Project Title:	**Conversion of BlueSky Framework into collaborative web service architecture and creation of smoke modeling application**
Announcement for Proposals:	Requested proposal April 2008
Principal Investigator:	**Narasimhan K. ('Sim') Larkin**
Affiliation:	US Forest Service Pacific Northwest Research Station
Address:	400 N. 34th St #201, Seattle, WA 98103
Phone:	206-732-7849; Fax: 206-732-7801
Email:	larkin@fs.fed.us
Federal Cooperator:	Narasimhan Larkin, US Forest Service PNW Research Station 400 N. 34th St #201, Seattle, WA 98103 Email: larkin@fs.fed.us; Phone: 206-732-7849; Fax: 206-732-7801
Federal Fiscal Representative:	
Duration of Project:	1 year (6/2008 through 6/2009)
Start Date End Date	06/01/2008 05/31/2009
Annual Funding Requested:	
Total JFSP Funding Requested:	
Total Value of In-Kind Contributions:	

Abstract:

This project will address the need for a collaborative architecture for scientific modeling that allows various scientific models to easily interact. By designing such a system to be modular as well, advantages derived from separating decision support user interfaces from scientific models can also be realized. The need for such a system has been documented by recent studies such as the JFSP Smoke Roundtables and the JFSP review of tools done by the Software Engineering Institute.

This project addresses these needs by modifying the BlueSky Modeling Framework so that it can better serve as a collaborative architecture, and then utilizing this architecture to create an advanced application that could not otherwise be created.

The BlueSky framework will be light-weighted so that it can better serve as a collaborative architecture. Models currently contained within BlueSky will then be wrapped into stand-alone modules. These modules will further be wrapped into web-services able to be run remotely through web-service protocols on the internet. This step removes the need for local installation and solves distribution issues.

Once this is completed a unique game-playing application where a user can step by step walk through all of the model steps in the framework from fire information to smoke impact maps. At each step the user can choose the model they want to use. The resulting application will provide consistent access to all of the models in the BlueSky framework, and will allow users to "game-play" fire consumption, emissions, and smoke impact scenarios in real-time – a capability never before developed. Changes made in fire size, fuel loadings, or any other data will automatically be carried through all the remaining steps, thereby allowing users to see the implications of these changes.

This application will be useful for both RX-410 classes just learning about the various component models, as well as for decision support for managers needing to run multiple scenarios and understand the implications of various choices.

PI and Federal Cooperator:	/s/ Narasimhan Larkin	
Federal Fiscal Representative:	Not signed	

I. INTRODUCTION

1. Project Justification

Several major efforts to organize fire and smoke research have been undertaken recently by the Joint Fire Science Program. The Smoke Roundtables (JFSP, 2007) have pointed out the large number of user interfaces developed in conjunction with various scientific models, and the problems associated with the lack of compatibility between the application systems that has resulted. The Carnegie-Mellon based Software Engineering Institute's (SEI) review of JFSP tools has pointed out the need for a collaborative scientific modeling architecture to make currently diverse models inter-operable. Specific advantages of a collaborative architecture are:

1. To separate the development of scientific models from user interfaces (UIs);
2. To allow for integrated UIs capable of driving multiple models;
3. To allow for faster development of models and UIs;
4. To allow for direct comparison between models; and
5. To allow for faster transition between developed models and operational applications.

This project is designed to produce an example collaborative scientific modeling architecture based on the BlueSky Modeling Framework (Larkin et al, 2008), and to highlight the advantages of such a system through the creation of a unique game-playing application. The BlueSky Modeling Framework was identified in the SEI report as a leading example of modularizing and connecting scientific models, and a likely candidate for creation of a collaborative architecture.

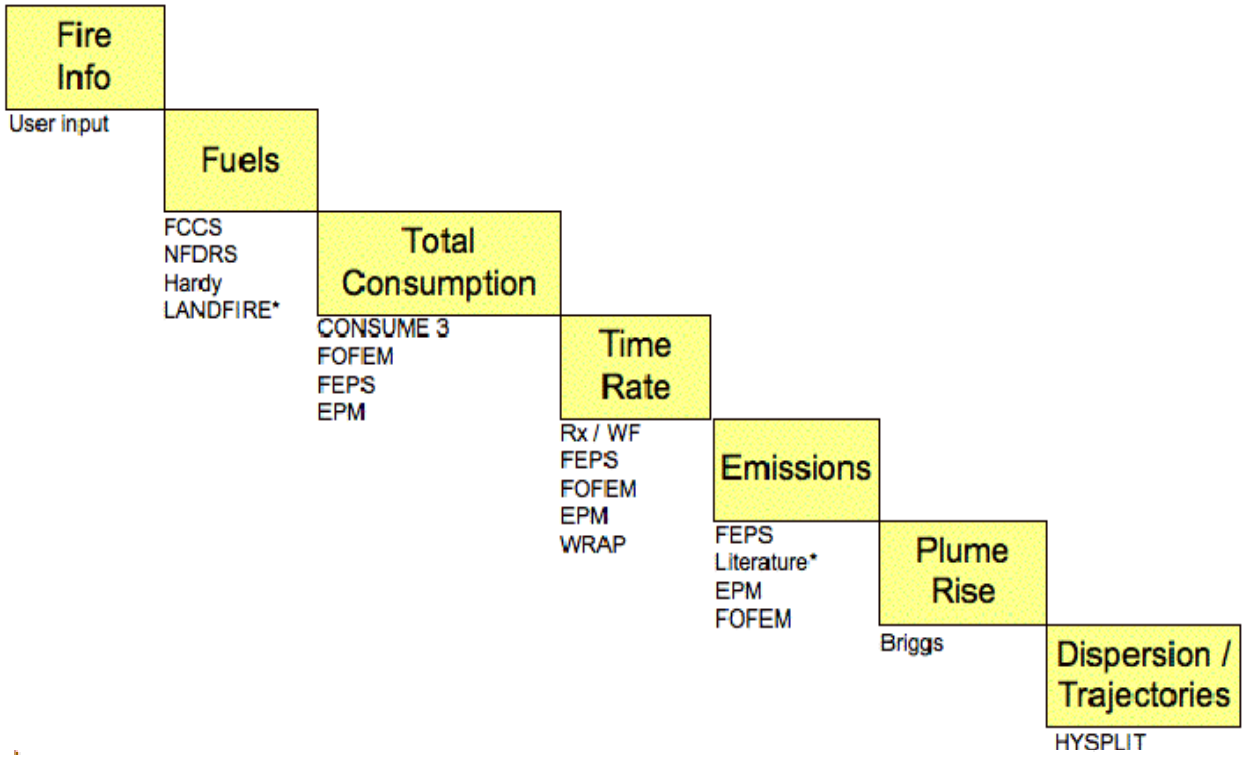

Figure 1: Modeling steps in the BlueSky framework as they will be implemented in the collaborative architecture developed here. Under each modeling step is a list of the models that will be utilized here.

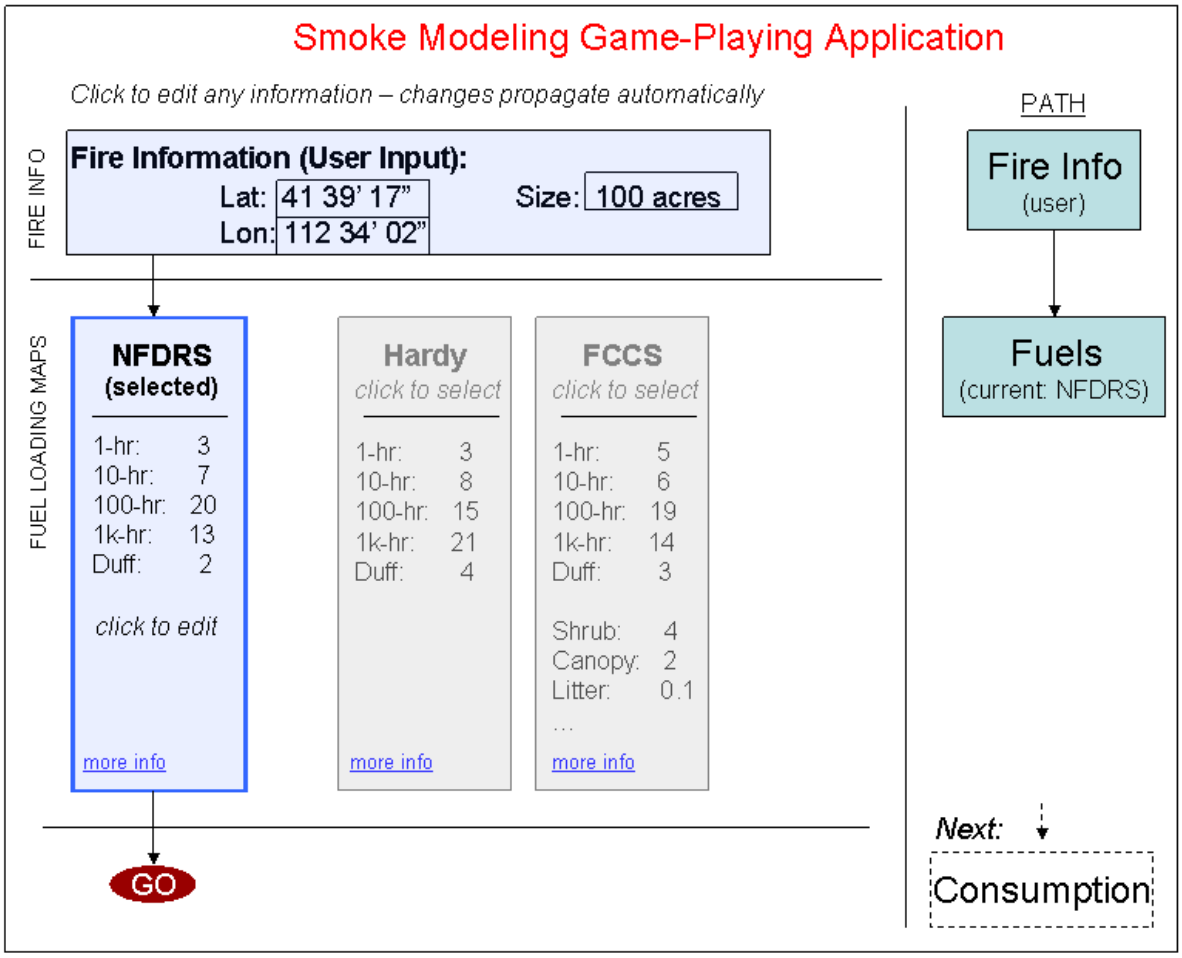

Figure 2: Crude sample drawing (for example purposes only) of prototype web application. Users would be able to select from any of the models available at each step. If a user changes information or model choice at an earlier step, information would automatically update. Users would be able to see the results of all of the model choices at each step before picking between them. The web application would use standard JavaScript/AJAX/web-service technology (similar to that used in large scale applications such as Gmail, Yahoo! Maps, etc…).

2. Project Objectives

This project has several objectives that can be grouped into two categories: those focused on the development of a collaborative architecture, and those focused on the utility of the applications created.

With respect to the collaborative architecture:

1. Modify the BlueSky Framework to lightweight it to serve as a collaborative architecture;
2. Create individual stand-alone modules for each of the models in BlueSky;
3. Wrap these stand-alone modules so that they can function through standard web-service architecture internet protocols;
4. Demonstrate the power of such an architecture; and
5. Document how to utilize such an architecture to create decision support applications.

With respect to the applications created for this proposal:

1. Create simple applications that can serve as examples;
2. Create an application that shows how both local and remote models can be combined;
3. Create a game-playing smoke modeling application useful in teaching RX-410; and
4. Demonstrate the full capabilities of this type of architecture.

3. Background

The BlueSky Modeling Framework is a software architecture that contains fire consumption, fire emissions, time profile, plume rise, smoke trajectories, and smoke dispersion models. The BlueSky Framework was recently completely rewritten under a grant from NASA making it completely modular with defined modeling steps and defined data structures between the modeling steps. BlueSky takes fire information and weather information and runs through the following steps to model smoke impacts:

- Fuel Loading
- Fire growth
- Total consumption
- Time rate of consumption
- Speciated emissions
- Plume rise
- Smoke transport and dispersion

Due to its modular nature, BlueSky incorporates multiple different equivalent models at each step, and the user can pick any of the available models at each step independently from choices at other steps. The result is that there are currently 1296 different model pathways possible within the BlueSky Framework.

BlueSky also contains core framework code that makes it easy to develop, define, and utilize new modules. BlueSky's core framework also allows for distributed processing across multiple computers, and other more advanced functionality.

This project will lightweight the core framework code in order to make it as simple as possible for others to take and utilize. (As defined in the Appendix, this will create a "Minimal Platform" version of BlueSky.) It will also wrap a number of the existing models within BlueSky into stand-alone modules that can easily be chained together. These stand-alone modules will also be wrapped so that they can be run remotely via web-service protocols on the internet.

II. METHODS

This project will require 4 things:

1. Revising the BlueSky Framework to serve as a lightweight collaborative architecture;
2. Wrapping the existing models into web-services;
3. Creating sample applications and documentation; and
4. Creating an advanced game-playing smoke modeling application.

Revising the BlueSky Framework

The BlueSky Framework currently contains features not needed for individually wrapped models (see "Full Platform" option in the Appendix). We will remove some of the most advanced features (load-balancing across various machines, etc…) and reduce the framework to its most basic function set needed

to maintain ease of use. This work, while conceptually simple, is expected to be 1/3 to ½ of the amount of effort required for this project. It is a "one-time-only" effort, with the benefits (the creation of a "Minimal Platform" option in the Appendix) being usable on an on-going basis afterwards.

Web-service versions of existing models

Each model contained within the BlueSky Framework will then be wrapped into its own stand-alone version that can be used via the command line and can also be incorporated web-service version. Web-service versions of the models listed in Figure 1 will be placed separately on specially purchased servers. A web-service allows a remote user to send the server the needed input information over the web; the server then runs the model and returns the model output. Documentation will be produced to show how both the stand-alone and web-service versions were produced, and explaining how others can incorporate their models into the new BlueSky collaborative framework produced here.

Note that because these web-service versions will utilize the standardized data structures defined in the BlueSky Framework, it will be easy to switch between various models of the same type (e.g. CONSUME, BURNUP, EPM and FEPS for total consumption) in any given application.

Simple applications and documentation

Note that alone, the web-service framework is not user friendly but requires a front-end to handle the data visualization. Fortunately, by placing the computational and database requirements on a remote server, applications can be developed that are extremely lightweight and focus only on the user interface (either as a web page or as a simple desktop program). We will create a sequence of extremely simple applications that show how this can be done, and show how an application can switch between utilizing a remote web-service model and a locally installed model (for use when not connected to the internet). Documentation will be produced to help anyone interested in building their own specialized application.

Advanced game-playing smoke modeling application

In order to show the full capabilities of this type of collaborative modeling architecture, we will create a unique, advanced, game-playing application for smoke modeling initially targeted at RX-410 classes but also useful for decision makers. An example of the application front end is shown in Figure 2.

The application would allow users to enter a fire size and location and would then lead the user through all of the steps in the BlueSky smoke modeling pathway up to and including the creation of smoke impact maps for the fire. At each step the user will be shown all of the various outputs from all of the various models (e.g. fuel loadings from LANDFIRE, FCCS, NFDRS, and Hardy) and allow them to choose one before continuing to the next step. Outputs would include: fuel loadings, total fire consumption, fire emissions, and smoke impact maps, although users could stop at any point. By seeing the outputs from all of the model choices at each step, users would be able to see the inter-model variability; by altering the model choices or information at any point, users would be able to see the effects automatically adjust any of the remaining modeling steps.

Results would be able to be shared through unique URLs that could be emailed. Variants of this application could be created by limiting the choices to a specific pathway (e.g. for specific decision support). The game-playing application may have a user-login and user preferences section.

III. Project Duration and Timeline

This project will last 1 calendar year, June 1, 2008 through May 31, 2009.

A detailed breakdown of milestones and tasks and a timeline is presented in Section VIII, Deliverables.

IV. Project Compliance – NEPA and Other Clearances

This project will conform to all required clearances, as well as all to safety, health, and legal standards.

V. Budget

VII. Science Delivery and Application

The applications created here will be delivered through the internet. As far as the overall architecture goes, this entire project can be considered to be an science delivery and application. The goal is to pave the way for the next generation of science delivery and application where science models are separated from their user inputs. This will allow the same tailored and customized user interface to be used no matter what model the user prefers. This project's output will also serve to test the NWCG's IT investment process. Documentation will be created to allow users to utilize the applications created here for RX-410 classes and decision support, as well as to allow scientists to add their own models.

VIII. Deliverables

Table 3. Deliverable, Description and Delivery Dates

Deliverable Type	Description	Delivery Dates
Website	Web-service versions of models in BlueSky Framework	Initial: 10/08 Final: 2/09
Non-Refereed Publication	Documentation for utilizing web-services to create applications	Initial: 11/08 Final: 5/09
Website	Website showing simple example applications and documentation	Initial: 11/08 Final: 5/09
Website	Game-playing smoke modeling application	Initial: 1/09 Final: 5/09
Peer-reviewed publication	Journal article or peer-reviewed GTR detailing the system and application	Submitted: 5/09
Non-Refereed Publication	Final Report	5/09
Presentation	At National Air Quality Conferences	Spring '09
Presentation	At BlueSky Stakeholders Meeting	Spring '09
Presentation(s)	At national fire conferences	Spring through Fall '09
Invited Presentation	At National Weather Service Air Quality Workshop	Fall '09
Training Sessions	Incorporated into RX410 classes where BlueSky is taught	Winter/Spring '09

Timeline Narrative:

Year 1: June 2008 – May 2009

- The application and server side work will proceed in parallel. The first applications available will be the example applications, with the full game-playing smoke modeling application to follow. Significant time is allowed for user interface and functionality improvements based on user feedback.
- Presentations and training on the application will occur in the course of other BlueSky presentations and RX-410 trainings that the AirFire routinely is involved with.

IX. Expected Benefits of the Proposal

The web-service models will:

- Serve as a prototype example of a collaborative modeling environment;
- Utilize standard data inputs and outputs;
- Be expandable with additional models and model types later; and
- Be available for other applications combining the models in different ways.

The web-based applications will:

- Show the power of having the models standardized and web-accessible;
- Provide an example for other applications;
- Provide a unique, novel game-playing application useful in teaching RX-410 smoke management classes; and
- Allow decision makers in need of smoke impact scenarios to see results in real-time.

Further, the process will serve as an example for the National Wildfire Coordinating Group (NWCG)'s IT Investment Process, although this will likely occur after the final deliverables to the JFSP.

X. Qualifications of the Investigators

The curriculum vitae of the PI, Dr. Narasimhan Larkin is attached. A summary of the key project personnel and their responsibilities are listed in Table 4.

Table 4. Personnel Involved in Project, and their Responsibility

Personnel	Responsibility
Dr. Narasimhan (Sim) Larkin (USFS)	Project Lead
Dr. Robert Solomon (USFS)	Lead on technology application and testing.
Dr. Tara Strand (USFS)	Co-Lead on testing; lead on usage documentation.
Sonoma Technology, Inc. (STI) staff - Daniel Pryden, Neil Wheeler, Sean Raffuse, etc…	Contracted to perform programming and web design.

Sim Larkin is a senior scientist with the USFS AirFire team and serves as the BlueSky Project Lead. He was the original software designer of the BlueSky framework and is co-lead of the NASA BlueSky rewrite. He has led and worked on several model intercomparison studies.

Robert Solomon is the head BlueSky modeler and has extensive experience working with numeric models of all kinds. **Tara Strand** is an air quality engineer with experience in statistical analysis of model/observation evaluations.

Other AirFire staff will contribute to testing the applications and consult on application creation, led by the personnel listed above. AirFire is a premier smoke modeling research group, with products such as the BlueSky smoke modeling framework used nationally and internationally by researchers and operational personnel.

STI is a nationally known air quality consulting company that works extensively for the EPA and state and local agencies. STI runs the EPA's AirNow program and performs operational forecasting of air quality for a number of cities and counties across the country. AirFire and STI have a lengthy history of successful collaboration. STI has developed many web applications including the EPA's successful AirNow Tech system.

XI. Literature Cited

JFSP, 2007: Smoke and air quality roundtables, research needs and assessment. Joint Fire Science Program. 16pp. Available at http://www.firescience.gov

Larkin et al., 2007: The BlueSky smoke modeling framework: design, application, and performance. *IJWF*, in review.

Appendix: Designing a collaborative system for scientific computing

This appendix is added as a primer on some of the concepts in use in the proposal. The proposal is designed to both create a prototype "Minimal Platform" option, and to create an application showing how such a system works.

A1. Introduction

Scientific models are generally created by groups working in isolation to answer questions first defined by outside circumstances, and then later refined by the scientists themselves. The result is that the models generated end up with specific input and output requirements, both in terms of data and formats, not directly compatible with other models. This creates a challenge for anyone wishing to utilize the models in combination or wishing to directly compare across models of the same class. Because of this, several negative outcomes result:

1. Comprehensive analyses comparing and evaluating similar models are not performed hindering advancement of the science;
2. Different models become coupled to different decision support tools confounding the analysis of the best scientific model with the attached user interface;
3. User choice becomes both too great (because of the number of competing decision support interfaces to learn), and too confined (because the user interface choice limits the choice of underlying model);
4. Development of decision support tools requiring combinations of models is hindered.

To overcome these issues the use of a set of standards defining a collaborative system architecture for connecting scientific models can be created and utilized. Note that such a collaborative system applies to only the computational modeling and not to the user interfaces, although implementation of such a collaborative system on the modeling side also has benefits separating models from their user interfaces.

There are many different possible ways to implement a collaborative system, each of which carries its own advantages and disadvantages. Below we examine 3 specific choices:

A. Open Standards: a minimal set of interface standards only;
B. Minimal Platform: adds a relatively light-weight set of core platform code that adds some minimal functionality; and
C. Full Platform: adds a more complete and complex core platform code with considerable functionality.

Note that each of these possibilities requires defining a set of model steps and data input / output formats. While the Open Standards option does little more, the Platform options also add in some standard functionality through shared core platform code. We therefore first discuss how to define model steps and data formats, and then discuss the platform choices.

We also discuss how such a system can be used to promote the technology transfer from research to operational application and the advantages and disadvantages of each option for this purpose.

A2. What is a "model"?

When discussing a collaborative platform for scientific modeling, the first question that must be confronted is what is meant by a "model." While scientific models are often categorized as dynamical or

empirical, here we are more concerned about the internal layers of the model. At their core, scientific models are simply complex calculators reading in some pieces of information and outputting others in order to answer a specific question. In question is the number of usefully distinguishable questions each model answers. For example, a model might read in fire location, then calculate fire growth for the day, then calculate the amount of carbon emissions from the fire for the day. Because the model has chained together these steps linearly, and because each of these steps has useful output of its own, we can consider this model to be ***"compound model"*** made up of several sub-models or ***"base models"*** that compute the individual steps (fire growth from fire location, carbon emissions from fire growth). Given this, the key question becomes what defines useful output, which brings us to how to define standard model steps.

A3. Defining model steps and interfaces

Standard model steps define the input and output points of models contained in a collaborative system. Because they define the output points, and therefore the information available from the models, they need to be defined to be maximally useful, both from a scientific as well as an applied point of view. Therefore, in creating a list of modeling steps both user groups as well as scientists should be consulted. This is most easily done by first examining the known application uses and then translating them into specific scientific modeling steps. For example, in the wildland fire area the National Wildland Fire Coordining Group, representing users, can define the information needed for specific decision support applications, and then knowledgeable specialists and scientists can further refine this information into the smallest useful steps – essentially, the outputs of the base models in the field. One example can be found in the BlueSky Smoke Modeling Framework, which encompasses multiple applications – fire consumption, fire emissions, fire smoke impacts, and more – by subdividing these into the specific step questions (see Figure 1):

- How much does the fire grow each day? (fire growth)
- How much fuel does the fire consume each day? (total consumption)
- How much fuel does the fire consume each hour via flaming? Via smoldering? (time rate consumption)
- What speciated emissions and heat does the fire produce each hour? (emissions)
- Where does the smoke go vertically? (plume rise)
- Where does the smoke end up, and what are the concentrations? (trajectories and dispersion)

Note that some of these steps are directly defined by applications (e.g. total consumption, smoke concentrations), while others, even though they are needed for later calculations, are more scientific in nature (e.g. plume rise). It is worth noting that the list of modeling steps does not mean that all models must produce these outputs – some models may be able to bypass some of the steps by using novel parameterization schemes. In the case of the BlueSky Framework this is most easily seen in satellite estimates of fire emissions that go directly from overall fire information to fire emissions without needing to go through the intermediate steps first.

By choosing the list of model steps judiciously, they can define a list of outputs usable for a large number of known and not yet considered applications, but not be so many that as to become unwieldy. Reducing the total number of modeling steps can also be accomplished by recognizing that the data at each step can contain both standard (required) as well as extra (optional) information.

Note that defining model steps by definition defines the location of interfaces between the steps (Figure 2). In fact, it is these interfaces that are the most important, as a given model can skip interfaces but must end at some interface. Data formats are then defined for each interface.

A4. Defining data formats

One of the most important steps in creating a collaborative platform is to define standards for data at each model step interface. The data format standard is the same for both input and output at this interface. The data formats are best thought of a an interface standard – if you end up producing output at this level it will look like this, and if your model needs input from this level it will receive it the same way. Creating such an interface is done on several levels: unit standards, file formatting, naming conventions, and required and optional fields. In most cases, only one standard should be created for each model step level (e.g. total fire consumption) and this standard should be used both for models producing this information (as output) as well as for models requiring this information (as input).

The simplest decision, although an important one, is units. Generally this is as easy as choosing metric or english units, and in general it is better to choose to use metric units for scientific models. One important note is that this is the unit convention for the model steps as used by the models, and not necessarily for the user interface. Lots of users will prefer english or even non-standard units – it is the job of the user interface to translate the data input in these units to the model standard format, and to translate the model output format into the graphs, tables, and other display functions critical to making the information useful to the user.

File formats are a more complex choice. The choice generally boils down to 3 options: human readable and editable (e.g. comma separated value files), human readable and computer editable (e.g. XML), and computer readable and editable (e.g. binary formats like NetCDF). In order to maximize the utility of the system it is best not to generate a new standard, but to adopt existing standards whenever possible. Comma separated value (CSV) formatted files are universally readable by everything from simple text editors to Microsoft Excel, and standard utilities exist for every programming language to read in such data. XML is a common standard for internet applications, and retains the advantage of being human readable, but is difficult to human edit due to its complexity and sensitivity to small formatting errors. XML is capable of representing relational data, however, that is more difficult in simple CSV format. Binary formats like NetCDF allow for large quantities of data to be compactly stored and quickly retrieved, but require reading and editing via specialized programs. Experience with the BlueSky Framework has shown that even within the same collaborative system, different data structures require the use of different file formats. For example, BlueSky uses CSV for simple data (fire location, total consumption, etc…), and binary NetCDF for gridded model data (weather model and dispersion model output).

Naming conventions are needed primarily for the fields within the data files, so that different models label the same output the same way (e.g. as “total_consumption,” not “Consumpt., Total”). Once a list of required and optional fields is generated, naming conventions are generally relatively simple, and only require dissemination.

Having both required and optional fields defined for each model step allows for incorporation of new model developments, and allows for model connections in ways that are beneficial when connecting specific models but are not universal. This generally works as follows: a model will utilize the optional data when available, but only needs to have the required data to run. An example would be requiring the fuel loading data standard to have a total duff measure, but allowing it to optionally have multiple duff layer information. A consumption model can then utilize the multiple duff layers if available, but will fall back on default behavior if only the required total duff information is available.

A5. Open Standards option

The Open Standards option for creating a collaborative system is little more than the creation of the standards for the model step interfaces and the definition of the data information and formats at each interface (Figure 3a). Yet implementation of this system like all collaborative standards, also requires one more factor: an authorizing entity. In other collaborative systems the interfaces are defined by organizational consortia that then have processes for maintaining and modifying the standard over time. An example is the HTML standard, used by all web designers and by all internet browsers, which is maintained by the World Wide Web Consortia (http://w3c.org). As new tags are needed or old ones become outdated this standard is then modified.

A major advantage of the Open Standards system is that it is extremely simple, and places the least requirements on modelers and others. All that need be done is for model developers to utilize the standard input and output data structure interfaces, thereby allowing others working to the standard to use the resulting model output directly.

The primary disadvantage of the Open Standards system is that it does not help the model user to chain together multiple models. The user must create the logic that takes the output file from one model and feeds it to the next model. Note that this step is vastly simpler than it would be without the Open Standard, however, as without the standard the user would also have to have knowledge of how to process and reformat the data before it to the next model. While such processing is conceptually simple, it is actually a huge deterrence in practical applications, and overcoming it is the primary benefit of adopting a collaborative system.

A6. Minimal Platform option

In this option, in addition to the standards and authorizing entity utilized in the Open Standards option, a set of core computer code – a platform – is also employed (Figure 3b). Models are wrapped in a way that accesses this core computer code to perform certain functions.

The advantage of this option is that the core platform can perform several useful functions. In particular, the following can be embedded in the core platform: (1) a sequence of utility functions for reading in and writing out the standard data formats, (2) a sequence of utility functions for carrying out certain common functions (for example, timezone and date handling), and (3) utility code for chaining together models. The result is a system that allows for faster development of new models and model chains because the researcher does not need to generate as much code on their own.

The disadvantage is that interfacing with this platform code can place hidden requirements on the model developer, particularly with respect to choice of programming language and coding style. The result of such requirements can be to discourage some model developers from adopting the standard. In the minimal platform option approach, however, efforts are made to limit the scope of the platform in order to reduce these inherent requirements.

A7. Full Platform option

In this option, a “heavy-weight” platform is utilized, in that a much larger platform codebase is developed with more extended and specialized functionality (Figure 3c).

Advantages of such an approach vary based on exactly what functionality is embedded in the platform. These will typically include: easy, standardized, and unified configuration processes; bundled software installers; load-balancing across multiple processors and computers for faster run-times.

The disadvantage of this type of approach is that the larger codebase will require more and more in terms of assumptions about the computer system being utilized, such as dependence on specific installed libraries, etc… The result is an even greater demand on the model runner and the potentiality for complex errors beyond the ability of the typical researcher to easily handle.

It is worth noting that the current BlueSky Framework (v3) is of this type. The BlueSky Framework includes hooks for cross-computer load-balancing, a unified configuration structure for all of the models contained within it, and a simple text file for choosing which models to run and outputs to produce.

A8. Applications for technology transfer

The approaches discussed here have several advantages for technology transfer stemming from the separation of user interfaces from the model computations, and the use of standard model interfaces. Because of these facts, user application development can happen separately from model development. This means that fixes and improvements to the user application and the model can happen on separate timelines, and that the best user application and the best model can be identified independently.

Models can be developed with user applications built-in. Indeed, the collaborative platforms discussed here work best if the underlying scientific models are written to be run via the command line. The model is then easily wrapped in the platform (if using one of the platform options), which also enables running the model locally in conjunction with others, and remotely by automatic wrapping of the model in a web-service (Figure 4).

Thin clients can be developed that focus on information input and display, not on running the models, because that is taken care of by the platform (Figure 5). The thin client architecture can rely on a web-server farm hosting the models as web-services, or can point to local installations of the models.

Utilizing a platform that automatically creates web-services out of the models offers the possibility of faster technology transfer because the new or revised model code does not need to be pushed to local machines. This means more uniformity because of the lack of versioning issues, and faster availability of fixes and updates. Use of web-services also means that thin clients can be developed as web pages using now standard JavaScript and AJAX functionalities. Examples of this type of application are found many places on the web including web-based mail applications like Gmail and Yahoo! Mail.

Figure 2: Example of model steps and interfaces. Defining standard model steps also defines standard interfaces. Not all models need stop or produce output at each interface step (e.g. M4 and M5). Because of the standard interfaces, combinations of models (e.g. M1 + M2 + M3, M4+M3, M1+M5) are easy. Also any application that uses one combination (e.g. M1+M2) can be easily switched to use a different equivalent path (e.g. M4).

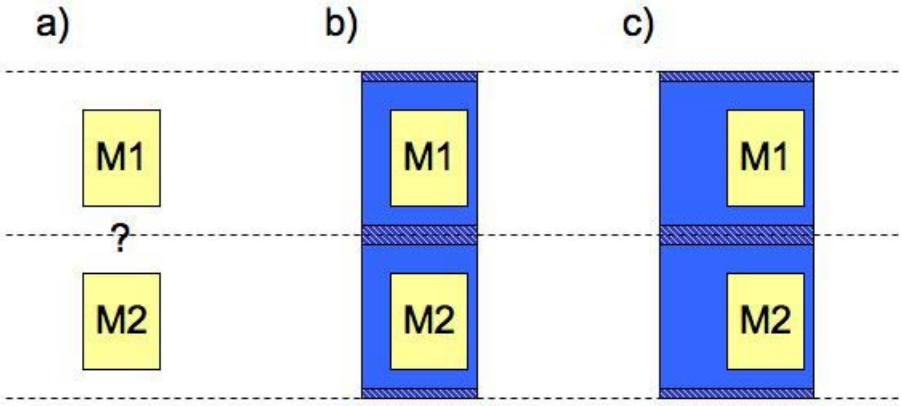

Figure 3: Example of the 3 collaborative system options. A) Open Standards. Models start and stop at defined interfaces and use standard format files, but chaining models together is left up to the user. B) Minimal Platform. Models are wrapped with a standard core code that also allows for chaining the models together more easily. C) Full Platform. Models are wrapped in a larger platform code with enhanced functionality. See text for advantages and disadvantages.

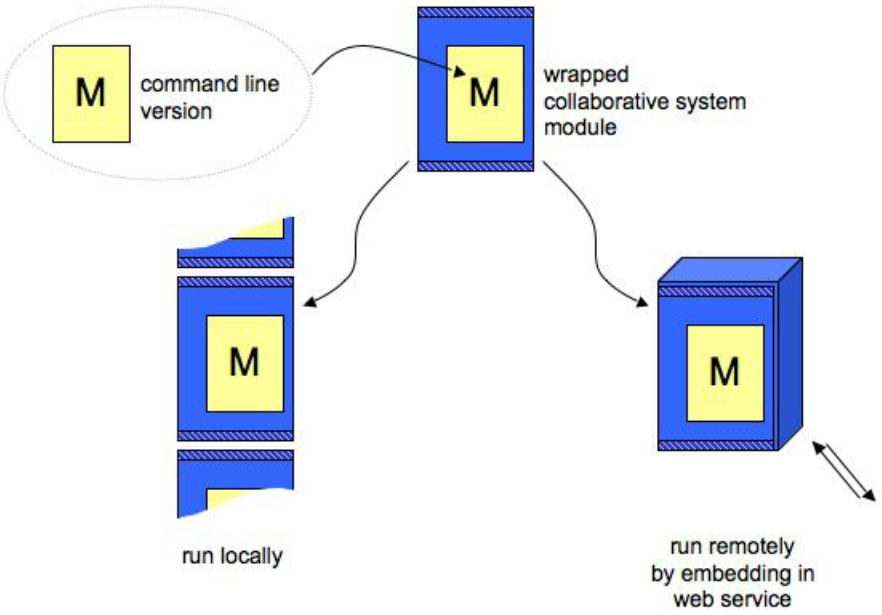

Figure 4: Multiple implementations of a model ("M") enabled by use of command line version. Model can be easily incorporated into collaborative platform module, which then enables running locally (alone or chained with other models), or running remotely by putting on the web as a web service. Model developer need only develop the one command line version to enable all uses.

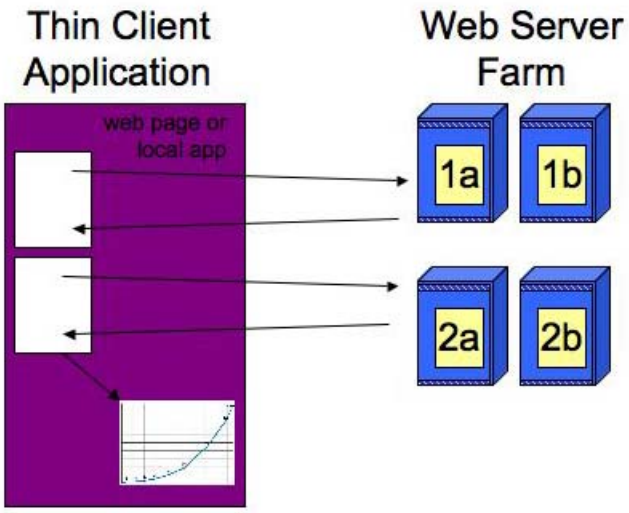

Figure 5: Functioning of thin client applications. Thin client relies on models installed on web farm as web services to do computations, but then interprets results from model output and creates display for user. Thin client can be either a web page or a local application. Thin client can also be redirected to point at local machine when web server farm is unreachable. Because of the use of standard interfaces, any available model can be used for each step (e.g. model 1b instead of 1a or 2b instead of 2a).

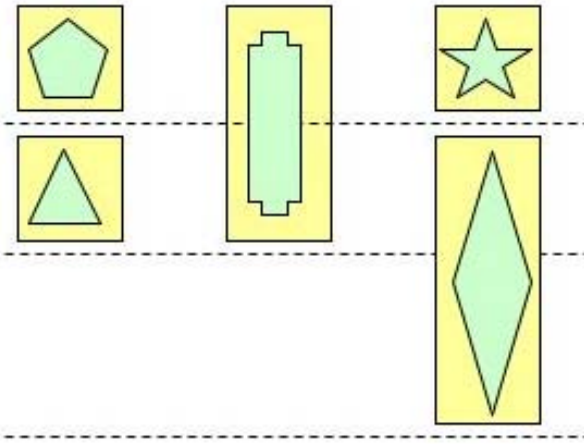

Figure 6: Wrapping existing models to make them work in a collaborative system. Existing models can be wrapped to make them compatible with newly defined standard interfaces.

Narasimhan K. ('Sim') Larkin

USDA FS / PNW / AirFire
400 N. 34th Street, Suite 201
Seattle, WA 98103

Email: larkin@fs.fed.us
Phone: 206-732-7849

EDUCATION

Ph.D.	Climate Diagnostics (School of Oceanography) University of Washington, Seattle, Washington	2000
B.A.	Physics (w/High Honors) University of California, Berkeley, California	1991

HONORS

National Fire Plan Excellence in Research Award, 2005
(BlueSky Modeling Consortium)
USDA Forest Service Merit and Excellence Cash Awards, 2001- (6 total)
NOAA PMEL Outstanding Scientific Paper Award, 1996
NDSEG Graduate Research Fellowship, 1993–1996
NSF Graduate Fellowship, 1993 (declined in favor of NDSEG Fellowship)
Phi Beta Kappa

PROFESSIONAL EXPERIENCE

2001-present	Research Physical Climatologist AirFire Team, PNW Research Station USDA Forest Service
2006-present	Deputy Team Leader, AirFire Team
2004-6	Interim & Acting Team Leader, AirFire Team
2005-present	BlueSky Smoke Modeling Project Lead
2005	NATO Advanced Studies Institute, Gallipoli, Italy
2000-2001	Post-doctoral Fellow, JISAO, University of Washington
1997	NATO Advanced Studies Institute, Les Houches, France
1992-2000	Research Assistant, University of Washington
1991-1992	Engineer, Center for Particle Astrophysics, University of California
1987-1991	Undergraduate Research Assistant, Lawrence Berkeley Laboratory

PROFESSIONAL AFFLIATIONS

Sigma Xi, 2007–present
International Association of Wildland Fire (IAWF), 2006–present
International Association of Landscape Ecologists (IALE), 2006–present
American Statistical Association (ASA), 2004-present
American Geophysical Union (AGU), 1993-present
American Meteorological Society (AMS) 1993-present

PROFESSIONAL SERVICE

Co-chair, 7th Fire and Forest Meteorology Conference (AMS), 2007
Special session co-organizer, Core Fire Science Caucus (US NFP), 2007
Session co-organizer, 2nd Fire Behavior and Fuels Conference (IAWF), 2007
Session chair and assist. co-chair, 6th Fire and Forest Meteorology Conference (AMS), 2006
Session chair and assist. co-chair, 5th Fire and Forest Meteorology Conference (AMS), 2004
AMS Fire and Forest Meteorology Committee, 2003-present
Co-chair, BlueSky Annual Meetings, 2002-present (USFS, 5 meetings)

RECENT SELECT PRESENTATIONS

(136 total / 74 personally presented / 42 invited)

2007 7th Forest and Fire Meteorology Conference, Bar Harbor, Maine (conference co-chair)
2007 2nd Fire Behavior and Fuels Management Conf., San Destin, Florida (session asst. org.)
2007 Canadian Smoke Forecasting Workshop (invited), Edmonton, Alberta
2007 National Air Quality Conferences (invited), Orlando, Florida

SELECT REFEREED PUBLICATIONS

(36 total publications/ 19 refereed)

Larkin, N.K., S.M. O'Neill, R. Solomon, C. Krull, S. Raffuse, M. Rorig, J. Peterson, and S.A. Ferguson (2007), "The BlueSky smoke modeling framework." *International Journal of Wildland Fire,* (in review)

O'Neill, S., **N.K. Larkin**, J. Hoadley, G. Mills, J.K. Vaughan, R. Draxler, M. Ruminski, and S.A. Ferguson (2007) "Real time smoke predictions" IUFRO book chapter (accepted, in press)

Riebau A., **Larkin N.K.**, Pace T., Lahm P., Haddow D., Allen T., Spells C. (2006) "The 2005 BlueSkyRAINS-West demonstration project: final report." USFS PNW Research Station, Portland, OR. pp.38.

McKenzie, D., S.M. O'Neill, **N.K. Larkin**, and R.A. Norheim. (2006). "Integrating models to predict regional haze from wildland fire." *Ecological Modeling*, 199, 278-288.

O'Neill, S., J. Hoadley, S. Ferguson, R. Solomon, J. Peterson, **N. Larkin**, R. Peterson, R. Wilson, and D. Mahany (2005). "Applications of the BlueSkyRAINS smoke modeling system." *Journal of the Air and Waste Management Association*, September 2005, 20-23.

Larkin N. K., and D. E. Harrison (2005). "On the definition of El Niño and associated seasonal average U.S. weather anomalies." *Geophys. Res. Lett.*, 32, L13705, doi:10.1029/2005GL022738.

Larkin N. K., D. E. Harrison (2005). "Global seasonal temperature and precipitation anomalies during El Niño autumn and winter." *Geophys. Res. Lett.*, 32, L16705, doi:10.1029/2005GL022860.

Harrison, D.E., and **N.K. Larkin.** (2002). "Cold events: anti-El Niño?" In *Cold Events*, ed. M. Glantz, United Nations Univ. Press., Tokyo, Japan, pp. 237-241.

Larkin, N.K., and D.E. Harrison. (2002). "ENSO Warm (El Niño) and Cold (La Niña) event life cycles: ocean surface anomaly patterns, their symmetries, asymmetries, and implications." *J. Climate*, 15, 1118-1140.

Larkin, N.K., and D.E. Harrison. (2001). "Tropical Pacific ENSO cold events, 1946-1995: SST, SLP and surface wind composite anomaly patterns." *J. Climate*, 14, 3904-3931.

Harrison, D.E., and **N.K. Larkin.** (2001). "Comment on Smith et al. (1999) 'Comparison of 1997-98 U.S. temperature and precipitation anomalies to historical ENSO warm phases.' " *J. Climate*, 14, 1894-1895.

Murphy, P.P., Y. Nojiri, D.E. Harrison and **N.K. Larkin.** (2001). "Scales of spatial variability for surface ocean pCO_2 in the Gulf of Alaska and Bering Sea: toward a sampling strategy." *Geophys. Res. Lett.*, 28 (6), 1047-1050.

Harrison, D.E., and **N.K. Larkin.** (1998). "Seasonal U.S. temperature and precipitation anomalies associated with El Niño: Historical results and comparison with 1997-98." *Geophys. Res. Lett.*, 25 (21), 3959-3962.

Harrison, D.E., and **N.K. Larkin.** (1998). "El Niño-Southern Oscillation sea surface temperature and wind anomalies." *Rev. of Geophys*, 36 (3), 353-399.

Harrison, D.E., and **N.K. Larkin**. (1997). "Darwin sea-level pressure, 1876-1996: Evidence for climate change?" Geophys. Res. Lett., 24 (14), 1779-1782.

Harrison, D.E., and **N.K. Larkin**. (1996). "The COADS sea level pressure signal: A near-global El Niño composite and time series view, 1946-1993." J. Climate, 9 (12), 3025-3055.