APPLICATION OF BIOMORPHIC PRINCIPLES IN DATA CENTER OPERATIONS

By

Michael L Williams

A MASTER OF ENGINEERING REPORT

Submitted to the College of Engineering at
Texas Tech University in
Partial Fulfillment of
The Requirements for the
Degree of

MASTER OF ENGINEERING

Approved

_____
Dr. A. Ertas (Chair)

_____
Dr. T. Maxwell (member)

_____
Dr. M. Tanik (member)

_____
Dr. J. Woldstad (COE Representative)

ACKNOWLEDGEMENTS

I could not have begun the task of pursing my postgraduate work without the complete support and understanding of my wife. While I may feel the pressure of balancing career with school, she has shouldered the greater responsibility for maintaining sanity amongst our household. I look forward to the completion of this undertaking and the return to my "real" job of being a husband and father.

The work for this report has been performed largely while employed over-seas on a high visibility program for Raytheon. I could not have completed the work without the assistance and understanding of the program management team. I would like to thank, in particular, Jane Chappel and Arlyn Chesley. While I hold them both responsible for my present role, they have been understanding and supportive of my need to complete this degree.

Within the cohort that makes up the Raytheon student body, Mike Smith and Daniel Moran have shared the experience of trying to complete a degree, over the phone and at strange hours, while working the EBorders programme. When else will we be surrounded by so much culture, with so little ability to appreciate and enjoy it? I'll never again lift a pint without remembering the time we spent, working together.

For the professors that support the Raytheon / Texas Tech program, I want to thank in particular Dr. Tanik for his thought provoking conversations, and Dr. Ertas for his unwavering support of our special needs (over seas assignment). I especially enjoyed Axiomatic Design, and was stretched to my limits with Dr. Maxwell's fundamentals classes. The program exceeded my expectations, and I believe that I have benefited from it personally, and with any luck, professionally as well. Many thanks for your hard work, dedication, and obvious joy in teaching.

TABLE OF CONTENTS

DISCLAIMER

The opinions expressed in this report are strictly those of the author and are not necessarily those of Raytheon, Texas Tech University, nor any U.S. Government agency.

ABSTRACT

System sizing is largely driven by an analysis of peak demand loads. While peak demand sizing provides a level of assurance that peak demands are met, it provides an inefficient model with regard to power and cooling needs. An idle machine may still consume up to 171 Watts of power [Horowitz, Noah (2005)]. Therefore, in a computing environment consisting of a plurality of server machines with fluctuating demand load, it would be desirable to implement a method to control the powered state of one or more machines according to an established method such that demand on power and cooling resources are minimized during off peak times while still meeting the performance and availability requirements of the system.

A Service Oriented Architecture (SOA) is a systems development methodology where functionality is grouped around business processes and packaged as interoperable services. SOA separates functions into distinct units, which are made accessible over a network in order that they can be combined and reused in the production of business applications [Bell, Michael (2008)]. A service-oriented architecture therefore provides an integration paradigm with focus on system behavior.

A Biomorphic system is a sub-discipline of engineering focused upon emulating the mechanics, sensor systems, computing structures and methodologies used by animals. In short, it is building systems inspired by the principles of biological systems [Wikipedia 2008]. In a Biomorphic system, there is no central controller explicitly orchestrating the action of the individual parts. And in the midst of all this seemingly chaotic interaction, still the organism forms, grows, lives, repairs itself, adapts, and survives [NASA 2004].

This report will examine the biomorphic paradigm and its application as a method of orchestration to achieve a service oriented system capable of meeting performance and availability requirements while minimizing power consumption in the datacenter.

LIST OF FIGURES

LIST OF TABLES

CHAPTER I
INTRODUCTION

Autonomous computing is a concept that has been explored in multiple facets. I became interested in autonomous computing while experimenting with a teaching tool released by IBM in the 1990s. Robocode provided a framework where a programmer could provide instruction to a "robot" tank and then cause the tank to compete against other robots. The software provided a framework for competition. Each robot included radar, gun, and a limited amount of energy that could be used. When the robot's energy was depleted, it could no longer move, and became a "sitting duck".

What fascinated me about the game was the infinite variety of strategies that could be employed and then observed during a game scenario. Programmers from around the world could submit robots online, and you could pit your strategies against theirs. Once your robot entered the arena, it was beyond human intervention. It had to observe, move, fire, retreat, conserve or waste energy based only on the strategy previously coded.

Last year, I had the opportunity to work on a system proposal, which was based on Service Oriented Architecture. SOA over the last few years has gained a great deal of attention and money, but like most new technologies, serious debate existed over its correct use and implementation. I wanted to put something together that would be true to the concept of the SOA. As I began to explore the COTS SOA marketplace, I found that the tools that existed did not go quite as far as I wanted. The architecture we proposed required that the Service Bus and Registry be fully integrated. We envisioned a system where the definition of the service existed in only one place; a single system of record where all aspects of operational services were captured and by changing only the record, the system functionality would change correspondingly.

Working with some very talented engineers at Raytheon, we were able to build and demonstrate a working system that embodied the principles of the architecture that had been put forward. The framework allowed us to define service level agreements (SLA) for system services. The system would measure the performance of the aggregation of any service type. If a service aggregation was found to be

performing better than required by the SLA, then a service instance of that aggregation was disabled. If a service instance was found to be underperforming its SLA, then a new instance of that service was added to the operational pool. Furthermore, when the pool of available services neared depletion, additional servers were started from a powered down state and automatically provisioned into the operational system. All service instances on the server became available for use within the pool. These system activities were carried out without the intervention of an operator. All capability was initiated by the system based only on defined SLA for the environment. Change the SLA, and the system behavior changed.

It turns out that starting a machine from a powered down state is about a 3.5 to 5 minute procedure. There are many variables that go into the required time, but the important factor is that the operation is not without cost. It would be very easy to have the system enter into a thrashing state where machines are booted and then powered off, only to be booted again.

You can examine the operation of a thermostat in a household heating and cooling system to easily understand the problem. In many homes, the thermostat must be set to "cool" or "heat" the home. It is a much more simple operation to cause a thermostat to start a system when a threshold has been exceeded (positive or negative). However, it doesn't take much imagination to see that system that can operate with cooling and heating simultaneously would be in danger of cooling only to then cause heating only to then cause cooling. Not a model of efficient operation.

The problem that I wish to solve then is similar to that of a thermostat in an environment where both cooling and heating can be applied. Instead of trying to track down the latest in thermostat technology, I became interested in biomorphic systems, or systems whose behavior is based at least in part on a biological system. Interesting studies exist on "flocking" behavior in birds, and other macro level group interaction behaviors, but what I found most interesting was the life cycles of cells within the human body.

Human cell development is a function of DNA. In the genetic process, a cell contains the instructions to become any cell that makes up the body. I found this interesting because it is how we

approached the server platform in the system we had developed. Any "node" hosted the full compliment of services available in the system. The performance and activities demanded by the current load on the system actually determined what the purpose of the node would be i.e. which services became active for that node. Having made this observation, I decided to look at the process of cell creation and destruction to see if any lessons could be derived that would help me solve the problem of when exactly to power down an operational node.

It turns out that cells have a programmed life span. A cell ages and dies, to be reclaimed by the body. While it seems very simple, the concept has merit for the solution I was seeking. The system could be constrained such that when a node was brought online, it would only operate for a specified period of time, and then cease operation upon the completion of its current tasking. As I considered the benefits of the solution it occurred to me that by powering up and down machines, it would also have the side benefit of clearing out any computing "cruft" that had accumulated i.e. memory leaks, etc. The resulting system would have the appearance of a factory where "production" was driven by demand measured against the system. Nodes could be created as needed (much like the production of red blood cells by the body). However, each node created would have a natural life span, which would cause it to cease operation and return to the pool of machines that were powered down. The number of operational machines would correspond to current demand on the system. Power consumption would be reduced when machines aged off the system. New nodes can be introduced at any point without interrupting the operational state. Old machines can be removed without interrupting the operational state.

I enjoy the power of simple concepts realized. I look forward to the full implementation of the system described and the operational observation to follow.

CHAPTER II
BACKGROUND

2.1 The Hidden Costs of Datacenter Computing

The low cost and widespread use of commodity computing has masked the related inefficiencies of current computing models. "In every era, the winning companies are those that waste what is abundant – as signaled by precipitously declining prices – in order to save what is scarce." [Gilder, George 2008] While past computing methodologies emphasized efficiency, today's datacenters are built on application stovepipe models.

2.1.1 Abundance

Companies are dumping billions of dollars of capital into constructing utility-class computing centers. [Carr, Nicholas 2008] It is estimated that the five leading search companies together have some 2 million servers; each shedding 300 watts of heat annually, a total of 600 megawatts. These are linked to hard drives that dissipate perhaps another gigawatt. Fifty percent again as much power is required to cool this searing heat, for a total of 2.4 gigawatts. With a third of the incoming power already lost to the grid's inefficiencies, and half of what's left lost to power supplies, transformers, and converters, the total of electricity consumed by major search engines in 2006 approaches 5 gigawatts. [Carr, Nicholas 2008]

Moors' law has driven the cost of computing components close to zero. However, the ancillary costs associated with owning and operating vast commodity resources cannot be ignored. A Gartner study estimates that in five years electricity will account for 20% to 40% of companies' entire IT budgets. [Gartner 2006]

2.1.2 Scarcity

For the client-server computing model, peak load sizing requires that many more computers be allocated to a computing task than are required. Servers are dedicated to running individual applications, and they're housed in data centers constructed to serve individual companies. The fragmentation of

computing has led to low levels of capacity utilization – 10% to 30%. The majority of computing capacity—and the electricity required to keep it running—is squandered. [Carr, Nicholas 2008]

2.2 Service Oriented Systems

2.2.1 Grid Computing and SOA Infrastructure

Grid computing has changed the way that systems are structured and allocated. Formerly, resources were allocated to systems based on peak performance or other static allocation schemes. Current grid architectures seek to dynamically shift processing to provisioned resources based on policy or system performance metrics measured in near real time. Grid tends to look toward the underlying system (bottom-up) and create a dynamic environment where execution can occur.

Service Oriented Architecture provides an integration focused architecture approach. The fundamental building block is the service. Orchestrations of services create new solutions. Solutions may execute across many environments, abstracting away the underlying environment upon which the service is dependent.

This study will include the use of a managed Service Oriented Architecture Framework (Framework). It is deployed in a Grid environment utilizing a single system of record to provide a dynamically modifiable run-time configuration without regard to processing platform, network location, or resource. The framework provides the following architectural features & benefits:

- Resources (Hosts and/or Services) can be added to an operational system on demand.
- Resources (Hosts and/or Services) can be removed from an operation system on demand.
- Resources (Hosts and/or Services) provisioning requires zero system downtime.
- Changes to Resource allocations through Resource Provisioning activities are immediately incorporated into the operational systems' load balancing and resource sharing features.
- Changes to Resource allocations require zero system downtime.

2.3 Biomorphic Computing

2.3.1 Biomorphic Orchestration

Biomorphic software is simply algorithmic design concepts distilled from biological systems, or processes. It is biologically inspired computing. [NASA (2004)] Within a biomorphic system, there is no orchestrating manager. Control is decentralized and emerges from the interaction between individual elements. It is the looseness of the distribution, decentralized behavior, pattern formation, and intelligence of the biological models that makes biomorphic architecture applicable to many computing problems. [NASA (2004)]

2.3.2 Short History of Biomorphic Computing

Biomorphic computing is a concept that has been around since at least 1948, when Alan Turing published a paper called "Intelligent Machinery". He invented a kind of neural network called a "B-type unorganized machine". In a 1950 paper, "Computing Machinery and Intelligence," Turing suggested that natural selection and evolution could be mechanisms in the effort to construct intelligent machines. [Copeland & Proudfoot 1999][NASA 2004]

In "Hitchhiker's Guide to Biomorphic Systems", Kenneth N. Lodding, Nasa marks the following important events:

- "Adaptation in Natural and Artificial Systems", published 1975 by John Holland.
- Three rules for flocking behavior in birds, 1986, Craig Reynolds
- Ant colony optimization to solve discrete optimization problems, 1992, Marco Dorigo
- Particle swarm optimization, 1995, Kennedy and Eberhart
- Swarm Intelligence: From Natural to Artificial Systems, 1999, Bonabeau, Dorigo, and Theraulaz

Research into biomorphic software styles and the span of applications continues to grow. The field of applicable systems includes distributed computing, sensor network systems, algorithms, etc. Biology has provided resilient systems, highly fault tolerant, and optimized for minimal resource consumption.

CHAPTER III
Architecture

3.1 Context

Biomorphic systems are a sub-discipline of engineering focused upon emulating the mechanics, sensor systems, computing structures and methodologies used by animals. In short, it is building systems inspired by the principles of biological systems. This paper will present an analysis of selected biological behaviors (systems) with regard to their suitability to provide a behavioral logic applicable to data center operations.

3.1.1 Goals

While peak demand sizing provides a level of assurance that peak demands are met, it provides an inefficient model with regard to power and cooling needs. An idle machine may still consume <<insert %>> power and cooling. Therefore, in a computing environment consisting of a plurality of server machines with fluctuating demand load, it would be desirable to implement a method to control the powered state of one or more server machines according to an established method such that demand on power and cooling resources are minimized during off peak times.

Table 1 – Architecture Goals

Goal
Reduce power consumed by data center.
Reduce HVAC consumed by data center.
Changes in allocation (add / remove) should seek to provide service at level appropriate to meet system demand.
Avoid over performance (too many allocated resources).
Avoid under-performance (too few allocated resources).
Seek to achieve optimal performance scenario (resource production equal to resource demand).
Determine best time (define what is meant by “best time”) to allocate server resource (start-up).
Determine best time (define what is meant by “best time”) to de-allocate server resource (shut-down).

3.1.2 Needs Hierarchy

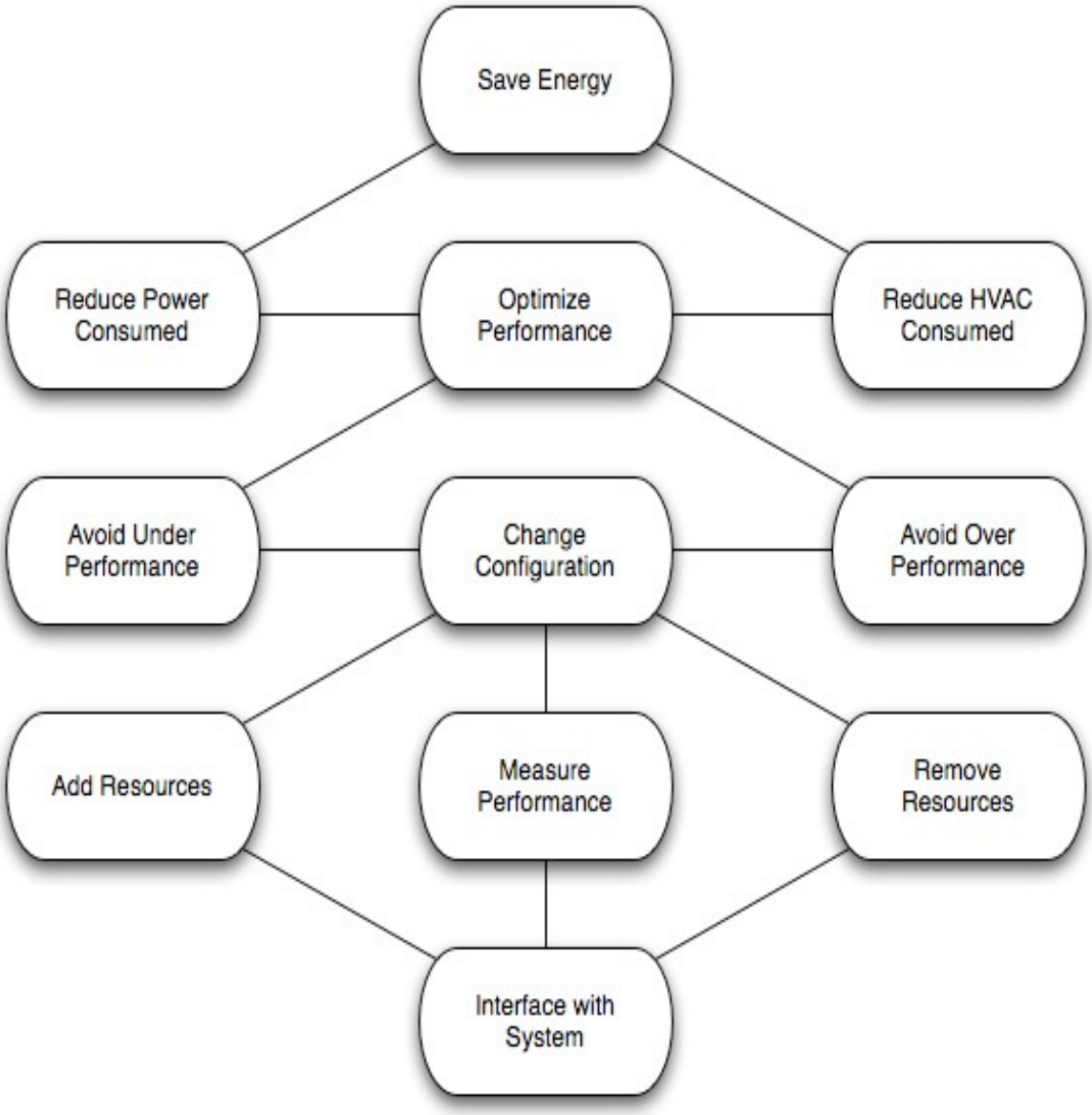

Figure 1 - Needs Hierarchy

3.1.3 Stakeholders & Viewpoints

Research interested parties include those who have a direct interest in the operational costs associated with large-scale data centers including:

Technology Owner / Technology Providers - Technology Owners, including government agencies and private firms, invest << how much>> annually in the operational aspects of large-scale data centers.

Technology Providers, such as consulting and contracting agencies in commercial and government business fields, provide fielded systems. For the interest of the study, Technology Owners and Providers will be considered to the same organization in that they both benefit from reduced operational costs.

Technology Consumers - Technology consumers including the general public benefit from the economic impact associated with a reduction in infrastructure costs for services they consume. Indirect benefit is possible if the reduction reduces market cost (scarcity) of the constrained resource (power).

3.1.4 Relationship Matrix

The relationship matrix reveals that the primary beneficiary of the study is the Technology Provider. The Technology Consumer indirectly benefits through a lower cost system, but is primarily interested in receiving the contracted service level (i.e. avoid under performing system).

Table 2 - Relationship Matrix

Goal	Technology Provider	Technology Consumer
Save Energy	X	
Reduce Power Consumed	X	
Reduce HVAC Consumed	X	
Optimize Performance	X	
Avoid Under Performance	X	X
Avoid Over Performance	X	
Measure Performance	X	X
Change Configuration	X	
Add Resources	X	
Remove Resources	X	
Interface with System	X	X

3.1.5 Quality Attributes

Quality attributes represent the viewpoints of the stakeholders. Quality is defined as:

- Reliability – In general, reliability (systemic def.) is the ability of a person or system to perform and maintain its functions in routine circumstances, as well as hostile or unexpected circumstances.

- Performance – There are three major classes of performance metrics: response time, throughput, and availability.
    - Response Time – In technology, response time is the time a system or functional unit takes to react to a given input.
    - Throughput – Throughput is the average rate of successful message delivery over a communication channel. The data may be delivered over a physical or logical link, over a wireless channel, or passing through a certain network node, such as data passed between two specific computers.
    - Availability – The degree to which a system, subsystem, or equipment is operable and in a committable state at the start of a mission, when the mission is called for at an unknown, i.e., a random, time. Simply put, availability is the proportion of time a system is in a functioning condition.
- Scalability – In telecommunications and software engineering, scalability is a desirable property of a system, a network, or a process, which indicates its ability to e3ither handle growing amounts of work in a graceful manner, or to be readily enlarged.
- Efficiency – Energy conservation is the practice of decreasing the quantity of energy used. It may be achieved through efficient energy use, in which case energy use is decreased while achieving a similar outcome, or by reduced consumption of energy services.

3.1.6 Quality Attribute Analysis

Analysis of the quality attributes was examined by applying Brosseau's technique. While the focus of the study is to produce a more efficient data center, we can see from the analysis, that the gain in efficiency must not come at the expense of reliability, availability, scalability, and response time. Throughput is not a factor in the system.

Table 3 - Quality Attribute Table

Goal	Response Time	Throughput	Availability	Scalability	Efficiency	Rank
Reliability	<	<	<	<	<	5
Response Time		<	^	^	<	2
Throughput			^	^	^	0
Availability				<	<	4
Scalability					<	3
Efficiency						1

3.1.7 Context

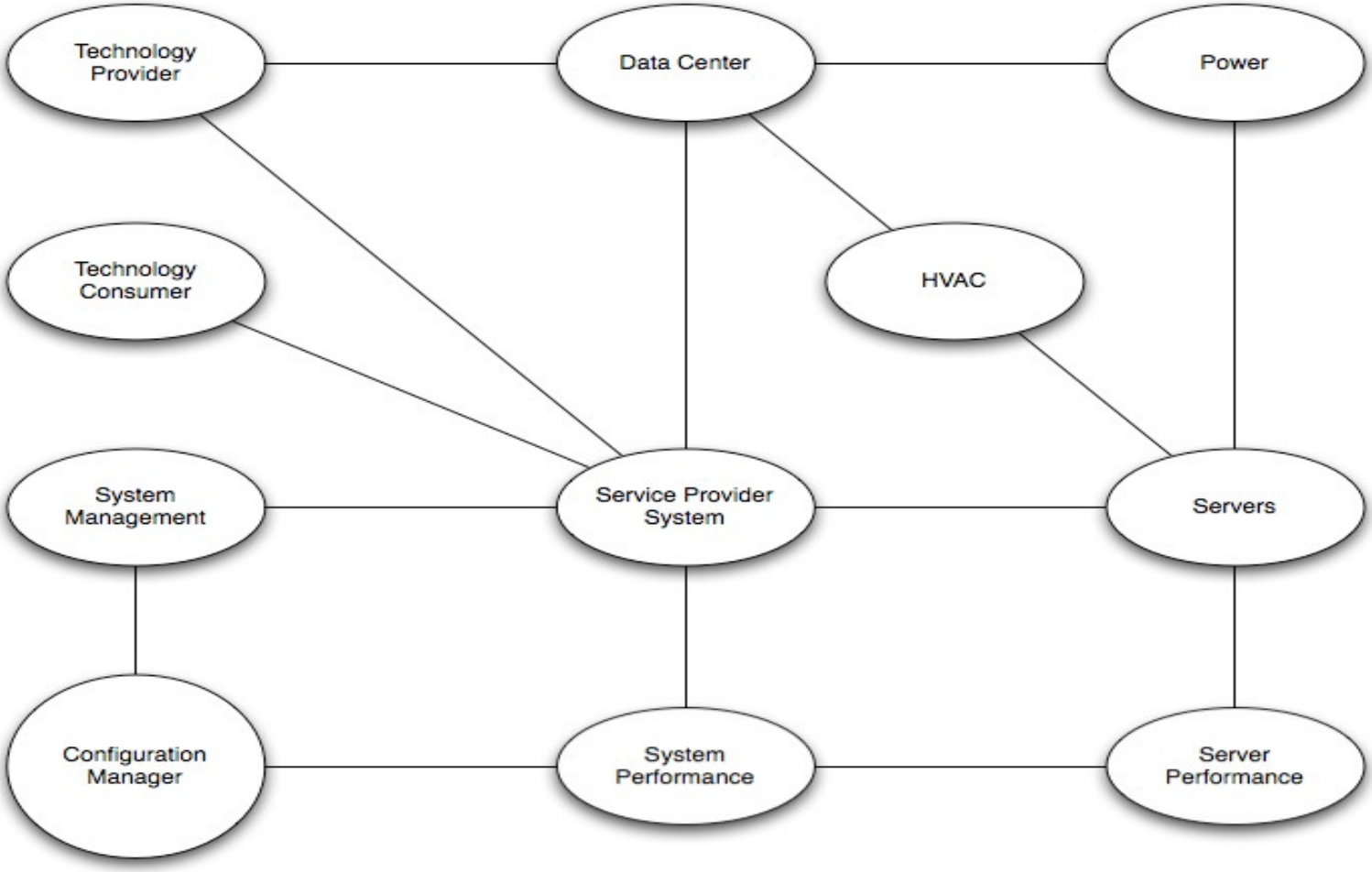

Figure 2 - System Context

3.2 Operational View

3.2.1 Concept of Operations (CONOPS)

System Nodes provide processing capability (node). The life cycle of a node can be time dependent, event dependent, or both. Time dependent nodes are expected to execute for a specified period of time (configurable per node) and then cease execution when the specified time period has elapsed. Event dependent nodes are expected to execute until they have completed a specified number of operations (configurable per node) and then cease execution when the specified number of operations have completed. TAE nodes (time and event) are expected to execute for a specified period of time or until they have performed a specified number of operations, whichever comes first, and then cease to operate.

The system monitor interfaces with the System and provides statistical information about the performance of the system over a defined period of time. The system monitor may also issue event notifications when specified performance scenarios are detected. Events include but are not limited to System Over Perform, System Under Perform, and System Optimal Perform.

The Biomorphic Control Mechanism (BCM) receives information from the system monitor, and acts according to its internal logic to determine the optimal time to cause new Nodes to be created within the System.

A system is in operation with well-defined and measurable service levels. A monitor measures the performance of the system as compared to the service levels at a given time interval. If the system performance is calculated to be over performing, that is providing more processing capability than is required, then system operational statistics are collected including but not limited to, number of operational nodes, and current system demand. If the system performance is calculated to be under-performing, that is providing less processing capability than is required, then the biomorphic control mechanism (BCM) will signal the system to add additional processing capability. If the system is

determined to be operating in an optimal fashion, that is system performance is within the defined Service Level Agreement (SLA) bounds, then no action is taken by the BCM.

3.2.2 Key Scenarios

The key scenarios identified in Table 4 represent key points in the system that can be used to drive system behavior.

Table 4 - Key Scenarios

Scenario	Description
System Optimal	System performance has been measured and reported to be performing in an optimal state.
System Under Performing	System performance has been measured and reported to be under performing according to its service level agreement. Additional resources are added to the system, and the system returns to an optimal performance state.
System Over Performing	System performance has been measured and reported to be over performing. While statistics are collected, no direct action is taken to reduce the number of operational nodes in the system.
Add Resource	Add resource is the act of causing one or more nodes to be added to the operational system.
Remove Resource	Termination of a node in the system is based on a metric that is specified during the creation of the node. A system based on life-cycle would, for instance, have finite metrics such as processing life (time span) or processing actions (performance metric) which, when expired, would cause the node to cease operation and removed as an operational node of the system.

3.2.3 Use Cases

3.2.3.1 Directed Method for the addition and removal of nodes in an operational system.

The directed method works under the assumption that the BCM will take action in any system performance scenario whereby the system is not operating in an optimal fashion. If the system is under performing, then nodes will be added. If the system is over performing, then nodes will be removed. The number of nodes to be added or removed is configurable.

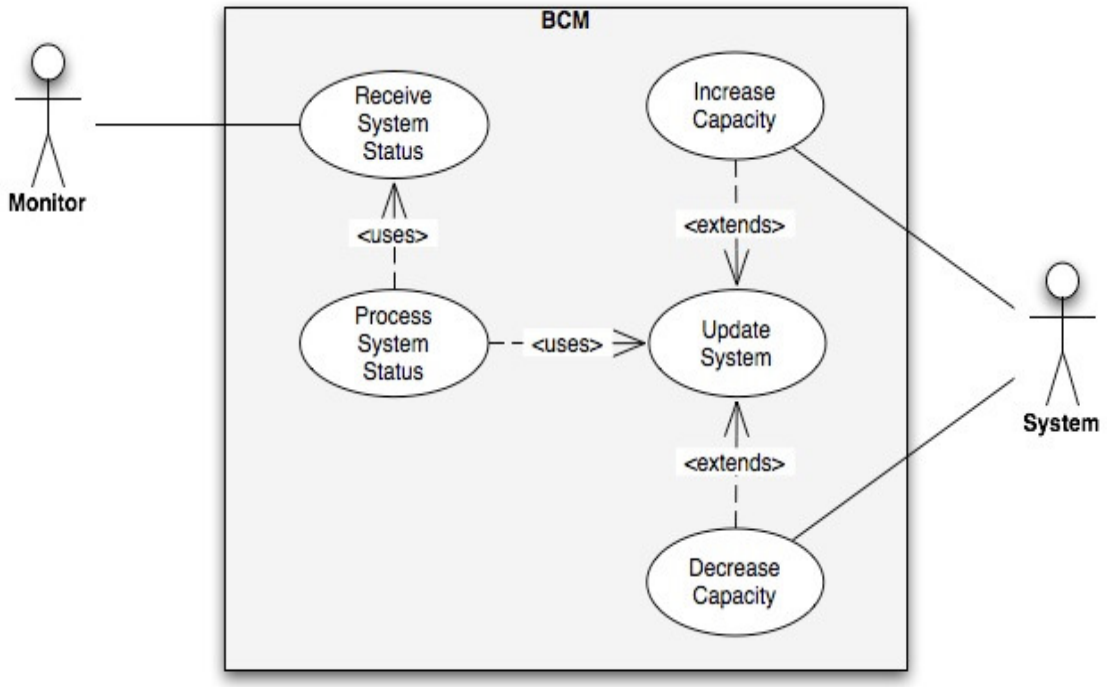

Figure 3 - BCM Use Case

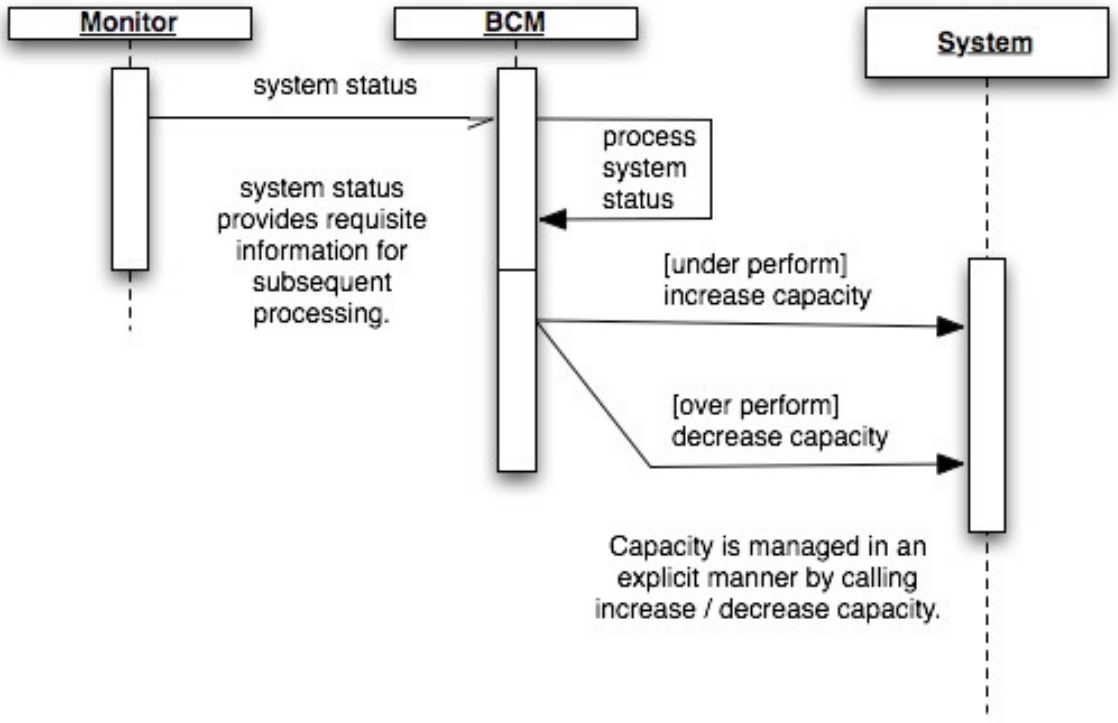

Figure 4 - BCM Sequence Diagram

3.2.3.2 Life-cycle method for the addition and removal of nodes in an operational system

The life-cycle method works under the assumption that the BCM will take action only in the event that the system is under performing. In that scenario, the BCM will act by causing one or more nodes to be added to the operational system. Each node will be configured by the BCM to the time or operation dependent, and when the specified threshold is exceeded, then the node will cease to operate without the direct intervention of the system, or the BCM.

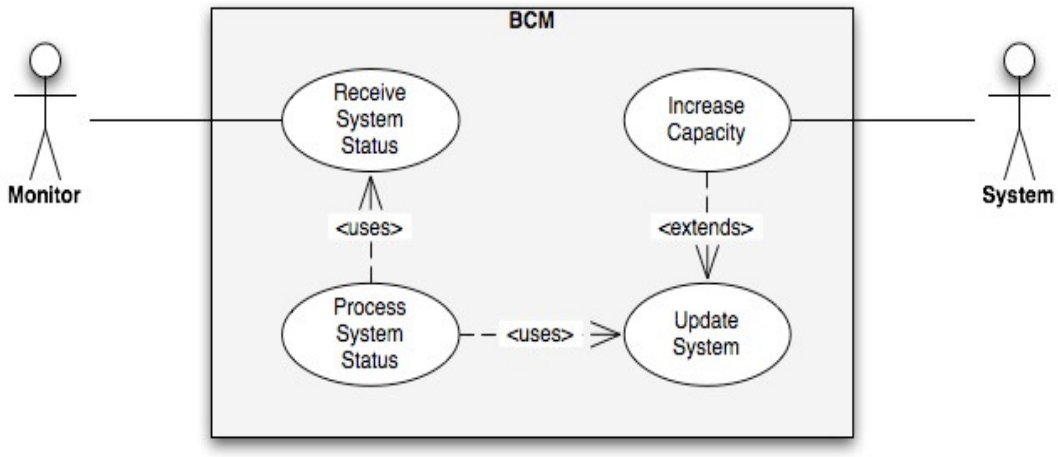

Figure 5 - Life-Cycle Use Case

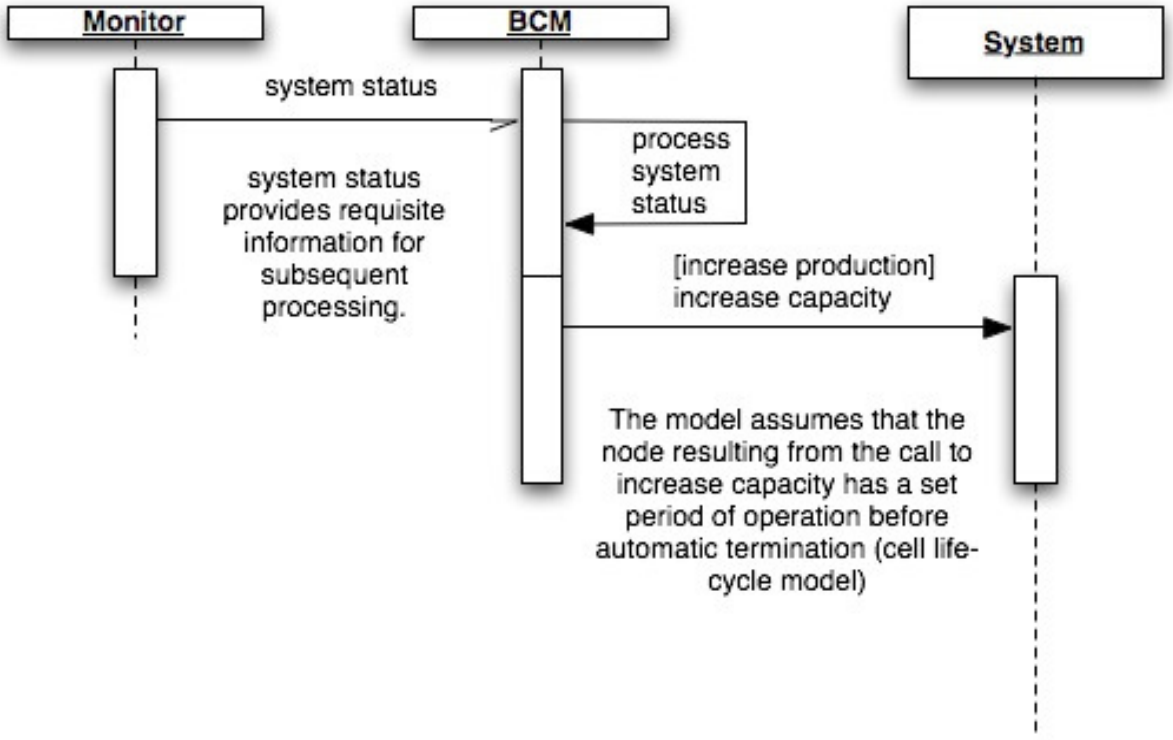

Figure 6 - Life-Cycle Sequence Diagram

3.2.3.3 Use Case

Table 5 - Use Case Example

Scenario	Description
Description	Process System Status
Actors	Monitor, BCM, System
Preconditions	System status available for processing, status event received by BCM
Basic Flow	Analyze the input (system status), determine if new nodes are needed, determine how many new nodes are needed, configure termination condition for new nodes, invoke node provision service on System.
Alternative Flow	Analyze the input (system status), determine if new nodes are needed, terminate without creating new nodes (no new nodes needed).
Exception Flows	Log error, wait for new status event to process.
Post Conditions	(n) number of new nodes added to the system.

3.3 Logical View

3.3.1 Functional Decomposition

The functional decomposition relates four aspects of the system: Communication, Control, Node, and Management.

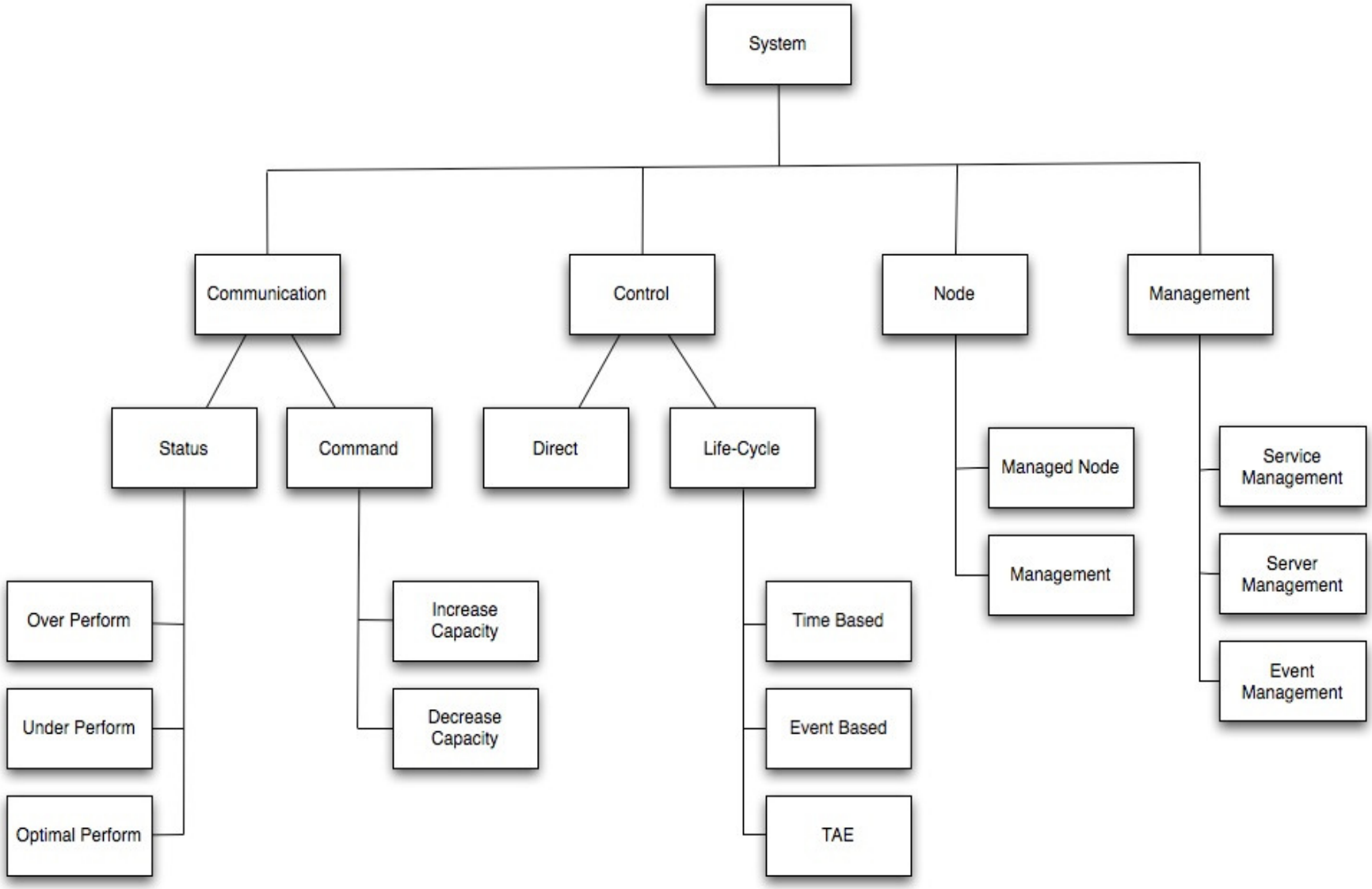

Figure 7 - Functional Decomposition

3.3.1.1 Communication

System communication takes the form of either status or command messages. Status messages relate the operational status of a system component, and decompose into over perform, under perform, and optimal perform messages. The definition of each is relative to the component's service level agreement, and can vary from component to component. Command messages result in service calls within the system, and convey instruction from a management node to a managed node. Two commands are defined for the system: increase capacity and decrease capacity. The command messages are representative of the required action. The actual semantics of the command will vary from types of managed nodes depending on the API employed.

3.3.1.2 Control

There are two aspects of control in the system: direct and life-cycle. Direct control implies that the management node is taking direct action to change the system configuration, including increase, and decreasing capacity (see command communications). Life-cycle control only makes use of capacity increase commands, and depends on the managed node to handle its own life cycle once created.

3.3.1.3 Management

There are three management capabilities in the system: service, server, and event. The scope of control for service management is bounded to web services. The scope of control for server management is physical hardware (managed nodes / service hosts) within the system. The scope of control for event management is status communications as events and how they are directed to either server or service management nodes.

3.3.1.4 Nodes

The system contains two types of nodes: managed and management. Managed nodes respond to communications from management nodes and carry out specific / configured behaviors. Management nodes direct the behavior of managed nodes via control commands.

3.3.2 Domain Model (logical view)

The domain model shows how the functional decomposition of the system relates to objects in the system. The provided example relates the Biomorphic Control Manager with a managed node in the solution.

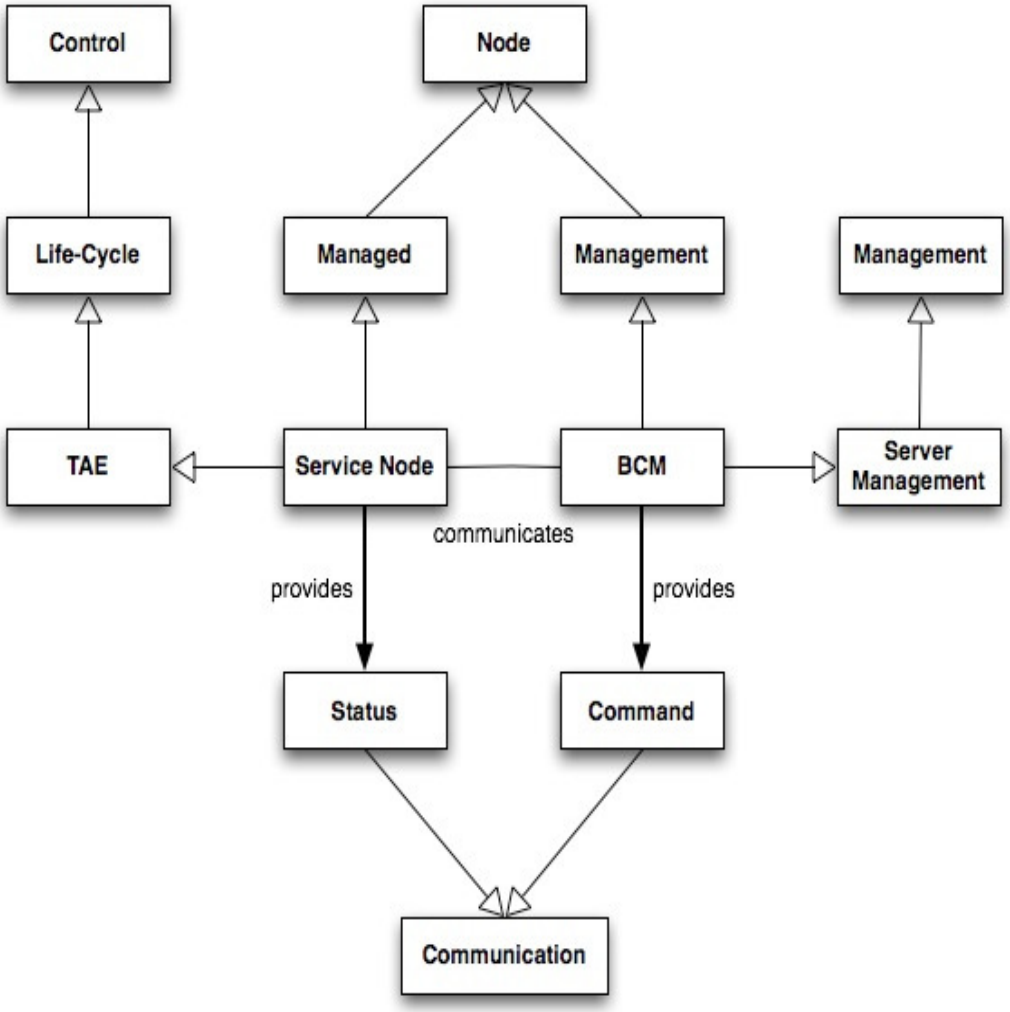

Figure 8 - Domain Model

From the diagram you can see two types of nodes (managed & management), two types of communication (status & command), one control type (TAE), and finally, one type of management (server).

3.3.3 Component Model (with data flow)

The component model shows the relationship between the system components and the data that passes between them. The Biomorphic Control Manager (BCM) is represented as the Server Management component in the component model. It receives system status messages as hardware events, and provides control messages resulting in new service nodes (hosted on blades) provisioned for the system.

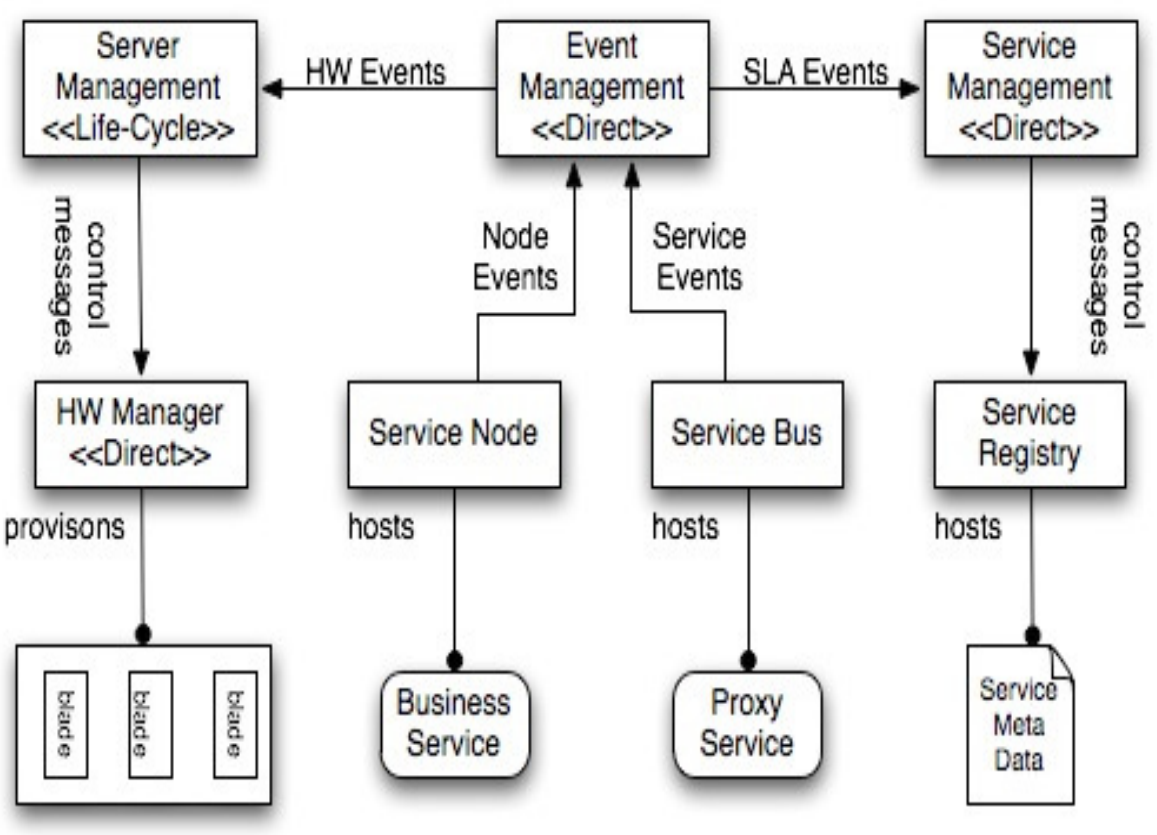

Figure 9 - Component Model

3.3.4 Interactions

Table 6 - BCM as a Server Manager

Interaction	Description
Receive System Status	1. Listen for status message. 2. [status message received] process system status.
Process System Status	1. Is system under performing? 2. [yes] determine number of nodes needed. 3. [yes] determine node configurations. 4. [yes] create new nodes. 5. [yes] invoke node provision service for System 6. [yes / no] end.
Invoke node provision service	1. Send provision node message to System.ProvisionNodeService 2. End.

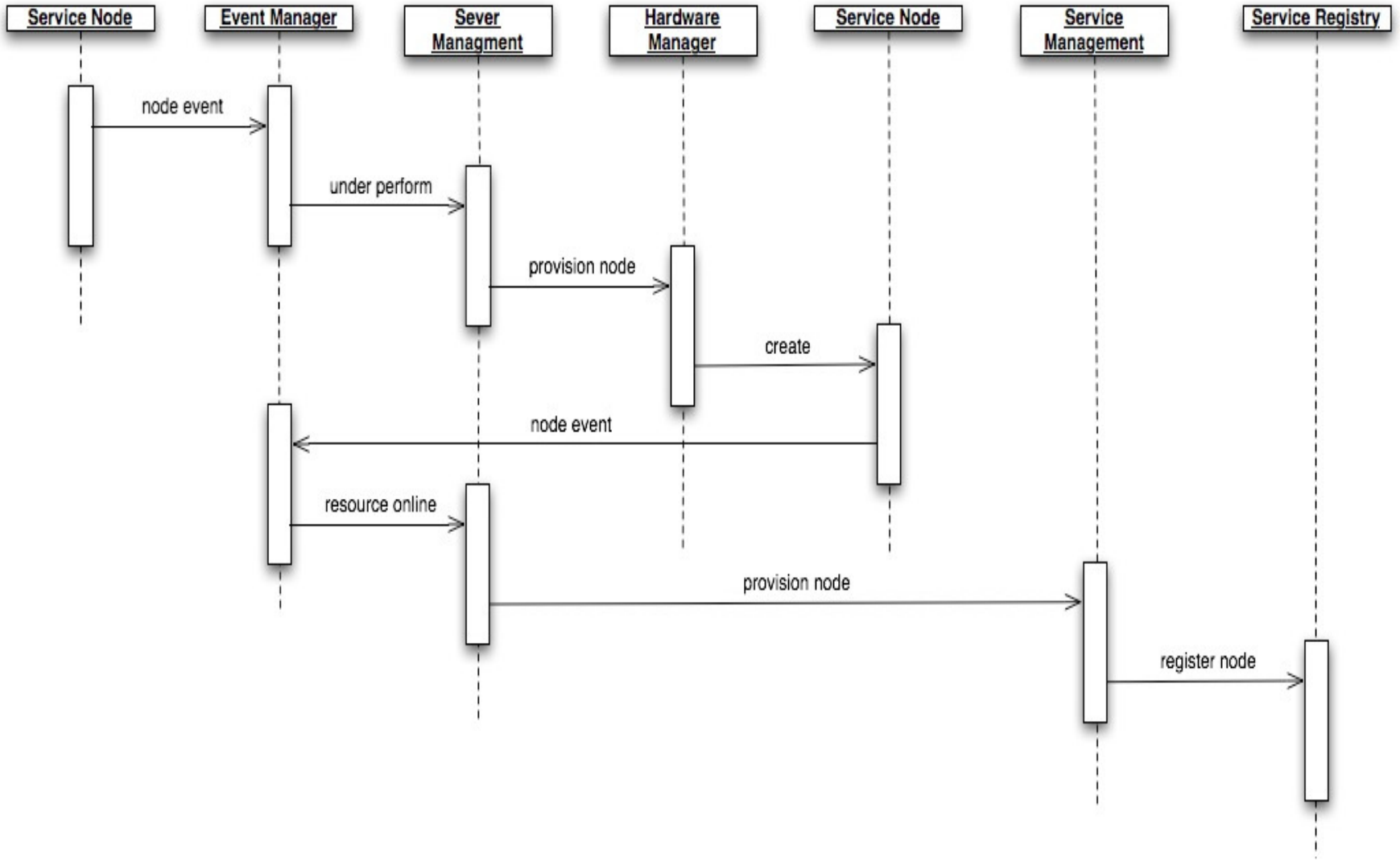

Figure 10 - Sequence Diagram (Provision Node)

3.3.5 Trades

Trades in the physical component space have been identified below. For the sake of completeness, some trades have been identified, but are considered outside the scope of the focus of the architecture. While they are important system aspects (Storage / Network infrastructure), they are not fully described within this context. They are assumed to be provided services, accessible via standards based means, and to be fully described in subsequent documentation.

Table 7 - Physical Trade Space

Trade Space	Description
Node Architectures	1. Blade Architecture 2. 'Pizza-Box' Architecture 3. Specialized Hardware
Hardware Abstraction Component	1. Lavanta Intrepid 2. Altiris 3. VM-Ware (ESX)
Storage Architecture	N/A
Network Architecture	N/A

Because the area of research directly relating to this architecture involves trades between direct control and biomorphic control for server (managed node) deployments, the subsequent paper produced from this exercise will form the basis of that trade. Biomorphic control mechanisms will be examined with the intent of identifying candidates for managed node provisioning such that data center costs are reduced through the reduction of power and cooling consumption. A key tenant of the concept is the ability to match computational power with near real time demand loads. Assuming non-overlapping peak distribution of computational services, the same physical resource (managed node) can be leveraged such that the total numbers of required assets are reduced (shared load). While these concepts are not new, the ability to measure service and node performance in the context of a Service Oriented Architecture (SOA) provides a new opportunity through intelligent management to not only reduce the number of nodes needed during any average slice of time, but also to reduce the number of nodes held in an operational (powered on) state.

3.3.5.1 Trade Criteria

The criteria for the trade are in alignment with the goals of the architecture (repeated for readability).

- Reduce power consumed by data center.
- Reduce HVAC consumed by data center.
- Changes in allocation (add / remove) should seek to provide service at level appropriate to meet system demand.
- Avoid over performance (too many allocated resources).
- Avoid under-performance (too few allocated resources).
- Seek to achieve optimal performance scenario (resource production equal to resource demand).
- Determine best time to allocate server resource (start-up).
- Determine best time to de-allocate server resource (shut-down).

The selected algorithm / model should be simple in concept, and fit within the component model identified within the physical section of this architecture.

3.3.5.2 Biomorphic System Candidates

While a thorough analysis of different biomorphic systems would be beneficial, I have chosen to focus primarily on the cellular life cycle. Other candidates include the endocrine system, and insect studies. With regard to the biomorphic systems not chosen, further study could be applied with the criteria identified in the preceding paragraph. A useful approach to comparing the relative merits of the candidates would be to apply an Axiomatic analysis.

3.3.5.3 Cellular Life Cycle

To understand the nature of the application of the process of the cell life cycle as the biomorphic control for the system, it is necessary to review the cell life cycle process. The cellular state in an adult

organism can be viewed as a steady state system. The system is so balanced, that the cell neither grows, shrinks, nor changes its function [W.H. Freemand and Company 2000]. However, cellular dynamics can best be understood by studying the full life cycle.

A new cell arises due to cellular division or fusion. Either event sets off a cell-replication program that is encoded in the DNA and executed by proteins. This program includes a period of cell growth, followed by cell division. An adult organism replaces worn out cells or makes more cells in response to need. Muscle cells grow in response to exercise or damage, and red blood cells are produced when a person ascends to a higher altitude or needs additional capacity to capture oxygen.

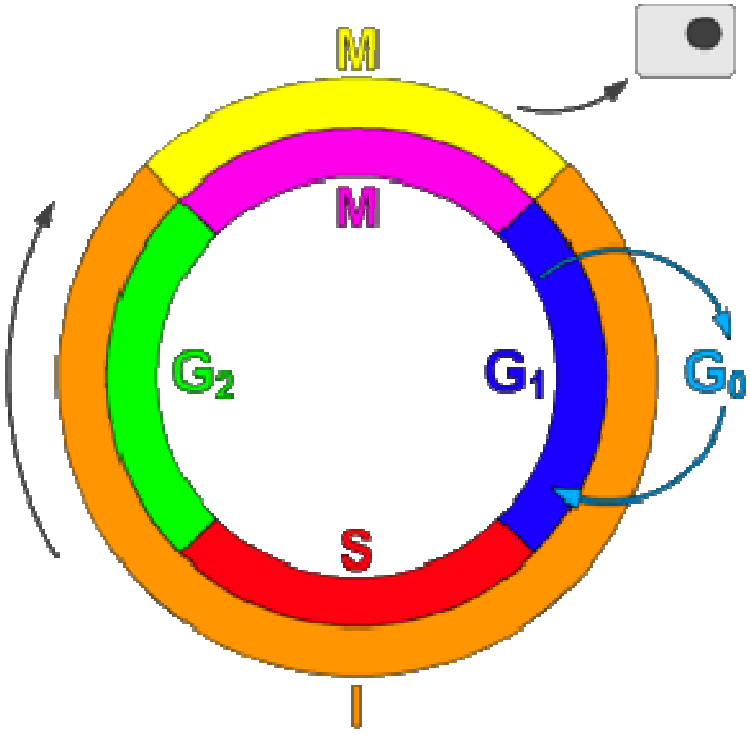

Figure 11 - Schematic of the cell cycle

Most cells live according to an internal clock. They proceed through a sequence of phases, called the cell cycle, during which DNA is duplicated during the synthesis (S) phase and the copies are distributed to opposite ends of the cell during miotic (M) phase [W.H. Freeman and Company 2000]. Progress is controlled at key checkpoints, which monitor the status of a cell. When certain conditions are

met, the cell proceeds to the next checkpoint. The cycle begins anew after the cell divides into two daughter cells, each containing an identical copy of the parental cell's genetic material.

Unchecked cell growth and multiplication produce a mass of cells. Programmed cell death plays the very important role of population control by balancing cell growth and multiplication [W.H. Freeman and Company 2000]. Cell death eliminates unnecessary cells.

3.3.5.4 Application of Cell Life Cycle as a Biomorphic System Control

Key characteristics of the cell life cycle can be used for system control. Of particular interest are the steady state nature of an adult organism, cellular production in response to need, programmed cell death, and multiplication control. The system modeled, through the use of Service Level Agreements has established criteria to determine the performance state of the system [see Table 4 - Key Scenarios]. An analogy of the system scenarios is presented with respect to cellular life cycle approach.

Table 8 - Cellular Life Cycle Analogy

Scenario	Description	Cellular Life Cycle Analogy
System Optimal	System performance has been measured and reported to be performing in an optimal state.	Balanced System.
System Under Performing	System performance has been measured and reported to be under performing according to its service level agreement. Additional resources are added to the system, and the system returns to an optimal performance state.	System expands according to need similar to muscle or red-blood cell production.
System Over Performing	System performance has been measured and reported to be over performing. While statistics are collected, no direct action is taken to reduce the number of operational nodes in the system.	Unchecked cell growth. Similar to a cell mass, or tumor.
Add Resource	Add resource is the act of causing one or more nodes to be added to the operational system.	Cellular growth and multiplication

Scenario	Description	Cellular Life Cycle Analogy
Remove Resource	Termination of a node in the system is based on a metric that is specified during the creation of the node. A system based on life-cycle would, for instance, have finite metrics such as processing life (time span) or processing actions (performance metric) which, when expired, would cause the node to cease operation and removed as an operational node of the system.	Cellular death.

3.3.6 Quality Attribute Review

The Quality Attribute Analysis revealed that although the focus of this study is to improve efficiency through the application of a biomorphic control the system could not afford to sacrifice other quality attributes (Reliability, Availability, Scalability, and Response Time). From our logical system, you can see the mechanisms in place for the measurement of the quality attributes. The system decomposition fully supports the independent measurement and control needed. Further, the trades outlined will afford opportunity for considerable study against a diverse set of biological systems. The pluggable structure of the resulting system will enable the application of BCM, without adverse affect on the high priority quality attributes.

3.4 Physical View

3.4.1 Domain Model (physical view)

The domain object model (DOM) further refines the component model (logical view). In the DOM, products have been assigned to components and the relationships further refined. From this model, we can see the introduction of new components and how they fit within the refined model of the system. The BCM is represented as the SVR Manager, and remains the focal point of this study.

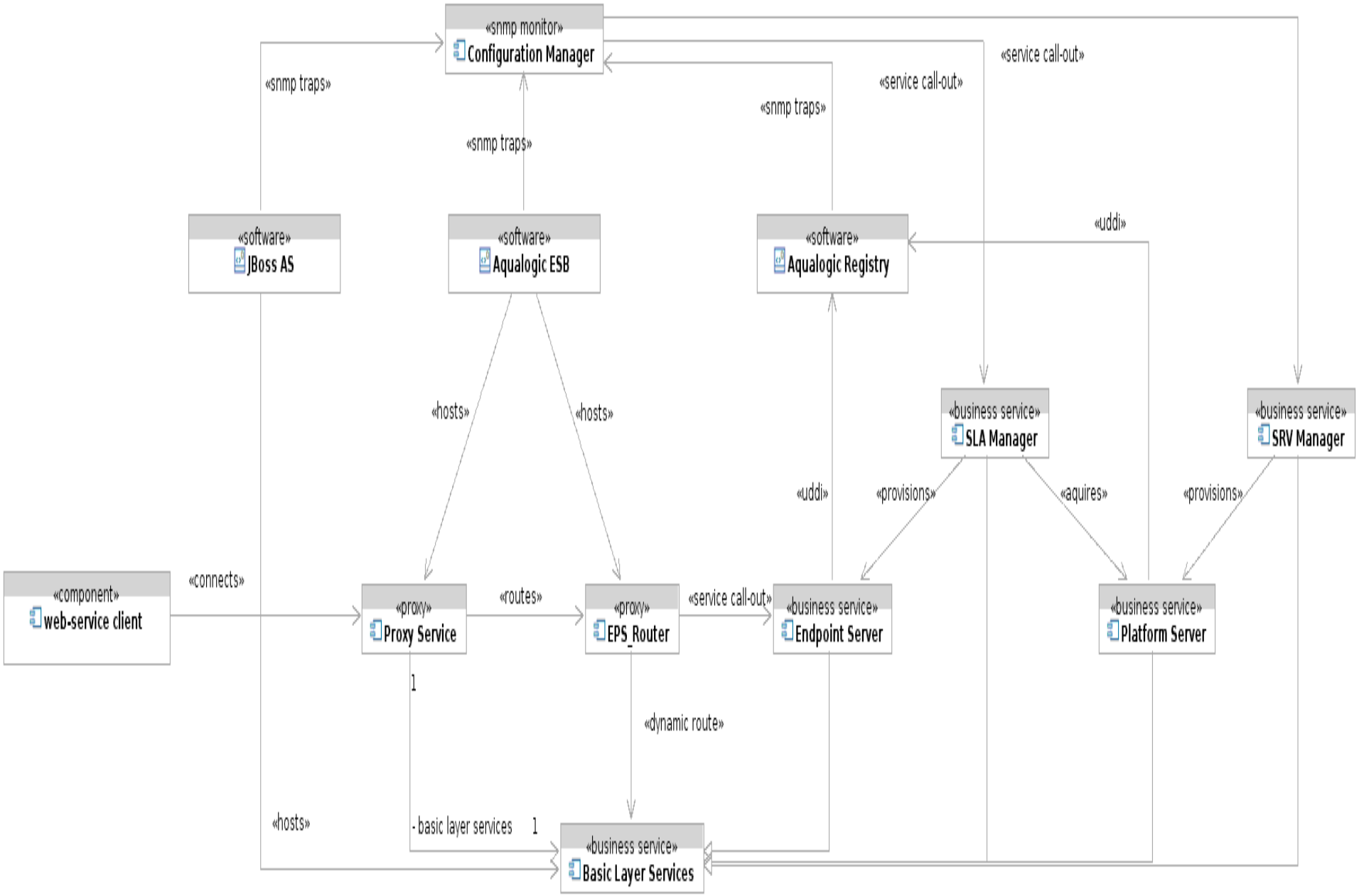

Figure 12 - Domain Model (physical view)

3.4.2 Deployment Model

Component deployment incorporates both clusters and grids. Service Oriented Architecture (SOA) off the shelf (OTS) components are deployed onto Application Server Clusters supporting scalability, high availability, and shared state. SOA OTS components include: Service Bus, and Service Registry. Management components (Policy Manager, Pool Manager, and Resource Server) are deployed as services and reside on Application Servers configured as a grid (independent stateless nodes).

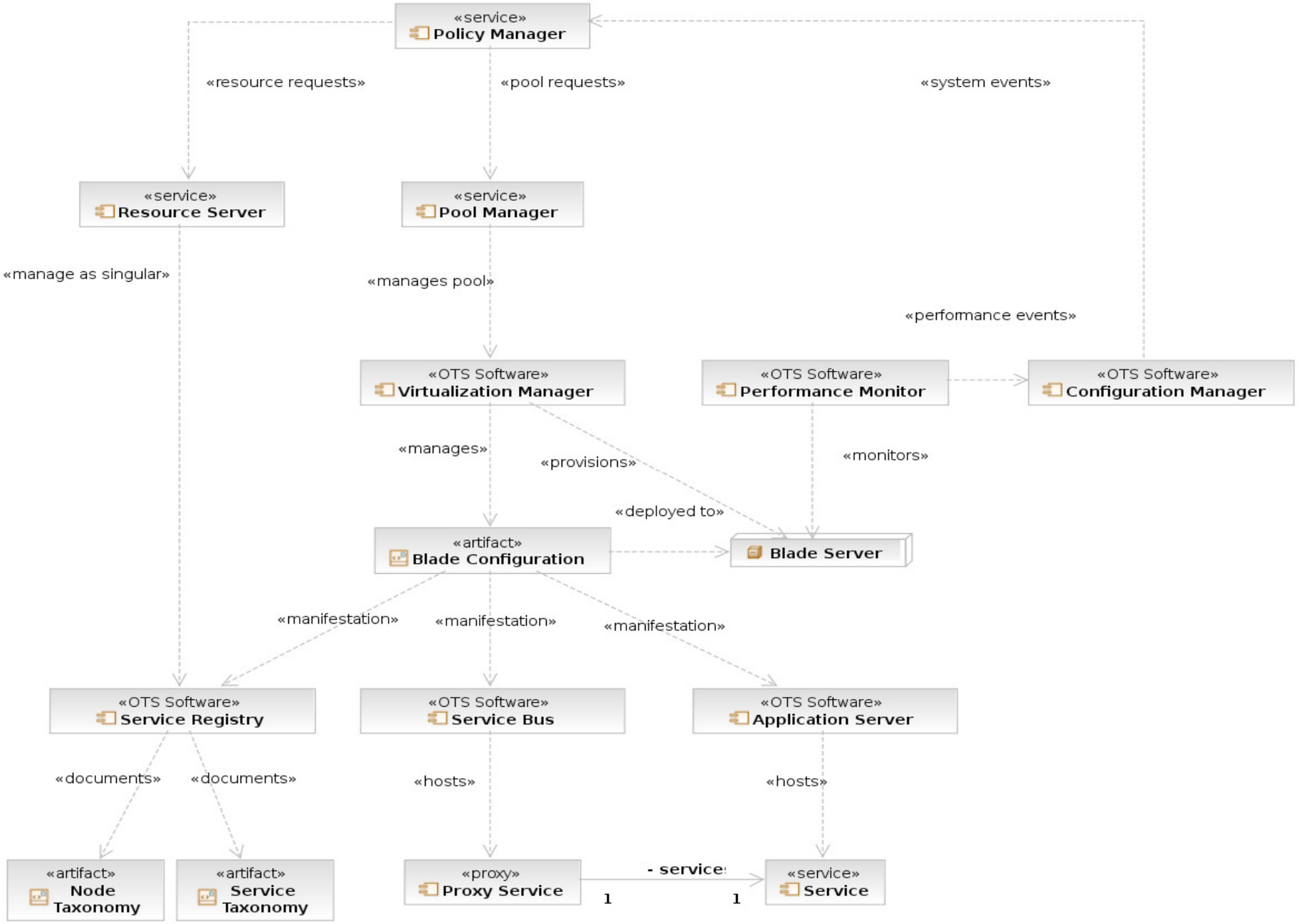

Figure 13 - Deployment Model

3.4.3 System Management Model

System Management is accomplished through the use of the Policy Manager, Pool Manager, and Resource Server components of the architecture. Each component provides a logical aspect of system management including: pooled resource management (Pool Manager), individual resource management (Resource Server), and implementation of performance policies (Policy Manager), such as Service Level Agreements (SLAs) for both Nodes (server machines) as well as system services (web services, etc). The combination of these management components provides a framework upon which system management schemes, both fully- and semi-autonomous, are implemented including (but not limited to):

- Run-time performance system management schemes,
- Schedule derived system management schemes,
- Heuristic driven system management schemes,
- Biomorphic system management schemes.

Figure 14 - System Management Model

3.4.4 Quality Attribute Analysis (Physical View)

Trades within the physical space include the management methodology best applied to the logical management structure identified in the System Management Component View. Candidates include:

- Run-time performance system management schemes
- Schedule derived system management schemes
- Heuristic driven system management schemes
- Biomorphic system management schemes.

Because this architecture is in support of a larger study on the biomorphic approach, the trade study is limited to a Run-time performance system management scheme (as the baseline), compare with a biomorphic approach. The previously identified quality attributes stand as the most relevant criteria by which to approach the problem. System management schemes must all meet minimum defined criteria for reliability, availability, scalability, and response time, but the best system will be the one that meets the minimum criteria while minimizing the demand for power and cooling.

CHAPTER IV
SUMMARY AND CONCLUSIONS

The power of analogy is that it provides a simplified means for understanding complex concepts. Application of a cellular life cycle to autonomous system operation demonstrates key concepts, which can be achieved if applied in an intelligent manner.

The study of biological systems as a means of understanding complex systems behavior isn't a new concept. Pursuant to that idea, the application of cell cycle inhibitors, and cellular death as a means of checking unbounded system growth seems to have merit.

I look forward to the application of the analysis provided in this paper. The resulting system should provide further insight into the proposed concepts.

As a continuation of the research, an axiomatic comparison between the cellular life cycle, endocrine system, and steady state insect colonies would be interesting study and perhaps better define the parameters of what is possible. As the system stands currently, given the simple mechanisms explored, it would be beneficial to provide a proof of concept implementation and publish the results as a follow on to the paper.

REFERENCES

1. [Horowitz, Noah (2005)] "Recommendations for Tier I ENERGY STAR Computer Specification", Natural Resources Defense Council. http://www.energystar.gov/ia/partners/prod_development/revisions/downloads/computer/RecommendationsTierICompSpecs.pdf

2. [Bell, Michael (2008)] "Introduction to Service-Oriented Modeling", Service-Oriented Modeling: Service Analysis, Design, and Architecture. Wiley & Sons, 3.

3. [Wikipedia 2008] Website: http://en.wikipedia.org/wiki/Biomorphic_robotics

4. [NASA (2004)] Lodding, Kenneth. "Hitchhiker's Guide to Biomorphic Software", ACM Queue vol. 2, no. 4 – June 2004. Website: http://www.acmqueue.org/modules.php?name=Content&pa=showpage&pid=166&page=1

5. [Hayes, Brian 2001] "The Computer and the Dynamo" American Scientist

6. [Carr, Nicholas 2008] "Welcome back to frugal computing", file://localhost/Website/http/::www.roughtype.com:archives:2006:11:welcome_back_to_1.php

7. [Gilder, George 2008] "The Information Factories", Wired Website: http://www.wired.com/wired/archive/14.10/cloudware_pr.html

8. [Gartner 2006] "Gartner Urges IT and Business Leaders to Wake up to IT's Energy Crisis", Egham, UK, September 28, 2006, Website: http://www.gartner.com/it/page.jsp?id=496819

9. [Copeland & Proudfoot 1999] "Alan Turing's Forgotten Ideas in Computer Science", Scientific American, Inc., April 19, 1999. Website: http://www.cs.virginia.edu/~robins/Alan_Turing's_Forgotten_Ideas.pdf

10. [W.H. Freeman and Company 2000] "The Life Cycle of Cells" Website: http://www.ncbi.nlm.nih.gov/books/bv.fcgi?rid=mcb.section.213

APPENDIX A
Phases of the cell cycle

The cell cycle phases are an extract from a Wikipedia entry. The source can be located at: http://en.wikipedia.org/wiki/Cell_cycle.

A.1 Phases of the cell cycle

The cell cycle consists of four distinct phases: G_1 phase, S phase, G_2 phase (collectively known as interphase) and M phase. M phase is itself composed of two tightly coupled processes: mitosis, in which the cell's chromosomes are divided between the two daughter cells, and cytokinesis, in which the cell's cytoplasm divides forming distinct cells. Activation of each phase is dependent on the proper progression and completion of the previous one. Cells that have temporarily or reversibly stopped dividing are said to have entered a state of quiescence called G_0 phase.

A.1.1 M Phase

The relatively brief M phase consists of nuclear division (karyokinesis) and cytoplasmic division (cytokinesis). In plants and algae, cytokinesis is accompanied by the formation of a new cell wall. The M phase has been broken down into several distinct phases, sequentially known as prophase, Prometaphase, metaphase, anaphase and telophase leading to cytokinesis.

A.1.2 Interphase

After M phase, the daughter cells each begin interphase of a new cycle. Although the various stages of interphase are not usually morphologically distinguishable, each phase of the cell cycle has a distinct set of specialized biochemical processes that prepare the cell for initiation of cell division.

A.1.3 G_1 phase

The first phase within interphase, from the end of the previous M phase till the beginning of DNA synthesis is called G_1 (G indicating gap or growth). During this phase the biosynthetic activities of the cell, which had been considerably slowed down during M phase, resume at a high rate. This phase is

marked by synthesis of various enzymes that are required in S phase, mainly those needed for DNA replication. Duration of G_1 is highly variable, even among different cells of the same species.

A.1.4 S phase

The ensuing S phase starts when DNA synthesis commences; when it is complete, all of the chromosomes have been replicated, i.e., each chromosome has two (sister) chromatids. Thus, during this phase, the amount of DNA in the cell has effectively doubled, though the ploidy of the cell remains the same. Rates of RNA transcription and protein synthesis are very low during this phase. An exception to this is histone production, most of which occurs during the S phase. The duration of S phase is relatively constant among cells of the same species.

A.1.5 G_2 phase

The cell then enters the G_2 phase, which lasts until the cell enters mitosis. Again, significant protein synthesis occurs during this phase, mainly involving the production of microtubules, which are required during the process of mitosis. Inhibition of protein synthesis during G_2 phase prevents the cell from undergoing mitosis.

A.1.6 G0 phase

The term "post-mitotic" is sometimes used to refer to both quiescent and senescent cells. Nonproliferative cells in multicellular eukaryotes generally enter the quiescent G_0 state from G_1 and may remain quiescent for long periods of time, possibly indefinitely (as is often the case for neurons). This is very common for cells that are fully differentiated. Cellular senescence is a state that occurs in response to DNA damage or degradation that would make a cell's progeny nonviable; it is often a biochemical alternative to the self-destruction of such a damaged cell by apoptosis.

APPENDIX B
Regulation of the cell cycle

Regulation of the cell cycle is an extract from a Wikipedia entry. The source can be located at: http://en.wikipedia.org/wiki/Cell_cycle.

B.1 Regulation of cell cycle

Regulation of the cell cycle involves steps crucial to the cell, including detecting and repairing genetic damage, and provision of various checks to prevent uncontrolled cell division. The molecular events that control the cell cycle are ordered and directional; that is, each process occurs in a sequential fashion and it is impossible to "reverse" the cycle.

B.1.1 Role of Cyclins and CDKs

Two key classes of regulatory molecules, cyclins and cyclin-dependent kinases (CDKs), determine a cell's progress through the cell cycle. Leland H. Hartwell, R. Timothy Hunt, and Paul M. Nurse won the 2001 Nobel Prize in Physiology or Medicine for their discovery of these central molecules. Many of the genes encoding cyclins and CDKs are conserved among all eukaryotes, but in general more complex organisms have more elaborate cell cycle control systems that incorporate more individual components. Many of the relevant genes were first identified by studying yeast, especially Saccharomyces cerevisiae; genetic nomenclature in yeast dubs many of these genes cdc (for "cell division cycle") followed by an identifying number, e.g., cdc25.

Cyclins form the regulatory subunits and CDKs the catalytic subunits of an activated heterodimer; cyclins have no catalytic activity and CDKs are inactive in the absence of a partner cyclin. When activated by a bound cyclin, CDKs perform a common biochemical reaction called phosphorylation that activates or inactivates target proteins to orchestrate coordinated entry into the next phase of the cell cycle. Different cyclin-CDK combinations determine the downstream proteins targeted. CDKs are constitutively expressed in cells whereas cyclins are synthesised at specific stages of the cell cycle, in response to various molecular signals.

B.1.2 General mechanism of cyclin-CDK interaction

Upon receiving a pro-mitotic extracellular signal, G_1 cyclin-CDK complexes become active to prepare the cell for S phase, promoting the expression of transcription factors that in turn promote the expression of S cyclins and of enzymes required for DNA replication. The G_1 cyclin-CDK complexes also promote the degradation of molecules that function as S phase inhibitors by targeting them for ubiquitination. Once a protein has been ubiquitinated, it is targeted for proteolytic degradation by the proteasome. Active S cyclin-CDK complexes phosphorylate proteins that make up the pre-replication complexes assembled during G_1 phase on DNA replication origins. The phosphorylation serves two purposes: to activate each already-assembled pre-replication complex, and to prevent new complexes from forming. This ensures that every portion of the cell's genome will be replicated once and only once. The reason for prevention of gaps in replication is fairly clear, because daughter cells that are missing all or part of crucial genes will die. However, for reasons related to gene copy number effects, possession of extra copies of certain genes would also prove deleterious to the daughter cells.

Mitotic cyclin-CDK complexes, which are synthesized but inactivated during S and G_2 phases, promote the initiation of mitosis by stimulating downstream proteins involved in chromosome condensation and mitotic spindle assembly. A critical complex activated during this process is a ubiquitin ligase known as the anaphase-promoting complex (APC), which promotes degradation of structural proteins associated with the chromosomal kinetochore. APC also targets the mitotic cyclins for degradation, ensuring that telophase and cytokinesis can proceed.

B.1.3 Specific action of cyclin-CDK complexes

Cyclin D is the first cyclin produced in the cell cycle, in response to extracellular signals (eg. growth factors). Cyclin D binds to existing CDK4, forming the active cyclin D-CDK4 complex. Cyclin D-CDK4 complex in turn phosphorylates the retinoblastoma susceptibility protein (RB). The hyperphosphorylated RB dissociates from the E2F/DP1/RB complex (which was bound to the E2F responsive genes, effectively "blocking" them from transcription), activating E2F. Activation of E2F

results in transcription of various genes like cyclin E, cyclin A, DNA polymerase, thymidine kinase, etc. Cyclin E thus produced binds to CDK2, forming the cyclin E-CDK2 complex, which pushes the cell from G_1 to S phase (G_1/S transition). Cyclin A along with CDK2 forms the cyclin A-CDK2 complex, which initiates the G_2/M transition. Cyclin B-CDK1 complex activation causes breakdown of nuclear envelope and initiation of prophase, and subsequently, its deactivation causes the cell to exit mitosis.

B.1.4 Cell cycle inhibitors

Two families of genes, the cip/kip family and the INK4a/ARF (Inhibitor of Kinase 4/Alternative Reading Frame) prevent the progression of the cell cycle. Because these genes are instrumental in prevention of tumor formation, they are known as tumor suppressors.

The cip/kip family includes the genes p21, p27 and p57. They halt cell cycle in G1 phase, by binding to, and inactivating, cyclin-CDK complexes. p21 is activated by p53 (which, in turn, is triggered by DNA damage eg. due to radiation). p27 is activated by Transforming Growth Factor β (TGF β), a growth inhibitor.

The INK4a/ARF family includes p16INK4a, which binds to CDK4 and arrests the cell cycle in G_1 phase, and p14arf which prevents p53 degradation. And the amount of chromosomes are able to double at the same rate as in phase 2.

B.1.5 Checkpoints

Cell cycle checkpoints are used by the cell to monitor and regulate the progress of the cell cycle. Checkpoints prevent cell cycle progression at specific points, allowing verification of necessary phase processes and repair of DNA damage. The cell cannot proceed to the next phase until checkpoint requirements have been met.

Several checkpoints are designed to ensure that damaged or incomplete DNA is not passed on to daughter cells. Two main checkpoints exist: the G_1/S checkpoint and the G_2/M checkpoint. G_1/S transition is a rate-limiting step in the cell cycle and is also known as restriction point. An alternative model of the

cell cycle response to DNA damage has also been proposed, known as the postreplication checkpoint. p53 plays an important role in triggering the control mechanisms at both G_1/S and G_2/M checkpoints.