SCRUM Project Architecture and Thriving Systems Theory

Leslie J. Waguespack
Computer Information Systems Department, Bentley University
Waltham, Massachusetts 02154-4705, USA
lwaguespack@bentley.edu

William T. Schiano
wschiano@bentley.edu

Abstract

Agile project management continues to gain a widening and enthusiastic following. Agile methods can achieve a high level of satisfaction among all project stakeholders (users, customers, business managers, developers, and project managers) in terms of productivity, product quality, cost containment, time-to-market, and overall morale. Success with agile requires focus on requirements and design as a continuous discovery process, posing challenges for practitioners of more traditional project management both in terms of method adoption and sustained commitment. Thriving Systems Theory clarifies the appeal of agile project structure and processes, helps project teams determine and achieve the optimal portfolio of quality characteristics, and better articulate their value to all stakeholders. Thriving Systems Theory is an emerging framework of systems design quality that translates the research of design pattern patriarch Christopher Alexander on physical architecture design quality into the domain of systems engineering. The satisfaction achieved through agile methods is explained by Thriving Systems Theory’s fifteen choice properties of systems design quality. We demonstrate by identifying the manifestation of the choice properties in SCRUM, an exemplar of agile software project management.

1. Introduction

Agile project management approaches continue to grow in appeal and adoption because of their well-documented success in raising the satisfaction level among all project stakeholders (users, customers, business managers, developers, and project managers) in terms of productivity, product quality, cost containment, time-to-market, and overall morale. [6] Agile methodologies, SCRUM in particular, reduce the confounding affects of imprecise and/or evolving project requirements. The focus on the frequent delivery of incrementally complete working products to customers is core to the agile philosophy. [6] However, its focus on requirements and design as a continuous discovery process challenges practitioners of more traditional project management both in terms of method adoption and sustained commitment.

The satisfaction achieved through agile methods is explained by Thriving Systems Theory’s fifteen choice properties of systems design quality. [13] We demonstrate by identifying the choice properties in SCRUM, a prominent agile software development methodology. Among agile methodologies SCRUM exhibits a particular focus on project management’s role in success.

We begin with a brief introduction to Christopher Alexander’s theories of the nature of order and living structure. Alexander’s properties of design quality once mapped to systems architecture in Thriving Systems Theory provide a vocabulary translated from the physical space of material artifacts to the cognitive space of abstractions. We then survey the fifteen choice properties of design quality, reflecting on evidence of how each is perceived and may be achieved in project structure and behavior. Finally, we reflect on project management quality as extending beyond traditional quantitative metrics to include subjective and at times aesthetic qualitative choices as reflected in Thriving Systems Theory. This extended view can help project managers determine and achieve the optimal portfolio of quality characteristics and better articulate the value of project management to all stakeholders.

2. Architecture in buildings, systems, and projects

From the earliest literature on building design and civic planning, the process of managing the construction project has been intrinsic to the identity of the product. [10] Project and product coexist in symbiosis. Each represents a system of structure and behavior (albeit with somewhat different components): one focusing on the artifact, the other focusing on the building project. To achieve a satisfying product, it is also necessary to design a satisfying project. The relationship between project and product in software development is identical to the relationship that Christopher Alexander describes between the design process and physical architecture, the process of becoming. [2] Alexander’s work with patterns of building was a seminal influence on patterns of programming as recog-
organized by his selection for the OOPSLA’96 keynote. [1,9]

Architecture reflects a philosophy of quality, identifying observable characteristics highlighted and emphasized in design. Design choices unfold within an architectural philosophy (sometimes, unfortunately, a vacant one) whether designing physical artifacts or IT projects. Design is about shaping the alignment of relationships between and among stakeholder intentions and artifact features. IT project management is about shaping a process that aligns the relationships between and among stakeholder intentions and IT system features. The challenge is that the alignment must satisfy the present while accommodating the unfolding evolution of reality into the future. Projects must employ architectural strategies for both structural and behavioral adaptability within the constraints of cost/benefit among the requirements.

Projects and project management are about systems: a collection of choices, a matrix of interrelated forces of influence, the milieu of human perception and communication, and conditions that lead to or hinder sustainable, “living” artifacts. Every project choice emanates either directly from a stakeholder’s intention or from the design team’s attempt to satisfy the intention in the product artifact. The architecture of SCRUM leaves much room for such adaptation, not only in the flexible nature of the process, but also in how the process itself unfolds.

3. Thriving Systems Theory

Christopher Alexander is an oft-referenced catalyst for the concept of patterns in programming languages and design. [9] Although his central focus as an architect is on physical design in building and construction, his theory goes to the heart of the process of creating architectural systems that are effective and efficient. [2,3,4,5]

In The Nature of Order, Alexander explains the principles of what he names the living structure. He references chemistry, physics, astronomy, biology, art, and engineering. Alexander’s use of the term life is manifold, drawing on its characteristics of evolution and growth, on its characteristics of nurture and interdependency, and on its fragility. As he explains, “living structures are the result of a structure preserving process of becoming.” [2] Alexander defines “life” as it manifests in physical architecture, its perceivable characteristics and the stepwise transformations that make up any process that is capable of producing living structure.

Alexander’s definition of life that explains design quality is based on wholeness and centers. Wholeness is the palpable experience of resonance between an observer’s concept of order and their observation. A center is a focal point within an observer’s experience that either contributes to or detracts from wholeness. The contribution of each center to wholeness is further explained by the confluence of fifteen center properties each reflecting a distinct facet of order. Alexander’s conception of wholeness and centers in living structure is grounded in the geometry of space and its physical attributes of position and distance. Applying Alexander’s theory to the abstractions in systems in general (and information systems specifically) requires a transliteration of his fifteen properties of design quality. In [13] Alexander’s concepts of physical structure are translated from a language of physical space to a language a cognitive space where concepts and connotation correspond to physical position and distance in “fields” populated by abstractions rather than shapes. In the cognitive space the term choice serves well as the transliteration of Alexander’s term center.

It would be convenient to say that Alexander’s vocabulary can be directly applied without any interpretation. However, his writings are clearly fixed on the physical architecture of rooms, buildings, towns, and cities, focused on rehabilitating the practice of architecture in physical construction. (The reader should consult [13] for a more thorough discussion of the translation from Alexander’s property vocabulary of the physical to one for models and systems.) Nonetheless, the principles that Alexander develops to study life in buildings are applicable in the design choices of modeling, design, and implementation in systems engineering.

Every human-conceived system is an organized and integrated collection of choices, in physical, procedural, and/or conceptual domains. In project management, such choices include determining the partitioning of resources and responsibilities, task sequence and precedence, communication patterns, milestones, and quality control measures.

Traditionally these choices are taken at different times and reflect different priorities and reflect a conviction toward organizing activities that eventually result in a project plan. The architecture of a project guides the sequencing and substance of such choices.

These choices are exactly Alexander’s centers, the perceivable elements of relationship and behavior whose resonance in the observers’ experience of them results in an attitude ranging from hostility to profound satisfaction. In Alexander’s terms, the degree to which these choices contribute to the whole (system) determines to what degree the system has life.

Choices are the centers that lie at the root of life in systems. A choice by its nature admits to alternatives and the prospect of reconsideration when appropriate. Alexander uses the term unfolding to explain the evo-
lution of an architectural conception toward a useful intensification of life. In this sense, a living system unfolds revealing a continuity of structure and function, and a consonance with the context within which it is intended to serve.

The choice properties inform both the structure and the process of constructing systems or artifacts – the design decisions. Each design choice contributes to (or detracts from) the overall experience of observer satisfaction. In [13] Alexander’s fifteen center properties are transliterated as choice properties in systems. (See the first two columns of Table 1 below.)

Choice properties are evident in every choice with varying strength or influence that impact the resulting satisfaction. The confluence of property strengths results from an alignment of a designer’s choice with the aggregate intention of the stakeholders. The combination of all choices and their respective property strengths result in the overall perceived design quality. The choice properties reflect design characteristics long recognized in systems engineering as well as extensions that bridge beyond engineering to address aesthetics. The effectiveness of the choice property framework in evaluating the design quality of extant production systems is reported in [12].

Table 1 – Properties of Thriving Systems and the Scrum Architecture

<table>
<thead>
<tr>
<th>Thriving Systems Theory Property</th>
<th>Property Strengthening Design Action</th>
<th>SCRUN Architectural Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stepwise Refinement</td>
<td>Factor: express as a product of factors</td>
<td>Incremental unfolding of task structure based on dynamically evolving, product backlog priorities</td>
</tr>
<tr>
<td>Encapsulation</td>
<td>Encapsulate: encapsulate the essential features of something succinctly by a protective coating or membrane</td>
<td>“Frozen” sprint backlogs, SCRUM master and sprint autonomy</td>
</tr>
<tr>
<td>Extensibility</td>
<td>Extend: render something capable of expansion in scope, effect or meaning</td>
<td>Dynamic product backlog management informed by sprint outcomes, Sprint structure, Sprint team determines sprint product commitment</td>
</tr>
<tr>
<td>Correctness</td>
<td>Align: put (things) into correct or appropriate relative positions</td>
<td>Sprint completion focused on delivering a verified product</td>
</tr>
<tr>
<td></td>
<td><strong>Table 1 – Properties of Thriving Systems and the Scrum Architecture</strong></td>
<td></td>
</tr>
</tbody>
</table>

| Transparency                      | Expose: reveal the presence of (a quality or feeling) | **Dynamic published product backlog prioritized by customer/business value** |
| Composition of Function           | Assemble: fit together the separate component parts of (a machine or other object) | Dynamic published product backlog of upstream sprint products |
| Identity                          | Identify: establish or indicate who or what (someone or something) as | • Sprints aligned to product backlog |
|                                   | • Downstream sprint assembly of upstream sprint products | **SCRUM team composition aligned with product quality features** |
| Scale                             | Focus: (of a person or their eyes) adapt to the prevailing level of light [abstraction] and become able to see clearly | • Sprint size aligned to customer consumable product component |
|                                   | • Delivery-relevant specification detail | **Continuous, consistent application of backlog planning, maintenance & review integrating product owner vision and priorities** |
| User Friendliness                 | Accommodate: fit in with the wishes or needs of | Self-organizing, cross-functional teams |
| Patterns                          | Pattern: give a regular or intelligible form to | Continuous, consistent application of backlog planning, maintenance & review integrating product owner vision and priorities |
| Programmability                   | Generalize: make or become more widely or generally applicable | Sprint review, Self-organizing teams |
| Reliability                       | Normalize: make something more normal, which typically means conforming to some regularity or rule | Incremental “working” deliveries, Product owner/sprint team coordination |
| Elegance                          | Coordinate: bring the different elements of (a complex activity or organization) into a relationship that is efficient or harmonious | Concurrent nurture of sprint team autonomy and self-organization with overall project vision and priorities |

4. SCRUM: an exemplar of agile methodologies

SCRUM is an exemplar of agile software development approaches initially formalized in the 1990s and explained in some detail in [11]. We chose SCRUM because it is the most widely adopted agile methodology and reflects adoption in organizations. [14] SCRUM boasts project architecture based upon the philosophy of “assess and adapt,” grounded in agile development. [6] It designates three project roles: **product owner**, **SCRUM master**, and **team member**. There are three ceremonies: **sprint planning**, **sprint review**, and **daily SCRUM meetings**. The roles participate in the ceremo-
5. SCRUM: through the lens of choice properties of design quality

Although we assert that the properties of Thriving Systems Theory are readily discernible in all agile methodologies, we choose to examine one in particular for clarity. This section examines how each of the fifteen properties of choice design quality is expressed in SCRUM’s architecture. Understanding the properties will help all those involved in an agile development to better manage the process and articulate the value. In the following enumeration of choice properties the discussion of one property often includes references to others. This is natural because the design quality properties do not occur in isolation but rather in confluence. Examples of SCRUM elements that reflect the design quality properties are summarized in the third column of Table 1 above.

5.1. Stepwise Refinement

As the name implies, it is an approach that presumes a problem should be addressed in stages through successive model elaboration. The stages may represent degrees of detail or an expanding scope of problem coverage. In either case, the goal of stepwise refinement is to demonstrate a cogent and complete representation of the solution at whatever level of detail or scope appropriate to this stage. This is evidenced in the SCRUM architecture, where sprints subdivide development and focus effort on successively elaborated work products. [11]

*Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.* [6]

To achieve this, the project management paradigm must support abstraction that allows generalization of the scope of interest and then the elaboration of that scope from one stage to the next. Phases and tasks are the abstraction of stages or levels of detail in projects. The number and nesting of phases are influenced by degrees of complexity and often by risk-specific methods or techniques closely connected to analysis, design, and/or implementation technologies. Within phases, constituent work may further elaborate the steps at progressively finer granules. In the simplest terms stepwise refinement leads to stepping stones that all stakeholders can recognize spaced conveniently apart, laying out a path(s) through a graph of tasks / decisions needed to reach artifact delivery.

5.2. Cohesion

This property reflects a consistency of responsibility distribution in a field of system components. Since every project task expects the tasks around it to fulfill their responsibilities to contribute to the whole, each task is in itself free to be single-minded in its focus on its own purpose. This is the result of a well-chosen work breakdown structure, or in SCRUM, a product backlog and sprint backlog refinement. [11] Each sprint targets a working deliverable. This independent sufficiency accentuates the divisibility of responsibility both in terms of each task’s individual purpose, its identity, and in terms of the clarity with which its purpose is exposed to the rest of the community of tasks in the project, transparency. The single-mindedness that results also increases the feasibility of task rearrangement, enabling a change in workflow to take advantage of parallelism within resource management while almost every task’s individual purpose remains fixed. The independent sufficiency of each task’s inner work-
ings couples with the project-wide interdependency of task cooperation to promote a texture exhibiting a sense of system-wide connectedness, elegance.

Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale. [6]

At the highest level of project granularity the designation of a single “keeper of the vision,” the product owner, unifies the intentions of the entire stakeholder community in a single expression of project purpose.

5.3. Encapsulation

This property formally separates the inner workings to complete a task while defining its relationships to other tasks. It partners with cohesion to provide task separation and maximize independence among tasks. The practice of “freezing” sprint backlog content at sprint planning compartmentalizes the work package, ensuring a protected and uninterrupted effort toward sprint completion (and the team’s freedom to self-organize further separates elements of project structure and practice).

By carefully identifying the flow of work products to/from the sprints the connections, interfaces of work packages, are clearly delineated. This information is critical to task-driven resource management techniques. The boundary defined by interface-dependent vs. interface-independent task elements uncouples the detailed inner workings of tasks from the external connections. All interface-independent task activities can be determined (and modified) without impacting multi-task workflows or connectivity.

Enclose the essential features of something succinctly by a protective coating or membrane. (The modeling action described in [13] strengthening encapsulation.)

Tasks that are clearly encapsulated offer opportunities for interchangeability and reuse across projects. Within software development, version control software enables projects to manage and maintain encapsulation and identity of code.

5.4. Extensibility

This property may be the most important in pursuing systems with life. It is the vehicle for seamless unfolding in project evolution, adaptation to and absorption of change.

Welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage. [6]

In project management it is pursued through the coordinated use of encapsulation, modularization, and programmability. The overall discipline of separation of concerns through these design quality properties minimizes the inertia that builds up during a project with the accumulation of work products. When separation is neglected, changes in requirements or priorities usually result in a cascade of rework. The essence of agile development is the graceful absorption of change due either to new stakeholder requirements or to the identification of emerging development opportunities as similarities and relationships among work products are recognized. To the extent that tasks are isolated, rework can generally be relegated to task interfaces. This reduces the cost risk relating to changed or expanded requirements (or opportunities) that arise in the project’s course. In SCRUM, openness to evolving priorities or requirement discovery is systematized with sprint planning, daily SCRUM meetings, and sprint review meetings. Agile encourages what had been seen at times in earlier methodologies as scope creep.

5.5. Modularization

Along with cohesion this property enables “divide and conquer” problem solving (divide: decompose product backlog tasks into sprint backlogs, and conquer: iterate decomposition as needed to crystallize required effort estimates and facilitate certifiable product deliverables). It is augmented by the flexibility of configuring and rearranging teams as cooperating agents.

Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done. [6]

Individual teams have clear compartmentalized goals and autonomy while the published product backlog orients their efforts toward the accomplishment of the whole. Modularization also supports scale permitting the composition of sprints of varying scope that hold details in abeyance until they require focus, as in product backlog features separated into sprint backlog tasks. [11] Enlightened task design exposes the solution structure envisioned in the project architecture and publishes intentions for further extensibility by separation of concerns and isolation of risk. The granularity enabled through modularization may be applied to facilitate the product owner’s formulation of structure,
5.6. Correctness

This property in project management depends heavily upon two outcomes: 1) the clarity and fidelity of the represented understanding of system goals and characteristics, validation and 2) the completeness and effectiveness of product feature testing both individually and in composition, verification.

In SCRUM, validation proceeds from the systematic and repeated work assignment, review, and reprioritization of product backlog items; that through the stages of stepwise refinement, the sprints, the essential aspects of system characteristics are brought forward maintaining their integrity (as in Fred Brooks’ essence and accidents [7,8]). Modularization aids in cataloging and focusing on individual essential characteristics. Choices in the earliest stages of unfolding (as in the product owner’s articulation of vision in product backlog) will frame the unfolding work packages that follow. To the extent that each sprint produces correct results, correctness is cumulative. Correctness is the only choice property that supports itself. Correctness must be a priority at each stage; correctness shortcomings grow progressively more expensive to rehabilitate as evolution progresses – notice “rehabilitate,” to restore to normal life.

Continuous attention to technical excellence and good design enhances agility. [6]

Verification depends on the effective testability of each choice to certify it as “consistent with stakeholder understanding.” The open invitation to participate in sprint review meetings both enables and encourages continuous verification of alignment between work products and stakeholder expectations. Modularization aids in the verification of individual choices or tasks and then relying on the correctness of individual task products verification can progress to the certification of behaviors resulting from composition of function. Well-chosen tasks yield reliable quality control points where overall correctness can be incrementally evaluated. A clear relationship of ownership between developer team and task product transparently aligns quality responsibility onto team structure. Experience often leads to dependable patterns of tasks and task assignments that can be adapted and applied to related project requirements. Verification in these situations can focus on known areas of fragility/risk and thus limit the effort required to reach a desired level of reliability.

5.7. Transparency

The key here is evident structure, revealing how things fit and work together. In project management “fit together” and “work together” are defined by the workflow of task products along the graph of task precedence and through the project team structure. Individual tasks may represent clearly delineated and encapsulated choices, but their cooperation and cumulative progress are defined by relationships.

Business people and developers must work together daily throughout the project. [6]

The consistency of publishing up-to-date project priorities through the product backlog facilitates everyone being on the same page. Precedence explains the structural relationship of tasks through the propagation of task products through their interfaces. Task precedence defines the feasible unfoldings of the product’s “becoming” and the breadth of alternative paths opportunities – opportunities that enable and encourage stakeholders to review their requirements and priorities continuously.

In SCRUM the task precedence is dynamically determined in the sprint planning, sprint review, and daily SCRUM meetings. These project elements equally expose the goals, tasks, and demonstrated progress of the project for all the stakeholders to see.

5.8. Composition of Function

As a fundamental tool for managing complexity, humans regularly attempt to decompose problems, issues, or tasks into parts that in themselves are either sufficiently simple to permit direct solution or can through some iterative subdivision can become sufficiently simple. This is a defining aspect of modularization. But, when the conception of the parts also anticipates reuse or rearrangement, then the part takes on a larger significance. The combination of specifying a choice consistent with the essence of system characteristics and then designing the choice as an interchangeable component in multiple super-ordinate choices is a step toward elegance. Design with the ability to reuse or rearrange choices represents an understanding of the essence of the project at a deeper level than an individual task or requirement. It represents awareness of the intention, perhaps even the philosophy, of the project domain. In a sense composition of function is the dual of “divide-and-conquer,” it is “constitute-and-compose.”
5.9. Identity

This property at its root is recognition. In the physical world identity is literal based upon direct sensorimotor experience: by sight or touch and in some cases by sound or smell – a human experience of the “real” world. When one can name something – that gives it focus; it becomes tractable; the name can take the place of the concept in communication. When accepted by a community the name and its concept contribute to the culture.

In the abstract world of ideas, models, and language identity becomes more complex. In the abstract domain identity is based on definition. If that definition is not made explicit the inevitable result will be confusion: coexisting, inconsistent understanding. The essence of identity as a property of design quality is the promotion of shared consistent understanding. With that in place, communication sustains coordination rather than confounding it. The consistent publication of product backlog and sprint backlog items contributes to a shared conception of project identity.

In SCRUM, the clear definition of roles and the distribution of responsibility contribute to the identity of the players. This accentuates the contribution potential of every individual whether they can provide a “solution” themselves or whether they act as a sentinel for unresolved questions they are empowered to contribute because of the role structure.

5.10. Scale

A couple common idioms reflect this property well: “You can’t see the forest for the trees.” and “Let’s get a view from 10,000 feet.” They reflect the importance of context in recognition and decision-making. Alexander’s focus on the whole composed of a field of strong centers is reflected in the choice property of scale. Scale captures the quality imperative that all choices must be kept in perspective because the connectedness of a choice to the whole is essential.

*At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.* [6]

Tasks in the sprint backlog combine to fulfill features in the product backlog that combine to yield the final product. [10] Coupled with *stepwise refinement* as it is, *scale* is used to focus team and stakeholder attention to achieve the contextual understanding needed to address constituent concerns within the whole. PERT, CPM, and WBS are examples of project management tools used to zoom in and zoom out of the detail in large projects, allowing the observer the option of breadth or depth in the observable detail as the need arises.

5.11. User Friendliness

This property’s impact may be easiest to consider in its absence. A design choice that is unfriendly to stakeholders is confusing, hard to comprehend, unwieldy, and perhaps worst of all, of indeterminate correctness. That which defies understanding fails to exhibit life or produce stakeholder-satisfying designs. Satisfaction is cumulative. Sensitivity to the stakeholders’ conceptions of essence in the system is key to Alexander’s life-preserving process of unfolding.

Authoring project plans where tasks correspond almost one-to-one with the real-world concepts and entities transparently evident to stakeholders and team members will be intrinsically easier to understand and interact with. Transparently incorporating the stakeholders’ experience exhibits a fundamentally friendly quality. It respects the stakeholders’ perceptions and it welcomes them into the processes of verification and validation that are intrinsic to correctness. The unified structure of what a task consumes and what a task produces correlates so naturally with observers of business models and process models that natural clarity in that communication improves understanding and diminishes the proclivity for mistakes in communication and/or implementation.

5.12. Patterns

Templates for communication and process are essential to economy of effort in any project. They provide consistency that accelerates assimilation of information resources and the acclimation of personnel. Experience is one of the more important resources. Any project of significant size may benefit from the consultative and organizing role of a project management office. SCRUM masters provide the consistency and continuity of process application that ensures interoperable work products and work processes. Process templates provide standard questions designed to promote the discovery of important information, checklists to focus analysis on areas of historical risk where teams have overlooked issues or failed to recognize problems in the past. *Patterns* institutionalize organizational experience to benefit even the least experienced team members and raise the overall quality of team performance. Recognition of patterns’ role in project quality explains the motivation for project management offices to promote project standards for documentation, metrics, and procedures. The work breakdown structure hierarchy by itself may be the best example of applying patterns to projects.
5.13. Programmability

This property is closely allied with extensibility above and addresses the need for projects to be flexible to welcome the future. Sprint planning provides a standing opportunity for extensibility as product backlogs are parceled out for attention. As with extensibility, successful accommodation of change relies on an understanding of the fundamental options governing the structure and behavior within a particular domain. Programmability is epitomized by the self-organizing prerogative of the individual team. A team in a sprint is encapsulated and completely free to react on a daily basis to internally discovered obstacles or opportunities and to act creatively without external permission or collaboration.

What sets programmability apart from extensibility is a facility that permits altering the project’s behavior without having to reconstruct tasks (or in the case of SCRUM reconstruct sprints) – the project’s behavior can be sensitive to the context determined by a team in real time. This versatility is not accidental but planned. Programmability is a hallmark of agile development methods. [6]

5.14. Reliability

Project reliability issues from project patterns with frequent and well-positioned review practices. The consistency of “light on the subject” from every stakeholder vantage point encourages concern and diligence for correctness – essential to project success. When quality control is integrated as in sprints, cost does not explode exponentially when defects are discovered belatedly. Careful modularization and encapsulation of tasks provide the means of stepwise quality assessment at the task level where discrepancies can be detected early before their effects are propagated onto one or more downstream tasks where the cost of rehabilitation will multiply rapidly.

Business people and developers must work together daily throughout the project. [6]

Reliability cannot ignore intention. Reliability balances on a thin line between flexibility and austerity: a flexibility that enables a graceful, cost-effective assimilation of change and an austerity that resists work products or features not essential to the stakeholders’ intentions.

Simplicity—the art of maximizing the amount of work not done—is essential. [6]

5.15. Elegance

“Pleasing grace and style in appearance or manner” that’s how the dictionary expresses the meaning of elegance. (Oxford English Dictionary) Project methodologies promoting behaviors that are consistent, clear, concise, coherent, cogent, and transparently correct exude elegance and nurture cooperation, constructive criticism, and stakeholder community confidence. These are projects and plans that confess to their own shortcomings because their clarity obscures nothing, even omissions. These are projects that satisfy stakeholders. They appear intuitively obvious. The clarity of their composite structure is so self-evident that they seem simple.

Fred Brooks, in defining essence and accidents of software development, crystallized the distinction between that which the stakeholders need and that which must be done to achieve it. [7] At their core, agile methods focus on eliminating unnecessary overhead in determining and accomplishing project goals and tasks. Elegance largely proceeds from the efficient and effective representation of essential project behaviors: beneficial, integrated, mutually supporting as Alexander describes in systems with living structure. [2]

6. Choice properties: aesthetics of design quality in project management

The choice properties of design quality cast a definitely aesthetic flavor over a traditional software engineering perspective of product and project quality. In all engineering, there is a healthy desire for quantification, for clear discrimination, and for objectivity. But that objectivity consistently teeters on the understanding of stakeholders’ intentions in the requirements and policies that motivate the system building activities we attempt to shape through project management. Whereas metrics can provide consistently objective quantifications, they do not always clearly represent satisfaction, as the numbers do not consistently reflect the entirety of the stakeholder context. Perhaps those unsatisfying projects of the past using agile methodologies focused too heavily on the form of the agile method rather than on the quality properties that the method was conceived to express.

The impetus in agile methods for continuous engagement between stakeholders and developers, and the frequent, iterative “product” delivery (both hallmarks of SCRUM) is a tacit acknowledgement of the importance of aesthetics in project management. It raises trust in the project team’s ability to assess and adapt above any rubrics of methodology and formal-
ism. It relies on the individuals and the team to be professionals.

In the end, an assessment of quality in project management and the product depend in no small degree upon the stakeholders’ perception of beauty in both. The choice properties of design quality defined in Thriving Systems Theory evince a comprehensive explanation of stakeholder satisfaction with SCRUM in particular and agile methods in general. Thriving Systems Theory applied to IT project management offers:

- A broader perspective on project and product quality for the project architect,
- An opportunity to channel project energy toward particular design quality properties of immediate concern,
- A justification to resist excessive reliance on objective metrics that may succeed in yielding efficient projects, but often fail to deliver highly effective products, and
- An explanation of project and product quality applicable to cost/benefit assessment that allows stakeholders to coalesce for business decision-making.

Thriving Systems Theory asserts that a desirable balance of strength among the fifteen properties of design quality leads to a Thriving System. We believe that applying Theory Systems Theory to project management leads to a Thriving Project.

7. Acknowledgements

We are thankful for the insightful referee comments. Special thanks are due our colleagues at Bentley University for their critical discussions and comments on these ideas.

8. References


[9] Coplien, J., Introduction of keynote speech by Christopher Alexander. At ACM Conference on Object-Oriented Programs, Systems, Languages and Applications (OOPSLA), (San Jose, CA, USA), October 1996.


