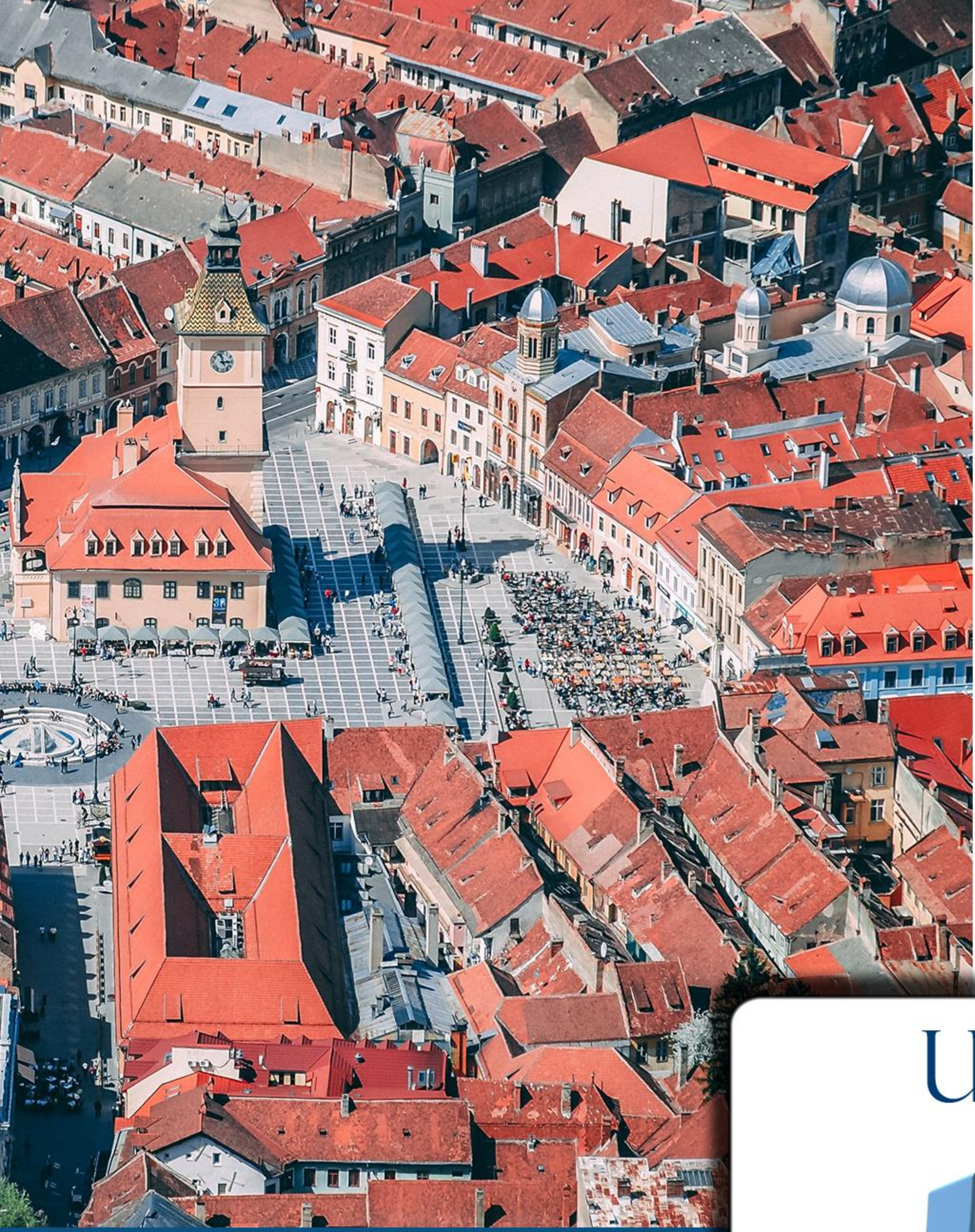


URBAN VISUALISATION & MANAGEMENT GMBH

UVM SYSTEMS

MANUAL

Basics

CityGRID® 2025

Content

1. Introduction ... 4

2. Products ... 5

3. Installation ... 6

3.1. System requirements ... 6

3.2. Installation of CityGRID® Software ... 7

3.2.1. Installation CityGRID® Database ... 7

3.2.2. Installation Program Modules (Administrator, Modeler, etc.) ... 7

3.3. Installation of Licences ... 7

3.3.1. On Licence Server ... 7

3.3.2. On the Client machine ... 8

3.4. Update of the Software ... 11

3.5. Database Update ... 11

4. Buildings semantics in CityGRID® ... 12

4.1. Model ... 12

4.2. Unit ... 12

4.3. Object ... 12

4.4. Element Complex ... 13

4.5. Element ... 13

5. Description of the Face Generation Algorithm ... 14

5.1. Face creation of CityGRID® Elements ... 14

5.1.1. Roof/Ceiling ... 14

5.1.2. Facade ... 14

5.1.3. Vertical Roof Faces ... 15

5.1.4. Floor ... 15

5.1.5. Roof Protrusion ... 15

5.1.6. General Element ... 15

5.1.7. Tubes (e.g. Subway Tunnels) ... 15

5.2. Overbuilt Objects ... 15

5.2.1. Cutout Object ... 16

5.2.2. Boolean Object ... 16

5.3. Solid ... 16

5.4. Attached Roof Details ... 17

5.5. Penetration Analysis ... 17

5.5.1. Face generation options of Penetration Analysis ... 18

6. Version History ... 21

6.1. Working versions ... 21

6.2. Historical versions ... 21

7. Special relations of polylines/polygons ... 23

7.1. Master/slave relations ... 23

7.2. 2D relations ... 23

8. Texture T ... 24

8.1. Guidelines for facade image capturing T ... 25

9. Glossary 26

10. Contact 28

Cover picture: Brasov, Romania, Dan Novac, Pixabay

❗ *<u>Note</u>: This manual was produced by automatic translation. Strange expressions and sentence structure may therefore occur. In case of doubt, the German manual is to be used as a reference.*

1. Introduction

Which city planner or architect has never wished to be able to visualize certain planning areas in 3D without having to generate expensive aerial photographs and photo assemblies?

With the City**GRID**® products you have the most efficient system for visualizing and modelling buildings in 3D objects with photo-realistic surface texture that currently exists on the market. The required areas can continuously be updated by city survey authorities and allow city planners to utilise a realistic model of the surrounding area of a planned object. They can show their clients or interested residents (stakeholders) easily, how a planned object in the context of its real environment looks like and which new views result from it.

The 3D city model provides for a reality-conformal, complete and geodetically accurate description of the surrounding urban area. With the registration, measurement, three-dimensional reconstruction, modelling, maintenance and visualization of buildings and infrastructure a new perspective is opened. Service orientation of the city administration is strengthened. Tasks can be fulfilled faster with improved quality on a more economical and citizen focused basis.

The City**GRID**® city model is

- reality-conformal
- actual
- easily generated
- extensible
- extremely versatile
- compatible with your corporate GIS

The City**GRID**® is a modern aid for planning, simulation and visualization in urban areas. With the use of 3D representation city-relevant data offered in an illustrative and intuitive manner, even for non-specialist users. Clarity of understanding of proposed schemes, via visualization of the information contributes crucially to the reduction in lead times of projects and improving decision making processes.

2. Products

The following products are part of the City**GRID**® family:

City**GRID**® **Manager**

Management system for maintaining building models of whole districts or cities based on a data base management system (Oracle or SQL Server). The data base serves as basis for working with City**GRID**® Modeler and Planner. City**GRID**® Manager stays in background and provides functionality for all other City**GRID**® Modules but owns no user interface on its own.

City**GRID**® **Administrator**

Utility program for data administration. CAD data can be converted in a standardized XML-format and imported into the data base. Furthermore, the data can be exported in various formats. Beneath administration of city model data special functions like automatic building texturing are hosted by this program.

City**GRID**® **Modeler**

Supports city planners or city surveyors with the production and maintenance of building models of whole districts or whole cities. The Modeler can be used in conjunction with data held within a data base (City**GRID**® Manager) or in files in a standardized XML format. Modeler is available in three editions: Inspector, Editor and Texture. Inspector includes the base functionality, whereas Editor enables the complete editing tool palette on City**GRID**®, including the face generation algorithm. Texture contains the tools for managing textures. Combine all modules for the full version of City**GRID**® Modeler.

City**GRID**® **Builder**

Enables planers to integrate their drafts into the existing city model to generate visualizations of different planning variants and to create scenes optimized for City**GRID**® Scout.

City**GRID**® **Scout**

Is an interactive 3d real time viewer that allows for Virtual Reality visualizations with highly realistic appearance of streets with buildings and vegetation. It allows for simulating the effect of the planned objects in the context of the real surrounding environment.

City**GRID**® **Solid**

Is a module for processing city model data for 3D printing. The areas to be printed are scaled down, cut to printing space, hollowed out and the surface mesh rendered waterproof. The result can be sent directly to a 3D printer.

3. Installation

3.1. System requirements

Recommended (optimal) hardware:

1. Processor (CPU)

   Intel Core i9-13900K/KF (13. generation) or AMD Ryzen 9 7950X3D

   These processors offer a high number of cores and threads, which is an advantage for multitasking and computing-intensive tasks such as rendering.

   (Minimum: i7/Ryzen 7)

2. Memory (RAM)

   64 GB DDR5-5600

   With 64 GB or more, you can easily run multiple applications simultaneously and work with complex scenes.

   (Minimum: 32GB)

3. Graphics card (GPU)

   NVIDIA GeForce RTX 4080 or AMD Radeon RX 7900 XTX

   A dedicated graphics card is essential for 3D visualization and rendering. Look for a card with plenty of memory (12 GB or more) and a high number of CUDA cores (for NVIDIA) or stream processors (for AMD).

   (Minimum: 8GB dedicated memory)

4. Hard drive (SSD)

   2 TB NVMe PCIe 4.0 SSD + 4 TB HDD

   A fast NVMe SSD ensures short loading times for programs and projects. An additional HDD with plenty of storage space is useful for archiving large files.

Required pre-installed software:

- **Operating system**: Windows 10 or higher /64-Bit
- **Database management systems:**
  - **Oracle** (Version 11g and above)
  - **Microsoft SQL Server** (Version 2017 and above, for database-based Administrator)
  - **PostgreSQL** (Version 15.3 and above)
- **Modeling and data processing software:**
  - Autodesk 3D Studio Max Version 2023 +; Basic program for Modeler and Builder plugins.
  - FME 2022 + as a basic program for the City**GRID**® FME Reader/Writer.

  This version was developed and tested on 3ds Max 2025.

❗ *Note: In case the City**GRID**® setup detects installations of 3D Studio Max and Max Design on the same machine, Max Design will be chosen as the host application for City**GRID**® Plugin.*

3.2. Installation of CityGRID® Software

3.2.1. Installation CityGRID® Database

Operating a City**GRID**® database requires utilization of a database management system from Oracle or Microsoft SQL Server. This DBMS must be provided and configured by the customer's IT support. In particular, the creation of a tablespace and the user with appropriate rights (read, write, schema update, import, export, creation of views, triggers, sequences, materialized views) is assumed. Creating the City**GRID**® schema itself is done by importing SQL files provided by UVM Systems.

3.2.2. Installation Program Modules (Administrator, Modeler, etc.)

The City**GRID**® modules are installed via the installer package CityGRID Software_en.exe. As part of the setup, the package automatically detects existing host programs (Autodesk 3D Studio Max and FME) and integrates City**GRID**® as a plugin in it. Furthermore, the setup ensures the installation of all necessary third-party modules that are needed to operate City**GRID**®. This applies in particular to the CodeMeter license software.

*Note: The City**GRID**® setup can also be installed as a silent installation via command line. To do this, start the installer package with the parameter "/ quiet". The parameter "/ help" calls the quick guide with all available installation options.*

*Note: Current and previous City**GRID**® versions can be obtained from www.uvmsystems.com.*

*Note: For a complete City**GRID**® setup, the program files must be copied to the installation directory (typically C:\Program Files\CityGRID) and to the Common Files folder. In addition, the setup writes entries to the Windows registry as well as the path variable. The setup should therefore be performed with a user who has the required rights.*

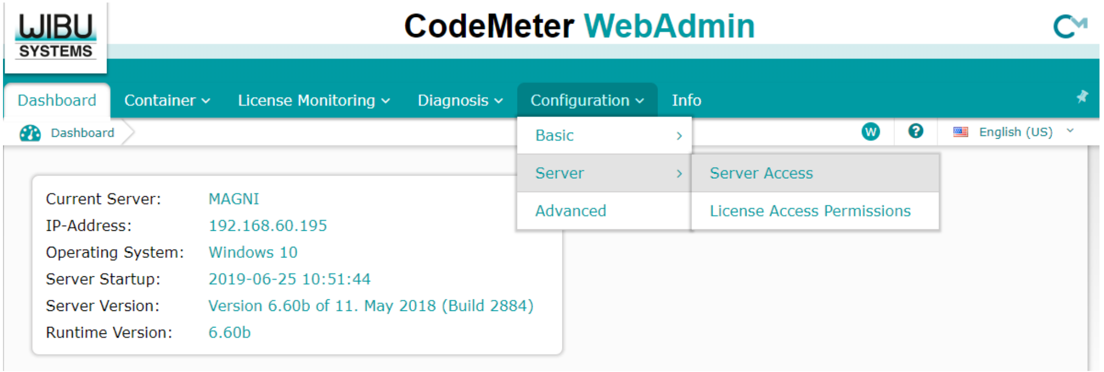
3.3. Installation of Licences

3.3.1. On Licence Server

In general, the license server does not simultaneously act as a client machine, but it is not prohibited to use a client machine as a license server.

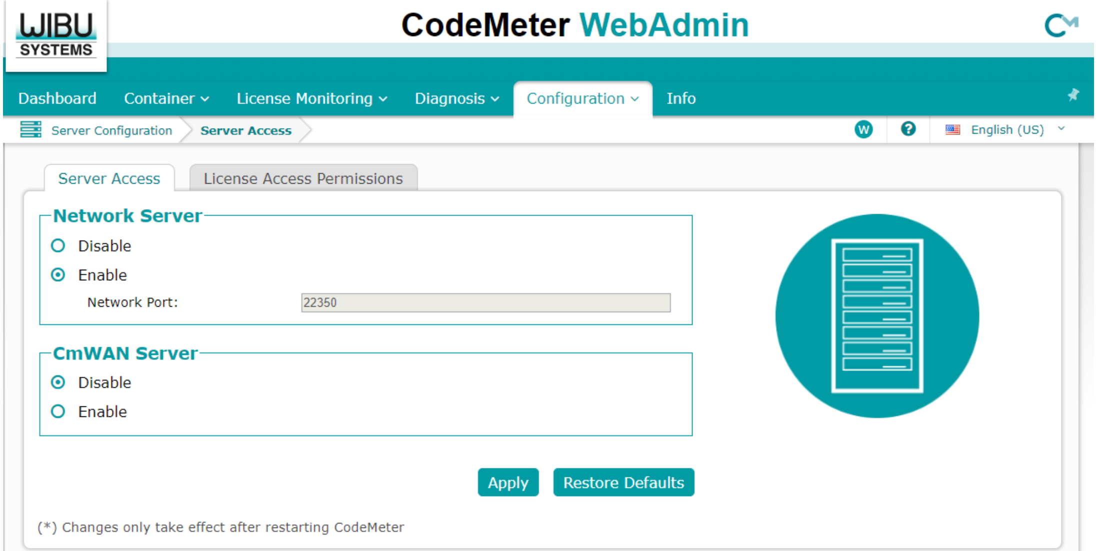
1. Install the CodeMeter server software on the license server. The instal pack can be obtained directly from CodeMeter, or the UVM Systems homepage www.uvmsystems.com. Follow the instructions in the installation package. The codemeter runtime is available for different operating systems, please select the appropriate installer package for your license server!
2. Insert the USB Licence Dongle into a free USB port.
3. After successful installation the CodeMeter symbol appears in the Windows system tray.

4. Right-click on the CodeMeter Symbol and choose **WebAdmin** in the context menu. The Standard Web Browser opens with the CodeMeter homepage.
5. Click on the tab **Configuration → Server → Server Access**.



Current Server: MAGNI
IP-Address: 192.168.60.195
Operating System: Windows 10
Server Startup: 2019-06-25 10:51:44
Server Version: Version 6.60b of 11. May 2018 (Build 2884)
Runtime Version: 6.60b

6. Set the radio button to Enable on *Network Server*. If required, an alternative port number should be set in the *Network Port* field if the proposed port is not suitable for client-server communication.



Network Server
○ Disable
● Enable
Network Port: 22350

CmWAN Server
● Disable
○ Enable

(*) Changes only take effect after restarting CodeMeter

7. Check if the *Network Port* can be reached by the client machines in your network, otherwise change to an available port. Contact your IT department in case communication can't be established.

Note: If a firewall is activated, the communication must be allowed via the specified port.

8. Reboot the licence server.

Note: The license server must be able to connect regularly to a CodeMeter Time Server in order to update the certified time. If the certified time is older than 7 days the license becomes invalid.

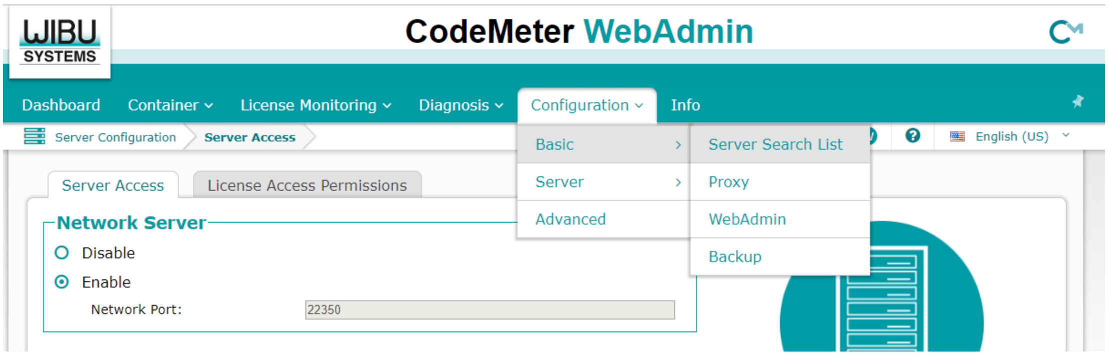
For more detailed information, please refer to the Online Help of CodeMeter, which can be accessed via WebAdmin. As a City**GRID**® customer, you can always contact the Codemeter support for any problems with City**GRID**® software licensing.

3.3.2. On the Client machine

1. Install the City**GRID**® software package of on the designated client machine. During install process the Client Licence Software CodeMeter by WIBU Systems will also be installed. After successful installation the CodeMeter symbol appears in the Windows system tray.

2. Right-click on the CodeMeter Symbol and choose WebAdmin in the context menu. The Standard Web Browser opens with the CodeMeter webpage.

3. Click on the tab **Configuration → Basic → Server Search List**.



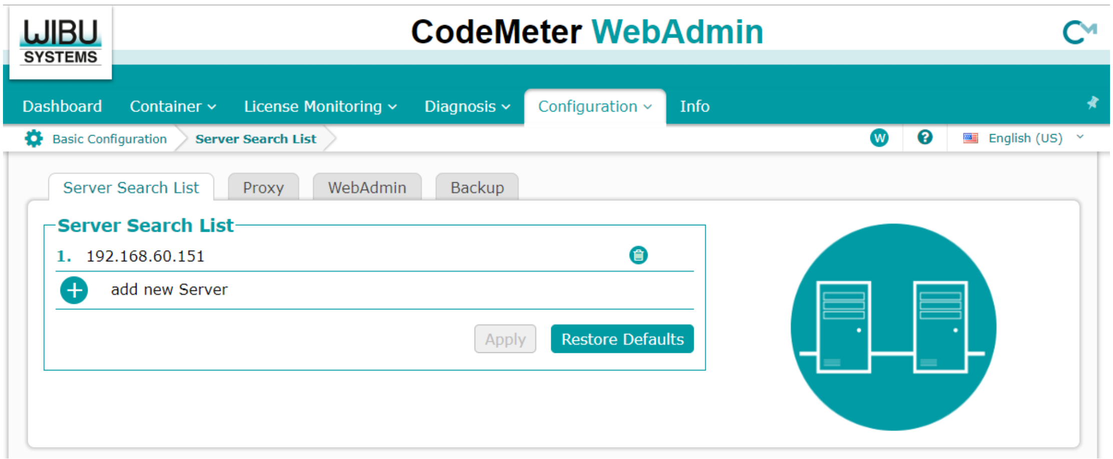
Network Server

○ Disable

● Enable

Network Port: 22350

4. Click add new Server and enter IP-address to the CodeMeter server



Server Search List

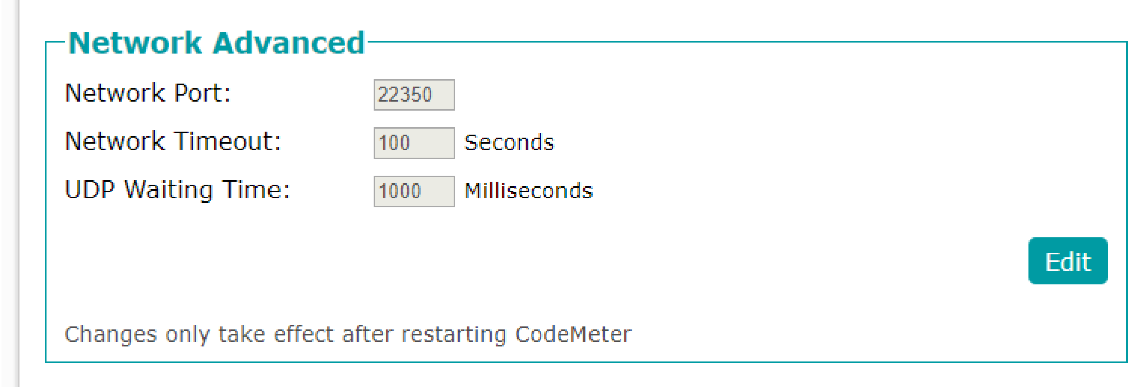
1. 192.168.60.151

add new Server

Note: By setting the server search list, CodeMeter is instructed to look for licenses only at the specified addresses. As a result, the access time can be significantly reduced, especially for large networks. If a valid license is not found at any of the specified addresses, CityGRID® will not start!

5. Click `Apply` to confirm updated Server List entries.

6. On tab *Extra* under **Configuration → Advanced**, check whether the port specified on the license server is specified.



Network Advanced

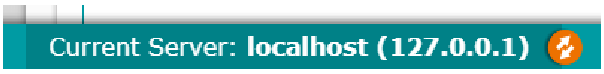
Network Port: 22350

Network Timeout: 100 Seconds

UDP Waiting Time: 1000 Milliseconds

Changes only take effect after restarting CodeMeter

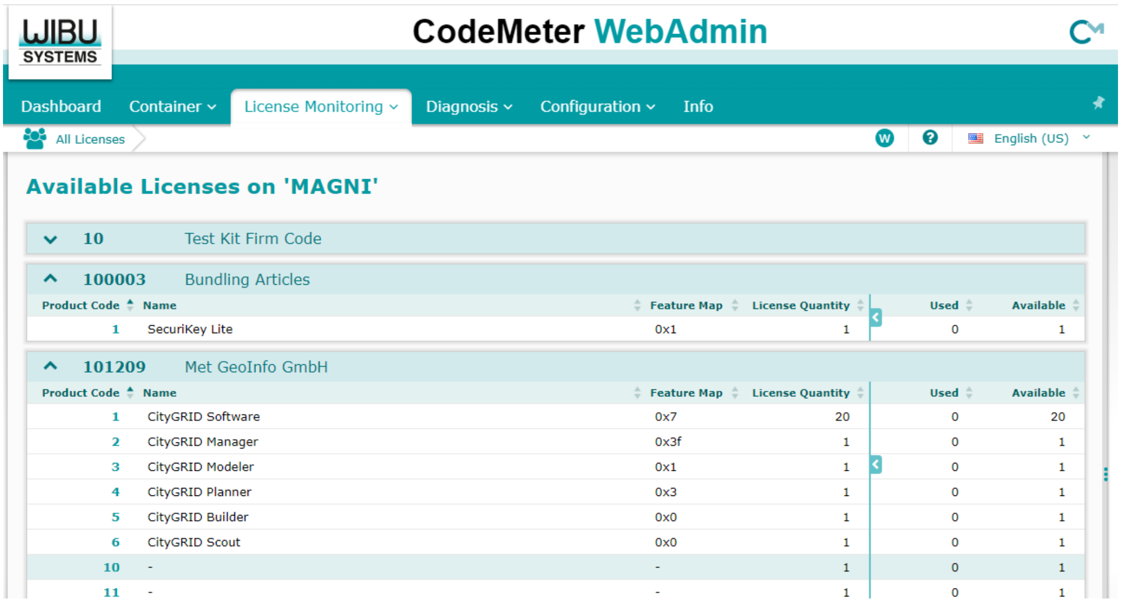
7. Switch to the license server by clicking the field *Current Server* in the bottom left WebAdmin window. Select the license server from the list



Current Server: **localhost (127.0.0.1)**

Note: If the license server does not appear in the specified list, remove the entry in the Server Search List and try again. The list contains entries only if CodeMeter licenses are found at the specified IP addresses!

8. Use **License Monitoring → All licenses** to check the available licenses on the license server. The displayed licenses vary depending on the licensed City**GRID**® modules.



Available Licenses on 'MAGNI'

10 Test Kit Firm Code

100003 Bundling Articles

Product Code	Name	Feature Map	License Quantity	Used	Available
1	SecuriKey Lite	0x1	1	0	1

101209 Met GeoInfo GmbH

Product Code	Name	Feature Map	License Quantity	Used	Available
1	CityGRID Software	0x7	20	0	20
2	CityGRID Manager	0x3f	1	0	1
3	CityGRID Modeler	0x1	1	0	1
4	CityGRID Planner	0x3	1	0	1
5	CityGRID Builder	0x0	1	0	1
6	CityGRID Scout	0x0	1	0	1
10	-	-	1	0	1
11	-	-	1	0	1

9. Close the Web Browser.

10. Start City**GRID**® software.

11. Repeat these steps for each client machine, on which you want to install City**GRID**®.

3.4. Update of the Software

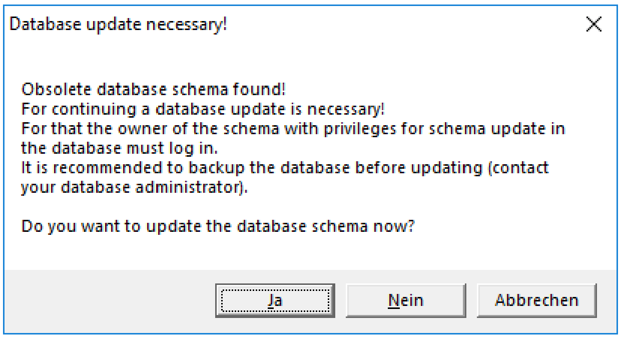
UVM Systems will provide software updates on a regular basis. For installing a new version of the software, it is typically not necessary to uninstall older versions. It is sufficient to run the `Setup.exe`.

Afterwards it may be necessary to update the database. (cf. 3.5)

3.5. Database Update

For the City**GRID**® Manager with database connection:

To support some of the new functions introduced in this version the database scheme needs to be extended. This is done automatically with the new version of the City**GRID**® Administrator or City**GRID**® Modeler. If it is necessary, the following message appears when connecting with the database:



Database update necessary!

Obsolete database schema found!
For continuing a database update is necessary!
For that the owner of the schema with privileges for schema update in the database must log in.
It is recommended to backup the database before updating (contact your database administrator).

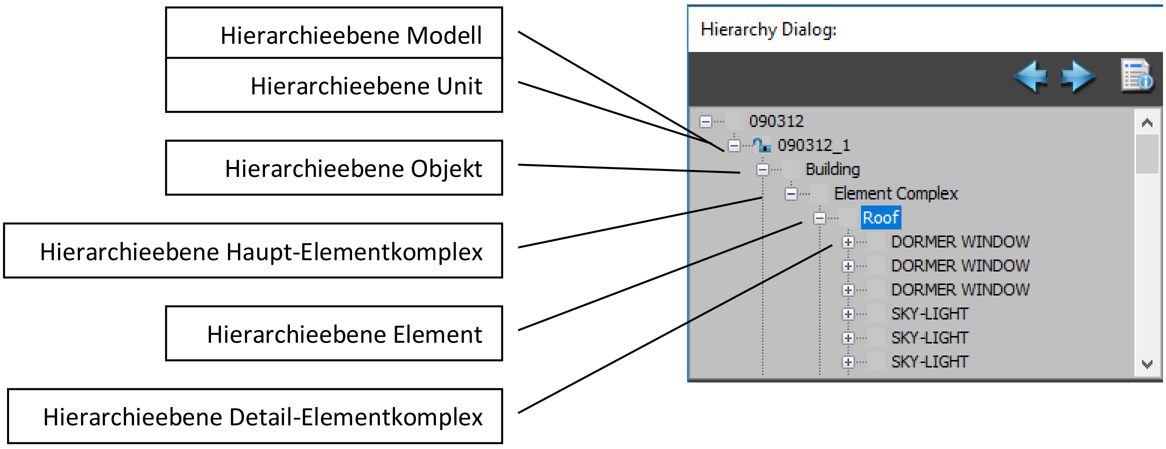
Do you want to update the database schema now?

If you want to perform the update, the database login window appears again. Thus, you have the possibility to login with another database user, because the update usually needs privileges such as adding columns to tables, adding tables or procedures, and sometimes deleting tables or procedures.

❗ *Note: We always suggest making a backup of the database before undertaking updates of this nature.*

4. Buildings semantics in CityGRID®

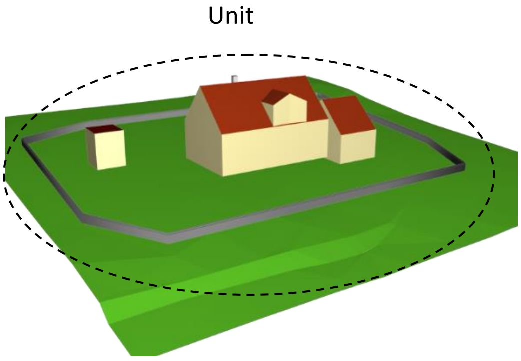
In City**GRID**® buildings are grouped logically to complexes. The multi-level hierarchy of building complexes will be described in this section. These levels of hierarchy are shown in the window of loaded Units (City**GRID**® Modeler).



4.1. Model

A Model is a logical group of any number of Units. The other way around, each Unit can belong to any number of Models.

Models facilitate working with the database version of the Modeler as each ensemble of buildings can easily be grouped to a Model and Models can easily be deleted again. Deleting a Model does not have any effect on the Units, they will not be deleted.



4.2. Unit

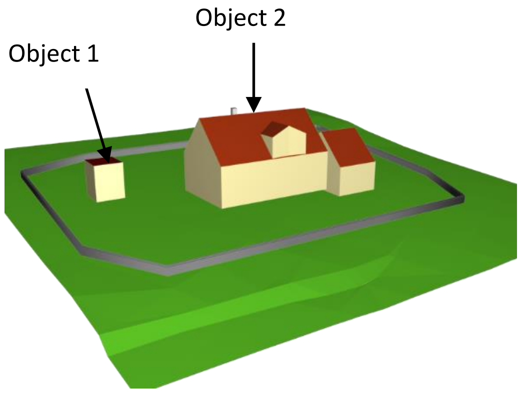
A Unit is a group of buildings from an administrative point of view. It consists of all buildings belonging for example to one property (e.g. a residential building together with its garage, etc). A Unit consists of any number of, but at least one Object.

4.3. Object

An Object is a single building. Each Object is part of exactly one Unit. Each Object consists of any number of, but at least one Element Complex. Objects of a unit are in general geometrically separated.

The Object has pre-defined object classes (e. g. Building, Bridge, Underground, Cutout, Boolean etc)

Special Objects are Cut-out Objects which are used for modeling overbuilt areas (passageways etc.). They can be used only in combination with other objects.



4.4. Element Complex

An Element Complex is a group of Elements which all together describe a solid. Each Element Complex is part of exactly one Object.

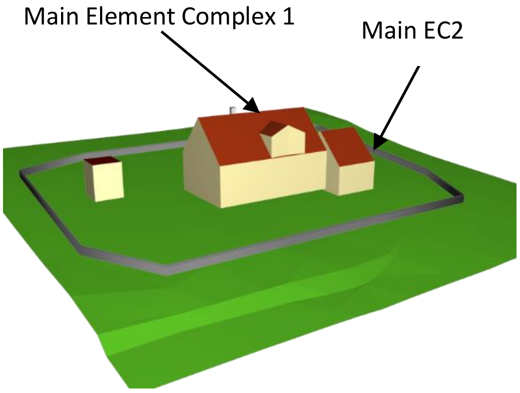
An Element Complex can either be the main solid of an Object (Main Element Complex) or an attachment to it (Detail Element Complex, e.g. dormer windows etc).

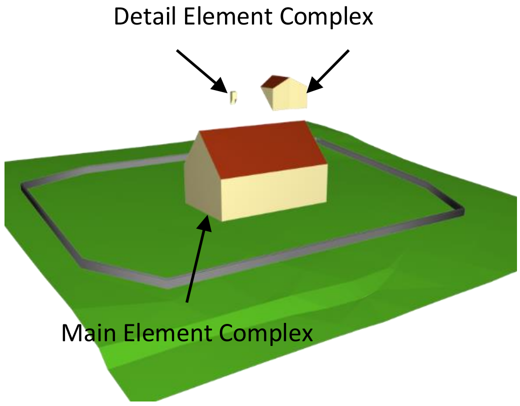
An Object has at least one Main Element Complex.

A Detail Element Complex is often termed as Child Element Complex; its corresponding Main Element Complex as Parent Element Complex.

Main Element Complexes belong to the Level of Detail 2, whereas their children usually belong to Level of Detail 3.

Each Element Complex can have a name of choice (e.g. dormer, chimney, etc....)





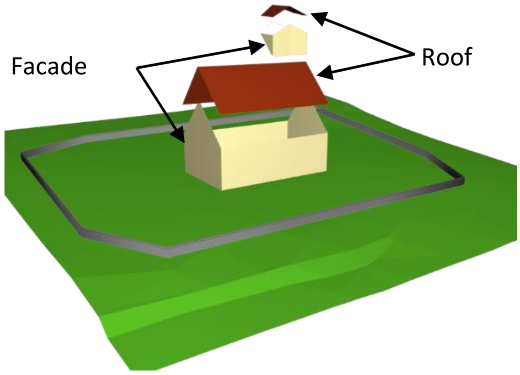
4.5. Element

An Element is a part of an Element Complex and describes the surface of a part of the solid (e.g. the roof, the facade, the protrusion, or the floor). Each Element has its own Face Generation Algorithm which will be described in section 5.

Roof and/or Ceiling elements may have Detail Element Complexes. Every Element Complex has at least one element and may have the following element combinations (from up to down):

- 0 or 1 roof- or ceiling-element,
- any facade-elements,
- 0 or 1 floor-element.
- 0 or 1 vertical roof element
- Between facade- and roof/ceiling-elements protrusion elements may occur.

The Element carries the geometric information.



5. Description of the Face Generation Algorithm

Modelling and maintaining building models in the system City**GRID**® can be done efficiently due to the premise to derive the surface („boundary representation“) of the models automatically from the edge model („wire frame representation“). The main edges defining the shape must be modelled; additional edges are created automatically during the face generation process.

The main task of the operator is to edit the line model until the face generation algorithm creates a correct building model. The line model is stored in the data base and can be updated in a revision cycle. This concept allows for maintaining a city model in an efficient way.

For generating the faces a 3D triangulation algorithm specially optimized for building data is used. Different face generation rules are applied to the roof, the facade, the protrusion, to detail elements and to overbuilt objects of a building. Generally, all lines inside the outline (e.g. eave line) are used. Lines in auxiliary and user defined layers are ignored.

The creation of an area is mainly based on a vertical extrusion direction. For elements, which possess a local “vertical” direction deviating from a global vertical direction, the extrusion direction must be set, or will be extracted automatically from attached detail element complexes.

In the course of a face generation process several actions of consistency check and possible corrections will be carried out.

- Additional knots are inserted if necessary (e.g. for lines within an element, which intersect (in 2D or 3D).
- The topology of breaking edges is examined.
- Missing Master/Slave relations are set.
- Missing elements are generated (e.g. roof protrusions, but also missing facades).

The user will be notified by a warning for most of the correctional measures.

In the following sections the features of these algorithms are described.

5.1. Face creation of CityGRID® Elements

5.1.1. Roof/Ceiling

The Element “roof” is generated by triangulation inside the *Outer Polygon:Eave* lines where *Ridge lines* and further *Roof lines* are used as constraint edges. The *Outer Polygon:Eave* must be a closed polygon (sum of all line segments). *Inner eave (Courtyard)* lines are introduced as cut-out areas of the triangulation. *Break lines* allow for modelling vertical faces on the roof.

Regulations for the creation of faces also allow inverted areas where the face normal is directed downwards.

5.1.2. Facade

The faces of the Element “facade” of a building are generated by one of the following two methods:

- *Extrusion*: if there is no polygon in the layer *Lower Facade Polygon*, by extrusion of the upper edge of the facade vertically downwards and
  a) *intersection with the terrain model*. The derived lower edges of the walls are modelled horizontally in the elevation of the lowest intersection point with the terrain and thus do not follow the terrain exactly. In that way the facade faces are modelled by rectangles which make texturing easier. Facades are generated for courtyards in the same way.

  If the building does not lie inside the domain of the terrain model, the facades are generated with a height of 50 m. We strongly recommend creating a precise terrain model before starting modelling with City**GRID**®.

b) *intersection with the parent element*: the derived lower edges of the facade follows the exact intersection with the parent element.

- *Triangulation*: if there is a closed polygon in the layer *Lower Facade Polygon*:
  by triangulation of a tube between the *Upper* and the *Lower Facade Polygon*.

5.1.3. Vertical Roof Faces

Vertical faces of the roof that are typically formed by a break line system are transferred from the element Roof to the element Vertical Roof Faces. This element consists only of the vertical faces in the corresponding face layer but has line information stored and can't get assigned one interactively. Due to data technical restriction the layer Outer Polygon is present but contains no line information itself.

By default, all vertical roof surfaces are displayed in the color of the façade, creating a pleasing impression of the building. Vertical roof surfaces are excluded from automatic roof texturing and can be textured as standalone elements.

5.1.4. Floor

Buildings above ground usually do not have an Element of type "floor". However, if there is given a floor, the faces are generated similarly to roofs, where the normal of the triangles are inverted. For Element Complexes consisting of facade and floor, the *Lower Facade Polygon* of the Element Facade and the *Outer Floor Boundary* of the Element Floor have to be identical and be in a Master/Slave relation (cf. 6). Element complexes of sub terrestrial objects should be modelled with a Floor Element.

5.1.5. Roof Protrusion

If the upper edge of the wall and the eave line of the roof are not identical, protrusion faces are generated between these polygons. The eave line serves as bounding polygon and the wall polygon defines a cut-out area. Normal vectors of faces are oriented downwards.

5.1.6. General Element

3D Objects without any further information are stored as "general Element". They are created for instance by the BLOB Conversion Manager or by the City**GRID**® Writer in FME. Every General Element can be changed into any other available City**GRID**® Element by setting the appropriate type in Element Property window (cf. Manual City**GRID**® Modeler).

5.1.7. Tubes (e.g. Subway Tunnels)

Tubes are modelled in the same Element structure even if the names of the Elements are not appropriate in that case. The outer wall of a tunnel is an Element of type "Facade", the covering faces (if available) are of type "Roof" and "Floor".

5.2. Overbuilt Objects

For modelling of subtraction bodies two *City**GRID**®* object classes are available, Cutout and Boolean Object. Both objects can be utilized like normal building objects, and therefore contain normal roof and facade elements. Only by assignment of corresponding object class triangulation algorithm interprets constructed body as subtraction object and adjust buildings faces accordingly.

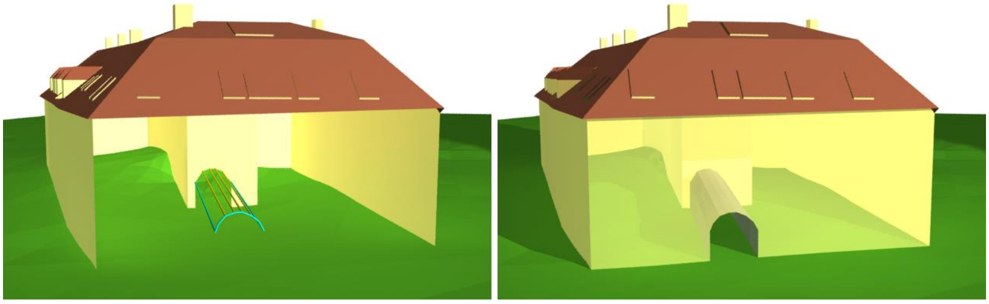
In order to trigger subtraction action an intersection of eave line from subtraction object to upper facade polygon from as build object has to be established. Otherwise subtraction body remains without effect.

Note: Cutout objects are faster during triangulation but not as versatile like Boolean objects. Application of Boolean objects is therefore recommended.

5.2.1. Cutout Object

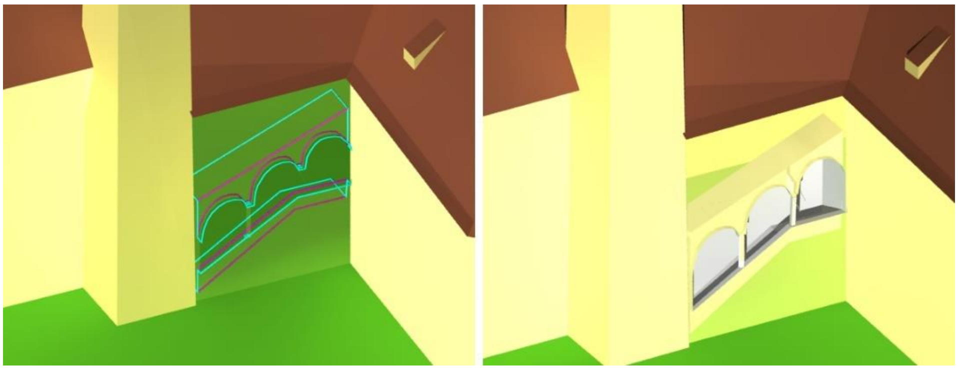
If a Cut-out Object is placed under an Object, the respective facades are extruded only to that Cut-out Object. The roof of the Cut-out Object is generated with inverted normal vectors. The same is true for those facades of the Cut-out Objects which are not below the facades of main Objects. In that way entrances remain open.

Cutout objects affects only extruded facade faces and therefore are always triangulated until intersection with terrain model. In case the facade element contains lower facade edges with attached floor element, Cutout objects are not applicable. Cutout objects can't have floor elements on their own as well.



5.2.2. Boolean Object

Boolean objects act the same way like cutout objects but have the ability to contain floor elements on their own and affect every *City**GRID***® Elements from as build Objects, regardless if extruded or triangulated. Floor elements have an effect on face generation of every object of containing Unit. As a consequence, Boolean objects can be used for cutting single or multi storied sections from buildings. All faces from Boolean objects always will have inverted face normal.



5.3. Solid

At the conversion of data with data type Solid using the BLOB Conversion Manager (cf. Manual City**GRID**® Modeler) or FME City**GRID**® Writer, the face net will be transferred into City**GRID**® data structure and saved as Element type Solid. Typically, solids are waterproof (i.e. all side closed) 3D objects.

5.4. Attached Roof Details

Roof detail elements can be converted into a thin line structure with corresponding face extrusion directions. In that case they do not have closed polygons and usually a secondary (e.g. horizontal) extrusion direction. Thus, the intersection with the parent roof will be created correctly.

5.5. Penetration Analysis

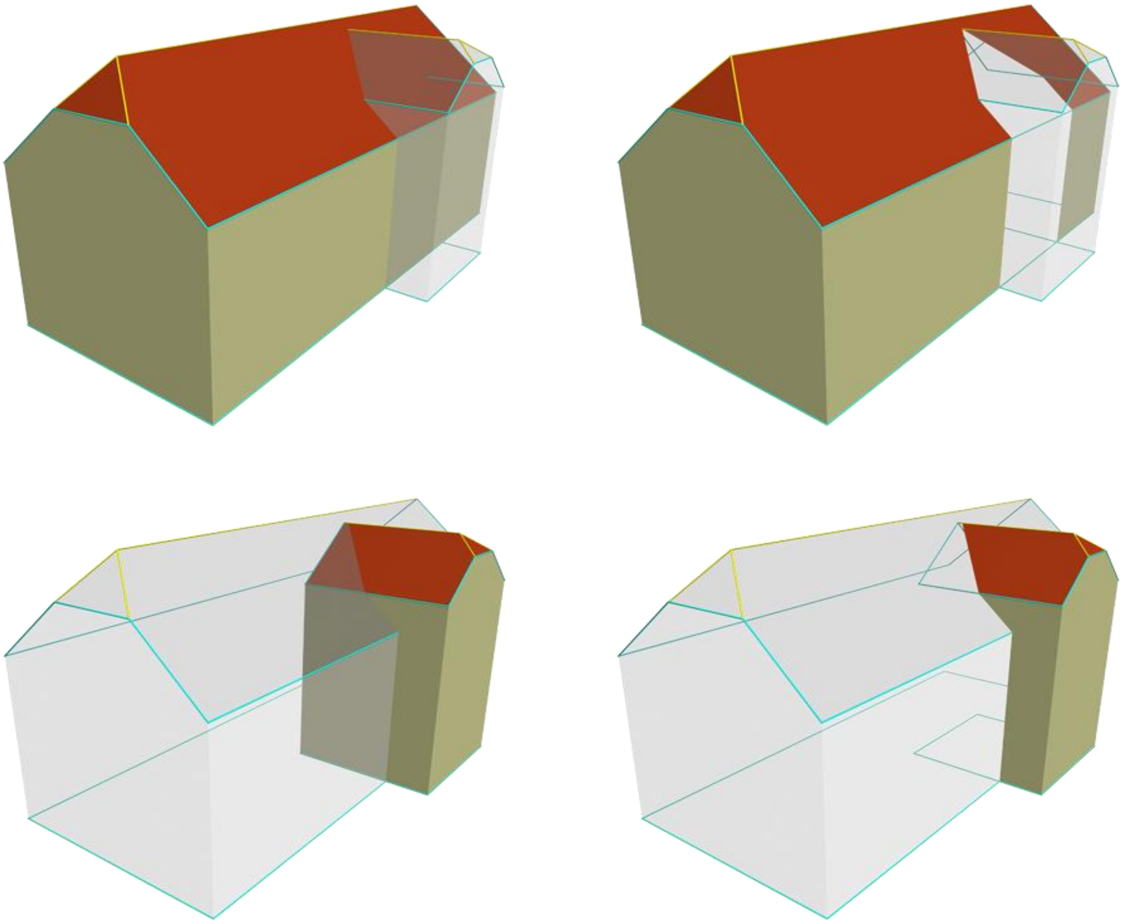
City**GRID**® derives faces by default from the line structure of the building models (see. Manual City**GRID**® Basics). Depending on the accuracy of restitution buildings can thus overlap and form surfaces inside of another structural body. To remove these commonly unwanted faces, Penetration Analysis is available. It is an extension of the triangulation algorithm, for detection and removal of interior face sections. Penetration analysis affects element complexes, the face generation options of surface manipulation (see. Manual City**GRID**® Basics) can be adjusted individually.

*Note: Penetration Analysis acts purely on the surface network of buildings. The line structure is not changed by this method of surface creation. The penetration analysis can be applied to triangulated City**GRID**® buildings as well as to frozen face nets (BLOBs).*

If the penetration analysis is deactivated in triangulated buildings, the original surface mesh of the structural lines of the building arises again. With frozen area networks, however, the penetration Analysis is irreversible.

Example:

The following figures show the changes in the surface network by the penetration analysis. The left column represents the element complexes of a Unit without, the right column with enabled penetration analysis. Since this is an area operation, the line structure remains unchanged. Therefore, the initial state of triangulation can be restored by disabling penetration analysis.



Penetration analysis intersects all faces at Element Complex level of a Unit in 3D in order to find any existing cutting edge. The cutting edges are then incorporated into triangulation as preparation of removing of unwanted sections. Finally, the investigation is carried out for each surface triangle, whether it is inside or outside of another building (part). Depending on the activated option of penetration analysis this analysis results in the removal or retention of the triangular faces.

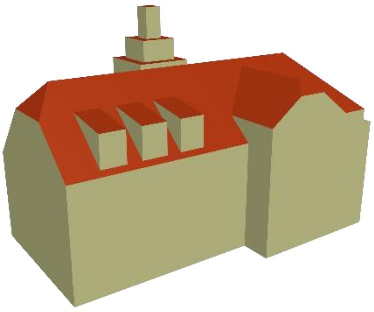
For the correct functioning of the penetration analysis, it is necessary that the surfaces of the examined building complexes (Element Complexes) form a body, otherwise the analysis can't perform within/outside investigation. If no floor surfaces are present, the penetration is analysis assumes a horizontally mounted floor and temporarily calculates the missing faces from lower facade edges at run time.

5.5.1. Face generation options of Penetration Analysis

Note: The figures below show the interior view of the adjacent unit and the impact of each option the penetration analysis has on the surface network.

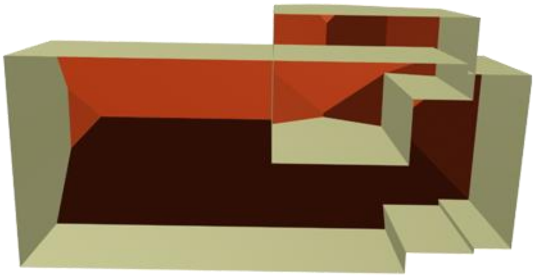
To increase clarity some images don't show the complete network area or represents combinations of several options of penetration analysis.

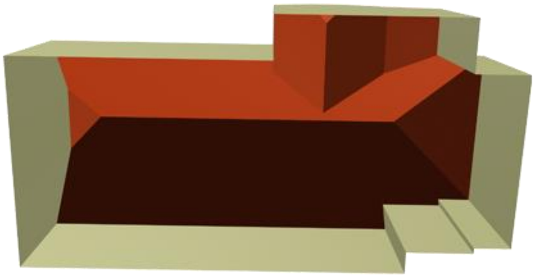
Left the unit without and right with penetration analysis can be seen.



- Main Element Complex with Main Element Complex:

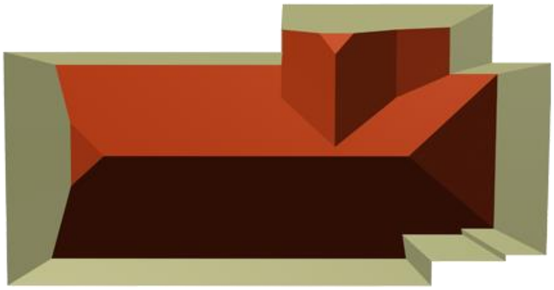
  All main Element Complexes of a Unit are intersected with each other. If faces of an element complex are detected inside another element complex, these are eliminated. Faces of lower Element Complexes are removed first. If Element Complexes have the same height, it is randomly decided by which faces are removed.

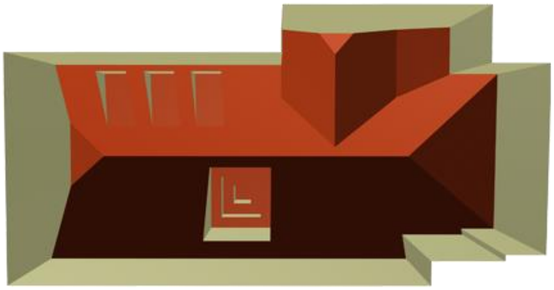




- Main Element Complex with Detail Element Complex (all LoDs):

  All main Element Complexes are intersected with all detail Element Complexes of the same Object. Faces of main Element Complexes underneath detail Element Complexes are thrown away.

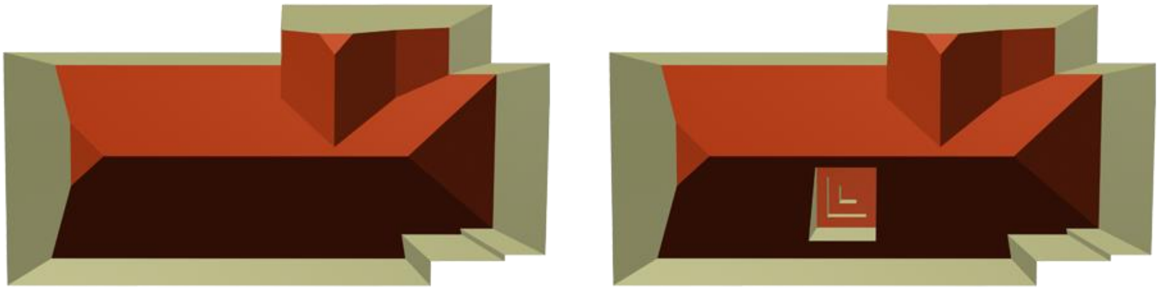




Note: Automatically derived LoD 2 representations of the Unit in database close the exempted holes in the main Element Complex roof faces when associated LoD 3 detail Element Complexes don't exist.

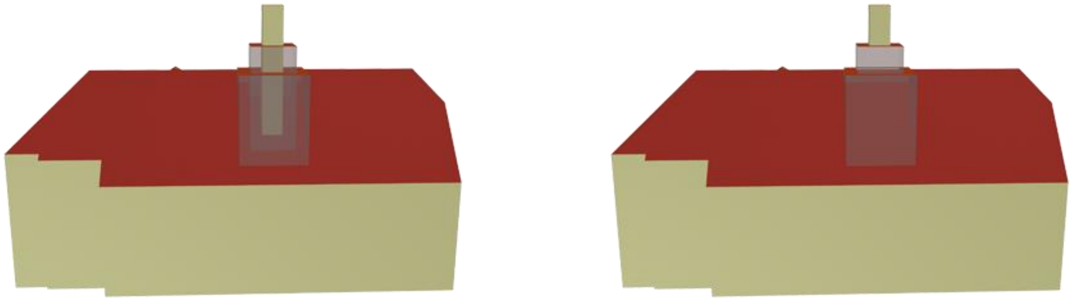
- Main Element Complex with Detail Element Complex (LoD 2 only):

Works similar to the previously described option, with the difference that only detail element complexes in LoD2 are considered for cropping of roof faces of main Element Complexes. Areas underneath LoD3 detail Element Complexes remain unchanged.



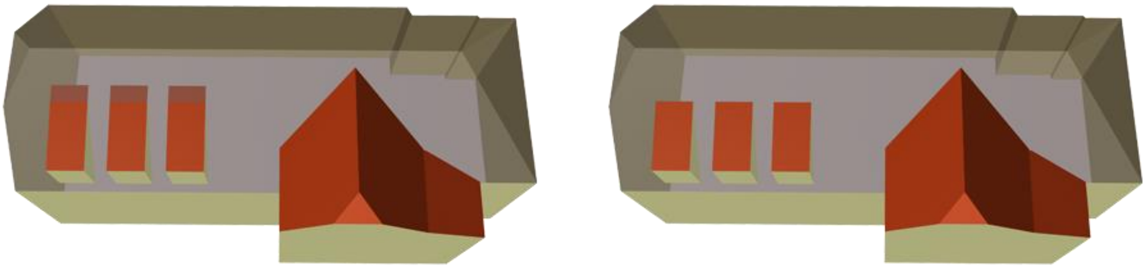
- Detail Element Complex with Detail Element Complex:

  Examines detail Element Complexes of a main Element Complex for mutual intersection situations and adjusts them.



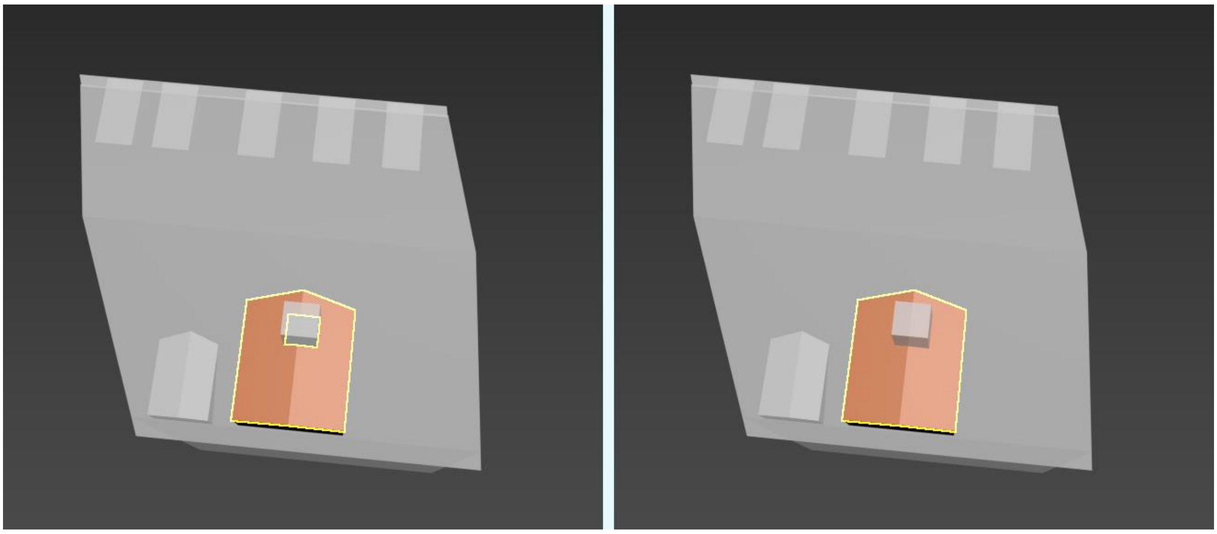
- Detail Element Complex with Main Element Complex

  All detail Element Complexes are intersected with all the main Element Complexes within the same Object. Penetrating surfaces of detail Element Complexes in the main Element Complex, are removed by penetration analysis. This ensures a precise blend of detail Element Complex surfaces with the main Element Complex roof.



- Detail Element Complex with Detail Element Complex (LoD <=)

  If an LoD3 detail touches an LoD2 detail, this option eliminates the affected faces of the LoD3 detail, while the faces of the LoD2 detail are retained. Details of the same LoD eliminate each other's faces.

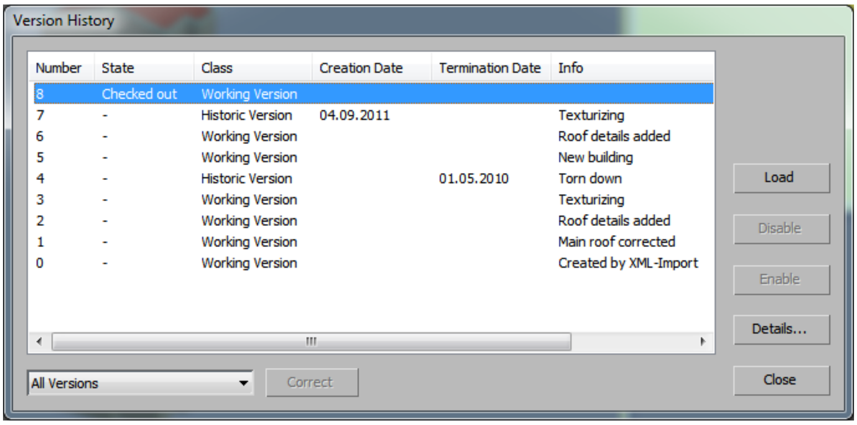


The penetration analysis is (de-)activated by setting the appropriate options in the modules City**GRID**® Administrator, City**GRID**® Modeler or City**GRID**® FME Writer.

6. Version History

Every unit can have numerous version in database. Each version contains following attributes:

- Number: Increasing number for every new version of unit in database
- Status:
  - Checked in: default state of version (indicated by „-„ in version history window).
  - Checked out: indicates versions prepared for editing
  - Disabled: declared version as invalid and prevents unit to show up at selection polygons.
  - Torn down: Version without geometry. Earlier unit versions remain in database and can be addressed if applicable.
- Class:
  - Working version
  - Historical version
- Creation- and Termination date (historical versions only)
- Info (any comment)



Number	State	Class	Creation Date	Termination Date	Info
8	Checked out	Working Version			
7	-	Historic Version	04.09.2011		Texturizing
6	-	Working Version			Roof details added
5	-	Working Version			New building
4	-	Historic Version		01.05.2010	Torn down
3	-	Working Version			Texturizing
2	-	Working Version			Roof details added
1	-	Working Version			Main roof corrected
0	-	Working Version			Created by XML-Import

6.1. Working versions

New versions are always working version at the beginning. Only such versions can be checked out for editing (last version only). During modelling various working versions might be created for backup purposes (main roof modelled, roof details attached, texture applied etc.)

6.2. Historical versions

Whenever modeling on a building comes to an end, the most actual version can be turned over from working version to a historical version. These versions are assigned a date or creation and termination. Doing so a lifetime of a building can be set.

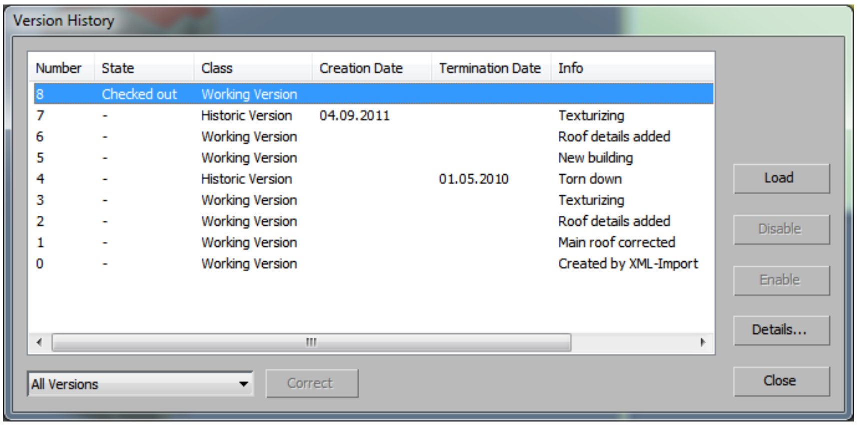
Undefined creation date is interpreted as “always existing”

Undefined termination date is interpreted as “will ever exist”

All historic unit versions should exist in a correct timeline, which means, that no version has a creation date older than a termination date of a previous version.

CityGRID® modules offer possibility for unit loading and exporting at dedicated dates, depending on previously created historic versions.

__Example__:



Number	State	Class	Creation Date	Termination Date	Info
8	Checked out	Working Version			
7	-	Historic Version	04.09.2011		Texturizing
6	-	Working Version			Roof details added
5	-	Working Version			New building
4	-	Historic Version		01.05.2010	Torn down
3	-	Working Version			Texturizing
2	-	Working Version			Roof details added
1	-	Working Version			Main roof corrected
0	-	Working Version			Created by XML-Import

Figure above shows example of a units historic versions. Versions are listed according to their creation date. Versions 0-2 contain modelling steps and are therefore indicated as working versions. On 1th of May 2010 building was torn down. Version 3 contains complete modelling of building at that date with unknown creation but set termination date as historic version.

Following reconstruction at same parcel (same unitID) again contains some working version that were created during modelling for backup purposes. Version 6 finally contains complete new building structure and has creation date 4.9.2011 set.

Version 7 is a checked out working version

*Whereas historic versions are only valid or a given timeframe, City**GRID**® modules offer functionality to question units at a specific date. Eg. Questioning database for date 1.1.2000 will return version 3 to the user.*

7. Special relations of polylines/polygons

The special relations between polylines facilitate the modelling of lines by following preconditions among them. On the one hand the surrounding polygon of adjacent elements has to be identical (3D-identical), so that no hole originates between elements (master/slave relations). On the other hand, upper breaking edges and lower breaking edges, which describe a vertical break-off on the roof, have to be identical also concerning position (2D).

7.1. Master/slave relations

Closed *Outer Polygons* of elements (cf. 4.5) can be in master/slave relation with Elements adjacent to polygons. Therefore, it can be guaranteed that when creating a face no hole between the faces of the Elements originates.

Example: The Outer Polygon: Eave line of a roof element and the Upper Facade Polygon are in a master/slave relation. If you modify the geometry of the Eave, also the geometry of Upper Facade Polygon changes.

The direction of the relation is peripheral, i.e. it is also allowed to edit the slave. By doing so its master (and consequently other slaves of the master) also changes.

The following line types can be in a M/S relation:

- Outer Boundaries (Eave line, Upper Facade Polygon, Outer Floor Polygon)
- Inner Boundaries (Inner Eave (Courtyard), Upper Facade Polygon, Cut-out Polygon Floor),
- Lower Facade Polygon

Note: missing master/slave relations are often set automatically at the process of triangulation.

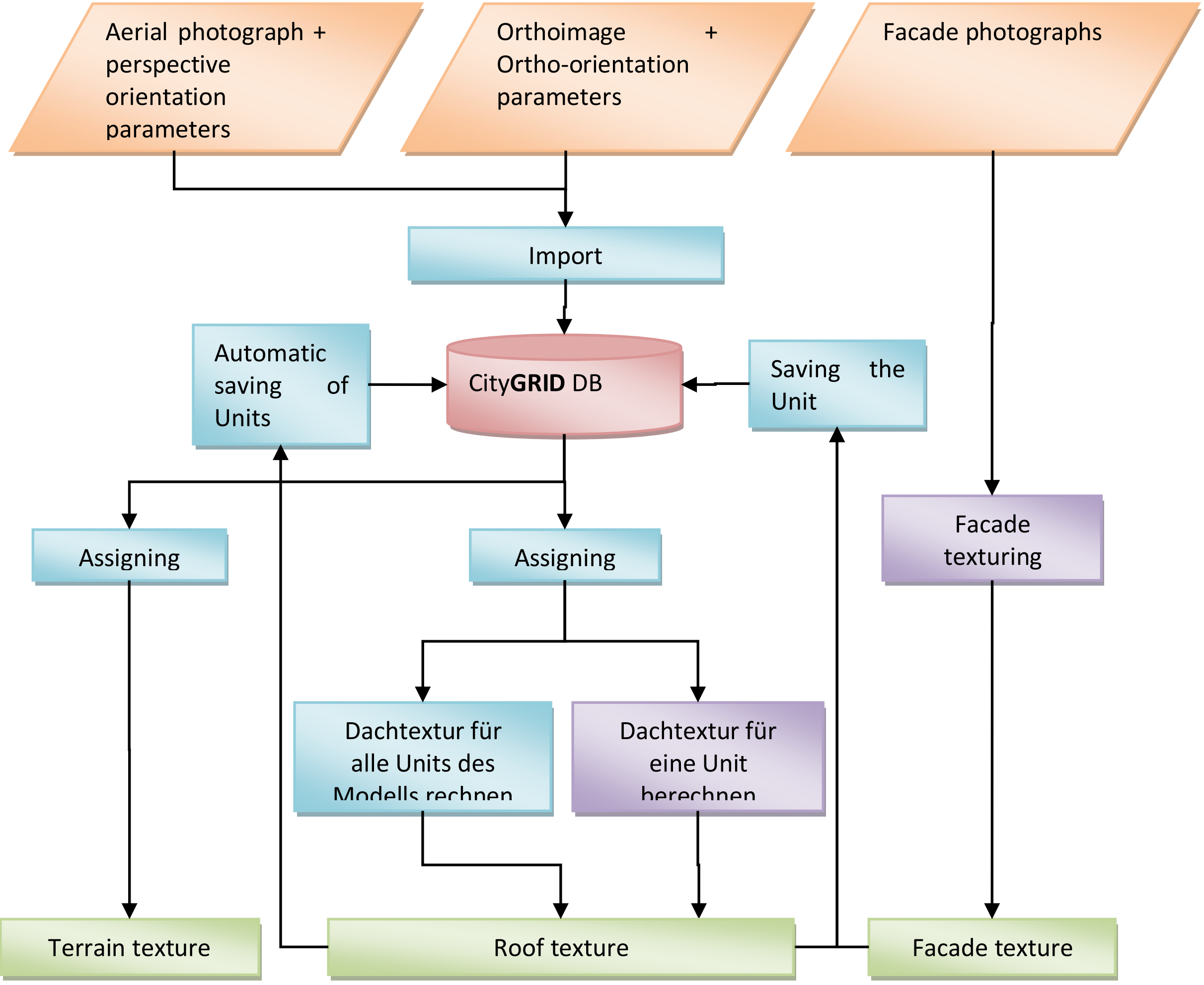
Special Case - protrusion: If the Outer Polygon: Eave of a roof and the Upper Facade Polygon (partially) do not run identically and a M/S relation is introduced between them, in the process of face generation a new Element for the protrusion is created automatically.

7.2. 2D relations

Between breaking edges which describe a vertical break-off on the roof, there are 2D relations. The Upper Break Edge and the Lower Break Edge are therefore always kept at the same 2D position. For a correct triangulation result all knots of breaking edges must correspond to each other in a 2D context.

8. Texture

Photo realistic texture can be applied to all building faces as well as to the terrain models. The following figure shows the workflow for texturing terrain, roofs and facades:


Aerial photograph + perspective orientation parameters
Orthoimage + Ortho-orientation parameters
Facade photographs
Import
Automatic saving of Units
CityGRID DB
Saving the Unit
Assigning
Assigning
Facade texturing
Dachtextur für alle Units des Modells rechnen
Dachtextur für eine Unit berechnen
Terrain texture
Roof texture
Facade texture

Roof and terrain are textured automatically, if a texture image together with its correct orientation parameters was imported into the data source and is assigned to the Model. The facade texture is assigned interactively or fully automatic from oriented images.

Terrain texture is always calculated online, whereas roof texture and facade texture is calculated for each Unit and saved with the Unit.

Roof texture is derived automatically and can be calculated for each Unit separately, where the texture coordinates can be revised and corrected if the orientation parameters and/or the building model are not accurate enough) or for all Units of a Model in one step.

Orthophotos of the facades are generated by simple affine transformation in the texture tool. The photos are loaded, assigned to the respective facade of the Unit and saved with the Unit in the data source.

Note: Changing the roof texture image of a Model does not immediately change the roof texture of a Unit. Only when the roof textures of the Units are recalculated, the new image will be used. But, a new terrain texture image is directly used.

City**GRID**® offers two possibilities for the process of texturizing:

- semi-automatically via oriented image files
- interactively via freely-taken picture

The (semi-)automatic process of texturizing requires the import of oriented image files into the data source including additional assignment to the working model; the texturizing process can run automatically subsequently for all units of a model or only for parts of the same.

The process of interactive texturizing takes place on the element in City**GRID**®, unit per unit. Carrier of the texture information is the face and not the structure line of a building like at the modelling process. It is not of relevance which element the face belongs to so that every face in City**GRID**® can also carry a proper texture.

8.1. Guidelines for facade image capturing T

When photographing facades for the texturing the following guidelines should be considered:

- A facade should be photographed by as few photos as possible. If for some reasons it is not possible to go away far enough in order to take a frontal photograph of the facade while still covering the whole facade, we recommend to take a diagonal photo.
- The more detailed the facade is the more frontally the photo should be taken. Protrusions and setbacks, which are not modelled, cannot be corrected in the orthophoto and remain distorted in the texture image.
- If possible, photos should be made from elevated points of view. By doing so, parking cars, bushes and parts of the street furniture can be avoided in the images. In addition displacement effects are smaller.
- The transition of one in the next photo is possible only if the facade's upper edge has a vertex at that position. A vertex is given at positions with slight corners (or with modelled protrusions/setbacks).

The last point causes sometimes problems. If a long facade does not show such structures, you can proceed as follows. The condition for it is

- that there is a (diagonal) photo, which covers the entire facade (even if it has already too poor quality for texture at the end of the facade). This image is only used as an overview image for the breaking up of facades.
- that there are several photos for sections
- and that there are elements on the facade, for which the mapping of the detail photos to the overview photo is possible.

9. Glossary

Bindings	Coupling of vertices of different → layers. Example: If the binding between roof lines and ridge lines is active, moving the ridge line causes those vertices of the roof lines which lie on the ridge to be moved, too, so that they need not be corrected separately.
Break Lines	delimit vertical roof faces at the upper and lower edge.
Check-in	an earlier → checked out → Unit will be changed again to a stable version in the → database
Check-out	Creating a new version of a → Unit in the → database for editing. Changes may be saved at any time. To accept changes, the Unit is → checked in.
Complex	General term for a part of a building (building complex), i.e. either an → Element Complex or an → Object or an entire → Unit
Database	The City**GRID**® Modeler uses an Relational Database Management System (Oracle or SQL Server databases are supported at version 3.4.2) for the management of 3D data and → textures
Data Source	The data source is either a → database or an → XML file.
Debug	Support for finding bugs for software developers
DTD	Syntax description language for → XML (**D**ocument **T**ype **D**efinition)
Element	Part of an → Element-Complex. Contains geometry data.
Element Complex	Smallest → Complex. Consists of several → Elements which form a solid. Examples: dormer windows, chimneys. An → Object consists of one or more Element Complexes
Layer	Line Type, e.g. Outer Eave, Ridge, Roof Line, etc.
LoD	"Level of Detail" of display: LoD0 = Plan (only eaves), LoD1 = Box Model (only rectangular solids, LoD2 = with the main roof shape, LoD3 = with roof details (e.g. dormers, chimneys, ...)
Master	Outer polygons of adjacent → Elements might be related via a Master/Slave relation, linking these Elements geometrically and logically. A Master can own multiple Slaves.
Model	Group of → Units in a certain area, e.g. all Units of a district may be grouped to a model. A Unit can belong to any number of Models
Object	Second smallest level of → Complex hierarchy. A → Unit consists of several Objects (e.g. the residential building + a garage, shed, etc). An Object has at least one → Element Complex (the Main Element Complex) and can have further Detail → Element Complexes.
Orthoimage/ Orthophoto	corrected photo with orthogonal facade borders and correct width/height relations (parallel projection), e.g. facade orthophotos, aerial orthophotos, etc. Serves as base for adding → texture to surfaces.
Selection Polygons	Outlines of → Units; in order toy be able to select them graphically; they are displayed in the drawing windows in gray, green (→ checked out in the current session) or red (checked out in another session).
Slave	see Master
Texture	For a photo-realistic appearance the surface is overlaid by an image, in most cases an → Orthophoto.
Triangulation	Derivation of 3D triangles for the surface of Units
Unit	Building which may have auxiliary buildings; an administrative unit; is composed of several → Objects; Examples: residential buildings, factory buildings, etc.).

UnitID

An Identifier for a → Unit which can be selected by the operator. It must be unique in a → data source. The ID may consist of letters and digits and is case-sensitive, i.e. lower/upper case of letters must be obeyed.

Version History

Documentation of the editing history of → Units. For each → Check-out or Import a new version is created.

VRML

Data Format (**V**irtual **R**eality **M**odelling **L**anguage), designed to support 3D scenes and can be read by various software

Weld Vertices

A feature of Autodesk 3D Studio Max and refers to the joining of separate lines to form a longer polyline. In doubt, answer with No!

Working Set

Subset of all → Units of the → data source which are available for loading in a working session

XML

Data Exchange Format (E**X**tensible **M**ark-up **L**anguage)

10. Contact

UVM Systems GmbH

www.citygrid.at

www.uvmsystems.com