

Narrative-Led and Indicator-Driven Scenario Development

A Methodology for Constructing Scenarios

Eric Kemp-Benedict

Version: 1.0

Date: 2 February 2006

© 2006 Eric Kemp-Benedict

This document may be freely reproduced and distributed, as long as this copyright notice is included.

Table of Contents

Scenarios for Sustainability.....	1
The NLIDD Methodology.....	3
General Strategy.....	3
Roles.....	3
Principles.....	4
Strategies.....	5
Phases.....	5
Initiation.....	6
Elicitation.....	6
Scoping.....	6
Evaluation.....	8
Elaboration.....	8
Finalization.....	8
Work Products.....	9
Internal.....	9
External.....	12
Tools and Methods.....	13
Project Management.....	13
Use Project Management Software.....	13
Use an Outlining or Brainstorming Tool.....	14
Find a Way to Keep Notes.....	15
Use Information Radiators.....	15
Modeling.....	16
Track Progress Using Indicators.....	16
Use Version Control.....	17
Use Diagrams to Explain Models.....	18
Use Rapid Development Tools for Modeling.....	19
Re-use Existing Scenarios.....	21
Create a Model Portfolio.....	22
Document Preparation.....	22
Establish Ownership of Documents.....	22
Use a Document History.....	22
Use Templates.....	23
Create a Document Portfolio.....	23
Use a Bibliographic Database.....	24
Use Version Control.....	24

Scenarios for Sustainability

Scenario analysis has, at this point, half a century of development behind it. During that time it has become one of the central techniques of Futures Studies, and a fixture of business, government, and military planning. It is particularly useful in situations in which the impact of current decisions is critically uncertain; that is, where the size of impacts is potentially large, but their direction is unknown, so that taking the uncertainty into account is of critical importance for planning.

Critical uncertainty is a notable feature of sustainable development strategies. Although definitions of sustainable development vary (to say the least), a common defining characteristic is that sustainable development seeks to improve people's lives without compromising the natural systems on which their livelihoods depend. This means that a sustainable development strategy acts at the intersection between three complex and poorly-understood systems – economies, societies, and ecosystems. In spite of the considerable challenges this poses, many consider the pursuit of sustainable development to be a necessity for the world today. Given the nature of the sustainability challenge, scenario analysis is a promising planning tool.

Developing scenarios for sustainable development is difficult, partly because there is little money for strategic planning for sustainable development (unlike the more glamorous business, high-tech, and military applications). Projects must be carried out using inexpensive tools under tight budgets with incomplete data sets and incomplete knowledge. Nevertheless, although the problem may be difficult, it is not impossible.

For the past few years, the author and his colleagues have been developing tools and procedures for the rapid development of scenarios. The *narrative-led, indicator-driven development* (NLIDD) approach to generating scenarios has emerged from reflections on what has seemed to work. The NLIDD approach is being continually developed. It has gone through a few rounds of development, during which some features have stayed the same, especially the core elements that give the approach its name: the process begins by creating the elements of a scenario narrative, and the modeling is driven by the identification of indicators. A further constant feature is the iterative and incremental nature of the approach. What has changed is how thoroughly the narratives and models are developed at each stage, and the role of different actors in the process.

The goal of the NLIDD approach is to produce relevant, compelling, and well-constructed scenarios when resources (financial and informational) are scarce. To be most effective, projects that use the NLIDD methodology should satisfy three criteria:

1. Scenario analysis plays a supporting role – it is not the main purpose of the project.

2. The project sponsor and team intends to produce scenario-related outputs that will be distributed to an external audience.
3. The scenarios will feature a narrative supported by quantitative illustration provided by a modeling effort.

In the NLIDD approach, quantification can be accomplished by rapidly developing custom models adapted to each specific study. It is also possible to use existing models if they are well suited, but a feature of the NLIDD approach is that the models are adapted to the needs of the study, and not the other way around.

The NLIDD Methodology

The NLIDD approach is defined in terms of *roles, principles, strategies, phases, and work products*. The work is supported using tools and procedures, but these are flexible, and not specified.

General Strategy

The general strategy of an NLIDD project is summarized in the name of the methodology: *narrative-led, indicator-driven development*. The substantive work in the project begins by eliciting narrative descriptions from experts and stakeholders. Those inputs are used to direct the initial work of the people who will develop the models and write the extended narratives. The modeling work is organized by first identifying meaningful indicators that can inform the narratives. The goal of the modeling effort is to produce scenarios for the indicators using meaningful, defensible, robust procedures. The scenario development is carried out in an iterative and incremental manner, as the narrative writers and modelers periodically present their work to the experts and stakeholders and incorporate their feedback.

A caveat: The NLIDD methodology is an approach to organizing a scenario project. It does not (and cannot) guarantee *meaningful* narratives and models. This requires knowledge and insight, and no methodology can compensate for a lack of these crucial ingredients. The goal of a methodology is to organize the scenario development effort so that the outcome can be more predictable, budgets and timelines can be more easily monitored, and communications can be more purposeful and clear.

Roles

The roles are defined in terms of “teams,” but in practice they might or might not be carried out by distinct people, and may or may not involve teams – the names are meant to be suggestive. The collection of everyone involved in the project will be referred to as the “project team.” Within the project team, the different roles are:

Experts and Stakeholders

The experts and stakeholders will (usually) be external participants who have expertise in the region or subject matter under study. They are responsible for providing inputs to the narratives and critically reviewing the outputs from the writing and modeling teams.

Writing Team

The writing team is responsible for generating the scenario narratives. Although they may not be expert in the region or subject matter, it is best to have a writing team composed of domain experts as well as scenario experts.

Modeling Team

The modeling team is responsible for developing scenario models to support the narratives. The members should be expert at developing quantitative scenario models, but may not have domain expertise.

Project Manager

The project manager (PM) typically oversees the work of the writing and modeling teams, although in some projects the writing may not entirely be under the PM's control. The PM is mainly responsible for making sure that the necessary things happen at the right time, keeping an eye on the schedule and budget.

The burden of the effort should be carried by the writing and modeling teams, which will usually be composed of internal staff. The experts and stakeholders often have other responsibilities in the project, and may not feel comfortable providing extensive written or quantitative input. Any experts or stakeholders who do feel comfortable with those tasks and are willing to contribute may join the writing or modeling teams.

Principles

The manner in which a project is carried out to some degree reflects the values of the project team, project manager, or project sponsor. The NLIDD approach attempts to make these values explicit, by listing the principles that underlie the approach:

1. The central principle is *respect for the experts and stakeholders*. These are the people who can make the project outputs relevant. They are critical to the project's success. This principle is captured in the limited tasks assigned to this set of actors, as well as in some of the other principles.
2. The writing and narrative teams should be *open to skeptical attitudes toward scenarios*. This principle is captured in part through the use of the Scoping phase, after which the project team can decline to continue with the scenario work if it wishes.
3. Regarding quantitative models, *all are invited but none are required to use models*. That is, everyone has access to the models, but no one has to use them or review them in order to participate in the project. This principle is captured in the division of roles as well as in the limited *mandatory* exchange between the modeling team and other participants in the project. (Project participants may choose to have more involved exchanges if they wish.)
4. There should be *collective agreement on the balance between relevance and legitimacy*. When the system under study is poorly understood (nearly always the case in sustainable development projects), then the most relevant statements are usually accompanied by a great deal of uncertainty. The goals of relevance and legitimacy are therefore almost always in conflict. No single project role is best suited to determining this balance. A disagreement on this point can undermine a project, so there should be a strong degree of consensus. This principle is

captured in part by having the modeling team provide information that can help the rest of the people involved assess the trade-offs.

Strategies

There are only two strategies specifically identified in the NLIDD approach. In general, any strategy that seems to be effective should be adopted. However, these two strategies drive some of the NLIDD procedures, and deserve special mention. They are:

1. Use formal presentations to convey critical information to experts and stakeholders.
2. Hold brief meetings frequently between the writing and modeling teams.

Both strategies focus on reducing the risk of a breakdown in communication. Even in a friendly environment, communication can break down easily. The experts and stakeholders are busy and in most cases are only partly engaged in the project. They may make a determined effort to keep up with the materials produced by the writing and modeling teams, but without ongoing engagement, it is difficult to convey the overall approach the teams are following. Since ongoing engagement is not usually a feasible alternative, formal presentations offer a useful substitute. Preparing a formal presentation encourages the writing and modeling teams to focus on the most critical issues, while access to the presenter gives the experts and stakeholders a chance to pose questions to the writing and modeling teams.

Brief and frequent meetings between the members of the writing and modeling teams can help to catch misunderstandings early on, and can give members of both teams more confidence in their work. This only works if the meetings are held often enough. In the (typical) situation of scarce financial resources, frequent meetings are only affordable if they are brief, but as it is easier to make them brief if they are frequent, this can be an achievable goal. Such meetings should have a consistent structure and a predictable agenda.

Phases

One important way in which the NLIDD approach satisfies the principle of “respect for the experts and stakeholders” is by making the process as smooth and simple-seeming as possible for them. Achieving apparent simplicity is labor-intensive, and there are several phases to the entire process of creating a scenario. An NLIDD project has the following phases:

1. Initiation
2. Elicitation
3. Scoping
4. Evaluation
5. Elaboration
6. Finalization

Initiation

There are two goals for the Initiation phase: to generate the Terms of Reference (ToR) for the project, and to decide on the provisional set of procedures and tools that the writing and modeling teams will use.

The Initiation phase should begin with a presentation to the internal team members by the project manager. The goals of the presentation are, first, for the team to understand the program manager's vision for the project, and, second, for the team to agree provisionally on the project goals.

After the presentation, the project manager ensures the production of the ToR. The ToR may or may not be written by the project manager, but the he or she is responsible for its production and contents.

The ToR should have two parts: detailed terms of reference through the Scoping phase, and less-detailed terms of reference for after the Scoping phase. The outputs from the Scoping phase will partly determine the rest of the ToR.

Following the initial presentation, the writing and narrative teams should decide on a provisional set of procedures to follow and tools to use. For the writing team, this may include a particular document template, bibliographic software, a style guide, rules for tracking and accepting changes, and ownership of different documents. For the modeling team, this may include the modeling tool, the use of version control, procedures for sharing documentation, and coding standards.

Elicitation

The goal of the Elicitation phase is to orient the scenario activities by eliciting brief descriptions of what could happen in the future from the experts and stakeholders. The focus is on compelling narrative inputs that draw on the participants' expertise and experience – the quantitative scenarios are not prominent in this phase.

The Elicitation phase is best accomplished through a brainstorming workshop lasting between 2-5 days . The workshop structure is not described in this document, because there are well-known strategies for developing the scenario frameworks (or "scenario logics") that help to structure the outputs from this phase.

The end product of the Elicitation workshop is the raw material for what follows. It is important to emphasize to the experts and stakeholders that nothing is "set in stone" after the Elicitation meeting, and that their outputs will not be copied uncritically into the final report. Instead, the outputs will be used to orient the scenario activities, in that the narrative and modeling activities that take place during the Scoping phase take the outputs from the Elicitation phase as a starting point. Furthermore, the writing and modeling teams consult with the experts and stakeholders throughout the scenario development process.

Scoping

The goal of the Scoping phase is to present sufficiently concrete outputs that the study participants can reasonably evaluate the role of the scenarios. At the end of the Scoping phase, the project team (which may – and should – involve some or all of

the experts and stakeholders) can decide whether to continue developing the scenarios.

Two outputs from this phase are provided to the experts and stakeholders: a narrative outline and a “thin model.” These are presented to the experts and stakeholders in a formal presentation. The presentations should include both what has been done so far and what the writing and modeling teams expect to produce by the end of the project.

The thin model is a working quantitative model that produces provisional outputs for key indicators. It is “thin” in that some indicators may not be implemented, and also that the indicators that are implemented are provisional. The calculations behind them may be considerably simpler than in the final version of the model.

There are several steps to the Scoping phase, as shown in Illustration 1. As shown in the illustration, the outputs from the Elicitation phase are used to drive the activity during the Scoping phase that results in the narrative outline and the thin model.

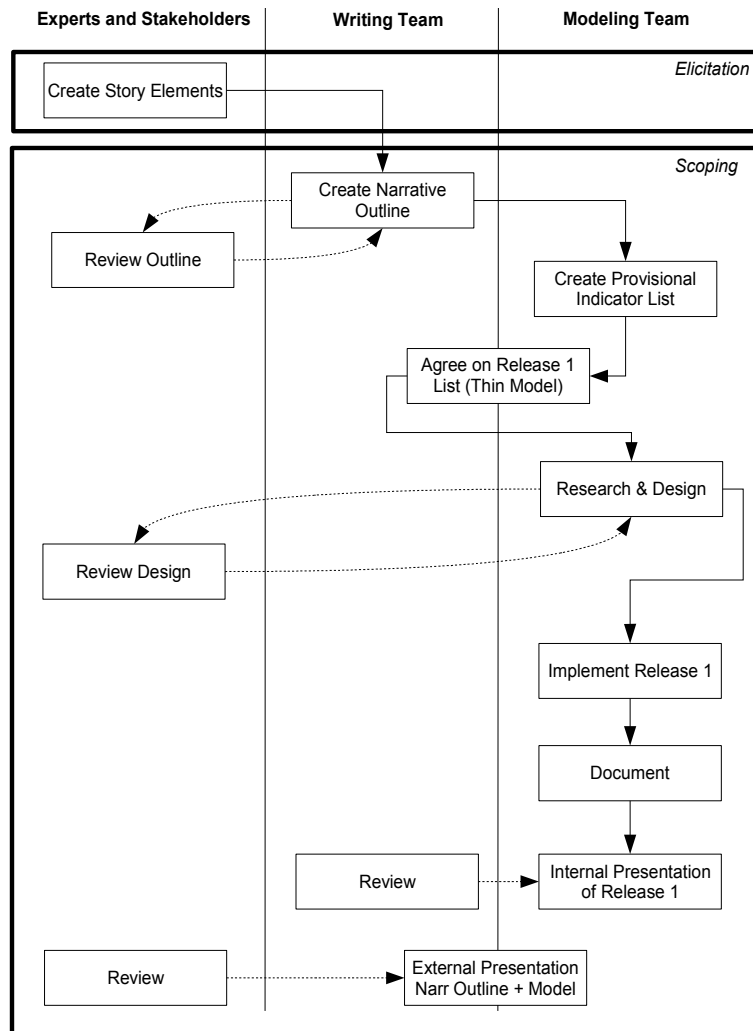


Illustration 1: Elicitation and Scoping phases

The experts and stakeholders participate during the Scoping phase by reviewing the approach followed by the writing and modeling teams. First, through interim review of the narrative outline and the the plan for implementing the indicators (the model design). These are communicated in (brief) written proposals to the expert and stakeholder groups for review. Second, at the end of the scoping phase, the experts and stakeholders review the narrative outline and the thin model during the formal presentations by the writing and modeling groups.

Evaluation

After the Scoping phase, the project team decides whether to continue with the scenarios. Although they may decide not to continue, at this point they should have some useful products: the output from the brainstorming workshop, a coherent narrative outline, and a working and documented (although very incomplete) model.

If the project team decides to continue, then in the Evaluation phase, the ToR is revisited, and revised if necessary. The revised ToR takes into account the lessons learned during the Scoping phase and incorporates the feedback given to the writing and modeling teams after their presentations at the end of the Scoping phase. After the presentations, the experts and stakeholders should have a much better idea of the role of scenarios and of their possibilities and limitations. This should make for a more robust ToR that has fewer surprises for the project team members.

Elaboration

The scenario narrative outline and the “thin model” produced during the Scoping phase are very incomplete. The narrative outline needs to be developed into a full set of scenario narratives, while the model needs to be finalized. Finalizing the model involves both quantifying indicators that have not yet been quantified, and improving on the provisional quantification of the indicators that have already been implemented.

The Elaboration phase moves both the narratives and the model towards a final state, through a sequence of iterations. Internally, between the writing and modeling teams, there should be frequent exchanges of information, for example through weekly meetings. Communications with the experts and stakeholders should be less frequent, unless initiated by the experts and stakeholders themselves. As major parts of the narratives and models are completed, key outputs should be shared with the experts and stakeholders.

Finalization

In the Finalization phase, all project outputs are prepared for release. The scenario narratives are finalized and approved, and if the model is to be distributed, then it is cleaned up and prepared for distribution.

Work Products

Several work products are produced in the course on an NLIDD project. Some are only shared internally, while others are final work products intended for external distribution.

Internal

Terms of Reference

The Terms of Reference are developed in two stages. The first Terms of Reference document covers the project through the Scoping phase in detail, with an estimate of the budget. The second Terms of Reference document is produced after the Scoping phase (if the project team decides to continue with the scenarios) and incorporates the knowledge gained through the Scoping phase activities and feedback from the presentations at the end of the Scoping phase.

Project Schedule

The project schedule in an NLIDD project is a living document. An NLIDD project can be broken down into phases, iterations, and sub-iterations, as shown in Illustration 2. As shown in the illustration, iterations and sub-iterations only appear in the Scoping and Elaboration phases.

The writing and modeling teams communicate with the experts and stakeholders in between *iterations* that might last 1-3 months (iterations are shown as shaded arrows in Illustration 2). The writing and modeling teams create their own internal deadlines in *sub-iterations* lasting 1-2 weeks (shown as small white arrows in the illustration).

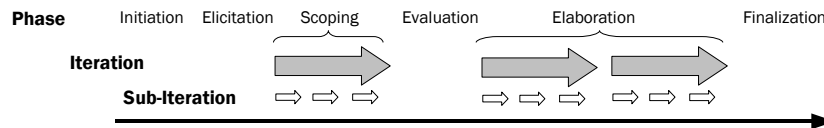


Illustration 2: NLIDD timeline with iterations and sub-iterations

During the Scoping phase, the writing and narrative teams present the narrative outline and model only at the end of the phase, so there is one iteration. However, within that iteration, the teams may have several internal deadlines. For example, these may be matched with the modules in the model, or themes in the narrative outline. Within the Elaboration phase, the model and narratives may be presented to the experts and stakeholders more than once. Between presentations, the writing and modeling teams might have several internal deadlines.

Try These	Keep These
Ongoing Problems	

As each internal deadline is met (and possibly more frequently), the project schedule is revisited by the writing and modeling teams, in a *project planning meeting* led by the project manager. In this approach, a rough schedule is created during the Initiation phase that covers the entire project. Detailed task breakdowns are created at the start of each sub-iteration.

At the project planning meeting, the writing and narrative teams should revisit the procedures (such as the use of style guides, choice of software, and use of version control) they are following to carry out the work.

One effective way to do this is to use the layout shown in Illustration 3.¹ With this technique, a poster paper is put up on the wall with sections as shown in the illustration. The team brainstorms the procedures they want to continue using

Illustration 3: Layout for project planning brainstorm

(“Keep These”), procedures they are not currently following but that might help (“Try These”), and continuing problems.

By keeping the project schedule a living document, the team keeps in frequent communications, problems are discussed, and solutions are proposed. Also, by creating a fine-grained schedule frequently, problems with the project schedule can be identified early.

Model Design Proposals

The goal of a model design proposal is for the modeling team to give the writing team, and the experts and stakeholders, enough information to evaluate the modeling approach. In particular, the proposals should provide enough information that the non-modelers can evaluate where the balance between relevance and legitimacy is being struck.

Depending on the audience, the written proposals sent to the experts and stakeholders may be extremely technical or extremely non-technical. In many cases, it is best to offer a non-technical document with a technical appendix for interested readers, in keeping with the principle that “all are invited, but none are required to use models.”

In any case, the document should be very brief, and may consist simply of an e-mail. In keeping with the principle of “respect for the experts and stakeholders,” the barriers to reviewing the model design should be very low. Also, in keeping with the principle of “collective agreement on the balance between relevance and legitimacy,” the proposals should provide information that helps others on the team assess potential trade-offs between the two goals.

¹ From Alistair Cockburn’s *Crystal Clear: A Human-Powered Methodology for Small Teams*.

Some strategies for communicating models to an audience that does not want to involve itself in the technical details are:

- Writing clear, brief, plain text descriptions
- Providing citations
- Using diagrams
- Using examples
- Providing a technical appendix
- Rating the degree of knowledge and consensus for the approach

The first and most important goal is to write briefly and clearly about the modeling approach. This text description should be backed up by citations where appropriate. This can help the experts and stakeholders to assess the quality of the background material used to devise the modeling approach. Illustrations, whether diagrams, example results, or sample calculations, can help to make the approach clear and concrete. For interested readers, a brief technical appendix might be added.

Rating the degree of *knowledge* and *consensus* about the chosen approach can be very helpful for external reviewers. A simple rating in each category (such as “weak”, “moderate”, and “strong”) can be sufficient. Note that as these ratings require judgment, the experts and stakeholders may disagree with the modeling team’s ratings. Any such disagreements should be resolved and the final agreed-upon rating recorded.

Thin Model

The thin model is a working, tested model. However, it should not try to do everything the final model will do, because it must be produced rapidly during the Scoping phase. It might be simplified in several ways:

- Some modules are not implemented
- Some indicators are not implemented
- Data are not fully disaggregated
- Some connections between modules are not implemented
- Variables that will later be calculated are set by hand

Enough of the model should be constructed that it can help communicate the form of the final model. In the presentation to the experts and stakeholders, the modeling team must make clear where the modeling activity is leading so that a decision can be made whether to continue with the scenario development.

Figures and Notes Document

A figures and notes document contains one figure per page, with explanatory text and, optionally, a data table accompanying each figure. The accompanying text interprets the figure and provides any citations or data sources required to support the figure. Such a document can provide a simple, convenient way for the modeling team to provide its inputs to the writing team. It offers the benefit that the writing

and modeling teams do not need to closely coordinate their activities. The writing team can incorporate the figures at its own convenience.

External

Final Model

The final model will most likely be released to a broad audience. It should be well-constructed, thoroughly documented, and not contain any extraneous code. In the course of developing the model, it is likely that some approaches will be tested and abandoned. The abandoned code should not be included in the final model.

Producing the final model should be undertaken with the care given to an external report. The reputation of the team rests in part on the quality of the released model.

The Scenario Report

The scenario report consists mainly of the narrative. Other important content includes an overview of the purpose of scenarios, the role of scenarios in the current project, and the conclusions drawn from the scenario exercise. The conclusions should appear in summary form very near the beginning of the report, even if they are described in detail in the body of the report.

To supplement and illustrate the narrative, the modeling team will provide figures and descriptive text to include in the report. One way to communicate this to the writing team is through the use of a “figures and notes document.”

Background Technical Reports

The modeling team should be documenting the model in the course of its work. This can take a considerable amount of time, and time should be provided for it in the schedule and the budget.

The final model documentation can, and usually should, be included as a background technical report. It is the main way to establish the technical legitimacy of the modeling effort. Even if the client does not want to distribute technical reports, the client may agree to let the team post the reports themselves on the team’s web site.

Tools and Methods

There are no required tools or methods for carrying out an NLIDD project. The ideas presented in this chapter have been useful to the author, and are offered in the hope that they may be useful to others as well.

Project Management

There are many tools available to support the project manager and his or her teams. Some of these are complete project management packages, while others are supporting tools. Consider the following suggestions if project management is creating obstacles.

Use Project Management Software

One possibility is to use dedicated project management software. Both commercial and free, open-source software (FOSS) for project management is available.² Also, both desktop and on-line (web) versions are available. All have strengths and weaknesses.

Microsoft Project

Microsoft Project is a commercial desktop product, and is the “canonical” project management software. Other desktop software packages seek to emulate some of the features of Microsoft Project. However, Project is intended for very large projects, and may be unwieldy for smaller projects.

Advantages to using Microsoft Project are that it is produced by a stable company that is likely to support the product over a long period, and that several books have been written about its use. One disadvantage is that partners may not find it affordable, limiting the degree of information exchange. A second possible disadvantage is that the software may need to be upgraded periodically, which is an ongoing cost. Continuing to use older versions of the software may not be practicable if partners choose to upgrade, or if Microsoft discontinues support for older versions.

Open Workbench

Open Workbench is a FOSS alternative to Microsoft Project. It is specifically designed to reproduce all of the functionality of Microsoft Project. However, as of this writing, the documentation is incomplete and does not seem to be consistent with the most recent release. As with Microsoft Project, it may be unwieldy for smaller projects.

² The term “free” does not mean “zero cost” (although FOSS software is usually available at no cost), but instead means “free as in freedom.” For the people producing the software, there is a large philosophical difference between “free” and “open source,” but for the consumer the differences can usually be ignored. The abbreviation FOSS is a compromise term for consumers.

DotProject

DotProject is a FOSS, on-line, project management system. It is a good choice for dispersed teams that have ready access to an Internet connection, or for an intra-net within a company. It allows project managers and their teams to keep track of clients, projects, tasks, and contacts.

GanttProject

GanttProject is a FOSS desktop tool. It offers only the basic features of Microsoft Project. It features a small download, and is well-suited to small teams. However, as of this writing, the interface is quirky, and may prove frustrating to some users.

Spreadsheet

A spreadsheet program, such as Microsoft Excel or OpenOffice.org Calc, can be a very handy tool for keeping track of projects. The layout does not need to be complicated to be effective.

Use an Outlining or Brainstorming Tool

Outlining and brainstorming tools are extremely flexible and powerful ways to organize ideas and keep a record of them. A very useful FOSS outlining and brainstorming tool is FreeMind. Although it can be a bit confusing at first, FreeMind is easy to learn. It has a few simple but useful features that make it easy to generate and manipulate hierarchical “mind maps.” Illustration 4 shows a mind map in FreeMind.

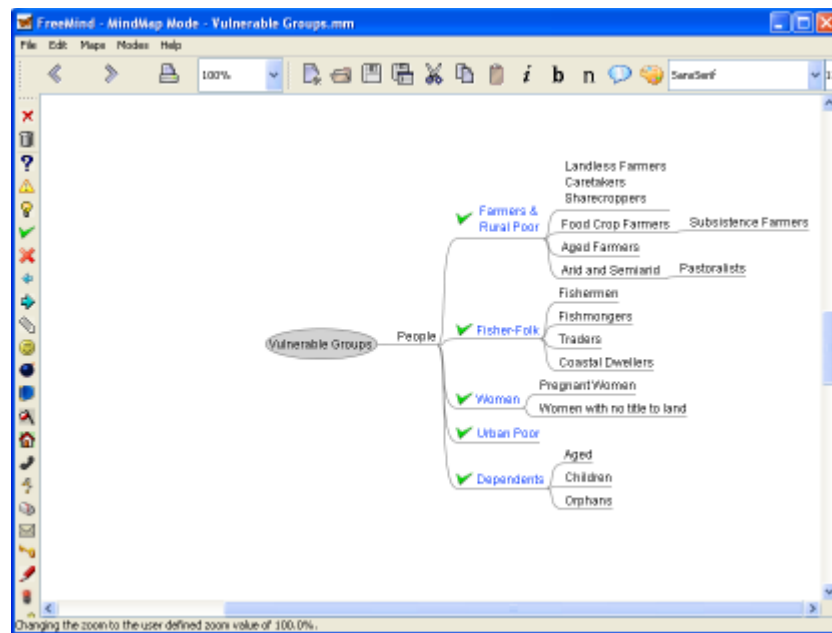


Illustration 4: FreeMind mind map

FreeMind can be used in an open brainstorming session, using a computer with a projector. One person can act as recorder, projecting FreeMind up on a wall, while

the rest of the group formulates ideas. The recorder can move elements around in a FreeMind diagram. This can be used in any situation in which hierarchical information is being used. When the group is finished, the hierarchy can be exported to HTML.

Find a Way to Keep Notes

One of the most difficult challenges the writing or modeling teams faces is restricting themselves to only doing what they are supposed to do during the current sub-iteration. It is very important that team members remain focused on the goals of the sub-iteration, but as there are many cross-cutting relationships in a typical sustainable development study, this is difficult.

One solution to this problem is to have a good system for keeping notes. A note-taking system can also lessen “cognitive overhead,” since the writer or modeler no longer needs to keep the idea in mind as they continue. For such a system to work, it must be possible to organize notes for easy retrieval later. One possibility is to use an outlining tool, such as FreeMind. Another is to use a hierarchical notepad, such as KeyNote.

KeyNote is a free (meaning “no cost” – not FOSS) editor. It stores text in Rich Text Format (RTF), and allows the user to organize notes in a combination of tabs and trees. Illustration 5 shows a KeyNote document with several tabs defined. In the illustration, the “Memos” tab is shown, with notes defined for several modules.

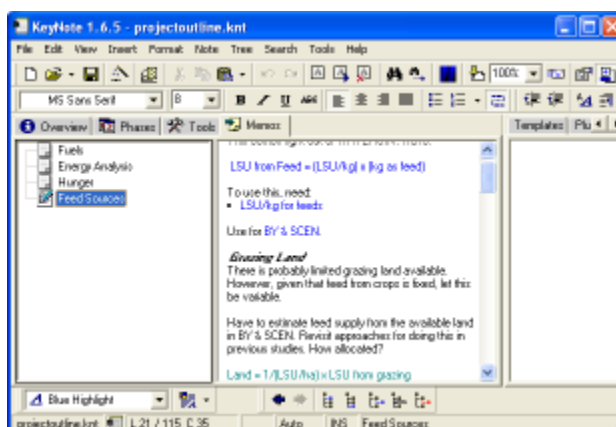


Illustration 5: KeyNote document

Use Information Radiators

An “information radiator” is a work product – a printout, a web page, a flipchart page – that broadcasts information about the status of a project.³ The idea behind an information radiator is that while people may not read or properly absorb the memos that are sent to them about projects, they will pick up the information if it is present in their environment.

³ From Alistair Cockburn’s *Crystal Clear: A Human-Powered Methodology for Small Teams*.

Illustration 6 shows an information radiator, a printout of a FreeMind document that contains a list of the indicators for a project and their current status. Anyone looking at the document – especially people who have been introduced to the format – will be able to learn the status of the project. In particular, they will be able to learn the focus of the current sub-iteration.

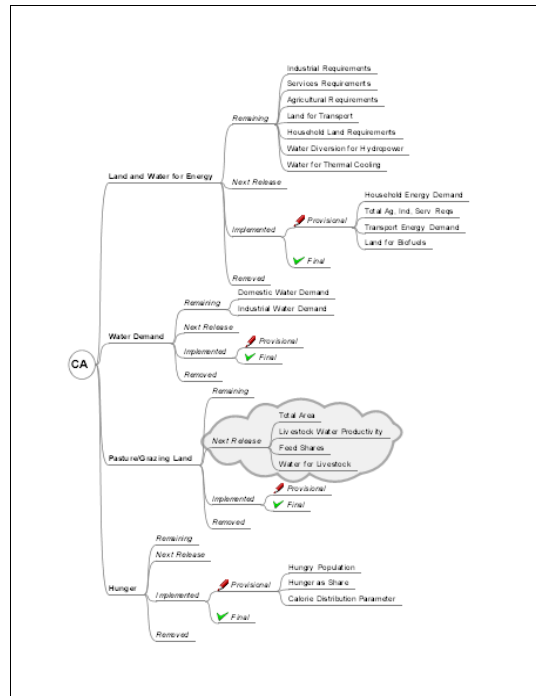


Illustration 6: Information radiator - printout of a FreeMind document

Information radiators serve at least two purposes. First, they help people keep up to date on the projects underway in their organization, even if they are not involved. Second, they reduce the number of interruptions. Unless he or she has questions about the information broadcast in the information radiator, a supervisor does not need to interrupt the staff to learn the status of their projects. It is there for all to see.

Modeling

There are several tools and methods that are particularly relevant for the modeling team.

Track Progress Using Indicators

In an NLIDD project, the modeling effort is driven by the identification of indicators. Early in the Scoping phase, the writing and modeling teams agree on a provisional list of indicators. Subsequent modeling work focuses on estimating those indicators.

Sub-iterations can be defined in terms of the indicators they will produce, and progress on the project can be tracked based on the number of indicators that have

been implemented (either provisionally or completely). This gives a convenient way to track the modeling effort.

In the Scoping phase, the goal is to produce provisional indicators. In the Elaboration phase the goal is to include all of the indicators and move the “provisional” indicators to “final” status. The status of the project can be reported as the fraction of indicators completed.

While there are many ways to track indicators, it can be done very conveniently in FreeMind, as shown in Illustration 7. When the initial list of indicators is being generated by the writing and modeling teams, they can record their decisions in the structure shown in the illustration. As they decide on the modules needed for the project, they create an indicator tree for each module. As they create the indicator list for each module, they put the list of indicators under the *Remaining* category under each module.

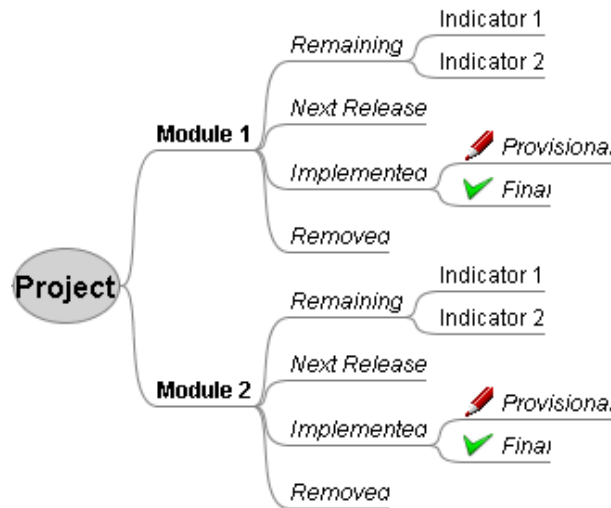


Illustration 7: Tracking indicators in FreeMind

At the start of each sub-iteration, indicators can be moved from *Remaining* to *Next Release* or (in some cases) *Removed*. After the sub-iteration, if the tasks were carried out successfully, then the indicators can be moved from *Next Release* to *Implemented*, either in the *Provisional* or *Final* branch. The FreeMind document can be printed out and posted on the wall as an *Information Radiator*, as shown in Illustration 6.

Use Version Control

A major challenge in developing a model of any complexity is to reduce “cognitive overhead,” or the need to keep many things in mind. One source of cognitive overhead is trying to keep track of changes to models, model documentation, and data. If the model changes it should be possible to both remove all traces of the

original approach from the working version of the model and to recover those changes if it turns out that the new approach is flawed.

In many cases, modelers are driven to making multiple copies of their models, saving them in archives, and commenting out pieces of code that they might need in the future. However, this only partly reduces cognitive overhead, and is error-prone. It is all too easy to open up and work with an old, deprecated, version of a model. Also, when the model is finally cleaned up for release, it is easy to delete needed code along with code that has been commented out.

These problems can be avoided by using version control. A popular FOSS version control system is Subversion (sometimes abbreviated as SVN). Although Subversion can be a challenge for new users, there are excellent books and supporting tools. Using the TortoiseSVN client in Windows, it is relatively simple to manage a project.

Use Diagrams to Explain Models

As with any computer program, it helps to create a schematic of the design before actually writing the program. In usual software development terminology, this is called “modeling” the program. When the program is itself a model, the usual term can be confusing, but the activity is very useful.

Different kinds of models lend themselves to different kinds of diagrams. Systems dynamics models have a standard diagrammatic language that is expressive enough that the diagrams can be turned into running code almost directly. Illustration 8 shows an example in the modeling software Vensim. The Vensim interface allows the modeler to diagram the model and then, by assigning values to the components in the model, go directly to running the model.

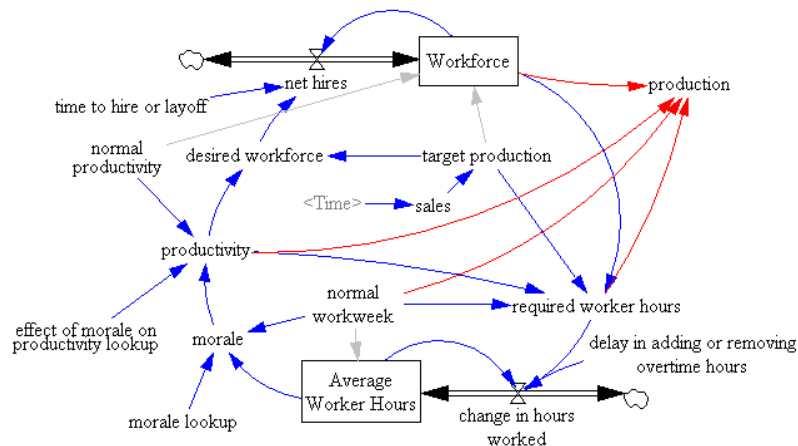


Illustration 8: System dynamics model in Vensim

Illustration 9 shows a diagram for a different kind of model that uses an accounting framework to keep track of the water and land requirements for energy production.

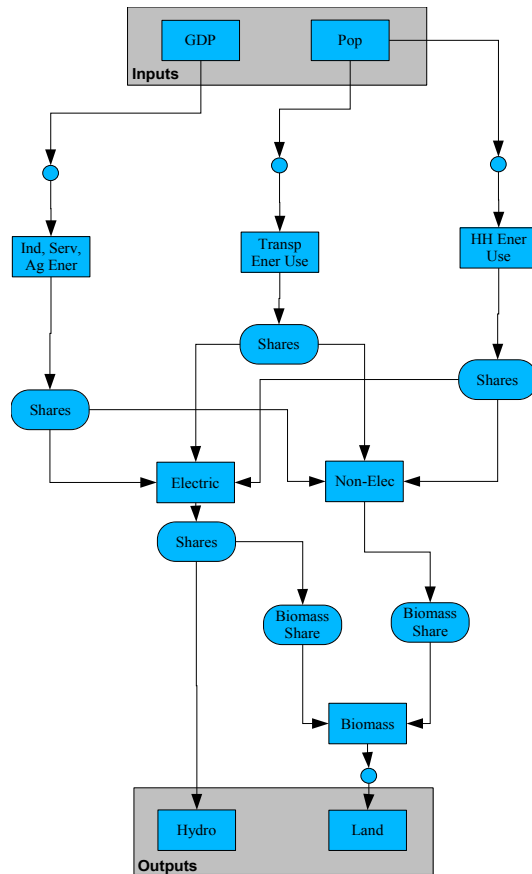


Illustration 9: Design for accounting model

By first creating designs of models, it is possible to debug the model design before the models are ever built. It is a much more efficient use of resources to work out potential problems with models before expressing them in code.

Use Rapid Development Tools for Modeling

Ideally, most of the time spent on model development should be devoted to research and model design (including documenting the design). That is, it should go into the main creative aspects of model development. Creativity is also required when expressing a model in code. However, rapid development tools try to remove the more routine and repetitive steps of model building.

Some modeling tools have been designed explicitly for rapid model development, while others can be adapted to it. Also note that one person's rapid development tool may seem very slow to someone else, and vice versa. It is best to learn something about a potential tool and, if possible, test it out in a realistic situation before adopting it. Different tools are categorized and listed here.

General Modeling Languages and Tools

- GAMS (<http://www.gams.com/>)

GAMS was designed to run quickly and allow modelers to write modeling code that looks as much like the algebraic statement of the problem as possible. Modelers can use meaningful text labels for array indices (such as “maize” or “tobacco” in a list of crops) and assign values to an array in a grid. The language is powerful, compact, and expressive.

- IPAT-S (<http://ipat-s.kb-creative.net/>)

The IPAT-S language is a modeling language specifically designed for sustainability scenarios.⁴ As with GAMS, modelers can use meaningful text labels for array indices, and the language strives to be compact and clear. IPAT-S supports algebraic models and linear programming (LP) models, which makes it suitable for many sustainability studies. The IPAT-S interpreter and supporting tools are all open source, and freely available.

- M (<http://www.m.rivm.nl/>)

From the M web site, “M is an integrated environment for the development, visualization and application of simulations of dynamic systems.” It was developed by the Dutch National Institute for Public Health and the Environment (RIVM). The language structure makes it easy to encode mathematical models that describe variables that change over time. Also, visualization tools are closely integrated with the language, so that with little effort on the modeler’s part, a user of an M model can “dig into” the model to a deep level using an easy-to-learn graphical interface. The M software translates an M model to a C-language program. The M modeler needs a C compiler to create an executable version of the model. RIVM limits the distribution of the software. Institutions can apply for a waiver, but in general the software must be purchased.

- STELLA (<http://www.iseesystems.com/index.aspx>) and Vensim (<http://www.vensim.com/>)

Both STELLA and Vensim are environments for developing systems dynamics models. They each offer a graphical interface for creating models using “drag and drop.” Both are commercial products, but Vensim has a limited (but still powerful) version available for free for educational use and at low cost for commercial use. A sample Vensim model is shown in Illustration 8.

- PoleStar (<http://www.seib.org/polestar/>)

PoleStar is a tool for developing sustainability scenarios. It offers both a database and a simple modeling language. It has an easy-to-learn graphical interface, and is a good choice for people who are not professional modelers. It is a commercial product, but the fee may be waived in some cases.

⁴ Full disclosure: IPAT-S was designed by the author of this booklet. It was designed with the NLIDD process in mind.

- Spreadsheet software
Any spreadsheet program, such as Microsoft Excel, or OpenOffice.org Calc, can be used to create models. Spreadsheet programs are also very useful for quickly trying out a modeling approach.

Special-Purpose Tools

There are many special-purpose modeling tools, such as:

- LEAP (<http://forums.seib.org/leap/>) and MARKAL (<http://www.etsap.org/markal/main.html>) for energy studies,
- WEAP (<http://www.weap21.org/>) and PODIUM (<http://www.iwmi.cgiar.org/tools/podium.htm>) for water,
- MAGICC/SCENGEN (<http://www.cgd.ucar.edu/cas/wigley/magicc/>) for climate,
- Spectrum (<http://www.futuresgroup.com/index.cfm>) for demographics and health.

Scientific Computing Packages

There are packages for general scientific computations. They may be used to create complete models, or to do supporting calculations that are later brought into a model in another modeling tool. Prominent packages include:

- Mathematica (<http://www.wolfram.com/>), a powerful commercial program.
- Maxima (<http://maxima.sourceforge.net/>), a FOSS system similar to Mathematica.
- Matlab (<http://www.mathworks.com/>), a system that focuses especially on matrix calculations but that supports a wide array of calculations.
- Mathcad (<http://www.mathsoft.com/>), a mathematical analysis program aimed specifically at engineers that features a “what you see is what you get” interface. Calculations can be embedded easily into documents.
- Scilab (<http://www.scilab.org/>), a powerful, full-featured, FOSS scientific computing system.

Re-use Existing Scenarios

Many scenario studies have been carried out by various organizations.⁵ A project can benefit from re-using the outputs from existing scenarios in at least two ways. First, it reduces the amount of work that the project team must do (reducing the budget at the same time); second, it lets the team leverage the legitimacy and context of the original project.

For example, using the IPCC SRES scenarios (available from <http://www.ipcc.ch/>) as a starting point for a scenario releases the team from repeating the many intensive

⁵ The Millennium Institute maintains an on-line list of scenario studies, classified by topic, at <http://www.acunu.org/millennium/information.html>. Their list is probably the most comprehensive one available anywhere.

consultations that went into deciding on the form of the scenarios and the key assumptions. Also, the results are more easily compared with the results of any other study based on the IPCC scenarios.

One model deserves special mention. The International Futures (IFs) model (<http://www.ifsmode.org/>) is a full global model that offers completed scenarios that may be used as they are. However, it is also a modeling tool that allows the user to change assumptions and even choose different sub-models within the larger model to reflect different assumptions about how the system under study is expected to behave. IFs offers a very flexible starting point for a wide variety of scenarios.

Create a Model Portfolio

The members of the modeling team are likely to generate several models over time. Later projects can benefit from past projects by reusing models, either in part or in whole. By organizing models into a portfolio that the whole team has access to, projects can be completed more rapidly.

Some publicly-available model portfolios are:

- The GAMS library
<http://www.gams.com/modlib/libhtml/subindx.htm>
- The WWW-Server for Ecological Modeling
<http://eco.wiz.uni-kassel.de/ecobas.html>
- The sample scripts for the IPAT-S language
http://ipat-s.kb-creative.net/IPATS_SampleScripts.html

Document Preparation

Just as with the modeling team, it is important for the writing team to have good tools and procedures. The section offers some recommendations.⁶

Establish Ownership of Documents

It is a good idea to have clear ownership of a document. This is useful for two reasons. First, people outside of the team (such as outside partners or internal administration) know whom to contact for questions about a document. Second, when the document is to be finalized, for example, by removing all traces of changes in “track changes” mode, it is clear who is responsible for finalization.

Use a Document History

Until a document is ready for final external release (and sometimes even then), it is helpful to keep a running document history. This can take the form of a table at the start of a document. An example is shown in Illustration 10. A document history makes clear what the status of the document is and the scope of the intended audience. A document history may also include the major changes between versions.

⁶ Most of these recommendations are adapted from the book *Agile Documentation*, by Andreas Rüping.

<i>Date</i>	<i>Initials</i>	<i>Status</i>	<i>Version</i>	<i>Circulation</i>
2 February 2005	EKB	Draft	0.1	Within team
8 February 2005	JF, EKB	Draft	0.2	With partners
15 February 2005	EKB	Review	1.0	Reviewers

Illustration 10: Example of a document history

Use Templates

Major word processing programs, such as Microsoft Word and OpenOffice.org Writer, make it possible to create templates – skeleton documents that provide a consistent framework for new documents. Carefully creating a set of templates up front can reduce some of the routine steps of creating new documents in the future.

Create a Document Portfolio

A document portfolio is a list of possible documents. The different documents might also be associated with document templates. There are three main reasons for using a document portfolio. First, it can be used as a checklist of work products when first developing the ToR and the schedule. Second, it provides a consistent terminology for the various work products. Third, it provides a repository for decisions made over time by the team. As the team finds different documents to be useful, they can be added to the portfolio. A possible document portfolio to support an NLIDD process is shown in Illustration 11.

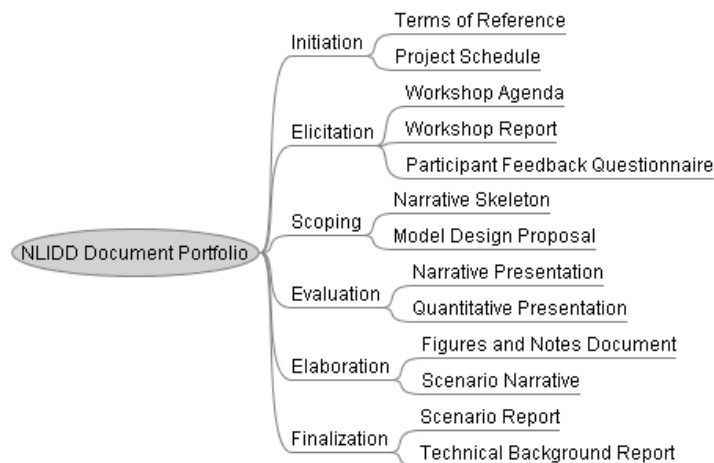


Illustration 11: Possible document portfolio for an NLIDD project

Use a Bibliographic Database

The importance of a bibliographic database for a project team cannot be overstated. A lot of time can be wasted trying to track down resources, and a lot of ground can end up being re-covered as forgotten resources are rediscovered. Also, if the team does not have a bibliographic database, then either the team must put a lot of work into preparing the bibliography or (as often happens) the bibliography is neglected, leading to a less professional – and ultimately less useful – report.

The major commercial bibliographic database is EndNote. As far as this author knows, there is no equivalent FOSS software to EndNote, which is a very full-featured program with a long history of development and use. However, there are still some very useful FOSS alternatives, even though they may not be as full-featured. One FOSS bibliographic database that is well worth looking at is JabRef (shown in Illustration 12). It has an easy-to-use interface and exports bibliographic entries in a variety of formats. Additional formats can be defined by the user. A further FOSS option is the bibliographic database built into the OpenOffice.org suite of tools. However, the interface is quirky and setting up formatting for bibliographic entries can be time consuming.

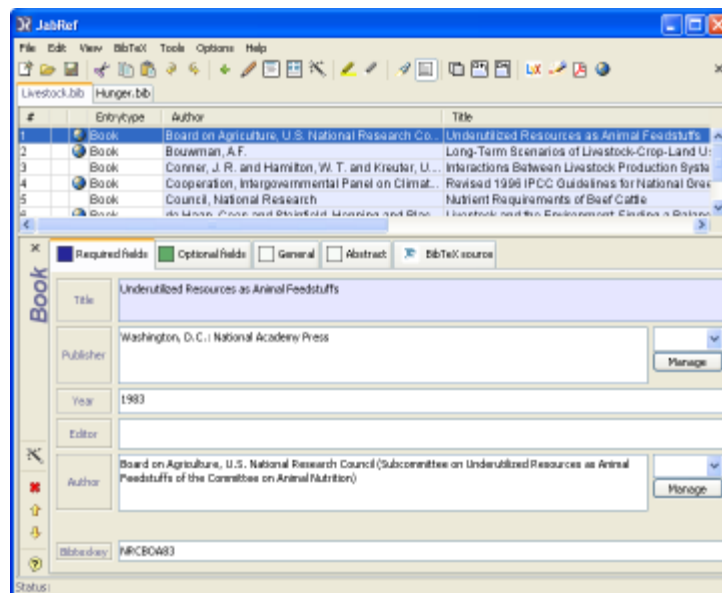


Illustration 12: JabRef bibliographic database

Use Version Control

Just as with models, using version control software for documents can reduce cognitive overhead for users. However, whether a version control package like Subversion is used or not, the special-purpose “track changes” version control feature available in both Microsoft Word and OpenOffice.org Writer should certainly be used. This mode separately tracks and highlights the contributions of different authors.