There's more to a model than code: understanding and formalizing *in silico* modeling experience

Janet Wiles[1,2], Nic Geard[1,2],
James Watson[1,2], Kai Willadsen[1,2]
[1]School of Information Technology and Electrical Engineering, The University of Queensland
[2]ARC Centre for Complex Systems
QLD 4072 Australia
+71 6 3365 2902
<wiles, nic, jwatson, kaiw>@itee.uq.edu.au

John Mattick[3], Daniel Bradley[3,4],
Jennifer Hallinan[1,3,4]
[3]Institute for Molecular Biosciences
The University of Queensland
[4]ARC Centre for Bioinformatics
QLD 4072 Australia
+71 6 3346 2615
<j.mattick, d.bradley, j.hallinan>@imb.uq.edu.au

ABSTRACT
Mapping biology into computation has both a domain specific aspect – biological theory – and a methodological aspect – model development. Computational modelers have implicit knowledge that guides modeling in many ways but this knowledge is rarely communicated. We review the challenge of biological complexity and current practices in modeling genetic regulatory networks with the aim of understanding characteristics of the *in silico* modeling process and proposing directions for future improvements. Specifically, we contend that the modeling of complex biological systems can be made more efficient and more effective by the use of structured methodologies incorporating experience about modeling algorithms and implementation. We suggest that an appropriate formalism is Complex Systems Patterns, adopted from Design Patterns in software engineering. First steps towards building community resources for such patterns are described.

Categories and Subject Descriptors
D.2 [Software] D.2.10 [Design]
D.2.11 [Software architectures]: D.2.m Rapid prototyping, reusable software
I.6 [Simulation and modeling]
J.3 [Life and medical sciences]: biology and genetics

General Terms
Algorithms, Documentation, Performance, Design, Reliability, Experimentation, Human Factors, Standardization, Languages.

Keywords
Complex systems modeling, *in silico* biology, complex systems biology, design patterns, complex systems patterns, genetic regulatory networks, artificial genomes, visualization

Gecco'05, June 25--29, 2005, Washington, DC, USA.

1. INTRODUCTION
The promise of *in silico* biology – the development of computational models of biological systems – is yet to be fully realized. After centuries of reductionist science's spectacular successes in revealing the components of biological systems, increasing recognition is being given to integrating such knowledge to understand systems as a whole. Methodological as well as domain-specific issues need addressing to realize the potential.

In this paper we review the challenges of *in silico* modeling, both methods and domain (section 1). Communication is a major issue in modeling, both between computational and biological experts and between modelers within and across different projects. By reflecting on the modeling practices in our own group and collaborators (section 2) we propose a way to improve modeling methods by understanding the inherent aspects of biological modeling and formalizing the experience of modelers. Although publications report what was done in a study, they rarely report why it was done or what alternatives were rejected. Such knowledge is the implicit knowledge of experienced modelers and is critical both for training new modelers and streamlining the process of model development. We propose that the field is reaching a degree of maturity where knowledge about common practices of modeling can be identified and formalized using a technique developed for other fields called *patterns* (section 3). In particular, we call the types of patterns in this field *complex systems patterns* in recognition of specific methodological issues related to the challenges of *in silico* modeling (described in sections 1.1 and 1.2).

1.1 The challenge of biological complexity
One of the most complex processes in biology is the development of an organism from a single cell, via division and differentiation [38]. Each cell in the adult organism contains essentially the same control program – the genetic regulatory network embedded in its genome – but each cell follows its own specific trajectory, based on a combination of intrinsic dynamics, interactions with neighboring cells, morphogen gradients and other environmental factors [6].

The control of development spans multiple temporal and spatial scales from molecular motion on the nanosecond time scale to the lifespan of the adult. Through organization at the level of genes, cells, tissues, organs and whole organisms, development links a

genotype to a phenotype via an incredible variety of entangled processes.

When development is framed in an evolutionary context, the temporal and spatial ranges involved increase dramatically. Evolutionary changes occur at the molecular levels of single nucleotides and stretches of DNA, but evolutionary selection acts proximally on the organism as a whole, and evolutionary events such as speciation can occur over millions of years and across vast geographical distances. The relationship between development and evolution is bidirectional: while micro-level evolutionary changes to genomes are the ultimate agents of change that underpin evolving morphologies, the effects of these changes are mediated by development in ways that can eventually constrain the direction of macro-level evolution. The mechanisms and their effects may be orders of magnitude apart in temporal and spatial dimensions.

1.2 Roles for complex systems modeling

A computational model encapsulates a theory about the essential mechanisms and the way in which those mechanisms give rise to the behavior of a system.

A core challenge for computational modeling is to investigate how the functionality of a system arises from its component parts. In this context, 'functionality' is more than just structure and dynamics: it requires simulating the system's components and their interactions and then observing the degree to which the resulting behaviors match the phenomenon of interest. We refer to models of such systems as *complex systems models*.

Complex systems modeling is a way to formalize theories about the nature of components and their multi-scale interactions. Because these models can be implemented using computer simulations, it is possible to develop and check the internal consistency of biological theories that have too many nonlinear interactions to be understood intuitively. They also provide a way to explore the behavior of systems, formulate and evaluate hypotheses and they have the potential (albeit as yet rarely realized) to guide the direction of future empirical research.

The complex systems field has developed an impressive set of tools for studying systems that *generate* complexity. An early discovery was how simple systems can generate complex patterns of behavior and how networks of interacting components behave. In any system with several elements – nodes in a network, cells in a cellular automata, rules in a generative grammar – even a modest number of interactions between the elements can transform the system from exhibiting stable dynamics to combinatorial complexity and even chaotic behavior [20, 27].

However, the results of applying such models to understanding biological functionality have been modest as measured by its limited impact as a methodology for mainstream biology. In applying complex systems methods to biological systems, it is apparent that generating complexity from simple components is not the difficult issue; rather, it is controlling complex behavior in a way that is simultaneously robust and flexible. The difficulty lies in reverse engineering a complex behavior to identify the underlying generators. In this sense, complexity is easy to generate but hard to control.

In recent years, there has been an increasing appreciation of the need for more rigorous models and frameworks that bridge the gap between biological grounding and computational theory of multi-scale interactions [9, 22, 39].

In addition to models and frameworks, we also think that attention to software engineering practices in the field as a whole can provide substantial benefit to modelers and biologists alike. To understand the options for improvement the next sections review current practices.

1.3 What's wrong with the status quo?

It might be argued that the current *ad hoc* approach to modeling is perfectly adequate. After all, the current approach has been successfully applied to problems in many domains and (continues the argument) there is nothing special about complex systems, biology, or the interface between them.

A further suggestion is that standards for modeling already exist. Indeed, it could be argued that biology has an embarrassment of standards. As well as attempts at biological standardization, such as gene ontologies [7] and a plethora of databases defining genes, genomes, proteins, interactions, etcetera, there are computational standards such as XML [31] and various cell modeling languages like CellML [25] and the Virtual Cell modeling and simulation framework [30]. Adding yet another layer of complexity to the process of in silico modeling might be seen as futile at best, and a waste of time and energy at worst.

The main evidence supporting the contention that the incorporation of additional structure into the modeling process is unnecessary and even counter-productive lies in the advances that have already been made using the existing approaches to complex systems modeling. Particularly noteworthy in this regard are the achievements of network researchers over the past five years or so. During this time network analysis has moved from an abstract topic of purely mathematical interest (e.g. [11]) to a vital field applicable to domains as diverse as social networks [23], the internet [3], economics [12], physics [24] and biology at all scales [15, 36]. The realization that aspects of network evolution, topology and dynamics are common to all of these areas and more has arisen directly from the application of standard modeling techniques and existing standards to issues in complex systems.

This success is, however, the exception rather than the rule. Although modelers have embraced techniques such as neural networks, agent-based modeling and cellular automata with enthusiasm, such approaches tend to be, in most biological contexts, solutions in search of a problem. Although the current approach has been valuable in some domains, biological systems tend to be too complex and poorly understood to model effectively. None of these approaches has, to date, yielded insights into biological systems which have been heralded as unique and valuable by biologists themselves, although some individual models show promise of providing biologically valuable results.

In addition, it cannot be denied that the current approach of small teams of modelers working more-or-less in isolation, using tools and techniques with which they are familiar has two serious drawbacks. One is the problem of reinventing the wheel; that is, squandering valuable time and resources on solving problems which have already been addressed successfully by others; while the other is that of inventing square wheels, that is, solving a problem but in a manner inferior to that which has already been done. Both of these drawbacks lead to less effective and efficient models.

2. FIRST HAND EXPERIENCES

Our own work concerns the multiple levels of complexity from genotype to phenotype (see Fig 1). In particular, we are interested in designing suites of models that traverse the path from DNA to organism via a series of interlinked simulations, each addressing the transition between two adjacent levels of description.

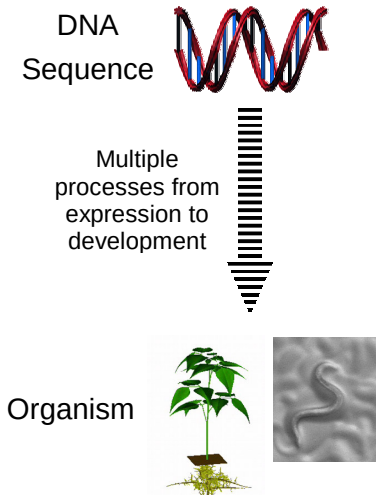


Figure 1. The processes that link DNA to organism can be studied at many different spatial and temporal levels.

Finding the 'right' components and interactions to include in a model requires understanding their functional consequences for the theory under consideration. Such a criterion is easy to state, but honing a model until it satisfies this requirement constitutes a major part of the design effort in many modeling projects.

We use a framework based on the structure, dynamics and function of systems. The core computational component of this chain of models is the genetic regulatory network in a developing organism (see Fig 2).

Here we briefly review the range of studies in our group, and refer the reader to published papers for more details. The research in our group can be divided into two methodological categories: the development of tools and theoretical insights to aid in exploring and understanding the behavior of models; and the development of models of biological systems, at both a general and more specific level.

2.1 Structure, dynamics and function

Our more abstract models are aimed at developing a theoretical understanding of gene interactions in purely network models [26], network behavior and its dynamics [17, 18], and stability to perturbations [29].

Grounding our understanding gained from these gene interaction models in biological simulations requires understanding both the lower level of detail that gives rise to the network components and interactions, and also the higher level of behavior that the networks are intended to control.

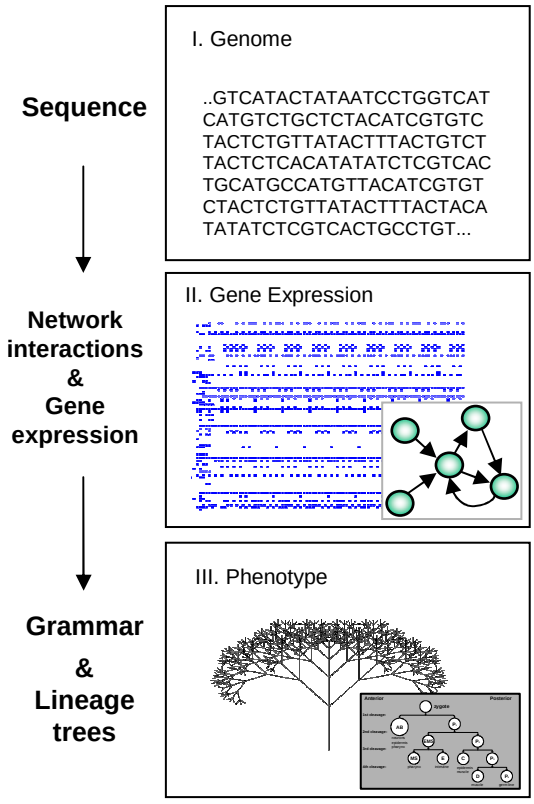


Figure 2. Three levels at which computational models of genotype to phenotype can be expressed. The genome is represented as a sequence of nucleotides; the regulatory system is represented as a network of interacting nodes; and the phenotype is represented either as a linear tree or as a grammar. The genetic regulatory network is the computational controller in the developmental dynamics.

Genetic regulatory systems that control the division and differentiation of cells are modeled as networks of interacting elements (which can represent a variety of factors such as genes, noncoding RNA, intra- and extra-cellular signals and environmental factors). The nodes in the network are derived from a model of a genome specified as a nucleotide sequence, and the links between nodes that indicate their interactions are derived from a simplistic model of molecular chemistry. To model evolutionary effects, changes in the links between the nodes in a network can be derived from mutations at the level of the nucleotide string. By using an artificial nucleotide string and modeling mutation operators (e.g. point mutation, duplication, translocation), more realistic change operators can be included in the development of the genetic regulatory network [32]. An artificial genome also enables us to investigate the power of control elements such as non-coding RNA, that are outside the standard model of gene regulation [21, 29].

To integrate our models into a broader behavioral context, we use the genetic regulatory network models to control a variety of different representations of phenotypes. A representation that can be derived directly from biological data is the diverging cell lineages of simple multi-celled organisms (Fig. 3). One of the simplest multi-celled organisms much studied by developmental biologists is the nematode *C. elegans*. It has just 959 cells as an adult and each worm develops in almost exactly the same

sequence of cell divisions, to the extent that the lineage tree for the complete organism has been well characterized [28]. Our group is using models of genetic regulatory networks to control the division and differentiation events of a variety of lineage trees from simple artificial systems to the full complexity of the *C. elegans* lineage [5, 14].

2.2 *In silico* phenotypes

The generation of lineage trees forms a family of benchmark problems. By 'generation of lineage trees', we do not just mean a description of the tree structure itself, but rather, the task requires a control system analogous at some level to cellular control to produce signals that drive the division and differentiation of cells in each branch of the lineage.

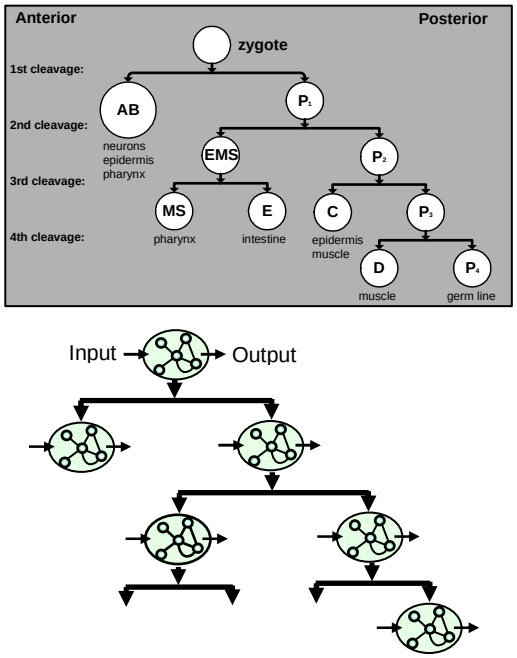


Figure 3. Lineage trees. (above) Branches of nematode lineages can be used as a biologically grounded reference on the lineage complexity that occurs in real organisms. (below) The controllers for lineage trees can be defined in different ways. In this example the controllers are genetic regulatory networks (indicated by networks) that interact with their cellular environment (indicated by the input arrows) and produce signals such as division and differentiation (indicated by the output arrows). A family of benchmark problems can be designed for a particular type of controller by varying the specificity of the output signals required and the size and complexity of the lineage trees.

A graded family of problems can be created by varying the difficulty of the tasks, in terms of the depth of the tree, the amount of information available at each cell, and the complexity of the lineages. The models can also incrementally incorporate additional components of biological complexity [4, 5] (see Figure 3).

Another benchmark problem is the control of plant development, which can be computationally specified by L-systems. Again, in our research we use a genetic regulatory network to control the system, but in this case, the network only needs to control the parameters of the L-system, rather than every cell in the plant. Mutating the parameters of a plant model produces a broad range of phenotypes [34].

The evolution of the plant L-systems can be compared to the *C. elegans* lineages. The architectures for the two models differ in complexity, as hundreds or thousands of parameters are used in the lineage tree simulations compared to less than ten variables to control the plant phenotype. The variables in plant growth are expected to be easy to evolve because the complexity of the system has been encapsulated in the L-system, which is a generative grammar that expresses just those aspects that enable the system to evolve coherently. A specified set of parameters enables the mutations to be easily aligned with the phenotypic selection.

In current work we are extending the plant studies using selection criterion based on measured properties of the resulting phenotypes, trading off factors such as leaf coverage versus water consumption. Many additional methodological challenges remain to be addressed. One, in particular, concerns the connection between the control of cell divisions that create lineage trees, and the way in which such systems can be understood as grammars, or parameterized morphological forms, such as L-systems.

2.3 Long term research questions

The family of models developed to date constitutes a framework that includes the levels of artificial genome (nucleotide strings), genetic regulatory networks, and different models of phenotype based on both lineage trees and generative grammars.

This overarching framework facilitates the study of a range of complementary questions that address relationships between levels of description, questions about developmental processes, and evolutionary question. Specifying one aspect of the system, such as the lineage tree model of development enables comparison between different models of genetic regulatory systems and the effect of different genotypic representations.

Conversely, fixing the model of genetic regulation enables comparison of the control of different phenotypic systems and exploration of the evolutionary history of various body plans and the phylogenetic relationships between them.

The models also enable the investigation of a variety of questions in the control of complexity. In particular, we are investigating the role of non-coding RNA in cell regulation and its potential influences on evolution and development of complex organisms [21, 29]. This is a long term project, with many converging lines of evidence. The *in silico* studies provide a platform for comparing the computational power of genetic architectures with and without non-coding RNA regulation.

Cellular control systems are remarkably robust, but many factors can damage or subvert the natural controls. When gene regulation is interfered with, the resulting cells frequently die, but those that survive have the potential to form tumors. Cancers were once thought of as diseases of genes, but are increasingly being studied

as diseases of gene regulation. Networks can be used to simulate the dynamics of the genes and signals that regulate P53, a gene that has been implicated in more than half of all cancers [16].

3. MODELING PRACTICES

The aim of the final section of this paper is to discuss modeling issues and practices and ways to enhance them. As one might expect, the characteristics of model development for *in silico* research differ from traditional software engineering. It helps to highlight where major differences lie and consequences for effective research.

3.1 Issues specific to *in silico* modeling

As in any science, the ultimate goal of an *in silico* model is to gain insight into the phenomena of interest. However, the structure of models that might provide insights into system-level properties is often non-obvious. Biological systems span multiple levels of time and space, and interactions link the systems from the very bottom to the top and back again. In model design a constant tension exists between fidelity to biological detail and the right level of abstraction at which to model causal factors of interest.

Three issues are fundamental to the modeling of complex biological systems:

1. Choosing tasks that incorporate appropriate challenges

A modeling task can be thought of as what a system does. Typically tasks have a high level description and a computational specification. For example, in the *C. elegans* lineage simulations, the conceptual task is for a genetic regulatory network to control the development of the lineage tree of a nematode starting from a single cell. Simpler versions of the task could involve controlling branches of the tree. To turn the conceptual task into a simulation, it needs to be specified as a mapping between a set of inputs and outputs. For example, in the cell lineage task, the inputs could be the position of each cell at each point in time and information from their neighbors. The required output from the controller could be whether each cell divides or differentiates at each point in time.

Tasks are points of communication between biologists and computational specialists and for a model to be relevant, the conceptual tasks need to be interesting and important to biologists. A complaint from biological colleagues against some artificial life models is that they stray too far from real biological systems for their results to be applicable [10]. Clearly, modeling tasks need to be chosen with care, as there is a risk that toy worlds can focus attention on problems that are rare or non-existent in biology while omitting or sidelining fundamental issues that are core to biological theorizing. When models abstract away from biological details to the extent that they can no longer relate to real world data, the onus is on the modelers, not the biologists, to demonstrate the usefulness of their techniques.

A counter complaint from computational modelers is that many biologists appear to be disinterested in simulations that address general characteristics of the systems they study. At the two extremes, these positions need not communicate, since there are biological questions for which insight into higher levels of organization has little utility, just as there are complexity questions for which biological grounding has minimal impact. However, where complex systems modeling is being used to develop and advance biological theory, dialogue is critical.

2. Choosing the appropriate level of abstraction

One of the most common phrases the computational modeler hears, on (she believes) grasping the essence of a biological system of interest is "Well actually, it's not that simple." In biology, the devil is in the detail. There is a temptation to believe errors in biological models always occur because too little detail has been included and that additional biological facts would enhance the model.

However, not all aspects of a system have equal explanatory power. Searching for the causal factors requires the right level of detail, omitting details not relevant to functionality of the system.

Consider the phenomena of a traffic jam. The components on a highway could include cars, trucks and other vehicles. In trying to understand traffic jams, some understanding of vehicles is needed, as well as their speeds, directions and distances from one another. However, specific details about the details of each vehicle's engine would detract from understanding the *essence* of a traffic jam. An appropriate level of abstraction would transfer to other traffic jams such as bicycles or people in subways, which have speeds and directions, but not engines.

All models, by necessity, neglect aspects of biological reality. However, some models, when framed at an appropriate level of abstraction, provide an understanding of a system's behavior that could not have been obtained at a higher or lower level of detail. Choosing the right level at which to model components and their interactions requires understanding both how the components arise from a lower level of detail, and how the behavior that they generate integrates into the system at a higher level of description.

3. Appreciating the nature of the design process

The design process for *in silico* research involves both technical and social aspects. Models are not created in a single design session at the beginning of a project, but rather they evolve with the understanding of both the modeler and the domain experts. Details are added and removed, tasks are refined, and understanding the link from mechanism to behavior is gradually unfolded. There is typically a long lead time in developing an effective *in silico* model.

For example, a classic simulation of the Baldwin effect (the effect of learning upon evolution) was first published as an elegant model using a vector of just 20 elements [19]. It is a simple model that can be replicated easily. However, the original modeling effort consisted of more than twenty designs over several months as the essential aspects of the system were understood and the simulations were refined.

The inherently iterative nature of the *in silico* design process has to be taken into account in communications within the modeling team and between modelers and domain experts.

3.2 More than code

Computer modeling is sometimes mistakenly seen as merely programming, with a model comprising nothing but its code. A central tenant of programming (succinctly expressed in the title of a classic text by Wirth [37]) is that the finished product – the program – is the sum of its algorithms and data structures:

Algorithms + data structures = programs

Software engineers realized that there is more to programming than the finished product. The human team in a modeling project brings a wealth of experience to the design task. Such experience includes background knowledge of the strengths and weaknesses of algorithms, efficient implementations, parameter choices, reusable and extendable designs, rapid prototyping techniques, extensive knowledge of the domain of interest, and much more.

Wirth's tenant could be adapted for the modeling process as:

Algorithms + data structures + experience = modeling

The art of modeling includes understanding the options and the tradeoffs that are appropriate to the domain to be modeled. Modelers frequently share code or pseudocode, but code only contains the options that were chosen, not why they were appropriate or what the alternatives were. We argue that the complex systems modeling community is ready for the explicit acknowledgement and communication of such information. The key question is how to formalize the knowledge of experienced modelers.

3.3 Lightweight software practices

The majority of models in our group (and amongst others we have informally surveyed) are developed by teams of one to six people. In practice, no matter how much thought is put into a prototype model, or how experienced the software engineer, virtually all prototype simulations are substantially rewritten as appropriate directions for generalization emerge. An appropriate balance between breadth and depth is required to facilitate effective modeling.

Recently we have been reviewing complex systems practices and software engineering literature to determine the characteristics of *in silico* modeling projects, and to find appropriate techniques to enhance the speed and quality of modeling studies. Through an online survey of members of the ARC Centre for Complex Systems Genetic Regulatory Network (GRN) group and focused interviews, four characteristics were identified as important for understanding current GRN research projects [33]:

1. Team sizes are typically small and even with a modeler and a domain expert collaborating, the modeler is normally the sole user of the software.
2. Software is commonly configured to address a single research question, and frequent redesign and reuse of components from current and previous projects is normal practice.
3. The specifications for any project are rarely stable for any length of time. Both the modeler and the domain expert iteratively improve their understanding of both the underlying biological system and how the evolving model converges on essential mechanisms and behaviors of interest.
4. The non-linear behavior of GRN models means that outcomes are generally unknown before run time.

The characteristics of small teams and rapidly evolving projects mean that lightweight techniques are the most likely to be beneficial in practical studies. Lightweight techniques can be used to aid implementation at the component level, manually tracking interactions for very small systems, and extensive use of visualization of system structures and behaviors [33].

The software that supports model development is a *de facto* language for expressing and testing theories. Some software may be so specific that it facilitates only a single simulation. Other software may be so general that any simulation requires considerable effort. The choice of which is more appropriate depends on the size and scope of the modeling project.

3.4 Complex Systems Patterns as a formalism

We believe that it is too early to standardize specific theories or complete methodologies for mapping biology into computation. Computational modeling of the processes from genotype to phenotype uses many different methodologies to represent the components and interactions of biological control mechanisms.

However, all experienced modelers possess implicit knowledge about modeling; insights into effective and efficient ways of designing and developing models, visualizing system behaviors, and analyzing issues of interest such as robustness and stability, scalability and evolvability. These insights – practical techniques, efficient algorithms, useful rules of thumb, extendable model designs – are the implicit wisdom of individual modelers and small groups. Experienced modelers also have insights into seemingly sensible approaches that do not work in practice. Unfortunately, this wisdom is rarely communicated, even within the modeling community, and many of the same insights are rediscovered time and time again. A widely accepted medium for communicating such knowledge is urgently required. To this end, we have begun to formalize insights from our own modeling projects and started collaborations with colleagues in ACCS, CSIRO and COSNet[1] using the software engineering concept of *patterns* [8, 13].

A pattern is a proven solution to a commonly recurring problem [13]. They can occur in both model design and in systems performance. A pattern is expressed as a brief description of the problem and its solution, using ten standard headings: Name; Intent; Motivation; Applicability; Example; Consequences; Implementation; Sample Code; Known Uses; and Related Patterns. An example pattern could be a standard visualization process, such as an activation diagram (shown in the next section).

Complex systems modelers would benefit from patterns that occur in the design of models, such as patterns guiding the choice of benchmark problems, software platforms, model architectures, analysis and visualization techniques. Patterns also occur in system behaviors, such as emergent robustness, evolvability, efficient connectivity, and modular design. Since computational models encompass all the implicit wisdom of modeling (not just software), we call these 'Complex Systems Patterns'.

A more detailed description of the ideas behind the use of patterns as a method for capturing the implicit wisdom of the complex systems modeling community, and an example of a visualization pattern is described in [35].

3.5 Example Pattern: Activation Diagram [35]

One commonly-recurring problem in genetic regulatory network modeling is the visualization of system behavior, where interesting behaviors span multiple levels in time and space. To provide a concrete feel for the nature and scope of a complex system pattern, this section illustrates a prototype pattern which solves this problem.

Name: Activation Diagram (Classification: Dynamics-Local Visualization)

Intent: Visualize micro level interaction of components over time to see macro level characteristics.

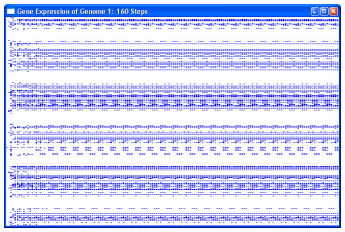
Also known as: Gene expression diagram, gene activation diagram, expression pattern, activation signature

Motivation: To gain insight into initial and long-term dynamics of a set of interacting components. Interactive extension allows the user to manually change the value of any component at an arbitrary point in time to visualize the effects of perturbation.

Applicability: Use the Activation Diagram pattern to:

- visualize the characteristics of component interactions over time when components have binary or real-valued states
- determine characteristics of macro level behavior such as ordered, cyclic, or chaotic activity
- assess the lifecycle of macro level behaviors (such as the number of steps before a network settles into a certain state)
- manually investigate robustness of macro-level behavior

Example Visualization: Time is shown along the *x* axis, and each component is positioned along the *y* axis. Active components are denoted by blue shading. This diagram shows the component interactions falling into a cyclic state after a short transient period.



Consequences: The Activation Diagram has the following consequences and inherent limitations:

- it requires access to the values of all components for each time step
- only a single starting state and trajectory is shown per diagram
- depending on screen size, large numbers of components can make viewing difficult
- very long cycles can appear similar to chaotic trajectories

Implementation: The Activation Diagram has the following important implementation variations:

1. time can be expressed along the *x* or *y* axis
2. more than two component states can be visualized through shading or color variations

Sample Code: Omitted for brevity. For details see [35].

Known uses: Gene expression, random Boolean networks, cellular automata, neural network dynamics.

Related patterns: State Space Diagram, 3D Network Display

4. CONCLUSION

In this paper we have reflected on the complex systems modeling process itself, with the aim of understanding the progress made to date, and proposing directions for future improvements. Specifically, we contend that the modeling of complex biological systems can be made more efficient and more effective by the use of structured methodologies incorporating experience about modeling algorithms and implementation.

Mapping biology into computation has both a domain specific aspect – biological theory – and a methodological aspect – model development. The power of an *in silico* model lies not just in the algorithms, but also in the task, the representational system and the architecture that facilitates the types of questions that the modelers wish to ask.

All models have common methodologies. More than just code or algorithms are required to design and communicate complex systems models effectively and efficiently. Design Patterns are a well established formalism in software engineering that describe algorithms and contextual information about them in a structured manner. We suggest that they can fill the need to incorporate experience into the modeling process for the field of complex systems biology modeling.

Our aim in proposing the formalism of complex systems patterns is to build a community of modelers sharing knowledge and experience to mutual benefit. All models contain aspects that can be efficiently and effectively described in the patterns formalism and we suggest that all modelers could contribute to codifying their experience by defining their own patterns and/or refining proposed patterns from other modelers. As a point of focus for collecting and distributing patterns we have established an online patterns repository [1] and meetings and workshops to discuss the proposal and refinement of patterns [2].

5. ACKNOWLEDGMENTS

This research was funded by an ARC grant to JW, JSH, and JSM, an ACCS grant to JW and JW and an APA to NG.

[1] ACCS: ARC Centre for Complex Systems http://www.accs.uq.edu.au/
ARC: Australian Research Council http://www.arc.gov.au/
CIS: Division of Complex and Intelligent Systems Research
COSNet: Complex Open Systems Research Network
http://www.complexsystems.net.au/
CSIRO: Commonwealth Scientific and Industrial Research Organisation
http://www.csiro.au/

6. REFERENCES

[1] "Complex Systems Patterns Repository. http://www.itee.uq.edu.au/~patterns/repository/."

[2] "Complex Systems Patterns Workshop I June 2005 http://www.itee.uq.edu.au/~patterns/PatternsWorkshopI.html ; Complex Systems Patterns Workshop II December 2005 http://www.itee.uq.edu.au/~patterns/PatternsWorkshopII.html."

[3] R. Albert, H. Jeong, and A.-L. Barabasi, "Error and attack tolerance of complex networks," *Nature*, vol. 406, pp. 378-382, 2000.

[4] R. B. R. Azevedo, R. Lohaus, V. Braun, M. Gumbel, M. Umamaheshwar, P. M. Agapow, W. Houthoofd, U. Platzer, G. Borgonie, H. P. Meinzer, and A. M. Leroi, "The

simplicity of metazoan cell lineages," *Nature*, vol. 433, pp. 152--156, 2005.
[5] V. Braun, R. B. R. Azevedo, M. Gumbel, P. M. Agapow, A. M. Leroi, and H. P. Meinzer, "ALES: cell lineage analysis and mapping of developmental events," *Bioinformatics*, vol. 19, pp. 851--858, 2003.
[6] S. B. Carroll, J. Grenier, and S. D. Weatherbee, *From DNA to Diversity: Molecular genetics and the evolution of animal design*. MA: Malden: Blackwell Science, 2001.
[7] T. G. O. Consortium, "Gene Ontology: tool for the unification of biology," *Nature Genetics*, vol. 25, pp. 25-29, 2000.
[8] J. O. Coplien, "Software design patterns: common questions and answers," in *The Patterns Handbook: Techniques, Strategies, and Applications*, L. Rising, Ed. NY: Cambridge University Press, 1998, pp. 311-320.
[9] E. A. Di Paolo, J. Noble, and S. Bullock, "Simulation models as opaque thought experiments," in *Artificial Life VII: Proceedings of the Seventh International Conference on Artificial Life*, M. Bedau, J. McCaskill, N. Packard, and S. Rasmussen, Eds. Cambridge, MA: MIT Press, 2000, pp. 497--506.
[10] D. Endy and R. Brent, "Modelling cellular behaviour," *Nature*, vol. 409, pp. 391-395, 2001.
[11] P. Erdos and G. Katona, "Proceedings of the colloquium held at Tihany, Hungary, September 1966," 1966.
[12] J. D. Farmer and F. Lillo, "On the Origin of Power-Law Tails in Price Fluctuations," *Quantitative Finance*, vol. 4, pp. 7-11, 2004.
[13] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*: Addison-Wesley, 1994.
[14] N. Geard and J. Wiles, "A Gene Network Model for Developing Cell Lineages," *Artificial Life*, vol. 11, 2005.
[15] P. M. Gleiser, F. A. Tamarit, and S. A. Cannas, "Self-organised criticality in a model of biological evolution with long-range interactions," *Physica A : Statistical Mechanics and its Applications*, vol. 275, pp. 272, 2000.
[16] J. Hallinan, "Cluster analysis of the p53 genetic regulatory network: Topology and biology," in *IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, vol. 7-8 October. San Diego, 2004.
[17] J. Hallinan, "Gene duplication and hierarchical modularity in intracellular interaction networks," *BioSystems*, vol. 74, pp. 51-62, 2004.
[18] J. Hallinan and J. Wiles, "Asynchronous Dynamics of an Artificial Genetic Regulatory Network," presented at ALife IX The 9th International Conference on Artificial Life, Boston, Ma, 2004.
[19] G. E. Hinton and S. Nowlan, "How learning can guide evolution," *Complex Systems*, vol. 1, pp. 495 -502, 1987.
[20] S. A. Kauffman, *The Origins of Order: Self-Organization and Selection in Evolution*. Oxford: Oxford University Press, 1993.
[21] J. S. Mattick, "The hidden genetic program of complex organisms.," *Sci Am*, vol. 291, pp. 60-7, 2004.
[22] G. E. Miller, "Artificial Life as Theoretical Biology: How to do real science with computer simulation," School of Cognitive and Computing Sciences, University of Sussex 1995.
[23] M. E. Newman, "The structure of scientific collaboration networks," *Proceedings of the National Academy of Sciences of the USA*, vol. 98, pp. 404 - 409, 2001.
[24] M. E. J. Newman, C. Moore, and D. J. Watts, "Mean-field solution of the small-world network model," *Physical Review Letters*, vol. 84, pp. 3201, 2000.
[25] P. Nielsen, "CellML http://www.cellml.org/ Last accessed 27 April 2005."
[26] B. Skellett, B. Cairns, N. Geard, B. Tonkes, and W. J, "Rugged NK landscapes contain the highest peaks," in *To appear in the Proceedings of GECCO*, 2005.
[27] R. V. Sole, P. Fernandez, and S. A. Kauffman, "Adaptive walks in a gene network model of morphogenesis: insights into the Cambrian explosion," *International Journal of Developmental Biology*, vol. 47, pp. 685--694, 2003.
[28] J. E. Sulston, E. Schierenberg, J. G. White, and J. N. Thompson, "The embryonic cell lineage of the nematode Caenorhabditis elegans," *Developmental Biology*, vol. 100, pp. 64--119, 1983.
[29] B. Tonkes, J. Wiles, and J. S. Mattick, "Controlling Complexity in Biological Networks," in *Poster presented at the 11th International Conference on Intelligent Systems for Molecular Biology*, 2003.
[30] VirtualCell, "National Resource for Cell Analysis and Modeling (NRCAM) http://www.nrcam.uchc.edu/ Last accessed 27 April 2005."
[31] W3C, "Extensible Markup Language (XML) http://www.w3.org/XML/ Last accessed 27 April 2005."
[32] J. Watson, N. Geard, and J. Wiles, "Towards more biological mutation operators in gene regulation studies," *BioSystems*, vol. 113, pp. 239--248, 2004.
[33] J. Watson, J. Hanan, and J. Wiles, "Practical software engineering techniques for regulatory models in biology. http://www.itee.uq.edu.au/~patterns/papers/ unpublished," Technical Report for the ARC Centre for Complex Systems. 49 pages. 2005.
[34] J. Watson, J. Wiles, and J. Hanan, "Towards more relevant evolutionary models: Integrating an artificial genome with a developmental phenotype," in *The Australian Conference on Artificial Life (ACAL 2003)*, 2003, pp. 288-298.
[35] J. Wiles and J. Watson, "Patterns in Complex Systems Models," in *IDEAL*, M. Gallagher, et al., Ed. Brisbane: Lecture Notes in Computer Science (LNCS), Springer Verlag, in press., 2005, pp. 8 pages.
[36] R. J. Williams and N. D. Martinez, "Simple rules yield complex food webs," *Nature*, vol. 409, pp. 180-183, 2000.
[37] N. Wirth, *Algorithms + Data Structures = Programs*: Prentice-Hall, 1976.
[38] L. Wolpert, *Curr Biol*, vol. 13, pp. 120, 2003.
[39] R. S. Zebulum, D. Gwaltney, G. Hornby, D. Keymeulen, J. Lohn, and A. Stoica, "NASA/DoD Conference on Evolvable Hardware," IEEE Press, 2004.