Project number:	288577
Project acronym:	UrbanAPI
Project title:	Interactive Analysis, Simulation and Visualisation Tools for Urban Agile Policy
Instrument:	STREP
Call identifier:	FP7-ICT-2011-7
Activity code:	ICT-2011.5.6 ICT Solutions for governance and policy modelling

Start date of Project:	2011-09-01
Duration:	36 month

Deliverable reference number and title (as in Annex 1):	**D3.5 Data Integration Components Documentation**
Due date of deliverable (as in Annex 1):	
Actual submission date:	*see "History" Table below*
Revision:	**1**

Organisation name of lead contractor for this deliverable:
Fraunhofer IGD

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Title:
Data Integration Components Documentation

Author(s)/Organisation(s):

Jens Dambruch / Fraunhofer IGD, Jan Peters-Anders / AIT, Ernst Gebetsroither / AIT

Working Group:

WP3 – Fraunhofer IGD, AIT

References:

Description of Work, D2.1, D3.2, D3.3, D4.1

Short Description:

A description on the components used to integrate the data for the UrbanAPI applications. It explains the approach to integrate the data provided by the cities into the UrbanAPI applications. The basic approach to integration with CityServer3D is outlined as well as integration of GSM and urban-growth data.

Keywords:

Interoperable, integration, data, GIS

History:

Version	Author(s)	Status	Comment	Date
001	Jens Dambruch	New	Initial draft and table of contents	5.11.2012
002	Ernst Gebetsroither	Rfc	Added urban growth modelling data	18.12.2012
003	Jan Peters-Anders	Rfc	Added urban motion explorer	19.12.2012
004	Jens Dambruch	Rfc	Rework after comments from Achim Rix	11.01.2013
005	Ernst Gebetsroither	Rfc	Rework after comments from Achim Rix	11.01.2013
006	Jan Peters-Anders	Final	Rework after comments from Achim Rix	14.01.2013

Review:

Version	Reviewer	Comment	Date

006	Wolfgang Stemberger, Wolfgan Loibl		15.01.2013

About this Document

The UrbanAPI applications are tools to analyze and visualize various data sets of the cities involved. Even urban-growth modelling can be conducted on population density distributions. Mostly these data sets have some spatial reference or are exported from some GIS solution already in use inof the city. This document explains the approach to integrate the data provided by the cities into the UrbanAPI applications. According to the description of work for UrbanAPI [1] it is the deliverable *D3.5 – Data Integration Components* of work package WP3: *Software Platform Development: Database, Integration, Visualisation and Simulation Tools*. The document will be superseded by an improved version if required at any time in the project life cycle. The finalized version of this document is planned for month 30.

1 Introduction

Urban planning involves a lot of geospatial data, from cadastral data to infrastructure plans like streets or electricity, 3D-models of buildings or data from aerial laser scans. This data shall be used to enable a ICT governance of the city. The goal is to have an integrated view on aspects covered by that data. This means not only to consider one aspect like infrastructure planning but also to asses impacts on other data sets like pollution for planned actions. The UrbanAPI applications offer tools that will be used to visualize and analyze these data sets that way, but for this the data sets must be integrated and some must be probably harmonized. In the following chapters our approach and the tools that will be used for this integrations will be explained.

2 The 3DVR application and CityServer3D

The CityServer3D is motivated by the fact that spatial data is growing in volume and gains more and more importance, but also the heterogeneity among the data sets is an obstacle for the efficient use of it. So a interoperable system is a crucial point for any geo-data-infrastructure. The CityServer3D is based on a object oriented architecture derived from experiences with 3D-city-models and their applications. Figure 1 shows an internal structure of such an interoperable System [2]. All components of the system are arranged as spokes to the central hub or core of the system: The Metamodel. The first layer in yellow is about interfaces, data processing and data sources, which are all based on the same generalized Metamodel. The green layer is getting closer towards specific technologies like network protocols, files and processing applications. The last layer in grey represents concrete applications like OGC services W3DS, WMS or route planing applications, which are available to the clients of the GIS. The CityServer3D is an Implementation of such an architecture and can server as a basis for modules that will be developed in the UrbanAPI context. For example centralized data-management with database support and visual support for administration is helpful for the integration of the collected UrbanAPI data sets.

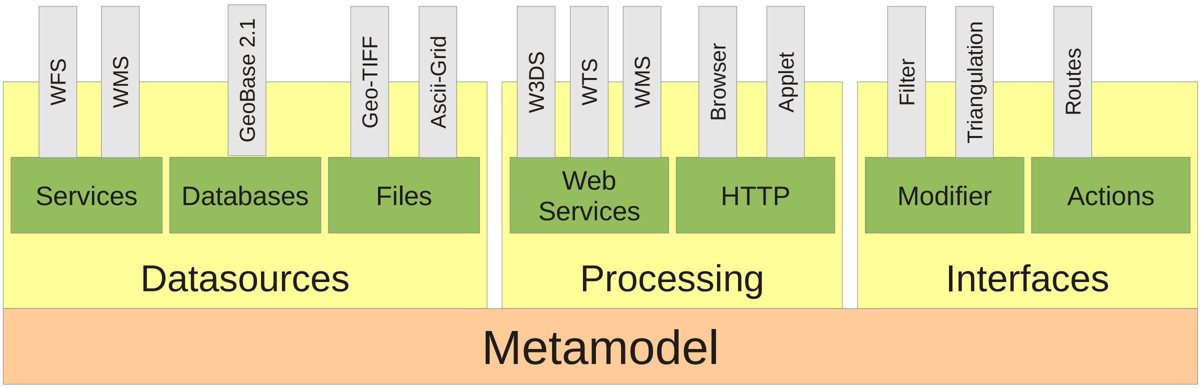


Figure 1: Overview of the architecture of an interoperable GIS. Transfered and simplified from [2].

- *Orange: The Metamodel is the foundation for all data-management*
- *Yellow: Categories of usages – The Metamodel serves as a hub for datasource adaptations, model for processing logic and serves standardized interfaces*
- *Green: Characteristical domains, protocol layers or connectors uses*
- *Grey: Practical applications – Concrete applications of the categories where the Metamodel is used, e.g. implementation of W3DS or importer for Ascii-Grids*

2.1 The Metamodel

The cornerstone of an interoperable GIS is how data is represented. The design goal for CityServer3D is a GIS that is extendable, flexible and adaptable to different data formats to cover all important standards.

The concept to realize this is a data model that integrates all aspects of GIS data. This is an internal model, the communication to external sources in external formats is completely covered by specialized data access components and never exposes its internals to the outside. Abstraction of a data access is a typical design pattern is software architecture that hides peculiarities of the data access. All modules for data processing can now work with this common Metamodel and there is no need to deal with the peculiarities of the external data formats. Based on this it is very easy to implement converters for data, as all functionalities that is already available for the management of Metamodel based data can be used right away. The supported formats are listed in Supported formats of CityServer3D, all these can be used as input to the UrbanAPI database without additional development efforts. in table 1. From the interface perspective also the conversion from data in relational databases or models like WFS/GML3 are now easy to adapt.

Format	*Import*	*Export*
CityGML 1.0	y	y
NAS/ALKIS	y	n
DXF/DWG	y	y
Shapefiles	y	y
GeoTIFF	y	n
DGM(ESRI TIN, ESRI Grid)	y	y
XYZ/ASC	y	n
Waypoint+	y	n
Garfile	y	n
VRML	y	y
X3D	y	y
GeoSciML	y	n
PCI+RAW	y	n
KML/KMZ	n	y

Table 1: Supported formats of CityServer3D, all these can be used as input to the UrbanAPI database without additional development efforts.

2.2 Details of the Architecture – Data-Providers

A convenient abstraction for programmers to the data access is the so called Data-Provider. This is a abstraction pattern in software engineering which targets on hiding the technical details and peculiarities on retrieving data. For example the data can be a file, an entry in a database or on a website. On the other hand data may be stored in different formats, for example images can be stored JPG or PNG. From a users perspective this is irrelevant as the user wants only the bitmap data, no matter what the file format may be. Therefore the data provider pattern defines what should be supplied and not how. The aspect of how data is gathered is therefore hidden as an implementation detail. These data providers can be used to access a data source in a format agnostic way, all entities of data are represented by CityServer3D Metamodel elements. Based on this it is very easy to build common processing logic like for example the rule based system as described in [3] which is used for data setup and data processing in the UrbanAPI applications.

2.3 Special developments – The CityGrid-Adapter

The city of Vienna uses the CityGrid solutions of UVM-Systems [4] for the management of their 3D-city-model. This software offers to export the data contained in a proprietary XML-format that cannot be converted to a standard format. As outlined in the previous sections, an adapter to this format will be developed to be able to import the 3D-city-model of Vienna into the UrbanAPI applications. This adapter transfers the data in the CityServer3D Metamodel and from there all standard formats supported by CityServer3D are available.

3 Motion Explorer Data

The Motion Explorer application will be based on GSM and socio-economic data imported from proprietary file formats (e.g. ESRI shapefiles, A1 binary log files, text files, MS Excel files etc.). The A1 mobile device tracking data requires to be heavily pre-processed in order to be applied in a Motion Explorer database.

The most important requirement is to ensure privacy of the contained (already anonymised) mobile device location data, which allow to track individual trips within a certain location accuracy) and to remove location errors. As a result, the Motion Explorer application will have only few possibilities to upload data to the underlying database. Individual mobile device movement data cannot be uploaded to the system, as only aggregated data – aggregated for certain raster cells to avoid interfering privacy - can be published.

Following is a short description of the GSM data that have been delivered to the UrbanAPI project (which also shows the huge diversity of the GSM data available):

- A1 Telecom data: during the first months of the project the data sets delivered by the A1 telecom company differed dramatically in format, content and quality which had to be dealt with identifying problems within the datasets. Later data showed better content. An example of the necessary data analyses and transformations to have the data integrated into urbanAPI’s Motion Explorer is depicted in Figure 2 and Figure 3.

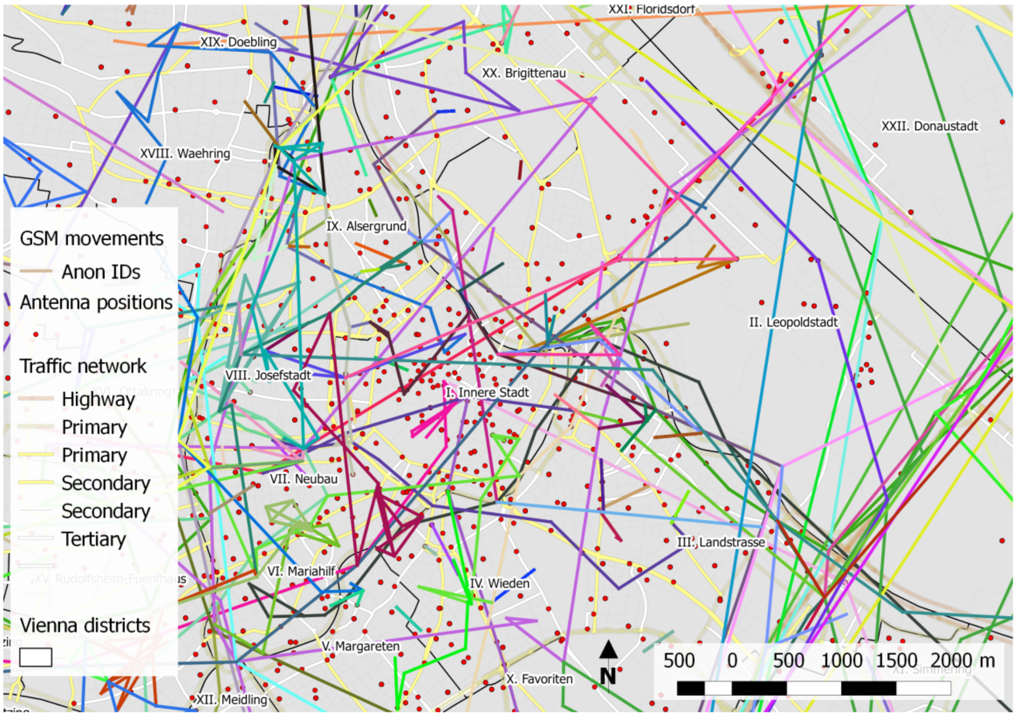


Figure 2: Data integration example: Visualisation of primary data of A1 GSM movement data derived from a generic binary log file converted to a GIS dataset showing a subset of 3000 anonymous IDs movements.

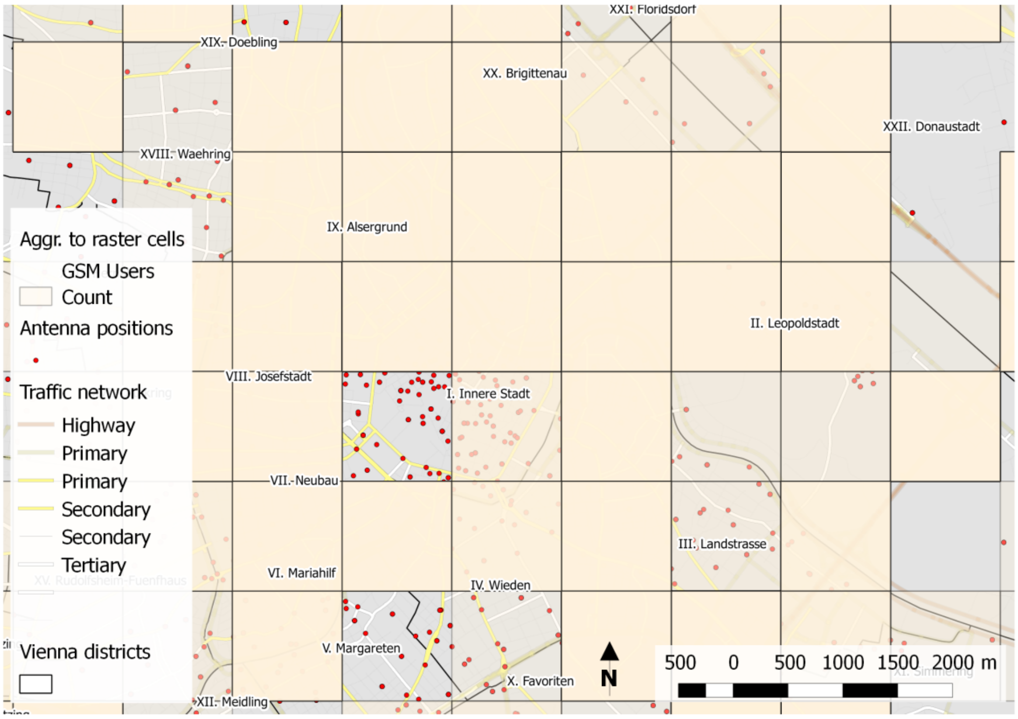


Figure 3: Data integration example: Visualisation of derived data of A1 GSM movement data: Aggregation of users based on a 1km raster grid and a timestamp.

- Telefonica data: The first datasets from the Spanish Telefonica company for the Vitoria-Gasteiz Region are standard CSV tables which show geographic coordinates of the beginning and end of phone calls or text transfer. Here it is also necessary interpolate the location for certain useful time steps and then to aggregate the data to raster cells (Figure 4).

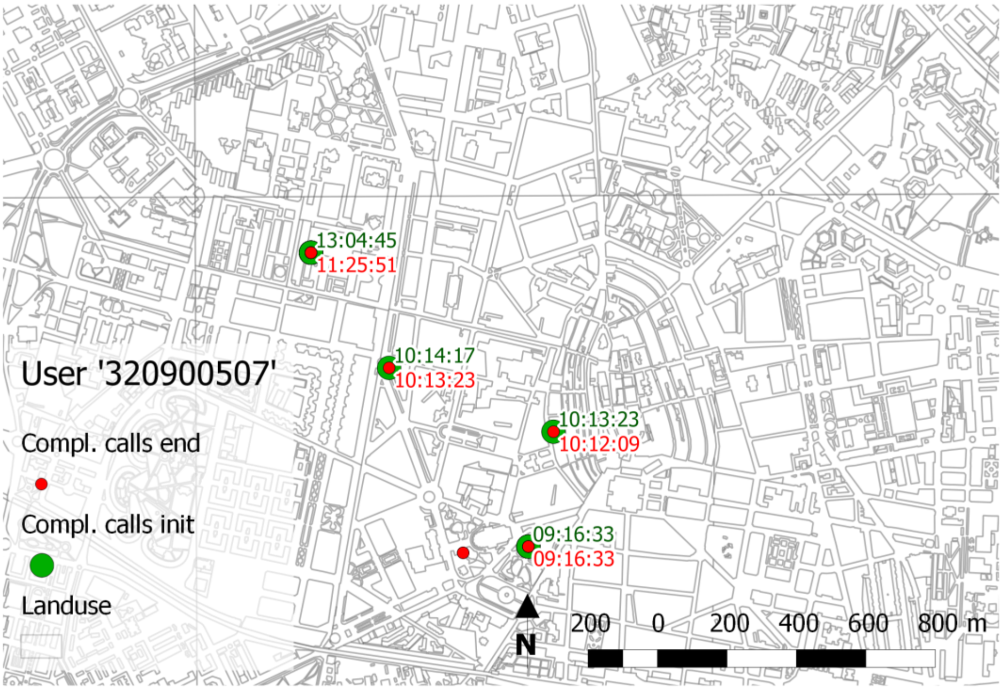


Figure 4: Data integration example: Visualisation of primary data (tables as text files converted to a point layer) received from Telefonica Spain for Vitoria Gasteiz showing GSM antenna positions and a user's coordinates when initiating (init) and ending (end) a call on Monday May 16th, 2011

- TIM data: The datasets delivered by TIM (Italia) for Bologna Region arrived in early January 2013 and were re-projected from the proprietary a national coordinate system to geographical coordinates (WGS84) by TIM which means that they will have to be projected to LAEA/ETRS89 (which is used in UrbanAPI) in a next step. The data description states:
  “The traffic files contain the information on the number of active users in each TIM network cell. These number is associated with the pixels of the city of Bologna considering both the radioelectric coverage of each cell, calculated through the TIM network planning software tool, and the probability of having some user traffic on each pixel.”[1]
  As far as the test TIM data sets could be read until now, the files contain the number of telephone users within certain raster cells. The time stamp is delivered too, which allows then mapping of mobile device user densities (Figure 5).

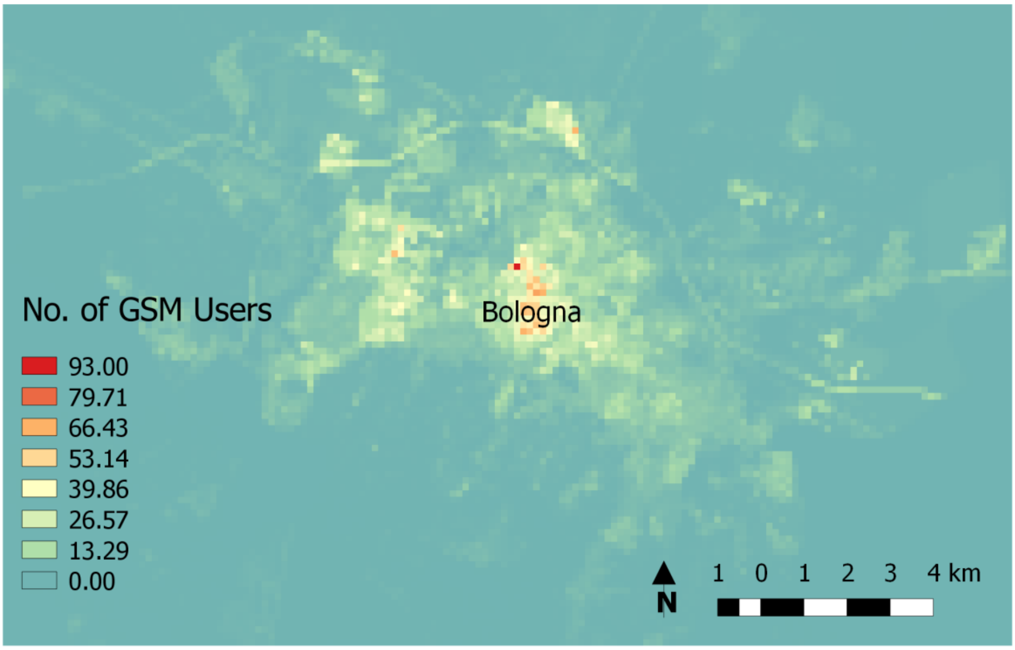


Figure 5: Data integration example: Depiction of primary data (proprietary ASCII raster) received from TIM showing a heat map of the number of GSM users in Bologna on May 4th, 2012, 16:30 to 17:00

The pre-processing of these data is described in deliverable D4.1 “Integrated and Harmonized Data” chapters 5.2, 6.2 and 7.2. As parts of these data show varying record structure and formats over time, others are just test data sets or data newly arrived without proper coordinates, until now no final rules on integrating these data could be defined.

3.1 Import of Pre-processed Data

The Motion Explorer application uses PostgreSQL as its backing database. In order to load the pre-processed data an urbanAPI administrator has to be contacted to do this on the urbanAPI database server.

[1] Source: FileDescription_UrbanAPI_20121119_eng.doc, accompanying file to the data sets delivered by TIM in January 2013

He or she will use command line tools like PostgreSQL’s psql1 terminal or graphical tools like pgadmin III 2 to load the datasets as tables into the Motion Explorer database. To ensure good performance during application requests to the database PostgreSQL functions have been implemented within the database to make use of the built-in Query Optimizer (see below for an example of a PostgreSQL function implemented for urbanAPI showing the creation of so called time slices of the movement patterns in Vittoria Gasteiz).

```
DECLARE
   tblVar text := 'myTable';
   count int4 := 21759720;
   result int4 := 0;
   minutes int4 = 0;
   ts_minutes int4 = 21758280;
   tblName text = relname || '_slices';
   --relname_debug text = 'night_user_from_1kme4791n2810_during_day_the_geom';


BEGIN
--EXECUTE 'ALTER TABLE '|| quote_ident(schema) ||'.'|| quote_ident(relname) ||' ADD 
COLUMN the_geom_4326 geometry;';
EXECUTE 'CREATE TABLE '|| quote_ident(tblName) || ' (count int4, cellcode varchar, 
timestamp timestamp)';
BEGIN
     LOOP
      result := result + 1;
      --tblVar := tblName || to_char(ts_minutes, '99999999');
EXECUTE 'INSERT INTO '|| quote_ident(tblName)  || '
SELECT count(*),  a.cellcode,'''|| to_timestamp(ts_minutes*60) ||'''  FROM(
select anonid,the_geom,ts, cellcode,row_number()
over (partition by anonid order by the_geom,ts desc)
from '|| quote_ident(relname) ||'
where  ts <= '|| ts_minutes ||') AS a
WHERE row_number = 1
GROUP BY cellcode';


 minutes = minutes + 15;
 ts_minutes = 21758280 + minutes;
      EXIT WHEN ts_minutes = count;
    END LOOP;
    RETURN minutes;
END;
END;
```

Example of an urbanAPI PostgreSQL function

3.2 Resulting data structure

In order to have a best practice for future users of the urbanAPI Motion Explorer application it will be necessary to have a well-defined final data structure for the pre-processed and imported data. This structure is (at the moment) based on a 1km grid (finer resolutions seem not to be appropriate due to the relatively coarse information contained within the GSM data) and is consisting of three attribute columns:

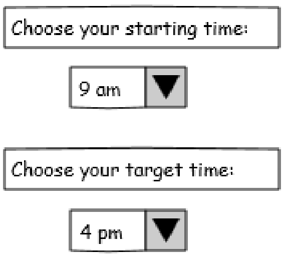
- Timestamp
- Raster id and
- User count

where the time-stamp is divided into 15 minutes steps for better performance of the data base queries and their visualisation.

This simple, yet powerful data structure will be organised in a way that allows for easy visualisation of the data in a 3D environment like CityServer3D to integrate a GSM motion layer into its graphical user interface.

3.3 Data Export Functionalities

The web-application will be capable of exporting data according to the users' needs. First the user chooses the way he or she wants to have the result of their request visualised (Figure 6, Figure 7 and Figure 8).



Choose your starting time: 9 am

Choose your target time: 4 pm

Figure 6: User chooses desired data

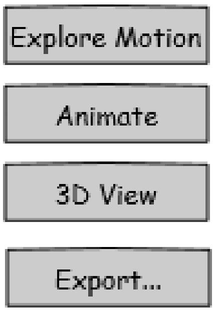


Figure 7: User chooses the way the result shall be visualised

The resulting map (Figure 8) will be displayed and the user will have the possibility (via the "Export" button) to choose which kind of format they would export.

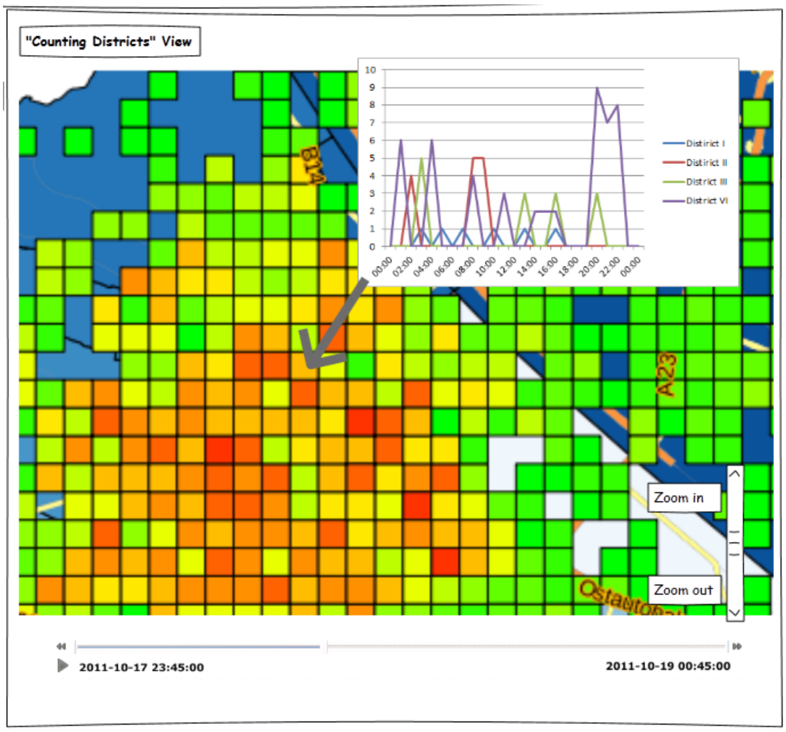


Figure 8: Result map

The database will be queried by the Motion Explorer web-application and will deliver data according to the users' requests. The user can choose between images (i.e. a screenshot), animations (i.e. a video of the time range he or she has chosen) or the result data in table format, e.g. an MS Excel spread sheet or a CSV file holding e.g. O-D matrices. Additionally it might be possible to export 3D formats like X3D to reuse them in other applications as well.

4 Urban Growth Modelling Data

The Urban Growth Model (UGM) will be developed using MASGISmo a (**M**ultimethod **A**gent-based, **S**ystem Dynamics and **GIS mo**delling platform [5]) which former had been created at the AIT (Austrian Institute of Technology). MASGISmo currently includes a database connection to a PostgreSQL database, wherein tabular data can be stored. Geo-data at the moment can only be imported and exported in a non-proprietary ASCII[2] raster[3] format developed by ESRI[4], which is a commonly used raster format. The ASCII text files are imported during the simulation. Simulation results as spatial maps can be exported as ASCII grid enabling to use the data afterwards in specialized Geographic Information System software as Quantum GIS[5]. For the UGM is it planned to introduce a PostGIS[6] connection. This allows including support for geographic objects within the PostgreSQL database. Thus ESRI-shapefiles or other vector based geo-data[7] can be used for the simulation as well. Since PostGIS 2.0 also raster data is supported, nevertheless it has to be tested if this would be a better way to use the raster data compared to the current version of using simple ASCII text files. The CityServer3D will support both the raster data as ASCII format as well as vector based geo-data (e.g. ESRI-shapefiles). It should to be noticed here, that the current version of this documentation for the UGM lacks the information of the needs for the urban planners of the City Ruse (Bulgaria), the test case of the UGM. Currently it is not clear which kind of data in which format really exits and therefore the data integration components only can be described in a very preliminary way. One aim of the project is to support the respective city in its planning purpose as far as possible and therefore special data handling functions which are currently not known might be necessary. Because MASGISmo and the use of open source software as PostgreSQL and PostGIS enables to adapt to the user’s needs in a wide range of cases by e.g. programming new methods in JAVA, also import and export from or to specific data formats seems possible if necessary.

4.1 Pre-processing of tabular data

As mentioned above will the UGM use a PostgreSQL database to store tabular data. MASGISmo, the used modelling platform, already can import and export tables stored as CSV-files[8]. These files can easily be created from Excel spreadsheet files. MASGISmo uses a determined structure for these CSV- files because this enables the user to store metadata for each column in the DB within a separate table. This means during data import the information (metadata) as *UNIT, COMMENT* depicted in the figure 9 below are automatically stored in a separate table. Nevertheless normal CSV-files, missing this special structure, can also be imported to the DB.

For the urbanAPI project a Database Browser will be developed enhancing the already existing data handling possibilities of MASGISmo. Figure 10 shows a screenshot of the draft version of this tool.

[2] American Standard Code for Information Interchange http://en.wikipedia.org/wiki/ASCII, tested Dec. 18, 2012

[3] Raster images are visual representations of numerical information such as elevation. They are resolution dependent, this means they cannot scale up to an arbitrary resolution without loss of apparent quality.

[4] http://resources.esri.com/help/9.3/arcgisengine/java/GP_ToolRef/spatial_analyst_tools/esri_ascii_raster_format.htm, tested Dec. 18,2012

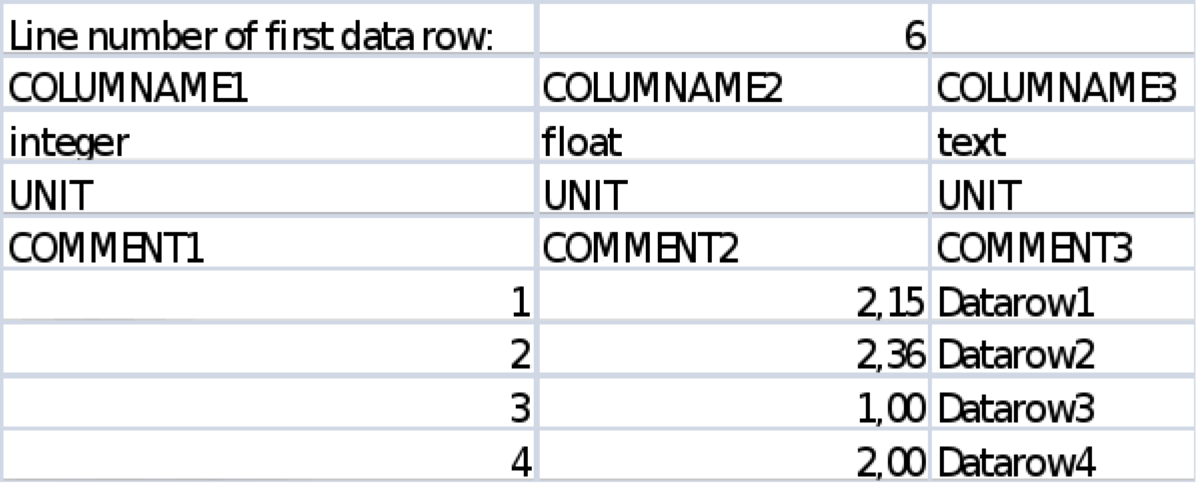
[5] http://www.qgis.org/ , tested Dec. 18,2012

[6] http://postgis.refractions.net/ . tested Dec. 18,2012

[7] Vector data defines features by their geometry as a point, line, or polygon. This kind of geographic layer is infinitely scalable without loss of quality.

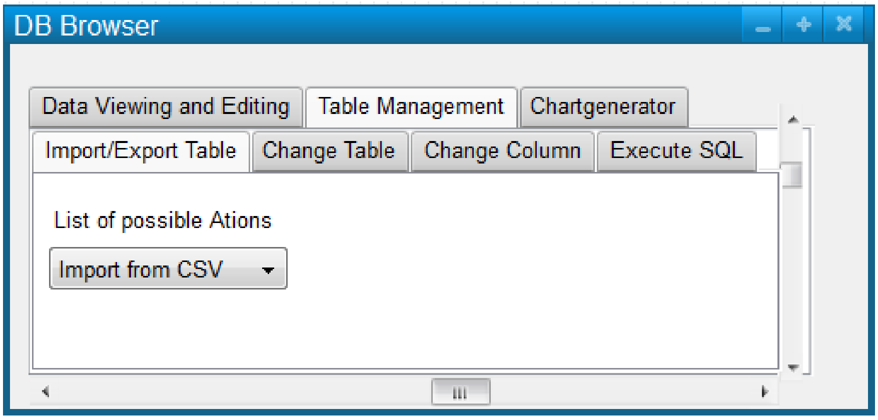
[8] Comma-seperated values files

The DB Browser has also the functionality to view and edit data within a table see figure 10 below. Thus the user can retrieve metadata information for each value in a table in an user-friendly way.



Line number of first data row:	6	
COLUMNAME1	COLUMNAME2	COLUMNAME3
integer	float	text
UNIT	UNIT	UNIT
COMMENT1	COMMENT2	COMMENT3
1	2,15	Datarow1
2	2,36	Datarow2
3	1,00	Datarow3
4	2,00	Datarow4

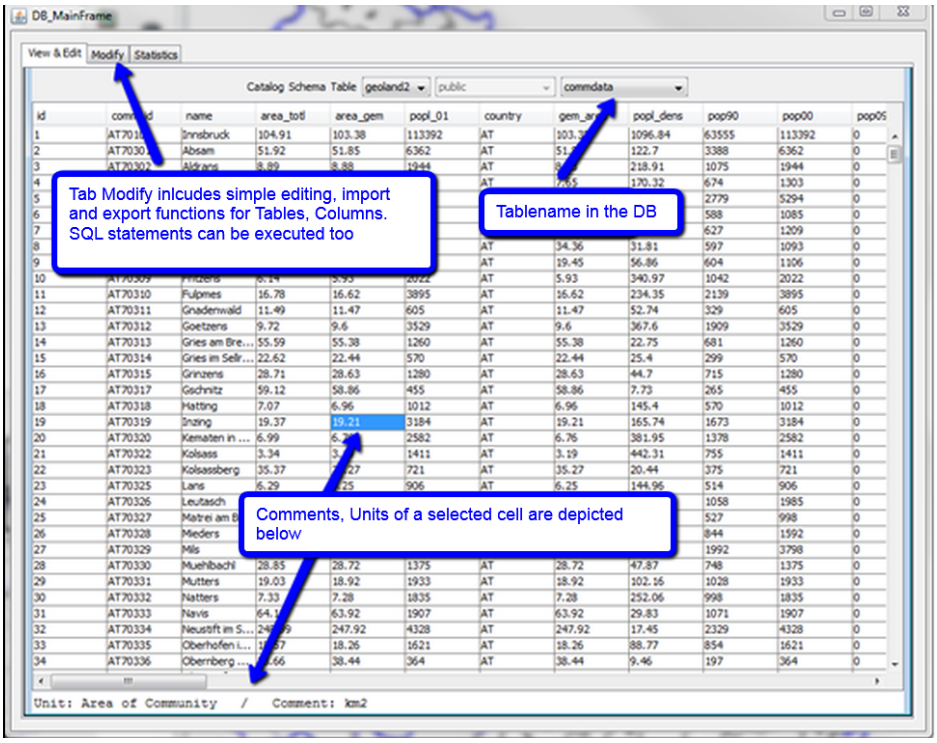
Figure 9: CSV-file structure for data import



List of possible Ations

Import from CSV

Figure 10: Database Browser (draft version)



Catalog Schema Table geoland2 public commdata

id	com d	name	area_totl	area_gem	popl_01	country	gem_ar	popl_dens	pop90	pop00	pop05
1	AT701	Innsbruck	104.91	103.38	113392	AT	103.3	1096.84	63555	113392	0
2	AT7030	Absam	51.92	51.85	6362	AT	51.	122.7	3388	6362	0
3	AT70302	Aldrans	8.89	8.88	1944	AT	8.	218.91	1075	1944	0
4						AT	7.65	170.32	674	1303	0
5									2779	5294	0
6									588	1085	0
7									627	1209	0
8						AT	34.36	31.81	597	1093	0
9						AT	19.45	56.86	604	1106	0
10	AT70309	Fritzens	6.14	5.93	2022	AT	5.93	340.97	1042	2022	0
11	AT70310	Fulpmes	16.78	16.62	3895	AT	16.62	234.35	2139	3895	0
12	AT70311	Gnadenwald	11.49	11.47	605	AT	11.47	52.74	329	605	0
13	AT70312	Goetzens	9.72	9.6	3529	AT	9.6	367.6	1909	3529	0
14	AT70313	Gries am Bre...	55.59	55.38	1260	AT	55.38	22.75	681	1260	0
15	AT70314	Gries im Sellr...	22.62	22.44	570	AT	22.44	25.4	299	570	0
16	AT70315	Grinzens	28.71	28.63	1280	AT	28.63	44.7	715	1280	0
17	AT70317	Gschnitz	59.12	58.86	455	AT	58.86	7.73	265	455	0
18	AT70318	Hatting	7.07	6.96	1012	AT	6.96	145.4	570	1012	0
19	AT70319	Inzing	19.37	19.21	3184	AT	19.21	165.74	1673	3184	0
20	AT70320	Kematen in ...	6.99	6.7	2582	AT	6.76	381.95	1378	2582	0
21	AT70322	Kolsass	3.34	3.	1411	AT	3.19	442.31	755	1411	0
22	AT70323	Kolsassberg	35.37	27	721	AT	35.27	20.44	375	721	0
23	AT70325	Lans	6.29	25	906	AT	6.25	144.96	514	906	0
24	AT70326	Leutasch							1058	1985	0
25	AT70327	Matrei am B							527	998	0
26	AT70328	Mieders							844	1592	0
27	AT70329	Mils							1992	3798	0
28	AT70330	Muehlbachl	28.85	28.72	1375	AT	28.72	47.87	748	1375	0
29	AT70331	Mutters	19.03	18.92	1933	AT	18.92	102.16	1028	1933	0
30	AT70332	Natters	7.33	7.28	1835	AT	7.28	252.06	998	1835	0
31	AT70333	Navis	64.1	63.92	1907	AT	63.92	29.83	1071	1907	0
32	AT70334	Neustift im S...	24 9	247.92	4328	AT	247.92	17.45	2329	4328	0
33	AT70335	Oberhofen i...	1 57	18.26	1621	AT	18.26	88.77	854	1621	0
34	AT70336	Obernberg ...	.66	38.44	364	AT	38.44	9.46	197	364	0

```
Unit: Area of Community   /   Comment: km2
```

Tab Modify inlcudes simple editing, import and export functions for Tables, Columns. SQL statements can be executed too

Tablename in the DB

Comments, Units of a selected cell are depicted below

Figure 11: Database Browser main frame

4.2 Data handling of geo-data

As mentioned above is the plan to enhance MASGISmo’s capability to use geo-data in the form of ESRI ASCII raster files by connecting it to PostGIS. Due to this connection it will be possible to use a lot of different geo-data files as ESRI shapefiles or GeoTiff’s. The DB Browser will therefore be enhanced including simple import and export functions for such kinds of geo-data, similar to that one for CSV-files.

References

1: J.Rix et al., Grant agreement: Collaborative project Annex I - Description of Work, Project: Interactive Analysis, Simulation and Visualization Tools for Urban Agile Policy Implementation, 2010

2: Thorsten Reitz, Architektur eines interoperablen 3D-GIS unter besonderer Berücksichtigung von Visualisierungsanwendungen, 2005

3: Michel Krämer, Andreas Stein, D3.3 - Rule User Interface Documentation, 2012

4: http://www.uvmsystems.com/index.php/en/

5: http://systemsresearch.ac.at/exchange/gebetsroither/Tutorial_von_MASGISmo/Welcome.html