interactive networked virtual reality system

—

Code Conventions

Christoph Anthes and Roland Landertshamer

June 15, 2009

Contents

Chapter 1

Code Conventions

This document provides the code conventions for the *inVRs* framework [AV06]. Some preliminary definitions are given to provide a common understanding in the document. They might seem obvious but are useful to avoid misunderstandings.

- Member functions are functions that are associated with classes.
- Data members are variables that are part of classes.
- Headers are the actual files which contain includes and class declarations.
- Source files contain the implementation of the given headers.

1.1 Files and File Names

Each class has an according header and a source file with the same name as the class. For each header an appropriate source file with the same name exists. There are a few exception to this rule which are:

- Events
- Navigation Models (OrientaionModels.h, TranslationModels.h, SpeedModels.h)

Events and Navigation Models often contain small classes and thus sets of them are kept in header and with te appropirate source files.

1.1.1 Filenames

Header file names end with the suffix .h while source files end with the suffix .cpp. The filenames always begin with a capital letter. In general camel-case writing is appropriate.
Abbreviations with more than one letter have the following letters written in lower case. An example is given below:

```
GlutMouseDevice.h
GlutMouseDevice.cpp
```

1.2 Classes

Classes are to be declared in headers with the same name. The implementation is to be fully kept inside the according source file.

1.2.1 Naming

Reasonable naming identifying the task of the class has to be found. Each *inVRs* module or core component contains a management class which handles most parts of the component. This class carries the same name as the component.

1.2.2 General Structure

The structure of the headers is to be build as follows:

- Copyright Header
- License
- Changelog
- Define Header
- Includes
- Defines
- Forward Declarations
- Class Documentation
- Class Declaration

The appropriate copyright header, license and changelogs are rather generic and are given in Chapter 4.

1.2.3 Includes

The source file contains as a first include the according header file. The order of external includes inside the headers and the source files is given in the following list. A single empty row is to be kept between the different types of headers.

- C++
- STL
- GMTL
- irrXML
- OpenSG
- Internal *inVRs* includes

1.2.4 Visibility Order

- Public
- Protected
- Private

Classes are declared in the following order: public member functions, protected member functions, private member functions, protected data members, and private data members. No data members may be declared public. A more fine grained example is given below:

```
class ClassName
{
  public:
     // Constructors:
     // Destructor:
     // Functions:
        // if available init
                    // loadConfig
                    // cleanup
        // other functions,
        // modifiers (set),
        // selectors (get)
     // Attributes visible by scope of instantiation and use
  protected:
     // Attributes visible to descendents
  private:
     // Local attributes
};
```

Listing 1.1: Example Member Declaration

The order of the member functions has to be the same in the header as in the source file.

1.3 Data Members

The names of data members begin with small letters. Abbreviations with more than one letter have the following letters written in lower case. Hungarian notation is to be avoided (no types in the variable name)

```
//bad
void update(float fTimePassed);
//good
void update(float timePassed);
```

1.3.1 Constants

Constants in the the code are to be written with capital letters only.

```
static const unsigned APPENDED_MESSAGE;
```

1.3.2 Enumerations

The same is true for Enumeration data types. They are to be written as well with capital letters.

```
/// Enum for possible argument types
enum ARGUMENT_TYPE {
  BOOL, INTEGER, UINTEGER, FLOAT, STRING
};
```

1.4 Member Functions

The names of member functions begin with small letters. Abbreviations with more than one letter have the following letters written in lower case. No underscores are to be used in the function name.

Chapter 2

Comments

To improve the readability and the understanding of the code comments and documentation are necessary. The *inVRs* framework makes use of Doxygen (http://www.stack.nl/~dimitri/doxygen/) as an automatic code documentation system. Additional comments inside the code are often available as well. The Doxygen documentation for inVRs is available under http://doxygen.invrs.org/)

2.1 Doxygen Documentation

The Doxygen code documentation is to be kept fully inside the headers. No Doxygen comments are to be placed anywhere in the source files.
All classes are to be tagged with the class and the brief attribute. Additional attributes like authors or version can be used as well but are not required. An example for the documentation of the Navigation class can be found below.

```
/*****************************************************************************
 * @class Navigation
 * @brief This is the main class of the Navigation module.
 *
 * This class makes the whole navigate calculation based on different models
 * set by calling the registered Orientation-, Translation-,and Speedmodels)
 */
class Navigation : public NavigationInterface{
public:
    ...
```

Listing 2.1: Example Class Comments

An example for a member function is given below.

```
    /**
     * The loadConfig method parses a passed String, which should contain XML
     * data.The configuration contains the names for speed-, orientation-, and
     * translationmodel, which are subsequently parsed and evaluated.
     * @param configFile the XML configuration for the navigation models
     * @return in case the parsing does not detect errors in the configuration
     * true is returned, if errors are detected false is returned
     */
    virtual bool loadConfig(std::string configFile);
```

Listing 2.2: Example Member Comments

The member functions should be documented as well. If documentation for a member function is available the attributes param and return have to be set.

2.2 C++ Comments

Unless they are Doxygen comment comments always use //, even if they range over multiple lines of code. It is to be avoided to use C-style comments, even it they are multi-line, due to consistency and thus better readability in the code style.

Chapter 3

File Structure

The *inVRs* framework consists of a variety of independent as well as intertwined components. The code parts are written in C++ while the configuration is stored in an XML file format.

3.1 Code

The code directories of the *inVRs* framework match the component types and the individual components. Interfaces, Modules and System Core are the basic directories where the libraries are located. Each module as well as the whole system core compiles to an individual library.
Additionally a set of tools is available and external libraries which are to be used with *inVRs* are stored in separate directories under the external directory.

3.2 Configuration

Ideally the configuration directory structure for an *inVRs* application is to applied as illustrated in Figure 3.1.

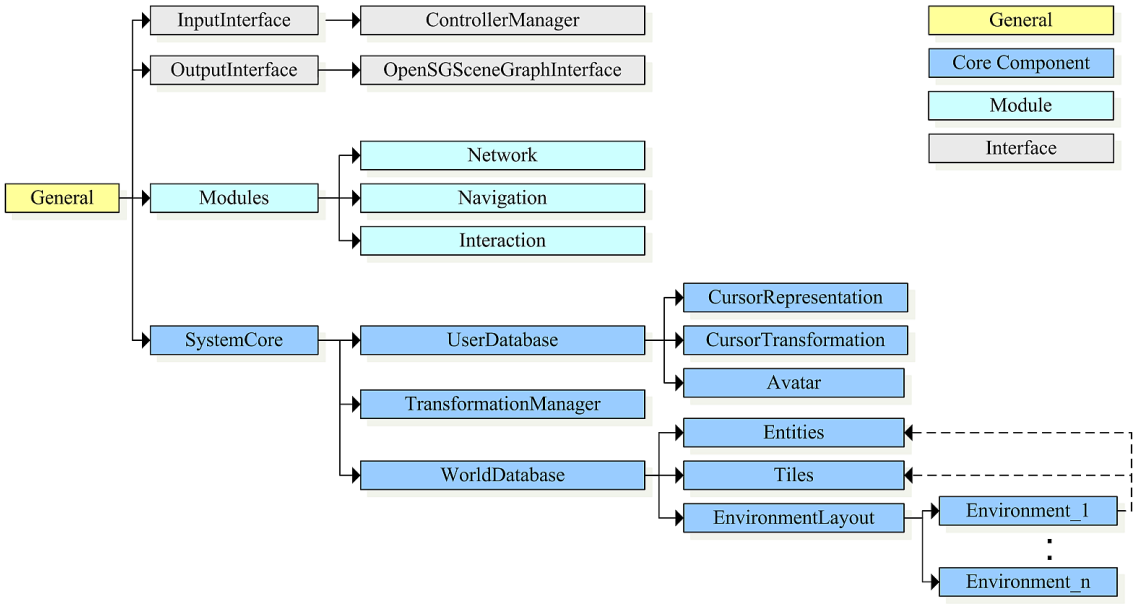

Figure 3.1: Standard Configuration Hierarchy

Chapter 4

Example Header

The structure of the headers is to be build as follows:

4.1 Copyright Header

```
/*-----------------------------------------------------------------------*\
 *            interactive networked Virtual Reality system (inVRs)       *
 *                                                                       *
 *    Copyright (C) 2005-2009 by the Johannes Kepler University, Linz    *
 *                                                                       *
 *                             www.inVRs.org                             *
 *                                                                       *
 *               contact: canthes@inVRs.org, rlander@inVRs.org           *
\*-----------------------------------------------------------------------*/
```

4.1.1 License

```
/*-----------------------------------------------------------------------*\
 *                               License                                 *
 *                                                                       *
 * This library is free software; you can redistribute it and/or modify it   *
 * under the terms of the GNU Library General Public License as published    *
 * by the Free Software Foundation, version 2.                           *
 *                                                                       *
 * This library is distributed in the hope that it will be useful, but       *
 * WITHOUT ANY WARRANTY; without even the implied warranty of                *
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU          *
 * Library General Public License for more details.                      *
 *                                                                       *
 * You should have received a copy of the GNU Library General Public         *
 * License along with this library; if not, write to the Free Software       *
 * Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.              *
\*-----------------------------------------------------------------------*/
```

4.1.2 Changelog

```
/*-----------------------------------------------------------------------*\
 *                               Changes                                 *
 *                                                                       *
 *                                                                       *
 *                                                                       *
 *                                                                       *
 *                                                                       *
\*-----------------------------------------------------------------------*/
```

4.1.3 Defines

```
#ifndef _NAVIGATION_H
#define _NAVIGATION_H
```

4.1.4 Includes

The includes in an inVRs file are to be entered in the following order:

- External Headers
- inVRs Headers

```
#include <gmtl/Quat.h>
#include <gmtl/Matrix.h>

#include <irrXML.h>

#include "OrientationModels.h"
#include "SpeedModels.h"
#include "TranslationModels.h"
#include "../SystemCore/EventManager/EventManager.h"
#include "../SystemCore/ComponentInterfaces/NavigationInterface.h"
#include "NavigationEvents.h"

#ifdef WIN32
#pragma warning( disable : 4507 )
#endif
```

Bibliography

[AV06] Christoph Anthes and Jens Volkert. invrs - a framework for building interactive networked virtual reality systems. In *International Conference on High Performance Computing and Communications (HPCC '06)*, volume 4208 of *Lecture Notes in Computer Science (LNCS)*, pages 894–904, Munich, Germany, September 2006. Springer.