodes to way

Sometimes, nodes act as a POI and as a way point at the same time. Examples are crossings at streets or entrances of buildings. Later, only features which possess a primary tag (feature type) will be evaluated. Both the node and the way have a primary tag. The problem arises that the node is evaluated twice. In this case the *OP NODE CREATE* operation is linked to the node and the way has an *OP WAY ADD NODE* operation. If the coordinates of the node are moved, both features are influenced by the *OP NODE UPDATE COORDINATE* operation. This is the only operation which is linked to both features. The same situation applies to nodes which are shared between two or more ways.

Extract way segment

The current version of a way sets the way segment which will be examined for the previous versions. If the way had more nodes beyond the start or end node in an earlier version, these excess nodes (not part of this way in current version) are not considered. In Figure 11 the red way is examined. Version 3 is the current version of this way. The blue way is another way. The red way has been shortened twice. As a result only the way segment between the third and fifth node (from left to right) is checked for node list changes. The segment between the second and third segment belongs to the blue way in its current version and therefore will be examined with the blue way. The right segment between the fifth and sixth node is not examined as this segment does not exist the current map.

On the other hand if a start- or end node disappears in an earlier version (which means that the node has been added to the way), the examined way segment will be shortened.

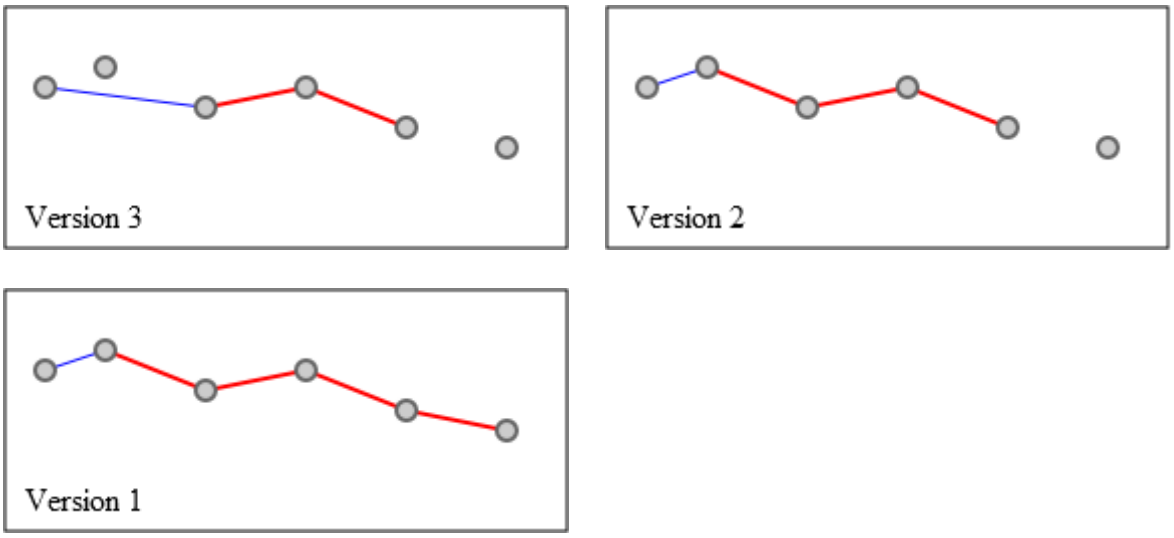


Figure 11: Way cuts

The way versions are compared as long as at least two nodes are in the node list.

Find related features

This section explains how the related features are found. See section 13.1.5 for explanations and examples of the operations.

Way Splitting (Create Way)

- Examined if the *OP WAY CREATE* operation has been found. Only created ways can have this operation.
- Find new attributes, which do not exist in old way
- Approach
  1. Look for start and end node in other ways in same changeset
  2. The way has been split if at least two nodes (of this way) exist in other way's previous version

Way Splitting (Modify Way)

- Examine if the *OP WAY START END NODE CHANGED* operation has been found. The way version is at least 2.
- The *OP WAY SPLIT* operation will be saved at the created way which has been separated from this way.
- Approach
  1. Extract the new way-segment from the previous version

Way Merging

- Examined if the *Op Way Start End Node Changed* operation has been found. The way version is at least 2.
- This operation will be found for the feature which retains the ID.
- Approach
  1. Take start- and end node of this and of the previous version
  2. Look for other ways in the previous changeset which use these nodes
     - Other ways are ways which are modified or deleted in the same changeset (way version at least 2)

Replace feature

By using the current method (OSM API) to download the features, replaced features can only be detected if the deleted feature is in the same changeset as the new one. If this situation does not occur, it is assumed that the feature is created for the first time. The following three items are checked in order to find replaced features:

- Geographical location
- Name
- Feature type

5.2 Aggregating Operations to Actions

Once all operations have been extracted from the OSM data they are processed. The next step is to merge operations which belong together. For example, if a way is created, a set of nodes and also tags is added to the way. The following operations are extracted in that situation: *Op Way Create*, some *Op Way Add Node* and some *Op Feature Add (Primary) Tag*. All those operations will be merged to an action. For this example, the *Ac Create Line/Polygon action* is created. Sometime later, several nodes of this way are moved to new coordinates, which results in *Op Node Update Coordinate* operations. All those operations will be merged to one *Ac Update Line/Polygon Coordinate* action. A set of rules defines which operations are aggregated to which actions. These rules are defined in a modular and hierarchical design and can be modified easily. Only features with primary tags (feature type) will be aggregated. All other features do not have any sense and are ignored for that reason.

What is the difference between actions and operations? Actions are a set of one or more operations. Every action has a highest-level-operation which selected the action and can contain an unlimited number of lower-level-operations. Actions are sets of operations which belong together.

5.2.1 Hierarchy levels

defines a hierarchy level for each operation. The type in the first column of the table indicates the OSM element type which is involved in the operation. Operations with the type F apply to every feature; N means node, W means way and R means relation. These hierarchy levels are used to order the operations before aggregating them to actions. The operation with the highest level within a group determines the actions. E.g. a feature has one *Op Way Create* and two *Op Way Add Node* operations. The *Op Way Create* has the higher level (level 29); therefore this operation will decide which action will be used. *Op Way Add Node* belongs to several actions. If this operation would have a highest hierarchy level, the correct action could not be determined. All hierarchy levels can be adjusted in order to meet the project requirements.

Table 6: Operation hierachy levels

Type	Operation	Hierarchy level
F	Op Feature Recreate	40
F	Op Feature Replace	39
W	Op Way Split	35
W	Op Way Merge	34
N	Op Node Create	30
W	Op Way Create	29
R	Op Relation Create	27
N	Op Node Delete	25
W	Op Way Delete	24
R	Op Relation Delete	23
F	Op Feature Add Primary Tag	15
F	Op Feature Update Primary Tag	14
F	Op Feature Remove Primary Tag	13
F	Op Feature Add Tag	12
F	Op Feature Update Tag	11
F	Op Feature Remove Tag	10
W	Op Way Reversed	7
W	Op Way Add Node	5
R	Op Relation Add Member	5
W	Op Way Remove Node	4
R	Op Relation Remove Member	4
W	Op Way Start End Node Changed	3
N	Op Node Update Coordinate	2

F … Feature; N … Node; W … Way; R … Relation

5.2.2 Actions Definitions

Up to now, the operations have been (1) allocated to groups based on the timestamp and the user and (2) ordered by the hierarchy level within the group. The next step is to aggre-

gate the operations to actions. Table 7 contains examples of action definitions, which contain the aggregation rules. The complete list of action definitions is in the appendix (section 0). The first row gives an explanation of the table content. The numbers in the square brackets indicate how many operations of the particular kind are necessary for one action. Valid values are 1 (one operation), 2 (two operations) and/or ∞ (unlimited). The prefix "Ac" indicates Actions, "Op" is Operations.

The first operation within a group determines the action. Every operation can be the first one. (There is one exception *Op Way Start End Node Changed*, which always comes along with either *Op Way Add Node* or *Op Way Remove Node*.) In order to guarantee a correct assignment, every operation corresponds to an action. After finding the action, all operations which belong to this action are added to the action. For example, an *Ac Feature Replace* action also acquires the *Op Node/Way/Relation Create*, *Op Feature Add Primary Tag*, *Op Way Add Node* and *Op Way Start End Node Changed* operations. Some operations may remain in the group as they are not included in the action definition (e.g. *Op Way Add Tag*). They built up another action based on the highest-leveled operation which is left.

Table 7: Aggregation group examples

Ac Action name • **[Number of operations) Op Operation name (Hierarchy level)**
Ac Feature Replace • [1] Op Feature Replace (39) • [1] Op Node Create • [1] Op Way Create • [1] Op Relation Create • [1] Op Feature Add Primary Tag • [1] Op Way Add Node • [1] Op Way Start End Node Changed
Ac Create Line/Polygon • [1] Op Way Create (29) • [1] Op Node Add Primary Tag • [2..∞] Op Way Add Node • [∞] Op Node Update Coordinate

The hierarchy levels make sure that more significant operations are considered earlier. For example if a way is created, the *Op Way Create* is more important than *Op Way Add Node*. *Op*

WAY ADD NODE operations are later added to the action as a result of the *AC CREATE LINE/POLYGON* action definition.

5.2.3 Example

This is a follow up of the Identifying Operations examples in section 5.1.3.

Only two operations were found for the node example. The two operations *OP NODE CREATE NODE* and *OP NODE UPDATE COORDINATE* are separated by nine months and were made by two different users. Therefore, two groups (A and B) are found (see Table 8.a). As both groups have only one operation, the two resultant actions look very similar (see Table 8.b).

Table 8.a: Node example: Groups before aggregating operations to actions

Group	Operations	Hierarchy level
A	2007-11-28T21:07:33Z: wapi001: Op Node Create Node	30
B	2008-08-31T21:02:14Z: pronti: Op Node Update Coordinate	2

Table 8.b: Node example: Resulting actions

Action	Operations	Hierarchy level
1	**Ac Create Point**	
	2007-11-28T21:07:33Z: wapi001: Op Node Create Node	30
2	**Ac Update Point Coordinate**	
	2008-08-31T21:02:14Z: pronti: Op Node Update Coordinate	2

The way example is more extensive as there are more operations involved. After ordering the operation by user and timestamp, two groups (A and B) have been generated (see Table 9.a). In group A, the first operation is Op Way Create; therefore the first action is Ac Create Line/Polygon (see Table 9.b). Other operations in this action's definition are *OP FEATURE ADD PRIMARY TAG*, *OP WAY ADD NODE* and *OP NODE UPDATE COORDINATE*; therefore these operations are added to the action. One operation from group A is left; this builds another action *AC ADD ATTRIBUTE*. In group B, the operation with the highest level is *OP FEATURE ADD TAG*, therefore the next action is again *AC ADD ATTRIBUTE*. The second operation of the same type is also added to this action. Of the remaining four operations, *OP WAY ADD NODE* has the highest level. This operation builds the new action *AC UPDATE LINE/POLYGON ADD NODE*. The other three operations can be added to this action according to the action definition. Example 2 has transformed twelve operations into four actions.

Table 9.a: Way example: Groups before aggregating operations to actions

Group	Operation	Hierarchy level
A	2007-04-17T22:49:31Z: cdaller: Op Way Create (4475775)	29
A	2007-04-17T22:49:31Z: cdaller: Op Feature Add Primary Tag (highway)	15
A	2007-04-17T22:49:31Z: cdaller: Op Feature Add Tag (name)	12
A	2007-04-17T22:49:31Z: cdaller: Op Way Add Node (27435304)	5
A	2007-04-17T22:49:31Z: cdaller: Op Way Add Node (27435283)	5
A	2007-04-17T22:49:31Z: cdaller: Op Way Add Node (27435284)	5
B	2011-01-30T15:43:48Z: pronti: Op Feature Add Tag (addr:city)	12
B	2011-01-30T15:43:48Z: pronti: Op Feature Add Tag (addr:country)	12
B	2011-01-30T15:43:48Z: pronti: Op Way Add Node (445066391)	5
B	2011-01-30T15:43:48Z pronti: Op Way Add Node (445066399)	5
B	2011-01-30T15:43:48Z pronti: Op Way Remove Node (27435284)	4
B	2011-01-30T15:43:48Z pronti: Op Way Remove Node (27435283)	4

Table 9.b: Way example: Resulting actions

Action	Operation	Hierarchy level
1	**Ac Create Line/Polygon**	
	2007-04-17T22:49:31Z: cdaller: Op Way Create (4475775)	29
	2007-04-17T22:49:31Z: cdaller: Op Feature Add Primary Tag (highway)	15
	2007-04-17T22:49:31Z: cdaller: Op Way Add Node (27435304)	5
	2007-04-17T22:49:31Z: cdaller: Op Way Add Node (27435283)	5
	2007-04-17T22:49:31Z: cdaller: Op Way Add Node (27435284)	5
2	**Ac Add attribute**	
	2007-04-17T22:49:31Z: cdaller: Op Feature Add Tag (name)	12
3	**Ac Add attribute**	
	2011-01-30T15:43:48Z pronti: Op Feature Add Tag (addr:city)	12
	2011-01-30T15:43:48Z pronti: Op Feature Add Tag (addr:country)	12
4	**Ac Update Line/Polygon Add node**	
	2011-01-30T15:43:48Z pronti: Op Way Add Node (445066391)	5
	2011-01-30T15:43:48Z pronti: Op Way Add Node (445066399)	5
	2011-01-30T15:43:48Z pronti: Op Way Remove Node (27435284)	4
	2011-01-30T15:43:48Z pronti: Op Way Remove Node (27435283)	4

5.2.4 Workflow for aggregating operation to actions

An activity diagram which shows the aggregating process is shown in Figure 12. The main loop is called *INITIALIZATION*. This loop iterates through all features until no more feature are found. Here the operations of the features are group based on the timestamp and the user. One input parameter defines the maximum time between two operations. If this time is exceeded, a new group is created. Only operations performed by one user are allowed in a

group. If there are operations from two or more user within a short time, more groups are created.

The *PREPARATION-AGGREGATION-EVALUATION* section iterates through the groups. The *EVALUATION* part prepares the groups, the *AGGREGATION* part aggregates the operations to actions and the *PREPARATION* part post-processes the actions.

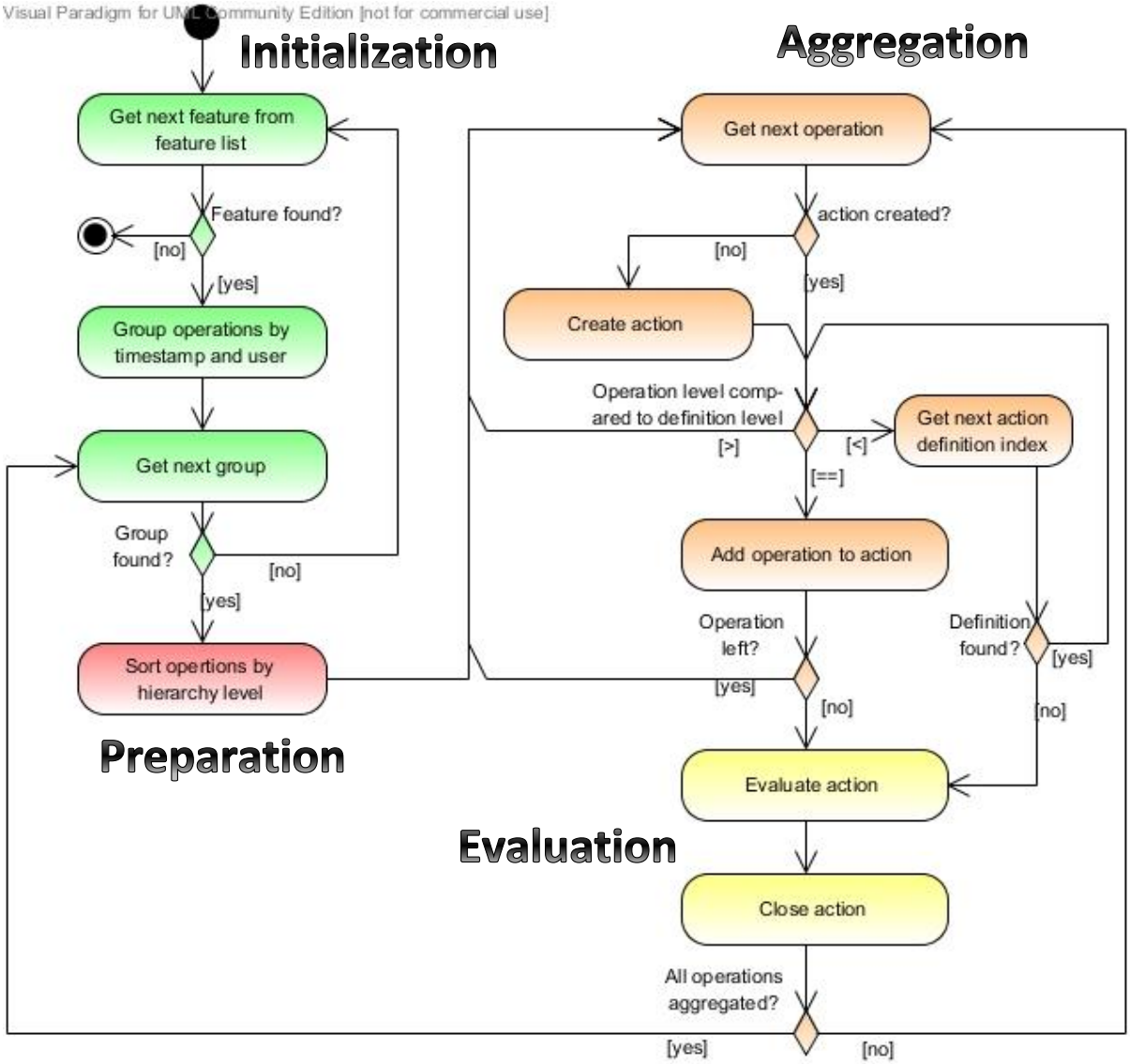


Figure 12: Workflow of aggregating Operations to Actions

PREPARATION includes ordering the operations within a group by the operation hierarchy level (see

Table 6: Operation hierachy levels). At the beginning, no action is selected; therefore the action which is associated with the first operation is chosen. This action has an action definition which defines the operations which belong to this action. The first operation in this definition matches the selected operation, so it is added to the action. If there are operations left in the group, the next operation is called; otherwise the action will be evaluated and

closed (yellow part in the diagram). If an operation is called and an action already exists, the operation is compared to the current action definition index. If the operation matches the definition, the operation is added to the action; otherwise the next definition index and the next operation are called. If no more definition indexes are available, the action will be evaluated and closed.

If all operations within the group are aggregated, the next group is called. If all groups for a feature are processed, the next feature is called. If all features are processed the aggregation is finished.

Eventually, every feature has a set of actions consisting of one or more operations. These will be assigned to activities in the next section 5.3.

Table 10 gives a detailed description of the aggregation example in Table 9.a (Group A). The first operation *Op Way Create* has the highest definition index and therefore selects the action *Ac Create Line/Polygon.* The next operation in the group (*Op Node Add Primary Tag*) has a lower hierarchy level than the first definition entry; therefore the next index is activated. Now, the operation equals the definition at index 2 and the operation is added to the action. The next operation in the group is *Op Node Add Tag*. This operation has a lower level than the definition; therefore the next index (3) *Op Way Add Node* is activated. Now the current operation has a higher hierarchy level than the definition; therefore this operation will be put on hold. The same applies for the second *Op Way Add Tag* operation. The next operation in the group is *Op Way Add Node*. This one equals the current definition index level and is added to the action. The same applies for the remaining three operations. Eventually, all operations have been processed and the action is closed. The last definition entry (*Op Node Update Coordinate*) is not used in this example. Two operations (*Op Node Add Tag*) have been put on hold. Another action will be created in order to aggregate these.

Table 10: Aggregation example

Operations in group	**Action definition index**
Op Way Create (29)	Select action *Ac Create Line/Polygon*
	1. Op Way Create (29)
Op Node Add Primary Tag (15)	2. Op Node Add Primary Tag (15)
Op Node Add Tag (12)	3. Op Way Add Node (5)
Op Node Add Tag (12)	3. Op Way Add Node (5)
Op Way Add Node (5)	3. Op Way Add Node (5)
Op Way Add Node (5)	3. Op Way Add Node (5)
Op Way Add Node (5)	3. Op Way Add Node (5)
Op Way Add Node (5)	3. Op Way Add Node (5)
	4. Op Node Update Coordinate (2)

5.3 Assigning Actions to Activities

Every action is assigned to one of the following activities. Activities are the improvement of one of the quality parameters defined by the standard ISO 19157. Those activities act as quality criteria in this algorithm.

Improvement of Completeness

The creation of new features affects the completeness in a positive way. As only features with a feature type (primary tag) are considered for the result, creating way nodes does not influence completeness.

Improvement of Logical Consistency

This includes actions which are invisible in the map but important for certain tasks like routing. If certain tags are changed, the logical consistency is influenced. An important issue is topology, which defines the kind of relation between two or more features.

Improvement of Positional Accuracy

The *AC NODE UPDATE COORDINATE* redefines its location, therefore the positional accuracy improves. Also if the node list of a way changes, this quality parameter is influenced in a positive (more nodes) way. The way is redirected to a new list of nodes here. The same situation applies for the relation element.

Improvement of Thematic Accuracy

Creating, modifying, and deleting tags influence the thematic accuracy. Only non-primary tags are considered here. On the contrary, primary tags describe the type the feature (e.g. highway, building, natural …). Therefore, they do not influence thematic accuracy (except *AC FEATURE UPDATE FEATURE TYPE*).

Temporal Accuracy

The fifth quality parameter Temporal Accuracy is represented as an input parameter of the algorithm. The time span defines which operations are considered for the evaluation.

5.3.1 Activity table

Section 5.2 introduced actions. The current section links these actions to activities. Four activities are considered: (1) Improvement of Positional Accuracy, (2) Improvement of Attribute Accuracy, (3) Improvement of Completeness, and (4) Improvement of Logical Consistency. The fifth quality parameter Temporal Accuracy is covered by the input parameter time span. Evaluation results only consider operations, actions and activities within the given time span.

Table 11 gives an overview of the influence of actions on the quality parameters. It is assumed that every action has a positive impact on quality. The actions are defined that they influence exact one quality parameter. This assignment is marked with a "+"-symbol.

Table 11: Assignment of editing actions to activities

Action	Activity			
	Improvement of Positional Accuracy (P)	Improvement of Thematic Accuracy (T)	Improvement of Complete-ness (C)	Improvement of Logical Con-sistency (L)
Ac Create Point			+	
Ac Update Point Coordinate	+			
Ac Delete Point			+	
Ac Create Line/Polygon			+	
Ac Update Line/Polygon Coordinate	+			
Ac Line Add Node	+			
Ac Line Remove Node	+			
Ac Split Line/Polygon				+
Ac Merge Line/Polygon				+
Ac Reverse Line/Polygon				+
Ac Delete Line/Polygon			+	
Ac Add Feature-Relation			+	
Ac Add Merge-Relation				+
Ac Add Feature-Relation Element				+
Ac Add Merge-Relation Element				+
Ac Add Feature Type			+	
Ac Update Feature Type		+		
Ac Remove Feature Type			+	
Ac Add Attribute		+		
Ac Update Attribute		+		
Ac Remove Attribute		+		

5.3.2 Quality values

By default, the quality value of an activity is set to 1. In three situations different values are set: (1) long/short ways, (2) few/many attributes, (3) small/large merge-relations. The size of the features or the attribute list should have an impact on the evaluation result. For example, long ways should have a higher quality value than short ways. Every action which is linked to a way, a relation or to attributes will get a customized quality value. This value will be calculated with a value function. Malczewski (1999) introduced the value function as a method to transform values to a defined range. For the purpose of this work, the following function was used:

$$x = 1 - \left(\frac{maxValue - value}{value}\right)^{p} * (1 - minResult)$$

Depending on the action subject, three parameters are set based on the values in Table 12. The value is the number of operations involved (e.g. number of attributes, number of way nodes, number of relation members). The maximum value sets bounds in order to eliminate upper outliners, for example if there are more than 100 way operations, 100 is used as the value. The p value sets the curvature of the function. Minimum Result is the smallest possible result value.

Table 12: Value function settings

Subject	Maximum Value	p	Minimum Result
Ways	100	4	0.2
Merge Relations	200	4	0.2
Attributes	20	3	0.2

Figure 13 shows a diagram which includes the results for the value function for attribute actions. Actions with one attribute have a quality value of 0.31, actions with four attributes have a quality value of 0.59, and actions with more than seven attribute operations have a quality value of 0.8 or higher.

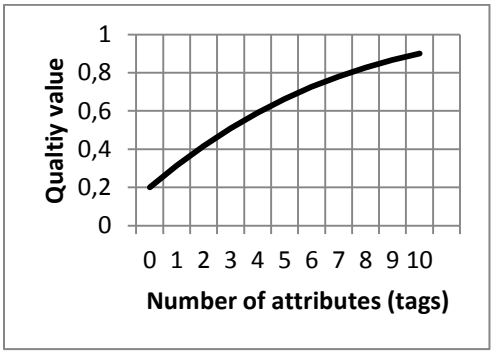


Figure 13: Value Function for evaluating attributes

5.4 Evaluation

The evaluation phase starts after preprocessing the data. Preprocessed data means that (1) OSM data has been downloaded for a selected bounding box; (2) operations have been extracted and (3) aggregated to actions, which (4) have been assigned to activities (improvement of ISO 19157 quality parameters completeness, logical consistency, positional-, and thematic accuracy). The easiest way to find a result would be to sum up positive actions within activities. The sum would be the rating for the selected OSM area.

Aspects which distort the evaluation result are differences between short (2 - 10 nodes) and long (100 – 500 nodes) ways, small (3 members) and large (200 – 500 members) relations

and few (1 tag) and many (10 tags) attributes. As a consequence, a quality value is calculated which balances all features. Now all resultant values can be used for comparisons between the following aspects:

- Number of operations/actions
- Number of users
- Distribution of quality parameter
- Distribution of features types
- Different areas at the same time span
- Different time spans at the same area

The (1) size of the evaluation area, (2) the node density, and (3) the evaluation time span must be considered in order to produce comparable outcomes. This ensures that regions with different landuse may be compared.

Research in related work (see section 4.4) gives an idea about OSM data quality for a project-specific region and time frame. A rating for those regions can be used as an entry for further evaluations. Based on that, further regions can be rated and compared with the base ratings. The analysis of the related work indicates that urban areas have a better OSM data quality than rural areas. This thesis will compare city centers located in four different environments (urban and rural).

Data quality can be defined as “better” or “worse” for a specific purpose. After analyzing some training data good or bad ratings can be determined and used for later analysis. Evaluations can be done by comparing data sets with a different spatial and/or temporal dimension.

Apart from calculating a rating, other methods can be used to compare selected regions. The ratio between create and modify actions indicates if more features are created or modified. By using this approach the user can recognize the development status for the region. Another method is to analyze the time of the last editing actions for a feature.

5.5 Summary

The section 5 (Data Processing) explained the functionality of the algorithm in detail. It is primarily organized in three layers: operations – actions – activities. Operations are the atomic editing actions which are extracted from the OSM data. They are aggregated to action in order to merge operations which belong together. For example, creating a way and adding the way points is one action which creates a line (or polygon). Next, every action is assigned to an activity based on the type of the action. Activities are the improvement of four of four quality parameters (Positional Accuracy, Thematic Accuracy, Completeness, and Logical Consistence). The fifth parameter is covered by the evaluation period, which defines

the time span for the extracted operations. As soon as the actions are linked to activities, the data is ready for the interpretation and evaluation.

6 Evaluation Protocol

This section explains the evaluation protocol which is the output of the data-processing. Figure 14 shows the content of this protocol which includes some general result statistics and the quality matrix. The statistics give some indications about the evaluation sample. The detailed evaluation result can be extracted from the quality matrix. The quality parameter distribution diagram can be derived from the matrix.

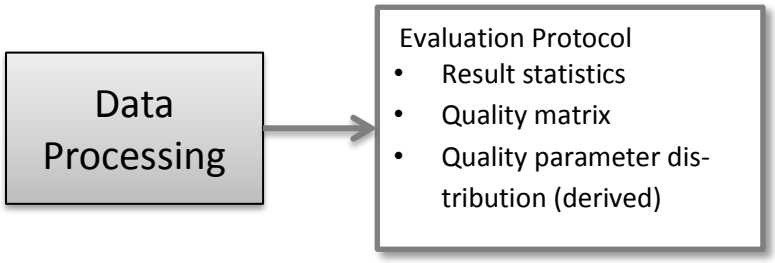


Figure 14: Output of the algorithm

6.1 Statistics

Figure 15 shows the statistics which are produced by the algorithm.

```
1  Bounding box
2   - http://www.openstreetmap.org/?box=yes&maxlat=47.8011192&maxlon=
      13.0494622&minlat=47.7961192&minlon=13.0444622
3  Time Span
4   - Start: Sun Jun 12 15:45:57 CEST 2011
5   - End: Tue Jun 12 15:45:57 CEST 2012
6  OsmElementType (with Feature Type)
7   - NODE: 1704 (102)
8   - WAY: 387 (387)
9   - RELATION: 6 (6)
10  - Sum: 2097 (495)
11 Operations: 3563
12 Actions: 51
13  - Avg Operations per Feature: 7.198
14  - Avg Actions per Feature: 0.103
15 Area: 212891.14300861815 m2
16  - Nodes per 1000m2: 8.004
17 User: 15
18  - Avg Operations per User per Year: 237.533
19  - Avg Actions per User per Year: 3.4
```

Figure 15: Resultant statistics

It includes the following facts:

- [row 1 – 2] At the beginning the bounding box is written within with a link which shows the map.
- [row 3 – 5] Next, the start- and end date of the evaluation is mentioned.

- [row 6 – 10] The number of features is separated into the OSM elements (nodes, ways and relations). Additionally the number of features which include a feature type (primary tag) is listed. Nodes often act as a way point. In this situation they do not include a primary tag. That is the reason why the overall number of nodes is much higher than the number of nodes which include a feature type. On the other hand, ways and relations must have a feature type as there is no usage of ways and relations without it.
- [row 11 – 14] Next, the number of operations and actions is revealed. Also the average number of operations respectively actions per feature is mentioned. In average, ten operations form one action.
- [row 15 – 16] Fourth is the area of the bounding box in square meters (m²). Also the nodes per 1000 m² are calculated in order to get an idea about the node density within project area.
- [row 17 – 19] Finally the number of users which have performed the operations is listed. Here, also the average number of operations respectively actions per user is mentioned. Those figures are normalized on a one-year-basis.

6.2 Quality Matrix

Beside the statistics, the evaluation protocol also includes a quality matrix. See Table 13 for a simplified matrix. An entire quality matrix is available in section 13.3. All actions are listed in the rows along with the corresponding activities. The columns contain all feature types. The table body sums up all quality values (see section 5.3.2) for all actions and feature types. In the following example, the maximum value is 108.9. This means that the sum of all quality values of building features and the action *AC LINE CREATE* is 108.9.

Table 13: Quality matrix example

Action	Activity	Highway	Building	Amenity	Shop	Natural	Sum
Ac Point Create	C	1.0	0.0	4.0	2.0	5.0	**12.0**
Ac Line Create	C	4.5	108.9	0.7	0.0	0.0	**114.1**
Ac Add Attribute	T	7.2	14.2	6.5	1.6	0.0	**29.5**
Ac Line Split	L	3.3	1.3	0.0	0.0	0.0	**4.6**
Ac Line Reverse	L	4.0	0.0	0.0	0.0	0.0	**4.0**
Ac Line Add Node	P	10.6	1.2	0.0	0.0	0.0	**11.8**
Sum		**30.6**	**125.6**	**11.2**	**3.6**	**5.0**	**176.0**

C … Improvement of Completeness; T … Improvement of Thematic Accuracy; L … Improvement of Logical Consistency; P … Improvement of Positional Accuracy

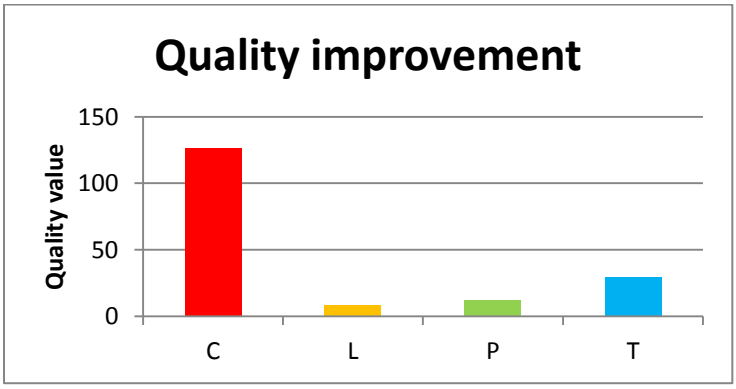
Some quality values are integers (e.g. *AC POINT CREATE*, *AC LINE REVERSE*) while others are float numbers (e.g. *AC LINE CREATE*, *AC ADD ATTRIBUTE*). The difference between these two groups is the value function which is only applied on some actions (see section 5.3.2 Quality values) whose value requires some adjustment because of different number of (1) way nodes, (2) tags or (3) relation members. Other action values are always 1. For example, *AC POINT CREATE*

actions do not use the value function; therefore it can be said that one point feature with the feature type highway and five point features with the feature type natural have been created.

6.3 Activity Distribution

Every action in the quality matrix is linked to an activity. It is assumed that mapping activities have a positive influence on the four quality parameters (1) Positional Accuracy, (2) Thematic Accuracy, (3) Completeness, and (4) Logical Consistence. The fifth parameter Temporal Accuracy is covered by the input parameter time span. In Table 13, all values for the actions are summed up in the last column. For example, the value for all *AC POINT CREATE* actions is 12.0. This action is linked to Completeness. If the sum of all actions of this activity is again summed up, a quality value for the entire activity is the result. In the mentioned example there are two Completeness actions, which result in an overall sum of 126.1 (12.0 + 114.1). The same procedure is applied to the other three activities.

Eventually, a quality parameter distribution diagram which compares the merged quality values can be derived from the matrix. Figure 16 shows the result for the example in Table 13. The diagram reveals an extensive peak at *IMPROVEMENT OF COMPLETENESS*. The remaining three quality parameters are much lower, with *IMPROVEMENT OF THEMATIC ACCURACY* clearly second. This result indicates that during the associated time span, many new features have been created. A possible reason might be the creation of buildings which produce such peaks of the *IMPROVEMENT OF COMPLETENESS* activity. In order to get proof for this assumption, the quality matrix has to be analyzed.



C … Improvement of Completeness; L … Improvement of Logical Consistency; P … Improvement of Positional Accuracy; T … Improvement of Thematic Accuracy

Figure 16: Merged activity values

6.4 Summary

The Evaluation Protocol section explains the instruments which are used to evaluate the OSM data. The algorithm produces a set of statistical data and a quality matrix. The statistical data includes information about the features (number of nodes, ways, and relations), the

operations and actions, the project area (size), and the mapper along with the input data (bounding box and time span). The quality matrix includes all actions (with their activities) in rows and features types in columns and the corresponding summed up quality values. The matrix allows an easy filtering of actions, activities, or features types. The activities can be used to create a diagram which shows the quality parameter distribution. Methods how to interpret these results are introduced in the next section 7.

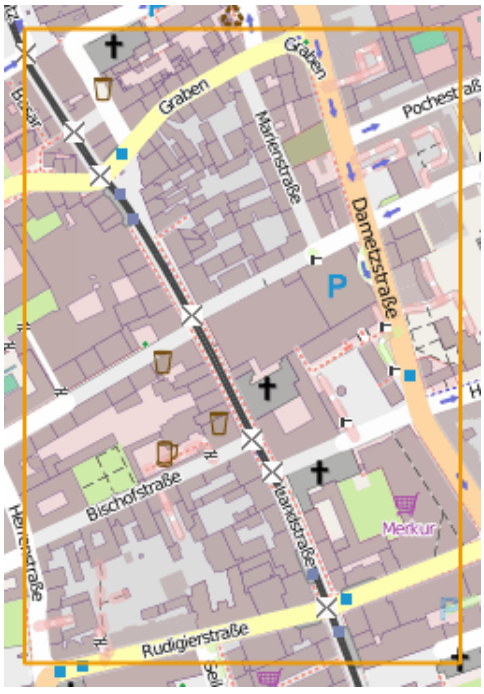
7 Validation and Conclusion

This section discusses how the introduced algorithm can be used to determine the quality of OSM data. It is divided into three parts: (1) the first one introduces the test areas, (2) the second part explains methods to visualize and communicate data quality; (3) and third, internal and external restrictions are listed.

7.1 Evaluation test areas

Evaluation test regions are located in Austria and include (Figure 17.a) Linz, (b) Salzburg, (c) Villach and (d) Spittal. The algorithm will be applied on these cities. Data sets for the same location will be evaluated for a series of time spans in order to recognize data quality changes over time. All areas cover an urban environment in order to allow the comparison between the four areas. The four cities have been selected because of obvious quality differences which are visible on the derived maps.

On first sight, Linz and Salzburg have a good data quality as many different features populate the map. Villach has a rising quality score as many features have been added during the working period of this thesis. Spittal features many white areas and therefore a low data quality can be assumed. The bounding box is defined based on geographic coordinates. Cities in the north have a smaller area because of meridian convergence.



(a) Linz, Landstraße

(b) Salzburg, Dom

(c) Villach, Hauptplatz

(d) Spittal, Neuer Platz

Figure 17: Evaluation test areas

Table 14 gives additional characteristics (area and inhabitants) of the four evaluation areas. In inhabitants values represent the whole town.

Table 14: Characteristics of the evaluation areas

	Linz	Salzburg	Villach	Spittal
Area	195,488 m²	208,245 m²	212,891 m²	212,160 m²
Inhabitants (x1000)	191	149	60	16

7.2 Quality Indicator Validation

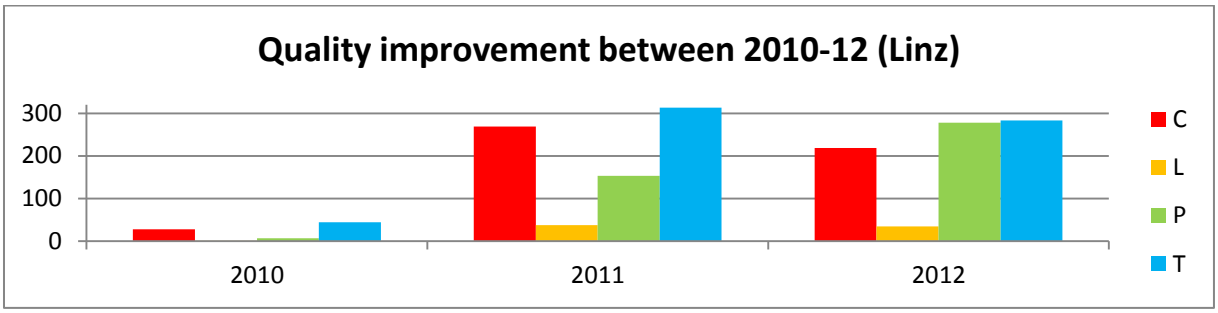
This section discusses if and how the output of the introduced algorithm can help to evaluate OSM data. Quality indicators are attributes which allow the evaluation of the data. The following quality indicators are analyzed:

- Activity distribution diagram
- Highways versus buildings
- Feature type variety
- OSM elements
- Mappers

Activity distribution diagram

In general, OSM mappers first add new features to the map. This results in actions which are linked to the activity *Improvement of Completeness*. Later, updates are performed on those features. These influence the parameters *Improvement of Positional Accuracy*, *Improvement of*

Thematic Accuracy, and *Improvement of Logical Consistency*. Figure 18 gives examples of an activity distribution diagram for the four areas over the period of three years. The bars in the diagram indicate the sum of all actions' quality values for the four activities.



C … Improvement of Completeness; L … Improvement of Logical Consistency; P … Improvement of Positional Accuracy; T … Improvement of Thematic Accuracy

Figure 18.a: Quality parameter distribution for the city of Linz (years 2010 - 2012)

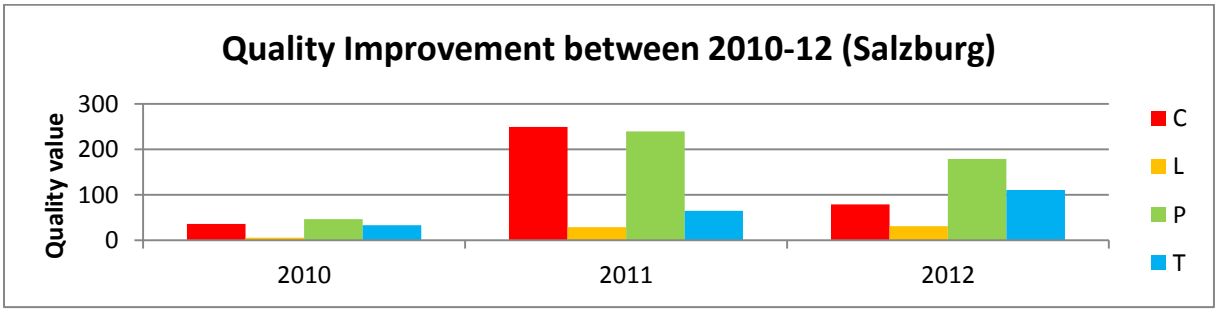


Figure 18.b: Quality parameter distribution for the city of Salzburg (years 2010 - 2012)

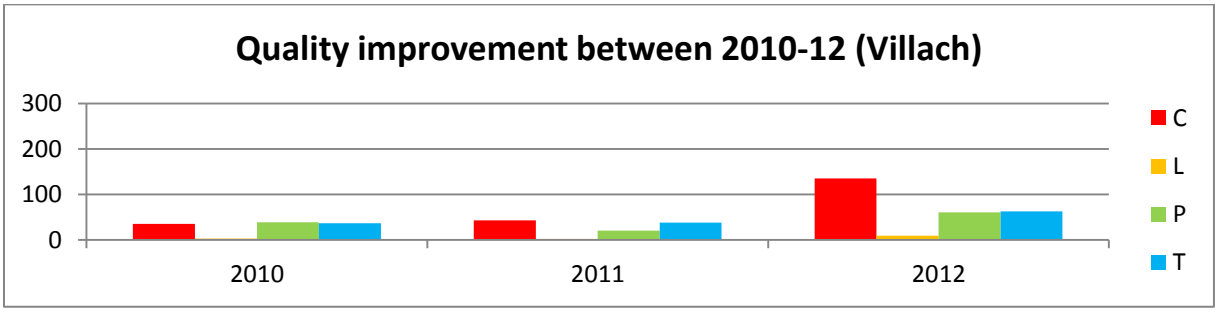


Figure 18.c: Quality parameter distribution for the city of Villach (years 2010 - 2012)

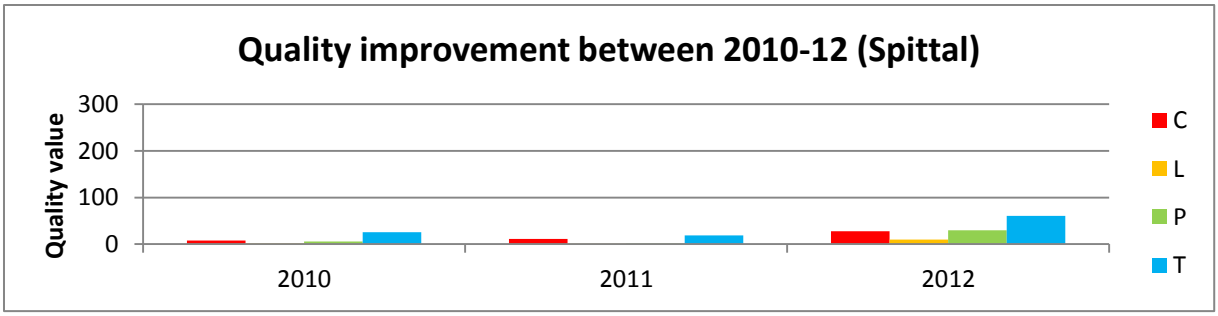


Figure 18.d: Quality parameter distribution for the city of Spittal (years 2010 - 2012)

This paragraph discusses the quality improvement example of Salzburg (see Figure 18.b). While all four values are small in 2010, *Improvement of Completeness* and *Improvement of Positional Accuracy* are both high in 2011. In 2012, actions which are linked to *Improvement of Positional Accuracy* and *Improvement of Thematic Accuracy* are dominant. The reason for the 2011 *Improvement of Completeness* value is the creation of many buildings (compare with quality matrix in Table 13). The analysis of several quality matrixes for different cities indicates that all buildings within the evaluation area are created within a short period of time. In contrast, features with other feature types including highways (streets) are created over a longer time span.

The example of the city of Spittal shows, that the poor bad data quality (as assumed) is also recognizable in the diagram as the activity bars are very short (see Figure 17.d).

Feature type: Highway versus building

Figure 19 shows an extract of the quality matrix for the city of Villach for three consecutive years. Only the feature types highway and building are considered in the example. As OSM is a street map, mappers are encouraged to create the streets first, and buildings at a later stage (Ramm and Topf, 2010).

		2010		2011		2012	
QualityParameter	Action	highway	building	highway	building	highway	building
LogicalConsistency	ActionLineSplit	0,0	0,0	3,0	1,0	4,0	2,0
Completeness	ActionPointCreate	0,0	0,0	1,0	0,0	0,0	0,0
Completeness	ActionLineCreate	1,0	0,0	6,0	28,0	5,0	222,5
Completeness	ActionFeatureAddType	0,0	0,5	1,5	0,0	0,0	1,0
ThematicAccuracy	ActionFeatureAddAttribute	0,0	0,5	10,0	29,0	12,0	39,5
ThematicAccuracy	ActionFeatureUpdateType	0,0	0,0	0,0	0,0	1,0	0,0
ThematicAccuracy	ActionFeatureUpdateAttribute	57,0	0,0	0,0	0,0	0,0	12,5
ThematicAccuracy	ActionFeatureRemoveAttribute	19,0	0,0	65,5	0,0	2,0	0,0
LogicalConsistency	ActionLineReverse	0,0	0,0	0,0	0,0	4,0	0,0
PositionAccuracy	ActionLineAddNode	1,0	0,0	28,0	1,0	22,0	5,0
PositionAccuracy	ActionLineUpdateCoordinates	21,0	0,0	49,5	1,0	16,0	1,0
PositionAccuracy	ActionPointUpdateCoordinate	0,0	0,0	12,0	0,0	2,0	0,0
		99,0	1,0	176,5	60,0	68,0	283,5

Figure 19: Quality matrix for the city of villach

In the example above, the streets are created prior to 2010 and other modifications are done in the years 2010 – 2012. The actions *Ac Feature Update Attribute*, *Ac Feature Remove Attribute*, *Ac Line Update Coordinates* dominate. In contrast, the first pack of buildings has been created in 2011, the majority in 2012. The relationship between streets and buildings is another indicator of the quality progress for the region. The example also confirms the assumption that streets are created prior to buildings.

Number of different Feature Types

The next quality indicator is the number of feature types which are involved in the operations. Table 15 gives an overview of the number of feature types for four cities and three consecutive years. The list of primary tags in section 4.2.1 (OSM data model) lists 26 different feature types. Two cities (Linz and Salzburg) have a high number of feature types involved throughout the years. The city of Villach has a low number in 2010, but the value improves. Spittal is below the average in all three years which indicates a poor data quality.

Table 15: Number of different feature types for four different regions

Area	2010	2011	2012
Linz	14	16	15
Salzburg	14	13	16
Villach	6	12	13
Spittal	3	7	7

OSM elements

Also the analysis of the OSM elements itself gives an idea about data quality. The normalized number of nodes reveals the node density for the evaluation area. Another quality factor is the number of relations. The relation is the youngest and most complex mapping element and it is not necessary to create the basic map elements like streets, amenity or buildings. Table 16 gives an overview of the node density and the number of relations for the four cities. Again, Linz and Salzburg have the best values in both categories, while Villach and especially Spittal have lower values.

Table 16: OSM element node density and relation count for four different regions

	Nodes	Nodes/1.000 m²	Ways	Nodes/Ways	Relations
Linz	2487	12,7	579	0,02	83
Salzburg	2904	13,9	674	0,02	53
Villach	1704	8,0	387	0,02	7
Spittal	247	1,1	51	0,02	10

The ratio between nodes and ways gives an idea about the number of way points. In cities with a good data quality, an average way has fewer nodes. A reason is the higher level of detail. For example, highways are divided more often in order to map additional attributes like speed limit.

Table 17 introduces a quality evaluation method, which tries to identify rural and urban areas based on the node density and the actions. In order to produce accurate quality evaluation results, information about the land-use of the area is important. OSM features a landuse tag, but it is not widely used (Rehrl et al, 2012). Therefore it is necessary to extract that information from the existing data. Rural areas have (1) a lower node density, (2) the variety of

feature types is lower and (3) as a result a lower number of operations and actions, but it does not mean a bad data quality for the region.

Table 17: Node density and new feature relationship

	Low node density	High node density
More new features	Bad quality in rural or urban area	Medium quality in urban area
Less new features	Good quality in rural area	Good quality in urban area

Mappers

Another quality indicator is the number of mappers, which perform operations within the evaluation area and time period. A higher number suggests a better data quality (Hochmair, 2010). This number can be compared to the inhabitants in the region.

Also the ratio of mappers per inhabitants can produce valuable results. Table 18 shows that Villach and Spittal have more mappers per inhabitants than Salzburg or Linz. Interestingly, data quality in Linz and Salzburg is better than in the other cities.

Table 18: Ratio between mappers and inhabitants

City	Mappers 2011	Inhabitants (x1000)	Ratio
Linz	34	191	0,178
Salzburg	37	149	0,248
Villach	23	60	0,383
Spittal	5	16	0,313

Inhabitant values are provided by Statistik Austria (2012)

7.3 Limitations

The current version of the introduced algorithm has some limitations. Some of them can be eliminated by remodeling the methodology, while others are bound to external sources.

Relative evaluation

By using the algorithm, only relative results can be achieved. Two or more OSM datasets can be compared and declared as better/worse than others.

Small evaluation areas

The OSM API has restrictions in order to avoid too many calls within a short time span and to protect the servers (OpenStreetMap Wiki, 2012e). The algorithm uses the map, history and changeset API functions to download the necessary data. This results in a too large amount of API calls. As a consequence, only small areas can be evaluated at once. During testing, areas with a size of 200,000 m² (400 x 500 meters) have been evaluated.

Real-life development

The algorithm cannot differ between quality improvement (e.g. mapping of an existing building) and quality maintenance (e.g. mapping of a new built building).

OSM tags

Due to the use of tags, the OSM data model is most generic. Every mapper can set arbitrary tags. The OSM Wiki[6] sets tag recommendations, but it is not obligatory to use them. It is recommended to use these tags to allow everyone (e.g. other mappers, map renderer) to understand and use the data. The same applies for primary tags which define the feature type of the feature (e.g. highway, building, landuse, and waterway). It is possible to create features without primary tags.

Very short ways

Short ways (for examples a bridge with two nodes) complicate the process of finding related features. Extracting operations like *OP WAY MERGE* or *OP WAY SPLIT* is difficult especially if one or both of the new nodes is created in the same process. If there is none or only one node which acts as a link during the split/merge, it cannot be identified of part of the (original) way.

Geometry types

The OSM element way is used for lines and polygons. There is no strict separation between them. Both polygons and closed lines are ways with the requirement that the first node equals the last node. The tags give information about the meaning and usage of the feature. Again the user or application (e.g. map renderer) which interprets the data, decides how to handle the feature. This will result is a less accurate evaluation results. But sometimes it is necessary to separate them; therefore it might be necessary to specify which tags form which geometry type.

Changes within the OSM data model

In order to expand the functionality of OSM the data model is adjusted sometimes. The last change to the API version 0.6 has been in 2009. The major adjustment was the introduction of changesets. Since then, all OSM modifications of one user within an editing session have been saved in a changeset. Changesets before that date have been generated subsequently (OSM Wiki, 2012b). Every change in the data model requires an adjustment of this work's algorithm. In the future, the node element may be separated into a POI node and a way node. Such changed would require extensive model adjustments.

[6] http://wiki.openstreetmap.org/wiki/Map_Features

Replace Feature

As section 13.1.5 (Other operations) explained, the operation *OP FEATURE REPLACE* is difficult to extract. The reason is that the replaced and the new feature have different IDs. One solution is to search within the changeset which includes the creation of the new feature. If a deleted feature with the same or similar tags and/or coordinates is found, the operation is identified. This works as long as the changesets are not too large. A changeset can contain up to 50,000 created, modified and/or deleted features. It is not possible to download these large changesets within a reasonable time. *OP FEATURE REPLACE* operations cannot be identified if this problem occurs.

A possible work-around is the usage of planet files. Such files contain the entire OSM map for a defined country and a specific time. Every time a feature is created, it could be compared with the planet files of previous months or years in order to find a predecessor feature.

7.4 Conclusions

Section 7.1 (Evaluation test area) defined four test areas, which covered four city centers in four different cities. On first sight, the data quality was good in two cities (Linz and Salzburg), medium in Villach and very poor in Spittal. The evaluation results of the algorithm confirmed the assumption. Thus, the question whether it is possible to analyze editing actions of VGI in order to evaluate data quality has been answered. The proposed approach reveals positive results which can be further analyzed in many aspects involving the feature types or the mappers. Although the algorithm proved successful at first sight, a number of open questions remain and have to be answered in the future.

Related works used methodologies which allowed much larger evaluation areas. As this work was relaying on the OSM API to acquire the data, only small areas (e.g. city centers) could be evaluated. As a consequence, the evaluation outcomes of this work cannot be compared to the other works' results. By the time when large areas can be processed, such comparisons have to be done.

8 Summary

Within the last five years, many VGI quality evaluation methods were developed. Most of them compared the data with (commercial) third-party data sets. Recently, members of the community, especially of the OSM project, introduced intrinsic evaluation methods. Some of them analyzed the mappers, some worked on the current version of the data, and the other investigated the editing history. This work primarily tries to determine the data quality by extracting and analyzing the editing actions of all features within a defined area and time period. Atomic edits (operations) are identified by comparing two consecutive feature versions. Operations are all single data modifications like creating a feature or adding an attribute or a way point. In the next step, the operations are logically aggregated to actions. For example, creating a way and adding its way points belong together. Later, if some nodes are moved to another coordinates, all update operations are merged. In average, ten operations form one action. All actions are later linked to an activity. Activities are the improvement of the quality parameters Positional Accuracy, Thematic Accuracy, Completeness and Logical Consistence. The fifth parameter Temporal Accuracy is covered by the time span (input parameter). Eventually, the results can be evaluated. In order to do that, general statistics and a quality matrix can be used. General statistics include information about the features within the project area, the number operations and actions, the area itself and the mappers. The matrix allows filtering actions, activities and features types to help identifying the data quality. The preliminary result interpretations suggest that this algorithm is a good foundation for more detailed evaluations of OSM data. As the OSM API is used to retrieve the data, only small areas like city centers can be evaluated. In the future, the algorithm should be expanded in order to evaluation large project areas (like a whole country).

9 Future Work

This section lists possible future developments:

The evaluation results should be analyzed for more areas and time spans in order to provide more proof for the quality indicators.

The algorithm should consider more details. It involves a clear distinction between line and polygon features. Also the tag values (not only the keys) can be considered in order to receive more accurate evaluation results.

Until now, the algorithm only retrieves the OSM data from the OSM API which results in a bad performance. Other data sources like the planet file or a history dump file can speed up the process and also allow the evaluation of larger areas or whole countries.

Spatial Action clusters can be derived based on the feature type. For example, many *AC CREATE LINE POLYGON* actions within a small area create a cluster; many *AC FEATURE ADD ATTRIBUTE* actions in another area form another one.

Additional information can be analyzed

- Number of nodes in line/polygon
- Time since last action
- Number of updates within time span

10 References

Ather A., 2009, *A Quality Analysis of OpenStreetMap Data*, M.Eng. Dissertation, University College London

Bennett J., 2010, *OpenStreetMap*, 1st ed, Birmingham: Packt Publishing

Coleman, D. J., Georgiadou, Y., Labonte, J., 2009, Volunteered Geographic Information: the nature and motivation of produsers, *International Journal of Spatial Data Infrastructures Research*, 4, 332-358

ESRI, 2012, *ArcGIS Editor for OpenStreetMap* [online] Available from: http://www.esri.com/software/arcgis/extensions/openstreetmap [Accessed 2012-07-24]

Goodchild M., 2007, Citizens as sensors: The world of volunteered geography, *GeoJournal*, 69 (4), 211-221

Google, 2012, *FAQ: What usage limits apply to the Maps API?* [online] Available from: https://developers.google.com/maps/faq#usagelimits [Accessed 2012-07-24]

Haklay, M. (2008). 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.

Haklay, M., Basiouka, S., Antoniou, V., Ather, A., 2010. How many volunteers does it take to map an area well? The validity of Linus's Law to volunteered geographic information. *Cartographic Journal* 47 (4), 315–322.

Hochmair, H.H., 2010, Spatial Association of Geotagged Photos with Scenic Locations. In A. Car, G. Griesebner and J. Strobl (Eds.), *Proceedings of the Geoinformatics Forum Salzburg* (pp. 91-100). Heidelberg: Wichmann.

International Organization for Standardization (ISO), 2011, *ISO 19157:2002* [online] Available from: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=32575 [Accessed 2011-10-17]

Kounadi, O., 2009, *Assessing the quality of OpenStreetMap data*, MSc Thesis, University College of London

Kuutti, K., 1996. Activity theory as a potential framework for human computer interaction research. *In Context and consciousness: Activity theory and human-computer interaction*, ed. B.A. Nardi, 17-44. Cambridge, MA: The MIT Press.

Malczewski J, 1999, *GIS and Multi-Criteria Decision Analysis*, New York: John Wiley and Sons

Mondzech J., Sester M., 2011, Quality Analysis of OpenStreetMap Data Based on Application Needs, *Cartographica*, 46 (2), 115-125

Mooney, P., Corcoran, P., Winstanley A., 2010, Towards Quality Metrics for Open-StreetMap, *Proceedings of the 18th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 514-517

Mooney, P., Corcoran, P., 2012a, How social is OpenStreetMap?, *Proceedings of AGILE 2012*

Mooney, P., Corcoran, P., 2012b: Characteristics of Heavily Edited Objects in OpenStreetMap. *Future Internet* 4 (1): 285-305

Neis, P., Zipf, A., 2012, Analyzing the Contributor Activity of a Volunteered Geographic Information Project — The Case of OpenStreetMap, *ISPRS International Journal of Geo-Information*, 1 (2), 146-165

Neis, P., Zielstra, D., Zipf, A., 2012, The Street Network Evolution of Crowdsourced Maps: OpenStreetMap in Germany 2007–2011, *Future Internet*. 2012; 4(1):1-21.

OpenStreetMap Foundation, 2012a, *Welcome, Apple!* [online], Available at http://blog.osmfoundation.org/2012/03/08/welcome-apple/ [Accessed 2012-07-24]

OpenStreetMap Foundation, 2012b, *Automated redactions complete* [online], Available at http://blog.osmfoundation.org/2012/07/17/redactions-progressing/ [Accessed 2012-08-01]

OpenStreetMap Wiki, 2012a, *Bing* [online], Available at http://wiki.openstreetmap.org/wiki/Bing [Accessed 2012-08-06]

OpenStreetMap Wiki, 2012b, *Changeset* [online], Available at http://wiki.openstreetmap.org/wiki/Changeset [Accessed 2012-08-01]

OpenStreetMap Wiki, 2012c, *Map Features* [online], Available at http://wiki.openstreetmap.org/wiki/ Map_Features [Accessed 2012-08-08]

OpenStreetMap Wiki, 2012d, *Types of Relation* [online], Available at http://wiki.openstreetmap.org/wiki/ Types_of_relation [Accessed 2012-08-08]

OpenStreetMap Wiki, 2012e, *Developer FAQ – Using OSM map data* [online], Available at http://wiki.openstreetmap.org/wiki/Developer_FAQ#I.27ve_been_blocked_from_the_API_for_downloading_too_much._Now_what.3F [Accessed 2012-08-08]

Ramm, F. and Topf, J., 2010, *OpenStreetMap*, 3rd ed, Berlin: lehmanns media

Rehrl, K., Gröchenig, S., Hochmair, H., Leitinger, S., Steinmann, R., Wagner, A., 2012, A conceptual model for analyzing contribution patterns in the context of VGI, In L*BS 2012 - 9th Symposium on Location Based Services*. Berlin: Springer.

Stark H., 2011, Quality Assessment Of Volunteered Geographic Information using Open Web Map Services within OpenAddresses, *Geospatial Crossroads @ GI_Forum '11*, 101-110

Statistik Austria, 2012, *Bevölkerung zu Quartalsbeginn ab 2002* [online], Available at http://sdb.statistik.at/statistik.at/ext/superweb/loadDatabase.do?db=debevstand [Accessed 2012-08-07]

Zielstra, D. and Hochmair H.H., 2011, A Comparative Study of Pedestrian Accessibility to Transit Stations Using Free and Proprietary Network Data, Jo*urnal of the Transportation Research Board*, 2217, 145–152

Zielstra, D., and Hochmair, H. H. (in press), Comparing Shortest Paths Lengths of Free and Proprietary Data for Effective Pedestrian Routing in Street Networks, *Transportation Research Record: Journal of the Transportation Research Board*.

11 List of Figures

12 List of Tables

13 Appendices

The following lists and tables are included in the appendix:

- List of all operations with detailed explanation
- List of all action definitions
- Full quality matrix

13.1 Appendix A: Operations

Each operation is explained using an easy-to-understand graphic. The left image shows the initial state, the right one highlights the changes. Figure 20 shows a legend for editing actions.

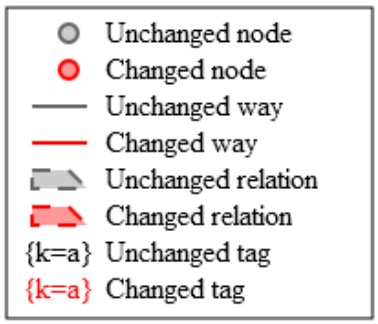


Figure 20: Legend for editing action graphics

13.1.1 Node-related operations

This section gives the details about operations related to nodes.

Create Node

The creation of a node may occur in the following cases:

- A new node without primary tags, which is not added to a way or relation. This is a neutral action as the node has no meaning.
- A new node combined with a primary tag creates a new POI object.
- A new node (with or without a primary tag), which is added to a way (within same changeset like Create Way or Modify Way)

Figure 21 gives two examples for creating a node. Example (a) creates a node which has (no) primary attribute. The second example (b) creates a node and adds it to a way in order to increase the positional accuracy or to build an intersection with another way at the node's location.

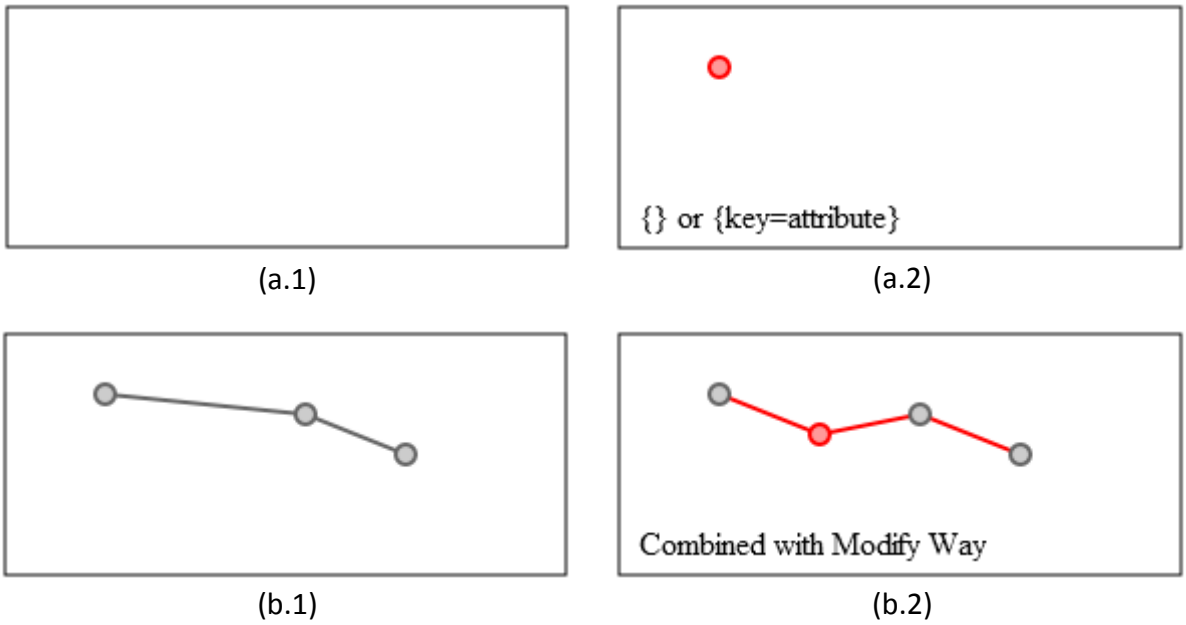


Figure 21: Create node examples

Modify Node: Update coordinate

Every node has a pair of coordinates which defines its location. Changing these coordinates generates an Update Coordinate operation. Figure 22 shows two examples. The (a) first one moves a node which does not belong to a way; the (b) second example changes the coordinates of a way node. There are no additional modifications to the way itself.

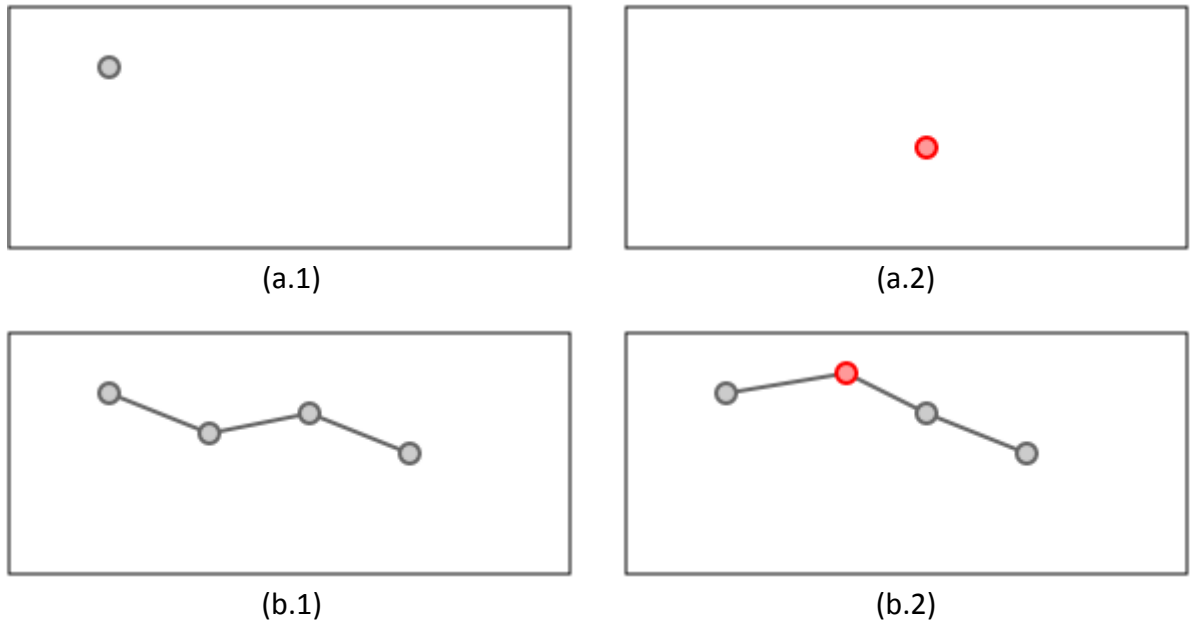


Figure 22: Modify node examples

Delete Node

The deletion of a node may occur in two different cases:

- Deletion of a node element which does not belong to a way.
- Deletion of a node element which belongs to a way. The node must be removed from all ways and relations before the deletion. This operation reduces the number of nodes of a line or polygon.

Both situations are explained in Figure 23. Example (a) deletes a POI node, while the second example shows the reduction of one node in a way.

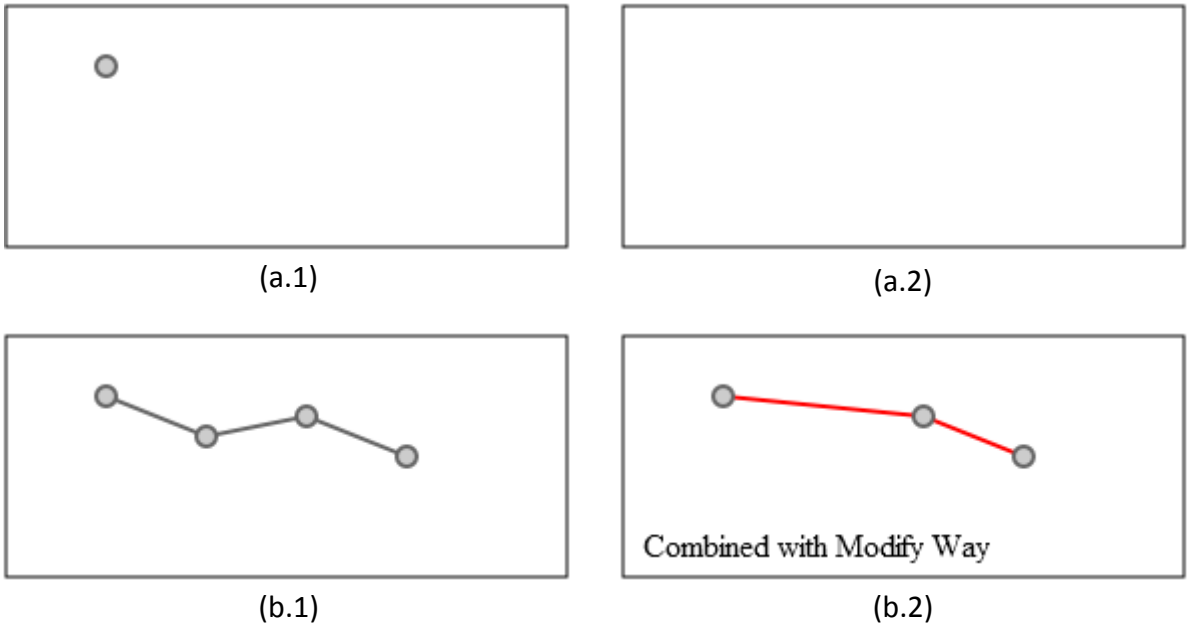


Figure 23: Delete node examples

13.1.2 Way-related operations

This section gives the details about the basic operations for OSM way elements. There are some special operations related to ways (e.g. Merge Way, Split Way, Reverse Way) which are explained in a later section (13.1.5 Other operations).

If the first node equals the last one and if a certain tag is attached to the way, the way is interpreted as a polygon. However, the presented algorithm does not differentiate between line and polygon features.

Create Way

A new line or polygon is created. All nodes which are used have to exist at the time of this operation. Figure 24 shows an example which creates a way with four nodes. Ways should always have at least one primary tag. In opposite to nodes (which can be used by ways), a way without tags has no purpose. Features without primary tags do not have an effect on the result of the quality evaluation (see section 5.4 Evaluation for further details).

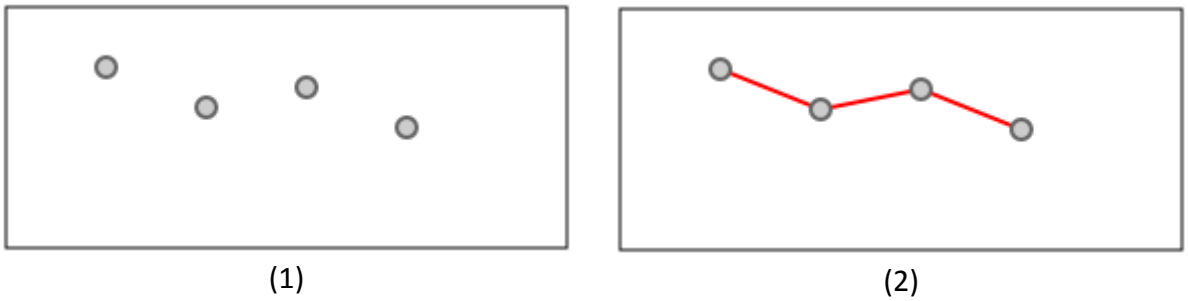


Figure 24: Create way example

Modify Way: Add Node, Remove Node

Ways have an ordered node list, which defines the route of the way. OSM mappers can add and remove existing and visible nodes to this list. The two operations Add Node and Remove Node result from this. Figure 25 gives examples; (a) adds one node to the way, (b) removes one node from it. Situations where new nodes are created are (1) adding nodes directly after the initial creation of a way, (2) adding nodes to improve the positional accuracy of ways, and (3) to connect one way to others at a location where no other nodes exist. Nodes are removed from ways because are not needed anymore (e.g. nodes in a straight way section).

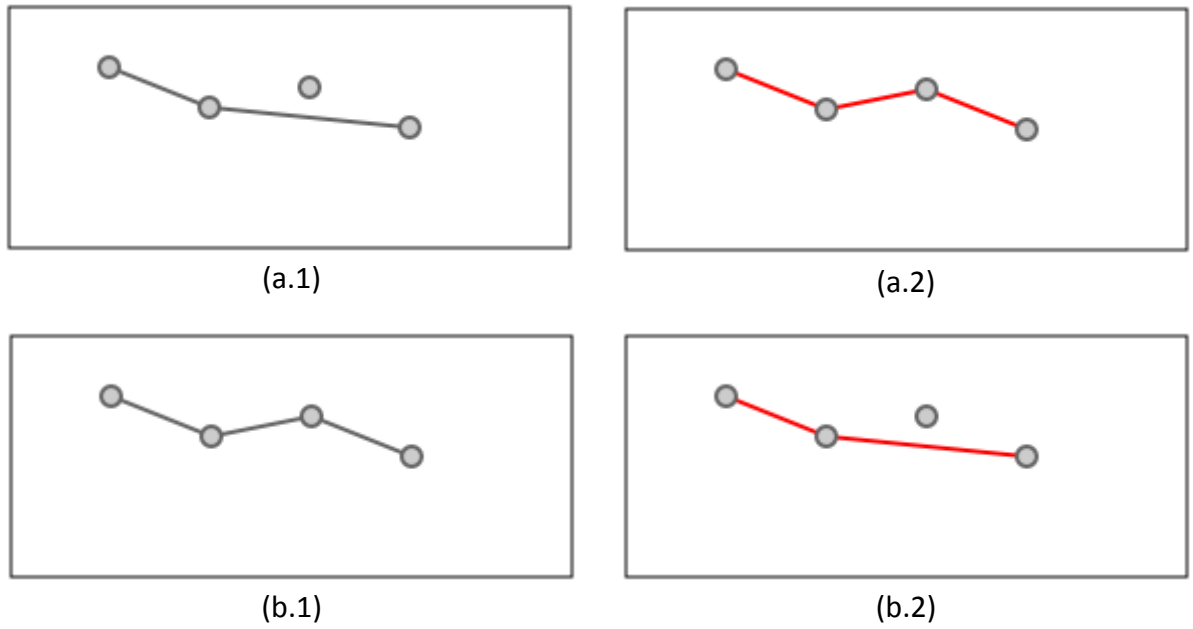
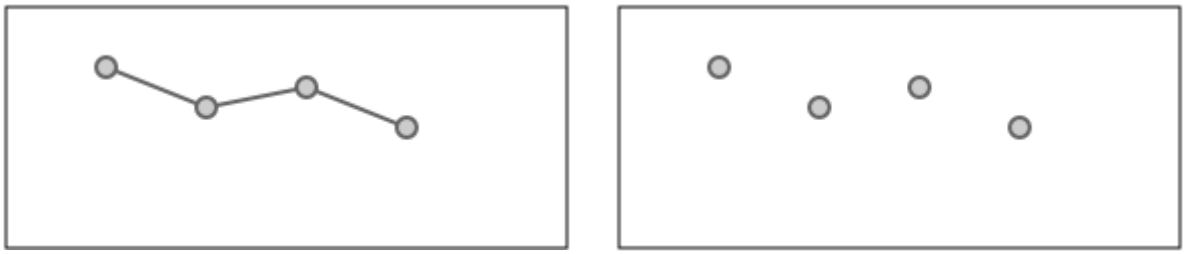


Figure 25: Modify way examples

Delete way

A whole line or polygon is deleted. The nodes belonging to a way will remain except the nodes are deleted too. Figure 26 gives an example of the deletion of a way element. In this example, the nodes are not deleted; thus the remaining nodes can be used for other ways (e.g. operation Merge Way, see section 13.1.5 for details).



(1) (2)

Figure 26: Delete way example

13.1.3 Relation-related operations

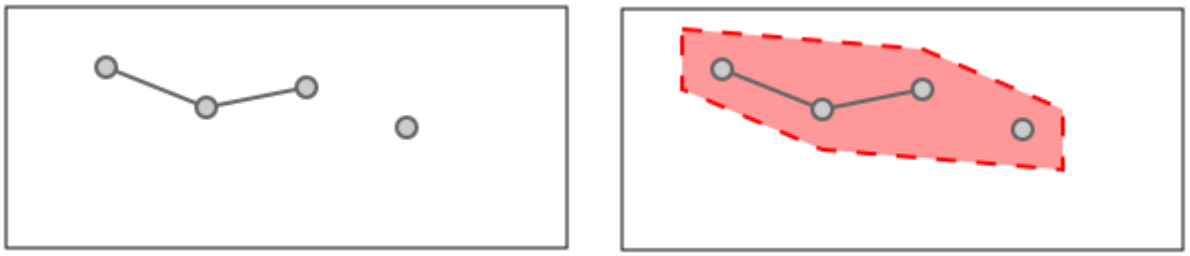
This section explains relation operations. As introduced in section 4.2.1 OSM data model, there are two relation types: (1) Feature Relations and (2) Merge Relations. Feature Relations create a new features, while Merge Relations combine features which belong together (e.g. associated street segments,). Table 19 summarizes assigns the relations to one type.

Table 19: Relation types

Relation type (primary tag with key 'type')	Feature Relation	Merge Relation
associatedStreet		+
boundary		+
bridge		+
destination_sign	+	
dual_carriageway (proposed type)		+
enforcement	+	
junction (proposed type)		+
multipolygon		+
public_transport		+
relatedStreet		+
restriction	+	
route		+
site		+
street		+
tunnel		+
waterway		+

Create (feature/merge) relation

A new relation between objects (nodes, ways or other relations) is created. For example a set of ways is connected to a street. The example in Figure 27 below shows a new relation which contains a way (of three nodes) and a node. The node can be the entrance to a building. The new relation connects both features as they are part of the same street.

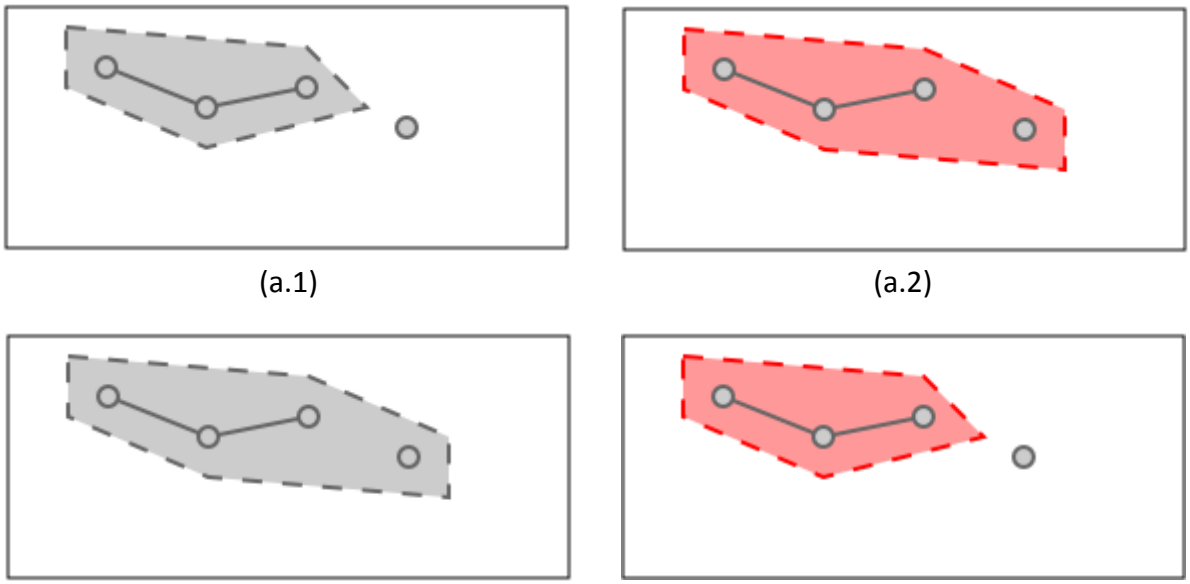


(1) (2)

Figure 27: Create relation example

Modify (feature/merge) relation: Add member, Remove member

Similar to ways (with nodes), relations can be modified. Mappers can add new members and remove old members. Both operations are explained in Figure 28. Example (a) shows the adding of a node feature, (b) the removal of this node feature from the relation. In many cases, new members are added in the background by OSM editors like JOSM or Potlatch. If a way which is member of a relation, is split, the new way which results from the split, is added to the relation automatically.



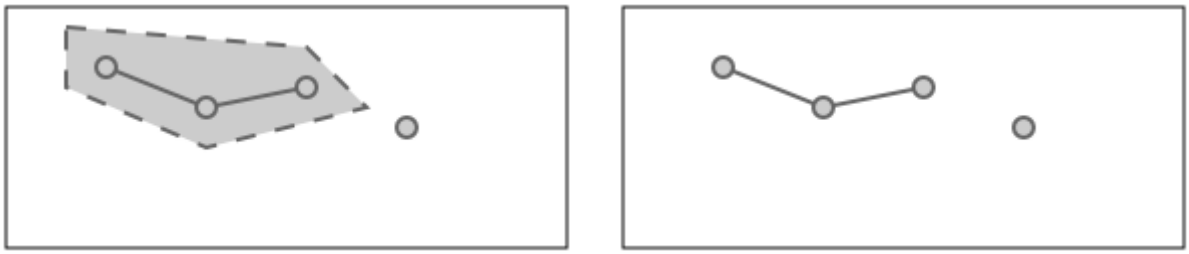
(a.1) (a.2)

(b.1) (b.2)

Figure 28: Modify relation examples

Delete (feature/merge) relation

A relation which consists of nodes/ways/relations is deleted. The members of the relation survive this operation as can be seen in the example in Figure 29 unless they are deleted too.



(1) (2)

Figure 29: Delete relation example

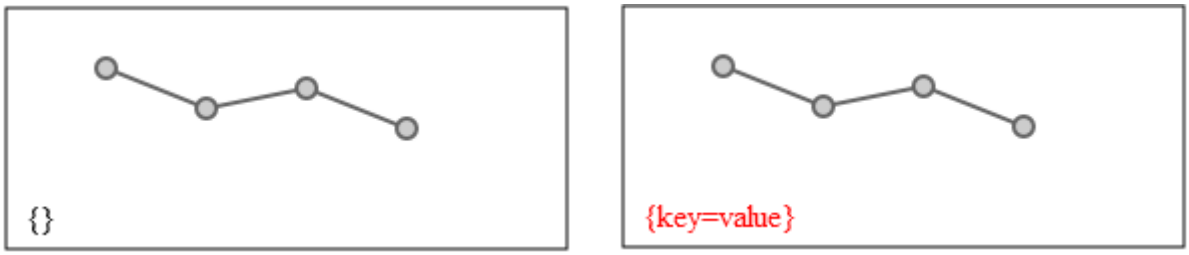
13.1.4 Tag-related operations

In addition to all previously introduced operations, mappers can also edit the feature's tags. Along with the create/modify node/way/relation operations, the attributes of OSM features are added, updated and removed. This section explains these tag operations.

All delete node/way/relation operations also delete all tags associated to the feature. This is not recognized as an operation.

Add Primary Tag, Add Tag

A new tag (key, value) is added to the feature. The tag can be a primary tag (see section 4.2.1 OSM data model (tags) for a list of primary tags) in order to define the nature of the feature. If primary tags are added to a feature, the feature gets typed; therefore it can have influence on completeness, especially in the action of creating features. Figure 30 gives an example, which adds a tag to a way feature.



(1) (2)

Figure 30: Add tag example

Update Primary Tag, Update Tag

A tag value for a feature is modified. The key cannot change. This operation corrects misinformation. The example in Figure 31 updates the value of a way feature.

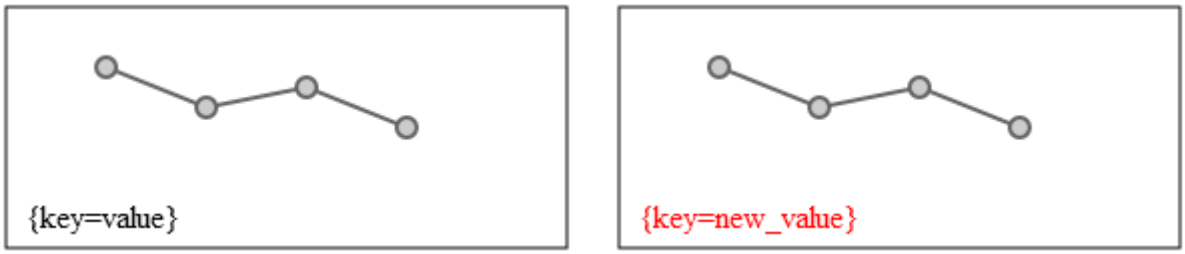


Figure 31: Update tag example

Remove Primary Tag, Remove Tag

If a tag is unnecessary, it will be removed. This tag simply disappears between two feature versions. Figure 32 shows the deletion of a tag of a way feature.

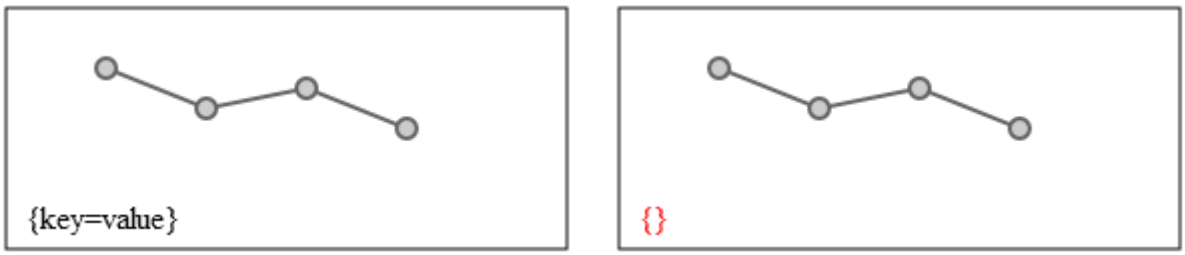


Figure 32: Remove tag example

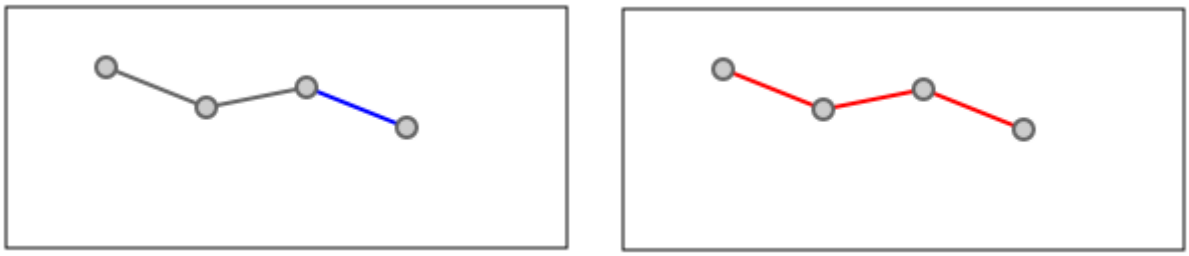
13.1.5 Other operations

During the process of extracting the operations, also some special operations can be extracted. The following list gives an overview:

- Merge Way
- Split Way
- Start/End Node of Way Changed
- Reverse Way
- Replace Feature
- Recreate Feature
- Merge Nodes

Merge way

If two or more subsequent ways with the same tags exist, mappers can merge them to one way. One way is deleted and the other way is modified. The nodes from the deleted way are added to the node list in the right order. Figure 33 gives an example, where the blue segment is added to the grey way, which results in the combined way (red line).



(1) (2)

Figure 33: Merge way example

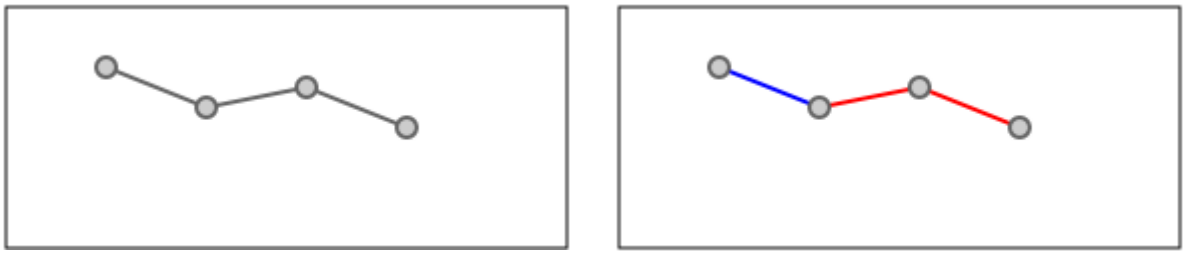
The blue segment represents the second way, which will be merged with the first way.

Split way

One way is created and another way is modified (shortened node list, other nodes are used for new way). The following list gives some examples for Split Way cases:

- Different road classification (e.g. highway type secondary until the beginning of a village, highway type residential for the rest of the way)
- Different surface (e.g. asphalt, gravel)
- Different speed limit
- Bridges and tunnels

An example is shown in Figure 34. The grey way is split into a blue and a red segment.



(1) (2)

Figure 34: Split way example

The blue segment represents the second way, which is created after the split.

Start/End Node of Way Changed

This operation indicates changes of the start or end node of a way. As both examples in Figure 35 show, this includes shortening or lengthening the way on one or both ends. Example (a) removes one node at the beginning of the way; example (b) adds one at the end.

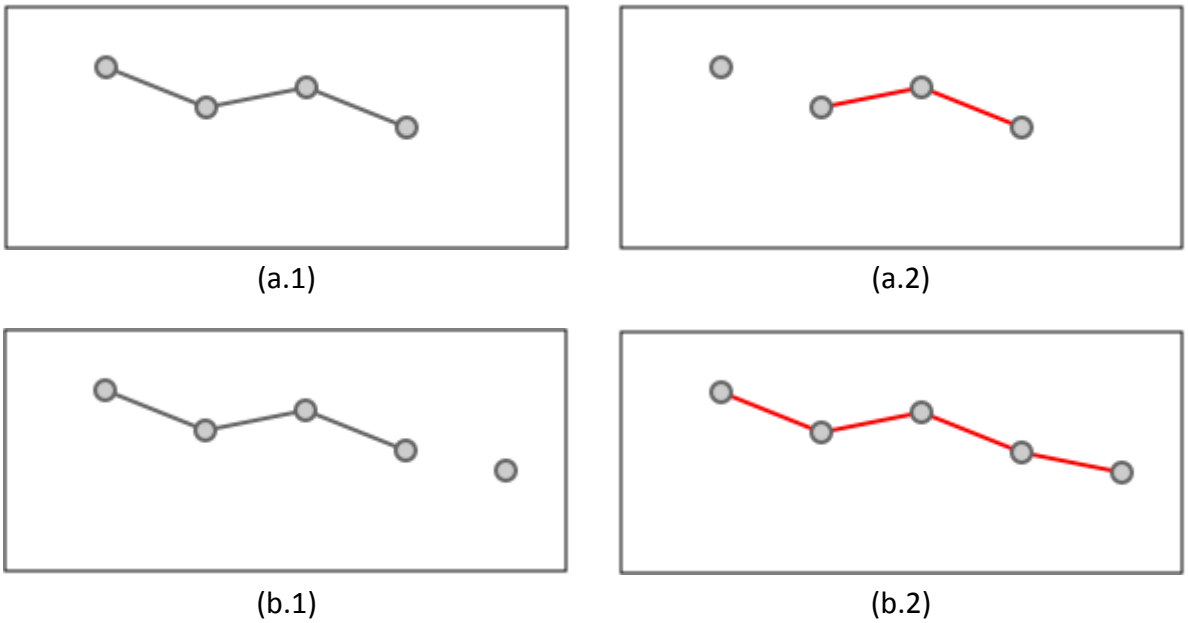


Figure 35: Start/End node of way changed examples

This operation is used to find other operations. Only if a Start End Node Changed operation is found, the algorithm has to look for Merge Way operations.

Reverse Way

In special cases the mapper has to reverse the direction of the way. The order of the nodes is changed, which means that the first node in the previous version is the last node in the successive version of the way. Exceptions occur if nodes are added or deleted in the same feature version. The following list gives examples:

- If a street is changed to a one way street and the way is in the wrong direction, the node order will be changed.
- Other feature types like aerial ways or water ways always have a direction (e.g. flow direction of water, slope indicator for aerial ways like cable cars or gondolas). If the feature was created incorrect its previous versions, the nodes will be reversed.

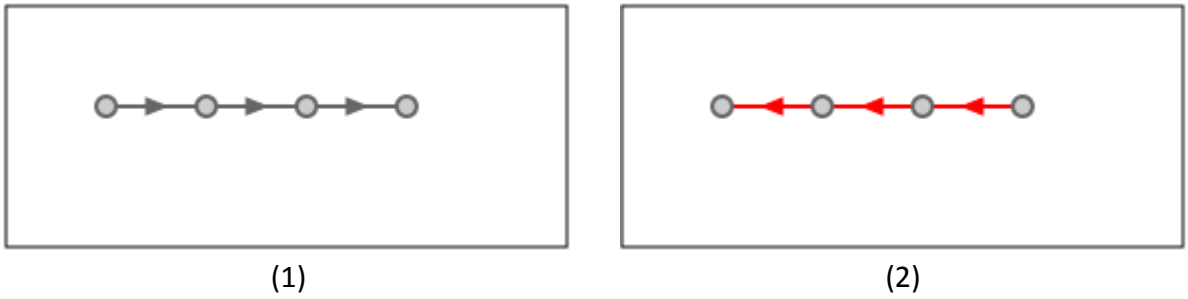


Figure 36: Reverse way example

Replace feature

There are two reasons to replace a feature. (1) The first one occurs if a feature is replaced by a similar geometry (e.g. within buffer around original feature) with the same type (node, way, relation). An example is the replacement of features with a very poor data quality (see Figure 37.a). (2) In the second case the new feature is another type (see Figure 37.b). Here, a point feature is replaced by a line or polygon. This can happen if a mapper replaces a point feature like a church with a polygon feature, which shows the outer borders of the building.

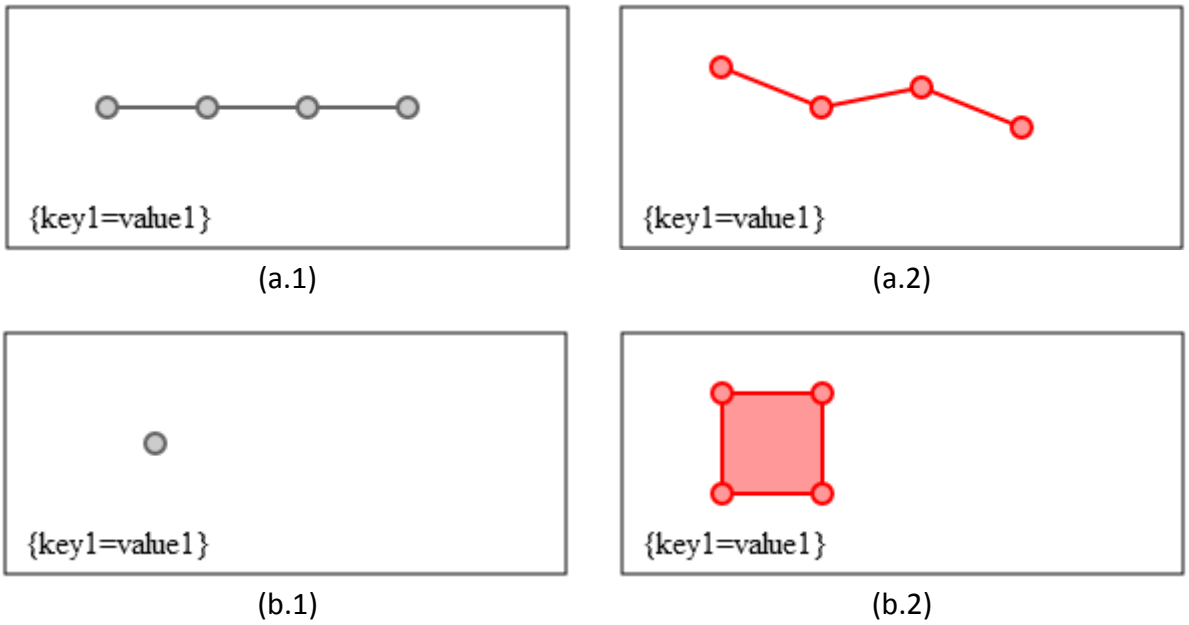


Figure 37: Replace feature examples

It is difficult to extract replaced features, because the feature ID changes.

Recreate feature

After deleting a feature, it is possible to recreate the feature by adding a new “visible” version. Deleted features have the version 2 or higher, therefore a recreated feature has a version number of 3 or higher. Figure 38 shows an example of recreating a feature. (1) First, a node with a certain ID is created, (2) later deleted, and (3) eventually, a new visible version is added to this node.

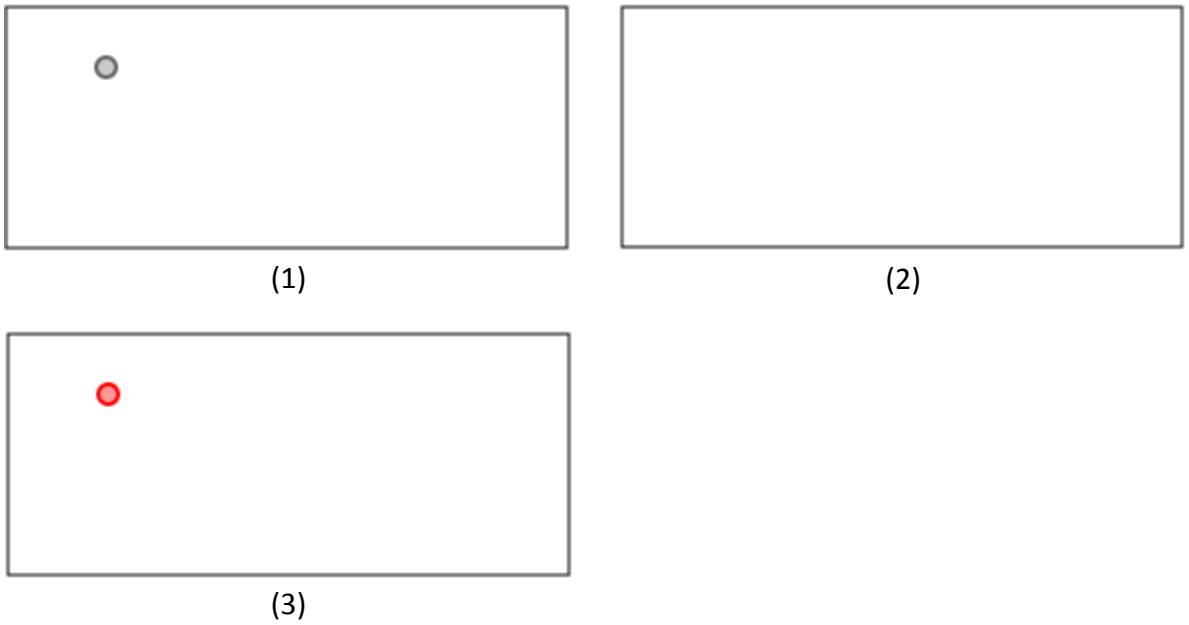


Figure 38: Recreate feature example

Merge nodes with same/similar geometry

If two nodes are located within a short distance they may be merged by a user. In this case, one node is deleted and zero, one or more ways and/or relations are modified. See Figure 39 for an example.

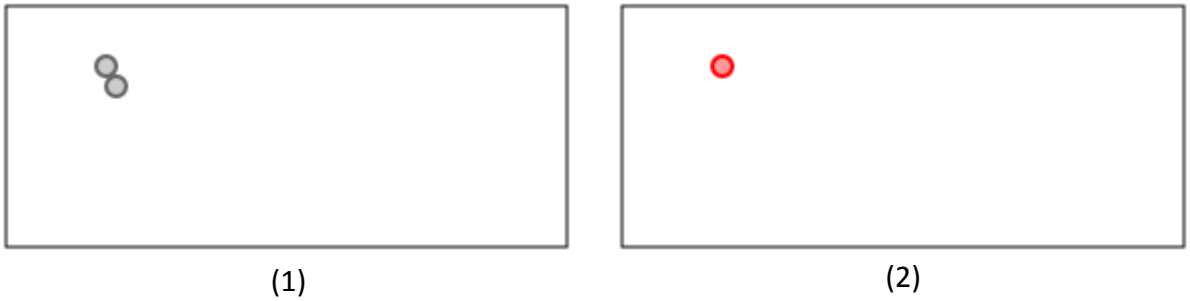


Figure 39: Merge nodes example

Note: This operation is not implemented yet, therefore it is not considered in the reminder of this work.

13.2 Appendix B: Action definition

Ac Action name • **[Number of operations) Op Operation name (Hierarchy level)**
Ac Feature Recreate • [1] Op Feature Recreate (40)
Ac Feature Replace • [1] Op Feature Replace (39) • [1] Op Node Create • [1] Op Way Create • [1] Op Relation Create • [1] Op Feature Add Primary Tag • [1] Op Way Add Node • [1] Op Way Start End Node Changed
Ac Split Line/Polygon • [1] Op Way Split (35) • [1] Op Way Create • [1] Op Way Add Primary Tag • [∞] Op Way Add Node • [∞] Op Way Remove Node • [1] Op Way Start End Node Changed • [1..*] Op Node Update Coordinate
Ac Merge Line/Polygon • [1] Op Way Merge (34) • [∞] Op Way Add Node • [∞] Op Way Remove Node • [1] Op Way Start End Node Changed
Ac Create Point • [1] Op Node Create (30) • [1] Op Feature Add Primary Tag • [∞] Op Node Update Coordinate

Ac Action name • **[Number of operations) Op Operation name (Hierarchy level)**
Ac Create Line/Polygon • [1] Op Way Create (29) • [1] Op Node Add Primary Tag • [2..∞] Op Way Add Node • [∞] Op Node Update Coordinate
Ac Create Feature-Relation • [1] Op Relation Create (27) • [1..*] Op Relation Add Member • Condition: Relation Is Feature == true
Ac Create Merge Relation • [1] Op Relation Create (27) • [∞] Op Relation Add Member • Condition: Relation Is Feature == false
Ac Delete Point • [1] Op Node Delete (25)
Ac Delete Line/Polygon • [1] Op Way Delete (24)
Ac Delete Feature-Relation • [1] Op Relation Delete (23) • Condition: Relation Is Feature == true
Ac Delete Merge Relation • [1] Op Relation Delete (23) • Condition: Relation Is Feature == false
Ac Add Feature Type • [∞] Op Feature Add Primary Tag (15)

Ac Action name • **[Number of operations) Op Operation name (Hierarchy level)**
Ac Update Feature Type • [∞] Op Feature Update Primary Tag (14)
Ac Remove Feature Type • [∞] Op Feature Remove Primary Tag (13)
Ac Add Attribute • [∞] Op Feature Add Tag (12)
Ac Update Attribute • [∞] Op Feature Update Tag (11)
Ac Remove Attribute • [∞] Op Feature Remove Tag (10)
Ac Reverse Line/Polygon • [1] Op Way Reversed (7)
Ac Update Line/Polygon node • [∞] Op Way Add Node (5) • [∞] Op Way Remove Node • [1] Op Way Start End Node Changed
Ac Update Line/Polygon Add node • [∞] Op Way Add Node (5) • [∞] Op Way Remove Node • [1] Op Way Start End Node Changed
Ac Update Line/Polygon Remove node • [∞] Op Way Remove Node (4) • [1] Op Way Start End Node Changed
Ac Add Feature-Relation Member

Ac Action name • **[Number of operations) Op Operation name (Hierarchy level)**
• [∞] Op Relation Add Member (5) • Condition: Relation Is Feature == true
Ac Add Logical Relation Member • [∞] Op Relation Add Member (5) • Condition: Relation Is Feature == false
Ac Update Point Coordinate • [∞] Op Node Update Coordinate (2)
Ac Update Line/Polygon Coordinate • [∞] Op Point Update Coordinate (2)

13.3 Appendix C: Quality matrix example

Activity	Action	building	highway	type	route	amenity	railway	shop	landuse	public_ transport	office	leisure	barrier	tourism	sport	man_ made	craft	natural	SUM
Completeness	Ac Point Create	16,5	73,5	0,0	0,0	21,0	34,0	3,0	0,0	2,5	2,0	0,5	2,0	0,5	0,5	4,0	0,0	1,0	161,0
Completeness	Ac Point Delete	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
Completeness	Ac Line Create	41,3	105,7	0,0	0,0	3,3	1,2	0,0	32,0	2,5	0,0	1,5	0,0	0,0	0,5	0,0	0,0	0,0	188,0
Completeness	Ac Line Delete	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
Completeness	Ac Feature Add Type	9,3	16,5	0,0	0,0	3,3	16,0	4,0	0,5	0,0	0,0	0,0	0,3	0,0	0,0	0,0	0,0	0,0	50,0
Completeness	Ac Feature Remove Type	8,0	0,0	0,0	0,0	0,0	3,5	1,0	0,0	1,5	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	14,0
Completeness	Ac Relation Create	0,0	0,0	17,5	0,5	0,5	0,0	0,0	0,0	0,5	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	19,0
Completeness	Ac Relation Delete	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
LogicalConsistency	Ac Line Split	32,0	6,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	38,0
LogicalConsistency	Ac Line Merge	2,0	0,5	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,5	0,0	0,0	0,0	0,0	0,0	0,0	3,0
LogicalConsistency	Ac Feature Recreate	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
LogicalConsistency	Ac Feature Replace	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
LogicalConsistency	Ac Line Reverse	4,0	1,0	0,0	0,0	1,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	6,0
LogicalConsistency	Ac Relation Add Member	0,0	0,0	176,0	159,5	0,0	0,0	0,0	0,0	5,5	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	341,0
LogicalConsistency	Ac Relation Remove Member	0,0	0,0	19,5	16,5	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	36,0
PositionAccuracy	Ac Line Add Node	148,5	75,5	0,0	0,0	6,5	11,0	3,5	2,0	0,0	0,0	4,0	1,0	0,0	0,0	0,0	0,0	0,0	252,0
PositionAccuracy	Ac Line Remove Node	17,5	3,0	0,0	0,0	0,0	0,0	0,5	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	21,0
PositionAccuracy	Ac Line Update Coordinates	190,0	70,0	0,0	0,0	6,5	9,0	5,0	2,0	0,0	0,0	4,5	1,0	0,0	0,0	0,0	0,0	0,0	288,0
PositionAccuracy	Ac Point Update Coordinate	19,2	41,8	0,0	0,0	53,2	28,3	16,5	0,0	7,8	3,5	0,0	5,7	0,0	0,0	4,0	0,0	0,0	180,0
ThematicAccuracy	Ac Feature Add Attribute	535,0	505,2	30,0	2,5	178,0	56,2	48,5	8,5	22,7	8,0	3,0	2,0	3,0	1,5	10,0	0,0	0,0	1414,0
ThematicAccuracy	Ac Feature Update Type	19,3	3,5	0,0	0,0	2,8	1,0	1,5	0,0	0,8	0,0	1,0	0,0	0,0	0,0	0,0	0,0	0,0	30,0
ThematicAccuracy	Ac Feature Update Attribute	33,0	4,0	17,0	11,0	19,0	0,0	32,5	1,5	0,0	12,0	1,0	0,3	0,7	0,0	0,0	1,0	0,0	133,0
ThematicAccuracy	Ac Feature Remove Attribute	-137,5	-33,0	0,0	0,0	-127,5	-4,5	-59,5	-2,5	-3,0	-1,0	-0,5	-2,0	-1,0	0,0	-16,0	0,0	0,0	-388,0
	SUM	938,2	873,2	260,0	190,0	167,7	155,7	56,5	44,0	40,8	24,5	15,5	10,3	3,2	2,5	2,0	1,0	1,0	2786,0