Overview of the CHILD Source-Code Structure

Greg Tucker

June 2008

Contents

1 Overview **2**

2 General Header Files: Classes, compiler, Definitions, Inclusions, Template_Model, and trapfpe **2**

3 Basic Utilities **3**
3.1 Error Handling 3
3.2 Command-Line Options: tOption 3
3.3 Mathematics: Geometry, Mathutil, and Predicates 3
3.4 Vectors and Matrices: tArray and tMatrix 3
3.5 Linked List Management: tList, tPtrList, and tMeshList 3
3.6 Time Management: tRunTimer 5

4 Mesh Construction and Management: tMesh, tLNode and MeshElements **5**

5 Parameter and Data Input: tInput and tTimeSeries **5**

6 Output: tOutput, tLOutput **5**

7 Rainfall and Runoff: tStorm and tStreamNet **5**

8 Erosion and Transport: Erosion **5**

9 Stratigraphy **5**
9.1 Layers beneath Nodes 5
9.2 High-Resolution Gridded Stratigraphy: tStratGrid 5

10 Uplift, Subsidence and Baselevel Change: tUplift **5**

11 Special Processes: Channel Meandering, Floodplain Deposition, Eolian Deposition, Vegetation **5**
11.1 Stream Meandering: tStreamMeander 5
11.2 Floodplain Deposition: tFloodplain 5
11.3 Vegetation Dynamics: tVegetation 5
11.4 Dust Accumulation: tEolian 5

1 Introduction

The CHILD (version 2008) source code is divided among a number of different source files. These source files represent, more or less, the various modules and utility classes that together make up CHILD. This document gives a brief overview of the module and file structure. It is intended for developers who wish to add capabilities, create linking software, or couple CHILD with other models. I assume you are basically familiar with C++. File names and file sets are given in **bold** and class names are given in *italics*. By "file set" I mean a group of related files that reside in the same folder (for example, **tUplift.h** and **tUplift.cpp**).

The code consists of several different file groups, most of which reside in their own sub-folders and usually consist of a header (.h) and source (.cpp) file. Sometimes there will be only a header file, and sometimes more than one header and/or source file.

The high-level structure, including initialization and the implementation of a time loop, is handled by a relatively small "main" routine contained in the files **childmain.cpp** (for the development version) and **childrmain.cpp** (for the general release version). Most of the main modules, which handle processes such as water routing, erosion, and tectonic uplift, are implemented as classes (such as the tStreamNet, tErosion, and tUplift classes). (The file **toddlermain.cpp** implemented a now-obsolete release version).

2 General Header Files: Classes, compiler, Definitions, Inclusions, Template_Model, and trapfpe

The header files handle various things. **Classes.h** tells the compiler what classes to expect. **Template_model.h** and **compiler.h** are meant to handle differences between compilers (they aren't foolproof). **Definitions.h** provides a list of defined

constants and macros. **trapfpe.h**, which I think was written by Arnaud Desitter, enables floating-point-exception traps in Linux.

3 Basic Utilities

3.1 Error Handling

The small **errors** file set takes care of reporting errors and warnings.

3.2 Command-Line Options: tOption

The small **tOption** fileset parses and handles command-line options.

3.3 Mathematics: Geometry, Mathutil, and Predicates

The **mathutil** file set implements random-number generation through the *tRand* class. It includes an implementation of a Mersenne Twister pseudorandom number generator. The **geometry.h** file defines classes for 2D and 3D points with constructors and assignment operators. The **predicates** file set handles arbitrary precision floating-point arithmetic, using a subset of a public domain code modified by Stephen Lancaster.

3.4 Vectors and Matrices: tArray and tMatrix

The first version of CHILD was written before the Standard Template Library was created, so it uses its own array (vector) handling routines in the **tArray** file set. This implements the *tArray* class, which handles 1D arrays with bounds-checking, constructors, etc. It is widely used throughout the code. The **tMatrix** file set implements 2D arrays, or matrices, using the *tMatrix* class. It is not used especially frequently in the code, however.

3.5 Linked List Management: tList, tPtrList, and tMeshList

CHILD makes heavy use of linked lists, which come in several flavors. Basic two-way linked lists of objects are handled by the **tList** file set. It includes several classes: *tList* implements the list itself, *tListNodeBasic*, *tListable* and *tListNodeListable* implement the nodes that make up the list, and *tListIter* implements iterators that move up and down a list to access its nodes. Lists of pointers to objects, as opposed

to lists of the objects themselves, have to be handled a little differently, and are implemented by the **tPtrList** file set.

In addition to these, **tMeshList** handles a derived type of list designed specifically for handling lists of mesh elements (nodes, edges, and triangles). A *tMeshList* is a *tList* that is divided into two sections: an "active" section (at the "top" of the list) representing nodes and edges that fall inside the mesh boundaries. In other words, nodes that are part of the computed domain and whose elevations change, etc., are placed in the *active* portion of the list. Boundary nodes (whether open or closed to water and sediment) are placed in the *inactive* ("bottom") portion of the list. This makes it easy to access a list of active nodes. Edges are handled in a similar fashion. The *tMeshListIter* class provides iterators to access a *tMeshList*.

3.6 Time Management: tRunTimer

4 Mesh Construction and Management: tMesh, tLNode and MeshElements

5 Parameter and Data Input: tInput and tTimeSeries

6 Output: tOutput, tLOutput

7 Rainfall and Runoff: tStorm and tStreamNet

8 Erosion and Transport: Erosion

9 Stratigraphy

9.1 Layers beneath Nodes

9.2 High-Resolution Gridded Stratigraphy: tStratGrid

10 Uplift, Subsidence and Baselevel Change: tUplift

11 Special Processes: Channel Meandering, Floodplain Deposition, Eolian Deposition, Vegetation

11.1 Stream Meandering: tStreamMeander

11.2 Floodplain Deposition: tFloodplain

11.3 Vegetation Dynamics: tVegetation

11.4 Dust Accumulation: tEolian