An Integrated Workflow Architecture for Natural Hazards, Analytics and Decision Support

James Hilton, Claire Miller, Matt Bolger, Lachlan Hetherton, Mahesh Prakash

An Integrated Workflow Architecture for Natural Hazards, Analytics and Decision Support

James Hilton, Claire Miller, Matt Bolger, Lachlan Hetherton and Mahesh Prakash

CSIRO Digital Productivity Flagship
james.hilton@csiro.au

Abstract. We present a modular workflow platform for the simulation and analysis of natural hazards. The system is based on the CSIRO Workspace architecture, allowing transparent data interoperability between input, simulation and analysis components. The system is currently in use for flooding and inundation modelling, as well as research into bushfire propagation. The modularity of the architecture allows tools from multiple development groups to be easily combined into complex workflows for any given scenario. Examples of platform usage are demonstrated for saline intrusion into an environmentally sensitive area and image analysis inputs into a fire propagation model.

Keywords: simulation · workflow · natural hazard

1 Introduction

Computational simulations of natural hazards such as floods and fires can now be performed with high accuracy at speeds greatly exceeding real-time. Analytics from such models can give valuable prediction information to decision makers in the event of a crisis, aiding decision support, or allow detailed risk assessments to be performed on vulnerable regions. Such natural hazard models have specific architectural needs, such as the ability to handle large data sets, run complex computational methods and perform detailed ensemble analysis over the results of these models.

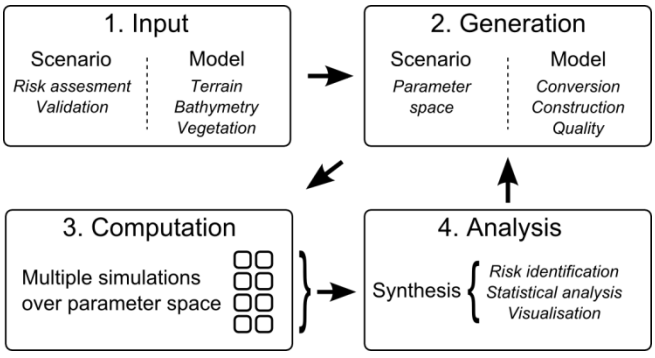

Fig. 1. Stages in modelling a natural hazard event.

adfa, p. 1, 2011.

A typical workflow in a natural hazard modelling process is shown in Fig. 1. This can be subdivided into a series of stages, which are:

1. Determination of the requirements for the given scenario, such as prediction of risk or impact assessment. The requirements for the scenario decide the input data requirements for the model, as well as the parameter space over which the model will be run. This input data may need to be gathered from local or remote data repositories and may be a static or real-time data source. For natural hazards this data may include items such as digital elevation models, wind predictions or measurements, precipitation levels, bathymetry measurements or land usage maps.

2. Generation of formatted data suitable for the model. The data is quality checked, cleaned, processed and converted into a format required for the model. For geophysical data this stage may involve processing such as filling missing values in data sets or geodetic re-projection into a common datum or format. The required parameters or sample space for the model are also generated during this stage. Examples of such parameter spaces include ranges of probable tidal extents for inundation scenarios, or the degree of variation in fuel load in bushfire scenarios.

3. Simulation of the given scenario over the parameter space using the cleaned, formatted data sets. One instance of an underlying computational solver is usually run for each element in the parameter space. These instances may be run sequentially or in parallel on local or remote compute nodes. Each instance requires the ability to read and write large data sets to storage which may be local or remote relative to the instance.

4. Synthesis and analysis of the multiple raw data items produced by the computation. This can be an automated or manual processes depending on the scenario, such as risk identification from spatial maps, statistical ensemble analysis or visualization of the results. In some applications, such as real-time scenarios with rapidly changing input conditions, the results from the final stage can be used to inform the choice of parameter space for subsequent simulations of the scenario.

The requirement for an end-to-end solution for these stages has led to the development of an integrated platform for modelling natural hazards. The aim of this platform is to provide all of this functionality in a user-friendly and rapidly configurable environment for different types of natural hazard. Once a particular scenario is defined, and the corresponding workflow has been constructed, the workflow can be run on multiple operating systems and hardware types, ranging from desktop systems to GPU based supercomputers. If required, interaction with the workflow can be exposed through a custom graphical user interface. The architecture behind this modular

framework for natural hazards is outlined in the following section, followed by examples of two current use cases.

2 Software Architecture

A schematic diagram of the architecture behind the framework is shown in Fig. 2. The natural hazard solvers sit within the CSIRO Workspace environment, which is based on the Qt framework. There are currently two operational natural hazard solvers: SWIFT, which is a hydrodynamic modelling tool for inundation and flooding and SPARK, which is a fire perimeter propagation tool.

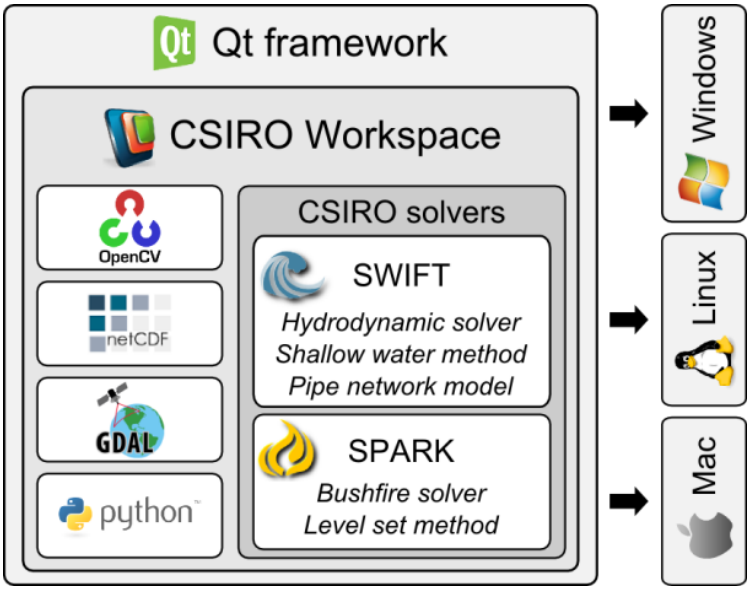

Fig. 2. Architecture of the CSIRO Workspace and natural hazard solvers.

CSIRO Workspace is a cross-platform, extensible workflow and application development framework which handles three main elements; data objects, operations, and user interface elements or widgets [1]. As well as natural hazards, Workspace has been applied to a number of other scientific areas including quantitative imaging, additive manufacturing and industrial process modelling.

In Workspace, workflow stages are encapsulated into individual Workspace operations with multiple input and output data objects connected together into a dependency graph. Data is passed in-memory between operations, allowing efficient, transparent communication of data through the system. This data can be inspected and interacted with at any stage, including during execution, with various data-type specific widgets. The plugin architecture exposes data types, operations and widgets to the framework through shared libraries. Workspace plugins are shared libraries (.dll, .so

or .dylib) which act as containers, exposing components to the framework. The Workspace editor includes code generation wizards to create stub code for new components and utilizes CMake for cross platform compilation. Development of complex user interfaces is supported with easy drag and drop designing within Qt Designer allowing any input or output throughout a workflow to be visualized or controlled from a simplified, abstracted user interface. Workspace provides a large library of useful data types, operations and widgets out-of-the-box, such as image manipulation, visualization, mesh processing, database support and plot generation. Workspace itself is written predominantly in C++ for performance and scalability and leverages the full power of the Qt toolkit providing a library of underlying capabilities for developers to leverage such as threading, file IO (XML, JSON etc.), networking, image processing, and 3D visualization using OpenGL.

A number of open-source modules have been exposed within the framework. These include OpenCV for image analysis, NetCDF for large data file support and GDAL for geospatial datasets (including remote OGC WMS and WCS web-services). Scripting is included in the form of ECMAScript and Python for non-C++ developers and calls to the statistical package 'R' are also available. Parallel APIs, such as OpenMP, are fully supported and the current bushfire and flood solvers both use OpenCL for GPU computation. Any other language, toolset or API with C bindings can also be incorporated. Workspace also features a distributed execution capability allowing it to farm out work to local CPU cores, remote machines via TCP and cluster job systems. This feature allows easy distribution of simulation jobs to PBS-based clusters and can be easily extended to support other job systems.

Workspace shares many similarities with other commercially available and open source workflow tools such as Taverna [2], Trident [3], KNIME [4], Kepler [5, 6], Simulink/Matlab [7] and LabView [8]. However, the Workspace framework has a stronger focus on being an application development platform rather than simply a workflow editor. It offers a natural path from early code development as part of a research workflow right through to the development of standalone applications for deployment to a collaborator or external client. Inbuilt support for automated testing, cross-platform installer generation and native user-interface development are some distinguishing features supporting this focus. Workspace is designed to be generically applicable to any kind of dependency-based workflow problem, and is easily extendable. It is not restricted to a particular domain, or particular type of workflow, and developers can easily extend its functionality by writing plug-ins. Some workflow engines (such as KNIME and Kepler) are similar in this way, while others have more domain specific focus (such as LabView for measurement and control systems and Simulink for simulation and model-based design). The system also transfers data in-memory between operations as opposed to using local storage or a database as the intermediary between operations, as implemented in workflow engines that focus on web-service connectivity such as Taverna and Trident. In-memory operation is very suited to the computationally intensive needs of natural hazard modelling, in contrast to local storage which can severely impede performance. Workspace is also the only workflow tool which can execute in continuous mode. In this mode, any modification to an operation input triggers an immediate downstream update of the workflow, in-

cluding all relevant visual outputs. This event-driven approach enables users to immediately visualize factors such as changes to their input conditions or post processing steps.

Workspace is built on free software license tools, such as the Qt framework. This allows researchers and developers to freely build and distribute applications based on their workflows, or integrate them into existing software packages. Workflow-based application development like this is not well supported in open source available alternatives while commercial packages such as Simulink/Matlab and LabView require paid licenses. Furthermore, although packages such as Simulink/Matlab also provide user interface customization features, Workspace's tight integration with the Qt framework enables developers to quickly and easily develop user interfaces for their workflows. A number of specialized Qt widgets have been developed specifically for natural hazard operations, such as interactive time-series inputs and two-dimensional layer views. These can be directly built into a user interface using the Qt Designer tool and linked with an underlying workflow. Furthermore, a design goal of Workspace was for computational fluid dynamics applications, and it therefore offers far greater interactive 2D and 3D scientific visualization capabilities than alternative workflow engines.

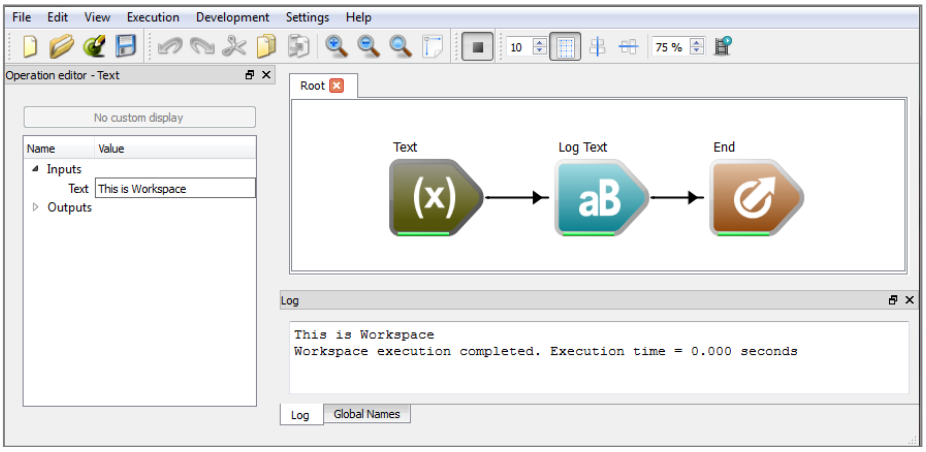

Fig. 3. A simple example of a workflow in the Workspace GUI.

An example workflow is shown in Fig. 3 within the Workspace editor GUI. Individual workflow elements, or operations, are represented as colored triangular blocks which are logically joined in the GUI by dragging output data from an operation to the input of another operation. The joins, and the direction of data flow, are shown as black connection arrows in the GUI. In this simple example some text data is created ('Text' block) and passed into an operation to output the text to a log ('Log Text' block). A final block acts as an operation representing the end of the workflow ('End' block). This example passes string data, but new data types can easily be defined and added. For the natural hazard solvers both two-dimensional gridded data sets and one-dimensional time series data sets were added as new data types within Workspace.

The workflow is saved as xml, and once a Workspace workflow has been developed it can be easily shared between collaborators. Workspace supports execution both from the GUI shown in Fig. 3, execution in batch mode from the command-line, or an API for stand-alone Qt-based applications.

The computational solver elements are implemented as individual operations. Each solver operation requires a set of configuration parameters specific to the particular operation. These parameters consist of items such as the total time for the solver to be run, how often to output simulation data and global physical variables for the solver. The solver inputs also include two dimensional input data sets for the starting conditions as well as two dimensional layers such as topography and land usage. The solver executes for a specified period before pausing and returning control back to Workspace. Once control is returned, Workspace operations outside the solver have access to internal data from the solver and can be used to analyze, output, display or collate the data. Once these steps are complete, control is returned to the solver which resumes, runs again for the specified period, then returns control back to Workspace. This loop is carried out until the solver reaches the specified total simulation time, at which point Workspace ends execution.

The solver operations incorporate an additional feature, called processor modules, which are exposed as a subclass of Workspace operations. These processors are subsolvers with full access to the solver data and are used to provide specific capabilities for any given scenario. The processor modules are separate operations which plug into a corresponding solver. Multiple processors can be connected to a solver, which transparently controls execution and data management within each processor. Examples of these processors for the hydrodynamic solver, SWIFT, include providing rainfall conditions, applying tidal boundary conditions, infiltration effects, evapotranspiration and coupling with a one-dimensional pipe network model. These processors can be developed separately from the solver, which was a key requirement in the design of the solvers as open platforms for collaboration.

3 Workflow for Saline Intrusion

The hydrodynamic model SWIFT is based on a finite-volume shallow water formulation, which is both conservative, positivity preserving and well-balanced [9]. The project in this example was designed to examine the effects of saline intrusion from potential sea level rise on environmentally and culturally sensitive areas of Kakadu National Park in the Northern Territory, Australia. Multiple tidal cycles were required to ensure adequate saline mixing with the freshwater from the catchments, so the simulations were run for a period of 30 days. The simulation domain was an area of approximately 150 km × 120 km, encompassing the major catchments of the park, at a resolution of 60 m.

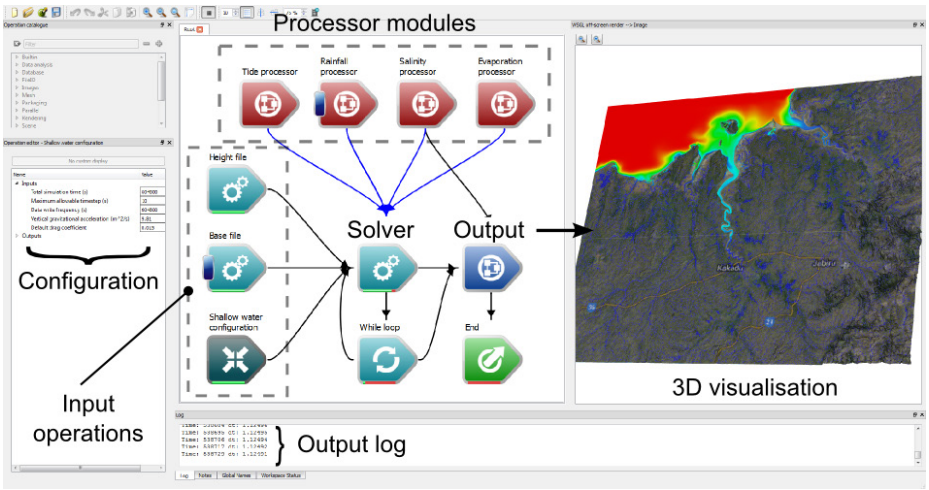

Fig. 4. Example workflow configuration for environmental impact assessment of saline intrusion in Kakadu National Park, Australia.

The Workspace GUI in Fig. 4 has been arranged with input operations on the left-hand side, the solver at the center and a 3D output of Kadadu national park on the right hand side. The inputs for this model consisted of a digital elevation model (DEM) for the bathymetry and topography as well as the initial salinity state. These were supplied as files in a standard ESRI ASCII format. The 3D output was configured for this example to show the textured digital elevation model with a water layer colored by saline concentration. The processor modules are shown as the set of upper red boxes. These included a processor module to apply the tides to the northern boundary, a module for rainfall based on a time series from measured data, a module for evapotranspiration and a module for advective-diffusive saline transport and mixing.

The actual simulations for the project were run in batch mode using PBS on the CSIRO Accelerator Cluster, rather than through the interactive GUI shown in Fig. 4. The simulations took 42 hours to complete on a Tesla K20 GPU, with an average simulation time-step of 0.7 s. Data from the simulation was written to disk in a binary format and an analysis step was performed on this data using a second workflow to provide various time-averaged metrics such as the maximum, minimum and median water levels and salinity concentrations over the 30 day simulation. These metrics are currently being used for an impact assessment of sea level rise on freshwater species and cultural sites in the region for the NERP Northern Australia Hub.

4 Workflow for Bushfires

The scenario in this second example is a comparison between a controlled experimental fire in a plot of cured grass and a level-set based bushfire computational model, SPARK [10]. The level set method is a computational model for the propagation of

an interface [11], and can be applied to numerous systems such as crystal growth, droplet dynamics and bubbles. SPARK uses the level set method to track the interface between regions in a fire that are burnt and un-burnt. The level set method has several advantages over other computational methods for interface propagation; it is raster based and therefore highly suited to GPU acceleration, it naturally handles the merging of interfaces and the interface can be given an arbitrary outward speed at any point.

The outward speed of the interface in SPARK is equal to the rate of spread of the fire. The functional form of rates of spread in real fires are known to depend on several factors relating to fire propagation including the wind speed, wind direction, fuel and moisture levels and slope of the ground. Many empirical expressions have been given for various fuel types [12, 13], but the functional form for rates of spread in two dimensions is currently an active area of research. SPARK is currently being used as a research tool for investigating new two-dimensional rate of spread empirical models based on recorded fire data. The experiments shown in this example are number of large-scale fire experiments were carried out in Ballarat, Victoria in Jan 2014. These experiments measured the speed and shape of fire propagation over flat fields of cured grass. The wind speed and direction during the fire were recorded during the experiment, and the progression of the fire perimeter was filmed using camera mounted on a quad-copter.

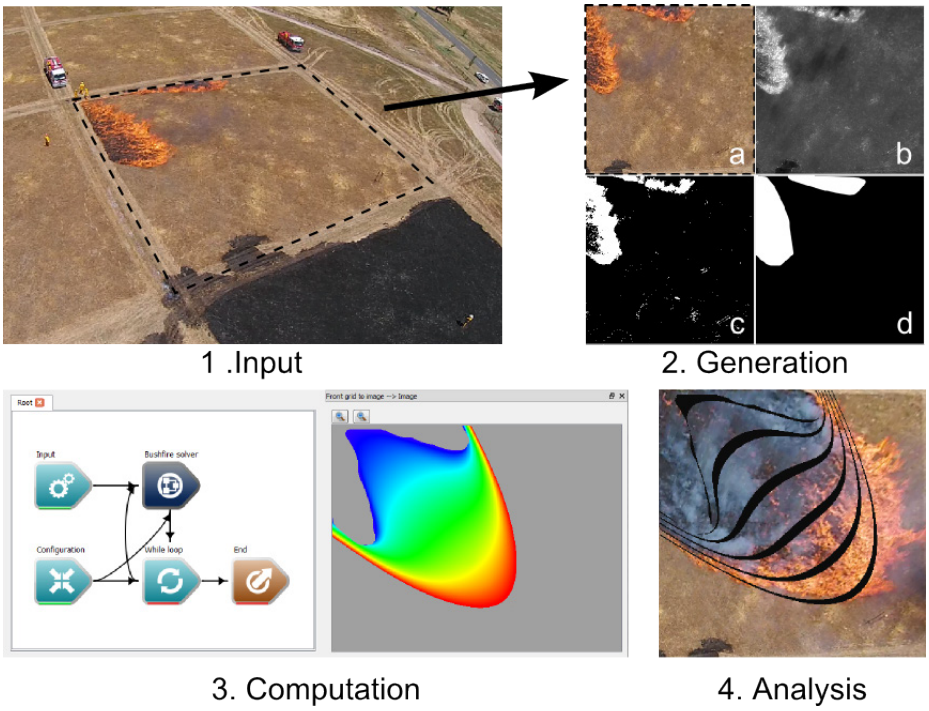

Fig. 5. Stages for comparison of experimental data to simulation results.

Comparison of the simulation outputs with the recorded experimental data, shown in Fig. 5.1, required a sequence of image processing transformations. These steps are shown in Fig. 5.2, including (a) rectification of the original image onto a Cartesian grid, (b) identification of the burning areas, (c) thresholding and (d) reconstruction of the interface between burnt and un-burnt areas. Each of these steps were carried out using OpenCV operations which have been natively incorporated into Workspace. Once the data had been rectified, it could be used as the initial condition in the simulations. Fig. 5.3 shows an example configuration of SPARK in operation, where the panel on the left hand side shows the individual operations comprising the workflow, and the image on the right hand side a preview of the results. The color scale in the image represents the arrival time of the fire, with blue equal to zero and red equal to the current simulation time. In the final stage, the results of the model can be used to forward predict the behavior of the fire perimeter, shown in Fig. 5.4. The perimeter locations predicted by the model at 5 second intervals up to 25 seconds from the start of the fire are shown as black bands in the image. These are superimposed over the recorded fire perimeter at 25 seconds from ignition time.

Each of the stages in this example consists of a number of self-contained processes which operated on similar data types. Using our architecture, all stages can be directly processed from start to end within the same framework. These were logically chained together in the workflow environment for each image processing step. An operation could then be used to convert the output image directly into the initial conditions required for the bushfire computational solver after the image processing steps were complete. Finally, data from the solver was visualized for prediction of the fire perimeter. As well as a research tool, SPARK is currently being built into an operational predictive model using published empirical expressions for fire rates of spread over grassland and Eucalypt forest.

5 Conclusion

The combination of computational solvers and the CSIRO Workspace framework gives a flexible, modular platform for modelling natural hazards. All of the steps in a typical natural hazard simulation, from reading and cleaning input data, simulation, analysis and visualization, have been implemented as individual operations on common data types. These operation can be either be contained within a single end-to-end workflow or, dependent on requirements, split into sub-workflows for pre-processing, execution and post processing. The Workspace framework has a number of advantages over existing workflow engines. These include in-memory data transfer between operations, which is advantageous for the complex, real-time applications and computationally intensive simulations found in natural hazard modelling. The framework also allows workflows to be run in an interactive continuous mode. In fast natural hazard models, such as SPARK, this interactivity enables conditions to be changed on-the fly. Alternatively, workflows for detailed simulations, such as the SWIFT application in this paper, can be executed in a batch mode on high performance computing nodes.

The architecture has been demonstrated in this paper for two very different applications, environmental impact analysis from saline intrusion and image processing of real fire events as input into a fire model. These applications represent only a small subset of the potential applications for the framework. Environmental modelling workflows can be rapidly tailored for any natural hazard application, such as operational usage from data streams or detailed simulations for environmental modeling. These workflows can, if required, be packaged and run behind a Qt user interface and freely distributed by virtue of the permissive software licenses on which Workspace is built. Future work includes adding further solver components to the system, including models for dust advection, granular dynamics and mudflows. However, the open nature of the architecture allows other users and research groups to add any new computational models to the system. Alternatively, processing units can be constructed and freely added to run sub-models on any of the existing natural hazard models.

We have demonstrated the potential of this system within this paper, and expect many applications for this system in the future within the environmental modelling space.

References

1. Workspace, 2014. http://research.csiro.au/workspace/
2. Taverna Workflow Management System, 2014. http://www.taverna.org.uk/
3. Project Trident: A scientific workflow workbench, 2014. http://tridentworkflow.codeplex.com/
4. KNIME, 2014. http://www.knime.org/
5. The Kepler project, 2014. https://kepler-project.org/
6. Altintas, I., Berkley, C., Jaeger, E., Jones, M., Ludascher, B., Mock, S., 2004, Kepler: an extensible system for design and execution of scientific workflows. Scientific and Statistical Database Management, 2004. Proceedings. 16th International Conference on., 423-424
7. Simulink simulation and model-based design, 2014. www.mathworks.com/products/simulink/
8. LabVIEW System Design Software, 2014. http://www.ni.com/labview/
9. Kurganov, A., Petrova, G., 2007, A Second-Order Well-Balanced Positivity Preserving Central-Upwind Scheme for the Saint-Venant System, Commun. Math. Sci., 5, 133-160
10. Hilton, J. E., Miller, C., Sullivan, A., Rucinski, C. Incorporation of variation into wildfire spread models using a level set approach. Environmental Modelling and Software (under review).
11. Sethian, J.A., 2001, Evolution, implementation, and application of level set and fast marching methods for advancing fronts, Journal of Computational Physics, 169, 503-555
12. Gould J. S., McCaw W. L., Cheney N. P., Ellis P. F., Knight I. K., and Sullivan A.L., 2007, Project Vesta – Fire in Dry Eucalypt Forest: Fuel Structure, Dynamics and Fire Behaviour, CSIRO Ensis and Department of Environment and Conservation: Canberra, ACT
13. Cheney, N. P., Gould, J. S., Catchpole, W.R., 1998, Prediction of fire spread in grasslands, International Journal of Wildland Fire, 8, 1-13