Fundamental Principles for Agile Systems Engineering

Rick Dove

Paradigm Shift International and Agile Security Forum

Box 289, Questa, New Mexico 87556, USA

dove@parshift.com

Abstract

Agile software development processes of various kinds have been proposed and put into practice. Generally they are specific-approach oriented at the operational-characteristic level. Ten underlying principles for agile systems were identified by examining over a hundred various non-software systems that exhibit agility. This paper outlines and discusses the first purposeful employment of these principles in an enterprise IT architecture design and implementation project at Silterra, a semi-conductor foundry. Results suggest that employment of these principals can increase the predictability of major software projects and the responsiveness to change.

Introduction

A body of knowledge characterizing the fundamental nature of agile systems was developed in the nineties, initially at the Agility Forum (Dove 1996), and in subsequent research, then generalized and reported in *Response Ability: The Language, Structure, and Culture of Agile Enterprise* (Dove 2001). This body of knowledge was primarily developed in industry-collaborative research that examined hundreds of systems of many kinds which exhibited agile response capabilities. Ten underlying design principles became evident in this research, as did eight general categories of reactive and proactive situational-response. As the collaborative research progressed, the eight response situations evolved into a requirements framework, and the ten design principles evolved into a solution-design framework. Both were applied during the research activity, which spanned some eight years, to affirm and evolve their structure and components.

The research phase for these principles transitioned into proof of concept activity at Silterra, a semi-conductor foundry start-up in Malaysia. Cy Hannon, the CEO of this $1.5 billon greenfield opportunity, put the development of an *agile* enterprise-IT infrastructure among the top priorities.

A principle concern was to break the rigid lock that ERP traditionally imposed on evolving business process and business strategy. He wanted to leverage the possibilities of the Internet and web-browser information access, build unprecedented transparency into the heart of the operating philosophy and culture, and facilitate an aggressive growth strategy with rapid integration of new-plants, acquisitions, and outsourced semi-conductor packaging firms.

The impact of the Internet on business models and strategy was in its infancy, and was expected to undergo rapid and continuous change - the infrastructure had to accommodate whatever might emerge as advantageous, rather than impede sustainable leadership. I was brought in as initial CIO/CSO to design and manage the implementation of these projects.

This paper outlines the frameworks of agile-systems principles and requirements

development that emerged from the earlier mentioned research, and shows how these frameworks guided both the infrastructure design and its implementation process at Silterra. Values of the approach are expressed in actual project-performance and subsequent operational metrics. Important lessons-learned will be discussed, that have since expanded the frameworks proposed for agile systems development from an original two to a current six (Dove 2005b).

A major application for real-time operational transparency at Silterra, known as MyFab, was also developed with these same principles; as was an unprecedented agile-security strategy. The agile-security strategy, unimplemented at the time my interim appointment ended, was necessitated by the vulnerability and value exposed through extreme transparency, the growing agility of the attack community, and the ethical norms of the region. Both MyFab and the agile-security strategy are beyond the scope of this paper, but subsequent evolution of the principle-based agile-security strategy are discussed in (Dove 2005a).

The fundamental principles discussed in this paper are applicable to any would-be agile-system, but the examples will focus on enterprise IT-infrastructure and, notably, its implementation process, which employed the same principles.

Requirements development for agile enterprise-IT support are discussed in (Börjesson 2004) addressing software improvement practices. Architectural principles for adaptable IT infrastructure have received a lot of attention in recent years. Notable are the service-oriented-architecture (SOA) (He 2003), and many academic pursuits referenced and linked in (University of Massachusetts at Dartmouth 2001). The principles exposed and suggested here have some overlap with these other architecture-focused principles - but differ in their genesis and nature in that that were developed by analyzing systems which exhibit agile capabilities, rather than synthesized with reasoned intent.

Systems Agility

It is necessary to delimit my focus by defining *agile* as I apply it to systems and systems engineering. The goal of agility is not rightfully a goal of all systems engineering.

Agile systems, as I define them, are concerned with *response ability* - for both reactive and proactive response needs and opportunities - when these are unpredictable, uncertain, and likely to change.

Four proactive and four reactive response categories emerged as a framework from the previously mentioned analysis research (Dove 2001, Chapter Three). Proactive response might be the development of a system to meet a new need; continual improvement or upgrade to increasing response-performance needs; migration to a better approach as new possibilities, knowledge or requirements emerge; or modification of a system by inserting subsystems with new capabilities. Reactive response might be the correction of a malfunctioning subsystem or unintended consequence; the response to input or response-need variation; a response to increased or decreased capacity needs, or reconfiguration of subsystem relationships. See Table 1 for a synopsis.

Response ability metrics that emerged from the research went beyond time, to include cost, quality, and scope (Dove 2001, Chapter Three). Response time and response cost are self evident. Response quality refers to the predictability and robustness of a response - predictability means not only on time and on budget, but also on spec; and robustness means when a response is activated, it is at least sufficient rather than an interim expediency. Scope refers to the range of response possibility, and is the glaring demarcation between agile and flexible systems - for agile systems, readily

reconfigured, can accommodate response needs outside of an embedded set of options.

At this point we have established a general framework of four proactive and four reactive response categories, and a framework of four response metrics. Both are instrumental tools for developing agile-system requirements from problem analysis, as well as for analyzing existing systems for agile capabilities.

All frameworks discussed here are structured categories for channeling thought. In their development, the principle of *parsimony* has pushed toward fewer categories, while the principle of *requisite variety* pushed toward sufficient categories. Experience with analysis (in principles extraction) and application (in design) have not shown them wanting.

Silterra Enterprise IT-Infrastructure

Requirements Development - Table 1 is a short synopsis of dynamic response issues that drove the subsequent design. Each issue is qualified with the response metrics felt most important at the time. The problem analysis exhibited in Table 1 was developed before a general design concept was developed.

An RFI was released initially to appropriate vendors for major elements of the general design concept: enterprise bus, data base, and ERP applications. Subsequent applications for planning, time accounting, web-accessible operations transparency, strategic-project portfolio management, and others were added later.

The RFI content succinctly spelled out the vision of the company through time, its consequent needs for responding effectively to new strategies and business models, new technologies for infrastructure and applications, business-net interconnects, and, importantly, vendor-independent ERP functional modules on an application-by-application basis. The content of that RFI is contained in Chapter 8 of (Dove 2001) and reproduced in the appendix to this paper. It is the textural discussion and rationale behind the synopsis in Table 1.

Table 1: Silterra Infrastructure Response Issues (circa late 1999)

Proactive Dynamics
Creation/Elimination - Create something new or eliminate something that exists. ▪ Creating new customer/supplier/partner business net-link [t,q,s] ▪ Creating acquisition business net-link [t,q,s] ▪ Creating interface to a new application [t,c,s]
Improvement - Incremental improvement. ▪ Improvement of interface performance [t,s]
Migration - Foreseen, eventual, and fundamental change. ▪ Migration to NT and COM/DCOM [c,q]
Modification - Addition or subtraction of unique capability. ▪ Addition of new foundry facility [q,s] ▪ Addition of new customer/supplier/partner data interface [t,s] ▪ Addition of new industry data-standard [t,s] ▪ Replacing the bus vendor [c,t,s]
Reactive Dynamics
Correction - Rectify a dysfunction. ▪ Fixing an interface bug that surfaces later in time (original engineer gone) [t,q]
Variation - Real-time operating change within the mission. ▪ Quality of data from production MES system [t] ▪ Variation in competency/availability of infrastructure operating personnel [t,s] ▪ Variation in real-time on-line availability of applications [t,s].
Expansion/Contraction - Increase or decrease of existing capacity. ▪ Increase the number of interfaced applications and business net-links [s]
Reconfiguration - Reorganize resource or process relationships. ▪ Reconfiguration of an interface for an application upgrade/change [t,c,q,s]
Notes
Metric focus is shown in brackets [c,t,q,s] t = response time c = response cost q = response quality s = response scope

Design Principles - From the research mentioned earlier emerged ten fundamental design principles evident in systems that exhibited agility. These principles are based

on reusable modules reconfigurable in a scalable framework. Figure 1 depicts a graphical representation of the technical elements of Silterra's infrastructure design, and Table 2 the specific application of the principles which includes both the technical and peopled elements.

The enterprise IT infrastructure is a peopled-system, in that it includes active roles for business system administrators (BSAs) and system strategy administrators (SSAs), as well as the IT technology elements.

The basic design concept employed an enterprise message-bus as the sole means of application interconnect - across the entire extended enterprise, and including communication between individual ERP applications. This ruled out the typical SAP or Oracle integrated approach of that time. Though direct through-the-bus application-to-application interface was enabled, the preferred mode was publish-subscribe, as this allowed asynchronous communication that was fault tolerant if any application servers were off-line for any reason.

Basic strategy and rules included:

- A BSA group, with individual members responsible for the configuration and evolution of the IT applications supporting business processes employed by designated departments; and for assisting the department managers, who had departmental IT-budget control, in selection and acquisition. BSAs have free access to all users and managers, employing the principle of flat interaction, and are *encapsulated modules* within the Infrastructure System.
- An SSA group, composed of IT-savvy managers, had responsibility for the evolution of the infrastructure technical framework, and for enforcing zero-deviation-tolerance on application-module encapsulation and bus-only intercommunication. SSAs are *encapsulated modules* of the system, with authority and the responsibility for satisfying objectives.
- The mandatory use of applications as provided from the vendor, without custom modification, in conformance to the principles of *facilitated reuse* and *plug compatibility*.
- The mandatory collaboration with users on response-dynamics requirements-analysis before solutions or evolutions were considered, employing the principles of *distributed control and information*, and *redundancy and diversity*.
- The configuration of applications, management of integration projects, and infrastructure architecture are internal responsibilities, and not to be outsourced. This rule virtually employs all of the principles.

The principle of *encapsulated modules* is evident in the BSA and SSA responsibilities and authorities. All technology applications are modularized in their bus-interaction requirements. Integrated ERP structures are disallowed, permitting mixed-vendor ERP applications connected to the bus through application-program-interfaces (APIs), extract-transform-load (ETL) modules, and the bus-interface modules. The bus itself is an encapsulated module, by virtue of the standardized bus-interface module (BIM) that sits between the bus and all applications.

The object-oriented concept of inheritance is not a part of the encapsulated module principle. It wasn't observed in any of the many cases analyzed during research, and it appears to have some conflict with the principles of *encapsulated module* self-sufficiency, *distributed control and information* and *flat interaction.*

It should be noted that the BIM and ETLs were home grown by necessity. Everything else was off the shelf.

ETL's were implemented from an evolving and reusable standardized template (fractal principles-based sub-system). This reduced the competency required for a specific ETL development, and enabled rapid development

- employing *deferred commitment* until the application and its interface needs were stable. The template approach also facilitated maintenance and update response, especially in cases where the original developer was unavailable.

Table 2 - Fundamental Principles and Their Application in Silterra's IT Infrastructure

Reusable
Encapsulated Modularity (Self-Contained Units) - Modules are distinct, separable, self-sufficient units cooperating toward a shared common purpose. ▪ Applications, data bases, ETLs, bus-interface modules (BIMs), bus, BSAs, SSAs **Plug Compatibility** - Modules share defined interaction and interface standards; and are easily inserted or removed. ▪ XML, message-data definitions, BIM spec, ETL-interface spec, rule on COTS **Facilitated Reuse** - Modules are reusable/replicable; and responsibilities for ready re-use/replication and for management, maintenance, and upgrade of module inventory is specifically designated. ▪ BSA group, business process maps, ETL templates, mandatory rule on COTS
Reconfigurable
Flat Interaction - Modules communicate directly on a peer-to-peer relationship; and parallel rather than sequential relationships are favored. ▪ Direct app-to-app dialog, BSA group user/management access and team collaboration **Deferred Commitment** - Module relationships are transient when possible; decisions and fixed bindings are postponed until immediately necessary; and relationships are scheduled and bound in real-time. ▪ Publish subscribe asynchronicity, ETL created after app is stable, rule that response-requirements be developed before solutions considered **Distributed Control and Information** - Modules are directed by objective rather than method; decisions are made at point of maximum knowledge; information is associated locally, accessible globally, and freely disseminated. ▪ Separate apps and data bases at each physical location, BSA independence and team collaboration, SSA/BSA separation, rule on mandatory user collaboration **Self-Organization** - Module relationships are self-determined; and module interaction is self-adjusting or negotiated. ▪ BSA autonomy, BSA teaming, SSA autonomous control, publish-subscribe options to pull information as needed
Scalable
Evolving Standards (Framework) - Frameworks standardize inter-module communication and interaction; define module compatibility; and have responsibilities designated for evolution and compatibility. ▪ SSA group, XML, message data definitions, ETL-interface specs, ETL template spec, BMI spec **Redundancy and Diversity** - Duplicate modules are employed to provide capacity right-sizing options and fail-soft tolerance; and diversity among similar modules employing different methods is exploited. ▪ Multiple app versions, multiple bus paths, replicated apps at each physical locations, ERP multiple-vendor apps, , rule on mandatory user collaboration, cross-trained responsibilities for BSA departmental responsibilities **Elastic Capacity** - Module populations may be increased and decreased widely within the existing framework. ▪ Virtually unlimited bus extension and capacity with compartmented parallelism
Notes
ETL=extract/transform/load, BSA=business systems analyst, SSA=software systems analyst, BIM=bus interface module, COTS=common off the shelf

Plug compatibility coupled with *encapsulated modularity* are obvious principles today, yet often the source of violation pressure. Rather than separate a module and develop an interface to the bus, more than once a strong campaign was mounted at Silterra to let a suite of applications remain separately integrated, with a single bus connection to the suite. In all cases the argument was eventually won for principle-adherence - primarily on evidence that other applications were likely to want interaction with individual members of the suite, and when they did, development

response time would be a critical factor. And of course, allowing one violation sets a precedent that leads to many.

Facilitated reuse has accent on *facilitated*. Agile systems rely on system assembly from modules that are inventory managed with designated responsibility. Module maintenance, configuration-for-use, and system assembly also has formal designated responsibility under this principle. Enabling effective *response ability* means having the system assembly elements ready to go when needed. The BSAs have principle responsibilities here, with technical assistance from the IT staff as needed. The rule on unmodified vendor applications (COTS) ensures that system upgrades are unimpeded, and reusable as is.

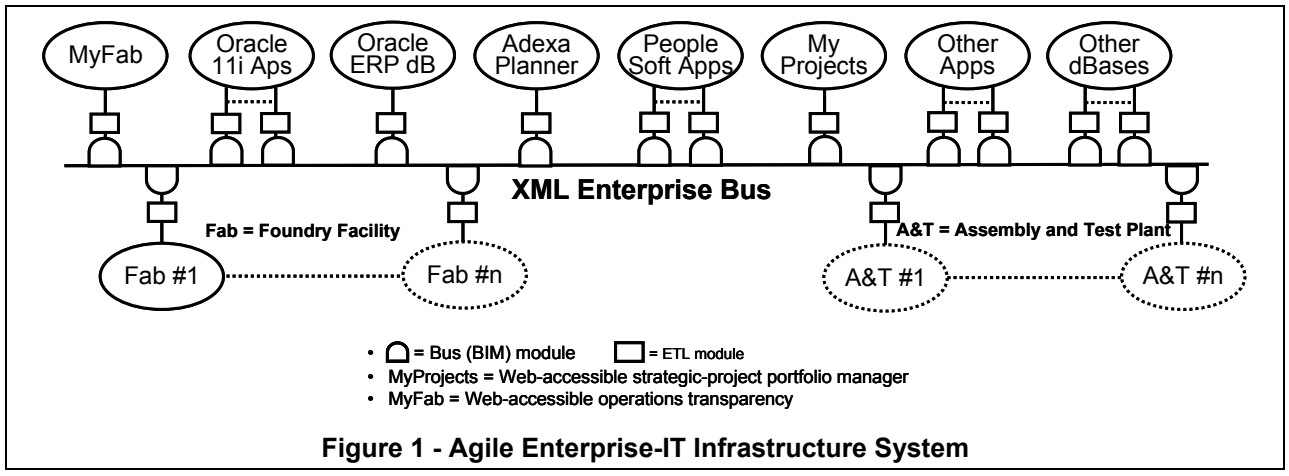

Figure 1 - Agile Enterprise-IT Infrastructure System

Flat interaction removes hierarchy and gate-keeper impediments to rapid information access among the people involved in requirements development and decision making. BSAs have free access to anyone with necessary information and decision-making responsibility, and collaborate as a team to minimize unintended consequences. Among application modules, flat interaction connects information clients with information servers directly (through the bus), without intervening processing and cascaded authorization verification.

Deferred commitment is a principle that recognizes change is constant, postponing actions to a just-in-time schedule to eliminate unnecessary rework and irreversible actions that result in scrap. The principle actions involved with infrastructure extension are the development of new or modified ETLs to accommodate new or modified applications. This principle led to the standardized template approach. which speeds up ETL work and permits postponement until the application interface is stable. It also is behind the rule that response-requirements be developed before solutions are considered. The preference for publish-subscribe application messaging employs this principle to effectively accommodate applications temporarily unavailable for whatever reason.

Distributed control and information recognizes that centralized data and information often impedes operations when access is interrupted, and people at point-of-application are generally more knowledgeable. A key issue at Silterra was the Adexa planning application, which encouraged the option for planning multiple plants from a central location. Not an uncommon practice in the semiconductor industry, but one that would put critical operations at other plants at the mercy of functioning communication lines and centralized application availability. This

would violate the principle of encapsulated modularity at a plant level, and impact the real-time any-time planning requirements established at Silterra. This concept was extended to all critical plant operation data bases and applications (during requirements and initial implementation development, multiple plants did not exist). BSA group members developed and maintained business process maps and ERP application configurations for their designated departments, with sole authority for change, but free viewing by anyone with access privileges. The SSA group had autonomous control responsibility for infrastructure framework standards - and the control responsibilities of BSA and SSA groups were sharply demarcated. The rule on user-collaboration honored distributed and localized knowledge among users.

Self organization is a principle more applicable to decision making elements than procedure followers - and readily observed in the peopled elements of the Infrastructure System. BSAs are collaborative self-organizers to preclude unintended consequences as new business process and upgrades are contemplated. SSAs have sole responsibility for the timing and nature of framework evolution, and self organize responsibility for planning, management and action as response situations occur. The publish-subscribe nature of the message bus allows individual applications to pull the information they need from other applications and data bases as and when needed, without centralized distribution control and push.

Pressure to relax these strategic rules and principles occurred frequently in the interest of expediency, and on occasion became major issues. But rules ruled in the end. That sounds like a conflict between rigidity and agility, yet agility gains its benefit only through these rules - so long as these rule follow what I call the *excellence principles*: parsimony, requisite variety, and harmony.

Gene Guglielmo gets credit for key advice in the infrastructure design and principle application. He succeeded me as Silterra's CIO to complete the implementation, and was the force behind the ETL and BIM interface concepts. His presentation (Guglielmo 2004) at a Delphi Group "Enterprise On Demand" conference sheds light on these important parts of the architecture, and key human-factors issues involved with requirements development.

Principles Applied to Process

These same *response ability* principles were applied to the initial implementation and integration process of the ERP application and enterprise bus. We considered the process to be a system in its own right.

Before looking at the process, the benefits that came from this approach are outlined:

- The company had functioning out-of-box (phase 1) ERP supporting the business within 90 days of implementation start, a custom business-process phase 2 implementation 90 days later, and a refined phase 3 implementation 90 days after that. Typical comparable implementations were taking 24-36 months, according to the Oracle implementers.
- The entire project was implemented on-time and below budget. Initial ERP applications were predominately Oracle 11i, as they had the only acceptable web-capability at the time. Licensing was budgeted at $5 million and implementation at another $5 million. Licensing was on budget and implementation came in close to $4 million. Comparable typical numbers were $15 to $25 million, according to the Oracle implementers. We were one of the first to implement Oracle 11i, meaning we wrestled with the usual new-software instability and bug discovery problems.
- A PeopleSoft HRM application collection was added in a 2-phase sequence: out-of-box and final. Three months were scheduled for

each phase. The total for both came in at five months, against a comparable typical expectation of 12-18 months - according to the PeopleSoft implementers.

Process - The process is depicted in Figure 2 and is beyond the scope of this paper to explain in the same detail as the infrastructure - but key points should make the application of principles obvious.

Vendors of software were required to take full responsibility for the methods of software

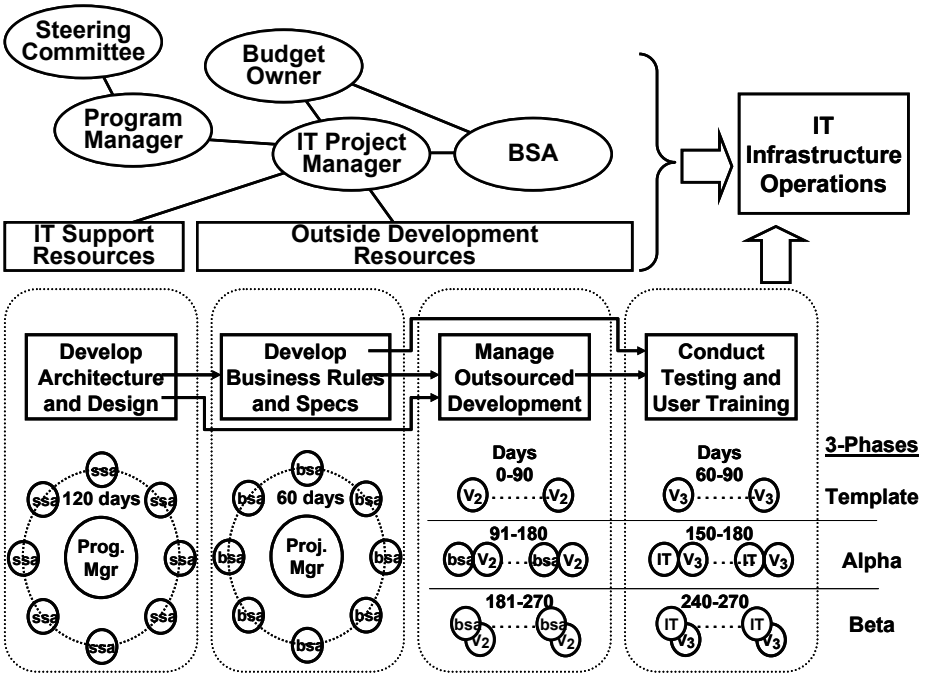

Figure 2 - Agile Implementation Process System

implementation, with no interference from others. Vendor responsibilities were encapsulated as project modules - with the rule that requirements (module input and interface) up front would not change during implementation. The vendor implementation process was required to occur in three (or 2 in the case of PeopleSoft) 90-day phases: 1) an out-of-box best-practice from the vendor, with no customization beyond the necessary, such as chart-of-accounts, 2) a reconfiguration according to business processes developed independently during phase 1 by the BSAs, and 3) a refined reconfiguration of business processes developed during phase 2 by the BSAs, based on experiences from operating the company under phases 1 and 2 configurations.

Encapsulation occurred along another phased-development dimension as well: architecture, business rules, integration management, testing and training. Clear interfaces between these modules were enforced with demarcated points of requirements vs implementation.

BSA business process development was similarly self-organizing and encapsulated in a three-phase 90-day-each approach. With no changes permitted once a phased-set of business rules were completed and intended for implementation.

In the 2nd-phase shown in Figure 2 the BSAs involved with process reconfiguration and the IT personnel involved with testing and training are shown side-by-side with vendor personnel - as they learn how to take long

term responsibility for these activities. The 3rd phase depicts them in front, with vendor over-the-shoulder advice as they transition into full ongoing maintenance and upgrade responsibility. These internal resources are transformed into encapsulated self sufficient modules of the Infrastructure System.

Responsibility for architecture, principal-application, integration process, and integration process management were internal - not put in the hands of the vendors or third-party integrators. This was a source of considerable wrestling. First with Silterra management, as it was a departure from traditional practice, and moved responsibility for failure internally without outside scapegoats or recourse. Secondly, and quite contentious initially, was vendor revolt. Each vendor had their own standard implementation and integration process quite at odds with our demanded three-phase, short-cycle, encapsulated approach. Most notably, after repeated refusal too conform by the Oracle US integration team in pre-implementation planning, the implementation contract was given to the Oracle Malaysian group - not nearly as experienced, but excited about the opportunity and appreciative of the potential for doing it a new way. Oracle's Malaysian project leader subsequently marveled at the results and expressed intentions to repeat the process elsewhere.

Everything did not go according to plan - The integration with the mandated-hands-off MES production system presented data integrity problems that took too many months to rectify. The principle reasons for this difficulty were a mandate to leave the manufacturing execution system (MES) untouched, the simultaneous Herculean effort by production people and management to bring the plant processes on line (precluding production involvement in rectifying data integrity issues), and, frankly, a culturally-embedded and uncooperative interest for the greater business issues on the part of the production group.

The initial Oracle applications were permitted to communicate through their standard "back-door" interfaces as opposed to the bus in most cases. This was necessary as Oracle 11i was still in final development as we implemented, and most APIs were undocumented.

The ETL template approach took some iteration, learning from the first few brute force approaches before standardization requirements became sufficiently understood.

Conclusion

The principles discussed here (Table 2) were developed from analysis of systems exhibiting agile characteristics of good reactive and proactive response to unpredictable events in uncertain environments (Dove 2001, Chapters 5 and 6). They appear to be fundamental, in that they exist in a large variety of system types, and are independent of system type. They contribute directly to improved responsive to change and predictability - as shown above in the Principles Applied to Process section. They are broader based, and somewhat different than software architecture and design principles synthesized as recommended and effective practices - though there is much overlap evident. They are, however, complimentary, not meant to supplant the principles employed in more detailed activities of design, coding, and implementation.

These principles are intended as root-level principles, which may instantiate in different ways by different practitioners. They are meant as underlying first-thought modes, Above all, they are meant to guide the construction of agile systems - not all systems.

Eventually at Silterra, when the CEO position and corporate management transitioned into more local (Malaysian) control, the corporate agile objectives and values became less appreciated. Nevertheless,

the basic infrastructure and implementation principle-based approaches delivered and proved their values. Continued understanding and maintenance remains a question.

The primary lesson-learned by this author, from the management transition, was the need for continued value-propositioning and culturally-embedded value-propositioning skills - as a corner-stone of agile systems that are to remain agile through time (Dove 2005).

References

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J. Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R., Mellor, S., Schwaber, K., Sutherland, J. and Thomas, D., *Manifesto for Agile Software Development.* www.AgileManifesto.org, 2001.

Börjesson, A. and Mathiassen, L., "Organizational Dynamics In Software Process Improvement: The Agility Challenge." *Proccedings IFIP WG 8.6 Conference*, Leixlip, Ireland, 2004.

Dove, R., *Tools for Analyzing and Constructing Agile Capabilities*, Agility Forum, PA96-01, Bethlehem, PA, Jan 1996, www.parshift.com/Files/PsiDocs/Rkd4Art4.pdf

Dove, Rick, *Response Ability: The Language, Structure, and Culture of The Agile Enterprise*. Wiley, New York, 2001. www.parshift.com/ResponseAbility/Preface.htm

Dove, Rick, "Agile Enterprise Cornerstones: Knowledge, Values, and Response Ability." Keynote paper, *Proceedings IFIP WG 8.6*, Atlanta, May, 2005.

Dove, Rick, "A Framework Driven Procedure for Developing Agile-System Requirements - With an Agile-Security Strategy Example," Unpublished paper.

Guglielmo, Eugene, "Bridging the Legacy Gap." Proceedings Enterprise on Demand conference, Delphi Group, Boston, MA, March 2004.

He, Hao, "What is Service-Oriented Architecture?" O'Reilly xml.com, September 2003, webservices.xml.com/ /pub/a/ws/2003/09/30/soa.html

University of Massachusetts at Dartmouth, "Software Architecture Resource Sites", 2001, www2.umassd.edu/SECenter/SAResources.html

Biography

Rick Dove is CEO of Paradigm Shift International, and Chairman of The Agile Security Forum. He has a BSEE from Carnegie-Mellon University, did graduate work at UC Berkeley in Computer Science, spent his early career as a systems software developer, and gravitated to start-up, turn-around, and change management. He has run companies producing software products, manufacturing machinery and services, and strategic planning and management services. He has led engineering, R&D, IT, cyber-security, sales, and marketing. He instigated the Department of Defense support for the agile enterprise program at Lehigh University, was co-principle investigator on its seminal formation project, and led the subsequent Agility Forum research and industry involvement activity. He is author of *Response Ability: The language, Structure, and Culture of the Agile Enterprise* and *Value Propositioning - Perception and Misperception in Decision Making* (www.parshift.com/ValueProp).

Appendix

Reprinted from Chapter 8 of (Dove 2001)

Defining the Problem

On the first of December, 1999, thirty days after the project began, we issued an RFI (request for information) to a variety of ERP and middle-ware vendors. This RFI embodied our intuitive problem definition, outlining the situation Silterra envisioned, and the requirements that must be addressed by any solution.

------ RFI -----

What follows is a description of a new semiconductor manufacturing company, Silterra, its projected growth environment over the next few years, and its concerns for remaining highly adaptable. Some of the description is fact, and some is the potential situation the company believes it must be capable of dealing with. The purpose of the description is to paint a picture which must be addressed by the company's information technology (IT) infrastructure and eCommerce/eBusiness/eMIS applications. Here, Internet/intranet facilitation of customer relationships we call eCommerce, of all external relationships we call eBusiness, and of total enterprise operation we call eMIS – each successively inclusive of the other.

-------- The Company and the Business --------

Silterra is a new company entering the semiconductor manufacturing field as an outsource. Its customers are other companies which either have no manufacturing facilities of their own (fabless companies) or do, but need extra production capacity, or need the latest in production and packaging technology. Construction of its first production facility is underway in Malaysia, and is expected to produce deliverable product by end of Q4 2000. The company is funded by the Government of Malaysia, which views the investment as a strategic national move, and expects to grow the company considerably with additional production facilities in a relatively short period of time.

A semiconductor fabrication facility typically costs in excess of a billion US dollars to bring up. Annual revenue from such a facility operating at capacity approaches the same number. Silterra expects its first plant to be operating at capacity before the end of 2001.

Currently Silterra has its first manufacturing facility under construction, production employees in training, and sales activities underway. It can begin filling orders in Q2 2000 from another company's production facilities, and then will fill orders from its own production capability by the end of Q4 2000.

As a new company Silterra as yet has no IT legacy to contend with. Now, however, is the time for Silterra to commit to an IT strategy, and to begin implementation. Implementation schedules will be driven by the company's emerging needs as they progressively awaken throughout the year of 2000. The strategy is still in the stage of initial formulation, and will in fact always be in a stage of re-formulation. The cornerstone of the strategy, throughout its evolution, will be an architecture that enables and facilitates constant change – in all dimensions.

It is anticipated that multiple vendors will participate in the creation of the IT infrastructure and applications; if for no other reason than to approach a guarantee that the resultant system will in fact accommodate software and hardware from multiple vendors as time unfolds. We intend to build an infrastructure that is independent of any software vendor, hardware vendor, or systems integrator; one that facilitates the quick inclusion of new applications, the replacement of any in-place applications or infrastructure element, and even the replumbing of the underlying structures and concepts.

This request for information seeks to help us understand how well you can help us address this intent of independence and adaptability, while providing elements of infrastructure and application necessary to support the enterprise IT and eMIS needs.

-------- Some Issues of Adaptability --------

We are looking for an adaptive ability beyond what the term flexibility generally implies, an ability more in line with what some call "agility". The adaptive ability we seek encompasses the ability to change quickly, change inexpensively, change robustly, and change without limit.

As in most industrial sectors today, the onrush of eCommerce has the semiconductor industry madly searching for new forms of electronic relationships with customers, suppliers, partners, and employees. At the same time many companies have just completed, or expect to soon, a multi-year ERP mega-project implementation, a highly disruptive process in itself, and now one that threatens to limit the potential for eCommerce exploitation by the very nature of its business-model-defining framework.

Silterra believes that eCommerce/eBusiness means a lot more than a web view of the traditional business model, and also believes that a state of turmoil is likely to exist for the indefinite future, before successively newer business models eventually converge, if ever.

Silterra plans to grow relatively rapidly, both by addition of new internal production capability and by acquisition of other companies with frontend and/or backend capabilities. Local sales and customer support operations will exist wherever in the world reasonable markets exist, with active offices in the USA, Europe, and Japan by the end of the year 2000. Acquired organizations will have legacy systems installed which are unlike Silterra systems.

Though growth is expected, so is the unexpected, as are the cyclical fluctuations which have historically characterized this industry. Relative to change associated with volume, the IT infrastructure and

applications should facilitate a contracting environment as well as an expanding environment.

Users of the company's IT systems and applications may be located anywhere at any time, may be connected with a variety of current and future data devices, be of virtually any national origin, and may have any of a variety of relationships to the company, including but not limited to employee, customer, supplier, and partner. Some will be heavy and repeat users of one or a few applications, while others may be occasional or one-time users of many, such as a manager making use of a financial planning application, an employee seeking information from corporate data, or a customer trying to resolve a problem. Users at customer locations may be large in number, diverse in nature, and change frequently. The ease and speed of becoming an effective user, as well as gaining appropriate and authorized access, is an issue for all types of users.

Recovery in the face of malfunction, dysfunction, and disaster is an adaptability aspect of some importance. The IT infrastructure and its applications should be both fail soft and fail safe as appropriate to the risk and penalties. The eCommerce user shows little tolerance for inaccessible or slow response, the eBusiness user can be expected to follow suit, and the eMIS user cannot afford to have the "dashboard" of the company disappear.

Security issues, always important, become even more so with the advent of eMIS; especially when this term includes the ability to control as well as monitor. If a customer, for instance, is to have a web-enabled ability to enter orders and change or reschedule existing orders, fail-safe, yet minimally intrusive, procedures must insure that only authorized persons may exercise these abilities. Security is a necessity, and so is a way to accomplish this end without unnecessarily intrusive procedures.

eCommerce/eBusiness has the potential to introduce unexpected surges and explosive increases in activity within the IT infrastructure. Computing platform choices are typically based on volume expectations. With the uncertainty associated with eCommerce/eBusiness, scalability is a concern.

Expressing our business as applications or executable models requires time, knowledge of the methods for expression, and knowledge of the business rules, processes, and practices. The nature and availability of the knowledge expertise, and the amount of time to capture and express the knowledge in applications or models are concerns.

Though Silterra is characterized as a semiconductor company because it addresses that market, it can also be viewed as a plant building company, considering a scenario of two new plants built each year for the next five years on the average. Reuse, reconfiguration, scalability, and evolvability of all supporting IT infrastructure elements and applications becomes a concern in this light.

-------- Things To Address --------

If they are applicable to what you are presenting, you should address the following questions in addition to whatever else you wish us to know:

- What products, services, and/or approaches have you got that can help us achieve any of our needs for functionality and adaptability?
- How do these address the situations and concerns we have raised in his document?
- What else can these address that we should be concerned about?
- What would be the process that takes these and turns them into operational functionality for us?
- What standards do these adhere to that give them interoperability, and how strict are those standards followed?
- How do these relate to XML, CORBA, and eJB?
- What portions of these require what other portions of these?
- What portions of these are completely independent of all other portions of these?

------ End RFI ------

Companies invited to present unfailingly addressed the first half of the first bullet above, and virtually ignored the rest. Our clearly stated intentions to focus on adaptability fell on resoundingly deaf ears. Nevertheless, the RFI document served as a solid problem definition that provided guidance throughout the project.