

LANDIS-II – Software Requirements

James B. Domingo
Robert M. Scheller

University of Wisconsin-Madison

Last Revised: March 28, 2005

DRAFT

Table of Contents

1	INTRODUCTION	2
2	FUNCTIONAL REQUIREMENTS	3
2.1	LANDIS MODEL.....	3
2.2	ADDITIONAL COMPONENTS	3
2.3	NEW COHORT TYPES.....	3
3	USE CASES	5
3.1	USAGE NARRATIVES.....	5
3.1.1	<i>Console User Develops Scenario</i>	5
3.1.2	<i>GUI User Develops Scenario</i>	6
3.2	ACTORS	6
3.3	SYSTEM SCOPE	7
3.4	USE CASE 1 – DEVELOP MODEL SCENARIO	10
3.5	USE CASE 2 – RUN SCENARIO	11
3.6	USE CASE 3 – MANAGE SCENARIO DATA.....	12
4	SUPPLEMENTARY REQUIREMENTS	13
4.1	WEB SITE	13
4.2	PERFORMANCE	13
4.3	PLATFORM.....	13

1 Introduction

This document describes the software requirements for the LANDIS-II project. The target audience has two groups of stakeholders:

- the **ecologists** who must be certain that the requirements describe the software that they need built; and,
- the **software developers** who need to understand the requirements so they can build the software that satisfies those requirements.

2 Functional Requirements

2.1 LANDIS Model

The LANDIS-II software implements the LANDIS forest landscape simulation model. The model simulates how ecological processes including succession, seed dispersal, disturbances, and climate change affect a forest landscape over time. The model is described in a set of accompanying documents:

- *LANDIS – Model Description*
Describes the core concepts of the LANDIS model. This document should be read before any other documents in this set.
- *LANDIS – Succession*
Describes the different versions of the succession component.
- *LANDIS – Wind*
Describes the different versions of the wind disturbance component.
- *LANDIS – Fire*
Describes the different versions of the fire disturbance component.
- *LANDIS – Harvest*
Describes the different versions of the harvest disturbance component.

2.2 Additional Components

The software must allow users to install additional components that have been developed by third-party developers. Developers can create new succession components, disturbance components, and output components. These new components can then be distributed to LANDIS-II users, who can install them on their systems and use in their simulations.

2.3 New Cohort Types

LANDIS-II must permit developers to create a new succession component with a different type of cohort data. For example, developers could create a biomass succession component that uses

species cohorts which have a biomass attribute in addition to the basic age attribute.

Because disturbance and output components can be designed to work with certain types of cohorts, the set of components that can be used in a scenario depends upon the succession component selected by the user. For example, a fire disturbance component that's designed to work with biomass cohorts can only be used in a simulation where the user selects the biomass succession component. However, a wind component that only needs age information for cohorts can be used with any succession component whose cohorts have age attributes (e.g., age-only succession, biomass succession).

3 Use Cases

The Unified Software Development Process is a use-case driven process: it utilizes use cases as an effective means to capture the behavioral requirements of the software system under development.

A **use case** describes how an individual will interact with a software system (LANDIS-II) to accomplish a particular goal. In essence, it describes the sequence of actions that the person (referred to as the **actor**) does to perform a certain task.

Actors can be viewed as roles that people play. A person may be different actors in different situations. For example, a person would act as an Administrator when installing a program, and then act as a User when running the program.

Use cases focus on the actor's intentions and the system's behavior, and do not include details about the user interface.

The goals for the use cases are hierarchical, in that a goal at one level consists of sub-goals at a lower level.

- **User goals** represent tasks that an actor can accomplish in a single work session in front of a computer. These are of the greatest interest.
- **Summary-level goals** include multiple user goals. They represent tasks that typically take a long time for an actor to complete, involving multiple sessions in front of a computer.

3.1 Usage Narratives

A usage narrative is a very specific, concrete example of an actor interacting with the system. These short narratives are written first to help identify the system's use cases.

3.1.1 Console User Develops Scenario

Kate is a modeler who is familiar with running console applications from a command prompt. She creates the initial version of the model scenario. Since a scenario consists of a set of text files and raster files (maps), Kate uses a text editor and a GIS to create these files. She runs LANDIS-II from the command prompt. She examines the various output files using the text editor to view text files and the GIS to view output maps. She also uses statistical software to analyze some of the output data. Based on her examination, Kate adjusts some of the input

data in the scenario's files, reruns LANDIS-II, and then examines the new set of output files. Kate continues this cycle – adjust input, run model, examine output – until she's satisfied with the output.

3.1.2 GUI User Develops Scenario

Gary is a modeler who uses applications via their graphical user interfaces (GUIs). He does not use console applications; in fact, he does not even know how to open a command prompt on his computer.

He creates the initial version of the model scenario using LANDIS-II. Although he can use LANDIS-II to enter some parameters (for example, species and ecoregion parameters), he uses a GIS to create the input raster files that the scenario refers to. Once Gary has finished setting up the scenario, he runs the scenario by pressing the "Run Scenario" button on the LANDIS-II toolbar. LANDIS-II displays a list of the output files from which Gary can select a file for viewing. If a selected output file is a text file, LANDIS-II displays its contents. If an output file is a map, LANDIS-II requests the GIS to display it. Gary also uses LANDIS-II to run other programs on some of the output files such as statistical software like IAN.

Based on his examination of the output data, Gary adjusts some of the input data in the scenario's using LANDIS-II and the GIS. He instructs LANDIS-II to run the modified scenario, and then examines the new set of output files. Gary continues this cycle – adjust input, run scenario, examine output – until he's satisfied with the output.

3.2 Actors

- **Modeler** – Uses LANDIS-II. No programming experience required.
 - **Console Modeler** – Uses LANDIS-II via its console interface. Has experience running console programs from a command prompt.
 - **GUI Modeler** – Uses LANDIS-II via its GUI. Has experience using applications with graphical user interfaces (GUIs).
- **System Administrator** – Installs LANDIS-II software on users' computers.
- **Developer** – Writes code for parts of LANDIS-II.

- **Core Developer** – Member of the project team who writes code for the LANDIS-II core framework.
- **Component Developer** – Programmer who creates a component (extension) that plugs into the LANDIS-II core framework.

3.3 System Scope

It is important that each use case clearly identify its scope: the extent or boundaries of the system that the actor is interacting with. Figure 1

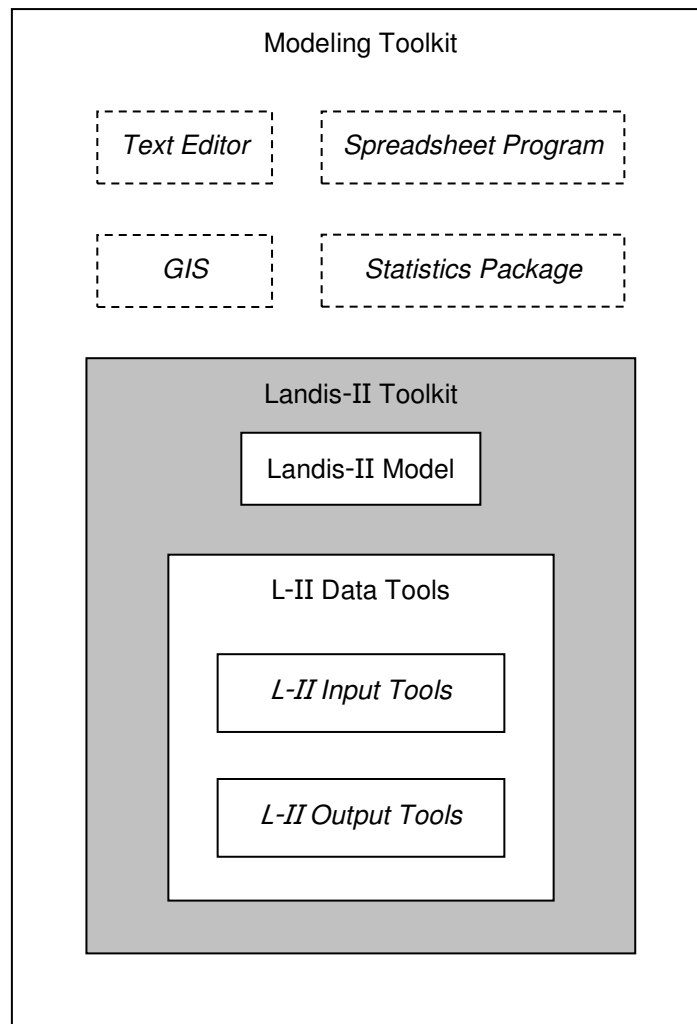


Figure 1 – Different System Scopes for Use Cases.

shows the levels of system scope for the LANDIS-II use cases.

- **Modeling Toolkit** – The collection of applications used by a modeler. The applications function as different types of tools:
 - **Display Tools** allow the modeler to view the contents of data files. Examples: text editors, spreadsheet programs, GISs, map viewers.
 - **Analysis Tools** allow the modeler to analyze data files in a particular manner (e.g., compute statistics). Examples: spreadsheet programs, statistical packages, GISs.
 - **Edit Tools** allow the modeler to create new data files and modify existing files. By nature, these tools can also be used as display tools. Examples: text editors, GISs, spreadsheet programs.
 - **Generation Tools** allow the modeler to generate data files by executing a set of instructions. Examples: models, macros in spreadsheet programs, procedures in statistical packages, scripts (e.g., Python, Ruby, Perl), programs for converting old LANDIS data into LANDIS-II format.
- **Landis-II Toolkit** – The set of tools specific to LANDIS-II.
- **Landis-II Model** – The program that actually implements the computational model. The modeler runs a LANDIS-II scenario with this program. It is always present in the Landis-II toolkit.
- **L-II Data Tools** – The set of tools specific to LANDIS-II data (input and output). These tools are optional; the minimum configuration for the Landis-II Toolkit is just the model by itself. With this minimal configuration, the modeler uses other tools in the Modeling Toolkit to work with LANDIS-II data.
- **L-II Input Tools** – The set of edit and generation tools that allow the modeler to create and edit LANDIS-II input data. These tools typically have graphical user interfaces (GUIs) and include the editors for the data common to all scenarios (e.g., species parameters), as well as the editors for the parameters for model extensions.

- **L-II Output Tools** – The set of display and analysis tools specific to LANDIS-II output data. Typically, these have GUIs.

3.4 Use Case 1 – Develop Model Scenario

Primary Actor: Modeler

Goal: To develop a model scenario for LANDIS-II.

Scope: Modeling Toolkit

Level: Summary (multiple user goals)

Preconditions: LANDIS-II Model has been installed.

Basic Course of Action:

1. Modeler creates a scenario (UC 3)¹ using edit and generation tools.
2. Modeler runs the scenario (UC 2) with LANDIS-II Model.
3. Modeler evaluates the output data² using display and analysis tools.
4. Based on the output analysis, the Modeler edits the scenario (UC 3) to adjust its parameters.
5. Modeler repeats steps 2 through 4 until satisfied with the output.

Alternative Courses:

- 1a. Modeler is developing a new scenario based on an existing scenario.
 - 1a1. Modeler creates a copy (UC 3) of the original scenario.
 - 1a2. Modeler edits the copy (UC 3) to create the new scenario.

Notes:

- ¹ An underlined phrase refers to another use case; the number of that use case follows the phrase as “(UC #)”.
- ² This use case has not yet been written; it may not be needed until work begins on the L-II Data Tools.

3.5 Use Case 2 – Run Scenario

Primary Actor: Modeler

Goal: To run a model scenario for LANDIS-II.

Scope: LANDIS-II Model

Level: User goal

Preconditions: Scenario exists. LANDIS-II Model has been installed.

Basic Course of Action:

1. Modeler starts the execution of the model scenario.
2. LANDIS-II reports that the scenario completed successfully.

Alternative Courses:

- 2a. Modeler interrupts the scenario:
 - 2a1. LANDIS-II confirms that Modeler wants to stop the scenario.
 - 2a2. LANDIS-II stops the scenario execution, reports the stoppage to the Modeler, and logs the stoppage.
- 2b. LANDIS-II detects an error in the input data:
 - 2b1. LANDIS-II reports the error to the Modeler, logs the error, and terminates the scenario.
- 2c. A programming error internal to LANDIS-II aborts the scenario:
 - 2c1. LANDIS-II reports that an internal error has occurred to the Modeler, and logs the details of the error.

3.6 Use Case 3 – Manage Scenario Data

Primary Actor: Modeler

Goal: To manage the data in a LANDIS-II model scenario.

Scope: Modeling Toolkit

Level: Summary goal (multiple user goals)

Preconditions: The Modeling Toolkit has been installed.

Basic Course of Action:

1. Modeler uses edit and generation tools to create a new LANDIS-II scenario.
2. Modeler modifies the data in the scenario using edit tools.

Alternative Courses:

- 1a. Modeler creates a copy of an existing scenario using edit tools or the file-copy functionality of the operating system.

Notes:

While both the Console Modeler and GUI Modeler will most likely use a GIS to produce input maps, they use different tools for the rest of the data in a scenario. The Console Modeler uses non-LANDIS tools to create and edit text files with input parameters. Often, she'll just use a text editor. In contrast, the GUI Modeler uses the L-II Data Tools to create and edit input parameters in a scenario.

4 Supplementary Requirements

4.1 Web Site

The project team will create and maintain a web site from which users can download the software. The web site will also provide a table (list, repository) of additional components that users may choose to add to their LANDIS-II installation.

4.2 Performance

LANDIS-II must take at most 25% more time than LANDIS 3.7 to do a simulation. For example, if LANDIS 3.7 does a simulation in 2 hours, then LANDIS-II must do the same simulation in no more than 2½ hours.

4.3 Platform

LANDIS-II must be run on Windows because it's the platform used by the project team.

Minimally LANDIS-II must be portable to other platforms in source form. This would allow 3rd parties to build binary distributions that run on other platforms. If such a party is willing, the LANDIS-II team may designate it as the maintainer for a particular binary distribution. At the team's discretion, that distribution may be available at the LANDIS-II web site, or the site will provide a link to the maintainer's web site.