CHAPTER 1.

GENERAL INTRODUCTION

1.1. Introduction of the main concepts and contributions of this project

It has been recognised over the past several decades that the world's natural resources are under stress and need to be managed or restored so as to prolong ecosystem sustainability. Recognition that natural resource management strategies need to deal with issues from a total system perspective has led to the concept of integrated ecosystem management. Understanding the links between physical and biological processes in both natural and developed ecosystems allows the causes of environmental degradation to be better understood, identified and strategically managed. However, limited availability of information on the status of natural resources within ecosystems has inhibited the implementation of integrated ecosystem management strategies (Syme *et al.*, 1999). Ecosystem computer models have become increasingly accepted as useful decision-support tools in bridging the gaps in ecological information and expected management outcomes (Shenk & Franklin, 2001).

All ecologists would agree that ecological systems are diverse and complex. They are often difficult to investigate, especially in a quantitative manner (Kremer & Nixon, 1978). Field and laboratory experiments are effective methods used to study ecological systems. However, it often becomes complicated, time consuming and expensive to carry out long term studies using these techniques (Bergez *et al.*, 2001). A computer model can be a substitute for a real system, and can be an alternative method of quantitative analysis (Alekseev *et al.*, 1984; Bergez *et al.*, 2001; Costanza & Gottlieb, 1998; Ford, 1999; Jayaweera & Asaeda, 1996). Thomann (1998) aptly describes ecological models as the "glue" between the perception of the problem, the observational data from the laboratory and from the field and the current state of scientific understanding. Accordingly, ecologists are fast recognising the importance of computer modelling for understanding and investigating the natural environment (Al-Khudhairy *et al.*, 2001; Reynolds & Irish, 1997; Shenk & Franklin, 2001). Jorgensen (1995) outlines three main factors that have determined the rapid growth of the ecological modelling field:

1. Access to affordable and increasingly faster computers,
2. The urgent need for more quantitative management solutions to solve increasing problems with environment degradation,
3. An increasing interest for quantitative approaches in ecology.

These factors have transformed the way that today's ecologists pursue their scientific research, especially when conventional methods have proven to be problematic. It should be stressed that models do not make management and control decisions, but only provide information to these processes. That information can be somewhat imperfect, but in response to ever-increasing needs for reliable methodologies models have evolved in many ways over the past 70 years (Mooij & Boersma, 1996; Steel, 1997; Thomann, 1998).

Since the 1970s numerous deterministic models based on differential equations have been adopted for the simulation of lake ecosystems, but this particular method of modelling ecosystems has proven problematic in some areas. Often such models are:

1. Highly complex and structurally rigid
2. Require extensive knowledge about a system, which can be sometimes lacking
3. Require extensive and high-quality data sets in order to properly calibrate and validate the model
4. Can be difficult to debug and maintain, and
5. Are not readily shared with other researchers due to incompatibilities with modelling tools and programming languages used to create the model.

Modular and generic models offer a solution to many of these problems. Adopting the object-oriented modelling paradigm to create modular models can facilitate the understanding, debugging and adaptation of complex models by providing a platform that is more flexible, user-friendly and transparent. Using the Java programming language for object-oriented implementation of models gives the ability to access models from the Internet, which facilitates model and information sharing. Generic models provide an alternative to developing ad hoc models for each specific lake site or condition. In addition, there is only one model structure to understand and maintain, and the reduction of input data also results in less calibration of a model (Hamilton & Schladow, 1997; Jorgensen, 1988). However, a major limitation with generic models is that in trying to achieve generality predictive power is often reduced, as compromises have to be made in order to apply the model to different conditions. The lake ecosystem model SALMO (Benndorf, 1979; Recknagel, 1980; Benndorf and Recknagel, 1982) has been designed as a generic model for the simulation of a wide variety of freshwater lake systems. It has been successfully applied to many lakes with different trophic states, morphometry and climate conditions. However, the model may not perform as well in a predictive capacity compared to model applications that have been developed specifically for a particular system.

One way to improve the generality and predictive capacity of the SALMO model is to apply a "model library" composed of alternative process-based functions from which different model structures can be tested based on different trophic states, climate conditions or lake morphometry. This model library could act as an additional validation process, where the original model can be further improved by exploring model behaviour through alternative model structures. Swartzman (1979) and Dale and Swartzman (1984) introduced the concept of a process-based model library, where alternative functions for temperature-limited phytoplankton growth were explored within a simulation framework to investigate algal production under thermal loading in Lake Ontario (USA). Generally, the inclusion of a model library of process-based functions can provide a knowledge base from which hypothesis testing or ecosystem behaviours can be investigated.

By adopting the object-oriented modelling approach the addition of a simulation library "tool box" could be easily integrated within the SALMO model. The original SALMO model was programmed using the FORTRAN programming language. This programming style tends to cause difficulties in adding or changing model components, debugging source code and can be non user-friendly in comparison to object-oriented modelling methods. Therefore, this has provoked the need to update and modernise the model by taking advantage of object-oriented programming by means of the Java programming language. Zhang (2006) has implemented the model into a new version called SALMO-

OO using object-oriented design and programming. This has greatly improved the models' flexibility and created an adaptable platform for which new components can be added to the model in an efficient manner. Following these design formats, the integration of the simulation library will enhance the model's applicability to a variety of systems. Thus, the questions and aims of this project are outlined bellow.

1. **Does the application of a phytoplankton growth and grazing simulation library improve the applicability and accuracy of SALMO-OO?**

Algae populations have historically been among the most commonly simulated water quality variables, and are often the target or focus of many lake management models. Therefore, this project proposes to include a variety of alternative process models that simulate phytoplankton growth and grazing by zooplankton, as these processes in particular can have wide-ranging effects on phytoplankton and zooplankton population dynamics, as well as nutrient budgets. Thus, the first project aim is directed towards testing the simulation library for a wide variety of lake datasets, using SALMO as the core model. This will offer an additional means of validation testing with the aim to improve model performance for a wide variety of lake conditions.

2. **Can generic model structures be found using the SALMO-OO simulation library for lakes with different trophic states, climate conditions or morphometry?**

A key characteristic of the SALMO-OO model is its generic nature, which allows the simulation of different lake conditions using the same model structure. The simulation library needs to be designed to maintain this generic characteristic and integrate with the current SALMO-OO structure. The objective is that alternative phytoplankton growth or grazing process models (or both) can replace the original SALMO growth or grazing process models and then be tested to see which combination of process models can best simulate phytoplankton dynamics. Different categories of lakes will be tested using the simulation library to see if a common structure can be found for certain lake characteristics For example, is there a structure within the library that performs the best for deep, eutrophic lakes with Mediterranean climate conditions? Also, to what extent can categories be found, therefore, are there only generic structures based on trophic state, or can generic structures be found for both trophic state and morphometry? If generic model structures can be deduced from alternative process models it is anticipated that the SALMO-OO model can be applied more readily to lakes with different conditions and with more confidence for management scenario analysis.

CHAPTER 2.

LITERATURE REVIEW

Ecosystem and water quality models have evolved over the years in response to several issues (Thomann, 1998):

1. Questions related to environmental decision making have grown in complexity. This has lead to the development of more sophisticate and “smart” ecological software, which include intelligent decision support systems (He *et al.*, 1999).
2. The need for a high level of credibility and scientific veracity has become a focus of modelling outcomes (Håkanson, 2001).
3. Modelling in recent years has increased rapidly in dimensionality (spatial and temporal models, number of state variables etc.) and degree of complexity (incorporation of additional physical, chemical and biological processes).
4. The technological “boom” of the past few decades, especially with the development of faster and more powerful computers at an affordable price, has greatly facilitated model development.

As a result of these forcing issues the current field of ecological modelling covers a wide range of model applications, such as population dynamics, oxygen balances in streams and lakes, eutrophication and models of toxic substances (Håkanson, 2001; Jorgensen, 1995). A brief record is presented within the next section to outline the progress of the ecological modelling discipline, with emphasis on deterministic ecosystem and water quality modelling. Understanding the history of ecological modelling and drawing upon previously gained knowledge and experiences is important. History can teach us to avoid previous flaws in model design and implementation, which allows us to create more rigorous and efficient models.

2.1. The establishment of simulation models in freshwater ecology

The pioneering and evolution of today’s ecological simulation models have spanned several decades. The development of ecological mathematical models began in the early 1920s, with the Streeter – Phelps model (Streeter & Phelps, 1925) of the oxygen balance in a stream and the Lotka – Volterra models (Lotka, 1925) of predator – prey relationships (used predominately for phytoplankton – zooplankton relationships in freshwater systems). In the 1950s and 1960s further development of dynamic population models took place. Since the development of basic computer technology in the early 1960s these models were amongst the first to be simulated in the form of more complex population dynamic models and some river models. The mass balance and empirical modelling approaches of Vollenweider, who modelled the response of lakes to phosphorous enrichment, are notable examples from this period (Vollenweider, 1968).

Substantial advances in the modelling of aquatic systems have occurred since the 1970s, with the development of more elaborate model structures (Cerco, 1999; Jorgensen, 1995). Modern eutrophication modelling was introduced by Di Toro *et al* (1971) in the form of the Sacramento River model, which moved beyond the then-current concepts of modelling biochemical oxygen demand and dissolved oxygen and recognised phytoplankton as a central focus of models of natural waters. This model was also the first realisation of ecosystem-scale modelling (Cerco, 1999), where model complexity increased due to the increased number of interactive state variables and included non-linear interactions between nutrients, phytoplankton and zooplankton (Thomann, 1998). This generation of models became more widely used as decision support tools in environmental management, with emphasis on point sources of waste in heavily stressed systems (Thomann, 1998).

The period between the 1980s and mid-1990s can be identified as the time when there was a rapid expansion in the size of models through the increase in number of state variables, incorporation of hydrodynamic models in multidimensional systems and internalisation of sediment processes. The water quality modelling framework became less subjective and much larger systems were modelled, such as the Great Lakes and Chesapeake Bay (Park *et al.*, 1974; Thomann, 1998). Also, during this period ecological modellers began to realise that the limitations in modelling were not due to computational techniques but with the data and knowledge about ecosystem processes (Jorgensen, 1994), which then lead to a more critical acceptance of models and a more quantitative approach to ecological problems (Jorgensen, 1994). Therefore, a greater understanding of model development occurred, with the standardisation of key procedures in model design and testing (i.e. conceptualisation, selection of parameters, calibration, validation, sensitivity analysis etc. (Harris, 1998; Jorgensen, 1986; Jorgensen, 1988; Jorgensen, 1995)). During this period the pitfalls of ecological modelling became apparent as well. It became clear that ecological models were rigid in comparison with the dynamic and complex characteristic of ecosystems. The feedback mechanisms present within ecosystems were not accounted for in such models, which rendered them incapable of predicting adaptation and structural dynamic changes, resulting in reduced accuracy (Harris, 1998; Jorgensen, 1995).

From the mid-1980s to today rapid progress has occurred to deal with these problems. Topics such as, parameter optimisation, uncertainty analysis, hybridised models, object-oriented models and individual-based models, have become more prolific within the ecological modelling literature in an attempt to create more flexible, realistic models with a greater explanatory and predictive power (Barlund & Tattari, 2001; Håkanson, 2001; Harris, 1994; Krambeck, 1995; Van Duin *et al.*, 2001; van Tongeren, 1995).

2.1.1. Deterministic ecological models

An important factor in ecological modelling is the investigation of the behaviour and relationships of ecological entities through time. These models are termed deterministic models (Håkanson, 2001; Harris, 1998; Jorgensen & de Bernardi, 1998; van Tongeren, 1995) and are used to predict the future outcome or understand past events of an ecosystem, hence they are dynamic in nature (Barciela *et al.*, 1999; Diehl, 2002). One of the most common mathematical methods used to develop deterministic models are differential equations (Hamilton & Schladow, 1997; van Tongeren, 1995; Walas, 1991; Zonneveld, 1998). In terms of ecological modelling, differential equations describe the relationships between pools and fluxes of any type of variable that can change over time or space. In ecological systems differential equations are commonly expressed as rates. Mathematically speaking, differential equations are relations between several variables and their mathematical derivatives (Walas, 1991). For example, a typical phytoplankton biomass differential equation simulating growth and grazing processes may take the form of:

$$\underbrace{\frac{dPHT}{dt}}_{\text{The change in phytoplankton (PHT) biomass over time (dt) =}} = \overbrace{\mathrm{Umax}\ f(\mathrm{P})f(\mathrm{N})f(\mathrm{I})f(\mathrm{T})\,PHT}^{\text{Phytoplankton photosynthesis}} - \overbrace{\mathrm{kr}\ f(\mathrm{T})\,PHT}^{\text{Phytoplankton respiration}} - \underbrace{\mathrm{kgz}\ Z\ f_{\mathrm{gz}}(\mathrm{T})\,PHT}_{\text{Phytoplankton grazing}}$$

In plain language this equation can be read as:

> The change in phytoplankton biomass (PHT) over time (dt) increases with phytoplankton photosynthesis (taking into account the growth rate (Umax) and limitations due to nutrients (N and P), light (I) and temperature (T)), but will decrease with phytoplankton respiration and grazing by zooplankton (based on respiration rates (kr), grazing rates (kgz) and temperature limitations). Therefore, for the biomass to actually increase over time, phytoplankton respiration and grazing processes must be less then the phytoplankton photosynthesis rate.

There are several different types of differential equations, however two common types are described here. The example above is an *ordinary differential equation*. An ordinary differential equation (ODE) involves functions and derivatives of only two variables, one independent (time) and one dependent (phytoplankton biomass) (Walas, 1991). Ordinary differential equations will be the focus of this study, in terms of phytoplankton dynamics. A *partial differential equation* involves partial derivatives of one or more dependent variables with respect to more then one independent variable and functions of some or all of the variables (Walas, 1991). Partial derivatives are represented by the rounded form of delta ∂ or by subscripts. Many ecological models that consider several factors changing over time use partial differential equations. For example, spatial models are based on partial derivatives as the ecological relationships rely on the change in time and space.

2.1.2. Current trends in dynamic freshwater ecosystem and water quality modelling

Although models developed for ecosystem analysis and management have evolved into sophisticated software applications, a great deal of ecological and mathematical theory from twenty or thirty years ago is still scientifically relevant to today's modern ecological models. Early mathematical theory for phytoplankton population dynamics is still used in today's deterministic models for aquatic systems. For example, it is now well known that phytoplankton photosynthetic rate is dependent on light intensity: the further down the water column the less light is available for photosynthesis (Reynolds, 1993). Therefore, a key process in phytoplankton growth models is the mathematical description of the

Photosynthesis-Irradiance (PI) curve (Pahl-Wostl & Imboden, 1990; Reynolds & Irish, 1997; Zonneveld, 1998). One of the most widely used mathematical formulations for the PI curve is from Steele (1962) (Equation 1.1), as it is simple, with one parameter to determine the general shape of the curve (Engqvist & Sjoberg, 1980).

$$f(I) = \left(\frac{I}{Is}\right)\exp\left(1 - \frac{I}{Is}\right) \tag{1.1}$$

where I = light intensity; Is = saturated light intensity

The Steele (1962) function assumes reduction of photosynthesis at intensities both above and below the saturation level (Scavia & Park, 1976). Table 2.1 illustrates the popularity of the Steele (1962) function, which is commonly integrated with the Lambert-Beer function (Equation 1.2) that describes light attenuation with depth, along with several other "classic" light-limiting functions that are still used in today's aquatic models. Different environmental situations may call for the use of different formulae, although some alternative equations give similar responses. However, the choice of mathematical equation to describe a certain ecological process may simply be a matter of preference for the model developer.

$$\mathrm{I} = I_0 * exp\left(-ke * z\right) \tag{1.2}$$

where I_0 = surface light intensity; *ke* = extinction coefficient; z = depth and I = light intensity at depth (z)

The emergence of the Sacramento River model by Di Toro *et al* (1971), the CLEAN model by Park *et al* (1974) and CLEAN's progeny CLEANER by Scavia and Park (1976) initiated the development of complex ecosystem-scale models. Such models of this era modelled the open water environment of lentic systems using differential equations to simulate key state variables (such as phytoplankton, zooplankton, and nutrients) that change over time. Food web interactions between phytoplankton and zooplankton are central to the model's rationale, and phytoplankton growth is controlled by water temperature, irradiance, nutrient concentrations as well as dispersion and advection flows. Phytoplankton is often lumped into one state variable to represent the whole population and environment inputs are generally included as empirical functions, particularly for water temperature and solar radiation. Units for phytoplankton and zooplankton are commonly given as biomass units, such as the phytoplankton dry weight per unit volume or number of organisms per unit volume (Di Toro *et al.*, 1971; Park *et al.*, 1974; Scavia & Park, 1976).

Progress from these "first generation" ecosystem models has included the definition of the epilimnion and hypolimnion layers to simulate stratification in lakes, segregation of phytoplankton into functional groups or separate species, inclusion of two zooplankton groups (herbivores and carnivores) and in some cases fish to expand the food web dynamics, and the use of measured input data to drive the model. Current trends in deterministic ecosystem modelling have focused on the development of coupled hydrodynamic and water quality models and/or spatial models. This approach has evolved in order to obtain a more fundamental understanding and representation of the major physical, chemical and biological processes that affect the biomass of phytoplankton and higher trophic levels in lakes (Cerco, 1999; Hamilton & Schladow, 1997).

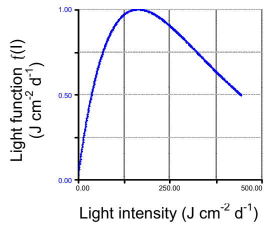
Table 2.1. Commonly used mathematical functions for the description of the phytoplankton photosynthetic rate dependence on light intensity. Graphs are shown only to give a general idea of the shape of each function, which defines the behaviour of the PI-curve.

LIGHT FUNCTIONS

I = Incident light
Ke and β_1 = Extinction coefficient
z = Depth
Is = Saturated light intensity
KI = Half-saturation constant of light absorbance by phytoplankton during photosynthesis
I_0 = Surface light intensity

1. Integrated Beer-Lambert Law and Steele (1962) function

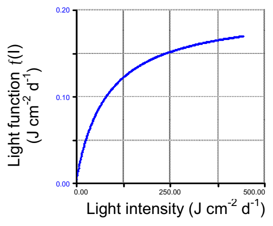
$$f(I)=\frac{2.178}{ke*z}*\exp\left(-\frac{I}{Is}*\exp(-ke*z)\right)-\exp\left(-\frac{I}{Is}\right)*\exp(-ke*z)$$



Arhonditsis & Brett (2005)
Bartell *et al* (1999)
Canale *et al* (1976)
Canu et al (2004)
Childers and McKellar, Jr. (1987)
Collins (1980)
Di Toro et al (1971)
Drago et al (2001)
Hamilton & Schladow (1997)
Hongping and Jianyi (2002)
Lehman *et al* (1974)
Lima et al (2002)
Mesple et al (1995)
Miyanaga (1986)
Park et al (1974)
Robson & Hamilton (2004)
Sagehashi et al (2000)
Varis (1984)
Yang et al (2000)
Yezzi & Uzzo (1979)

3. Jorgensen (1976) light function

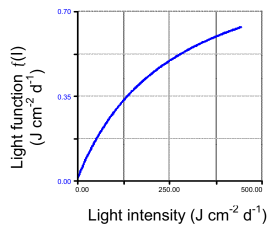
$$f(I)=\ln\left((I_0+KI)/\left(I_0*e^{(-ke*z)}+KI\right)\right)/(ke*z)$$



Krivtsov *et al* (1998)
Chen *et al* (2002)

4. Monod-based light function

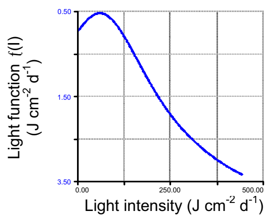
$$f(I)=\frac{I}{(KI+I)} \qquad \text{where } I=\left(I_0*e^{(-ke*z)}\right)$$



Recknagel and Benndorf (1982)
Varis (1993)
Sherman and Webster (1994)

5. Peeter and Eilers (1978) light function

$$f(I)=2*(1+\beta_1)*\frac{x}{x^2+2\beta_1*x+1} \qquad \text{where } x=\frac{I}{Is}$$



Thebault and Salencon (1993)
Bonnet and Poulin (2002)

Generally, most coupled hydrodynamic-water quality models are now very similar in structure and simulation approach. The hydrodynamic components are commonly one-dimensional models, with temporal and depth-averaged dynamics (usually using partial differential equations). However, two- and three-dimensional models have also been developed (Boegman *et al.*, 2001; Muhammetoglu & Soyupak, 2000; Rajar & Cetina, 1997; Tufford & McKellar, 1999). Two-dimensional spatial models operate within longitudinal and vertical boundaries. For example, an application of the two-dimensional model CE-QUAL-W2 to Lake Erie (USA), defined the longitudinal axis by the direction of hydraulic flow and plankton and nutrient gradients, and the vertical axis defined the thermal structure of the lake (Boegman *et al.*, 2001). Three-dimensional models can further include a lateral direction boundary, separating the entire lake environment into sections or boxes and calculate the hydrodynamic and water quality effects within each box, allowing for transport and boundary affects between boxes. This allows the investigation of heterogenous environments, especially when the lake is divided into boxes based on a specific environmental variable (e.g. salinity is a common choice in spatial estuarine models). Common processes simulated within the hydrodynamic components are physical structure of the lake (e.g. lake morphology), thermal stratification, advection and diffusion and particle transport (Al-Khudhairy *et al.*, 2001; Hamilton & Schladow, 1997; Riley & Stefan, 1988).

The water quality or ecological components are generally structured similarly to the hydrodynamic components, in terms of the spatial and temporal boundaries, with a distinct definition of thermal stratification layers (i.e. representation of the epilimnion and hypolimnion). Additionally, sediments are often added to the water quality model as a third layer. The water quality component commonly simulates primary and secondary production, nutrient dynamics and oxygen dynamics. Most of the water quality sub models of today's complex spatial hydrodynamics models do not differ so greatly from those models of phytoplankton dynamics developed by Di Toro *et al.* (1971), Park *et al.* (1974) and Scavia and Park (1976). Some good examples of comprehensive and widely used coupled hydrodynamic-water quality models are the DYRESM model (Hamilton & Schladow, 1997), the DESERT river and EVOLA lake models (Al-Khudhairy *et al.*, 2001), the MINLAKE model (Riley & Stefan, 1988) and the Lake Michigan model (Chen *et al.*, 2002; Ji *et al.*, 2002).

2.1.3. Problems and limitations of current water quality models

Even though there have been advancements in computational technology and an increase in our current knowledge base there are still some fundamental problems that ecological modellers need to deal with. Since models are simplified abstractions of complex ecosystems, their construction and performance will always be subject to many sources of uncertainty. Understanding these issues can aid in the development of more robust and confident models.

2.1.3.1 Data availability and quality

For many model users models are only useful if they have something to say about the real world. Therefore, model developers pay considerable attention to the ability of their models to describe the observed or measured data, which is a representation or instance of a real system. Data is what drives a model. Hence, it is desirable to use a good quality, comprehensive database when running simulations and testing the models structure. The idea of ecological modelling is to simulate the natural environment as closely as possible. Therefore, the data that drives the model needs to represent the natural environment as closely as possible also. However, it is often difficult to find suitable databases for modelling exercises, as data collection is time consuming, expensive and sometimes an inaccurate process. The lack of suitable data not only increases the assumptions within the model but limits the confidence of the modelling results (van Tongeren, 1995). It is often reported in ecological modelling publications that the failure of the model to predict or represent the target site can be greatly attributed to the lack of appropriate data, particularly for model validation testing (Angelini & Petrere Jr., 2000; Drago *et al.*, 2001; Krivtsov *et al.*, 1998; Salençon, 1997). As a consequence, it is becoming more common among ecological research groups to conduct data collection projects in conjunction with the development of ecosystem models, so that limitations from lack of data is reduced (Radford & Blackford, 1996).

2.1.3.2 Complexity

Model complexity refers to the amount of information or number of state variables included in the model structure, and is a problem that is continuously associated with dynamic modelling. The question of how complex to make a model is of primary concern in any modelling study (Costanza & Sklar, 1985). If a model is too complex it can produce a high number of errors, reducing predictive accuracy, and it can be expensive to build and difficult to run and maintain (Håkanson, 1999; Tsang, 1991). Increased model complexity makes it more difficult to understand model behaviour and becomes dependent on assumptions, parameter values and environmental forcing functions (Murray & Parslow, 1999). However, if a model is too simple then important ecological processes may not be accounted for, reducing the descriptive and realistic properties of the model. The phytoplankton population model developed by Baird and Emsley (1999) is a highly complex model, as it has been designed to simulate a variety of phytoplankton species at the cell level, with minimum data requirements. The model relies on parameter values that are able to be measured under laboratory conditions, rather than field conditions as field-determined parameters tend to include a higher degree of uncertainty due to the variability in the natural environment that they are collected in. Thus, the model does not require environmental time series input data and is generally free from calibration. The model also provides a way around the problem of uncertainties in arbitrary choice of functional forms that tend to reduce the predictive power of many plankton models. Therefore, in some cases higher model complexity can be desirable. There are numerous references in the scientific literature discussing and debating the problem of model complexity or optimum model size (Costanza & Sklar, 1985; Håkanson, 1999; Jorgensen, 1988; Jorgensen, 1995).

Many ecologists assume that complex systems (and questions) can only be addressed by complex models (Steel, 1997). The crucial point is that there will never be a model that sufficiently represents an ecological system. A model is to be used as a general surrogate capable of providing answers to questions about the real system (Saloranta *et al.*, 2003; Steel, 1997). There will always be uncertainties associated with our models through the data that is currently available, but our data collection techniques and equipment, and our knowledge of what is required to develop and test robust models is improving so that many of today's model can still provide useful information and predictions (Håkanson, 1999).

2.1.3.3 Duplication of effort: “reinventing” the ecosystem model

When searching through the scientific literature for deterministic lake models it becomes apparent that there are a vast number of models that have been developed, many using the same or similar mathematical functions to represent various ecological processes. This is especially true for primary production sub models. However, many of these models have been developed based on a different logic or rationale and purpose, justifying the need to develop new models from scratch – thus tailoring models to suit very specific research questions or specific lake conditions. Several researchers in the field of agricultural modelling, focusing on crop and plant models, find the same is true for their respective fields (Gauthier *et al.*, 1999; Lemmon & Chuk, 1997; Reynolds & Acock, 1997), and much debate has been made in respect to these issues. Reynolds and Acock (1997) particularly state that many crop and plant models are typically developed by an individual or a small group of scientists, documentation is not sufficiently published in order to replicate the published model, the model is not readily transportable to other platforms for reuse by others, and they are mostly maintained by their developers (if at all). Therefore, such models have limited application, rarely outlive their developers and represent considerable duplication of effort.

Lemmon and Chuk (1997) agree that many crop models are so complex that only their developers really understand them, particularly for models that have evolved over several years with a succession of researchers adding to them. Again, many of the model components are not reusable by others because only the developer can understand the underlying logic and interrelationships of the model and often the code is difficult to understand and adapt by others. Lemmon and Chuk (1997) aptly add that the “model retires when the developer retires (or become administrators)”. Thus, there are numerous models available that are never used by anyone but the developer, which is unfortunate as many of these models are extremely useful and well tested and others would benefit by having access to them.

Gauthier et al (1999) further discuss that most models developed by scientists and graduate students have a very short life span, with little commercial or industrial impact due to the tools and programming languages that ecological modellers use are relatively primitive compared to the state-of-the-art technology employed by engineers and computer scientists. As a result, many of the models produced by small research groups and individuals do not facilitate reuse or exchange of model components and are inflexible.

The comments presented by these researchers are very relevant to problems in the freshwater modelling field. There is an abundance of lake models that have been published for specific applications, using similar mathematical expressions to represent certain ecological processes. For example, all lake water quality models simulate primary production, specifically phytoplankton – zooplankton – nutrient interactions and there are only so many ways that this process can be represented mathematically. It seems that modellers chose not to build on the work of others, preferring to build from the ground-up, which seems to be primarily because existing models are not designed for incremental improvements. Reynolds and Acock (1997) suggest modular and generic models solve these problems.

Modular model design involves separating the key model components into smaller, meaningful sub models, which can be handled independently. Each sub model can then be reused or built upon to fit into other models that require commonly used sub models. For example, a well developed and tested phytoplankton sub model could be reused or adapted for use in lake models, without the need to spend time and effort building model components from scratch. However, ecological modellers tend to use different tools and programming languages that make developing modular model components difficult. Therefore, there is a need for programming approaches that support the evolution of models in a clean and non-destructive way, and that facilitates the creation of reusable software components that have more expressive power then conventional programming languages such as FORTRAN (Gauthier *et al.*, 1999). Object-oriented modelling is a way to achieve these goals, and is discussed in detail in section 2.3.

Generic models, in terms of lake simulations, are those that should be generally applicable to a range of lake conditions. Essentially, these models apply the same structure and constant parameters to simulate different lake conditions, with only environmental data necessary to define the system (Acock & Reddy, 1997; Fitz *et al.*, 1996; Reynolds & Acock, 1997). The principle appeal of generic models is economy (Reynolds & Acock, 1997). Economy of effort is achieved when a generic model provides an alternative to developing ad hoc models for each specific lake site or condition. In addition, there is only one model structure to understand and maintain, and the reduction of input data also results in less calibration of a model, which can be a tedious process (Hamilton & Schladow, 1997; Jorgensen, 1988). Reynolds and Acock (1997) present a detailed description and argument for the use of generic design in the development of plant and crop models. Specifically, Reynolds and Acock (1997) state:

1. A generic model should be suitable for application to ecosystems or a target group by the use of different model parameters or different modules.
2. A good generic design must be able to simulate functionally similar, yet different systems, by the addition (or subtraction) of modules.
3. Individual modules should be readily recognised by experts in the field as separate processes of the system under study. The purpose of each module should be readily apparent. Modules that combine several functions not normally considered together may be more difficult to parameterise for a new system.

The same design principles outlined in their paper can be applied to the development of lake models. Indeed, the generic lake ecosystem model SALMO (Benndorf & Recknagel, 1982; Recknagel & Benndorf, 1982) has incorporated many of these points successfully and is the main reason for choosing this particular model as the core model of this study. A detailed description of SALMO is given in Chapter 3 section 3.1. Modular generic models offer a flexible platform to test alternative hypotheses by replacing modules and can be easily updated as our knowledge improves (Reynolds & Acock, 1997).

2.2. Process-based model libraries for the simulation of phytoplankton dynamics

Deterministic ecosystem model development can be a subjective process that is dependent on individual judgements, decisions and experience, making model development “half science, half art”, with ample room for differences in opinions (Haag & Kaupenjohann, 2001; Zonneveld, 1998). This is to be expected as ecosystems are complex, and some processes are not well understood or quantifiable. However, deterministic ecosystem models can be robust tools for the examination of model behaviour under several alternative hypotheses. Swartzman (1979) proposed the use of simulation model "libraries" to explore plankton dynamics under extreme temperature conditions of nuclear power plant ponds by the investigation of alternative model structures. Swartzman's "model library" concept is based on the deductive modelling approach. Therefore, exploring alternative process-based models (i.e. deterministic models that are constructed using ordinary differential equations) that are formulated to represent real plankton processes in nature. The deductive modelling approach is considered as being knowledge driven, which can give more explanatory power as the models structure is based on the best understanding of the inner workings of the simulated ecosystem.

Dale and Swartzman (1984) adapted the process-based "model library" outlined in Swartzman (1979) to investigate the effect of thermal loading on phytoplankton production in Lake Ontario. The core model is a simplified ecosystem model of Lake Ontario by Thomann *et al* (1975), describing major state variables such as phytoplankton, zooplankton, detritus and nutrients (particularly nitrogen and phosphorous). The model library consisted of a number of phytoplankton models drawn from the scientific literature with a selection of temperature dependent functions coded into the model library. For example, phytoplankton growth is limited by temperature and different hypothesised effects of this process can be expressed mathematically in various ways. Temperature limited growth can take the shape of a linear, optimal or an exponential function (Figure 2.1), and can give an entirely different response that will affect the way phytoplankton growth is simulated by the model. Thus, alternative model structures tested by this approach can give an increased understanding on the underlying principles for which the model is based. However, often the predictive capacity of process-based models can be compromised due to the uncertainties and assumptions inherent in these types of models (as was discussed in section 2.1.3).

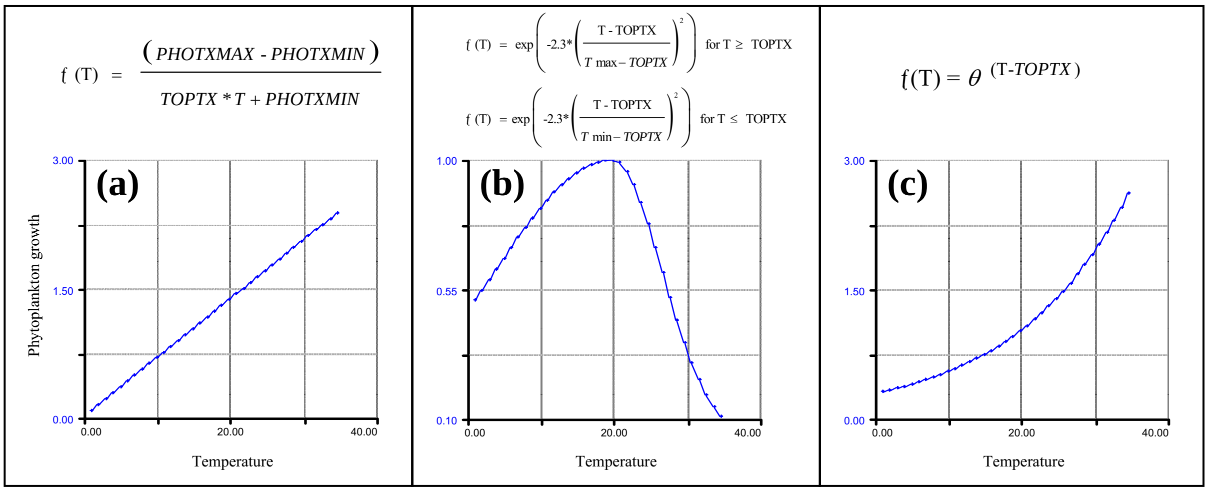


Figure 2.1. Different mathematical functions for temperature-limited phytoplankton growth, such as (a) linear, (b) optimal and (c) exponential functions. Example equations, which give the shape of the corresponding graphs, are also given.

Another way to address predictive power is to utilise the inductive modelling approach. Inductive modelling is data driven, where models are induced from measured time-series data by bio-inspired computational techniques, such as artificial neural networks and evolutionary algorithms. Such methods have a high predictive power but rely heavily on the availability of high-quality datasets. A comparable example of the inductive modelling approach to exploring alternative phytoplankton models is the LAGRAMGE model family (Atanasova *et al.*, 2006; Dzeroski & Todorovski, 2003; Todorovski, 2003), which involves heuristically exploring various model structures and parameter values (organised from process-based functions or "domain knowledge") and comparing each with measured data to find the best fitting model. The procedure of model induction and simulation with Lagramge is presented in Figure 2.2.

The domain knowledge is described within a "knowledge library", and the user specifies important variables and processes that take place in the observed system (model task specification step in the Largramge set up). Thus, Lagramge is more appropriately described as a combination of inductive and deductive modelling, which is used to direct the process of induction from real data. For each process defined by the user in the task specification the algorithm first checks the consistencies of the types of the variables involved in the process. Then, the algorithm matches the process from task specification against the generic processes in the knowledge library.

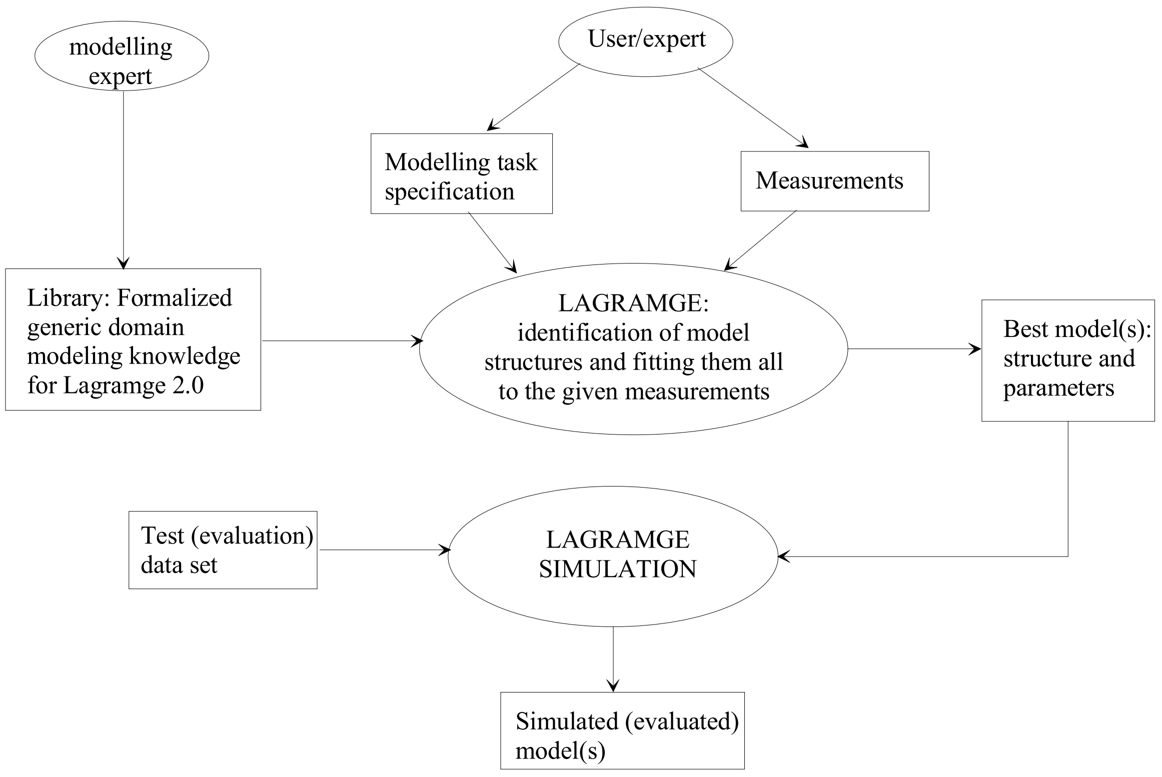

modelling expert
User/expert
Modelling task specification
Measurements
Library: Formalized generic domain modeling knowledge for Lagramge 2.0
LAGRAMGE: identification of model structures and fitting them all to the given measurements
Best model(s): structure and parameters
Test (evaluation) data set
LAGRAMGE SIMULATION
Simulated (evaluated) model(s)

Figure 2.2. Largramge automated modelling framework based on the integration of domain-specific modelling knowledge in the process of equation discovery (Atanasova *et al.*, 2006).

Once the match is found, a generic process definition is used to obtain a list of all alternative mathematical models (in the form of ODEs) for the particular processes. Thus, the result of this match is a list of specific mathematical model structures that can be used to model the processes specified in the task specification. At the same time Lagramge takes all model structures and matches each of them to the calibration data by fitting the values of the constant parameters, (Lagramge performs non-linear optimisation) according to the measurements. The models are ranked by their error on the training data set. The model with the lowest error is considered as the best model for a given conceptual model (task specification) and data set.

In this case, LAGRAMGE was applied to data from Lake Kasumigaura (Japan) by utilising the knowledge library of lake process equations to discover predictive ODEs for chlorophyll *a*. LAGRAMGE heuristically searches through the different models offered by the library and tests each of them with measured data after fitting constant parameter values (Atanasova *et al.*, 2005). Similarly to Swartzman (1979), the process-based functions contained within the library are taken from literature models developed for lakes and can be assembled to different levels of ecosystem structures, such as the simple Vollenweider model or a complex ecosystem model such as SALMO (Recknagel & Benndorf, 1982). Further details about LAGRAMGE can be found in Todorovski (2003). The LAGRAMGE model found 7 ODEs that best predicted chlorophyll *a*; one equation for each of the years tested. This demonstrates the dynamic and somewhat difficult nature of predicting chlorophyll *a*. An attempt was made to discover a generic chlorophyll *a* ODE for all years, however, the ODE that was found lacked predictive power in simulating chlorophyll *a*, although the model described general trends.

Both approaches have their advantages and limitations. The deductive, process-based methods have the advantage of giving understanding to what processes and feedback mechanisms are occurring, and provide a simulation framework where current knowledge of ecosystem dynamics can be explored. The inductive modelling approach, particularly grammar based models such as evolutionary algorithms, allow the development of automated "search engines" in order to discover functions, such as ODEs, or rule-sets to predict ecological variables and develop models with a high predictive power. However, trying to find generic model structures or rules to describe different lake conditions can prove to be extremely difficult without considering the processes occurring within the ecosystem. Therefore, the deductive, process-based modelling approach still remains the most favourable method to explore ecosystem dynamics and causally understand model structures in a generic manner.

Since Swartzman (1979) first proposed his "model library" concept in the late 1970s, there have been few advances in testing alternative model structures within a simulation framework to simulate a wide variety of environmental conditions. Much has to do with the lack of computational tools that were available at the time. However, the current computational methodologies that are available to ecologists are vast, easy to obtain, cheap to run and constantly evolving into better products. Object-oriented modelling is a novel computational methodology that is becoming increasingly popular with ecologists in developing complex simulation models, and is an appropriate modelling method to apply to the development of a generic, process-based simulation library for lake ecosystem modelling, which is the focus of this study.

2.3. Adopting the object-oriented paradigm for ecological modelling

There is a need for programming approaches and software design systems that support the evolution of ecological software so that models do not become redundant, and thus knowledge and effort is not lost (Acock & Reddy, 1997; Gauthier *et al.*, 1999; Sequeira *et al.*, 1991). Object-oriented programming and software design techniques offer a new approach to model development and propagation. A common misconception among researchers is that adopting object-oriented programming and design requires the discarding of all previous programs and code. Object-oriented programming is more of an evolutionary trend rather than a drastically new and different programming method (Sequeira *et al.*, 1997). Nevertheless, ecologists have been reluctant to implement object-oriented models for a number of reasons: there has been a low level of access to user-friendly object-oriented software, researchers must learn a new technique and set of tools and many new modelling projects often addresses not the building of new models from scratch, but rather the improvement of subcomponents of existing models (Gauthier *et al.*, 1999; Sequeira *et al.*, 1997).

These are acceptable reasons for maintaining the status quo. However, with the increased access to Java™ technologies (Sun Microsystems, 2005) this has improved the availability of true object-oriented software development systems and addresses many of the issues that ecological modellers have in implementing object-oriented models. Many pitfalls identified by other ecological modellers in adopting object-oriented programming can be solved using the Java programming language (Acock & Reddy, 1997; Ferreira, 1995; Sequeira *et al.*, 1997). The Java programming language is fast becoming the preferred language for the development of object-oriented models due to its rigorous adherence to the object-oriented paradigm (Baskent *et al.*, 2001; Ellis *et al.*, 1998; Gauthier *et al.*, 1999; Nguyen *et al.*, 2000). Many conventional programming languages can be used to develop object-oriented models, however, circumvention of true object-oriented concepts can be made, which can produce problems with model logic. Models built with Java technologies are portable and can be operated on any platform (e.g. Windows, Mac OS or UNIX) that has a Java virtual machine. Java also allows models to be embedded in traditional web pages, which can facilitate model sharing and data exchange (Nguyen *et al.*, 2000). An additional advantage of Java is that it is freely available at no cost to the modeller, whereas, many other modelling software systems need to be purchased and can be quite expensive.

Before we ask the question "why adopt the object-oriented paradigm to the development of ecological models?" we need to know what the object-oriented paradigm is and how do we apply it to the simulation of ecological systems. This next section attempts to explain and illustrate the key concepts of the object-oriented paradigm. For more detailed information see Booch (1991) and Gilbert and McCarty (1998).

2.3.1 Key principles in object-oriented modelling

Booch (1991) formally defines object-oriented modelling as:

> "A method of implementation in which models are organised as cooperative collections of objects, each of which represents an instance of some class, and whose classes are all members of a hierarchy of classes united via inheritance relationships".

Therefore, object-oriented modelling can be considered as modular design. Modularity is the essence of the object-oriented paradigm, where a complex problem is broken down into multiple small and simplified modules (He *et al.*, 1999; Sequeira *et al.*, 1991). For example, a complex lake ecosystem model can be modularised by categorising similar ecological processes such as dissolved substances, primary production, secondary production and physical properties. This develops a hierarchical structure among ecological entities, which can be further modularised into categories with a finer level of resolution defined by individual classes.

A *class* is a template for the creation (*instantiation*) of *objects* that share a common structure and behaviour (Baskent *et al.*, 2001; Booch, 1991; Gilbert & McCarty, 1998; Lemmon & Chuk, 1997). A class refers to an abstract concept with no real representation. An object is the specific, real instance or representation of the class concept. Therefore, a class has to be defined with the properties that describe an entity before an object can be created to represent that entity and interact with other objects. This is known as *abstraction*. Abstraction focuses on the external view of an object, separating an object's essential behaviour from its implementation (Sequeira *et al.*, 1991).

Objects belonging to the same class have a common structure and behaviour. The structure or state of an object is specified through the definition of *attributes* (variables or data), and the behaviour is specified through the inclusion of *methods* (functions) that can manipulate the state of an object (Gauthier *et al.*, 1999). Attributes and methods are grouped together within each object, which is known as *encapsulation* or *data hiding.* Where abstraction defines the external view of an object, encapsulation defines the internal structure of an object. Encapsulation occurs when the internal state of variables inside an object is not accessible outside the object except via messages to its methods. Therefore, internal data structures and functions may be modified as required without affecting the performance, logic or algorithmic efficiency of the rest of the program (Lemmon & Chuk, 1997; Sequeira *et al.*, 1997).

A great advantage of using classes becomes apparent when the concept of *inheritance* is introduced. Inheritance in object-oriented modelling is analogous to the concept of taxonomy in biology, where broad categories of organisms with similar attributes are further divided into more detailed and specific categories, which inherit attributes from higher in the hierarchy. The same hierarchical structure holds for object-oriented classes. We can treat different classes of objects as descendents of a common ancestor and need only add those particular features that have changed within the context of the model to the new descendents (Silvert, 1993). Figure 2.3 demonstrates a simple hierarchy structure for a primary production model. The first class in the hierarchy in Figure 2.3 is the `StateVar` class, which is known as the super or parent class. The `Phytoplankton` class is associated with the super class through inheritance links. Therefore, all the data and methods described by the `StateVar` class will be accessible to the `Phytoplankton` class and any classes or objects descending from it. For example, the `getData()` method in the super class is available for use by the `Phytoplankton` class to access its input data such as water temperature or solar radiation. The `getData()` method is also available to the `DissolvedMatter` class as this class is also a descendent of the `StateVar` super class. Inheritance provides a powerful mechanism for organising and structuring simulation models and allows the reuse of a class's behaviour in the definition of new classes (Baskent *et al.*, 2001; Sequeira *et al.*, 1997).

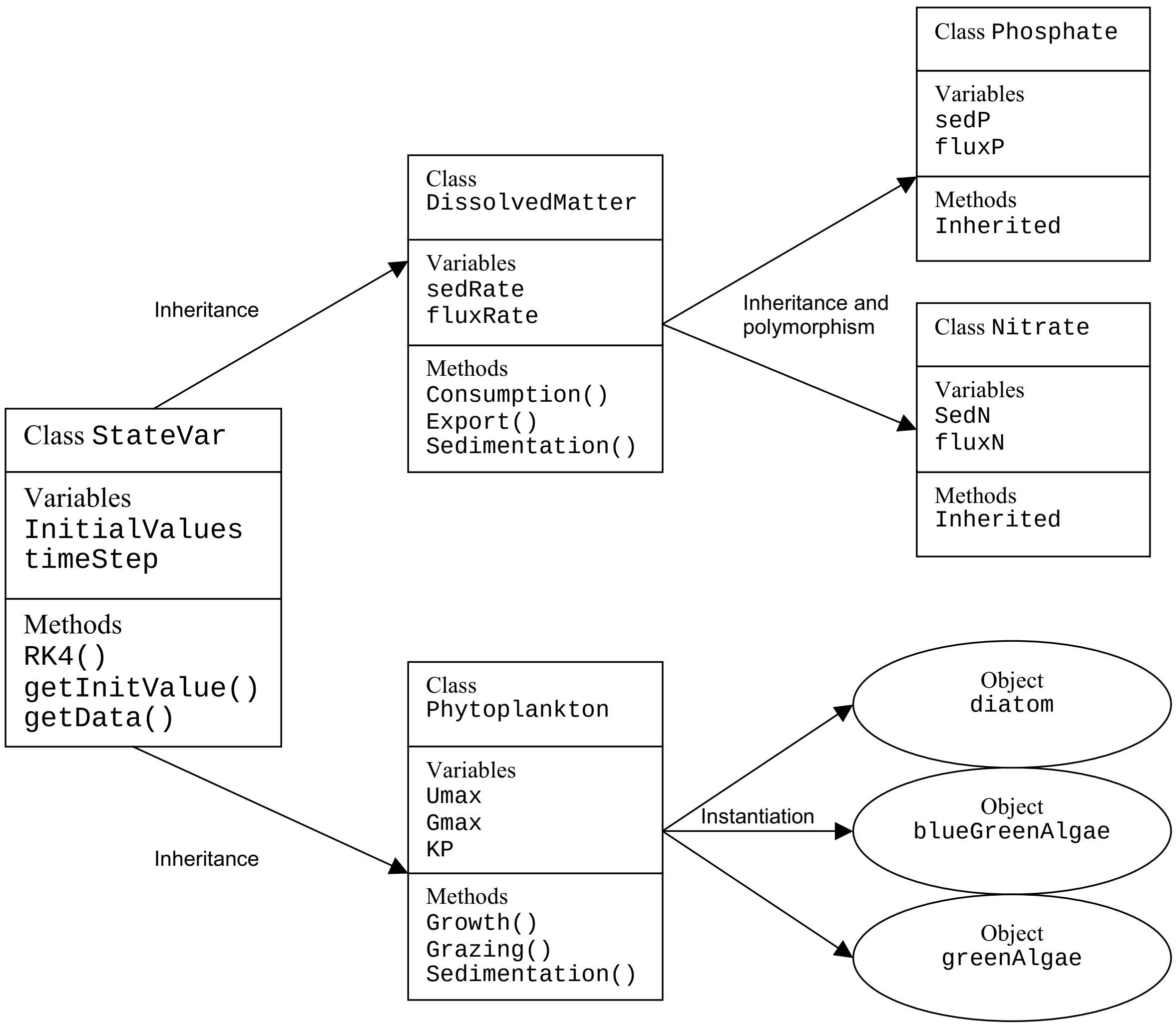


Figure 2.3. An example of the object-oriented paradigm applied to a simple primary production model. The `DissolvedMatter` and the `Phytoplankton` class are linked via inheritance relationships to their super class (`StateVar`). Classes `Phosphate` and `Nitrate` show further inheritance links to the `DissolvedMatter` class. Methods described by the `DissolvedMatter` class are available to the `Phosphate` and `Nitrate` classes. The instantiation of objects is shown by the `Phytoplankton` class, which creates objects for three functional groups of phytoplankton (`diatom`, `blueGreenAlgae` and `greenAlgae`). These objects encapsulate the data described by the Phytoplankton class.

Polymorphism is a type of inheritance concept. Polymorphism refers to the ability to call the same method name in different derived classes (Gauthier *et al.*, 1999). Therefore, the same message is sent, but a different response is invoked in different objects. For example, Figure 2.3 illustrates the structure and behaviour of two classes: `DissolvedMatter` and `Phytoplankton`. Both classes have a method called `sedimentation(),` which defines a different mathematical function for loss due to sedimentation for each class. When the `blueGreenAlgae` object calls the `sedimentation()` method a message will be sent to the `sedimentation()` method contained in the `Phytoplankton` class. Conversely, when the `phosphate` object needs to calculate its sedimentation rate it will invoke the `sedimentation()` method from the `DissolvedMatter` class. Polymorphism reduces the need to write code repeatedly and allows the reuse of programming components, which saves time and effort (Bian, 2003).

2.3.2 Advantages of adopting the object-oriented approach to ecological modelling

The limitations of procedural modelling

In the past, ecological models have largely been developed using procedural programming approaches. The primary mechanism in procedural languages (e.g. C, FORTRAN, Pascal, Ada) is the procedure, sometimes called a subroutine, where program execution is regarded as a partially ordered sequence of procedure calls manipulating data structures (Gilbert & McCarty, 1998; Jansson & Moon, 2001; Muetzelfeldt & Massheder, 2003; Sequeira *et al.*, 1991). Procedures are based on structured, top-down design, which receive and process data and return results. Because of this, the shape of a system developed using procedural design almost always takes the form of a tree, with centralised control invested in the top or main module (Gilbert & McCarty, 1998; Jansson & Moon, 2001; Muetzelfeldt & Massheder, 2003), which can become complicated and difficult to keep track of how functions relate (Baskent *et al.*, 2001; Sequeira *et al.*, 1991).

A simple example of procedural programming would be to look at the functions needed to pass a saltshaker across a dinner table. In real life, a person who asks for the saltshaker to be passed to them from across the dinner table would simply ask for the saltshaker, and receive it from the other person. However, to implement such an action using procedural programming the person requiring the saltshaker would have to say "Please remove your right hand from your wine glass, move it to the left until it contacts the saltshaker, grasp it, lift it from the table, move it in a natural arc in my direction, stop when it contacts my hand, wait until my hand has closed around it, and then release it". The need to recognise each step to implement a specific action is the basic configuration of procedural modelling. The procedural approach separates data and procedures that manipulate data. Therefore, one procedure can manipulate more than one data structure and one data structure can be manipulated by more than one procedure (Figure 2.4). Data structures that are of central importance to the overall program can be used by many procedures, causing interdependence, which is not a natural way to model real world problems (Baskent *et al.*, 2001). The use of procedural programming is very detailed, which can potentially cause many problems when developing complex models, which are often difficult to understand, build, refine and debug when conventional programming methods are used (Acock &

Reddy, 1997; Baskent *et al.*, 2001; Williams, 1995). Booch (1991) states that 60% of model development costs lie in maintenance of models developed using conventional methods. Therefore, there is a need for modelling approaches that support the evolution of computer applications that facilitate and encourage the creation of reusable model components that have more flexibility and more expressive power. The object-oriented paradigm offers new insights for both the programming and conceptualisation of complex simulation models, and resolves many of the problems associated with procedural languages.

Object-oriented modelling

The flexibility associated with modularity, abstraction and encapsulation provides many advantages when developing ecological models. A model that has been created using procedural programming would normally be rigid, with a subjective categorisation of state variables and processes, which is designed by the modeller. One modeller's idea of categorising a system may differ from another. Thus, processes from one model must usually be reprogrammed from the original mathematical equations before being tested in another model (Acock & Reddy, 1997). Using object-oriented design the modularisation and abstraction of state variables and process effectively de-clutters the structure of the model. The concept of an object remains the same throughout the development process and is therefore, standardised (Dawson & Swatman, 1999; Sequeira *et al.*, 1997). Thus, a class or object needs to be defined only once during model development. Not only is the model easier to navigate around and learn, it reduces the propagation and time needed to fix errors and facilitates the understanding of model logic (Acock & Reddy, 1997; Dawson & Swatman, 1999; Lemmon & Chuk, 1997).

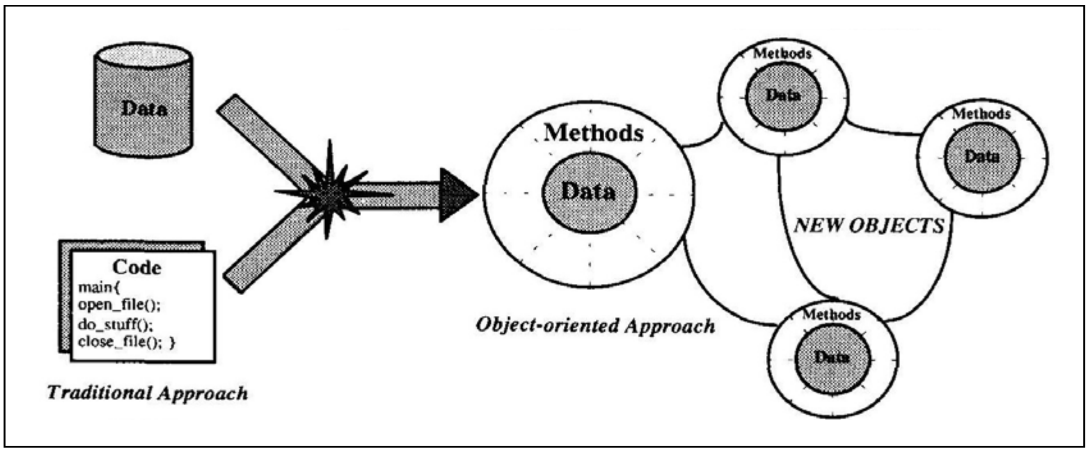


Figure 2.4. A key difference between object-oriented programming and the conventional programming approach is through encapsulation (Khebbal & Shamhong, 1995).

The ability to encapsulate an object's data and methods is a key concept that distinguishes object-oriented programming from conventional programming approaches (Figure 2.4). An advantage of encapsulation is the creation of independent objects, which can be inserted in different programs with little or no modification (Ferreira, 1995; Reynolds & Acock, 1997). If an object must be adapted, rather then altering the code a descendant can be created. Debugging is then limited to any new methods, since the base-type will already have been tested (Ferreira, 1995). Encapsulation gives the modeller better control over data and creates more independence within the models' structure, creating an easier way to introduce new knowledge into the model without affecting the existing structure (He *et al.*, 1999; Lemmon & Chuk, 1997). This leads to maximum reuse of code and thus reduces the complexity of the model and facilitates the development of new model components.

Object-oriented modelling can be an exceptional tool for building hybridised model systems (Khebbal & Shamhong, 1995). The use of objects as a representation scheme is particularly suitable for hybrid models because objects can represent different information processing techniques, which communicate via a well-defined message protocol. Message passing provides the basis of the most flexible form of dynamic linking in an object-oriented hybrid system (Khebbal & Shamhong, 1995). At the lowest level, a message is simply data passed from one object to the next. Combined with queues and the ability to send messages asynchronously, each object in a hybrid application could process a stream of data independently of other objects (Khebbal & Shamhong, 1995). Object-oriented programming can be used to integrate machine learning techniques, such as neural networks (Meyer, 1990; Whigham & Recknagel, 2001) and genetic and evolutionary algorithms (Bobbin & Recknagel, 2001; Whigham & Recknagel, 2001) to process-based models. The integration of individual-based models and Geographic Information System (GIS) applications and databases via an object-oriented framework is widely used (Chau & Chen, 2001; Chau *et al.*, 2002). Some key ecological examples are the ECLIPSS software environment for the investigation of ozone effects on forest ecosystems (Woodbury *et al.*, 2002); TIGMON model for the investigation of tiger/human interactions in forest systems (Ahearn *et al.*, 2001); the WILDSPACE decision support system for wildlife population and habitat investigations (Wong *et al.*, 2003); PRYSM (Power and Reservoir System Model) and the conceptual GIS data model developed by Mckinney and Cai (2002) for water resource management.

One of the key advantages of object-oriented modelling is its ability to represent the natural environmental or real world more closely. Martin and Odell (1992) state that models built with object-oriented programming reflect reality more naturally then models built using conventional programming languages. Sequeira *et al* (1991) and Bian (2003) both state that a key strength in the object-oriented paradigm is that the real world is represented in a manner that closely corresponds to the reasoning patterns that a human normally uses to solve a problem. Many ecological modellers have stated that the object-oriented paradigm is well suited to simulating ecological systems because the paradigm provides a greater conceptual approximation between natural ecosystem and interacting objects, compared to procedural methods (Baskent *et al.*, 2001; Ferreira, 1995; Mooij & Boersma, 1996; Sequeira *et al.*, 1997; Silvert, 1993). The object-oriented concept of inheritance is particularly attractive to ecological modellers, as the concept closely resembles many ecological principles. Class hierarchy is similar to taxonomy in biology. The concept of association (encapsulation and message passing) is close to the principles of social relationships between organisms and the concept of aggregation is similar to the principles of ecological assemblies (Bian, 2003).

Object-oriented programming and design is not a simple concept and it is not easily mastered. It is a completely different way of thinking about software design and the transition from conventional modelling approaches to object-oriented modelling is daunting, especially for ecologists who are new to the ecological modelling field. However, when the benefits of adopting the object-oriented approach outweighs the difficulties associated with conventional techniques, which is often the case for large, complex ecosystem models, the object-oriented modelling approach is best.

2.3.3 Application of object-oriented modelling to ecology

The concept of object-oriented programming and modelling has been utilised by computer science and business industries for many years, particularly for developing operating systems, graphical interfaces, computer aided design and optimisation software and database management (Dawson & Swatman, 1999; Maryanski *et al.*, 1986; Sequeira *et al.*, 1991). During the 1990s the use of the object-oriented paradigm began to emerge in ecological model design and implementation, with a dominance in the fields of agro-ecosystems (Acock & Reddy, 1997; Gauthier *et al.*, 1999; Sequeira *et al.*, 1997; Sequeira *et al.*, 1991) and landscape ecology (Ahearn *et al.*, 2001; Baskent *et al.*, 2001; He *et al.*, 1999; Woodbury *et al.*, 2002). A number of ecological models have been published that have chosen to adopt object-oriented design concepts to building vastly complex ecological models, particularly for the development of management tools. Such models not only have an extensive ecological component, but also include modules that interface with a user-friendly Graphical User Interface (GUI) (He *et al.*, 1999), modules that interface with database management systems (Woodbury *et al.*, 2002), link to expert systems for decision-support (Wong *et al.*, 2003) and even allow other components such as eco-economic evaluators (Bergez *et al.*, 2001) and risk assessments (Sydelko *et al.*, 2001). Many others have adopted the object-oriented approach for the development of hybridised models, particularly for individual-based and landscape modelling linked to GIS (Ahearn *et al.*, 2001; Bian, 2000; He *et al.*, 1999; Herve *et al.*, 2002; Jansson & Moon, 2001; Lorek & Sonnenschein, 1998; McKinney & Cai, 2002; Mooij & Boersma, 1996; Wong *et al.*, 2003). The object-oriented approach represents individuals more efficiently because they are represented as independent objects that behave similarly to real ecological entities. LANDIS, a forest landscape model that simulates interactions of large landscape processes and forest successional dynamics using satellite data (He *et al.*, 1999), is a comprehensive example of an object-oriented spatial landscape model. Other examples of hybridised object-oriented and individual-based models include the forest management decision support system developed by Baskent *et al.* (2001); SHALOM landscape model (Ziv, 1998), which is able to simulate various landscape scales, such as species, patch, populations and communities; the hierarchical patch dynamics model, developed by (Poole, 2002), simulates the influence of landscape processes on the structure and function of lotic ecosystems; and ECLIPSS (Woodbury *et al.*, 2002), a library of reusable and interchangeable terrestrial ecological sub-models.

In the field of agro-ecosystems, the object-oriented approach has been widely adopted for the development of management models and simulation frameworks. The Generic Plant Simulation Framework (GPSF), developed by Gauthier *et al*, 1999, is a generic object-oriented framework used to construct and implement crop growth and development models. At present the model framework is designed for continuously growing greenhouse crops, such as tomatoes, cucumbers and melons, but can be adapted for any crop species. The model has been redesigned from the TOMGRO model, which was originally programmed in Fortran procedural code, which is notoriously rigid. Adopting object-oriented modelling and design has improved the flexibility of the framework as new sub-models can be built without compromising the integrity of the existing models and can coexist within the same software environment. Then GPM system (Generic Plant Modelling), developed by Sequeira *et al.*, 1997, is another generic modelling platform for various crop plant species. The GPSF and GPM systems are very similar in concept and design. For example, both models separate plant parts and processes into two distinct hierarchies; one class to define plant physiology and another to define plant processes. Each of the two classes is also categorised similarly. The GPM system is a good example of building object-oriented models by aggregating sub-models and under some circumstances the interchanging of two sub-models. The revised GPSF showed that the implementation of a new program can take a fraction of the time to design compared to then the first model using conventional methods and much of the existing code can be reused.

The application of object-oriented modelling to aquatic ecosystems has taken on many different forms. Many of these deal with hybridised model frameworks, either linking GIS with dynamic management models (Freda & Jamieson, 1996; McKinney & Cai, 2002; Reitsma *et al.*, 1994; Tisdale, 1996) or integrating artificial intelligent expert systems with numerical models for assistance in selecting and manipulating water quality models (Chau & Chen, 2001; Chau *et al.*, 2002). Several individual-based models for the simulation of spatial aquatic environments or mobile aquatic organisms have been developed with object-oriented technologies. These individual-based models are particularly interesting to the development of freshwater ecosystem scale models. Mooji and Boersma (1996) discuss the generic model OSIRIS (object-oriented simulation framework for individual-based simulations), with an example given for the simulation of zooplankton *Daphnia* population dynamics. Bian (2000) combines an object-oriented GIS framework with an individual-based model to represent fish as objects and their habitat and environment as GIS raster cells and spatial data. Bain's model is a particularly comprehensive example of how all three modelling methods (object-orientation, GIS and individual-based models) can be integrated into a single simulation environment, which demonstrates the benefits of adopting object-oriented design and technologies for the development of complex, multi-optional models.

Object-oriented modelling has been applied to a small number of dynamic, process-based models for estuarine systems, (Coffaro & Sfriso, 1997; d'Oultremont & Gutierrez, 2002a, 2002b; Williams, 1995), however, there are currently no published examples of applications applying the object-oriented modelling approach to freshwater-specific ecosystem models. In spite of this, there are many processes in marine systems, which are comparable to freshwater environments, such as primary productivity and nutrient cycling. The analysis of existing object-oriented models for estuarine systems can present a starting point for the implementation of freshwater models into an object-oriented design. These models are similar in structure and simulation technique due to the use of differential equations for representing the dynamics of key aquatic processes. For example, the ECOWIN model (Ferreira, 1995) and a model created by Coffaro & Sfriso (1997) of the macroalgae *Ulva rigida* in the Lagoon of Venice, Italy have both developed a similar hierarchical structure. Both models start their hierarchical tree with a super class that defines a general description of a differential equation, the integration method, initial values and other general information that all descending classes will use. Further abstraction takes place by defining general ecological processes. This is where subtle differences between model structures occur, mainly due to the differences in the purpose for model development and use. The ECOWIN model has been developed as a general aquatic ecosystem model with the ability to perform scenario analysis for management of pollution and urbanisation. Therefore, several key classes are needed to describe the ecosystem, with the definition of more detailed classes further down the hierarchical tree (Figure 2.5). The *Ulva rigida* model defines fewer classes, as the purpose of the model is to understand the growth dynamics of the macroalgae and to evaluate the relative importance of factors controlling their growth (*Figure* 2.5). The difference between the development objectives of the ECOWIN and the *Ulva rigida* models has essentially determined the level of abstraction and inheritance that takes place.

The main appeal of object-oriented modelling for many ecologists is the modularisation of a complex problem into a manageable and logical structure that is easier to translate into a simulation model. Inheritance concepts reduce the need to re-write code repeatedly and are well suited to modelling the hierarchical nature of ecological systems. Encapsulation and abstraction allows a model to be more representative of the real world, increases model flexibility and improves model extendibility. Adopting the object-oriented approach can save development and programming time, due to the modularisation and abstraction of classes that can be interchanged within the same simulation environment. This will also facilitate model sharing within the ecological modelling community, which is an issue that greatly interests ecologists who are now more then ever involved in multidisciplinary modelling projects (Radford & Blackford, 1996).

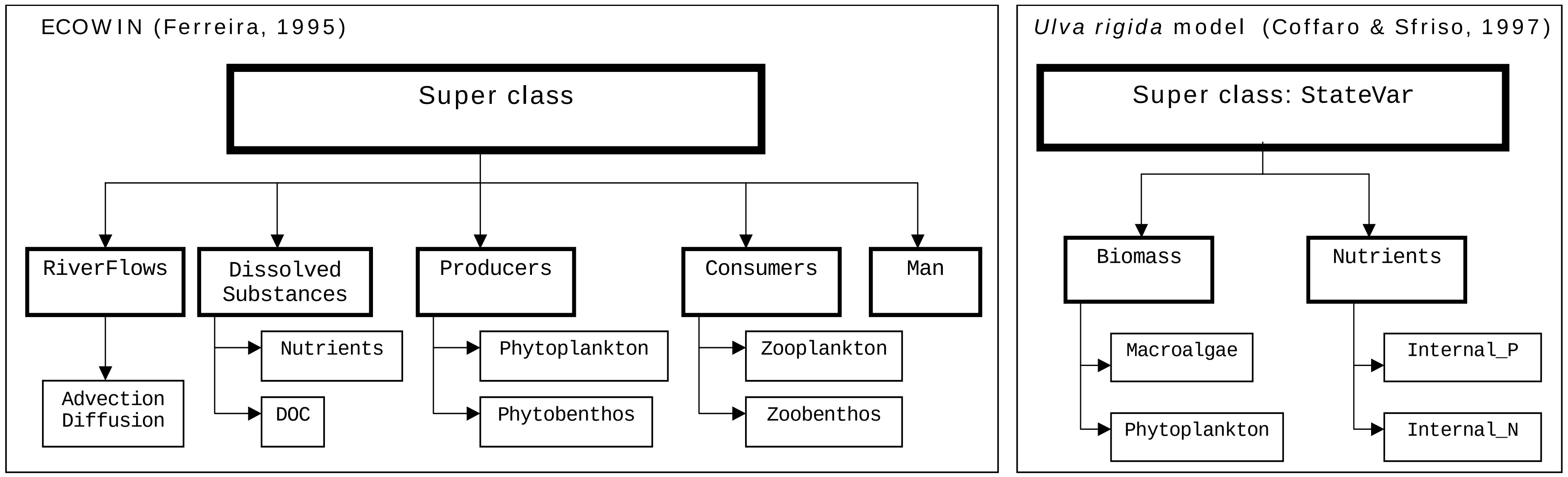


Figure 2.5. Hierarchical structure of two object-oriented dynamic aquatic models. Boxes represent model classes, all descendent from a parent super class. The ECOWIN model (Ferreira, 1995) is a general ecosystem model for estuarine systems and the *Ulva rigida* model, developed by Coffaro and Sfriso (1997), was developed to investigate growth dynamics of macroalgae. Differences in modelling objectives cause modularisation and abstraction of classes to occur at different levels. The ECOWIN model classes cover a more diverse perspective, whereas the *Ulva rigida* model classes are more specific to the factors that influence macroalgae physiology.

CHAPTER 3.

MATERIALS AND METHODS

3.1. SALMO: Dynamic Water Quality Model for Freshwater Lakes

The lake ecosystem model SALMO (Simulation by means of an Analytical Lake MOdel) was developed by Benndorf (1979) and Recknagel (1979) for scientific and practical water management applications, prompted by the increase in eutrophication problems in European lakes and reservoirs. Over the past two decades the model has been expanded, and has been successfully applied to many widely differing lake conditions, overseas and in Australia.

3.1.1 Description of SALMO

SALMO has been developed as a two-layer model that simulates the epilimnion and hypolimnion of stratified water bodies. The model assumes that the concentrations of matter or biomass are homogenously distributed within the water column or layer. The state variables currently simulated by SALMO-OO include orthophosphate, dissolved inorganic nitrogen, phytoplankton, zooplankton, detritus, and dissolved oxygen (Figure 3.1). Phytoplankton is further divided into three state variables for the simulation of three functional groups: blue-green algae (e.g. *Oscillatoria redekei*), diatoms (e.g. *Asterionella formosa*) and green algae (e.g. *Scenedesmus quadricauda*). The zooplankton state variable represents herbivorous zooplankton only (e.g. *Daphnia galeata*). Fish and predatory zooplankton are considered indirectly by incorporating their influence on the zooplankton mortality rate. Each of the state variables is described by an ordinary differential equation, which is solved by the fourth order Runge-Kutta method. The model uses daily time steps.

SALMO has been designed as a generic model thus, the majority of the model structure has been generalised as much as possible. Through rigorous and multiple sensitivity analyses using data from a range of conditions, all parameters values can be kept constant for all simulations (Recknagel, 1984). The model includes several driving variables, which require time-series data to define important environmental conditions. These include:

- Incident solar radiation (J cm^{-2} d^{-1})
- Water temperature of mixed and stratified layers (°C)
- External concentration of nutrients entering the system (g Nm^{-3}, mg P m^{-3}),
- Water inflow and outflow from mixed and stratified layers (m^3 d^{-1}),
- Mean and maximum water depth (m)
- Water body volume (m^3)

Important processes for dissolved nutrients simulated by the model include nutrient (phosphate and nitrate) import, export, consumption by phytoplankton, remineralisation and exchange between the epilimnion and hypolimnion. Co-precipitation can also be simulated by SALMO. In addition, rates for nitrification and denitrification are included.

Phytoplankton processes of growth, sedimentation, zooplankton grazing, import and export, and exchange between stratified layers is included. The rate variables considered for zooplankton are growth, mortality, import, export and migration. The oxygen budget is simulated by SALMO and important processes include import and export of oxygen, exchange of oxygen between the epilimnion and hypolimnion, production by phytoplankton and consumption by zooplankton (Figure 3.1).

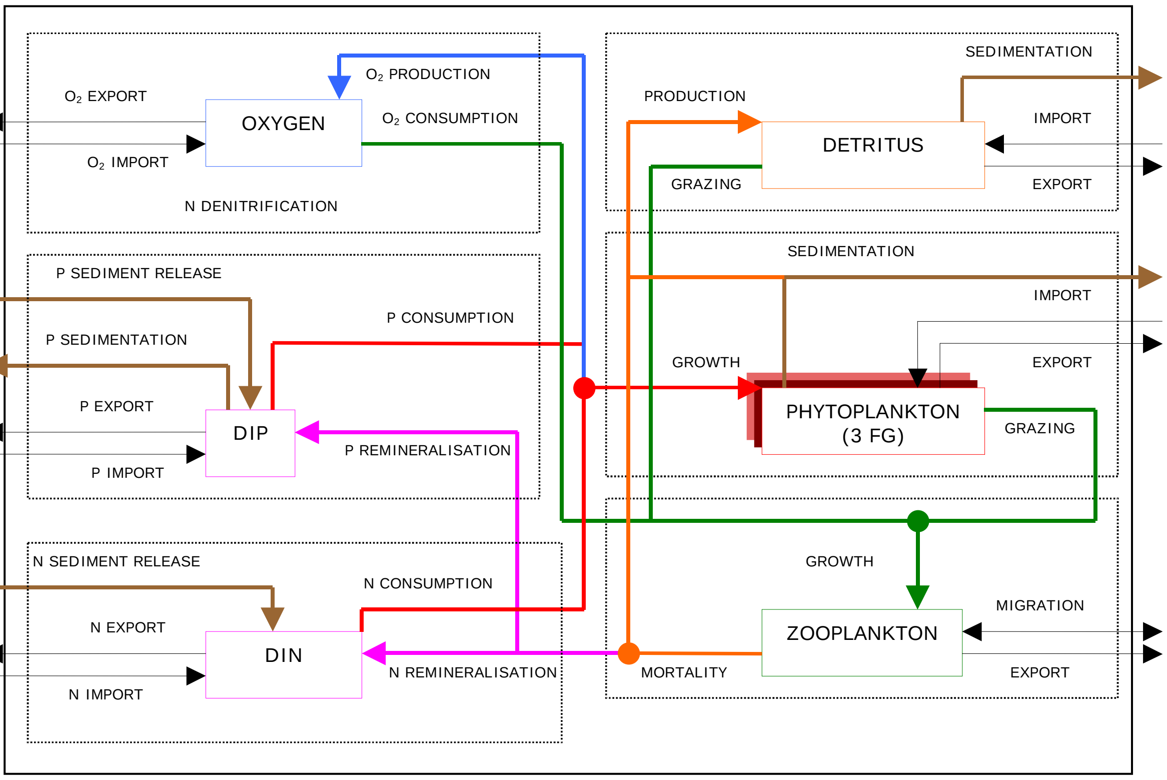


Figure 3.1. Model structure diagram of SALMO-OO illustrating all the state variables and key processes simulated by the model.

There are a number of problems with the original SALMO model that are currently being addressed. Firstly, the SALMO model is structurally complex due to the number of state variables, processes functions and parameters, which is typical of large ODE-based ecosystem models. This complexity often affects model transparency, implementation and maintenance. Secondly, SALMO was originally programmed using the FORTRAN programming language. This has resulted in a rigid and non user-friendly structure that presents problems in further developments to the model. Zhang (2006) has addressed these two problems by applying object-oriented programming and design using the JAVA programming language. As a result SALMO has been rebuilt into a portable, user-friendly and easily adapted software package, called SALMO-OO, which can be used and shared through the Internet. Adopting the object-oriented approach to improve the SALMO model has provided advanced functionality for further programming and has allowed the implementation of a framework that allows greater flexibility, particularly for maintenance and extension of the model structure. For greater detail on the object-oriented design and re-development of SALMO see Zhang (2006).

A third issue of improvement to the SALMO model is the simulation of phytoplankton functional groups. The original SALMO model could only simulate two algal functional groups for any given simulation run, although the model does contain ODEs and parameter values for three functional groups. During the redevelopment of SALMO into an object-oriented version, the model was restructured to simulate a greater number of algal functional groups. At present, three algal groups can be simultaneously predicted by the new SALMO-OO model: diatoms, green and blue-green algae. A fourth group can be included also, with extensions for other groups of algae easily made (Zhang, 2006). This has allowed the model to better describe algal seasonality and succession, and gives a more accurate prediction of the trophic state of a given lake or reservoir. Thus, with the new SALMO-OO model, experiments relating to the creation and testing of the simulation library are now easier to achieve within the object-oriented framework.

3.1.2 Phytoplankton growth process as simulated by SALMO-OO

The original SALMO model simulates two phytoplankton functional groups for a given simulation run, but includes parameter values for three functional groups: diatoms, green algae and blue-green algae. Each functional group is described by an ordinary differential equation of the same structure (equation 1.1), defined by different parameter values (Table 3.1). The sub-script *i* refers to each functional group for phytoplankton (*X*). The sub-scripts T, E, and H refer to the total mixed layer (T), epilimnion (E) and hypolimnion (H). There are three differential equations for phytoplankton dynamics for each seasonal state (i.e. mixed conditions, epilimnion and hypolimnion). A key difference between mixed and epilimnion conditions is the $AFLUXi_E$ term, which calculates exchange of phytoplankton biomass between epilimnion and hypolimnion layers (equations 1.2 and 1.3). Phytoplankton production in the hypolimnion is distinguished by the use of input data from the hypolimnion, rather then from the mixed or epilimnion layers. Growth is calculated rather than kept constant or given a value of zero, as is the case for some other primary production models (equation 1.3). Phytoplankton settling is calculated as the settling velocity divided by the mixing depth multiplied by the phytoplankton biomass. Values for the settling velocity are unique to each phytoplankton functional group (for BG = 0.005; G = 0.1; D = 0.1). The settling velocity can be a sensitive parameter in the calibration of the phytoplankton biomass ODE, but during the experiments with alternative growth and grazing models these values remained the same for each model tested, as the aim is to keep as many phytoplankton variables constant as possible. Parameter definitions and units are given in Appendix A and a complete documentation of the SALMO equations is currently in press.

Phytoplankton biomass ordinary differential equation ***where* i *= 1,2,3 functional groups***

$$\frac{dAi_T}{dt} = AGROi_T - AEXPi_T - ASEDi_T - AGRAi_T + AINi_T \tag{1.3}$$

Where $\mathrm{AGRO}i_{\mathrm{T,E,H}}$ = Phytoplankton growth process rate in the total mixed layer; epilimnion; hypolimnion
$\mathrm{A}EXP\mathrm{i}_{\mathrm{T,E,H}}$ = Phytoplankton export rate in the total mixed layer; epilimnion; hypolimnion
$\mathrm{ASED}i_{\mathrm{T,E,H}}$ = Phytoplankton sedimentation rate in the total mixed layer; epilimnion; hypolimnion
$\mathrm{AGRA}i_{\mathrm{T,E,H}}$ = Phytoplankton grazing by zooplankton rate in the total mixed layer; epilimnion; hypolimnion
$\mathrm{AIN}i_{\mathrm{T,E,H}}$ = Phytoplankton import rate in the total mixed layer; epilimnion; hypolimnion
$\mathrm{AFLUX}i_{\mathrm{T,E,H}}$ = Exchange of phytoplankton biomass between the epilimnion and hypolimnion

Phytoplankton biomass dynamics in the epilimnion

$$\frac{dAi_E}{dt} = AGROi_E - AEXPi_E - ASEDi_E - AGRAi_E + AINi_E + AFLUXi_E \tag{1.4}$$

Phytoplankton biomass dynamics in the hypolimnion

$$\frac{dAi_H}{dt} = AGROi_H - AEXPi_H - ASEDi_H - AGRAi_H \tag{1.5}$$

Phytoplankton growth as simulated by SALMO-OO is determined by the difference between photosynthesis and respiration rates:

$$\mathrm{AGRO}i_{T,E,H} = \left(\mathrm{PHO}i_{T,E,H} - \mathrm{RA}i_{T,E,H}\right) * \mathrm{A}i_{T,E,H} \tag{1.6}$$

Where $\mathrm{PHO}i_{\mathrm{T,E,H}}$ = Phytoplankton photosynthesis rate
$\mathrm{RA}i_{\mathrm{T,E,H}}$ = Phytoplankton respiration rate

Photosynthesis is calculated as the product of modifications to growth caused by light, temperature, and nutrients (phosphorus and nitrogen):

$$\mathrm{PHO}i = \mathrm{PHOL}i * \mathrm{PHOT}i * \mathrm{PHOP}i * \mathrm{PHON}i \tag{1.7}$$

Where PHOL*i* = Light limiting growth function
PHOT*i* = Growth modified by temperature function
PHOP*i* = Growth limited by dissolved inorganic phosphorus
PHON*i* = Growth limited by dissolved inorganic nitrogen

The function used to calculate the limitation of light on growth follows Michaelis-Menten kinetics (equation 1.6) and is attenuated with depth according to Lambert-Beer's Law (equation 1.7):

$$\mathrm{PHOL}i = \frac{\mathrm{IREDZ}}{\left(KIi + \mathrm{IREDZ}\right)} \tag{1.8}$$

$$\mathrm{IREDZ} = \mathrm{I} * \exp\left(-EPS * zmix\right) \tag{1.9}$$

Where IREDZ = Incident light attenuation with depth (zmix)
KI*i* = Half-saturation constant for absorption of light (J cm^{-2} d^{-1})
I = Incident light (J cm^{-2} d^{-1})
EPS = Total extinction coefficient

The dependence of growth on temperature is included as an optimum function, where growth increases linearly until an optimum temp is reached and then plateaus to a more constant rate:

$$\mathrm{PHOT}_i = \frac{\left(PHOMAX_i - PHOMIN_i\right)}{TOPTA_i * \mathrm{T} + PHOMIN_i} \qquad (1.10)$$

Where PHOMAXi = Maximum photosynthesis rate (d^{-1})
PHOMINi = Minimum photosynthesis rate (d^{-1})
TOPTAi = Optimum temperature for phytoplankton growth (°C)
T = Water temperature (°C)

Nutrient limitation is calculated based on Michaelis-Menten kinetics, with a threshold function that determines the limiting nutrient based on the N: P ratio (equation 1.9, 1.10 and 1.11):

$$\mathrm{PHONP}i = \begin{cases} \mathrm{PHOP}i \ ; \ \mathrm{N/P} \ \geq 0.0072 \\ \mathrm{PHON}i \ ; \ \mathrm{N/P} \ \leq 0.0072 \end{cases} \qquad (1.11)$$

$$\mathrm{PHOP}i \ = \ \mathrm{P/A}i/\left(KPi/\mathrm{KAP} + \mathrm{P/KAP} + KPi/\mathrm{A}i + \mathrm{P/A}i\right) \qquad (1.12)$$

$$\mathrm{PHON}i \ = \ \mathrm{N/A}i/\left(KNi/\mathrm{KAN} + \mathrm{N/KAN} + KNi/\mathrm{A}i + \mathrm{N/A}i\right) \qquad (1.13)$$

Where PHONPi = Function that determines which nutrient is limiting phytoplankton growth
KPi = Half-saturation constant for P uptake by algae (mg m^{-3})
KAP = Half-saturation constant of the inverse relationship between rate of photosynthesis and phytoplankton biomass at P limitation of phytoplankton growth (g m^{-3})
KNi = Half-saturation constant for N uptake by algae (mg m^{-3})
KAN = Half-saturation constant of the inverse relationship between rate of photosynthesis and phytoplankton biomass at N limitation of phytoplankton growth (g m^{-3})

Respiration is also temperature dependent:

$$\mathrm{RA}i = \left(RATOPTi - RATMINi\right) \ / \ TOPTAi * \mathrm{T} + RATMINi + 0.3 * \mathrm{PHO}i \qquad (1.14)$$

Where RATOPTi = Phytoplankton respiration rate at optimum temperature (d^{-1})
RATMINi = Phytoplankton respiration rate at 0°C (d^{-1})

3.1.3 Phytoplankton grazing by zooplankton process as simulated by SALMO-OO

The grazing process affecting phytoplankton biomass is represented similarly for each phytoplankton functional group, but parameterised with values relevant to each functional group. Table 3.1 gives the parameter values for the SALMO-OO grazing process equations and the definitions and units can be found in Appendix A.

The overall grazing process is calculated based on the sum of all food types that are grazed depending on zooplankton food preferences multiplied by a temperature dependent grazing function (*GT*) and zooplankton biomass (*Z*).

$$AGRAi_{T,E,H} = GT * \left(\begin{array}{l} \sum_{i=1}^{4} (\mathrm{A}_i \text{*} \mathrm{PFA}_i) / Z / \\ \left(\begin{array}{l} KAG / KZ + \sum_{i=1}^{4} (\mathrm{A}_i \text{*} \mathrm{PFA}_i) \\ / KZ + KAG / KZ + \sum_{i=1}^{4} (\mathrm{A}_i \text{*} \mathrm{PFA}_i) / Z \end{array} \right) \end{array} \right) * Z \qquad (1.15)$$

Where GT = Temperature term for ingestion rate (°C)
PFAi = Preference factor for the ingestion of each food source
KAG = Half-saturation constant for the ingestion of food by zooplankton (g m^{-3})
KZ = Half-saturation constant of the inverse relationship between ingestion rate of each food source and zooplankton biomass (g m^{-3})

Selectivity of available food by zooplankton is represented by a preference factor (*PFA$_i$*) that describes the different preferences that zooplankton have for different algal functional groups (*i* = 1,2,3) and detritus (*i* = 4). *KZ* represents the half-saturation value of the inverse relationship between the zooplankton ingestion rate and zooplankton biomass:

$$KZ_i = \begin{cases} 5.76 \text{*} \mathrm{A}_i^{0.41} & ; \ \mathrm{A}_i \leq 8.6 \\ KZMIN \ + \ 0.4 \text{*} \mathrm{A}_i^{1.5} & ; \ \mathrm{A}_i > 8.6 \end{cases} \qquad (1.16)$$

Where KZMIN = Theoretical minimum value of KZ (g m^{-3})

Grazing is also temperature dependent between minimum and maximum critical temperature ranges:

$$GT = (GMAX - GMIN) * \ \exp\left(-R * abs\left(\ln\left(\frac{T}{TOPTZ} \right) \right) \right) + GMIN \qquad (1.17)$$

Where GMAX = Maximum ingestion rate by zooplankton (g d^{-1})
GMIN = Minimum value of specific ingestion rate (g d^{-1})
R = Dependence of ingestion rate on water temperature
TOPTZ = Optimal temperature for zooplankton feeding activity (°C)

3.2. Data Requirements and Study Sites

3.2.1 Data requirements for the SALMO-OO simulation library

The SALMO-OO model makes use of measured time series data of physical and chemical factors relevant to lake dynamics, rather than using simulated environmental data as occurs in many other ecosystem models. The input data are in the form of daily values from 0 to 360 days. All data sets used in this project were interpolated into daily values using a linear interpolation method in MS Excel. The input data required by SALMO-OO are:

- Volume of the mixed layer, epilimnion and hypolimnion (m^3)
- Real mixing depth (m)
- Mean depth of mixed layer and of the hypolimnion (m)
- Water inflow (m^3 d^{-1})
- Water outflow (m^3 d^{-1})
- Strong rain events factor (highest ratio between water flow of two consecutive days) (-)
- Incident solar radiation (photosynthetic active radiation) (J cm^{-2} d^{-1})
- Water temperature in the mixed layer and the hypolimnion (°C)
- Dissolved inorganic phosphorus concentration in the inflowing water (mg P m^{-3})
- Dissolved inorganic nitrogen concentration in the inflowing water (g N m^{-3})
- Particulate organic matter concentration in the inflowing water (g m^{-3})

The SALMO-OO model has 128 parameters that are kept constant for all simulations. Parameter values from SALMO-OO, such as the half-saturation constant for PO_4-P uptake (KP) or optimum temperature for growth (TOPTA), were used where possible for the alternative process models in the simulation library (Table 3.1). However, in some cases it was necessary to include several new parameters to be used specifically for the simulation library (Table 3.2), and a literature search for parameter values was undertaken in order to establish parameter ranges. The optimum respiration rate (RO) and maximum grazing rate (Gmax) for the simulation library were given different names from those used by SALMO-OO (*RO* = RATOPT and *Gmax* = GMAX), as these two parameters are kept constant in SALMO-OO but are calibrated within the simulation library. All new parameters relevant to the simulation library were kept constant for all simulations (Appendix B).

The SALMO-OO output simulations for each data set were compared to daily measured data for phytoplankton biomass, phosphate concentration and where possible zooplankton biomass. No data were available for phytoplankton functional group dynamics, the exception being Lake Weida, however, we can determine the extent with which SALMO-OO can simulate algae functional groups based on phytoplankton succession theory for different trophic conditions as proposed by Reynolds (1993) and outlined in the SALMO-OO selection criteria discussed in section 3.6.3.

The data sets used to test and validate the SALMO-OO simulation library are categorised below according to trophic state, as a key study objective is to develop the SALMO-OO simulation library as a model that can simulate a wide variety of trophic conditions using a generic structure. During the initial development stages the SALMO-OO model was tested predominately with data from Bautzen and Saidenbach Reservoirs, Germany. These data sets were used to develop and validated the original SALMO model and have good quality measured data for phytoplankton and zooplankton biomass and phosphate concentrations with which to test the new SALMO-OO version. The use of these data sets also allows us to compare the new SALMO-OO model to the simulations of the original SALMO model and determine how effective improvements to the new model are.

Table 3.1. List of parameter values, definitions and units specific to the SALMO-OO phytoplankton growth and grazing process models.

Parameter	Definition and Units	Value		
BG = blue-green algae, G = green algae, D = diatoms.		*BG*	*G*	*D*
KI*i*	Half-saturation constant for absorption of light (J cm^{-2} d^{-1})	28	29	29
KN*i*	Half-saturation constant for N uptake by algae (g m^{-3})	0.0123	0.0095	0.0123
KP*i*	Half-saturation constant for P uptake by algae (mg m^{-3})	1.7	9.5	1.7
PHOMAX*i*	Maximum photosynthesis rate (d^{-1})	2.37	3.3	2.37
PHOMIN*i*	Photosynthesis rate at 0°C (d^{-1})	0.0	0.35	0.17
RATOPT*i*	Phytoplankton respiration rate at optimum temperature (d^{-1})	0.057	0.06	0.06
TOPTA*i*	Optimum temperature for phytoplankton growth (°C)	25	25	20
PAF*i*	Preference factor for the ingestion of each food source	0.3	1.0	1.0
GMAX	Max value of specific ingestion rate G (g d^{-1})	1.3		
GMIN	Min value of specific ingestion rate G (g d^{-1})	0.26		
KZMIN	Theoretical minimum value of KZ (g m^{-3})	4		
R	Dependence of ingestion rate on water temperature	2.0		
TOPTZ	Optimal temperature for zooplankton feeding activity (°C)	20		

Table 3.2. List of parameter definitions, units and values used in the SALMO-OO simulation library, in addition to the overall SALMO-OO model.

Parameter	Definition and Units	Reference	Value		
BG = blue-green algae, G = green algae, D = diatoms.			*BG*	*G*	*D*
IS	Saturated light intensity (J cm^{2} d^{-1})	Miyanaga (1986)	1800	1800	1800
RO	Constant phytoplankton respiration rate (d^{-1})		Calibrated		
TMAXA	Maximum temperature for growth (°C)	Scavia & Park (1976)	35	35	45
TMINA	Minimum temperature for growth (°C)	Scavia & Park (1976)	0	0	0
Kw	Extinction coefficient of water (m^{-1})	Krivtsov et al (1998); Jorgensen (1988)	0.2	0.2	0.2
Kc	Extinction coefficient due to phytoplankton biomass (m^{-2})	Krivtsov et al (1998); Jorgensen (1988)	0.1	0.1	0.1
Gmax	Maximum grazing rate by zooplankton (d^{-1})		Calibrated		
FMIN	Min concentration of phytoplankton available as food (g m^{-3})	Hongping & Jianyi (2002)	0.05	0.05	0.05
TMAXZ	Minimum temperature for zooplankton grazing (°C)	Scavia & Park (1976)	35	35	35
TMINZ	Maximum temperature for zooplankton grazing (°C)	Scavia & Park (1976)	0	0	0
Q10	Temperature coefficient	Scavia & Park (1976)	1.9	1.9	1.9

3.2.2 Study Sites

3.2.2.1 Bautzen Reservoir, Germany

Bautzen Reservoir is situated approximately 70 km north east of Dresden, Germany. The reservoir is relatively shallow (Table 3.3) and is classified as hypertrophic. The catchment area surrounding Bautzen Reservoir is densely populated with intense cultivation and industry (Deppe & Benndorf, 2002). Due to its high trophic state biomanipulation management strategies have been in place since the late 1970s to improve water quality (Dorner *et al.*, 2001). Phosphorus levels are very high (mean summer TP = 140 µg L^{-1} in 1997) and blue-green algae are the dominant phytoplankton functional group (Dorner *et al.*, 2003). Measured data for phytoplankton biomass, zooplankton biomass and phosphate concentration from 1978 (Recknagel & Benndorf, 1982) were available to test the SALMO-OO simulation library (figure 3.2).

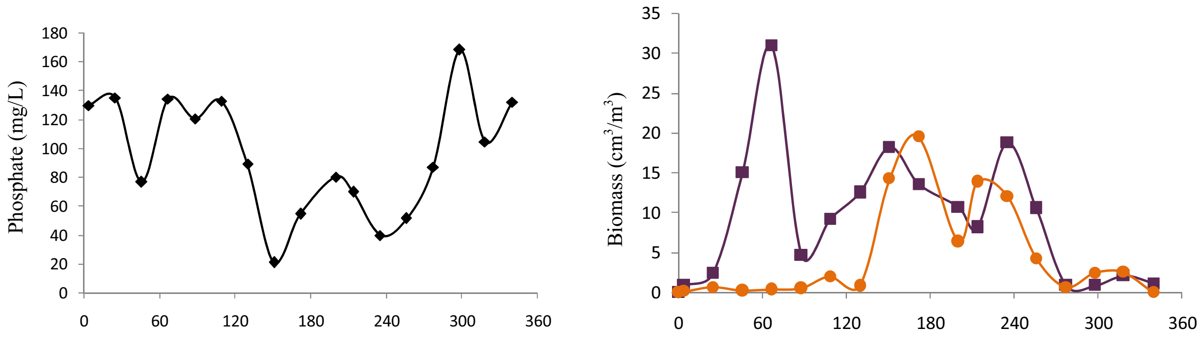


Figure 3.2. Bautzen rRservoir linearly interpolated measured data for ▬ phosphate concentration, ▬ phytoplankton and ▬ zooplankton biomasses. X-axis is in Julian days (measured data according to (Recknagel & Benndorf, 1982)).

3.2.2.2 Lake Arendsee, Germany

Lake Arendsee is located in northern Germany and is influenced predominantly by agricultural areas. The Lake is deep (Table 3.3) and is currently classified as highly eutrophic (Neumann *et al.*, 2002), with the dominance of blue-green algae. During the period that the data was collected (1975) restoration programs were introduced, such as a sewage treatment plant, which was built in 1973 and in 1976 a deepwater drainage system was constructed (Neumann *et al.*, 2002). In 1995, the mean phosphorus concentrations in the lake were estimated to be approximately 190µg/l. Phosphorus is remobilised under anoxic conditions from the lake sediments during summer and winter stratification. During the mixing period, nutrients are redispersed in the water body, becoming a source for new algal blooms (Neumann *et al.*, 2002). Measured data for phytoplankton biomass and phosphate concentration from 1975 (Klapper, *pers. com.*) were available to test the SALMO-OO simulation library (figure 3.3).

Figure 3.3. Lake Arendsee linearly interpolated measured data for - phosphate concentration and - phytoplankton biomass. X-axis is in Julian days (measured data from Klapper (1975), *pers. com.*)

3.2.2.3 Lake Roodeplaat, South Africa

Lake Roodeplaat is located 20 km north east of Pretoria, South Africa and has exhibited eutrophication problems since the 1980s. The lake is eutrophic to hypertrophic (Table 3.3), with annual TP concentration constantly above 130 μg L^{-1} and chlorophyll *a* values above 30 μg L^{-1} (DWAF/IWQG, 2000). The Lake and surrounding area (668 km^2) is largely used for recreation, thus the need for improved water quality conditions, and for agriculture. Measured data for phytoplankton biomass and phosphate concentration from 2003 (Van Ginkel, *et al.*, 2006) were available to test the SALMO-OO simulation library (figure 3.4).

Figure 3.4. Lake Roodeplaat linearly interpolated measured data for - phosphate concentration; - phytoplankton biomass; - blue-green algae; - green algae and - diatoms. X-axis is in Julian days (measured data from Van Ginkel, *et al.*, 2006).

3.2.2.4 Lake Hartbeespoort, South Africa

Lake Hartbeespoort is located 35 km west of Pretoria, South Africa and it is a warm, monomictic lake. Since the early 1980s the lake has had extensive eutrophication problems with chlorophyll *a* values generally ranging between 4 and 920 mg m^{-3} (Sommaruga & Robarts, 1997). Thus, lake Hartbeespoort is categorised as a hypertrophic lake (Table 3.3). The area is predominantly used as a source of irrigation and is a popular tourist resort area. High nutrient loads enter the system from catchment runoff (mean annual runoff = 1630m^3; mean annual rainfall = 700mm) influenced by heavy urban and industrial areas (Cawood & Friend, 2005; Cochrane *et al.*, 1987). Cyanobacteria are the dominant phytoplankton species throughout the year, comprising approximately 90% of the total algae abundances (Cochrane *et al.*, 1987). Measured data for phytoplankton biomass and phosphate concentration is available from 2003 (Van Ginkel, *et al.*, 2006) (figure 3.5).

NOTE: This figure is included on page 36 in the print copy of the thesis held in the University of Adelaide Library.

Figure 3.5. Lake Hartbeespoort linearly interpolated measured data for **-** phosphate concentration; **-** phytoplankton biomass; **-** blue-green algae; **-** green algae and **-** diatoms. X-axis is in Julian days (measured data from Van Ginkel, *et al.,* 2006).

3.2.2.5 Lake Klipvoor, South Africa

Lake Klipvoor is located north east of Lakes Roodeplaat and Hartbeesport, near the large town of Bela-Bela in the eastern provinces of South Africa. Lake Klipvoor is heavily degraded with poor water quality due to high levels of urbanisation and industrial discharges. The land surrounding the Klipvoor Dam and lake is used mostly for agriculture with some densely settled areas. Formations of algal scums are common during summer, with a dominance of blue-green algae species (*Microcystis* and *Anabeana*). Measured data for phytoplankton biomass and phosphate concentration is available from 2003, as well as biomass measured data for blue-green algae, green algae and diatoms (Van Ginkel, *et al.,* 2006) (figure 3.6).

NOTE: This figure is included on page 36 in the print copy of the thesis held in the University of Adelaide Library.

Figure 3.6. Lake Klipvoor linearly interpolated measured data for **-** phosphate concentration; **-** phytoplankton biomass; **-** blue-green algae; **-** green algae and **-** diatoms. X-axis is in Julian days (measured data from Van Ginkel, *et al.,* 2006). (Table 3.3).

Table 3.3. Key morphological data and trophic status of the data sets used to develop and validate the SALMO-OO simulation library.

	Mean depth (m)	*Max depth (m)*	*Surface area (km^2)*	*Water residence time (d)*	*Trophic state*	*References*
Bautzen Reservoir Germany	7.4	13	5.3	193	Hypertrophic	Deppe & Benndorf (2002)
Lake Arendsee Germany	29	49	5.2	183	Hypertrophic	Neumann et al (2002)
Lake Roodeplaat South Africa	10.6	40	3.97	No data available	Hypertrophic	DWAF/IWQG (2000)
Lake Hartbeespoort South Africa	9.6	32.5	20	No data available	Hypertrophic	Cochrane *et al* (1987)
Lake Klipvoor South Africa	5.78	20.9	7.58	131.4	Hypertrophic	Van Ginkel *pers. com.*;
Saidenbach Reservoir Germany	15.3	45	1.46	No data available	Mesotrophic	Schulze *et al* (2004); Kahl & Radke (2005)
Lake Weida Germany	10	20	0.93	106	Mesotrophic	Schmidt *et al* (2002)
Lake Stechlin Germany	23.3	69.5	4.23	No data available	Oligotrophic	Schulz *et al* (2004)
Lake Soyang South Korea	42	110	45	255	Mesotrophic - Oligotrophic	Kim *et al* (2000)

3.2.2.6 Saidenbach Reservoir, Germany

Saidenbach reservoir is situated in Saxony, Germany and is predominantly used as a drinking water reservoir. The reservoir has been classified as mesotrophic (Table 3.3) with soluble reactive phosphate concentrations of between 10 to 15 µg L^{-1} (Kahl & Radke, 2005). Green algae and large diatoms dominate the phytoplankton community. Measured data for phytoplankton biomass, zooplankton biomass and phosphate concentration from 1975 were available (from Benndorf and Recknagel, 1982) to test the SALMO-OO simulation library (Figure 3.7).

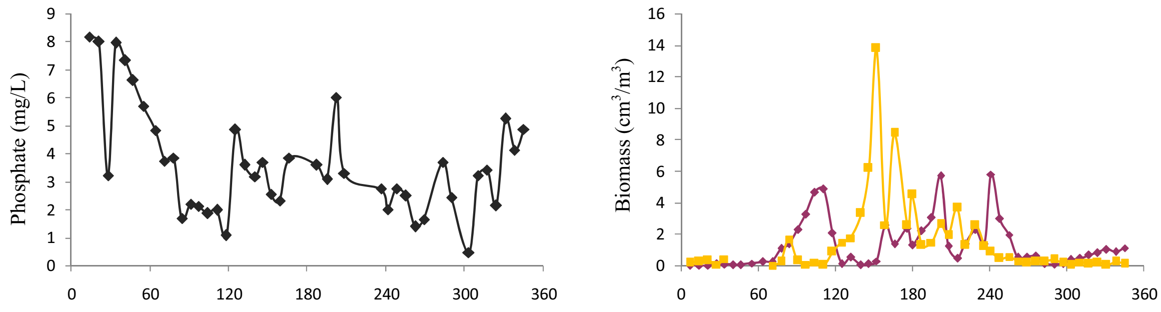


Figure 3.7. Saidenbach reservoir linearly interpolated measured data for ▬ phosphate concentration, ▬ phytoplankton and ▬ zooplankton biomasses. X-axis is in Julian days (measured data from Benndorf and Recknagel, 1982).

3.2.2.7 Lake Weida, Germany

Lake Weida is part of the drinking water reservoir system of Weida/ Zeulenroda/ Lossau in the eastern Thuringaria Mountains, Germany. The surrounding land is used predominantly for agriculture and forestry, with dense settlements affecting the water quality of the lake. Lake Weida is classed as mesotrophic (Table 3.3), but can be influenced by the occurrence of cyanobacteria in summer (Schmidt *et al.*, 2002). Measured data for phytoplankton biomass, zooplankton biomass, green algae biomass, blue-green algae biomass, diatom biomass and phosphate concentration is available from 1984 (data from Saenger, 1985) (figure 3.8).

NOTE: This figure is included on page 38 in the print copy of the thesis held in the University of Adelaide Library.

Figure 3.8. Lake Weida linearly interpolated measured data for ▬ phosphate concentration; ▬ phytoplankton biomass; ● blue-green algae; ▬ green algae and ▬ diatoms. X-axis is in Julian days (measured data from Saenger, 1985).

3.2.2.8 Lake Stechlin, Germany

Lake Stechlin is located approximately 100 km north of Berlin, Germany and is the second deepest lake in northern Germany (Table 3.3). It is characterised as oligotrophic with a phosphorus load of 0.04 to 0.07 g TP m^{-2} yr^{-1} and primary production of about 121 g C m^{-2} yr^{-1} (Schulz *et al.*, 2004). 80% of the catchment area is covered with forests, and calcite precipitation, among other factors, has been shown to cause this oligotrophic status (Holzbecher & Nutzmann, 2000). Diatoms and green algal functional groups dominate the phytoplankton community. Measured data for phytoplankton biomass, zooplankton biomass and phosphate concentration from 1975 (Benndorf and Recknagel, 1982) are available to test the SALMO-OO simulation library (figure 3.9).

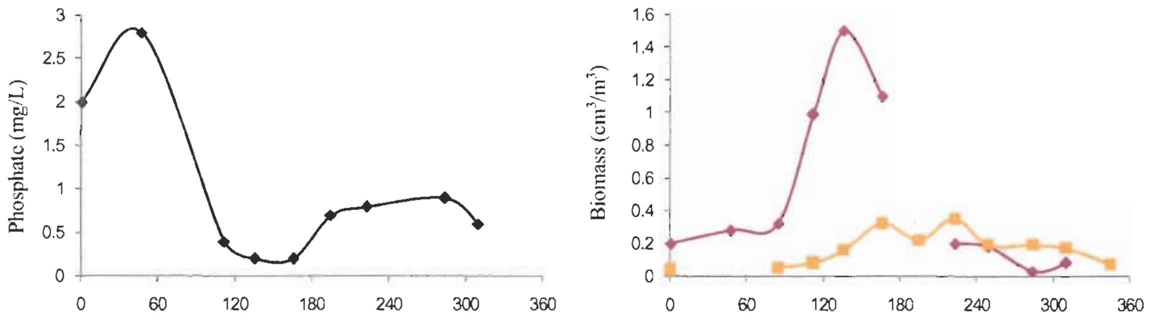


Figure 3.9. Lake Stechlin linearly interpolated measured data for ▬ phosphate concentration, ▬ phytoplankton and ▬ zooplankton biomasses. X-axis is in Julian days (measured data from Benndorf and Recknagel, 1982).

3.2.2.9 Lake Soyang, South Korea

Lake Soyang is the deepest and largest reservoir in South Korea (Kim *et al.*, 2000) and is located on the North Han River. Lake Soyang has been classified as mesotrophic to oligotrophic (Table 3.3). Most nutrient loading is from non-point sources and aquaculture. The catchment experiences summer monsoons with an annual rainfall of 1200 mm per year (Kim *et al.*, 2000). Measured data for phytoplankton biomass and phosphate concentration is available from 1998 (Kim *et al.*, 2000) (Figure 3.10). The data was obtained as daily time series.

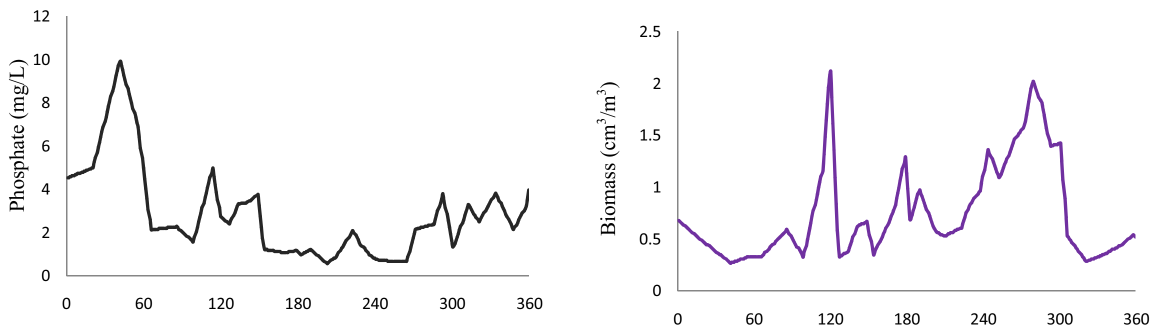


Figure 3.10. Lake Soyang linearly interpolated measured data for ▬ phosphate concentration and ▬ phytoplankton biomass. X-axis is in Julian days (measured data from Kim *et al.*, 2000).

3.2.3 A Comparison of Trophic States

There are several different trophic state indices that are available for classification of lakes and reservoirs. Many have been based on European lakes, but are equally applicable to other lentic water bodies in other parts of the world. Tables 3.4, 3.5, 3.6 illustrate the different parameters used to classify trophic conditions by different authors. Table 3.7 gives a trophic State Index (TSI) developed by Carlson (1977) which involves a simple regression function to calculate trophic state based on total phosphorus (TP), chlorophyll-*a* and secchi disk measurements:

$$\text{TSI TP} = 14.42*\left[\ln\left(\text{Mean } TP\right)\right]+4.15 \tag{1.18}$$

$$\text{TSI Chl-}a = 9.81*\left[\ln\left(\text{Mean Chl-}a\right)\right]+30.6 \tag{1.19}$$

$$\text{TSI Secchi} = 60 - \left(14.41*\left[\ln\left(\text{Mean } Secchi\right)\right]\right) \tag{1.20}$$

Each of the different approaches demonstrates that different parameters can be used to assess trophic conditions. With the datasets that are available for this study only orthophosphate and chlorophyll-*a* data can be used to classify the trophic state of each site, but these parameters are probably the most important to give a fair estimate of trophic conditions.

Table 3.8 lists the mean orthophosphate, mean chlorophyll-*a* and maximum chlorophyll-*a* and the Carlson's chlorophyll-*a* TSI for each of the study sites. Comparisons of the different indices for each site show that generally the trophic state is easily assessed based on these few variables, with minimal discrepancies as to what trophic category each site should be allocated. Bautzen reservoir and Lake Arendsee are both categorised as eutrophic by all indices. Similarly Saidenbach reservoir and Lake Weida are unanimously categorised as mesotrophic, Lakes Stechlin and Soyang as oligotrophic, and Lake Klipvoor as hypertrophic. For Lakes Roodeplaat and Hartbeeespoort some discrepancies were observed between classifying these sites as either eutrophic or hypertrophic. Trophic state indices from Ryding and Rast (1989) and Carlson (1977) both classify Lake Roodeplaat as eutrophic whereas Vollenweider and Kerekes (1982) and Forsberg and Ryding (1980) classify the site as hypertrophic. Only Ryding and Rast (1989) categorise Lake Hartbeespoort as eutrophic/polytrophic, whereas the other indices categorise this site as hypertrophic. However, according to the literature on these two South African sites, both lakes are considered as hypertrophic, with severe eutrophication problems.

Table 3.8. Trophic state classification of the study sites used to test the SALMO-OO simulation library according to four different indices. E = eutrophic; H = hypertrophic; P = polytrophic; M = mesotrophic; O = oligotrophic.

	Bautzen 1978	Arendsee 1979	Roodeplaat 2003	Hartbeespoort 2003	Saidenbach 1975	Weida 1984	Stechlin 1975	Soyang 1998	Klipvoor 2003
Mean Ortho-phosphate (mg/L)	0.1	0.2	0.16	0.05	0.003	0.015	0.001	0.003	0.76
Mean Chl-*a* (μg/L)	25.6	16.8	36.2	53.2	3.18	4.76	1.09	1.82	144.6
Max Chl-*a* (μg/L)	75	39.7	96.3	277.7	14.57	17.5	3.8	5.3	515.8
TSI (Chl-*a*)	62.4	58.3	65.8	69.6	41.9	45.9	31.4	36.5	79.4
Trophic State[1]	E	E - P	E - P	E - P	M	M	O	O	H
Trophic State[2]	E	E	H	H	M	M	O	O	H
Trophic State[3]	E	E	E - H	H	M	M	O	O	H
Trophic State[4]	E	E	E	H	M	M	O	O	H

[1] (Ryding and Rast, 1989)
[2] (Vollenweider and Kerekes, 1982)
[3] Forsberg and Ryding (1980)
[4] Carlson (1977)

3.3. Selection criteria for literature model evaluation

The development of the SALMO-OO simulation library required an extensive search of the scientific literature for models describing phytoplankton dynamics in aquatic environments. Figure 3.11 illustrates the procedure designed to select appropriate phytoplankton population models. Several criteria had to be met in order to finalise which models would be analysed for inclusion into the SALMO-OO simulation library. The initial focus was on phytoplankton models that were of the form of ordinary differential equations (ODEs), as these are compatible with the model structure from SALMO. It was important that those models considered showed that their particular model structure could simulate phytoplankton seasonal dynamics reasonably well. Papers that provided sufficient documentation to reproduce phytoplankton dynamics in simulation software were selected for initial experimentation.

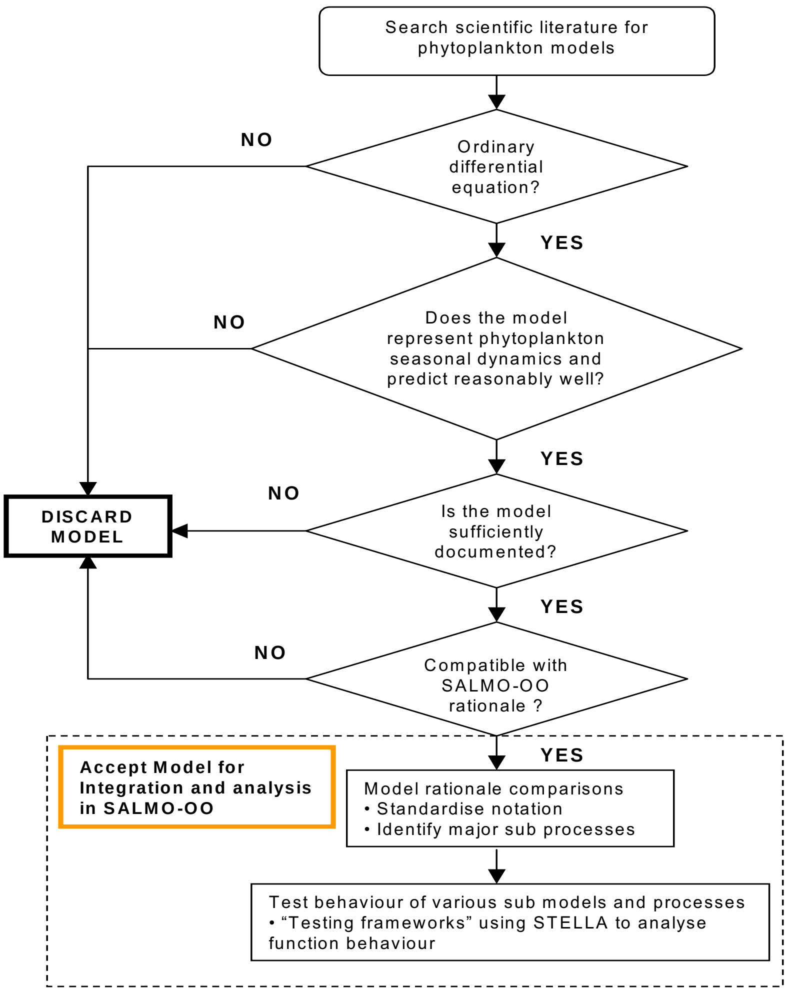

Search scientific literature for phytoplankton models
NO
Ordinary differential equation?
YES
NO
Does the model represent phytoplankton seasonal dynamics and predict reasonably well?
YES
NO
DISCARD MODEL
Is the model sufficiently documented?
YES
NO
Compatible with SALMO-OO rationale ?
YES
Accept Model for Integration and analysis in SALMO-OO
Model rationale comparisons
• Standardise notation
• Identify major sub processes
Test behaviour of various sub models and processes
• "Testing frameworks" using STELLA to analyse function behaviour

Figure 3.11. Flow chart illustrating criteria for the selection of literature models to be included in the SALMO-OO simulation library.

This eliminated many specific models from being considered, however the overlap between model structures and formulisations ensured that many approaches were well represented. Compatibility with the SALMO model rationale was also a very important consideration, such as:

1. The ability of a model to simulate phytoplankton functional groups,
2. Phytoplankton units as biomass rather then chlorophyll *a* or carbon units,
3. Phytoplankton production limited by light intensity, nutrient concentrations and water temperature,
4. Biomass impacted by grazing from zooplankton, and
5. Equations for various processes were developed based upon algae population dynamics theory, rather then through computational methods such as regression analysis.

Once the literature models were subjected to these criteria and accepted, further analysis of the model structure could proceed.

3.4. Initial "Testing Frameworks" for model analysis

In order to review and test many alternative models, a simple "testing framework" was established using the visual simulation software STELLA v 6.0.1. STELLA enables the development, testing and running of models based on differential equations in a manner that is quick and easy. The testing framework included the development of a series of ordinary differential equations representing phytoplankton biomass, with growth (photosynthesis minus respiration) and grazing considered as processes influencing phytoplankton production (equation 1.19). Each alternative growth model was tested for three phytoplankton functional groups: blue-green algae, green algae and diatoms, and differentiated by using different parameter values for each functional group. Standardisation of model notation and conversion of various parameters was conducted where necessary.

$$\frac{dt\ \text{Algae}}{dt} = \text{Growth - Grazing} \tag{1.21}$$

The time step for model simulations was daily and the simulations were run for 360 days using linearly interpolated time-series data as inputs for water temperature and light intensity.

3.4.1 Phytoplankton growth process models

Each literature model that satisfied the criteria outlined in section 3.3 was analysed within this "testing framework". Thus, several ODEs were programmed in STELLA to represent each different growth process for four different literature models (equation 1.20).

The testing framework included several ordinary differential equations based on the different representations of growth processes models and the same formulation for grazing by zooplankton.

$$\frac{dt \text{ AlgaeBiomass}}{dt} = (\text{Photosynthesis - Respiration}) * \text{AlgaeBiomass - Grazing} \quad (1.22)$$

A simple equation for grazing was included using measured data for zooplankton biomass and a constant value for the maximum grazing rate (Gmax):

$$\text{Grazing} = \text{Gmax} * \text{AlgaeBiomass} * \text{ZooplBiomass} \quad (1.23)$$

Therefore, the only processes that would be different between each model tested would be phytoplankton growth, enabling the behaviour of each growth process to be compared to each other, the SALMO growth process and the measured data for phytoplankton biomass.

Whilst reviewing many different phytoplankton models, it became apparent that the representation of nutrient limitations was mostly considered by the classic Michaelis-Menten function:

$$f(\text{P}) = \frac{\text{PO4}}{KP + \text{PO4}} \quad (1.24)$$

where PO_4 is the concentration of dissolved inorganic phosphorus and *KP* is the half-saturation constant for the uptake of PO_4 by algae. Therefore, the Michaelis-Menten function was adopted for all models tested to represent nutrient limitations. For the initial simulation library experiments only dissolved inorganic phosphorus was considered and measured data was used, rather than simulated phosphorus . Figure 3.12 illustrated the testing framework for the simulation library.

The data used to test each alternative growth process function was from Bautzen and Saidenbach Reservoir, Germany (see section 3.2 for description of data sets).

3.4.2 Phytoplankton grazing process models

A second "testing framework" was developed to examine phytoplankton predation by zooplankton process models. A somewhat more complex version of the growth models testing framework was developed, where zooplankton and dissolved inorganic phosphorus were considered as state variables and simulated by the framework. Three functional groups of algae were also included as in the growth model experiments. Each grazing model was tested within this framework, with the output being the phytoplankton biomass, zooplankton biomass (cm^3/m^3) and phosphate concentration (mg/m^3). This was compared to the phytoplankton and zooplankton biomasses produced by the SALMO grazing function and phytoplankton and zooplankton measured data (Figure 3.13). Each grazing and zooplankton growth model was tested with the same databases used for the phytoplankton growth experiments. Driving variables included water temperature, solar radiation, water depth, phosphate loadings, and water inflow and outflow.

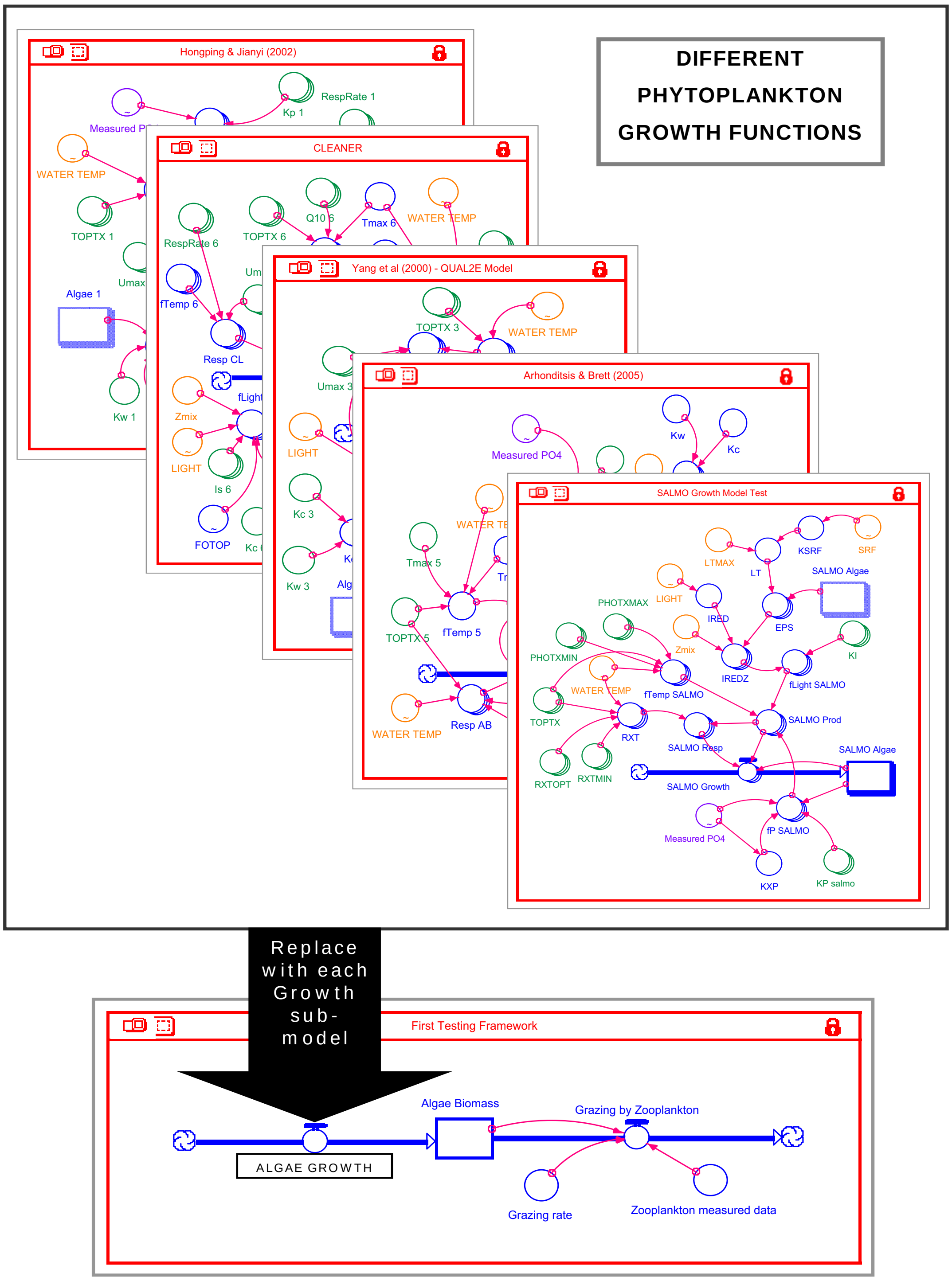
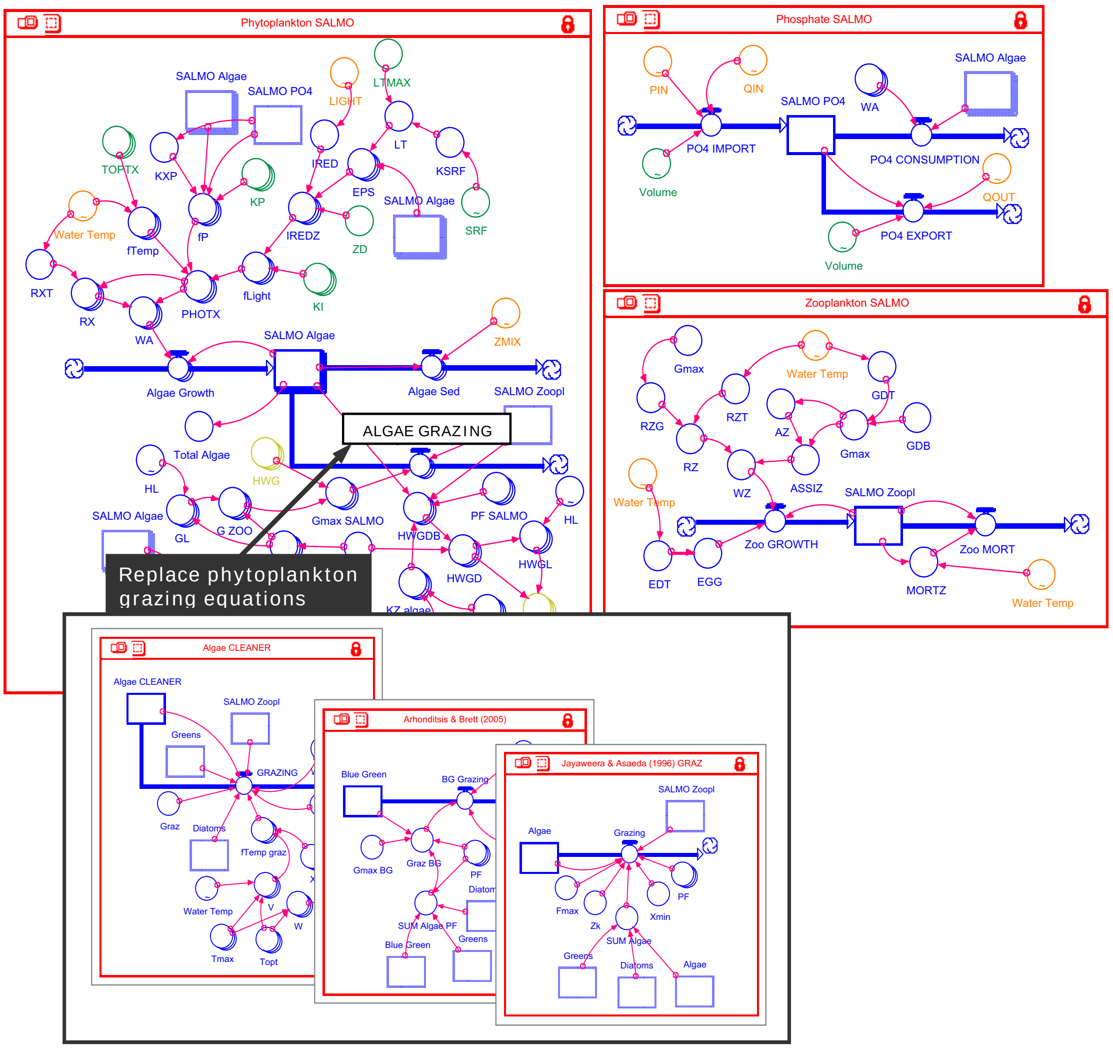


Figure 3.12. Testing framework structure using STELLA software for alternative phytoplankton growth models.



Phytoplankton SALMO
Phosphate SALMO
Zooplankton SALMO
ALGAE GRAZING
Replace phytoplankton grazing equations
Algae CLEANER
Arhonditsis & Brett (2005)
Jayaweera & Asaeda (1996) GRAZ

Figure 3.13. Testing framework structure using STELLA software for alternative phytoplankton grazing models.

3.5. Description of alternative process models for phytoplankton growth and grazing dynamics

As a result of the "testing framework" experiment three alternative models for phytoplankton growth and grazing were selected for inclusion into the SALMO-OO simulation library:

1. Hongping and Jianyi (2002)
2. Arhonditsis & Brett (2005)
3. CLEANER (Park *et al*, 1974; Scavia and Park, 1976)

Hongping and Jianyi (2002) developed a lake ecosystem model for West Lake, China to investigate eutrophication and perform scenario analysis of various management strategies, such as biomanipulation. The model was designed specifically for West Lake, which is a small, sallow lake with eutrophic conditions. The model simulates four phytoplankton functional groups (blue-green algae, green algae, diatoms and cryptophyta) using the same ordinary different equations, but defined for each group by specific parameter values. Only phosphate conditions have been modelled, as this is the limiting nutrient. The model does not simulate stratification, as West Lake is shallow and assumed to be well mixed.

Arhonditsis and Brett (2005) developed a more complex eutrophication model for the simulation of Lake Washington, USA. The model is similar to SALMO-OO but includes state variables for organic carbon and silica and a hydrodynamic model. The model simulates three groups of phytoplankton (blue-green algae, green algae and diatoms) and two groups of zooplankton (copepods and cladocerans). Many of the environmental forcing functions, such as water temperature and solar radiation are simulated rather than measured time-series data.

The CLEANER model was first published in 1976 by Scavia and Park and was based on an early model version called CELAN (Park *et al.*, 1974), with additions of nitrogen and phosphorus cycling, oxygen reformulation of the decomposition sub models and refinement of various process terms to make CLEANER. The CLEAN and CLEANER models was designed for lake ecosystem modelling particularly for the Great Lakes of North America, and was influenced by the work of Riley (Riley, 1965; Riley *et al.*, 1949) and Steele (Steele, 1958, 1962). CLEANER includes sub models for phytoplankton, zooplankton (herbivorous and carnivorous), nutrients (PO_4-P and NO_3-N), decomposers, macrophytes, suspended organic matter, fish, benthic insects and sediments. CLEANER simulates two groups of algae, based on size – nano-phytoplankton (extremely small) and net-phytoplankton (larger species). Appendix A gives the parameter definitions and units of each of the alternative growth and grazing models. Parameter values can be found in Table 3.1 and 3.2.

3.5.1 Phytoplankton growth process models

Phytoplankton growth is calculated for each alternative growth model as the difference between photosynthesis and respiration per unit of biomass (equation 1.23). In all cases, photosynthesis is limited by nutrients, light and temperature using the multiplicative function (equation 1.24), although Arhonditsis and Brett (2005) include the Liebig's Law of the Minimum function to decide which nutrient is the limiting factor (equation 1.25).

$$\mathrm{AGRO}i_{T,E,H} = \left(\mathrm{PHO}i_{T,E,H} - \mathrm{RA}i_{T,E,H}\right) \tag{1.25}$$

Where $\mathrm{AGRO}i_{\mathrm{T,E,H}}$ = Phytoplankton growth process rate in the total mixed layer; epilimnion; hypolimnion
$\mathrm{PHO}i_{\mathrm{T,E,H}}$ = Phytoplankton photosynthesis rate
$\mathrm{RA}i_{\mathrm{T,E,H}}$ = Phytoplankton respiration rate

Hongping and Jianyi (2002) photosynthesis function:

$$\mathrm{PHO}i_{T,E,H} = PHOMAXi{*}\mathrm{PHOL}i{*}\mathrm{PHOT}i{*}\mathrm{POHP}i{*}\mathrm{PHON}i \tag{1.26}$$

Where PHOMAXi = Maximum photosynthesis rate (d^{-1})
PHOL*i* = Light limiting growth function
PHOT*i* = Growth modified by temperature function
PHOP*i* = Growth limited by dissolved inorganic phosphorus
PHON*i* = Growth limited by dissolved inorganic nitrogen

Arhonditsis and Brett (2005) photosynthesis function:

$$\mathrm{PHO}i_{T,E,H} = PHOMAXi{*}\mathrm{PHOL}i{*}\mathrm{PHOT}i{*}\min\left\{\mathrm{POHP}i, \mathrm{PHON}i\right\} \tag{1.27}$$

The CLEANER photosynthesis function uses a slightly different construct to calculate limiting factors. Combined limitations of light and nutrients are represented as a normalised factor that is mathematically analogous to the inverse of the total effect of electrical resistors in parallel:

$$\mathrm{PHO}i_{T,E,H} = PHOMAXi{*}\mathrm{Ut}{*}\mathrm{PHOT}i \tag{1.28}$$

$$Ut = \frac{N}{\sum_{i}^{n}\left(\frac{1}{f(U)i}\right)} \quad \text{Therefore,} \quad Ut = \frac{3}{\left(\left(\frac{1}{f\left(PHOLi\right)}\right)+\left(\frac{1}{f\left(PHOPi\right)}\right)+\left(\frac{1}{f\left(PHONi\right)}\right)\right)} \tag{1.29}$$

The developers of the original CLEAN model were not satisfied that the typical formalisation of photosynthesis used in ecosystem models was appropriate to represent what was observed in nature. Therefore, they developed a new function, where *n* is the number of limiting functions and is used to normalise the total limitation term. If no nutrient is limiting, *Ut* shows no limitation. If any one function is absolutely limiting, the function is totally limiting.

Each growth model simulates underwater light conditions using an integrated Steele and Beer's Law function, to account for light intensities above and below the saturation level and integrated over depth. However, Arhonditsis and Brett (2005) and CLEANER models include the photoperiod in these functions, which defines a 12 hour light – dark cycle:

Hongping and Jianyi (2002) light limitation function:

$$\text{PHOL}i = \frac{\left[\dfrac{\text{I} * \left[1 - \exp\left(- \text{EPS} * zmix\right)\right]}{- \text{EPS} * zmix}\right]}{IS} * \exp\left(1 - \frac{\left[\dfrac{\text{I} * \left[1 - \exp\left(- \text{EPS} * zmix\right)\right]}{- \text{EPS} * zmix}\right]}{IS}\right) \tag{1.30}$$

Where EPS = Total extinction coefficient
Zmix = mixing depth (m)
IS = Saturated light intensity (J cm^2 d^{-1})

Arhonditsis and Brett (2005) and CLEANER light limitation function:

$$\text{PHOL}i = \frac{2.718 * FP}{\text{EPS} * zmix} * \left[\exp\left(\begin{array}{c} -\dfrac{\text{I}}{IS * FP} * \\ \exp\left(-\text{EPS} * zmix\right)\end{array}\right) - \exp\left(-\frac{\text{I}}{IS * FP}\right)\right] \tag{1.31}$$

Where FP = Photoperiod = 12 – 4 * Cos(2 * π * t/360) where t = time

The light extinction coefficient parameter (*EPS*), common to each growth model, is calculated based on the sum of the background extinction coefficient of water (*kw*) and of organic matter (*kc*), such as detritus and phytoplankton cells:

$$\text{EPS} = kw + kc * \sum_{i=1}^{3} \text{A}_i \tag{1.32}$$

Temperature is calculated as an optimum function by all growth models, where growth increases until an optimum temperature is reached (TOPTA*i*) and then gradually declines until a critical temperature range is reached:

Hongping and Jianyi (2002) temperature limitation function:

$$\text{POHT}_i = \frac{\text{T}}{TOPTA_i} * \exp\left(1 - \frac{\text{T}}{TOPTA_i}\right) \tag{1.33}$$

Arhonditsis and Brett (2005) temperature limitation function, which is based on that described by Lehman *et al* (1975):

$$\mathrm{PHOT}i = \exp\left(-2.3*\left(\frac{\mathrm{T} - TOPTAi}{TMAXA_i - TOPTAi}\right)^2\right) \quad ; \text{for } \mathrm{T} \geq TOPTAi$$

$$\mathrm{PHOT}i = \exp\left(-2.3*\left(\frac{\mathrm{T} - TOPTAi}{TMAXA_i - TMINA_i}\right)^2\right) \quad ; \text{for } \mathrm{T} \leq TOPTAi \tag{1.34}$$

Where TMAXA_i = Maximum temperature for growth (°C)
TMINA_i = Minimum temperature for growth (°C)
T = Water temperature (°C)

CLEANER temperature limitation function is a slightly modified function from one developed to represent the temperature dependencies of respiration and feeding rates of poikilotherms:

$$\mathrm{PHOT}i = \mathrm{T1}^{\mathrm{T2}} \exp\left(\mathrm{T2}*\left(1-\mathrm{T1}\right)\right) \tag{1.35}$$

$$\mathrm{T1} = \frac{TMAXA_i - \mathrm{T}}{TMAXA_i - TOPTAi} \tag{1.36}$$

$$\mathrm{T2} = \frac{\left[\mathrm{T3}^2*\left(1+\sqrt{1+{}^{40}\!/_{\mathrm{T3}}}\right)\right]^2}{400} \tag{1.37}$$

$$\mathrm{T3} = \ln\left(Q_{10}\right)*\left(TMAXA_i - TOPTAi\right) \tag{1.38}$$

Growth is limited by external nutrients according to the classic Michaelis-Menten function. Both phosphate (P) and nitrate (N) limitations have been included:

$$\mathrm{PHOP}\,i = \frac{P}{KPi + P} \tag{1.39}$$

$$\mathrm{PHON}\,i = \frac{N}{KNi + N} \tag{1.40}$$

Where KPi = Half-saturation constant for P uptake by algae (mg m^{-3})
KNi = Half-saturation constant for N uptake by algae (g m^{-3})
P = Dissolved inorganic phosphorus (mg m^{-3})
N = Dissolved inorganic nitrogen (g m^{-3})

In all cases, phytoplankton respiration is dependent on water temperature and a maximum respiration rate. The Hongping and Jianyi (2002) and Arhonditsis and Brett (2005) phytoplankton respiration function describes an exponential rate.

Hongping and Jianyi (2002):

$$\mathrm{RA}i_{T,E,H} = ROi * \exp\left(0.038 * \mathrm{T}\right) \tag{1.41}$$

Where $\mathrm{RA}i_{\mathrm{T,E,H}}$ = Phytoplankton respiration process rate (d^{-1})
$\mathrm{RO}i$ = Constant phytoplankton respiration rate (d^{-1})

Arhonditsis and Brett (2005):

$$\mathrm{RA}i_{T,E,H} = ROi * \exp\left(0.07 * (\mathrm{T} - TOPTAi)\right) \tag{1.42}$$

CLEANER phytoplankton respiration rate:

$$\mathrm{RA}i_{T,E,H} = ROi * TOPTAi * PHOMAXi \tag{1.43}$$

3.5.2 Phytoplankton grazing process models

Zooplankton grazing of phytoplankton is formulated similarly for each of the alternative grazing models. Generally, zooplankton grazing is described by a modified Michaelis-Menten function. Feeding ceases when food concentrations fall below a critical threshold, thus grazing rates are greatly reduced for food levels below a certain concentration. Also, a size selective grazing coefficient is included that assigns weights to the different food types. Therefore, the preference factor coefficient (PAF_i) allows zooplankton to select different types of food over others, depending on the zooplankton preference for certain algal functional groups. Both the Arhonditsis and Brett (2005) and CLEANER grazing models include a temperature limitation term (*GT*) that regulates on phytoplankton grazing. These terms are the same as for the growth limiting temperature functions (equations 1.32 – 1.36), but the TOPTA_i parameter (Optimum temperature for phytoplankton growth) is replaced with the TOPTZ parameter (Optimal temperature for zooplankton feeding activity). Also, the TMINA_i (minimum temperature for algae growth) and TMAXA_i (maximum temperature for algae growth) values are replaced with TMINZ (minimum temperature for zooplankton grazing) and TMAXZ (maximum temperature for zooplankton grazing) values.

Hongping and Jianyi (2002) phytoplankton grazing process model:

$$\mathrm{AGRA}i_{T,E,H} = \mathrm{GMAX} * \frac{\sum_{i=1}^{4}\left(\mathrm{PFA}_i * \mathrm{A}_i\right) - FMIN}{KAG + \sum_{i=1}^{4}\left(\mathrm{PFA}_i * \mathrm{A}_i\right) - FMIN} * PFAi * \frac{\mathrm{A}_i}{\sum_{i=1}^{4}\left(\mathrm{PFA}_i * \mathrm{A}_i\right)} \tag{1.44}$$

Where GMAX = Maximum ingestion rate by zooplankton (g d^{-1})
$\mathrm{PFA}i$ = Preference factor for the ingestion of each food source
FMIN = Minimum concentration of phytoplankton available as food (g m^{-3})
KAG = Half-saturation constant for the ingestion of food by zooplankton (g m^{-3})
$\mathrm{A}i$ = Phytoplankton biomass in mixed layer, epilimnion and hypolimnion (cm^{-3} m^{-3})

Arhonditsis and Brett (2005) phytoplankton grazing process model, with temperature regulation as given in equation 1.44:

$$\mathrm{AGRA}i_{T,E,H} = \left(\frac{(\mathrm{GMAX} * \mathrm{PFG}i * \mathrm{A}i)}{KAG + \sum_{i=1}^{4} (\mathrm{PFG}i * \mathrm{A}i)} \right) * \mathrm{PHOT}i \qquad \mathrm{PFG}_i = \frac{\mathrm{PFA}_i * \mathrm{A}_i}{\sum_{i=1}^{4} (\mathrm{PFA}_i * \mathrm{A}_i)} \tag{1.45}$$

$$\begin{aligned} GT &= \exp\left(-2.3 * \left(\frac{\mathrm{T} - TOPTZi}{TMAXZ_i - TOPTZi} \right)^2 \right) \quad ; \text{for } \mathrm{T} \geq TOPTZi \\ GT &= \exp\left(-2.3 * \left(\frac{\mathrm{T} - TOPTZi}{TMAXZ_i - TMINZ_i} \right)^2 \right) \quad ; \text{for } \mathrm{T} \leq TOPTZi \end{aligned} \tag{1.46}$$

Where PFGi = Auxiliary variable for phytoplankton preference factor for grazing;
TOPTZ = Optimal temperature for zooplankton feeding activity (°C);
TMAXZ = Maximum temperature for Zooplankton grazing (°C);
TMINZ = Minimum temperature for Zooplankton grazing (°C)

CLEANER phytoplankton grazing process model, with temperature regulation (GT) as given by equations 1.46 - 1.49:

$$\mathrm{AGRA}i_{T,E,H} = \mathrm{GMAX} * \left(\frac{(\mathrm{PFA}_i * \mathrm{A}_i)}{KAG + \sum_{i=1}^{4} (\mathrm{PFA}_i * \mathrm{A}_i)} \right) * GT \tag{1.47}$$

$$GT = \mathrm{T1}^{\mathrm{T2}} \exp(\mathrm{T2} * (1 - \mathrm{T1})) \tag{1.48}$$

$$\mathrm{T1} = \frac{TMAXZ_i - \mathrm{T}}{TMAXZ_i - TOPTZi} \tag{1.49}$$

$$\mathrm{T2} = \frac{\left[\mathrm{T3}^2 * \left(1 + \sqrt{1 + 40/\mathrm{T3}} \right) \right]^2}{400} \tag{1.50}$$

$$\mathrm{T3} = \ln(Q10) * (TMAXZ_i - TOPTZi) \tag{1.51}$$

3.6. Design and implementation of the SALMO-OO simulation library

3.6.1. Design of simulation library structure using the object-oriented paradigm

The simulation library was designed based on the existing object-oriented SALMO-OO model structure for the phytoplankton biomass state variable. The object-oriented programming language Java was used to create a new class which can be accessed by the SALMO-OO model, but is still independent from the overall main model structure.

AlgaeLibrary.class contains methods and fields for each new phytoplankton growth and grazing model, thereby, reducing the need to change code within the original model and not compromise its structural integrity (figure 3.14). Each alternative growth or grazing model can be inserted into SALMO-OO and replace the original growth or grazing functions.

NOTE: This figure is included on page 53 of the print copy of the thesis held in the University of Adelaide Library.

Figure 3.14. UML (Unified Modelling Language) class diagram for the model components of SALMO-OO and the simulation library (AlgaeLibrary) (Zhang, 2006).

Through inheritance mechanisms the AlgaeLibrary.class is derived from the Phytoplankton.class, thereby gaining access to all functions (methods) and parameters that are defined by the parent class (Phytoplankton.class). The Phytoplankton.class is derived from the Salmo.class (a super class), which defines many of the common parameters and control methods necessary to run the SALMO-OO model. Therefore, any common parameters that are defined within the Salmo.class or the Phytoplankton.class can be used directly in defining the alternative growth or grazing models. For example, the parameter that defines the optimal temperature for phytoplankton growth (TOPTA*i*) is a constant parameter defined in the `Salmo.class`, and is used to calculate the temperature limited growth function for phytoplankton (PHOT*i*) in the `Phytoplankton.class`. This TOPTA*i* parameter is also required to define the alternative growth models in the `AlgaeLibrary.class`, and can be freely accessed due to the `AlgaeLibrary.class` inheriting attributes from the `Phytoplankton.class`. Whereas, if it was necessary to access a parameter from another class that is not within the inheritance structure then an object would need to be created to access that parameter to overcome any restrictions (e.g. `Math.exp()` – `Math` is an object instantiated from the `Math.class` and the `exp()` method is used to calculate the exponential function. This class is not linked to SALMO-OO through inheritance mechanisms). Thus, through these mechanisms the whole model structure is protected from undesirable access of parameter values or functions. Any constant parameters, such as the Q_{10} coefficient for temperature limited growth or saturated light intensity (Is), that are specifically used by the models contained in the `AlgaeLibrary.class` are stored in the `Salmo.class` along with all other common constant parameters that are needed by other components of the model.

Each alternative growth model has the same basic code structure. All are public methods with inputs for the biomass of each algae functional group (A*i*), phosphate (P), nitrate (N) and an index parameter, which allows access to the measured input data such as water temperature and solar radiation. A loop function allows the calculation of growth for each phytoplankton functional group. Within this loop growth-limiting functions for temperature, nutrients and underwater light are calculated as well as respiration and photosynthesis. The output for each growth method is the variable AGRO, which represents the specific growth rate for each phytoplankton functional group (figure 3.15). The grazing models are structured very similarly as for the growth models, with a loop to calculate grazing for each phytoplankton functional group, as well as links to the measured input data such as temperature and to zooplankton biomass. The output for each grazing method is the variable AGRA, which is the grazing rate for each phytoplankton functional group (figure 3.16). Each alternative growth or grazing method has a different name such as `growthAB()`, which describes the growth model from Arhonditsis and Brett (2005) or `grazCLEANER()`, which refers the CLEANER grazing model.

Within the `AlgaeLibrary.class` there is a method called `growth()`, which requires links to the phytoplankton biomass (A*i*), phosphate (P), nitrate (N), time (t), and underwater light (Appendix C). This method is also found within the `Phytoplankton.class` under the same name. Overloading methods is a form of polymorphism, a concept in object-oriented modelling where the same method can have several different meanings or tasks, depending on which object utilises them. Within the `AlgaeLibrary.class` the `growth()` method contains a `switch` function that allows the selection of alternative growth models when selected by a user. Therefore, if a user selected to use the growth model from CLEANER then the `AlgaeLibrary.class` would return the AGRO parameter, that calculates the specific growth rate for phytoplankton, based on the function defined by the `growthCLEANER()` method.

```
public double[] growthHJ(double[] A, double P, double N, int index)
 {
   double[] AGRO = new double[XLENGTH];// temporary variable

   double[] PHOT = new double[XLENGTH];
   double[] PHOP = new double[XLENGTH];
   double[] PHON = new double[XLENGTH];
   double[] PHOL = new double[XLENGTH];

   double[] SUMI = new double[XLENGTH];

   double[] RA = new double[XLENGTH];

   double L = 0;
   double EPS = Kw[0] + Kc[0] * ( A[0] + A[1] + A[2] );

    // Loop to calculate growth for x number of algae
   for( int i=0; i< XLENGTH; i++)
   {
         // Temperature limitation function

        PHOT[i] = T[index]/TOPTA[i]* Math.exp(1 - T[index]/TOPTA[i]);

         // Nutrient limitation function
        PHOP[i] = P / ( KP[i] + P );
        PHON[i] = N / ( KN[i] + N );

         // Light limitation function
        L=I[index]*(1Math.exp((1)*EPS*ZMIX[index]))/(EPS*ZMIX[index])
   ;
        IOM[i] = L / Is[i] * Math.exp( 1 - L / Is[i] );

         // Calculate respiration
        RA[i] = RO[i] * Math.exp( Tcoef * T[index] );

        PHO[i] = PHOMAX[i]* PHOL[i]* PHOT[i] * PHOP[i] * PHON[i];

        AGRO[i] = (PHO[i] - RA[i]) * A[i];

   }// end loop for groups

   return AGRO;

 }// end growth method HJ
```

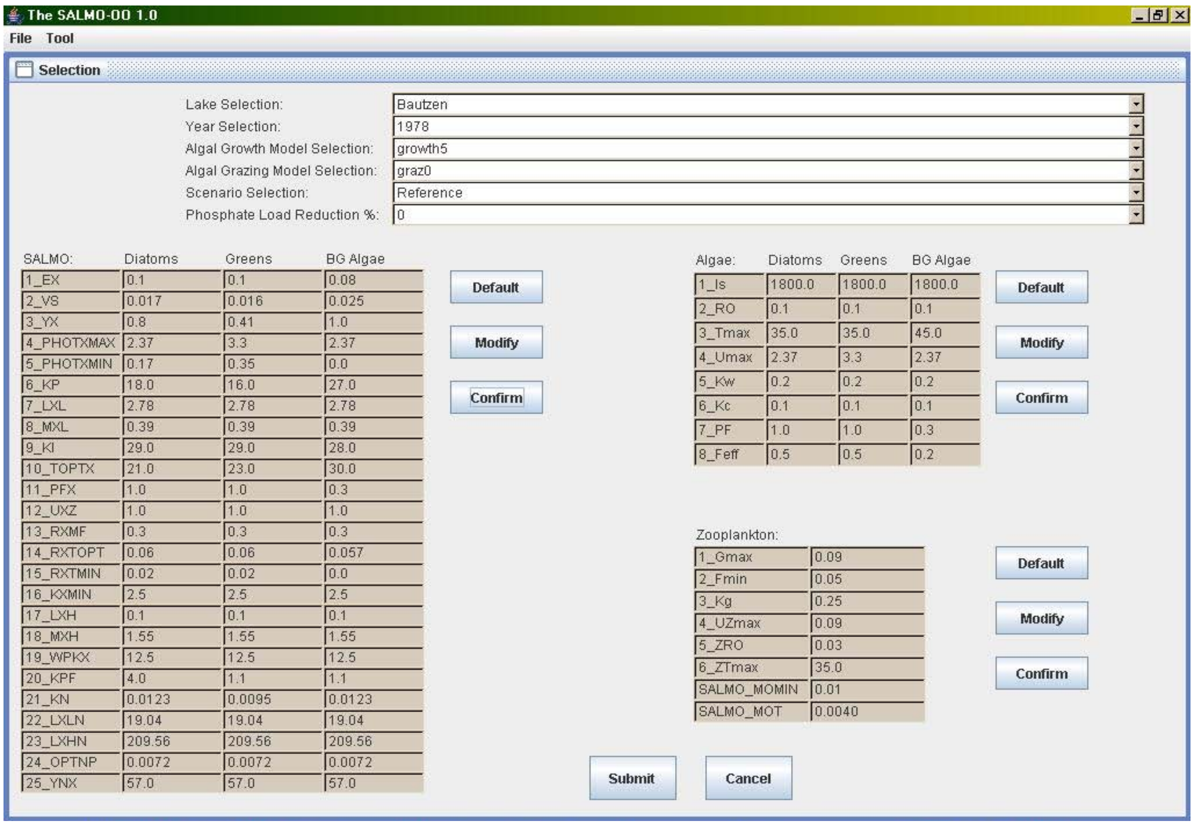
Figure 3.15. An example of the Java code for the growth method from Hongping and Jianyi (2002). Replacement of growth limiting functions, photosynthesis and respiration calculations can be made to customise for other growth models.

```java
public double[] grazHJ(double[] A, double Z, int index)
  {
    double[] AGRA = new double[XLENGTH];

    double F = 0;
    // Loop to calculate grazing for x number of algae
    for( int i=0; i< XLENGTH; i++)
    {
      F = F + PAF[i] * A[i];
    }

    F = F + PFD * A[3];// detritus

    // Detritus import
    GSUM[3] = GMAX * (F - FMIN) / ( KAG + F - FMIN ) * PFD * A[3] / F;

    for( int i=0; i<XLENGTH; i++)
    {
      GSUM[i] = GMAX * (F - FMIN)/(KAG + F - FMIN) * PAF[i] * A[i] /F;

      AGRA[i] = Z * GSUM[i];

    }

    return AGRA;
  }
```

Figure 3.16. An example of the Java code for the grazing method from Hongping and Jianyi (2002). Replacement of phytoplankton grazing and detritus grazing functions can be made to customise for other grazing models.

3.6.2. Implementation of the SALMO-OO simulation library

Once the alternative growth and grazing process model structures were encoded into the `AlgaeLibrary.class` the overall performance of the SALMO-OO model was tested, first using data from Bautzen and Saidenbach reservoirs. These data sets were used for testing because they were used to test the original SALMO model, are of high quality and include zooplankton biomass data, which can be difficult to obtain. A graphical user interface (GUI) was developed to facilitate the testing of each combination of alternative growth and grazing models for a variety of lake conditions. Parameter values that needed to be calibrated were included in the GUI as well as all of SALMO-OO's constant parameters. Drop down menus for selecting data sets, alternative growth and grazing functions, year and scenario analyses were included in the GUI (figure 3.17). The simulation results of state variable outputs are viewed directly within the SALMO-OO application (figure 3.18), including the measured data in order to compare observed and predicted outputs.



The SALMO-OO 1.0

File Tool

Selection

Lake Selection:	Bautzen
Year Selection:	1978
Algal Growth Model Selection:	growth5
Algal Grazing Model Selection:	graz0
Scenario Selection:	Reference
Phosphate Load Reduction %:	0

SALMO:	Diatoms	Greens	BG Algae
1_EX	0.1	0.1	0.08
2_VS	0.017	0.016	0.025
3_YX	0.8	0.41	1.0
4_PHOTXMAX	2.37	3.3	2.37
5_PHOTXMIN	0.17	0.35	0.0
6_KP	18.0	16.0	27.0
7_LXL	2.78	2.78	2.78
8_MXL	0.39	0.39	0.39
9_KI	29.0	29.0	28.0
10_TOPTX	21.0	23.0	30.0
11_PFX	1.0	1.0	0.3
12_UXZ	1.0	1.0	1.0
13_RXMF	0.3	0.3	0.3
14_RXTOPT	0.06	0.06	0.057
15_RXTMIN	0.02	0.02	0.0
16_KXMIN	2.5	2.5	2.5
17_LXH	0.1	0.1	0.1
18_MXH	1.55	1.55	1.55
19_WPKX	12.5	12.5	12.5
20_KPF	4.0	1.1	1.1
21_KN	0.0123	0.0095	0.0123
22_LXLN	19.04	19.04	19.04
23_LXHN	209.56	209.56	209.56
24_OPTNP	0.0072	0.0072	0.0072
25_YNX	57.0	57.0	57.0

Default Modify Confirm

Algae:	Diatoms	Greens	BG Algae
1_Is	1800.0	1800.0	1800.0
2_RO	0.1	0.1	0.1
3_Tmax	35.0	35.0	45.0
4_Umax	2.37	3.3	2.37
5_Kw	0.2	0.2	0.2
6_Kc	0.1	0.1	0.1
7_PF	1.0	1.0	0.3
8_Feff	0.5	0.5	0.2

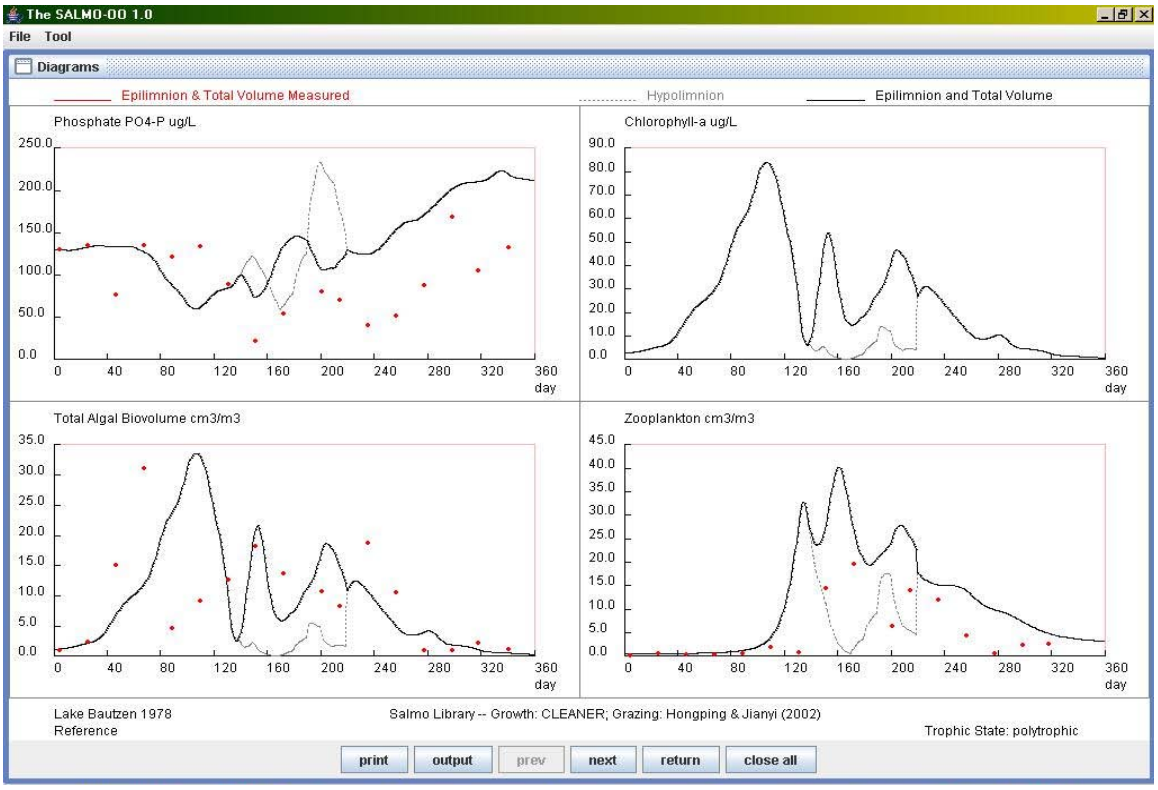
Default Modify Confirm

Zooplankton:	
1_Gmax	0.09
2_Fmin	0.05
3_Kg	0.25
4_UZmax	0.09
5_ZRO	0.03
6_ZTmax	35.0
SALMO_MOMIN	0.01
SALMO_MOT	0.0040

Default Modify Confirm

Submit Cancel

Figure 3.17. Graphical user interface (GUI) for the SALMO-OO simulation library.



Lake Bautzen 1978
Reference

Salmo Library -- Growth: CLEANER; Grazing: Hongping & Jianyi (2002)

Trophic State: polytrophic

Figure 3.18. Visual output of simulation results within the SALMO-OO application of each state variable and the measured data.

3.6.3. Selection criteria and assessment of the SALMO-OO simulation library performance

Nine lakes and reservoirs with different environment conditions were analysed using the SALMO-OO simulation library, with the aim of using alternative growth and/or grazing models to improve the SALMO-OO model results. A set of procedures and selection criteria was created to assess alternative model structures (figure 3.19). These assessment procedures and criteria were applied to each of the data sets tested, and are as follows:

1. Simulation runs were performed for each substitution of an alternative growth process model (i.e. one simulation run per each type of growth model), and similarly for each grazing process model. Manual (i.e. trial-and-error method) parameter calibration was performed for each model run until a common parameter set was obtained for the sensitive parameters RO (phytoplankton respiration rate) and Gmax (maximum zooplankton grazing rate).
2. Quantitative analysis of simulation results for phytoplankton, zooplankton (where data was available) and phosphate state variables.
   - Linear regression (r^2 values) comparing measured and simulated outputs for each state variable (performed in MS Excel)
   - Calculation of root-mean square error (RMSE) as an alternative measure of model fit

$$RMSE = \sqrt{\frac{(a_1 - c_1)^2 + (a_2 - c_2)^2 + ... + (a_n - c_n)^2}{n}} \qquad (1.52)$$

   where a = measured or observed data
   c = simulated data or model output
   n = number of data point that are compared

3. Based on the quantitative results each alternative model run was ranked according to phytoplankton predictions in order of best performing models.
4. Those models that produced improved statistical results compared to using the original SALMO-OO growth and/or grazing equations were selected for further analysis.
5. Qualitative analysis of simulation results for phytoplankton, zooplankton (where data was available) and phosphate state variables.
   - Determination of how well the model predicted the timing, magnitude and duration of major phytoplankton, zooplankton and phosphate peaks.
6. Seasonality of phytoplankton functional groups needed to be realistic of the trophic state of the site modelled. This was performed using the Ryding and Rast (1989) trophic state criteria, based predominantly on orthophosphate concentration (Table 3.4). Figure 3.19 illustrates the phosphate thresholds determining the trophic state of a given site. According to a lake's trophic state certain phytoplankton dynamics are typically observed. For example, for sites exhibiting low phosphate concentrations (0 – 5 μg L^{-1}) oligotrophic conditions are expected, where green algae and diatoms are expected to be dominant with very little observed blue-green algae (Reynolds, 1993). Therefore, the model combinations analysed needed to give realistic phytoplankton functional group simulations according to the trophic state of the lake before they were accepted for further analysis. The calibration parameter set obtained from step 1 was used during the

simulation of the combined phytoplankton growth and grazing model experiments.

7. Thus, the growth or grazing models that performed quantitatively and qualitatively better then the original SALMO-OO model were selected to test combination of different growth and grazing models.
8. Once simulation runs of different growth and grazing model combinations were performed then the same procedures for assessing the results (steps 2 – 6) were followed.
9. The final model combination was chosen based on a compromise between the best quantitative results and the best visual model fit of the measured data. In most cases the ability of a particular growth and grazing model combination to simulate phytoplankton functional group dynamics or give a closer fit to the measured data were the ones selected as the final results. Usually, these models performed highly in regards to the statistical analysis, but they were not always the ones that produced the best results for r^2 and RMSE.

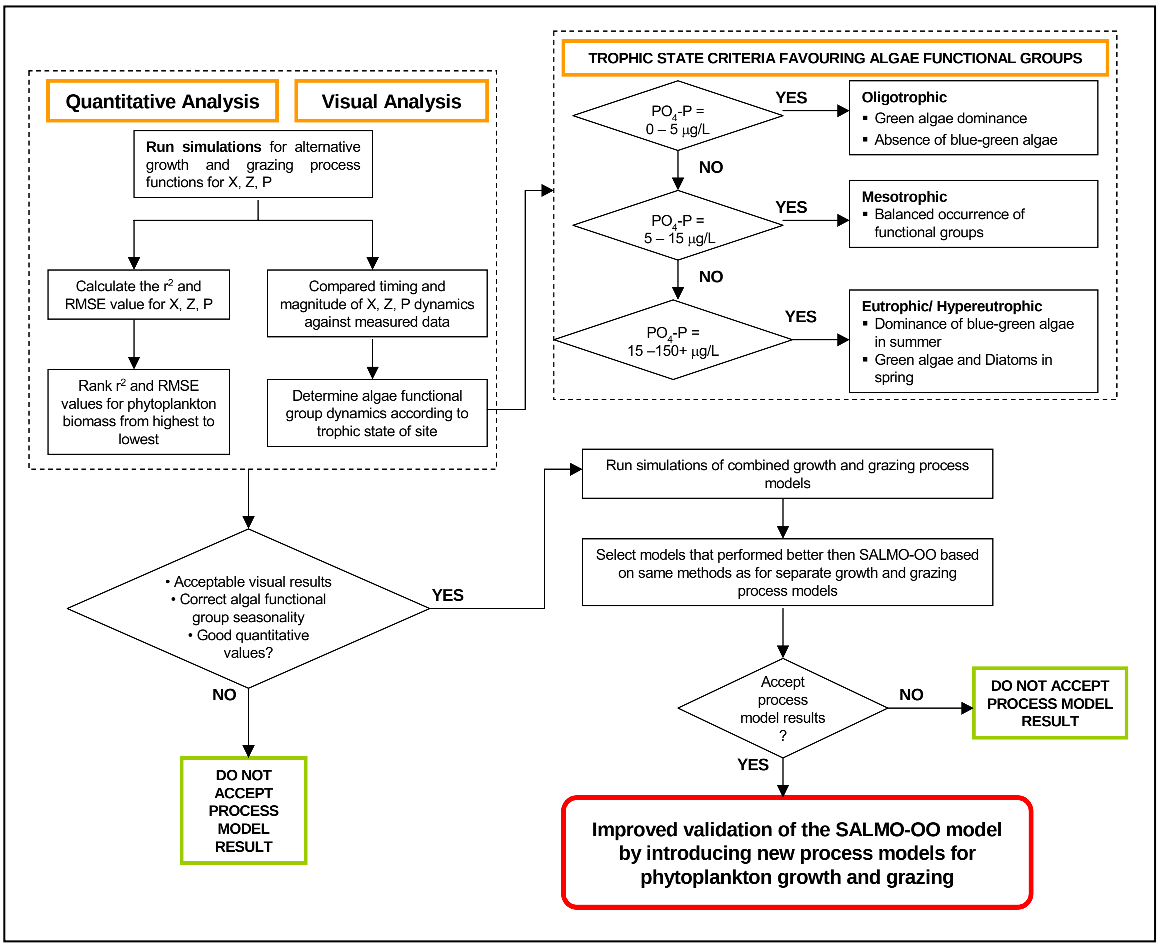

Quantitative Analysis
Visual Analysis
Run simulations for alternative growth and grazing process functions for X, Z, P
Calculate the r² and RMSE value for X, Z, P
Compared timing and magnitude of X, Z, P dynamics against measured data
Rank r² and RMSE values for phytoplankton biomass from highest to lowest
Determine algae functional group dynamics according to trophic state of site
TROPHIC STATE CRITERIA FAVOURING ALGAE FUNCTIONAL GROUPS
PO4-P = 0 – 5 µg/L
YES
Oligotrophic
▪ Green algae dominance
▪ Absence of blue-green algae
NO
PO4-P = 5 – 15 µg/L
YES
Mesotrophic
▪ Balanced occurrence of functional groups
NO
PO4-P = 15 –150+ µg/L
YES
Eutrophic/ Hypereutrophic
▪ Dominance of blue-green algae in summer
▪ Green algae and Diatoms in spring
Run simulations of combined growth and grazing process models
Select models that performed better then SALMO-OO based on same methods as for separate growth and grazing process models
• Acceptable visual results
• Correct algal functional group seasonality
• Good quantitative values?
YES
NO
DO NOT ACCEPT PROCESS MODEL RESULT
Accept process model results ?
NO
DO NOT ACCEPT PROCESS MODEL RESULT
YES
Improved validation of the SALMO-OO model by introducing new process models for phytoplankton growth and grazing

Figure 3.19. Selection criteria and assessment procedures for analyses of alternative growth and/or grazing model structures to improve the validation of the SALMO-OO model.