KNOSSOS

Requirements Specifications

Version: 3.4
Date: 2007.04.23
State: Final

Author	**Date**	**Signature**
Áfra Attila		
Bán Sándor		
Barok Botond		
Nagy Timea		
Pap Lőrinc		
Szilágyi Péter		

Approved by	**Date**	**Signature**
Barabás László		

Table of Contents

I. Document History

Version	Date	Author	Modification
0.1	2007.03.19	Nagy Timea	Document introduction
0.2	2007.03.19	Barok Botond	Use case chapter
0.3	2007.03.19	Szilágyi Péter	Model IO + path functionality
0.4	2007.03.19	Áfra Attila	Exploration functionality
0.5	2007.03.19	Pap Lőrinc	Behavioral model
0.6	2007.03.19	Bán Sándor	Limitations and constraints
1.0	2007.03.19	Szilágyi Péter	Document compilation
1.1	2007.03.20	Szilágyi Péter	Corrections + Table of contents simplifications
1.2	2007.04.01	Barok Botond	Use case diagram split
1.3	2007.04.01	Pap Lőrinc	Interface design
2.0	2007.04.01	Szilágyi Péter	Second compilation
2.1	2007.04.15	Szilágyi Péter	Added requirements keys
3.0	2007.04.15	Áfra Attila	Final revision
3.1	2007.04.22	Szilágyi Péter	Added sys requirements, technology usage and some corrections
3.2	2007.04.23	Szilágyi Péter	Technology addition, key additions to the limits, reformulated the settings dialog description
3.3	2007.04.23	Áfra Attila	Review and minor corrections for submission
3.4	2007.04.23	Nagy Timea	Corrected some spelling mistakes

II. References

Name	Description
Path	A Path is a predefined traversal of the 3D architectural model captured and saved to a file.

III. Introduction

1. Goals and Objectives

Knossos is meant to be an interactive architectural visualization software, used for real-time building exploration. In contrast with other rendering engines, this one has it's strength in being able to show the whole building complex, which can be explored, viewed and examined by the client. Architects can present the finished building, even before it's foundation is laid.

The models can be saved in its own file format, and imported also, from various other popular formats (described later).

The software will allow highly detailed geometry, physically-correct transparent materials of different colors, partial opacity and optional world collision detection. The client can even record the traveled path and save it into a special file format, for reviewing later. Interactivity and real-time representation provides the client with good understanding and accurate visualization of the finished building complex.

The two exploration modes of the software are the fly-through and walk-through mode. Walk-through is equivalent to the familiar first-person-shooter game view, with gravity and with (optional) collisions. If walk-through is the human perspective, fly-through can be perceived as the bird-view.

This software will ease the architects and clients communication, being a more user friendly way to visualize, than simply watching the building plan on paper. They can also find flaws in the structure, which could otherwise be missed.

2. Statement of Scope

Knossos is a real-time, architectural visualization software. To get started, the building model must be loaded from a file format that the software supports. After loading the model the client can start cruising around.

Choosing the walk-through or fly-through mode the building can be inspected into its tiniest details. The client will be able to walk around the complex, go into rooms, offices, walk up the stairs, or analyze the structure from outside, using only the keyboard and mouse. The traveled path can be recorded for later presentation.

3. Software Context

As mentioned above Knossos is meant to be used by architects and even by their clients to visualize the finished building, analyzing every detail, in order for the errors to be corrected, before they even start working on it.

Knossos can be very useful in conferences where the clients expect that the architects provide the best understanding of the project. The software can give the clients a perspective about what the complex will look like, what features need changing and what parts are ready to be built. They can also monitor the progresses made starting from the beginning to the end of the construction of the complex. The advantage of Knossos versus a rendered walk is obvious: interactivity.

IV. Usage Scenario

1. Model opening:

1. Brief Description

The user opens a model given in the programs own specific file format (#Req#Def#OpenModel). If the model has associated paths [Path], then they will be also loaded.

2. Prerequisites

To open a model, the selected file must exist.

3. Course of Events

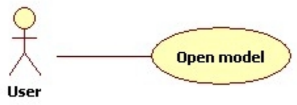
1. The system prompts the user to choose a model to be opened.
2. The user chooses the model to be opened.
3. The program verifies the file.
4. The program opens the model.
5. The program loads any associated paths [Path].

Alternatives:

1. The user chooses an inexistent or a bad file.
2. The program displays an error message.
3. Go to step 1 (Main path).
4. There are no associated paths [Path].
5. The program opens no paths [Path].

4. After-effects

The selected model will be opened and any existing associated paths [Path].



2. Model importing

1. Brief Description

The user imports a model from a different file format than that the default one (#Req#Def#ImportModel). If the model has associated paths [Path] (file with the same name), then they will be also opened.

2. Prerequisites

To import a model, the selected file must be exists.

3. Course of Events

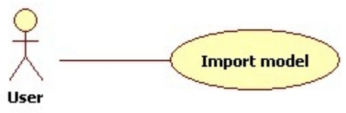
1. The system prompts the user to choose a model to be imported.
2. The user chooses the model to be imported.
3. The program imports the mode.
4. The program opens the associated paths [Path].

Alternatives:

1. The user chooses an inexistent model.
2. The program displays an error message.
3. Go to step 1 (Main path).
4. There are no associated paths [Path].
5. The program opens no paths [Path].

4. After-effects

The selected model will be opened and any existing associated paths [Path].



3. Path opening

1. Brief Description

The user opens a file containing one or multiple paths [Path] (#Req#Def#OpenPath).

2. Prerequisites

To open a path [Path], a model already must be opened.

3. Course of Events

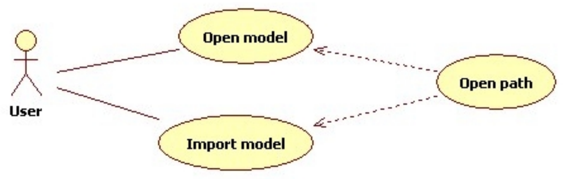
1. The system prompts the user to choose a path file to be opened.
2. The user chooses the file to be opened.
3. The program verifies the file.
4. The program opens the path [Path].

Alternatives:

1. The user chooses an inexistent or a bad file.
2. The program displays an error message.
3. Go to step 1 (Main path).

4. After-effects

The selected path or paths [Path] will be opened.



4. *Model Saving*

1. Brief Description

The user saves the imported or opened model into the programs own specific file type (#Req#Def#SaveModel).

2. Prerequisites

A model must already be loaded.

3. Course of Events

1. The program prompts the user to enter the file name.
2. The user enters the file name and presses the save button.
3. The program saves the model.

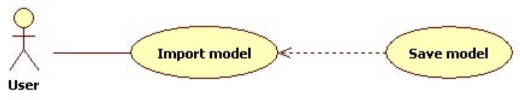
Alternatives:

1. The user enters an existent file name and presses the save button.
2. The program asks the user if he is sure.
3. The user presses the yes button.
4. Go to step 3 (Main path).

1. The user presses the no button.
2. Got to step 1 (Main path).

4. After-effects

The model will be saved.



5. Model Exporting

1. Brief Description

The user exports the saved model into the programs own special packed file type (#Req#Def#ExportModel).

2. Prerequisites

A model must be already loaded.

3. Course of Events

1. The program prompts the user to enter the file name.
2. The user enters the file name and presses the export button.
3. The program exports the model with all the associated data and paths [Path].

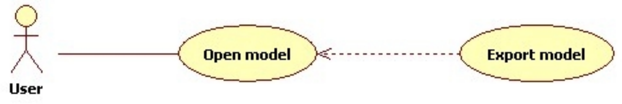
Alternatives:

1. The user enters an existent file name and presses the export button.
2. The program asks the user if he is sure.
3. The user presses the yes button.
4. Go to step 3 (Main path).

1. The user presses the no button.
2. Got to step 1 (Main path).

4. After-effects

The model will be exported.



6. Path Saving

1. Brief Description

The user saves the path [Path] (#Req#Def#SavePath).

2. Prerequisites

A path [Path] must exist in the workspace in order to be able to save it.

3. Course of Events

1. The program prompts the user to enter the file name.
2. The user enters the file name and presses the save button.
3. The program saves the path [Path].

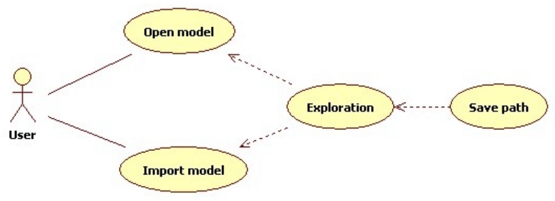
Alternatives:

1. The user enters an existent file name and presses the save button.
2. The program asks the user if he is sure.
3. The user presses the yes button.
4. Go to step 3 (Main path).

1. The user presses the no button.
2. Got to step 1 (Main path).

4. After-effects

The path [Path] traversal data will be saved.



7. Exploration

1. Brief Description

The user explores the architecture with different tools and options in fly-through or walk through mode, collision detection enabled or disabled (#Req#Def#Walk, #Req#Def#Orbit, #Req#Def#Flight, #Req#Def#Collision).

2. Prerequisites

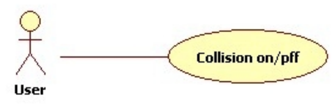
To take a tour in a building, a model must be opened or imported.

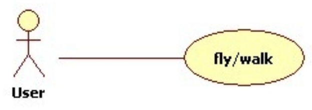
3. Course of Events

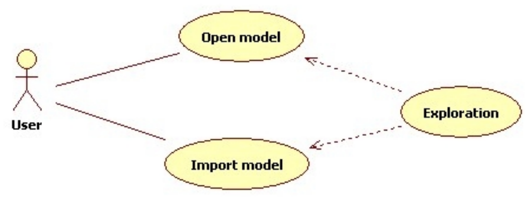
1. The user presets the direction with the arrow buttons or by moving the mouse.
2. The program calculates the new aspect, and displays it.
3. The user presses the W, S, A, or D button.
4. The program simulates the forward, backward, left or right moves.

Alternatives:

1. If the collision is on and the user tries to go through a solid object, the camera will evade it appropriately.







8. Replay

1. Brief Description

Replaying of a previously saved path [Path] for review or presentation (#Req#Def#PathPlayback).

2. Prerequisites

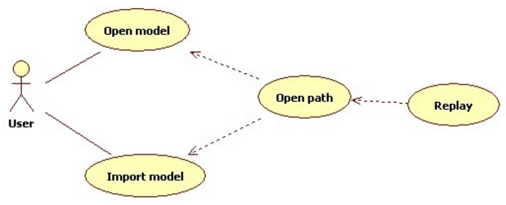
To replay a path [Path], a model must be opened, and a path [Path] must exist in it to be able to play it.

3. Course of Events

1. The user chooses the path [Path] to view.
2. The program starts the replay.

4. After-effects

The traversal will be displayed in a presentation mode to the user.



V. Functional Model and Description

1. Model Input/Output Functionality Description

1. General Presentation

This is one of the core functionalities of the software, the ability to open/load/import 3 dimensional architectural models from a multitude file formats (#Req#Def#OpenModel, #Req#Def#ImportModel). The program will also enable the user to add additional formats if these are not supported by default (#Req#Def#PluginAddition).

The program will provide a general interface for a module loader, which can be implemented by anyone, thus allowing third parties to create their own loaders to extend the variety of supported formats. These can include open formats not supported by default, or proprietary ones that companies might implement and potentially sell to be able to import their own specific models. These additional components must be easily addable to the program by the casual user.

Due to the vast variety of model formats, some are more complex than others, some require more processing time than others, thus the program will enable the user to export (#Req#Def#SaveModel, #Req#Def#ExportModel) the loaded and parsed model to a program specific file format which has been extendedly optimized for loading speed (for multiple subsequent loading of the same model).

2. Performance Issues

3D architectural models are very complex in nature, starting with geometry, materials, transparency and textures; but also very large. They thus require many calculations to convert to a generic one which can be processed by the graphical engine. With this in mind long and time consuming initial calculations could take place when importing a model from file.

3. Design Constraints

The module must be designed in such a manner that it will support runtime addition of new model format loaders, and must supply a way for third parties to implement and distribute their own custom formats without the need of special/sensitive material (#Req#Def#PluginFramework).

2. Exploration Functionality Description

1. General Presentation

The main purpose of Knossos is the exploration of the 3D architectural model. For maximum effectiveness and exploration freedom several navigation modes are included. In every case collision detection can be turned on to prevent the user from passing through objects (#Req#Def#Collision).

The orbiting mode enables the user to view the model from a bird's eye view, move around it on a spherical path and zoom in and out (#Req#Def#Orbit).

The flying mode provides the ability to move to any point in space without any limitations. By using this feature the model can be examined from virtually any angle and position (#Req#Def#Flight).

Finally the most natural and intuitive way to explore the model is using the walking mode. This is the closest approach to real-life examination and analysis and users familiar with 3D first person view games will find it very easy to use right from the beginning (#Req#Def#Walk). If collision detection is enabled it is also possible to turn on gravity for an even more realistic experience.

The exploration will also feature correct light source and colored transparency handling (#Req#Def#NaturalLight).

2. Performance Issues

The exploration is the most calculation intensive task from the whole program, using as much computing power as it get possibly get (both the CPU and GPU), but still in order to explore huge models lifelike, a powerful system will be needed.

3. Design Constraints

None worth mentioning.

3. Path Track/Replay Functionality Description

1. General Presentation

The program is not only aimed at architects but also wants to enable the user to hold presentations in which to show demos of the building, real time traversals which do not require (or very minimal) user interaction. Thus a way must be provided so that the presenter can create and work on the demo well before the conference and there be able to show everything and speak freely without concentrating on the program itself.

Path tracking will consist of two phases. The path build phase (#Req#Def#PathRecord), when the user will have the opportunity to record the path traversed, the camera positions and view angles, and save all this in a special file (#Req#Def#SavePath). This file could also contain multiple traversals (uniquely identified by the path's name). This path file could then be loaded (#Req#Def#OpenPath) by the program enabling to quickly switch between different traversals of the same model and start replaying it.

During the replay it will be possible to pause the system and resume, but also to fast forward much like with a media player slider (#Req#Def#PathPlayback). The camera could be unlocked from the path, thus enabling the user to look freely while automatically moving through the model. And of course leaving the predefined path is also a possibility.

2. Performance Issues

This functionality should not present the user with any performance issues whatsoever.

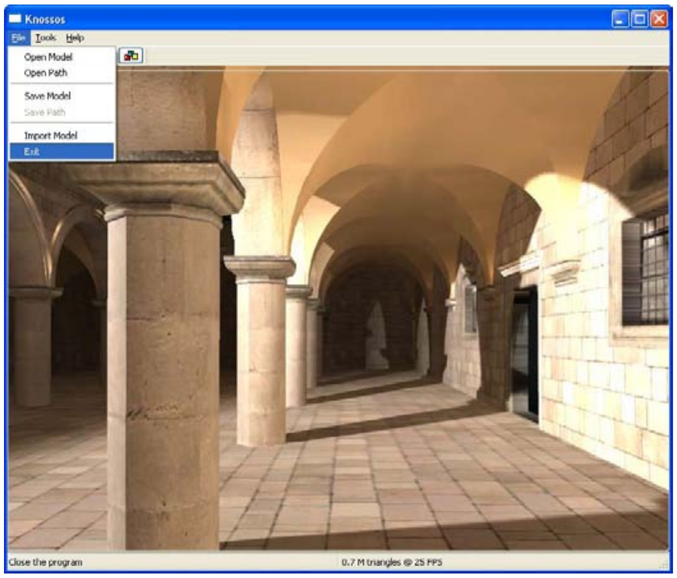
3. Design Constraints

The module should be easily expandable, to allow additional operations during the path traversal replay. (ex: addition of a renderer which would traverse the path and generate a short film from it).

VI. Graphical User Interface Design

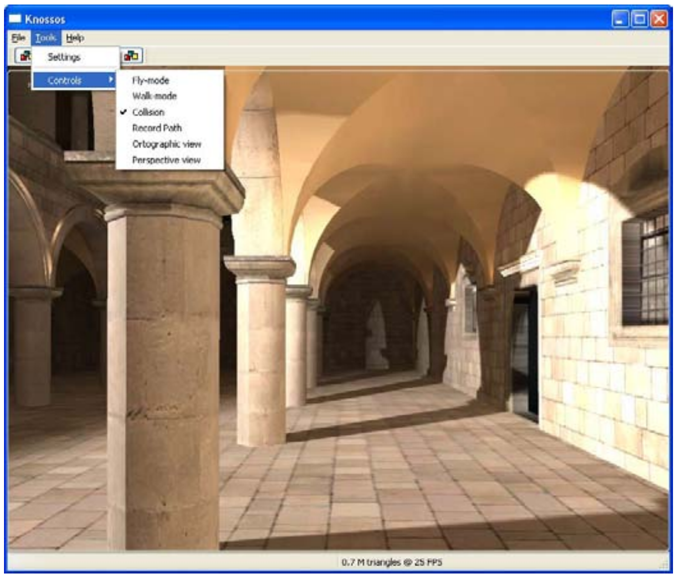
1. File Handling

Controls related to all load and save operations are here.



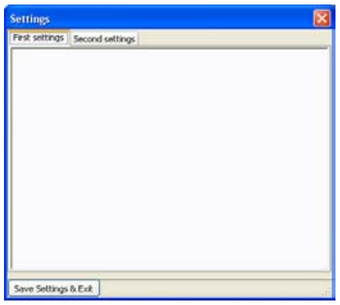
2. Controls

Each control will have a shortcut (#Req#Def#Shortcuts), a push button on the tool bar (#Req#Def#Toolbar) and a menu item (#Req#Def#Menu).



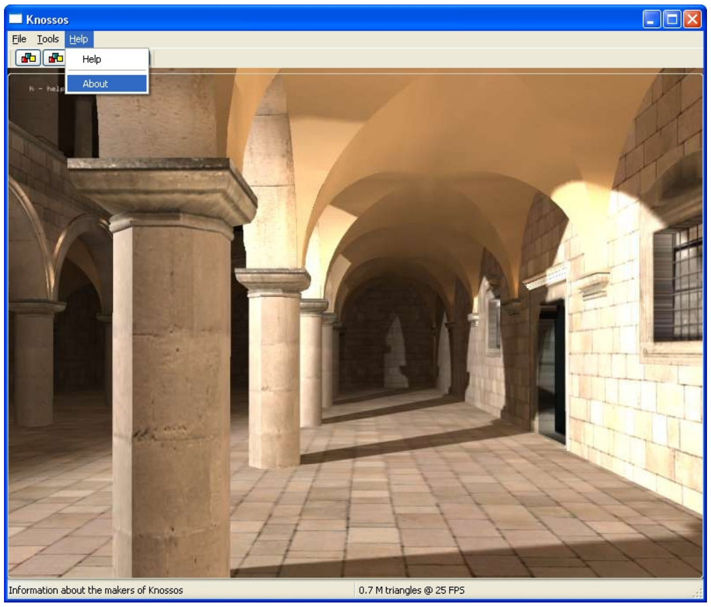
3. Settings

A settings menu will be available to adjust various graphics options to fine tune the performance according to the user's specific computer (#Req#Def#Settings). This will include the following options: sight distance, resolution and color depth for full screen mode, time span between two consecutive control points during path [Path] construction.



4. Help

A help menu is provided for learning the usage of Knossos (#Req#Def#Help).

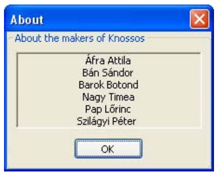


Information about the makers of Knossos

0.7 M triangles @ 25 FPS

5. Authors

The appropriate credits are given to each member, specifying the merits of the individuals.



About the makers of Knossos

Áfra Attila
Bán Sándor
Barok Botond
Nagy Tímea
Pap Lőrinc
Szilágyi Péter

VII. Restrictions, Limitations and Constraints

In Knossos, there are multiple factors that impact specification, design, and implementation of the software.

1. *Model Processing Calculations*

Importing a new model can result in demanding and time consuming calculations. Expect processing times of up to minutes. Of course hardware performance has a major impact on this.

2. *File Formats*

Knossos will support by default only loading from it's own special format and importing from the OBJ model format, but will be able to use input plugins for other 3D modeling file formats (#Req#Def#PluginFramework, #Req#Def#PluginAddition).

It will not have the functionality of converting one file format to another.

Besides model processing calculations, the importing of the model itself will take some time, depending on the size of the model being imported, and also on the loader plugin.

The program's own file format will be optimized solely for loading speed.

3. *Exploration*

In walk through mode (#Req#Def#Walk) you won't be able to visualize from every point of view because of gravity.

The ability to climb stairs and not too tall objects will be linear in appearance, the camera being point like.

In fly through mode (#Req#Def#Flight) there will not be gravity.

4. *Collision Detection*

Collisions detection (#Req#Def#Collision) will use a bounding box technique, thus it will not allow passing under arbitrarily small passages.

5. *Performance Bounds*

A minimum of 15-25 FPS (Frames Per Second) must be reached in order for the exploration to be enjoyable.

Using the program's special optimized file format, model loading should take only a few seconds.

VIII. System specifications

1. *Hardware requirements*

Knossos is designed to process in real time large sets of data, thus is requires a computer that's able to cope with the requirements:

1. Minimal

- 1 GHz processor
- 256 MB RAM
- OpenGL 2.0 compatible video card

2. Recommended

- 64-bit multi core and/or multi processor system
- 1+ GB RAM
- OpenGL 2.1 compatible video card

2. *Software requirements*

Knossos will be compatible with the following operating systems:

- Windows XP i386 and AMD64
- Windows Vista i386 and AMD64
- Linux 2.6 i386 and AMD64

IX. Appendix

1. Technologies used

Technology	Website, Description and Usage
OpenGL 2.1	http://www.opengl.org/ Cross platform 3D graphics API. Knossos will use OpenGL in the core rendering engine for displaying the 3D models in real time interactively.
DevIL 1.6.8	http://openil.sourceforge.net/ Cross platform image loading library. Knossos will use this library in the OBJ model loader plugin to enable the loading of any type of textures for use in the rendering.
OpenMP 2.0	http://www.openmp.org/ Multi-platform shared-memory parallel programming API. Knossos will use the OpenMP parallel technology for every task that can be parallelized to run on multi processor or multi core systems.
wxWidgets 3.8.3	http://www.wxwidgets.org/ Cross platform native GUI toolkit. Knossos will use wxWidgets for the graphical user interface.

2. Requirements Table

Key	Definition
#Req#Def#OpenModel	Open a model from a program specific file format
#Req#Def#ImportModel	Import a model from a foreign file format
#Req#Def#OpenPath	Open a path [Path] traversal file
#Req#Def#SaveModel	Save a model to the program optimized file format
#Req#Def#ExportModel	Export a model to a special compressed format
#Req#Def#SavePath	Save the paths [Path] associated to a specific model
#Req#Def#PluginAddition	Provide functionality to add additional model importer plugins to handle file formats originally not supported
#Req#Def#PluginFramework	A framework project which could be openly implemented to enable third party/commercial plugin development
#Req#Def#Walk	Walking exploration mode for the model
#Req#Def#Orbit	Orbiting exploration mode for the model
#Req#Def#Flight	Flying exploration mode for the model
#Req#Def#Collision	Detecting the collisions with solid objects
#Req#Def#NaturalLight	Handling light and transparency effects in as a natural looking way as possible
#Req#Def#PathRecord	Recording a traversal while moving through the model
#Req#Def#PathPlayback	Playing back a pre recorded path [Path] traversal
#Req#Def#Menu	Menu containing all the program functionality
#Req#Def#Shortcuts	Shortcut keys for accessing the same functionality by power users
#Req#Def#Toolbar	Tool bar buttons yet again for the same functions
#Req#Def#Settings	Menu to allow fine tuning the setting to match the host computer's performance
#Req#Def#Help	A user documentation to help getting started with the program