IDEA GROUP PUBLISHING
701 E. Chocolate Avenue, Hershey PA 17033-1117, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com
ITP4241

The Ecology of Software: A Framework for the Investigation of Business-IT Integration

Thang N. Nguyen
Candle Corporation, Thang_Nguyen@candle.com and California State University Long Beach, tnnguyen@csulb.edu

ANTHROPOMORPHIC AND ECOLOGICAL VIEWS OF E-BUSINESS

From an anthropomorphic (human-like) view, we consider e-business over the Internet as a giant living human body, hence living species. From an ecological view, we extend the concept of business ecosystem by Moore (1993, 1996) to particularly include software as living species within business ecosystems. As depicted in Figure 1, we establish a parallelism between "e-business" (giant living species) driven by software (also as living species) and non-software factors (e.g. funding – not shown in Figure 1) and "the natural ecology" conditioned by multiple living species and abiotic factors (such as temperature – not shown in Figure 1). Together, the two views give rise to the concept of e-business integration as a business-IT automation continuum that ranges from bits (microscopic) to business ecosystems (macroscopic). We argue that this parallelism particularly helps define a framework for the investigation of Business-IT integration, structurally (anatomically), functionally (physiologically) and behaviorally.

From a microscopic level, the trillions of bits (zeros and ones) in different data structures and streams float on the Internet and on its supporting infrastructure. These are created, shared, exchanged or modified by trillions of algorithms operating on them to keep business, scientific and industrial applications and systems alive and connected. This is in some fashion similar to the trillions of particles (electrons and protons) in the human body that are created, shared, or exchanged among different atoms, molecules or macromolecules. The trillions of chemical reactions on the latter in turn provide proteins, carbohydrates, or lipids to keep the human body healthy and to do physical or mental work. Thus, the Internet and its entire supporting system for e-business may be considered as a giant human body. From a macroscopic level, today's e-business and its supporting software involve many front-end enterprise portals using the Internet and a myriad of back-end supporting applications and database systems. This conglomeration of applications and systems includes (1) messaging infrastructure among different communities of businesses, (2) e-business application integration to make applications work together and (3) enterprise-level collaborations (e.g. strategic alliances, business partners, trading partners) among all kinds of business populations. The enterprise business organizations and the interoperability among them for collaboration and competition in the e-business environment may be considered as a collection of business ecosystems (term coined by Moore, 1993) that interact and compete for resources in a very complex fashion much like their natural ecosystem counterparts.

Figure 1: Parellelism between ecology and e-business

Particles	Bits
Atoms	Primitive data types (e.g. char, int, float, …)
Molecules	Complex data types (String, field, records,…)
DNA, RNA	Constructors methods
Organelles	Class methods
Cells	Object classes
Amino acids	Programming constructs (if-then, for-, while-, do-while, …)
Proteins	Algorithms
Tissues	Components
Organs	Applications
Organ system	Application systems
Organisms	Software products
Species	Software
Population	Product family
Community	e-Business
Ecosystems	Business ecosystems

GENERAL PARALLELISM BETWEEN ECOLOGY AND E-BUSINESS - FIGURE 1 EXPLAINED

The general parallelism in Figure 1 suggests a biologically/ecologically-inspired framework to exploit bio-ecological organization, interaction and behavior of digital species/digital organisms (denote software and software products respectively) and business ecosystems. In this paper, we will restrict our attention to such a framework for the investigation of business-IT integration.

At the microscopic level of this parallelism, *bits* may be considered as *particles*, *primitive data types* as *atoms* and *complex data types* as *molecules*. *Object classes* in the sense of object-orientation can then be considered as biological equivalence of *cells*. *Constructor methods* in object class are considered equivalent to *DNA/RNA* and basic *class methods* to *organelles* in cells. As proteins are composed of about 20 different amino acids, we suggest that *amino acids* are the equivalence of *programming constructs* such as if-statement, for-statement, while-statement, do-while-statement or other constructs built upon them. We postulate that self-contained *algorithms* (or general class methods of any class) play a role similar to that of *proteins* or *carbohydrates*. *Software components* (such as COM/DCOM) then may be biologically equivalent to *tissues*, *applications* to *organs*, *application systems* to *organ systems*, *software products* to *living organisms*, and *software* to *living species*. At the macroscopic level, *software product family* are considered as *population*, *e-business* as *community*, and *e-business ecosystems* (in the sense of Moore, 1993) as *natural ecosystems*. Thus, when ions, atoms and molecules involve in reactions to produce proteins and carbohydrates for living cells to maintain the life of an organism, zeros and ones in binary system and different data types make up software elements and constructs that define life for software classes, components, applications and systems. When the cell is the smallest unit of biological life, the software class (in the sense of OO) is the smallest unit of software life. When biological membranes in or between cells are junctions between biological pieces, programming methods (algorithms) and interfaces can be considered as junctions between pieces of software. Assembler program inline macros, COBOL copybooks, C and C++ include statements, Java import statements, and C++ and Java class hierarchy and inheritance are examples of implementing the concept of reusability. The reusability of code creates new code similar to the mating of species generates different phenotypes from a given set of genotypes. Each block of codes reused in a new class or class method carries a certain characteristic (property) to it, similar to genotypes and phenotypes. Thus,

This paper appears in *Issues and Trends of Information Technology Management in Contemporary Organizations,* the proceedings of the Information Resources Management Association International Conference.

the parallelism suggests that class inheritance and software reusability define the continuation of software life and its evolution in information systems supporting e-business.

While in the ecological continuum (left-hand side of Figure 1), there is a natural linkage between elements at one level and the next, a similar linkage between high-level strategic business thinking and low level business-IT operations, however, is not obvious in the digital continuum (right-hand site of Figure 1). Section 3 below adds another insight into why the digital linkage does not readily exist despite the effort in the previous biologically- and ecologically-inspired research and applications.

PREVIOUS BIOLOGICALLY AND ECOLOGICALLY-INSPIRED RESEARCH

Applying biological (cellular level and below) concepts to software in particular and to computer science/engineering in general is not new (Langton, 1995). As a matter of fact, many researchers for many decades have been looking at the analogy between biology and computer science for investigation and research on cellular automata, artificial life and the like (Mitchell, 2000). At the computer program execution level, CPU time has been thought of as "energy" resources memory as a "material" resource representing informational genetic patterns that exploits CPU time for self-replication. Mutation of these patterns produces new forms as digital genotypes. Different genotypes compete for resources such as CPU time and memory space. Concepts such as genomes, parasites, and ecological communities have emerged and been used to study digital life. At the computer program development level, the concept of genetic algorithms has been the focus of research by many computer scientists and has initiated many applications in computer and information discipline. These studies and research have been strictly geared toward low-level applications of biological insights and understanding.

Using an ecological approach to understand business and industry behavior has also been researched. Examples are financial-market prediction and population genetics as well as factory scheduling (Farmer, 2000, Mitchell, 1998). The most notable effort was that of James Moore (1993, 1996). In his award-winning article (More, 1993), Moore first looked at business competition as predators and prey. He raised such questions as, for example "How can a company like IBM create an entirely new business community such as PCs and then lose that market". In response, he provided a framework for understanding competition and strategy development. According to Moore, the businesses evolve as ecosystems where a business ecosystem is defined as "an economic community supported by foundation of interacting organizations and individuals – the organisms of the business world". The organisms can be anything, "a process, a department, business unit, or an entire company". The business ecosystems, he said, would go through four phases in co-evolution: Pioneering, Expansion, Leadership and Renewal (Moore, 1996). Moore's business ecosystem model, however, stays at the high-level of thinking for business strategy development.

The two approaches have been pursued separately. There has been no obvious connection or linkage between them although each approach has achieved excellent results and practical applications.

TODAY'S E-BUSINESS LANDSCAPE

Today's e-business, when restricted to integration, has two main issues: (1) (horizontally), it can be generally described as a collection of isolated, disconnected or fragmented information systems within an enterprise or across enterprises, and (2) (vertically), serious mismatches and gaps exist between the business part (e.g. high-level business strategies) and the supporting IT part (e.g. low-level IT operations). In fact, in a particular enterprise, departmental units are not readily and totally connected to others. Users have to log in and access many different and diverse enterprise applications for needed information and changes (e.g. Web applications, Lotus Notes databases, Relational DBMS, mainframe transaction systems) (Linthicum, 2001). Across enterprises, while e-business collaboration among them exists, the connectivity and data exchange between applications supporting different enterprise business operations are really limited to a handful of interconnecting applications.

In fact, the reason being that the automated interaction between the business applications can be very diverse. It can be as simple as a function call and data transfer between one short calling program and another (called) program (e.g. remote procedure call or RPC). The call may act synchronously (e.g. request and wait for reply) or asynchronously (with a message-queuing middleware such as IBM MQSeries or Microsoft Message Queue). It can also be as complex as commonly found in today's multi-threaded, load-balanced, load-distributed, disparate destination-based, n-tier client/server applications and/or enterprise-level process-based systems. They may belong to different object-oriented systems consisting of COM/DCOM or CORBA-compliant applications. The complexity in e-business integration arises since these applications, old and new, have been written in different computer languages and compiled on different operating systems. They run on different and incompatible hardware. They have different data structures, message types and formats. They use different transport protocols and use a wide range of different technologies and architectures (mesh, hub-and-spoke, bus or pipe).

The first effort in e-business integration may be identified as enterprise application integration (EAI). EAI has been driven primarily by businesses and industries. It offers mechanisms to make different applications and business services work together (Linthicum, 2001). From a different perspective, enterprises architectures have been promoted by Zachman (1996), DeBoever (1997) frameworks and many others. Recently, Service-Oriented Architecture has also been introduced by Gartner. These architectures and frameworks attempt to guide software development for systematic integration between high-level business mission/strategies to low-level IT applications as well as to respond to business changes. However, the gaps and mismatches remain wide between high-level enterprise architectures and low-level e-business application integration, between business strategies and IT strategies, between business strategies and business operations, between IT strategies and IT operations and between business operations and IT operations. This linkage effort is top-down and remain at high-level while the integration in making the e-business applications work together in EAI has become primarily patchwork because e-business integration, horizontally or vertically is complex, difficult, expensive and very time-consuming.

THE ECOLOGY OF SOFTWARE FRAMEWORK AS AN AUTOMATION CONTINUUM FOR E-BUSINESS

We define the ecology of software as an interdisciplinary field of study on the interaction between (1) software (as digital species) and software products (as digital organisms) and (2) their environment (Nguyen, 2001). In this paper, the notion of "ecology of software" is formulated as a framework for the investigation of business-IT integration. The framework is based on the parallelism between ecology and e-business to postulate that e-business has an automation continuum where software is considered as living species. Study of e-business integration from an ecological aspect would bring new insights to the understanding of software in terms of software survival, effective interoperability, investment impact on e-business success, and prediction of software usefulness and its contribution to the digital economy.

Within the digital automation continuum, our framework is specific in that it focuses on a particular digital species (software) in which digital organisms (software products) organize, evolve, interact, behave and impact on their environment. Our perspective is therefore complementary to existing effort by many previous researchers who have conducted investigations either (i) at the biological level e.g. cellular automata and artificial life or (ii) at the ecological level e.g.

business ecosystems as addressed in Moore (1996). Our framework particularly allows the investigation of the linkage between low-level automata and high-level business ecosystems, as described below.

Underlying Universal Concepts in the Automation Continuum of E-Business

In the following, three major universal concepts underlying the ecology of software are redefined and re-scoped. They are, namely, (1) *Object-Oriented (OO) class concept underlying the anatomy aspect* of e-business, (2) *business service concept underlying the physiology aspect*, and (3) *business policy concept to maintain and regulate business harmony*. The concepts are universal in that they exist everywhere in any enterprise.

A. OO object class concept in the anatomy of e-business

The OO class concept, hereafter, called briefly object class concept, has been used to describe any object or class of objects in the physical or mental world. When redefined and used in our framework, similar to cells in biology, the object class concept governs the creation and evolution of software constructs. In fact, the bits and their encoding configuration schemes (e.g. ASCII, EBCDIC) form simple data structures (integer, character, float, double, string, etc.). These in turn form advanced and complex data structures. Algorithms operate on these data structures. Based on the information hiding principle the data structures and algorithms/functions (called methods in OOP – objet-oriented programming) are encapsulated to create classes representing objects of any kind. An object class can be a physical entity or a mental idea. Examples are an employee record, a car, a loan or an order via the Internet. A new class can be generalized, specialized or aggregated from other classes. Thus, software concepts in OOP such as abstraction (generalization, specification or aggregation), encapsulation, inheritance, and polymorphism allow us to describe any object from bits to business objects, to business ecosystems and beyond. The *object class concept is therefore fundamental and universal in digital organism evolution* much like the cells in natural organisms. Not only can one declare and define any class, i.e. any physical (e.g. a building) or mental objects (e.g. a credit check), but one can implement it and use it in any OO programming languages such as C++ or Java or other pseudo-OO or component-based programming languages such as ActiveX. Thus, if cells are specialized to perform specific tasks and tissues are group of closely related cells adapted to carry out specific functions then OO classes and components (in the sense of Microsoft COM, Java packages or beans) are their non-biological parallels.

B. Services concept in the physiology of e-business

Just like cells in cellular communications, object classes are to service one another. One particular object of a class when instantiated may call on a service to be provided by a method of another class (via inheritance such as overloading or overriding function or interface). This service concept can be extended to levels beyond the basic object classes to components, applications and systems all the way to the business ecosystem level. In fact, the concept of service conveys the interrelationship between any two (or more) objects similar to the notion of cells or tissues, organs, organ systems or beyond) servicing other cells or tissues, organs, organ systems in natural living organisms or species. Note that *the service concept is also universal* because it exists at any level of business-IT granularity: coarse or fine-grained.

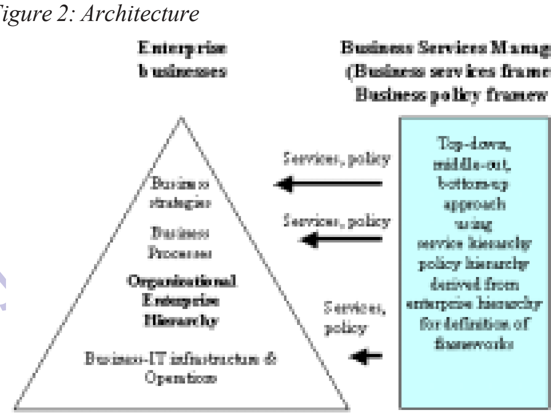
C. Policy concept in the controlling of health and harmony of e-business

The concept of business policy (rules, statements, conditions, thresholds, etc.) regulates the behavior of objects much like the biofeedback system and other mechanisms regulate homeostasis in the human body. In fact, all levels of living species or organism organization (cells, tissues, organs and organ systems) participate in these regulatory mechanisms. Similarly, policy exists everywhere: in a simple object class, a package, an application, a system, an enterprise or a group of enterprises. The concept of business policy governing the functioning of any organization (e.g. a department, a business unit, a company or a business ecosystem would help maintain the business harmony required for the health and wealth of the business. Therefore, like the object class concept and service concept, *the policy concept is universal*.

USE OF THE FRAMEWORK FOR THE INVESTIGATION OF BUSINESS-IT INTEGRATION

Essential to the ecology of software framework defined in section 5 are the three hierarchies: (a) an organization hierarchy, (b) business service hierarchy and (c) business policy hierarchy, that exist in any enterprise or business. The last two hierarchies are derived from the organizational hierarchy of the enterprise. *The organization hierarchy defines the anatomy of e-business*. Each organizational unit, as a component of the hierarchy, is created to fulfill a particular mission within the global enterprise mission that is visionary and measurable, similar to that of cells, tissues, organs and organ system in a human body or any living species. Whatever the specific mission of each organizational unit is, one unit is to service one or more other units in the organization and the entire enterprise is to service some consumer or user, some group of consumers or some business. *This service hierarchy defines the physiology of e-business*. The service hierarchy describes how the enterprise organization functions and behaves in a fashion similar to the circulation system, digestive system, reproduction system and the like in living species. Each and all services at any level of complexity of an enterprise can be conditioned by business policy that regulates the functioning of the services. *The policy hierarchy defines how the enterprise regulates itself* to maintain a healthy status just like homeostasis in humans. Thus, business services prescribe and implement the enterprise strategies and architectures, and business policy expresses and controls the dynamics of business changes and the health of the enterprise operations.

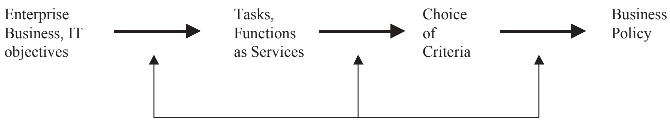
Figure 2: Architecture



The triangle on the left of Figure 2 represents an enterprise business, any business. Simply speaking, the business mission of the enterprise is supported by business strategies in response to business needs and business changes. The strategies are responsible by an organization. Business processes for the enterprise organization across all organizational units (marketing/communication, finance, accounting, R&D, manufacturing, production/distribution, etc.) are defined. The business processes in turn drive all business-IT operations supported by the infrastructure of the enterprise.

While the enterprise organization and business services define the business's structural and functional aspect (anatomical and physiological), it is the policy that regulates the successful functioning of the enterprise (behavioral aspect). It therefore addresses the dynamics

Figure 3: Derivation of business policy from enterprise business services



of the business, primarily business changes. This in turn controls the behavior of business and IT operations to obtain overall operational harmony. Figure 3 shows a simple scheme on how to derive business services and policies.

The three hierarchies defined are drawn based on the underlying universal concepts in section 5. They are in essence are part of the automation continuum framework generally described in Figure 1. Software and software products supporting the enterprise organization units are digital living species and digital organisms that service these organization units in maintaining the business harmony and wealth governed by business policies. This biologically-inspired view of enterprise e-business gives rise to the use of biological mechanisms well-known in biological and ecological systems in the understanding the business-IT integration in a particular enterprise or across enterprise. This will be the emphasis of our future research.

CONCLUDING REMARKS

If genetics and biological evolution are the continuation of species life, then software inheritance and reusability defines the continuation of software life and its evolution. Software classes (in the sense of OO), components, applications and application systems as services and e-services interact and behave in the e-business environment like cells, tissues, organs and organ systems that are kept operational by the interaction between macromolecules as seen in the respiration system, digestive system and others. In addition, our view of e-business as a giant human body would give insights into how the software pieces (equivalent to their biological counterparts: cells, tissues, organs and organ systems) would work to maintain enterprise business harmony similar to human homeostasis. The parallelism between human body and automated e-business suggests that lessons learned from studying human anatomy and physiology may reveal new techniques to be used in e-business automation.

Our future research will detail the mapping between digital living species and organisms to enterprise business organizational units, services and policies for the purpose of finding a systematic linkage in business-IT integration. Once this is understood, our research will extend to the investigation of software interoperability, behavior and impact to the digital economy.

REFERENCES

DeBoever. Larry. “Concept of Highly Adaptive Enterprise Architectures”, online, Meta Group, 1997

Farmer, J. Doyne (2000). Market Force, Ecology, and Evolution, SantaFe Institute.

Langton, CG. Editor (1995). Artificial Life: An Overview, MIT Press.

Linthicum, David. (2001). B2B Application Integration: e-Business-enable Your Enterprise, Addison-Wesley, 2001.

Mitchell, Melanie (1998). An introduction to Genetic Algorithms, MIT Press.

Mitchell, Melanie (2000). Life and Evolution in Computers, SantaFe Institute.

Moore, James (1993), Predators and Prey: A New Ecology of Competition, Harvard Business Review, May-June 1993.

Moore, James (1996), The Death of Competition, Harvard Business Review, May-June 1993.

Nguyen, Thang. (2001). The Ecology of Software, 6th International Conference of Decision Science Institute, Chihuahua, Mexico, July 2001

Zachman, John. “Concepts of the Framework for Enterprise Architecture:, Zachman International, 1996