Context-specific intellectual capital—The next link in the knowledge chain

by C. A. Singer

Intellectual capital reuse and sharing in the specialized domain of project planning and execution are the focus of this paper. The key expansion beyond ordinary intellectual capital is pairing a context-specific intellectual capital structure with a matching context-specific intellectual capital management process. Beyond the fundamentals of gathering, archiving, indexing, and subsequently retrieving the appropriate intellectual capital, we explore both the context-specific structure and the related context-specific process specifically tailored to project planning and project execution. This tailored structure and process allow intellectual capital from multiple, different projects to be artfully blended, combined, and shared among multiple organizations and project teams.

Traditionally, the two approaches to managing projects are: task-oriented and output-oriented. In the task-oriented approach, a work breakdown structure defines the phases, activities, and tasks that must be accomplished. In contrast, an output-oriented approach focuses on the work products that need to be created during the project. In this paper, we present a work-product-dominant approach that links to matching work breakdown structures. This primacy of outputs rather than tasks is orthogonal to many conventional approaches for process definition, but lends itself well to a robust intellectual capital (IC) management approach.

The author's experience with context-specific intellectual capital (CSIC) is primarily with developing IC for software development and maintenance projects, most recently within the IBM Global Services Method. However, the approach described in this paper is applicable to a wide range of process-oriented and project-oriented endeavors.

This paper first presents the structure of IC, defining terms and providing an example from software development and maintenance. It next reviews how a project manager might use CSIC during project planning. Having defined CSIC and shown how it is used, the paper then defines and examines the four elements of the CSIC Engagement Model Lifecycle: authoring, project preparation, project execution, and experience harvesting. After discussing these four elements, an organizational "superstructure" for managing CSIC is described. To evaluate the potential effectiveness of CSIC, the reasons why projects fail are discussed and the result is related to CSIC use as a mitigator. Finally, additional observations and conclusions are given. This paper is based on experience and is not intended to be a research paper.

Skills-based approach. *Skills* imply the capacity for action toward some application. As such, much IC is prescriptive and attempts to supplement individual skills to achieve a solution or solve a problem. In organizations, IC may also focus on the team as an integrated collection of individuals. Both for the individual and the team, the application of IC focuses

 0018-8670/03/$5.00 © 2003 IBM

on either performing specific tasks or producing specific results.

Skills and intellectual capital go hand in hand. To be useful, the IC must be geared to the skill level of the practitioner. In the project domain, the tasks to be performed must be within the capabilities of the assignee. Similarly, the task descriptions and work product descriptions must be suitable for that assignee. A task such as "assure that the database is normalized" may be sufficient for a practitioner of given skills and experience, yet baffling to one of lesser or different skills. Similarly, a template for a "performance test plan" may be sufficient only for those who have studied a particular approach to test planning. Others, even if experienced in performance test planning, may not be able to develop the desired plan. This mismatch is addressed by requiring practitioners to have specific skill sets, training, or certification, or by providing sufficient additional tutorial documentation, or by both. Conversely, IC that micromanages tasks may be resented or thwarted by a highly skilled practitioner.

IC is not, however, a shortcut to skill building or a replacement for skills. Within the project context, it is an enabler for practitioners possessing the appropriate basic skills to perform certain tasks in order to create work products. In the project domain, context-specific intellectual capital recognizes the availability of individuals with specific skill levels as necessary preconditions for accomplishing specific tasks and creating or modifying specific work products. Multiple skill levels or skill sets may warrant different, custom instances of IC. When planning and executing projects, different cost, completion time, and variance may be associated with these different skill levels. Consider, for example, that both a general handyman with a pair of pliers and a master plumber with his toolkit can each fix a leaky faucet, but the costs, completion time, and perhaps the quality (and perhaps certainty) of the results may vary. Additionally, the IC necessary to provide suitable guidance to different practitioners with different skill levels may vary in granularity, width, or depth.

Output-based approach. At the organizational level, expertise goes beyond skills (inputs) and relates to goals (outputs). Specifically, organizational expertise focuses on planning and executing processes to attain organizational goals. Because organizations produce goods and services, the associated goals are, in turn, either product- or service-related. This paper focuses its attention on useful, project-related IC in a goal-oriented organizational context.

In the specific context of software development and maintenance, the CSIC discussed in this paper pertains to the work products (outputs) that contribute to the development of a new software system or the maintenance of an existing system. In information-based activities such as software development and maintenance, the problems associated with the reuse of IC are acute. This situation contrasts to the traditional, manufacturing realm where processes are to a great extent repetitive and skills are embedded in the product or the production line. In the information-based realm, IC is embedded in unique or customized processes, augmented by the professional knowledge worker's skills. Try as we might, we cannot turn intellectually driven activities into repetitive production line functions. Nonetheless, there is an accumulated body of project-related expertise that can be shared and used to our advantage.

Comparing IC, CSIC, and project artifacts

IC is the overall domain; CSIC is the subset that is explored within this paper. Just as it is vital to understand the distinction between knowledge and IC, it is important to understand the distinction between IC and CSIC.

IC is knowledge that is of value to the organization. IC is reusable, valuable, and manageable. To be reusable requires that the IC be in a place, form, and context that can be reused. Additionally, to be valuable, IC must be subjective, filtered, locatable, assessable, modifiable, and usable: The project planner must be able to find the *most* appropriate IC among multiple alternatives. Additionally, the planner is able to determine how appropriate the IC is to his or her specific efforts. The planner must also be able to tailor or modify the IC for current and future use. For IC to be usable, the project planner must be able to integrate and use this IC within his or her projects. Finally, IC must be manageable, including the ability to control access, assess usage and usage patterns, gather feedback, measure satisfaction, and make changes. Specialized IC-related systems are used to manage IC and help ensure that it meets these criteria.

CSIC is a specialized form of IC with various specified types of IC (see next section) to serve different purposes. CSIC is appropriately structured with well-

defined interdependencies among these specialized types of IC. It is context-driven; that is, the IC is applicable only to specific systems development project contexts. CSIC is designed and structured to serve as a set of flexible building blocks for project plans.

Project artifacts include any and all project-specific materials required to meet project or contract needs, or both. They include temporal documents, drafts, and other material—inputs, outputs, working documents, customer deliverables, and so on, but only some of these artifacts will become IC. In contrast to IC (and CSIC), project artifacts are managed via a project-specific control book tool or system.

The reason for defining project artifacts is exclusionary to distinguish them from IC. The transition of some project artifacts to IC is part of an authoring process discussed later. Project artifacts are managed at a project level; IC is managed at an organizational level.

The structure of CSIC

The CSIC approach centers on an architecture that captures and communicates those constructs necessary to promulgate best practices for reuse and sharing both when building the project plan and when executing this plan. This may include an optional tool suite that helps throughout the CSIC life cycle.

The key construct is the work product, which must be produced in order to achieve the goals of the project. Complementing work products is the work breakdown structure, documenting the tasks that create the work products. All these items are wrapped into a package named an engagement model. We now describe operational definitions for several key constructs.

- The *engagement model* is a packaged project plan model that includes a set of work products that has to be produced to accomplish the project at hand, synchronized with the work breakdown structure delineating the tasks needed to build, populate, or maintain these work products, and technique papers and guidelines to provide specific help in performing tasks and building the work products. An engagement model is a starting point from which a project team tailors its specific project plan.

- The *engagement template* is the instance of an engagement model tailored to a specific situation. An engagement template may also serve as a starting point for subsequent projects. It is a precursor of a project plan.

- The *work product* represents anything produced during the execution of the project. The sum of the work products is the totality of things that have to be produced in order to achieve project success. During project execution, work products may be dependent on other work products as inputs. A key feature of work products is that they are designed to be shared among multiple engagement models. The following are related constructs:

  — A *work product description* portrays in detail everything that has to be produced in order to complete a specific type of work product.
  — A *work product template* provides a local instance of the work product description tailored for the project at hand.
  — A *work product instance* is the actual contents of the work product. The work product contents are created and evolve during project execution, possibly with multiple successive tasks, each impacting the same work product.
  — A *deliverable* (or deliverable package) is a grouping or packaging of work products suitable for a customer or some other external actor. A deliverable may include portions of several different work products. It is a convenient construct for grouping outputs and mapping them to contract or other project requirements.
  — A *thread* is a convenient grouping of work products that have some commonality and often appear together in a project plan. A thread is an ease-of-use extension that grew from common practice.

- The *work breakdown structure* is the phase, activity, task, subtask, etc. It is the decomposition of work to be performed in order to create, modify, or maintain a work product.

- The *technique paper* provides a specific instruction on how to perform a procedure that links to a specific work product or a task. Technique papers are frequently oriented to specific skill levels and skill sets. Thus they are specific, tactical documents, not general instructions on how to do something.

- The *guideline* is a general "how to" or tutorial that links to any component of an engagement model.

Figure 1 Architecture overview

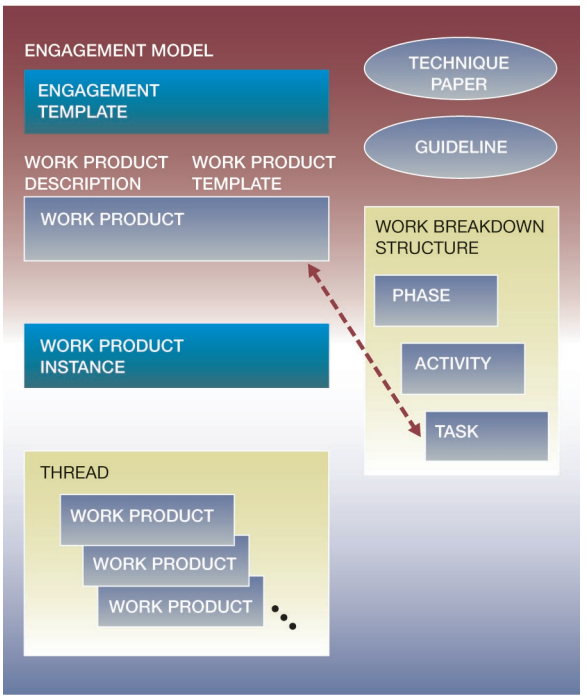
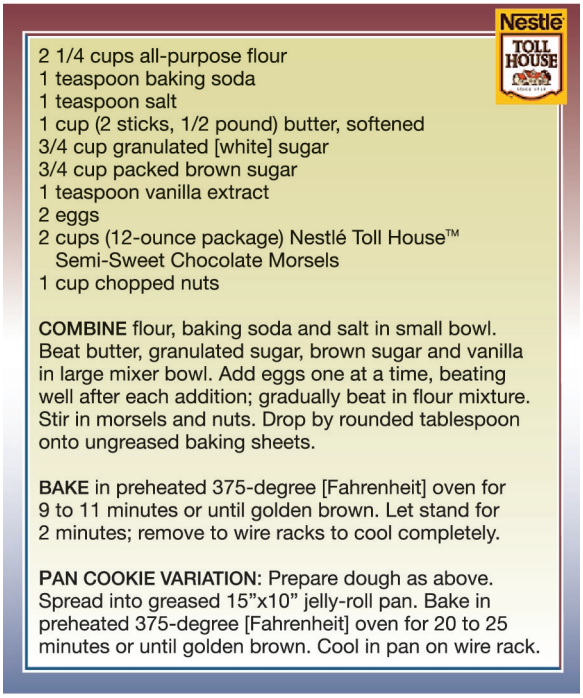


Figure 2 Nestlé Toll House™ chocolate chip cookie recipe



Recipe from bag of Nestlé Toll House™ Semi-Sweet Chocolate Morsels, reprinted with permission of Nestlé USA.

Beyond the engagement model and its components, a superstructure enables the management of models and their components. *Method architecture* is the "glue" that holds all of this superstructure together. The enhanced method architecture is work product dominant; that is, the design and subsequent selection of work products dominates the approach to building an engagement model. The work breakdown structure is a secondary feature that follows the work products. An additional key feature of the architecture is heavy reuse of work products. *Method* is a formal term for the sum total of contents within this architecture. For example, a role of "method author" is defined to more clearly describe the person who delves into the contents and authors an engagement model. *Method database* is a repository for all method components.

Figure 1 depicts the engagement model consisting of two primary component types, the work products and the work breakdown structure. Additionally, technique papers and guidelines are included within the scope of an engagement model. A thread is depicted as a simple grouping of work products.

A simple example of the capturing process

To be functionally useful, IC must tend toward the specific. Flexibility and generalization not only expand the opportunity for IC reuse into other contexts, but also challenge its applicability. In order to better appreciate this maxim, let us explore a simple recipe in extreme detail. In its simplest form a recipe is a single, unique process designed to produce a single unique output. It is clearly a form of context-specific intellectual capital. Note that significant documentation is necessary to fully describe even a simple activity such as baking cookies.

The recipe in Figure 2 includes a list of ingredients that are the *inputs*, such as flour, sugar, and eggs; a series of steps or *tasks*, such as combine, beat, add, and stir; and the expected *outputs*—Toll House** cookies. A recipe may also include an assumed generic *skill level* or understanding of specific tasks, such as how to crack an egg, mix ingredients or use a mixer, and turn on an oven. Additionally, there may be an implied list of required *equipment*, such as an oven, mixing bowls, and spoons. Lastly there is (implied)

preparation, such as gathering or purchasing the ingredients (inputs), equipment, and other items.

Adding decisions and offering variants provides the necessary flexibility and potential for customization; however, they may also complicate matters. There may be *decision criteria* or *task completion criteria*, for example, "bake until golden brown" vs "bake for 9 to 11 minutes." We see *options* in the recipe such as the "Pan Cookie Variation" that provides an alternative based on preference or resource availability. We must also consider *scalability*, which encompasses baking only a single cookie, or perhaps a hundred dozen cookies. Certain IC is readily scalable; others have limited ranges of applicability.

Again, the focus for the cookie baker is primarily on following the directions in the recipe. Little extraordinary skill is required; here "extraordinary" is defined within the context of normal cookie-baking activities. There is relatively little variation and only limited decision-making.

As a tutorial, let us look at this recipe expressed as a work breakdown structure:

Phase 1—Combine ingredients

- Activity 1—Measure and combine ingredients in small bowl
  —Task 1—Measure 2¼ cups all-purpose flour
  —Task 2— Measure 1 teaspoon baking soda
  —Task 3— Measure 1 teaspoon salt
  —Task 4—Combine in small bowl

- Activity 2—Measure ingredients for large mixer bowl
  —Task 1— Soften 2 sticks (½ pound) butter
  —Task 2—Measure ¾ cups granulated (white) sugar
  —Task 3— Measure ¾ cups packed brown sugar
  —Task 4—Measure 1 teaspoon vanilla extract

- Activity 3—Measure additional ingredients
  —Task 1—Measure 2 cups (12 oz. package) Nestle Toll House Semi-Sweet Chocolate Morsels
  —Task 2—Measure 1 cup chopped nuts

- Activity 4—Beat ingredients in large mixing bowl
  —Task 1—Beat ingredients from Activity 2, above, in large mixing bowl
  —Task 2—Add eggs one at a time into mixture, beating well after each addition
  —Task 3—Gradually beat in flour-based ingredients from Activity 1
  —Task 4—Stir in ingredients from Activity 3, above

Phase 2—Prepare and bake mixture

- Activity 1—Preheat oven
  —Task—Preheat oven to 375 degrees, Fahrenheit

- Activity 2—Drop mixture onto ungreased baking sheet
  —Task—Drop by rounded tablespoon onto baking sheet (until mixture is exhausted)

- Activity 3—Bake mixture
  —Task 1— Bake for 9–11 minutes or until golden brown
  —Task 2—Let stand for 2 minutes
  —Task 3—Remove to wire racks to cool completely

The in-depth description just given may not appear to be the best way to communicate how to bake tasty cookies. The work breakdown structure is lengthy, a bit stilted, and lacks the communication power and style of the printed recipe. However, if accurate, it is a most exacting description of the necessary tasks. In a CSIC environment, tasks could link to technique papers, documents that would supply additional information or guidance, such as how to break open an egg or grease a cookie sheet. Again, it may seem somewhat simplistic to consider having a guideline on "how to break open an egg" or "how to grease a cookie sheet," but how else can this be successfully accomplished in a consistent manner and in accordance with the recipe? The answer is three-fold—experience, skill training, and detailed intellectual capital.

In contrast, let us briefly consider a work-product-based approach. At first this may seem awkward because a task-based approach is more familiar. In looking at the recipe, we could define each of the intermediate components as a work product and then provide guidance as to how each of these work products is to be created. They are:

- Small bowl mixture
- Initial large bowl mixture
- Secondary large bowl mixture (with small bowl mixture added in)
- Final large bowl mixture (with nuts and chocolate morsels)

- Dropped mixture on cookie sheet, ready for baking
- Golden brown cookies

Details could be provided on how to prepare each of the previous work products, including the final work product, a deliverable: the cookies. Additionally, other recipes might benefit by reusing some of the previous work products, perhaps changing only the "final large bowl mixture" to, for example, delete the nuts, or replace the chocolate morsels with butterscotch morsels.

In the cookie recipe example, the focus is primarily on process execution skills as opposed to decision-making. Baking cookies can be considered a project. Hands-on training via an apprenticeship is likely to build the necessary skills for each task. IC in the form of a recipe serves as a vehicle for perpetuating this single process. Note that this example implies an individual activity without any significant organizational context or without any significant schedule or resource issues. Similarly, there is no formal planning, only preparation. Finally, the IC (recipe) is fairly stable; it is unlikely to be impacted by new experiences, changing technologies, strategic corporate change, environmental changes, etc.

Consider briefly a second, similar activity: baking bread. What do baking cookies and baking bread have in common? What cookie-baking skills are applicable to bread baking? Answers to these questions have implications on training, staffing, career paths, etc. Can a cookie baker be used when we have a bread baker shortage? Also, what IC associated with baking cookies can be gainfully applied (reused) to baking bread, and conversely, what IC would not apply or even have negative value (be wrong)? The CSIC framework addresses such issues.

Using CSIC for project planning

In relation to IC, project management has two primary phases: *planning*, which is determining and communicating how to go about doing something, and *execution*, which is performing to plan. Project managers frequently reuse contents from previous projects as they plan new projects. This reuse may be as simple as a fleeting thought or idea brought to mind when a current situation recalls memories of a similar previous situation, or it may be concrete, such as reusing portions of a previous project plan in the current planning effort. CSIC requires the gathering and storage of project-related IC for reuse and the retrieval and reuse of this IC both for project planning and for project execution. Characteristically, we are speaking about parts of previous experiences or plans.

What we are trying to accomplish. We are trying to develop CSIC as a set of reusable project plans and plan fragments that will provide a useful starting point for future project planning and that will then persist in order to support execution of the resultant project plan. A resultant benefit may be that the suite of project plans created within an organization adheres to what can be called a common "shape." The added benefits of this commonality are discussed later.

Project managers frequently reuse contents from previous projects as they plan new projects.

Sample scenario. Before delving into the details, let us explore a sample scenario: Suppose that you, the reader, are the project manager chartered to do a software development project. This project will produce an order entry and fulfillment system. The system will have a Web-based "front-end" (i.e., the customer will access the system via the Web), but you know that the new, integrated system will need to interface with several existing (legacy) systems, such as the corporate inventory management system.

Your first challenge is to develop a flexible, achievable project plan that will accomplish your goals. Without an IC system for reuse and sharing, you might begin from one of two extremes: a clean sheet of paper or a project plan that you encountered on a previous, similar project. You would then modify this plan and proceed.

A "best case" situation might be one where you have successfully planned and executed this type of project before—you can dust off your old plans and ideas, make a few experience-based adjustments, and then proceed. But what you may be doing is perpetuating an outdated or mismatched solution or, to be harsh, mediocrity. Note that a comfortable, tried-and-true solution may not be the current corporate best practice. There is the danger that this less-than-best prac-

tice will supplant the needed best practice. This situation can occur at the project or part level.

Given access to a library of reusable project plans, you need to intelligently search for an appropriate, best-fit project plan as a starting point. An *appropriate* project plan is one where assumptions and constraints are compatible with your project. A *best fit* plan is not only compatible with constraints and opportunities but also minimizes the amount of customization or tailoring that you will need to do.

Failing to find a suitable project plan for your "Web + legacy" project, you might seek to combine two plans: a Web-based development project plan and a second project plan for building systems that interface with legacy systems. But there is no assurance ahead of time that these two plans will be compatible or that you can successfully blend them together. The IC may not be of a form that can be subdivided into reusable components and thus "mixed and matched." Furthermore, the plans may incorporate different, incompatible development approaches, reflect different standards, use different terminology, and so on. By setting standards and keeping a constant eye toward future reuse and possible blending of IC from multiple project domains, the CSIC helps enable this blending of plans.

Ideally, a framework would be in place to ensure that all IC forms a compatible, interoperable set (that is, all IC would share compatible approaches, standards, terminology, and so on). Additionally, there would be a system in place to help locate and use this IC. The IC must be provided in a form that can be subdivided into reusable components. In this sample scenario, if there is no direct "Web + legacy" IC available, two separate sets of IC, one "Web" and one "legacy," might be located, and elements of each could then be blended together as a starting point for project planning.

This is our enhanced CSIC approach—building IC components within a framework that enhances the ability to later form a comprehensive whole from many parts. Common approaches are encouraged, thus increasing the opportunity to successfully blend multiple components.

Knowledge management life cycle

Before developing a detailed view of the Engagement Model Lifecycle, let us look at the general knowledge management life cycle. Mach and Owoc[1] cite a cycle (adapted from Mercier-Laurent, Jakubczyc, and Owoc) with four components: knowledge-accumulating, knowledge-creating, knowledge-sharing, and knowledge application. Lindvall, Rus, and Sinha[2] cite the fundamental Nonaka and Takenuchi model (circa 1995) with four types of knowledge conversion: socialization, externalization, combination, and internalization. They go on to cite "The Knowledge Lifecycle" from Wiig. This life cycle is depicted as: knowledge creation/acquisition, organization/storage, distribution, application/reuse, and (again) knowledge creation/acquisition. The Knowledge Management Forum[3] also cites Wiig in a position statement "On the Management of Knowledge" which identifies the following knowledge-related processes: create, build, compile, organize, transform, transfer, pool, apply, and safeguard knowledge. Similarly, S. Schoenert[4] designates the following life-cycle functions: (establishing) goals of knowledge, identification, generation, storage, distribution, usage, and evaluation. Gongla and Rizzuto[5] detail how communities of practice and accompanying knowledge networks have evolved within the IBM Global Services organization. Although a strong business context is embedded within this approach, knowledge use still is somewhat generalized.

Although there are multiple, different activities describing various elements of "authoring" (knowledge creation, compilation, storage, etc.) in each of the above cases, there is only a single category for knowledge use (application, usage, reuse, etc.). This reflects a context-free approach to knowledge management, where using the knowledge is likewise context-free, simply a single, broad, "umbrella" category.

It is not until we look at Basili and Seaman[6] and the Software Engineering Experience Factory that we see knowledge management applied to the specific endeavor of software development projects. We see a project organization with the traditional "plan" and "do" linked by the project plan and nourished by an experience base. This is closer to the CSIC approach described in the following section: Built on the foundation of the detailed structure provided by CSIC, the Engagement Model Lifecycle heavily emphasizes the use of IC in project planning and project execution. Thus the Engagement Model Lifecycle enables the level of IC reuse that is vital to successful project planning and project execution.

How to make the Engagement Model Lifecycle happen

The Engagement Model Lifecycle discussed herein is a specialized instantiation of the general knowledge management life cycles. The authoring activity is to a great extent analogous to knowledge gathering and filtering, classification, and storage. Project preparation and project execution each involve the reuse of IC, requiring location, selection, and retrieval of appropriate IC. Experience harvesting might be considered the maintenance phase of the life cycle.

In the previous example, valuable new CSIC, related to the specific scenario, may have been created during the planning process. Additionally, lessons learned during project execution may also contribute to the CSIC, either subtly as reflected in modifications to work products, or explicitly as additional work products, technique papers, or guidelines. The Engagement Model Lifecycle extends beyond project planning and project execution to this creation and modification of IC.

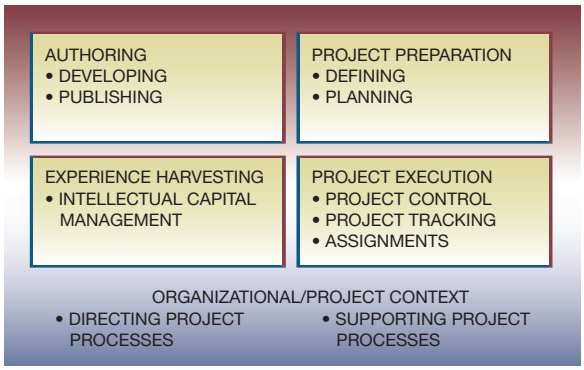
There are four major activities in the Engagement Model Lifecycle, shown in Figure 3. They reflect an idealized life cycle as follows:

1. Authoring—based upon or updated by experience
2. Project preparation (planning)—based on the authored engagement models
3. Project execution—based on the project plan
4. Experience harvesting—gathering experience as gained during project planning and project execution

All fit within an organizational/project context. This broader context better reflects the reality that one does not directly deliver process; one delivers *projects* based on process fundamentals, principles, and lessons learned. Similarly, one does not directly deliver systems engineering or quality initiatives such as the Capability Maturity Model (CMM) or ISO-9000; one delivers projects based on processes from these domains.

Authoring. Before exploring several cases of authoring uses, let us examine the roles employed in establishing best practices and their reuse as CSIC. These roles have evolved through experience. Because these are roles, not individuals, it is certainly possible for a single individual to fulfill multiple roles, or (more likely) for a team to share a single role.

Figure 3 Engagement Model Lifecycle



From an internal presentation by Paul Gignesi of IBM.

Nonetheless, we use the term "person" in describing each role. The *organizational SME* (subject matter expert) is the person who knows the business, the project, the program, and the account, and is a person who can also provide context. The *process SME* is the person who focuses on the work from the viewpoint of project management, project process, software engineering process, and quality. The *method author* is the person who translates or "ports" existing processes into a common, reusable central contents repository or database. The *contents facilitators* make up the central team that helps and enables the authoring process. These facilitators also enforce standards and encourage reuse. They have the broadest view of all available IC and look toward commonality and how the many may benefit from the work of the few. Conflict resolution is also part of this role. The *contents owner* is the person who owns and maintains the central contents repository. The *tool owner and developer* is responsible for the enabling IC tool platform, and so on.

Note that a detailed discussion of tools for managing CSIC is outside the scope of this paper, but see the following for an overview of the enabling IC tool platform.

Again, this roster is based on the author's experience and reflects the organization design developed by the sponsoring organizations. One should not confuse the number of roles with staffing requirements. The preceding roles reflect unique skills and responsibilities dictated by the structure of the CSIC and its supporting environment as well as the organizational dictates of those field organizations that perform

software development and maintenance projects. Some project participants may fill multiple roles, and, conversely, some roles require several workers. Clearly "one size does not fit all" because roles and organizational structure must be appropriate for the organizational context in which the CSIC is implemented. Further discussion of the CSIC-related organizational structure appears later.

The authoring process. The authoring process is treated like any formal project, with a project plan, a schedule, cost estimates, documents of understanding among organizations detailing responsibilities, and so on. This discussion, however, focuses more on the technical aspects of the authoring process, not the organizational or project aspects.

Simply put, either best practices or fragments of best practices are gathered for reuse. This gathering can be done proactively or passively. In the proactive mode, when management somehow senses a need for an answer (an itch), a project team is assembled to provide the answer (to scratch that itch). In the passive mode, best practices are sought, with the hope of finding that the answer already exists, such as, another team has been quite successful at dealing with a relevant opportunity, and therefore, there is an opportunity to use their expertise.

Evaluating (measuring) project success, evaluating project artifacts and IC, and gathering additional IC are important post-project execution steps. Unfortunately, many circumstances make it difficult to accommodate these vital project steps.

The proactive approach is essentially analogous to a design effort. A project is formed employing a team of experts to "design" an archetypical project plan and the components that support it. This approach may be followed when a series of similar projects are on the planning horizon. In contrast, the passive approach finds an existing best practice and somehow captures its essential pieces in a manner that is conducive to future reuse.

The authoring process—Use Case 1, freeze drying. The primary authoring paradigm is what we call *freeze-drying*. Analogous to the process that removes the water from coffee, leaving behind the essential good taste, this freeze-drying process takes the project practices of existing systems and extracts their substance and essence to make them available to other teams, consistent with the method architecture. Again, a key factor here is reuse. Authoring is not simply a translation of existing practices into work products, technique papers, and such, but an activity that involves analyzing the current IC and the method database contents and reusing or blending as much existing content as practical.

The method author, supported by the organizational SME and the process SME, reviews the (source) best practice, as is, and translates it into components, such as work product descriptions, technique papers, and a work breakdown structure. This first review does not consider blending with the current method contents database. Afterward the SMEs conduct a formal review for contents, redundancy, and completeness.

Many additional aspects of proposed IC need to be examined. These include: (1) decomposition—Is the IC at its most granular, decomposable level that enables reuse, or is it still a composition of several linked parts? Note that when decomposing IC, linkages should be maintained. (2) perishability—Does the IC have a reasonable period of freshness, or will it become stale with foreseen changes in technology or the business environment? (Consider, for example, the constructs developed for the anticipated problems deriving from the change in specifying dates as the twentieth century became the twenty-first century [the Y2K problem].) (3) applicability—How useful will the IC be in other contexts? Boundary conditions may be set, as, for example: "This work product is not to be used with real-time systems." (4) adaptability—Can this IC be transformed for use in other contexts and still be useful?

After the method author has isolated and documented this best practice, the contents facilitation team then works with the author to evaluate work products. An in-depth review of all proposed new work products is conducted, and reuse of existing work products from the method database is explored. Proposed new work products introduced by the new engagement model template can result in several outcomes: (1) A new, unique work product is added to the method database. It will be available to others in the future. Supporting material such as technique papers and selection criteria (when to use this work product vs a similar work product) are created. (2) An existing work product is determined to be a suitable replacement for the proposed work product, and the engagement model is modified accordingly. (3) The proposed work product is determined to be superior to a similar work product in the existing method database and is earmarked to replace

the existing work product. (4) A hybrid work product is developed, based on the proposed work product and similar, existing work products.

Work products, like all content within the method database, are under configuration control. Any changes undergo impact analysis (via an incidence matrix) involving all engagement models that currently reference the specified work product and are subject to a release process that includes changes to documentation, training materials, and so on. The various subject matter experts are part of the above process and also perform a formal review and approval. The contents facilitation team provides several additional quality checks ranging from technical writing issues (grammar, spelling, format, style) to overall technical reviews. This team also includes a publishing role for physically adding the engagement model and its components to the method database.

As the method database grows, it is necessary to provide better selection criteria and selection support to aid project planning.

The authoring process—Use Case 2, building new processes from existing components. This use case is a unique opportunity to build new processes, but not from scratch. As the process is detailed, existing model components, work products, and technique papers, for example, can be reused (best case) or serve as the basis for additional new components (next best case).

This use case is technically similar to Use Case 1; however, it frequently has several organizational or project differences, mostly focused around requirements. Among the requirements are determining what challenges the new process should be meeting, as well as review and approval processes.

The authoring process—Use Case 3, building new processes from scratch. For completeness, a *tabula rasa* approach should be considered, but in reality this solution is a worst-case example of Use Case 2—one where no existing model components can be reused. In most cases, this solution amounts to "reinventing the wheel."

The authoring process—Use Case 4, hybridizing multiple approaches. What should be done when there are multiple similar approaches within an organization to perform the same type of project? One answer is to select one by having a run-off. A second is to consider a hybrid, taking advantage of the strengths of each approach. This second method lends itself well to the analysis afforded by our work-product-dominant architecture. By extracting and then comparing the work products incident to each approach, a consistent basis for comparison can be developed. As appropriate, new engagement models can be built.

The authoring process—Additional use cases. Additional authoring processes, such as maintenance of existing IC, including both adaptive and corrective maintenance, need to be considered. However, they are outside the scope of this discussion.

The authoring process—The imperative of sharing and normalization. The ability to share (reuse) existing work products when building an engagement model is perhaps the greatest advantage provided by this enhanced approach. Various types of system projects encounter similar situations and can reuse work products to great advantage. For example, the team that the author led used the freeze-drying technique to capture four "best in class" maintenance-related engagement models. Then, given a method database with existing work products from "normal" customized application development engagement models, we needed to create only four new work products during our authoring process. In conjunction with the contents facilitators, we determined that one of our newly captured work products was superior to a similar work product already in the method database. After appropriate impact analysis was conducted, the switch from the inferior work product to the superior one was made in conjunction with a major new release of the method database.

Sharing may extend further via a concept called *threads*; for example, a natural clustering of work products related to testing may be defined as a "testing thread." In building a new engagement model, this testing thread may provide a convenient starting place for authoring that part of the engagement model.

Project preparation (planning). As was discussed, CSIC is the foundation for creating a project plan. During the project preparation and planning process, the CSIC is used to create a customized instantiation of an engagement template. This engagement template in turn serves as the basis for building a detailed project plan.

In practice, a method adoption workshop (MAW) is conducted to expedite the project-planning process. The MAW is a group meeting that requires participation by all project stakeholders both for their expertise and to assure buy-in. The central activity of the MAW is the creation of an engagement template. The starting point is the selection of engagement models or engagement templates as the basis for tailoring. Selection involves browsing through existing engagement models and engagement templates and their components, carefully examining constraints, strengths, and applicability, then determining which among several similar engagement models and engagement templates is a best starting point. The selection is followed by tailoring an instance that is the best fit for this particular effort. Tailoring includes identifying and selecting required work products, customizing work product instances as necessary, and then generating a work breakdown structure based on the (input/output) dependencies of these work products.

The work-product-dominant approach excels here. Experience shows that focusing on the work products that need to be produced in order to achieve project goals is most helpful in identifying and selecting an engagement model or engagement template as a starting point. Rather than concentrating on "what do we need to do," the focus on "what do we need to produce" is sharper. The details of a given work product can be explored to further determine suitability. Furthermore, tailoring work products is a straightforward means of customizing project plans to meet unique requirements.

Again, the CSIC is especially adept at supporting this rigorous planning process. The CSIC provides the appropriate IC alternatives when and where they are needed. In contrast, generic (unstructured) intellectual capital is unable to deliver this promise beyond the provision of search engines.

Although we do not discuss the intellectual barriers to the selection and adoption of other people's IC in this paper, it is worthwhile to highlight a general challenge. This challenge occurs when the project planner's mental model of how the project should be accomplished takes precedence over and is incompatible with those models in the IC. This is essentially a conflict of perception vs reality.

As the method database grows, it is necessary to provide better selection criteria and selection support to aid project planning. This is especially important when there are several similar pieces of content; for example, test plan for Web-enabled systems, test plan for enhancements, and test plan for corrective maintenance. Additionally, a library and retrieval system is needed to facilitate the reuse of engagement templates and other contents.

Project planning, an organizational context. Tailoring contents on an organizational or multiproject basis may be useful. For example, contents may be tailored to meet organization standards. Similarly, when a series of similar projects is on the planning horizon, tailoring may be done for this class of projects.

A MAW may take place at an organizational level as an engagement template is tailored to meet customer or contract needs. Work products may be added to meet unique requirements. Other work products may be deleted or modified. Still other work products might be designated as "mandatory" (not to be removed in subsequent MAWs) to satisfy management or contract requirements. The resultant engagement templates again provide a "closer" starting point for planning subsequent individual projects.

Method authoring may also take place for a similar group of projects; for example, a maintenance shop that is expecting several similar projects in the next six months might conduct a MAW for what it expects to be a typical project. The result of this class MAW is a tailored class engagement template. As each individual project begins, the project manager verifies that the class engagement template is suitable for that specific project and then continues.

A variant of this approach is to go beyond tailoring to produce a single class project schedule (using commercial project-scheduling software) for this class of projects. Each subsequent project then begins at the scheduling level: verifying plan appropriateness, estimating, adjusting roles and resources, resetting the project start date, and so on.

Project execution. Project execution is the logical "work the plan" phase within most project approaches. The normal project execution steps (as

specified by the project plan) take place here, as they would with any project, only they are supported much more effectively by the CSIC. The CSIC provides additional resources for project execution but does not fundamentally change the way in which project execution is performed or managed.

During project execution, work products are populated when and as appropriate. It is here that a support tool provides quick links to *relevant* supporting material such as technique papers, guidelines, or work product templates. This support is a key lever that the CSIC environment provides during project execution.

Tools that provide scaffolding for project execution, such as configuration management and issue management, are beyond the scope of this paper, as are tools that support the selection of, navigation through, and use of the engagement models.

Experience harvesting. Experience harvesting, although a vital component of knowledge management, is often neglected because it is seen as a help to other (future) projects but not as a direct benefit to the current project. In a worst-case scenario there is neither funding for this activity, nor management guidance to encourage it, and thus nothing is in the project plan for experience harvesting. The project team disbands and goes off to other projects without having done the appropriate post-project analysis and experience harvesting.

Because our approach makes good use of existing assets, it must have experience harvesting to hone and expand method contents. The impetus for inclusion of these activities into project plans and, especially, funding for these activities may need to come from organizational rather than project-specific resources.

The experience harvesting process then blends back into authoring. Feedback from projects helps to evaluate the utility and appropriateness of existing IC. This may result in proposed changes to existing IC or modifications to guidance, such as usage or applicability. Additionally, new IC, such as a new work product template instance, may be harvested from a current project. This new IC is then evaluated for inclusion in the method database.

The CSIC enabling tool platform

Just as the CSIC is a detailed instantiation of more generalized IC, so too, the CSIC enabling tool platform is a detailed instantiation of more generalized IC management systems. It manages the IC throughout its life cycle from authoring through analysis. This is not unique.

What sets the CSIC enabling tool platform apart is that it accommodates the various engagement model components described earlier. It deals specifically with both work products and a work breakdown structure, and is capable of linking the two. The generation of an appropriate work breakdown structure from a selected set of work products is key here. Additionally, the tool platform serves, both during project planning and during project execution, as an intelligent assistant providing direct linkage to supporting information as and where needed. Thus a work product template is only a "click" away when examining a work product. Similarly, technique papers and guidelines are appropriately linked and available.

Another outstanding feature of the CSIC enabling tool platform is the ability to provide secure access to (and downloads of) specific extracts of the engagement model database. The project planner can scope out the areas of potential interest and receive (only) the appropriate IC. In response to the planner selecting engagement models of potential interest, the tool examines the incidence among components in order to generate the linked advance (like a "bow wake") and then extracts an appropriate, nonredundant subset of the engagement model components.

Organizing for CSIC

The CSIC superstructure is the organizational framework for using CSIC in project management. In this section, we consider how to best set it up.

Organizational imperatives for successful implementation. There are four primary imperatives for successful implementation: (1) The approach requires a significant level of commitment at both the management and practitioner level. Long-term management commitment is mandatory. (2) A separate, dedicated organization for maintaining the method database must be established. This organization should also maintain the tool suite, also discussed in this section. (3) CSIC-oriented skills management must be integrated into the organizational professional development management approach. (4) All participants must receive appropriate orientation and training.

Experience has taught us that there must be a separate, adequately funded organization whose sole charter is to manage the CSIC. By overseeing each of the four phases of the CSIC engagement life cycle, this central organization can ensure the utility and the quality of the method database. This organization maintains the method database, helps blend content, and resolves conflicts among engagement models. Additionally, this organization maintains the tool suite that enables the CSIC. This organization also maintains standards and trains and certifies practitioners.

It should be cautioned that this organizational design will not be appropriate for everyone. Factors that led to this organizational design include: a great quantity of high-quality IC to be "mined" and shared; a large, geographically dispersed organization; many different participating groups, each with unique challenges; a requirement for specific, in-depth expertise; limited cross-organizational socialization or technical contact; a wide variety of different projects with differing project environments and contexts (outsourcing, in-house development, consulting, etc.). This design is a complex, expensive, large-scale solution that may not necessarily scale down to be economically viable for smaller organizations.

Although project managers (as project planners) are most intimately involved with the CSIC, there is a need to train *all* project participants, even those only involved with project execution. At a minimum, everyone needs to know how to access and understand the IC necessary to perform their tasks. Conceptually, it is important that everyone be aware of the "sharing" environment.

Does the IC "own" project management best practices? Ownership of project management best practices is a key organizational issue. The two extremes for establishing, sharing, and disseminating best practices have centered on the "ivory tower" (remote, centralized) approach and "the field" (hands-on, practical) approach. This dichotomy is especially relevant if there are several organizations or teams doing similar work in the field.

The ivory tower approach traditionally is having a central organization charged with the responsibility for all process-related functions. Staffed by a central team of experts and typically augmented by individuals brought in on temporary assignment from the field, this group establishes standard practices and processes and selects or builds tools and artifacts to help this standardization along. Years ago, they would have provided a large, three-ring binder filled to the brim with goodness. Today, that three-ring binder is likely electronic.

In contrast, the field approach recognizes that there is much goodness learned from practitioners doing actual work and that a small central activity works in conjunction with practitioners in the field to find, hone, and then disseminate goodness. Frequently there is an informal liaison between this central activity and the field. The results are often provided to other field activities as optional with minimal support, typically in "as is–where is" fashion.

The CSIC superstructure is the organizational framework for using CSIC in project management.

The field approach was established in part because of shortcomings in the ivory tower approach. Although tools and architecture were centralized, it differed by having process expertise distributed in the field; that is, the project and process SMEs in the field were responsible for virtually all (technical) content. Corporate centers of excellence such as those dealing with project management provided additional resources in support of this effort. Content (method) authoring was also "in the field" with formal documents of understanding establishing and regulating authoring efforts.

Again, the enhanced approach as implemented in the author's organization is a hybrid with a small core group that manages the central repository of contents, the method database. The group provides method architects and authoring support. Additionally, this group manages the supporting tool suite. There is limited conflict between the need of the field for control of content and the desire of the core group for organization-wide content uniformity.

This asset-based CSIC framework can be a most useful aid in institutionalizing best project planning and execution practices. The work-product-dominant approach strongly supports the definition of processes

and reuse. A robust authoring environment is needed to hold all of the pieces together.

Supporting tool suite. Here we briefly discuss the supporting tool suite. The tool suite stores a linked database of architectural components, in this instance the method database. It assists in the authoring process that in essence populates the method database. During project planning, the tool suite provides access to the method database and the capability to tailor an engagement model into an engagement template. During project execution, the tool provides active links to technique papers and to guidelines. Finally, the tool supports metrics and other experience harvesting.

Maintaining IC context is a key characteristic of the tool suite; that is, the various components are treated interdependently, and links among content are always maintained.

The benefits in how CSIC addresses key project issues

We might ask why is it that some projects *still* fail; moreover, we might ask how CSIC addresses the following issues:

1. Poor plans—The culprits here are often incomplete project plans or ill-fitting project plans. The root cause is frequently lack of suitable plans (IC) as a starting point for the planning process. Building a plan instance from IC that is not fully suitable may result in extraneous steps as well as omissions, such as failing to perform a certain testing activity. A second contributing factor is lack of a formal planning process by which plans are customized to the situation at hand.

   CSIC addresses this head on. It provides the IC, the tools for accessing the appropriate IC, and a formal process for building project plans. The planning process also involves all of the various roles (and constituencies) required to build a successful project team. This process not only develops the project plan but also provides a critical review of the plan and buy-in by project stakeholders.

2. Skill and execution issues—Project plans assume specific execution capabilities and skill levels. Specific issues include inability to perform tasks, resource estimation, and deviation from plan or unexpected outcomes. Even simple tasks can cause problems. With reference to the cookie recipe example, we might ask: How does one soften butter (Phase 1, Activity 2, Task 1)? Will the technique used impact the quality of the cookies? How much time should we allocate for this activity?

   CSIC addresses these issues by integrating skills and skill management into the overall organizational approach to project management. Additionally, it formally matches skill requirements with project roles.

3. Ambiguities and process errors—Looking again at the cookie recipe example, we see an artificially imposed sequence in Phase 1, Activity 1 of the work breakdown structure. There is no required sequence for the first three tasks (measuring various ingredients). Because there is no statement to the contrary, we may presume that the ingredients can be added to the small bowl in any sequence. In contrast, the tasks in Phase 1, Activity 4, are sequence-dependent. Performing them out of sequence may cause problems. This type of error may occur in project planning or in project execution.

   CSIC addresses these errors very well via detailed planning and supporting IC such as work product templates and technique papers. Skills training is an additional requisite component to addressing process errors.

4. Resource issues—Poor estimates and the inability to track resources against project progress comprise these issues.

   In addition to enabling detailed plans, CSIC addresses these issues through IC that is oriented toward project estimation and resource management that takes into account various skill levels.

5. Scalability and applicability—Failure of planning to accommodate significant variations in scale results in ill-suited plans that can lead to failure. Similarly, lack of applicability (or inappropriateness) of IC also contributes to project failure.

   CSIC addresses this issue to the extent that the IC properly reveals scalability and applicability issues and constraints. Additionally, the MAW process for building project plans focuses on this problem.

6. Leadership and communications—This issue is a catchall for many project issues.

   CSIC addresses this issue to the extent that clear project plans are easier to articulate. Additionally, the in-gathering of stakeholders as part of the MAW can be a useful precursor to team building and to establishing project leadership. Also, the CSIC clarity of roles enhances communications during project execution.

7. Unclear or changing requirements—If a customer wants walnuts in the chocolate chip cookies and the recipe is read as implying pecans, the project may fail. Clarity requires that all ambiguities be resolved in a timely fashion. In this case it would be most preferable to determine the type of nuts desired before purchasing (or costing) takes place. If this discrepancy is not noticed before Phase 1, Activity 4, Task 3 (stir in ingredients), the mixture may have to be scrapped and the process started again. Similarly, if the customer at this point asks for pecans instead of walnuts, it is necessary to have a change management process in place to evaluate this request.

CSIC addresses requirements management issues by establishing work products and processes that are specifically oriented toward requirements management and change management. This action helps reduce but not eliminate problems related to requirements and change.

In summary, CSIC *can* address many key project management issues. However, CSIC, even when properly implemented and appropriately managed, is not a panacea.

Additional observations

In this section, we discuss some additional observations about this approach.

Benefits of commonality. As noted above, when properly implemented, CSIC addresses many symptoms of project failure. A key additional benefit of this more consistent approach to project management comes into play at the program (multiproject) management level. A portfolio of projects based on this set of more consistent project plans and plan components is easier to manage. Project plans share a more consistent "shape," common work products, and a standardized approach (as driven by technique papers and guidelines). This commonality enables program managers to regain control of the project portfolio. Often it is an unmeasured bonus. Additionally, practitioners moving among projects have less relearning and transition issues—everyone uses the same terminology. As a result, the movement of personnel (such as specialists) among multiple projects is more efficient. There are related positive implications for training, testing, documentation, and other activities.

A discussion about managing in the multiproject (program) environment would entail more than is intended for the scope of this paper.

Usability of the approach. The approach is intuitive and works well. However, planners who are accustomed to a task-oriented (work breakdown structure) approach to project planning will need to reorient their thinking toward output (work product) dominance (similar to the issues that were encountered many years ago in moving to "object-oriented" approaches to systems development). As with any new approach with its own nomenclature and structures, there are transition issues.

The key to usability is the uniform quality of the various parts of the method database. This maxim applies especially to work products.

Conclusions

Context–specific intellectual capital provides significant benefits to those organizations that invest in building and maintaining the appropriate CSIC infrastructure and contents. The focus on work products and building a robust collection of work products that are useful for multiple, different projects is especially worthwhile. CSIC is clearly of value in the domain of software development and maintenance project planning and management, and likely has applicability to other process-oriented domains.

The establishment of an appropriate level of support and infrastructure (as well as tools) for CSIC is a significant organizational undertaking. Although there are no hard and fast guidelines, this approach is most likely cost effective for larger organizations that have a significant portfolio of systems development and maintenance projects.

Acknowledgments

This paper is based on the creativity and hard work of several IBM teams both past and present. It has

been fun working with them, and the author thanks them for their contributions to his understanding and appreciation of context-specific intellectual capital.

**Trademark or registered trademark of Societe des Produits Nestle S. A. Corporation, Switzerland.

Cited references

1. M. A. Mach and M. L. Owoc, "Validation as the Integral Part of a Knowledge Management Process," *2001 Informing Science Conference*, Krakow, Poland (June 19–22, 2001).
2. M. Lindvall, I. Rus, and S. S. Sinha, "Technology Support for Knowledge Management," *Proceedings of the 4th International Workshop on Learning Software Organizations* (August 6, 2002).
3. *What Is Knowledge Management*, from B. D. Newman, "An Open Discussion of Knowledge Management," the Knowledge Management Forum, at http://www.km-forum.org/what_is.htm (August 2002).
4. S. Schoenert, Powerpoint Presentation, University of Koblenz-Landau, German, available at fac.cgu.edu/~chatters/mgt516/slides/KM3.PPT.
5. P. Gongla and C. R. Rizzuto, "Evolving Communities of Practice: IBM Global Services Experience," *IBM Systems Journal* **40**, No. 4, 842–862 (2001).
6. V. R. Basilli and C. Seaman, "The Experience Factory Organization," *IEEE Software*, 30–31 (May/June 2002).

Accepted for publication April 18, 2003.

Carl A. Singer *IBM Advanced Business Institute, Route 9W, Palisades, New York 10964 (csinger@us.ibm.com).* Dr. Singer is currently a senior consulting faculty member developing and managing several executive-level courses. He has a B.S. in organizational science from Case Institute of Technology and did graduate work there in operations research. He has an M.S. in industrial and operations engineering from the University of Michigan and a Ph.D. in industrial administration from the Krannert School of Business at Purdue University. He is also a graduate of the U.S. Army Command and General Staff College and the U.S. Army War College. Dr. Singer joined IBM in 1995 to develop tools for formally leveraging project expertise as intellectual capital. His teams have developed several engagement models for software maintenance and for Rapid Custom Development, a software development methodology that can be used when time to market is critical. He serves as a member of the executive committee of the IEEE Software Engineering Standards Committee (SESC).