An Object-Oriented Software Framework for the Farm-Scale Simulation of Nitrate Leaching from Agricultural Land Uses – IRAP FarmSim

[a]Good J. and [b]J. Bright

[a]Lincoln Ventures Limited, Lincoln, New Zealand, [b]Aqualinc Research Limited, Lincoln, New Zealand, Email: goodj@lincoln.ac.nz

Keywords: *agricultural modelling; object-oriented; nitrate leaching.*

EXTENDED ABSTRACT

The purpose of this project is the creation of a framework that will allow the prediction of drainage flux and nitrate leaching from a whole farm taking into account a full range of agricultural activities. Ultimately, the simulation framework will be used to provide nitrate and drainage flux input values for a regional groundwater model. In addition, the project aims to provide a highly effective and adaptable farm-scale simulation framework that has application well beyond the scope of nitrate leaching prediction. It is being implemented as a component-based simulation, utilising models produced by research collaborators within the Integrated Research for Aquifer Protection (IRAP) programme and from the public domain.

The simulation framework consists of a variable number of individual paddock simulations controlled by a detailed farm-scale management component. When complete, FarmSim will be able to represent a wide range of agricultural activities including pasture-based grazing (sheep and dairy farming) and cropping using a wide range of cropping models applicable in New Zealand.

The paddock simulation comprises vadose zone and soil components that represent unique soil and vadose conditions of each individual paddock within the farm. At any time, each paddock contains a particular crop component. The farm management module contains numerous components that enable the farm to be managed as any typical agricultural unit. These components include irrigation, soil fertility and stock management, crop rotation, cultivation and harvest. The farm management module mediates between these components to effect the overall running of the farm, which is subject to climatic inputs that are specific to the subject site.

The primary challenge in creating the framework is to ensure that it is able to integrate interchangeable model components from a number of different researchers. This has been achieved utilising modern object-oriented software design techniques and tools. FarmSim runs on the Microsoft .NET Framework and makes extensive use of techniques such as the Model/View/Controller aggregate pattern, generic interfaces to components, and an object-oriented design approach incorporating inheritance, polymorphism and object-oriented design patterns. The representation of data input and outputs, as well as transfer between components, makes extensive use of Extensible Markup Language (XML). This greatly assists interfacing of FarmSim with other applications and the cross platform operation of the product. While the framework has been developed in the Microsoft Visual C#.NET language, components developed in a wide range of other languages can be incorporated.

The development of a modelling and simulation framework where the modelling components are drawn from a number of different sources poses a considerable number of challenges, In particular, the componentisation of models to enable the operation of different combinations of components requires careful consideration. This requires the development of agreed interfaces and careful design of components to perform clearly defined roles within the framework. This is achieved by ensuring that the simulation entities closely emulate the functioning of objects in the natural system.

1 INTRODUCTION

Nitrate contamination of groundwater from agricultural land uses is of widespread concern due to the potential to affect surface water habitats and drinking water quality. The primary aim of the New Zealand Integrated Research for Aquifer Protection (IRAP) is to produce tools that predict the cumulative effects of land use changes on the quality of surface and groundwater at the aquifer scale (IRAP 2004).

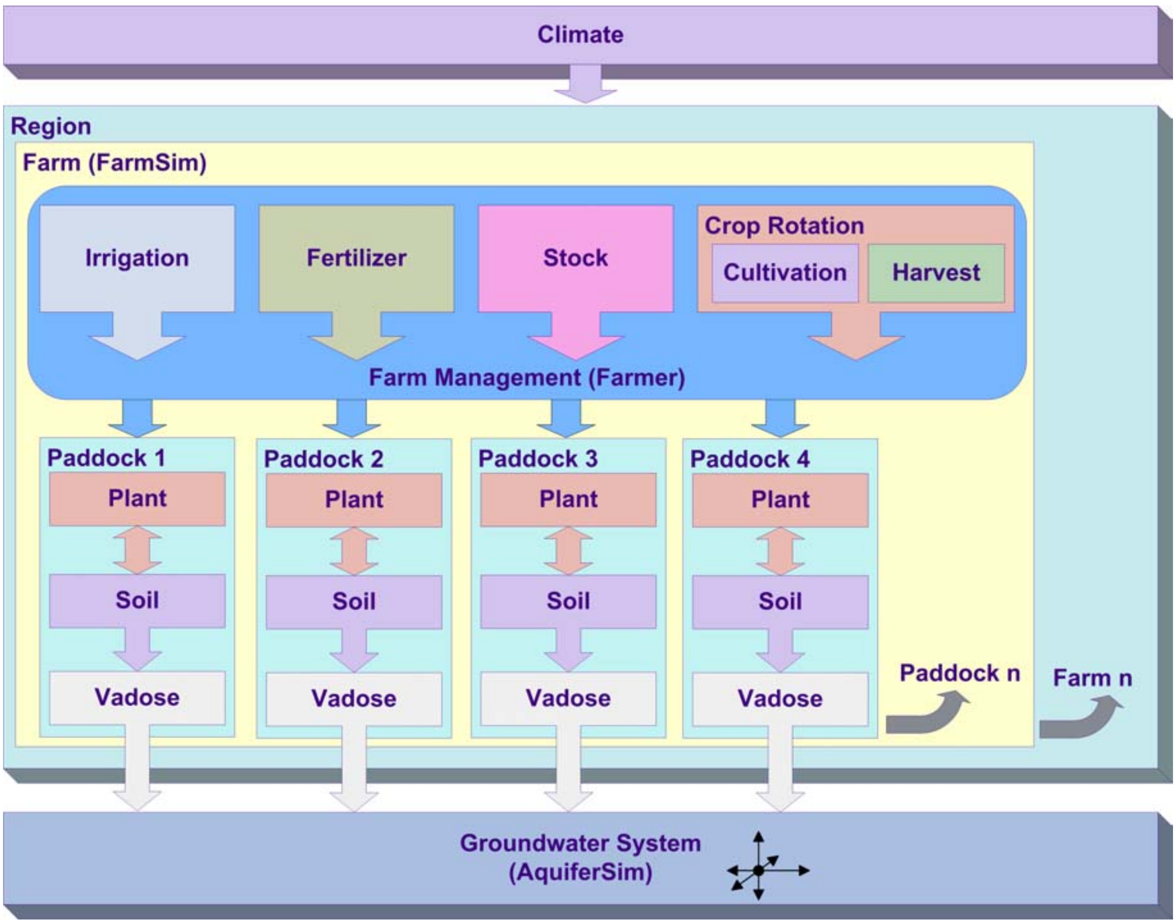
The IRAP programme involves specialist contributions from eight New Zealand science research agencies and an End User Advisory Group (EAG) made up of regional and district council, central government, farmer and tangata whenua representatives (Maori from the Canterbury region). The research partners are Crop & Food Research, AgResearch, Dexcel, Landcare Research, Environmental Science and Research, Lincoln Environmental Research (Lincoln Ventures), Aqualinc Research and Environment Canterbury.

The IRAP FarmSim model is an integral part of the IRAP tools. Its purpose is to simulate at a farm-scale the operation of cropping and pasture grazing (sheep and dairy) farms under differing management regimes in a manner that reflects the unique conditions of each individual paddock within the farm. The decision to build FarmSim was brought about by no suitable model being available to the project. Existing farm systems models do not enable the on-farm management of land and water to be simulated in a way that realistically reflects the New Zealand situation. The primary focus of the simulation is to determine for each paddock the time-varying effect of these land uses on drainage flux and nitrate leaching from the vadose zone into the underlying groundwater. In addition, FarmSim should provide a practical and usable framework for the integrated operation of a number of agricultural models both from the public domain and produced by IRAP participants.

2 FARMSIM STRUCTURE

FarmSim is essentially a daily time step simulation of a single unit of agricultural production on alluvial plains, where run-off transfer is not a feature. The simulation responds to time-varying inputs from the climate and daily farm-level management decisions. The farm model responds to these inputs and the state changes of the various components can be analysed. Of these state changes, the daily drainage flux and nitrate concentration from the vadose zone model component can then be passed to the AquiferSim

Figure 1 – FarmSim Model Structure and Interactions



catchment-scale transport model (Bidwell *et al.* 2005). However, as development progresses FarmSim will become a comprehensive simulation of a wide range of agricultural systems. This will result in the simulation having numerous applications beyond predicting outputs from the vadose zone. The vadose zone is modelled separately as the processes involved are simpler than those in the soil and in general the environments being modelled have a vadose zone of considerable depth.

FarmSim is produced as an open modelling framework that can be arranged in a number of different configurations. Figure 1 shows the conceptual structure of FarmSim and the basic interactions between the model components.

The initial release of the FarmSim will be based on the entry-level IRAP toolkit. This toolkit contains all the simulation entities. The entry-level version is based on public domain and widely documented models. This will provide a proof of concept for the FarmSim and a useful test framework for developing more advanced components that result from the extensive field research being carried out as part of IRAP programme.

2.1 Farm

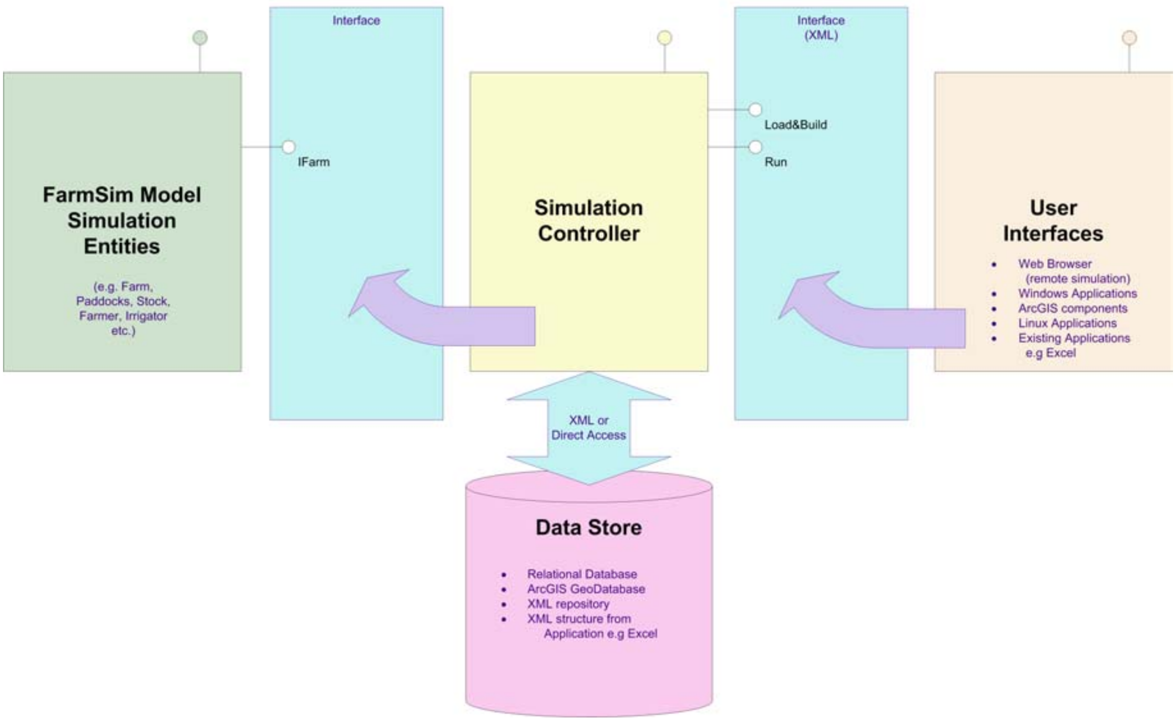
The Farm is the main simulation entity and represents the highest-level unit of the simulation. From an object-oriented perspective the Farm 'owns' all the other components in the simulation. The Farm is the first simulation entity to be created by the simulation controller and it then initiates the creation of all the entities that make up the Farm. The primary models that make up the Farm are the Land (collection of paddocks), Irrigation, Water Supply, Stock and the Farmer (the farm management component).

2.2 Farmer

The component that manages the operation of the Farm is referred to as the Farmer. The Farmer contains the logic for the scheduling of farm operations such as crop sowing, harvest, cultivation and rotation. It comprises of several stand-alone management components that manage irrigation, land fertility (fertilizer application) and stock.

For example, the Farmer's FarmIrrigationManager component receives information from the various paddocks describing the daily water needs of the crop (including pasture) in that paddock. The FarmIrrigationManager then ranks these demands based upon the relative time-varying management priority of the crop. The Manager then analyses the available water for that day and the location, availability and capacity of irrigation equipment. The irrigation decision is then made and irrigation carried out. The FarmIrrigationManager is capable of making the irrigation using any number of different algorithms including using optimization techniques.

Figure 2 – FarmSim Components



2.3 Paddock

The fundamental simulation unit is the Paddock. It is made up of a single Crop, Soil Profile and Vadose zone model. This enables varying Crop, Soil and Vadose conditions to be represented across a particular Farm and management decisions to be based on these varying conditions. The ultimate output of drainage flux and nitrate concentration from the Vadose zone to catchment-scale aquifer models is on a paddock scale. The Vadose component is based upon a one-dimensional advective-dispersive transport model (Bidwell 2000).

3 DESIGN PRINCIPLES

FarmSim is intended to be an open modelling/simulation framework developed using industry standard software engineering techniques. FarmSim will be well documented and provide extensible and adaptable platform for agricultural simulation. The process of developing the initial design for FarmSim involved reviewing experience with the development of similar frameworks. In particular FarmSim uses a similar approach to the Dexcel Whole Farm Model (Sherlock and Bright 1999) and the US Department of Agriculture GPFARM framework (Shaffer *et al.* 2000). This process also resulted in a number of fundamental simulation design principles being adopted many of which were drawn from experience with other simulation frameworks (Post 2003). These principles guide the development of the simulation as a whole and the design of individual components.

3.1 Object-Oriented Design

While the simulation is developed within an object-oriented platform, it is quite possible to deviate substantially from an object-oriented design. It has been a clear priority for the development of FarmSim that the object-oriented design of the simulation entities should emulate (as close as practical) the operation of the entities in the natural system. This is considered essential for the proper operation of an interchangeable component simulation. An example of this is how the framework incorporates the process of evapotranspiration. Crop models commonly determine the evapotranspiration from both the soil and the plant. However, in ensuring the natural system is emulated as close as possible it is more appropriate that evaporation from the soil surface be determined by processes and state variables related to the soil and soil surface. Transpiration is a plant process, so transpiration methods are defined in the plant component. The methods use plant, soil and climate parameters and state variables. Soil and climate data is obtained by the plant component via interfaces to the soil and climate components. Evaporation from the soil surface is considered a soil process and methods are defined in the soil component. The methods use soil and climate parameters and state variables, but in this case, the climate is different to that seen by the plant component. The climate seen by the soil is a function of the state of the plant. The transport of water and chemicals within the soil is a soil method. Mass balance is a key element of all transport methods associated with the soil component. Alternate transport methods may be used, depending on data availability and application purpose. These will range from a simple single-layer water balance method through to a Richard's equation based method. Separately defining plant, evaporation and soil processes in this way simplifies the substitution of alternate methods for describing a process. To accommodate differences in data availability from project to project, for example.

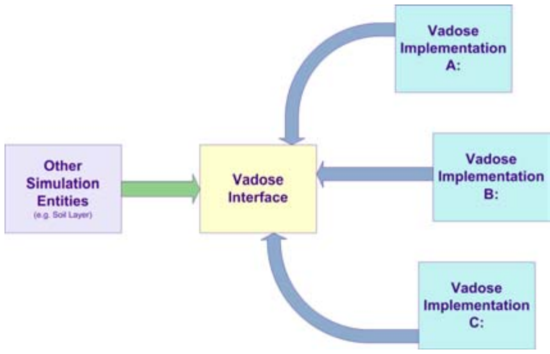


Figure 3 – Coding to Interfaces

3.2 Componentised

From the outset, the separation of the various functions of the software into components was regarded as essential. Figure 2 shows the component structure which is based upon the Model/View/Controller aggregate pattern (Gamma *et al.* 1994). This generic design pattern is considered very suitable for simulation frameworks (Post 2003). The simulation entities (the farm and its components) are kept deliberately separate from the parts of the system that control the simulation, manage data and gather user input. Likewise, the user interfaces work through a defined interface to the simulation controller. This decoupling of the components and use of interfaces enables the components to be developed separately and limits the impact of changes to one component. This approach has already been beneficial with Aqualinc Research Limited (Aqualinc) developing an experimental user interface for testing of the FarmSim.

The principle of coding to interfaces rather that directly to the implementation (Metsker 2004) is also applied to situations where there are a number of possible models for a particular simulation entity. Figure 3 shows how numerous vadose zone model implementations can communicate with other simulation entities through a consistent interface. This has the benefit that the other simulation entities only need to be aware of the interface rather than the specifics of each implementation.

4 USER INTERFACES

The Model/View/Controller approach to the development of FarmSim results in there being a wide range of options for the simulations user interfaces. The transfer of data inputs and outputs into and out of FarmSim is achieved exclusively using Extensible Markup Language (XML) (W3C 2004). The Simulation Controller only requires XML documents that conform to specific schema to initiate the simulation. Once the simulation is completed, an output XML file is returned. The experimental user interface developed by Aqualinc is a windows application that utilises a tree view to explore and edit the input data (very similar to Windows Explorer). The output XML data can be explored in a similar fashion and a time-series graphing component is used to examine the time-varying nature of the state variables.

In addition to the approach of a using a tree-view based windows applications, there are a number of different options for user interfaces for FarmSim. These include accessing the simulation over the internet as a web application, running vast numbers of simulations in a parallel-processing environment and controlling FarmSim from a GIS application that contains the majority of the source data required for the simulation. In addition, there may be end-user interest in user interfaces that are suitable for farmers to test the effect of their management practices on the likely nitrate pollution of groundwater.

5 DEVELOPMENT PLATFORM AND TOOLS

The development environment chosen for FarmSim is the Microsoft .NET framework. This framework already exists on most modern personal computers as it is installed as part of the Windows XP operating system. The .NET framework provides a highly managed Common Language Runtime (CLR) system with an enormous library of base classes. The CLR can run code produced by any .NET language compiler, of which there are currently around 100 languages (Ritchie 2005).

It is also possible to utilise non-.NET libraries and components using Component Object Model (COM) interfacing. This is a practical means of incorporating legacy model components but negates many of the benefits of using the .NET framework. While the .NET Framework was initially deployed by Microsoft for the Windows platform only, the open-source Mono Project now allows .NET applications to be developed and run in the Linux and Apple Macintosh (OSX) operating systems (Mono Project 2005a).

The primary language for the development of FarmSim is Microsoft Visual C#.NET. This language, which bears considerable similarities to Java and C++, was development specifically for use with the .NET framework by Microsoft. The C# language is a well-structured and efficient language that promotes the use of sound object-oriented programming techniques. It also allows for the easy utilisation of the vast library of .Net base classes. The Mono Project also provides a C# compiler (Mono Project 2005b) that allows C# to be used in a cross-platform environment. Recent experience converting a significant legacy model component developed in un-managed (non-.NET) C++ code shows that the conversion is quite strait-forward, with the majority of the work being in redirecting calls to the appropriate base classes in the .NET framework.

XML is an industry standard data exchange language that provides a self-describing format for data representation. It is widely implemented in modern applications and allows FarmSim to integrate with end-user tools and databases, and work in a cross-platform environment. More detail about the use of XML within FarmSim is provided in a companion paper at MODSIM05 (Good 2005). The development of FarmSim is supported by the use of test-driven development (TDD) techniques using NUnit (NUnit 2005). This enables a complete suite of tests to be repeatedly run against the code to ensure correct functionality is maintained.

The use of the Microsoft .NET framework and XML has previously been identified as having considerable potential for the development of environmental simulation frameworks (Argent 2004). In practice, these tools have proved to be highly suitable for this project, providing great flexibility without any noticeable cost in the performance of the simulations.

6 APPLICATIONS WITHIN IRAP

The IRAP programme is able to utilise FarmSim in a number of ways. It will be the primary tool for determining farm-scale impacts of land use changes on the nitrate leaching to groundwater. FarmSim provides a framework for the development and testing of individual model components by the IRAP participants.

Once completed to an appropriate level of accuracy FarmSim will be use to determine the farm-scale inputs to the GIS-based catchment-scale model of nitrate distribution in aquifers (AquiferSim). AquiferSim is described in detail in a companion paper at MODSIM05 (Bidwell *et al.* 2005).

7 APPLICATIONS OUTSIDE IRAP

FarmSim has a number of applications outside the IRAP programme. In particular, it will be of considerable use within the Sustainable Groundwater Allocation Research (SUGAR) programme. This programme is focussed on sustainable groundwater abstraction and how to efficiently, effectively and equitably allocate this water between groundwater users. FarmSim is able to simulate water use and aquifer recharge from a wide range of agricultural activities under differing management scenarios. The output from FarmSim will be used in conjunction with an catchment-scale groundwater model to determine the effects of abstraction on groundwater levels and water supply reliability. This will enable the likely impact of a large number of allocation strategy alternatives to be considered.

In addition to use within the SUGAR program FarmSim will be used in applications where accurate estimations of water use from agricultural land uses are required such as the design of regional irrigation systems. Other potential applications for FarmSim include use as the simulation engine behind farm management training tools that examine the effects of management practices on the environment.

8 CONCLUSIONS

The development of the IRAP FarmSim provides a unique opportunity to create an open farm-scale modelling framework. The aim is to develop a framework that not only satisfies the primary purpose of accurately predicting drainage flux and nitrate leaching, but also is a very effective general-purpose farm simulation framework.

These objectives are met by utilising best practice object-oriented development techniques and design patterns. The development of the framework on the Microsoft .NET platform and extensive use of XML for data representation and transfer has had considerable benefits. The experience of developing the entry-level application has shown that using these techniques and tools results in a product that is highly adaptable, extensible and useful in a number of different computing environments.

It is considered that the techniques and tools described in this paper have proven to have considerable potential for the development of custom modelling and simulation software. This is particularly the case where the project involves the development of a simulation framework and involves modelling contributions from a number of sources.

The challenge for the continued development of FarmSim is to ensure that the incorporation of more comprehensive simulation components does not compromise the design principles and practices that have been used for the entry-level product.

9 FUTURE WORK

At the time of writing, the development of the initial release of FarmSim and entry-level IRAP Toolkit is considerably advanced. It is expected that following initial release there will be a considerable period of refinement. In particular, the nature of the user interfaces required by the various end-user groups will need to be finalised and these interfaces developed.

Once the initial release has been completed and refined, the effort shifts into developing more advanced individual model components. These will be based on the result of fieldwork and research being carried out by the IRAP partners at present.

10 ACKNOWLEDGMENTS

This project is funded by the New Zealand Foundation for Research Science and Technology. Particular thanks go to Rob Zyskowski, Peter Jamieson, Linda Lilburne, Trevor Webb, Vince Bidwell, Peter Brown and Elizabeth Post.

11 REFERENCES

Argent, R. M. (2004), An overview of model integration for environmental applications - components, frameworks and semantics, *Environmental Modelling and Software,* 19(3), 219-234.

Bidwell, V. J. (2000), A formula for predicting advection-dispersion in the vadose zone at uneven drainage intervals, *Water Resources Research,* 36(10), 3077-3080.

Bidwell, V. J., Lilburne, L. R., and Good, J. M. (2005), Strategy for developing GIS-based Tools for management of the effects on groundwater nitrate leaching from Agricultural Land Use*, paper presented at the International Congress on Modelling and Simulation - MODSIM05*, Melbourne, Australia, December 12-15.

Gamma, E., Helm, R., Johnson, R., and Vlissades, J. (1994). *Design Patterns: elements of re-usable object-oriented software*. Reading, Massachusetts: Addison Wesley Longman.

Good, J. M. (2005), The Benefits and Practicalities of using Extensible Markup Language (XML) for the Interfacing and Control of Object-Oriented Simulations, *paper presented at the International Congress on Modelling and Simulation - MODSIM05*, Melbourne, Australia, December 12-15.

IRAP, (2004), *What is IRAP?*, http://www.irap.org.nz/Default.asp Last Accessed June 29, 2005.

Metsker, S. J. (2004). *Design Patterns in C#*. Boston: Pearson Education.

Mono Project, (2005a), *About Mono*, http://www.mono-project.com/Mono:About Last Accessed July 7, 2005.

Mono Project, (2005b), *MCS: The Ximan C# Compiler*, http://www.go-mono.com/c-sharp.html Last Accessed July 25, 2005.

NUnit, (2005), *NUnit Homepage*, http://www.nunit.org/index.html Last Accessed 25 July, 2005.

Post, E. (2003), Designing and implementing computer simulation models for portability and reuse, *paper presented at the International Congress on Modelling and Simulation - MODSIM03*, Townsville, Australia, July 14-17.

Ritchie, B., (2005), *dotnetpowered.com - .NET Languages*, http://www.dotnetpowered.com/languages.aspx Last Accessed 25 July, 2005.

Shaffer, M. J., Bartling, P. N. S., and Ascough II, J. C. (2000), Object-oriented simulation of integrated whole farms: GPFARM framework, *Computers and Electronics in Agriculture,* 28(1), 29-49.

Sherlock, R. A., and Bright, K. P. (1999), An object-oriented framework for farm system simulation, *paper presented at the International Congress on Modelling and Simulation - MODSIM99*, Hamilton, New Zealand, December 6-9.

W3C, World Wide Web Consortium (2004), *Extensible Markup Language (XML) 1.0 (Third Edition) - W3C Recommendation 04 February 2004*, http://www.w3.org/TR/2004/REC-xml-20040204/ Last Accessed June 28, 2005.