Preface

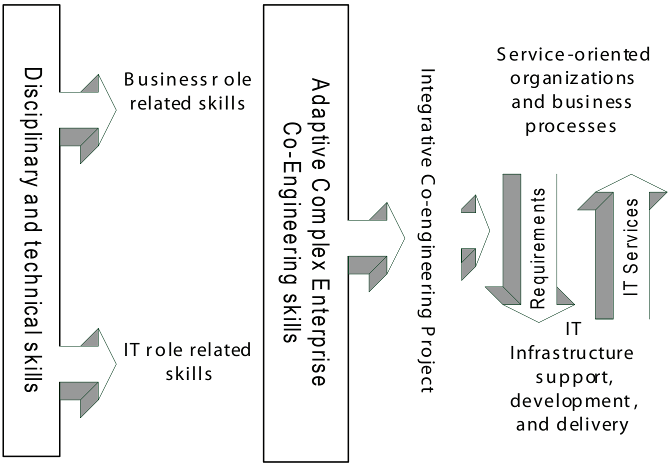
MOTIVATION

What is the book about? This inter-disciplinary and integrative book combines recent evolutions in enterprise architectures and services to provide a prescriptive framework for the practitioner architect and the needed background for a researcher.

This book covers the:

- *Solution-driven* development and improvement of IT (Information Technology) services for Business Efficiency, Innovation and Agility.

Figure 1. The Adaptive Complex Enterprise framework provides integrated business and technical Co-engineering skills that can be applied to improve changing business processes and complex IT systems.


Disciplinary and technical skills
Business role related skills
IT role related skills
Adaptive Complex Enterprise Co-Engineering skills
Integrative Co-engineering Project
Service-oriented organizations and business processes
Requirements
IT Services
IT Infrastructure support, development, and delivery

- *Methods* for practicing architects, business analysts, project managers, and software engineers to achieve solution success through creating *Work Products* that reflect a shared understanding of:
    - Customer requirements
    - Business goals
    - Product/service life-cycle
    - IT Infrastructure support needs
- *Context* for better IT-related decision making through alignment within the complex enterprise.
- *Integrates business and engineering knowledge needed for* practice and research through the Adaptive Complex Enterprise service ontology and framework. This integrated knowledge seeks to give students and professionals conceptual tools for improved practice and a basis for the formalization of experiential learning for further research advancement.

Why is this needed?

Dynamic economic context demands service agility and requires overcoming challenges in the way we:

- Govern complex Business-IT systems and their evolution
- Deal with constant change in underlying business Agents – organizations, processes, and technology components
- Deliver value despite significant business constraints

Motivating Adaptive Complex Enterprise (ACE) Frameworks for

- Adaptive architectures that deploy quickly to allow new behaviors to emerge
- Holistic, interdisciplinary work products that guide effective practice.
- Deeper principles critical to improved analysis, increased re-use, and reduced costs.
- Continued leverage of investments in installed enterprise systems.
- Formal and integrative interdisciplinary knowledge critical to useful academic advancement through leverage of industry practice.

What is the Adaptive Complex Enterprise (ACE)?

- Agents organized by a customer-provider service ontology representing their fine-grained interactions and value contributions (IDC 2007)[a]

- A conceptual framework by which we can continuously improve the value delivered by a complex system of business agents in dynamically changing environments
- A way to integrate the application of interdisciplinary business, systems engineering and IT methods and knowledge

Dynamic Context: Driven by a rapidly changing global environment that includes communications, competition, business growth, and technology innovation, 'services'[b] now account for a pre-dominance of the labor force in most economies. To thrive, the service enterprise must sense, respond, and adapt[c] to external conditions and leverage new opportunities (Haeckel 1999). In addition to addressing externally-driven uncertainty, the service enterprise must also embrace change and variation in interactions within its own organizations and systems that are getting more and more complex. These have been identified as issues to be addressed in the emerging field of *Services Science Management and Engineering*[d].

"Economic conditions like the recent recession are likely to make real-time delivery of products and services all the more attractive. Real-time enterprise technology can enable companies to answer queries instantly, monitor business on a continuous basis for quicker responses to shifting market conditions, and offer new products and services as new data becomes available. Behind the drive towards real-time enterprise is new hardware that gathers more real-world data than ever before, and software technology that eases or eliminates the trade-off between integration and flexibility. The technology is poised to have a tremendous effect on internal corporate operations while at the same time increasing the economy's fluidity--and possibly its turbulence[e]".

These trends severely challenge traditional disconnected business, technology, and architecture frameworks that are designed to manage within simpler and more static and environments. While IT (Information Technology) with SOA (service-oriented architecture concepts and related technologies) is now positioned, it is not yet practically implementable as a tool for the service economy. This requires the development of a new *integrated discipline* for managing complex systems. Such a discipline must enable new tools and methods that are responsive to the external environment, making adaptive behaviors possible. This book is a contribution towards that goal.

Motivating Adaptive Enterprise Frameworks: As companies seek to stay competitive and offer more services through product innovations and variations new processes and information needs arise. These needs are often not met despite the overall investment in enterprise applications (Gartner 2003)[f]. The resulting systems – composed of both organizations and software - are often found to be rigid (Nates

2003)[g]. To succeed we have to think of systems differently – as complex systems of made of a small number of interacting patterns[h].

Services innovation requires emergent architectures frameworks: Service-oriented organizations typically have the dual challenge of embracing complexity due to externally-driven variation while simultaneously managing the evolution of enabling complex IT systems. However, confusing build-versus-buy-versus-adapt options confront the businesses looking to overcome IT limitations. Of these, the buy option is usually pursued first. However, when off-the-shelf applications fall short, service-oriented organizations continue to invest in enterprise-level initiatives and custom integration to improve performance. These investments often fail to deliver the expected value and return. This is due to application challenges such as overlapping application functionality, existing lack of services, and the lack of a practice methodology for developing realistic estimates of costs and risks. It is no wonder that only a small fraction of enterprise projects are on time and on budget. As IT use gets more pervasive within the extended enterprise, these challenges make innovation and adaptation to externally-driven change even tougher. These challenges need to be jointly addressed by the Business and IT organizations. This requires enterprise architectures that can facilitate the emergence of new externally-driven adaptive behaviors.

Creational Versus Operational and Continual Improvement: Traditional Computer Science curriculum has focused primarily on the creational aspects of software and hardware systems. The challenges of enhancing the overall functionality and performance of a deployed system is left primarily unaddressed within the curriculum. With the increasing complexity of deployed systems, there is a need for continual improvement approaches that take a more systemic view and system engineering methods for creative tasks in the context of an operational system.

Holistic, interdisciplinary practice frameworks are required for effective practice. Thousands of relevant books and publications provide an explosion of information - strategies, initiatives, technologies, and practices – all promising to improve the performance of today's Business-IT systems. Five Forces, Balanced Scorecard, Enterprise Architecture frameworks, Customer Relationship Management, Enterprise Resource Management, Supply Chain Integration, Product Data Management, eStrategy, Re-engineering, World Class, Six Sigma, Total Quality Management, Just In Time, Lean, Concurrent Engineering, Enterprise Integration, Workflow Management, Middleware, Web services, Service-Oriented Architectures, Security, and Requirements and Organization Engineering are just a few among the topics covered.

However, this material is very poorly integrated for effective practice (Denning 2003, Armour 2003). Today's knowledge for engineering complex systems is at best descriptive, ad hoc, and pre-scientific[i]. Underlying principles are not abstracted for high-bandwidth communication and effective application by interdisciplinary implementation teams. How can we work locally in technical teams, yet make decisions that further overall business goals? When do we know that the issues we address are adequate and complete? When do we know when we have a truly new idea? How do we asses project impact on organizations and existing systems? How do useful best-practice frameworks relate in a deployment? How can we reduce custom code and reduce maintenance costs? These are just some of the questions that need to be addressed (Computing Curricula 2005).

Re-invention is in Part Due to Failure in Industry and Academic Collaboration. Despite the common elements across projects and businesses, most major IT projects in enterprise integration are often traditionally treated as "one-offs". Further, there is no shared *ontology* and basis for shared principles. The various disciplines involved understand the same enterprise integration concepts in a unique way. For example, the ubiquitous notion of a 'process' means different things to the distinct practicing disciplines of business, systems engineering, and computer science. To illustrate the point, a systems engineer will be concerned with the *behavior* of the process while the computer scientist is more concerned with the *representation* of process behavior.

Consequently, strategies for integration are not only re-invented on a project-by-project basis but left to the particular experiences of the team of practitioners[j]. With some exceptions, much of the needed knowledge is still proprietary and deployed at high costs. To counter the proprietary trends, both end customers and technology and service vendors such as IBM and Hewlett Packard promote open practices and re-use at the more conceptual levels (Herzum 1999).

While the practitioner, analyst, and academic communities have developed useful frameworks over the last few decades they typically have two shortcomings. Either they focus on a very limited *slice of the problem* or the problem is *too abstracted.* In either case research in successful IT practice and deployment is scarce. (See Denning 2003 for a discussion). Within academia green field assumptions also make research less relevant. Thus, unrelated and abstract frameworks lead to a lot of overlapping materials, inefficient communication, and re-invention without the evolution of fundamental underlying concepts.

Deeper Principles are Critical to Increased Re-Use and Reduced Service Costs. Service organizations often focus on *differentiation* as a competitive strategy; on the other hand the business return on IT is tied to increasing *reuse*. Businesses defining their success based on uniqueness are indeed different when examined on the *surface* but often very similar when represented at *deeper* levels (Long & Denning 1995)[k]. In contrast, requirements communicated at the *surface* level generates a remarkably diverse number of IT implementaions. If every IT need is treated as a unique service project, it becomes difficult for technology vendors and IT workers to provide IT supported solutions and systems that are cost effective.

What is the Adaptive Complex Enterprise (ACE)?

ACE is a conceptual view of the complex interactions between services provided by any collection of organizations and systems; and *Co-engineering* is the theory that allows us to analyze, reason about, and improve the ACE performance. The service-interaction-based ontology is also introduced as the basis of the ACE representation (Adaptive Complex Systems 2005). This dynamic structure allows us to create work products that surface performance issues that can be analyzed and Co-engineered.

The ACE framework herein is based on a *service interaction ontology* that provides a *deeper* approach for defining principles that can be taught, practiced, and researched. Leveraging the representational methods of computer science, the analysis techniques of systems engineering, and business decision-making, the ACE framework leads to a shared understanding and increased communication bandwidth for Co-engineering of the Business-IT attributes of complex systems. Requirements and solution cost-benefit tradeoffs can now be discussed at deeper levels. Thus richer dialog can now replace the feeling of helplessness and déjà vu felt by experienced project managers as they face a lack of understanding within each new project team. New tools and methods can be developed. The integrative framework is based both on recent technology advances and conceptual successes of Complexity Theory.

Co-engineering simultaneously engineers the *Business tactics, IT, Operations, and Strategy (BioS)* performance based on established business and systems engineering principles and technology. The *ACE framework,* including both the representation and the Co-engineering theory, builds upon transaction theory to integrate Porter's Model, The Balanced Scorecard, Lean, The Open Group Architecture Framework, Organizational Engineering, IT Infrastructure Library, Component Business Modeling, Business Process Management, Service-Oriented Architecture, and Autonomic Concepts.

The prescriptive performance-centered Co-engineering method integrates these proven and existing best practices at a deep level using precise service-interaction ontology. The result of Co-engineering is a *roadmap* for achieving overall complex system goals and business aspects, like risk management, value creation, return, and payback. Within the organization, business requirements for IT are more clearly defined and deployed to enable business service solutions.

Building upon the core of proven frameworks, Co-engineering also allows us to analyze the ACE Work Products and align performance to the value add desired by the BioS stakeholders. At the same time, it provides a framework to leverage emerging opportunities for managing complex IT systems better with IT itself. Such an *active ACE architecture* advances adaptation by providing a conceptual structure for real-time monitoring of the BioS layers. The alignment across the BioS is achieved by defining global policies for adaptation and providing the context for locally improving autonomous resources.

Using case studies throughout, we show how a small number of ACE framework patterns can be dynamically applied to permit local and lightweight implementations by teams of practitioners. Thus, the ACE framework provides a body of integrated knowledge for high-bandwidth communication and for implementation by interdisciplinary project teams in a highly distributed fashion.

APPROACH

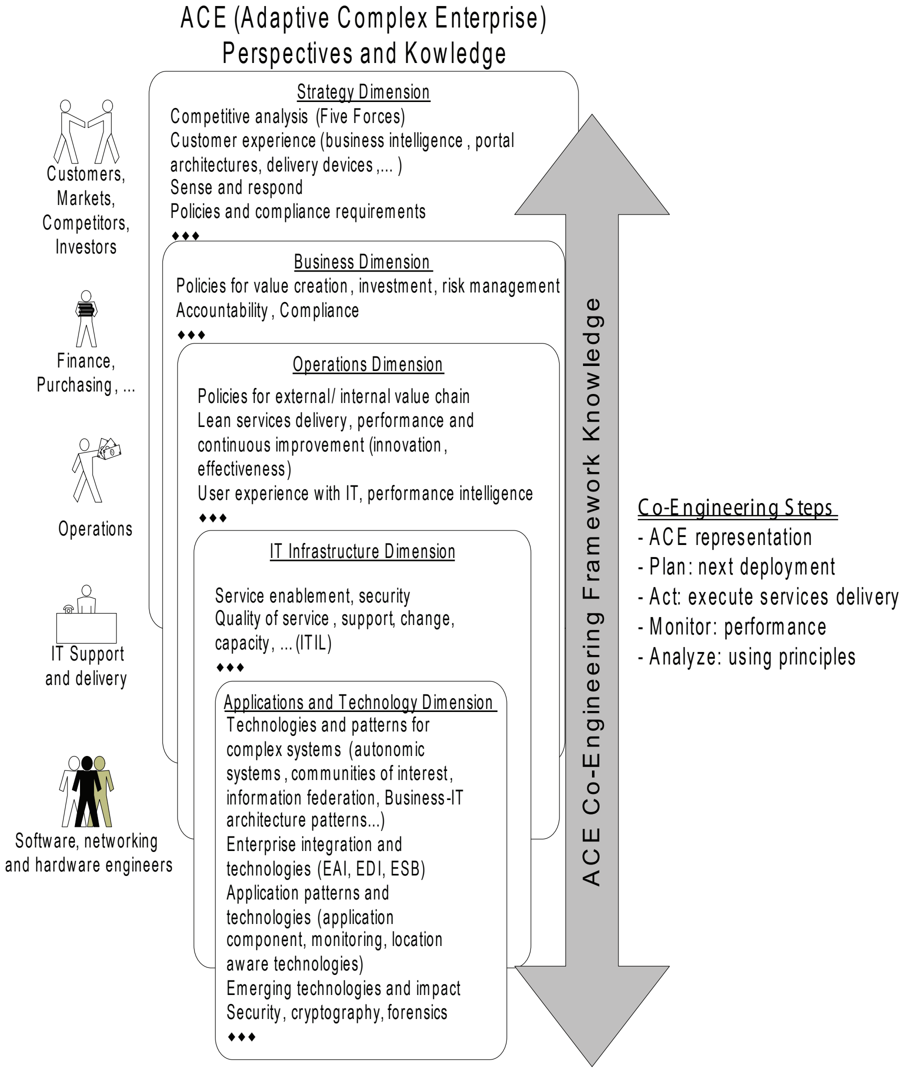
What is the ACE approach to Knowledge Integration?

- *ACE ontology* is based on the well-studied notion of a customer-provider 'interactions' needed to implement a business transaction using services and related patterns.
- *Captures horizontal and vertical nested interactions to* structure the associations between Business tactics, Business Processes, IT Infrastructure and Strategy.
- *Conceptual architecture and structure* allows measurements and application of Systems Engineering methods that relate IT and Business.
- *Facilitates the application of best practices* through this integrated performance-centered structure.

ACE further leverages the following concepts:

- Complex systems can be represented at ever-deeper levels using and re-using simple patterns that leverage component services.
- Patterns also relate global coordination policies to local business agents responsible for implementing change.
- Integrated framework for measurement leads to predictive and continuous improvement strategies and also improved global policies.
- Leverage technology for componentization and distribution.
- And, technology trends evaluated in the context of priorities and requirements.

Figure 2. ACE framework scope and integrated knowledge


ACE (Adaptive Complex Enterprise)
Perspectives and Kowledge
Customers, Markets, Competitors, Investors
Finance, Purchasing, ...
Operations
IT Support and delivery
Software, networking and hardware engineers
Strategy Dimension
Competitive analysis (Five Forces)
Customer experience (business intelligence, portal architectures, delivery devices,...)
Sense and respond
Policies and compliance requirements
♦♦♦
Business Dimension
Policies for value creation, investment, risk management
Accountability, Compliance
♦♦♦
Operations Dimension
Policies for external/ internal value chain
Lean services delivery, performance and continuous improvement (innovation, effectiveness)
User experience with IT, performance intelligence
♦♦♦
IT Infrastructure Dimension
Service enablement, security
Quality of service, support, change, capacity, ... (ITIL)
♦♦♦
Applications and Technology Dimension
Technologies and patterns for complex systems (autonomic systems, communities of interest, information federation, Business-IT architecture patterns...)
Enterprise integration and technologies (EAI, EDI, ESB)
Application patterns and technologies (application component, monitoring, location aware technologies)
Emerging technologies and impact
Security, cryptography, forensics
♦♦♦
ACE Co-Engineering Framework Knowledge
Co-Engineering Steps
- ACE representation
- Plan: next deployment
- Act: execute services delivery
- Monitor: performance
- Analyze: using principles

Why is ACE effective?

- Provides us with a lens that focuses on dynamic variation and how customer - provider agents provide value when they react.
- Co-engineering refers to ‘engineering in context’. Allows us to relate the agent interactions to goals that must improve though prescriptive actions.
- Addresses bounded rationality by permitting improved decisions without requiring detailed modeling of the enterprise.

- Spans Business, Systems Engineering and IT silos of knowledge. The technical and organizational ACE framework complements the business case method with measurement and predictive tools for proactive leverage using IT.

This is a book on service-interaction-based creation of value from business, information flow, operations, and technology performance perspectives. It is not just a book on any single aspect - such as integration, or security or compliance - but on an interaction language for representing and meeting *commitments between all the stakeholder and resource perspectives*. The result is a conceptual ACE representation of an enterprise with performance traceability that allows the Co-engineering of BioS perspectives. Thus the ACE framework allows service-oriented organizations to leverage the speed and dynamic flexibility provided by IT to run the ever more adaptive businesses. (See Christensen 2004 for the need for a more predictive capability and the special issues - Service Orientation 2007 and Services Science 2006 for and excellent background on services).

This book is an *evolutionary* one based on consolidated experiences of hundreds of companies. It is a book for the *successful introduction* of SOA projects into the services organization. And it is a book that describes more than planning for SOA implementation; it provides patterns for strategic planning, execution, and *improvement of business services using specific Work Products.* Finally, it is a book on *in-the-large and in-the-small ACE representation to empower teams* to identify and engineer the Business-IT tradeoffs through good decision making, investments, and implementations.

Thus, this is an integrative, rigorous, yet practical book intended for the advanced student or the practitioner with some real-world experience. It also identifies future research and opportunities for industry–university collaboration that is critical to furthering the discipline of *services science* (Services Science Management and Engineering*)*. To this end, the book is presented in a manner suitable for the practitioner within a services organization and it also has extensive end-notes and annotated references into a vast body of knowledge in three disciplines - Business, Systems Engineering, and Computer Science.

The ACE and Co-engineering framework of this book has evolved over the last two decades of business process solution experience of the authors applying workflow technology to adapt industrial-age enterprise systems to enable new services and business processes. The underlying change-management patterns themselves have been abstracted and validated though projects with over a hundred and fifty different companies and the experiences of even larger communities such as International Standards Organization (e.g. ISO 9000, ISO 20000), Supply Chain Council, Object Management Group, OGC, ITIL, W3C, OASIS, and others.

The main motivation of this book is to provide knowledge that more successfully Co-engineers - anticipates and better predicts - and thus enables an organization to become an Adaptive Complex Enterprise for services delivery.

ACE Knowledge Integration

The approach taken in the ACE Framework is to define a small number of patterns based on the universal and fundamental ontology of customer-provider service transactions. The ontology serves to *unify* a selection of patterns and methods discovered and often re-discovered across the disciplines. (See Alexander 1999 for an overview of patterns).

Componentization: The recent progress in IT technologies, especially in the areas of middleware, workflow execution, and Web services, makes it possible to compose business process enabling solutions by leveraging existing applications (Berners-Lee 2001). There is now also a wealth of experience that allows us to re-use specifications and write adaptors to existing enterprise applications treating them as components within larger systems and business components (Herzum 1999). Vendors such as Hewlett Packard, IBM, and Microsoft and organizations like the OASIS[l], Object Management Group[m], W3C[n], TOGAF[o], and National Institute of Standards and Technologies[p] now represent the experiences of numerous companies in this area.

Complementing the Case Method with Predictive Approaches: Broadly speaking, the widely accepted case method[q] has been used to develop decision-making skills by focusing on the behavior of the business . Here within the business, the IT details are primarily treated as a *black box* (Christensen 1999). The Co-engineering method of this book *complements* the case method by *relating* the business behavior to its internal layers and dimensions - operations, execution, and IT. This treatment of the internal layers as a *white-box* enables teams to explore cause-and-effect relationships and better *predict* the exact areas of maximum system improvement and IT leverage. By reasoning about the internal workings of the Business-IT system, management can now better identify implementation strategies. In summary, for improved business management, the *black-box* case approach identifies the faults of omission. The *white-box* Co-engineering approach discovers faults of commission, pinpointing those parts of the system whose *implementation* is faulty and needs to be adjusted.

ACE can now be Represented at Ever-Deeper Levels Using a Single Pattern Framework: ACE is based on easy-to-learn patterns incorporating uncertainty – dynamically and recursively applied patterns called fractals (Mandelbrot & Freeman 1983)[r] - to elaborate each black box layer with white-box details, repeatedly (CACM 2005). Fractals allow all these internal dimensions to be represented and related more precisely with external events. Thus, macro-level business services and objectives are related to micro-level operating performance targets. The resulting structure now provides cause-and-effect traceability leading to better adaptation.

There is a growing and diverse community that has been applying fractals, and the more general Systems Theory (Skyttner 2005)[s], Complexity Theory (Kiel 1994, Kelly 1998), or Chaos Theory, to characterize and understand the behaviors of complex systems and the manner in which they respond to unexpected demands and changes in their environment. Some examples of these works and initiatives are reported within the Santa Fe Institute[t], the Next Generation Manufacturing (Jordan & Michel 2000, Kiel 1994)[u], and the European Union's Sixth Generation Framework[v]. In general, the exact use of Complexity Theory varies within projects. It has been applied as a model for simulations aimed at understanding growth in biological systems, fluctuations in economic systems, and turbulence in fluids. Complexity Theory also has been used to increase understanding through *analogy* between socio-economic systems and biological systems for greater understanding of socio-economic phenomena[w].

Continual Improvement: Finally, we integrate the concepts underlying Complexity Theory with the pattern in Deming 1982[x] - *plan, do, study, and act* - to locally improve the *service transactions* performed within the ACE at both in-the-large, for example business, and in-the-small; IT, dimensions. The service transaction is itself achieved through the interactions and services of the IT infrastructure made of people, processes, and software applications.

We use examples and case studies to show how the methods and principles can be deployed. By using them along with current technologies we can implement patterns for more flexible cost-effective SOA solutions for the enterprise. We will show how the ACE Co-engineering methods can form the basis for experimentation and science.

Audiences and Learning Objectives

The book is intended for the practicing professional and the advanced student in any of the disciplines of Computer Science, Operations Management, Management Information Systems, Systems Engineering, or Public Policy. Aimed at the advanced or mature student, the book uses an integrative interdisciplinary approach to applying materials from several disciplines to understand and solve concrete business problems using IT and other technologies, as and when applicable (Figure 1).

For practitioners at the executive and senior technical management levels with a background in business, operations, or information technologies, the book provides the precise knowledge necessary to understand and isolate the deeper issues and challenges in applying IT to develop solutions to business problems.

For the advanced senior or graduate student with work experience, the book should be used as part of an integrative program enabling the student to make deep linkages and thus better decisions in the professional world. It is expected that the student will participate in a practicum – such as a two-term *industry-sponsored and industry-relevant project* - applying the methods of the book. Basic project steps and deployment work products are introduced at the end of each chapter to facilitate the learning process.

Solution Architect: We introduce the *solution architect* role as an analyst that has the skills of both the *Business* and *IT systems analyst* to function as an integrator. The business-type benefits from the "white-box" approach followed here in many ways. The Co-engineering method *connects* the *Business analyst* and the IT *systems analyst* with Work Products and tools to negotiate the ACE solution tradeoffs *without* needing very detailed disciplinary knowledge.

Finally, the knowledge presented here benefits IT workers in *technology companies* as well as IT workers working in *companies with internal IT departments*. IT workers in technology companies can deliver products better targeted to the service economy. Similarly, IT workers within internal IT departments can better understand the larger context in which their services have to be delivered. For background, a basic knowledge of business strategy, cost accounting practices, project management, databases, programming, and process/data modeling is assumed. Knowledge of UML and sysML[y] is a requirement and is used to represent enterprise architectures.

Learning Objectives

The student or architect will learn to represent and manage the improvement of complex systems with the following skills:

- Ability to develop conceptual representations and Work Products of Business, IT infrastructure services, Operations, and Strategy and linkages and perform analysis.
- Ability to lead an interdisciplinary team and apply Co-engineering analysis methods to deploy effective and innovative service solutions.
- Ability to apply enterprise architecture concepts to deal with the *conflicting forces* in real-world situations. By articulating tradeoffs and through more complete decision-making, the student will be able to contribute towards successful communication and implementations within the organization.
- Ability to perform gap analysis to identify IT or Business services that have to be created or modified to meet business objectives.
- Ability to identify the role of emerging technology trends in innovation.

- Ability to facilitate decision making through a more thorough articulation of Functional (business) and Non-functional (operational) requirements.
- Ability to implement the governance needed for effective in-the-large and in-the-small program management.
- Ability to apply principles that take into account trade-offs between Business function and IT cost-factors and risks as in any engineering discipline.
- Ability to govern and deploy best practices (like ITIL, Lean, and SOA) more effectively.
- Ability to enable services using technologies (such as Enterprise Services Bus, Enterprise Application Integration, Electronic Data Interchange, J2EE, and architecture patterns).
- Ability to conduct research in the emerging field of services science.

The skills are based on an enterprise framework that brings order and meaning to observations that may otherwise seem chaotic. It explains complexity in a simpler manner by establishing principles. This positions the student to conduct future research in more predictive management techniques and technologies. With this knowledge, the student will be able to play the role of a solution architect within the organization.

Scope and Learning Objects

The scope of topics and relationships is presented in Figure 2. The ACE Co-engineering framework begins by analyzing external influences on the organization. These in turn create Strategy requirements that guide the Business, Operations, and the enabling IT infrastructure of the organization viewed as ACE. The Co-engineering principles help identify evolutionary and revolutionary methods for improvement. The related learning objects[z] are alos as illustrated Figure 2.

Finally, the chapters contain examples of the business-related IT problems solved by the prescriptive Co-engineering steps. They address the real world challenges using rigorous principles. We also include the rationale for why the prescriptive methods work. Typically this will be presented in the form of end notes and references that index into a large body of related knowledge in three disciplines. This background work is presented for academic pursuit and the main flow is not interrupted for the reader with primary interest in practice.

Thus this book is written as a framework and an organized introduction to a vast amount of related interdisciplinary knowledge and underlying concepts that are the basis for Co-engineering complex systems. Many ideas are also presented for additional research. An extensive glossary and templates are provided at the end. We also take advantage of links to Wikipedia and other accessible sources of reasonably factual information. The papers referenced here are selected for historical value and accessibility for the researcher[aa]. Finally, each chapter has suggested readings that form a good background for future chapters.

PREPARING FOR PROJECTS

The instructor is encouraged to identify enterprise architecture projects and project sponsors. Preferred sponsorship would be from the senior management looking to advance the use of IT to improve efficiency or for service innovation.

REFERENCES

Alexander, C. (1999 September/October). The Origins of Pattern Theory: The Future of the Theory, and the Generation of a Living World. *IEEE Software, 16*(5), 71-82.

Armour, P. (2003 September). Closing the Learning Application Gap. *Communications of the ACM, 46*(9), 27-31.

Author, A. (2004). *The Engineer of 2020*. Washington: National Academies Press.

Berners-Lee, T., Handler, J., & Lassila, O. (2001 May). The Semantic Web. *Scientific American*.

Adaptive Complex Systems. (2005 May). *Communications of the ACM*.

Christensen, C., Anthony, S., & Roth, E. (2004) *Seeing What's Next: Using the Theories of Innovation to Predict Industry Change*. Boston: Harvard Business School Press.

Computing Curricula 2005. (n.d.). Retrieved November 23, 2008, from http://www.computer.org/portal/pages/ieeecs/education/cc2001

Deming, W. E. (1982). *Out of the Crisis*. Cambridge: Massachusetts Institute of Technology, Center for Advanced Engineering Study.

Denning, P. J. (2003 November). Great Principles of Computing. *Communications of the ACM*, *46*(11), 15-20.

Haeckel, S. (1999). *Adaptive Enterprise*. Boston: Harvard Business School Press.

Hammer, M., & Champy, J. (1993). *Reengineering the Corporation: A Manifesto for Business Revolution*. New York: Harper Business.

Herzum, P., & Sims, O. (2000). *Business Component Factory*. New York: John Wiley.

IBMSSME. (2008). http://www.research.ibm.com/ssme/ Accessed 2008.

Jordan, J., & Michel, F. (2000). *Next Generation Manufacturing*. London: J. Wiley.

Kelly, S., & Allison, M. (1999). *The Complexity Advantage*. New York: McGraw-Hill.

Kiel, L. (1994). *Managing Chaos and Complexity in Government*. San Francisco: Jossey-Bass Publishers.

Long, J. G., & Denning, D. (1995 January) Ultra-Structure: A Design Theory for Complex Systems and Processes. *Communications of the ACM*, *38*(1), 103-120.

Mandelbrot, B. (1983). *The Fractal Geometry of Nature*. San Francisco: W.H. Freeman.

Natis, Y. V. (2003). Predicts 2004: Application Integration and Middleware. Gartner Report AV-21-8190, Gartner Research.

Official Introduction to the ITIL Service Lifecycle (Official Introduction). (2007). Stationery Office Books (TSO).

Service Orientation. (2007 November). *IEEE Computer, 40*(11).

Services Science. (2006 July). *Communications of the ACM, 49*(7).

Services Science Management and Engineering. (n.d.). Retrieved November 23, 2008, from http://www.research.ibm.com/ssme/

Shaw, M. (1990 November). Prospects for an Engineering Discipline of Software. *IEEE Software*, *7*(6), 15-24.

Skyttner, L. (2006). *General Systems Theory*. City: World Scientific Publishing Company.

ENDNOTES

[a] According to IDC (Filing Information: March 2007, IDC #EMT1P, Volume: 1) Business transaction Management (BTM) is an emerging IT management

concept that can potentially address both IT complexity and business alignment requirements. At its essence, BTM is aimed at detecting and resolving problems at the granular level of interactions between IT elements that form a business transaction (e.g., online stock trade, travel booking).

[b] Some aspects of services: Close interaction between supplier and customer, Nature of knowledge created and exchanged, Simultaneity of production and consumption, Combination of knowledge into useful systems, Exchange as processes and experience points, Exploitation of ICT and transparency (Services Science 2006, p 37).

[c] In Haeckel 1999, the need for service businesses to be externally facing organizations in order to be responsive to changing market requirements is motivated.

[d] This, according to a recent May 2006 Summit, motivates the need for a 'Services Science Management and Engineering'(IBMSSME 2008).

[e] While this is a post Dot Com observation according to the Economist 2002, it well applies to the more recent mortgage crises and future unanticipated events.

[f] Gartner Dataquest (August 2003) forecasts - $536 billion worldwide IT services industry will grow through 2007 to reach $707 billion, with a compound annual growth rate of 5.7 percent. . Financial services providers have learned to bargain with IT services providers but remain dissatisfied with business case development for IT investments.

[g] According to Gartner Research (Natis 2003) industry trend is to replace monolithic, isolated application stovepipes by systems that are partitioned, distributed, integrated and designed-to-be-integrated. A third option of composing systems is being introduced to the more traditional build versus buy decision. However, much research needs to be done in this new arena of coexistence and cooperation of independent components. "Reducing complexity will be imperative to the software industry if it is to avoid massive setbacks and user revolts", according to Natis. The treatment of evolving business requirements, changing software components, and organizations as the Adaptive Complex Enterprise puts forth a framework for reducing the complexity of software engineering, deployment and maintenance. By treating complexity as approached in this book we will contribute towards engineering principles in the design, development and maintenance of complex systems.

[h] Alexander's pattern theories are among those few which have been perceived as so inspiring, or perhaps so fundamental, that they have successfully crossed over and taken root in disciplines for which they were never originally intended for. In this keynotes address, Alexander welcomes the integration of his ideas into the computing profession and hints that he believes it may have a deeper

relevance in this new field than it has thus far been able to acknowledge, perhaps because it has been taken too literally. When computer scientist are introduced to patterns they are often presented chiefly as time saving devices, or little chunks of information that can be written down nicely and referenced as building blocks when new problems are introduced. But do they play a role in a large process and can they truly be generative from a systems building perspective?

[i] An early paper (Shaw 1990) describes evolution of a typical engineering discipline. In comparison with a typical engineering discipline software engineering is still an art. The evolution of software engineering will require a good coupling between the practice and underlying science.

[j] Armour 2003 has a good discussion of the devastating effect of this “knowledge gap” on project success. This is also discussed in (Clements, Kazman, Klein 2002).

[k] Early work (Long and Denning) illustrates differences between deep versus shallow systems. Additionally in the field of artificial intelligence, shallow knowledge is said to be task dependent, additive and brittle. On the other hand deep knowledge is said to be task independent, describes underlying causal relations, and is complete at a certain abstract level.

[l] OASIS: http://www.oasis-open.org/home/index.php

[m] OMG - http://www.omg.org/

[n] W3C http://www.w3.org/

[o] TOGAF The Open Group Architectural Framework. How does TOGAF help deliver an effective IT architecture? TOGAF represents the world’s best practice in IT architecture development: Developed by the members of the Open Group, a not-for-profit body consisting of experienced users and vendors working in combination. Based on Commercial Off-The-Shelf (COTS) open standards. Genuinely vendor-neutral. Does not imply or favor any particular technology or paradigm. Backed by tools and professional services that are similarly independent of specific technology solutions, are practical, and reduce the costs of planning, designing, and implementing architectures based on open systems solutions. Endorsed by a body representing 25% of the world’s computing purchasing power. Demonstrated to work in practice by leading user organizations, who have documented their experiences in case studies that are freely available for review ..Available in the public domain (published by the Open Group on it’s Web site http://www.opengroup.org)

[p] NIST – National Institute of Standards and Technologies. Enterprise Integration projects.

[q] Case method - http://en.wikipedia.org/wiki/Case_method

[r] The conceptual ACE *patterns presented here are* based both on recursive Interactions at different scales associated with Complexity Theory as well as the problem solving pattern template approach, forming the basis for re-use and extensibility.

[s] International Society for Systems Science: http://isss.org/world/index.php

[t] Santa Fe Institute: http://en.wikipedia.org/wiki/Santa_Fe_Institute.

[u] In Next Generation Manufacturing: Methods and Techniques (Jordan & Michael 2000) manufacturing consortium projects using Holonics (http://www.cam-i.org/ngms) as part of the Intelligent Manufacturing System is described.

[v] European Commission: http://ec.europa.eu/research/index.cfm.

[w] Kiel 1994 provides an overview of the concepts of chaos theory and the science of complexity and demonstrates how public administrators can apply these concepts and create a new paradigm of organizational change and transformation of government.

[x] Deming is known as the father of quality. Deming stressed the need for Profound Knowledge, consisting of four parts 1) Appreciation of a system: understanding the overall processes involving suppliers, producers, and customers (or recipients) of goods and services, 2) Knowledge of variation: the range and causes of variation in quality, and use of statistical sampling in measurements, 3) Theory of knowledge: the concepts explaining knowledge and the limits of what can be known, and 4) Knowledge of psychology: concepts of human nature. Deming cycle provides a prescriptive process for applying this in the Plan, Do, Study, Act.

[y] SysML: http://www.sysmlforum.com/FAQ.htm, expresses systems engineering semantics for requirements management and performance analysis and facilitates automated verification and validation (V&V) and gap analysis. SysML model management constructs support the specification of models, views, and viewpoints and are architecturally aligned with IEEE-Std-1471-2000 (IEEE Recommended Practice for Architectural Description of Software-Intensive Systems).

[z] A learning object has been defined in the following ways. An entity, digital or non-digital, that may be used for learning, education or training, web-based interactive chunks of e-learning designed to explain a stand-alone learning objective. More at http://en.wikipedia.org/wiki/Learning_object

[aa] And hence we have not included many specialized journal references that address very limited aspects of the solution.