vervint.

Enterprise Architecture as a Practice

BRIAN K. PARTIN
PRINCIPAL & ENTERPRISE ARCHITECT

Contents

A Problematic Approach to Innovation

By their very nature, digital innovation or research and development activities of any kind have heightened costs and risks compared to well-known and proven approaches. In fact, nascent domains like IT, where engineering practices and associated controls have yet to mature fully, have increased innovation risks over more traditional ones.

As a case in point, recent PwC Strategy research indicates that only **10% of companies succeed** in generating new revenue from digital innovation, and **40% saw no significant impact** whatsoever (Roeder, 2018). A similar study from Accenture found only 22% of large organizations are able to reach the scaling stage of digital innovation with up to a 9.9% ROI (Ohr, 2020).

Despite the costs and risks, digital innovation has become an existential problem for most organizations transitioning from traditional to digital forms. Customers, partners, and employees have expectations that directly impact engagement and retention. Our own experience working with clients, as well as current industry research, strongly suggest that organizational structure, expertise, governance, culture, appropriate scoping and scaling, and clear, honest, and continuous examination of risk/reward are key to success. In short, meeting the demands of digital is no different than traditional endeavors — **discipline is strongly associated with positive outcomes.**

Even with diligence, in practice, few new ideas bear fruit. Time and time again, organizations find themselves in the position the Red Queen described in Carroll's *Alice's Adventures in Wonderland*.

> *"Now, here, you see, it takes all the running you can do, to keep in the same place. If you want to get somewhere else, you must run at least twice as fast as that!"*
>
> – CARROLL, 1865

This makes the process of arriving at a decision on when and how to innovate a particularly critical one. At a bare minimum, the following questions have to be answered carefully and honestly:

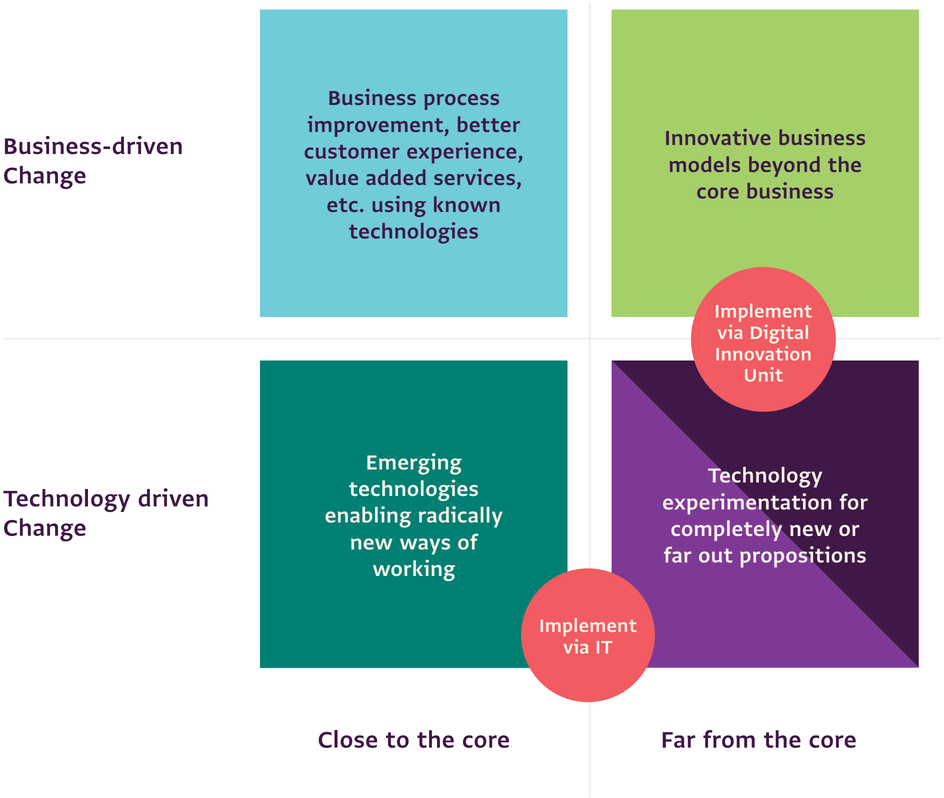
- Is the idea fully or partially **unique** within its proposed context of application?
- Does the idea have **measurable** and understood value within its proposed context of application?
- Does the idea **depend** upon other as yet achieved ideas?
- Is the idea **plausible** to evaluate and scale?
- Does the organization have demonstrable **competencies** to build, evaluate, and scale the idea?
- Does the organization have the **time** to build, evaluate, and scale the idea?

- Does the organization have the **funding** to build, evaluate, and scale the idea?

If the answer to any of those questions prior to beginning innovation or at any point along the way is 'No,' it is best not to take on or continue the initiative.

Presuming the answer to all of those questions is Yes, the authors' own experience, as well as industry research, clearly shows that innovation work is substantially different than other organizational activities and is best addressed in a manner accepting of that.

Based on PwC and other researchers, the following model is an effective one for thinking about who best owns change, including innovation, within the organization:


Business-driven Change
Technology driven Change
Business process improvement, better customer experience, value added services, etc. using known technologies
Innovative business models beyond the core business
Implement via Digital Innovation Unit
Emerging technologies enabling radically new ways of working
Technology experimentation for completely new or far out propositions
Implement via IT
Close to the core
Far from the core

Effectively, the requirements of digital innovation and experimentation are sufficiently unique to warrant them being addressed outside of normal business or technology operations — as part of a digital innovation unit or comparable organizational structure — **with technology incubation and adoption owned by IT and business incubation and adoption owned by business units.**

One point of clarity — innovation and invention are different activities. Save for the obvious exceptions, an organization's activities should not focus on inventing new digital technologies but adopting maturing ones. Technologists within the organization should most assuredly develop demonstrable and mature capabilities in emerging technologies but development of specific technologies or novel technical implementations are best left to organizations structured and funded for that. This approach will ensure a constantly refactored technology base supporting digital innovations dependent upon it.

Recommended Practices

Limit innovation to clear strategic or tactical business advantage

The risk of innovation and the low potential for return means that there must be clear advantages brought in the near term or over time to justify the outlay of financial and human capital. As discussed above, we would strongly encourage organizational IT to engage in adoption and incubation of maturing technologies rather than investing in innovation within its domain.

Make many small innovation bets rather than a few large ones

While innovation is risky — per PwC and Accenture, approximately 20% of large organizations bring innovation efforts to the point of scaling and those efforts have less than a 10% return — there are strong pressures for digital maturation and innovation coming from customers and employees. Organizations who are successful avoid large bets and instead focus on many small ones. Quoting Yates:

> *"[They] know how to remove the greatest amount of risk on the least amount of capital. They know how to identify the risks and run experiments or pilots — small bets — to demonstrate they can mitigate those risks. They identify the metrics and milestones that will tell them they have eliminated those risks."*
>
> – YATES, 2022

Some of the most costly risks of innovation are behavioral. Big innovations lead to behavioral and organizational inertia that impact the dynamism and iterative validations requisite for success. They foster sunk cost and too big-to-fail anti-patterns. We strongly encourage organizations to approach innovation via small, iterative initiatives for the greatest positive impact.

Do not base innovation on ideas or technologies that are not fully mature

Let others take risks first while building internal competencies. Oscar Wilde said, "Talent borrows. Genius steals." Rather than adopt new technologies while they are popular and unproven, wait for the point where others have worked out the best architectural and operational patterns and then build upon that. A strong enterprise architecture practice combined with teams specifically built and staffed solely for exploring candidate technologies and then driving incubation and adoption would ensure an organization's business has a strong foundation to innovate upon.

Avoid innovation in ideas or technologies outside of organizational core competencies

The ability to successfully launch and operate a business innovation over time is 100% dependent upon the technology it is built upon. Developing those capabilities in concert with the development of a product dependent upon them is a fool's errand. We strongly encourage organizations to discontinue the practice of holding IT responsible for developing fully mature capabilities simultaneously with business innovation dependent upon them. Again, a strong enterprise architecture community working hand-in-hand with the business while shepherding new and appropriate technologies will help mitigate this risk.

Set formal gates on innovation initiatives and cancel the initiatives that do not pass them

To avoid over-investing in initiatives where it becomes obvious that the desired return is not going to be realized, there must be governance in place at **consistent, well-established milestones** to evaluate whether there are pivots to be made in the direction of the initiative or potentially a need to scrap the initiative altogether. We encourage organizations to develop formal governance policies and practices for digital innovation and then rigorously follow them.

Keep Innovation discrete from Incubation and Foundational

For sustainable digital innovation, it can be useful to separate organizational systems and solutions into categories:

Foundational: Systems with broad adoption and/or central to an organization's core activities. **Low acceptance of risk due to broad or critical reach.** Well documented and well understood operationally; mature. Low relative rate of change. Highly instrumented and highly observable.

Incubation: Systems with growing adoption or considered to have potential for core organizational value. **Low to moderate acceptance of risk due to growing production use.** Well documented but with continually refined operational practices, growing and maturing. Low to moderate rate of change, with most change to non-functional features. Feature development is generally limited to refinement to task. Instrumentation is biased towards performance and utilization.

Innovation: Exploratory systems limited in scope and studiously minimized organizational blast radius. **High acceptance of risk due to exploratory nature and limited cost of failure.** Design and intent are well documented, and operational practices are poorly understood and immature. Very high rate of change to functional and non-functional components. Regular feature churn and bloat during initial stages with refinement over time. Instrumentation is biased towards utilization patterns.

Consider open innovation where required

Where innovation is considered of high value, but the organization lacks sufficient expertise, partnering with external organizations can be effective but comes with real risks, particularly in regard to its ability to exercise control and ramp internally to support incubation/adoption. Where external expertise is brought in, we strongly encourage organizations to have **clear hand-off plans and controls in place** to ensure their ability to take ownership of their products and their lifecycle when optimal.

Cultural Considerations

Recommended Practices

Communicate clearly and openly horizontally and vertically

This should clearly go without saying, but success over time and at scale requires collaboration and trust. Partial messages or withheld information erodes this and reinforces tribalism and siloing. Endemic reticence to speak to immediate and aggregate risks up the chain of command is of particular concern. Concerns about speaking truth to power greatly reduce the information required to make decisions. Organizations should also be aware of unintended consequences of certain communications. For example, praising extreme actions taken by individuals while seemingly a positive action can lead to organizational patterning and many problematic behaviors discussed in this section. Similarly, using the number of lines of code written per unit time by developers while a common productivity metric actually incentivizes complexity and adds carrying costs to products.

Learn to accept failure and reorient

Complex work at scale and certainly any form of innovation or adoption of new ideas or technologies will involve failure. An organization that punishes failure and/or fails to learn from it will absolutely incentivize the hiding of risk or issues by practitioners, increasing the radius and cost of failure dramatically. Feedback loops requisite for pushing learnings vertically in the organization will atrophy, and technical debt will compound. As Toyota readily showed, there is high value to an organization where mechanisms such as the Andon Cord are emplaced and issues are reported and resolved immediately. For organizations to have success in reducing their technical debt and seeing a return on innovation, they must acculturate behaviors that acknowledge risks taken and pay them down over time.

Demonstrate expected culture from the top down

Practitioners' behaviors reflect those of their leadership. More so than all other roles in the organization, those in charge must precisely, continually, and very visibly display the culture the organization has deemed strategic to its mission. Organizational leadership must incorporate and visibly reflect the culture it wishes to attain.

Don't reinforce hero/expert culture

It is common practice to praise individual behaviors that go above and beyond what is expected. However, done inappropriately, this can have insidious and unintended results on the culture at large. Information and power hoarding become obvious techniques to grow or maintain individual status within the group. This amplifies other cultural problems such as tribalism, removes incentives to document or share learnings, and most assuredly eliminates any hope of encouraging individuals to identify issues and problems — failure cannot be admitted.

Respect domain expertise from inside or outside the organization

This is particularly difficult for organizations with entrenched hero cultures and other tribal behaviors to address but is critical to successfully adopting and incubating new ideas and technologies within an organization. It is greatly exacerbated when teams are composed of internal and external subject matter experts. Overvaluing or undervaluing expertise or specific work based on the silo producing it will impose unnecessary risks and delays. Strong policies, clearly defined zones of responsibility, and rigorous dispute adjudication practices are necessary to blunt these negative behaviors. Leadership must continually focus on and pattern these behaviors for broad and lasting success.

Form studios (product-based) and practices (expertise-based) communities

One highly successful technique digital organizations can use to ensure dissemination and collaborative adoption of cultural principals and patterned behaviors is ensuring practitioners belong to multiple communities aligned with discipline (practices) and cross-cutting work efforts (studios). This drives cohesion more rapidly, reduces tribalism, and bubbles up efficiencies or issues throughout the organization.

Develop aligned but unique cultures for Foundational, Incubation, and Innovation

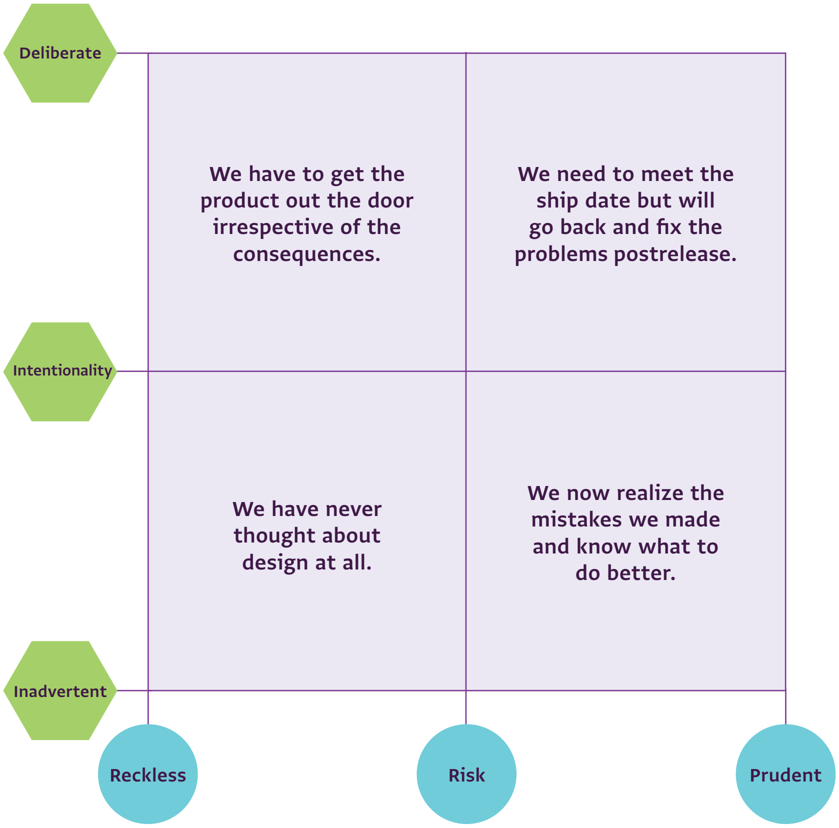
The simple reality is that culture directly drives behavior in practice and that poor cultures will reduce the likelihood of success while good cultures will improve it. While certain cultural behaviors will be universal across all organizational subgroupings, the appropriate cultural behaviors must be engendered within these subgroupings congruent with their work. Patterns of thinking, work, and leadership are and should be different for practitioners assigned to Foundational products (high-value, low rate of change), Incubation products (adoption and growth, low to moderate rate of change), and Innovation products (speculative, high-rate of change). The authors recommend that organizations adopt this or a similar framework in practice.

Unmanaged Technical Debt as a Limiting Factor

One of the most substantial impediments to an organization's ability to digitally innovate is the **business and technical debt of legacy decisions.** Viewed outside of the context of digital technology, Ward Cunningham's 1992 description of technical debt was nothing new. The need for engineering remediation in the form of deferred execution in order to achieve temporal goals is a well-documented and well-known form up to and including the need to understand accrual and interaction of resultant risks over time. What Cunningham did was socialize this premise in software engineering in a manner that was approachable and readily introduced. What makes his metaphor particularly useful arguably is not that it made it simple to understand for engineers but, rather, that it makes it something readily shared and tangibly tracked by those on the business side of the house working with the engineers towards release and maintenance of a digital product over time.

The experience of the authors and a substantial number of researchers, however, is that neither the engineering nor business side of the house make appropriate use of this valuable means of quantifying the compounding cost of risks and creating rational schedules for paying it down. At best, organizations applying Agile or related methodologies will address the most granular level of digital debt, bugs, with episodic and mostly ungoverned events — bug smash parties, bug triage on steroids, etc.

Technical debt, much like financial debt, needs to be well-known, accepted, tracked, and there must be the capacity to pay it down over time or it should not be taken on. Fowler and subsequently Ernst, et. al. posit that technical debt must be understood along two dimensions — risk and consideration:



We have to get the product out the door irrespective of the consequences.

We need to meet the ship date but will go back and fix the problems postrelease.

We have never thought about design at all.

We now realize the mistakes we made and know what to do better.

As is the case with the typical quadrant chart, the upper right is most desirable. In this case, technical debt is absolutely reasonable when taken on in a deliberate and prudent manner.

Phrasing in the chart above is particularly poignant for practitioners as many if not all of us would argue that we almost always land our technical debt in the upper right quadrant and then are not provided the resources to perform post-release resolutions. There is a logic flaw in that argument because it assumes that the acceptance and ownership of technical debt is on the technical team delivering the product. This is a fallacious assumption. As with any debt, ultimately, it is held by the organization itself. The charge on the technical team is not to make the decision but to fully inform the stakeholders of the debt taken on, the risks accrued over time, the value assuming this risk brings, and the payment schedule. This is critical because even prudent and deliberate debt will directly impinge on the product in question and the organization as a whole and must be accepted and managed at that level. (Brown et. al., 2010) found that the accumulation of technical debt “without regular and rapid pay down can lead to unforeseen consequences when finally addressed due to complexities and entanglement”. (Brown, et al., 2010). Per Maki, Netta, et. al., “up to 20 percent of…new product budget ended up going toward resolving existing issues related to technical debt” (Maki, Penttinen, & Rinta-Kahila, 2020).

As is the case with financial debt, **technical debt taken on with intent and full understanding of the risks posed until it is paid off is a valuable business tool.** We discuss below practices we have found greatly improve an organization’s ability to appropriately gain the advantages of technical debt while avoiding its pitfalls.

Recommended Practices

Prioritize quality and lifecycle (non-functional) requirements over features for Foundational and Incubation products

As the footprint and reliance expand during the Incubation and Foundational portions of the product lifecycle, the importance of focusing on issues relating to operations and maintenance greatly increases. Matters such as change control, instrumentation, security, scaling, contingency operations, etc., become increasingly tied to success. While these items must be addressed to some degree during Innovation, if only to meet the needs of change velocity, they must be mature before a point where the organization or its clients depend upon them to any substantive degree during Incubation.

Apply rigor to requirements elicitation, requirements engineering, and requirements prioritization

While Agile practices have changed the timing and means by which requirements are gathered, it has arguably made it far more important that an organization manage the identification, engineering, implementation, and ultimate usage of requirements. Two industry stats drive this home — **70% of software products are deemed not**

to have met user expectations upon delivery, and 80% of software features are never or seldom used. Ensuring that end users, as well as practitioners, continually refine a product over its lifecycle mandates care and consideration of its scope. We strongly recommend that organizations continually refine their methodologies, especially ensuring visibility to the practitioners who will be implementing the requirements in code or infrastructure.

Require all products (including Innovation) to have architectural design and documentation that meets enterprise architecture requirements

Ensuring fidelity to desired goals is nearly impossible without written specifications and clear design. Onboarding times for engineers responsible for implementation are increased, and velocity is diminished when tribal experts need to be consulted continuously. Even more critically, when a problem arises during implementation or, far worse, in production, there is no known desired state to validate against. Resolving the problem often becomes an exercise in trying possibilities until something works.

Accrue, budget for, and manage technical debt in a deliberate and intentional manner

There is a common saying that if something isn't documented, it doesn't exist. In practice, this is effectively true regarding what decision makers at most organizations have to work with when it comes to making decisions about technology. Because existing technical debt is not carefully, precisely, and openly quantified, the valuation of initiatives is likely made within their individual context and without a full measure of how they play out against currently carried risk. We strongly encourage organization IT to manage technical debt openly in a deliberate and intentional manner.

Consider both code and configurations as inventory with carrying costs (maintenance) over time

In practice, we have observed that the vast majority of organizations carry far more code, infrastructure, and software products than is prudent. In addition to the initial outlay, carrying costs are associated over time. Speaking solely to software products, approximately **80% of a project's real costs are maintaining the software over time.** (Ernst, Kazman, & Delange, 2021). Another component in the carrying budget is the impact on troubleshooting, scaling, or developing other products. Complexity has a cost. We recommend that organizations consider and account for this at all phases of the product lifecycle.

Leverage tools to review code and do architecture analysis against defined standards.

While assuredly not a panacea, using linting tools with organization-specific standards for ensuring that expected styles and conventions are followed greatly improves quality for internal and external developers. While we have directly seen these tools in use at most organizations, there are typically no documented settings we can be pointed to for use across the firm or by-product. Instead, the base vendor settings were almost always used. From our experience, this is a missed opportunity, particularly considering the percentage of new or outside developers most organizations depend upon. While base settings are a good start, far more value is obtained where the organization or product has a deliberate point-of-view regarding coding styles. Beyond coding style, tools are readily available to assess and report on architectural aspects of solutions. In particular, tools for assessing security, modularity violations, cycling issues, inheritance violations, etc., exist and should be used. It would, in fact, be unsurprising to find they are being used. While these tools assuredly do not eliminate the need for highly skilled architects and practitioners, they do provide tremendous value, ensuring that the organization's point-of-view is met over a product's lifecycle.

Implement CI/CD and related DevOps practices aggressively

Continuous deployment of necessity drives small and atomic units of change in the organization that are far easier to test and greatly reduce risk (Dingman, Kralapp, Kreitz, Scott, & Bush, 2023). In addition, it creates useful cultural tensions that drive ownership towards the practitioner rather than responsibility, which evolves the organization structure and causes it to lose context. Where ownership by the practitioner doesn't turn into blame shaming but rather pride of practice, substantial quality improvements are a result and, indeed, one of the value propositions of CI/CD as a methodology.

Instrument products to a feature level for usage patterns, as well as failures, performance, and change management

Quoting Ernst et al., "Evidence suggests only 20% of the features in a given piece of software account for 80% of use." (Ernst, Kazman, & Delange, 2021) This makes it critical that usage patterns be tracked and adjusted over the full lifecycle of a product, particularly during the pre-scaling phases of new software. Removing a feature and the lifetime cost of carrying it is far easier to do before the reach of a product is maximized. We strongly encourage organizations to instrument and respond to patterns of user behavior starting at the earliest stages of the product lifecycle to release and support in production only those demonstrably useful features.

Enterprise Architecture's Place in Digital Innovation

There is no denying that the release of the Agile Manifesto in 2001 changed the way software and, subsequently, how other project work was viewed in technology and seemingly unaligned fields. It offered a way to see progress almost immediately and its regular iterative cycling on atomic units of functionality made it easy to provide metrics to stakeholders. Among the manifesto's twelve principles was this:

> *"The best architectures, requirements, and designs emerge from self-organizing teams."*
>
> – BECK, ET AL., 2001

The problem with this statement taken out of context and intent is that it directly opposes what we categorically know about important endeavors of any kind — arriving at a desired endpoint in a timely manner requires planning and consideration. Ernst et. al. in “Technical Debt in Practice: How to Find it and Fix It” have a particularly evocative example of why architecture cannot simply be discarded:

> *“Would you fly in an aircraft if you learned that its flight control system had simply emerged from a self-organizing team?”*
>
> – ERNST, KAZMAN, & DELANGE, 2021

The answer is of course not. Any system of import must have thoughtful and careful consideration of specification to purpose and the proven engineering methodologies best employed to achieve those purposes within time and budgetary constraints. The reality, however, is that Agile is often taken out of context and, ignoring intent, is often misapplied1. The allure of velocity and delivery of products with seemingly impossible timelines captured the zeitgeist. To quote Martin Fowler within a couple years of his signing the Agile Manifesto:

> *“In its common usage, evolutionary design is a disaster.”*
>
> – FOWLER, IS DESIGN DEAD?, 2004

While the role of architects never truly disappeared, their role as designers, as owners of the specification and order of operations, and arbiters of risk over reward was greatly diminished within many organizations. Along with that came the demise of the idea that specification in the form of common documentation is critical to success not only for initial delivery but also over the product’s lifecycle. In the authors’ direct experience, the discipline required to innovate successfully digitally, the patterning of cultural behaviors requisite to sustaining it, and the knowledge of how to leverage technical debt appropriately to increase velocity are **only obtained when enterprise architecture as a practice is a core organizational competence.**

Recommended Practices

Appoint Enterprise Architects with requisite skills

Gartner defines enterprise architecture as:

> *“A discipline for proactively and holistically leading enterprise responses to disruptive forces by identifying and analyzing the execution of change toward desired business vision and outcomes. EA delivers value by presenting business and IT leaders with signature-*

> *ready recommendations for adjusting policies and projects to achieve targeted business outcomes that capitalize on relevant business disruptions."*
>
> – GARTNER, N.D.

An enterprise architect, therefore, is an individual who understands technology within the context of business strategy. This last is key. Without a firm and direct understanding of how the business currently functions and how it intends to function in the future, technical puissance is meaningless. These individuals and the body of work they produce, curate, and communicate are the lynchpin of digital innovation. In practice, however, this proves problematic. For example, Winter finds that:

> *"Although many architects tried to position themselves as a linking-pin 'between' corporate management, business/project owners and IT, their backgrounds and competency profiles often kept them close to the corporate IT functions, limiting their credibility on the business side."*
>
> – WINTER, 2022

It is the authors' experience that this mismatch often occurs where organizations use the title, "Enterprise Architect", as a means of denoting status or longevity within an organization rather than as a means of describing a set of requisite skills and responsibilities. It is our strong recommendation that organizations disabuse themselves of this practice and firmly and clearly establish this role and its responsibilities if they wish to achieve any measure of success with digital innovation.

Empower enterprise architects

As a strategic role within the organization, enterprise architects must be empowered to engender change and hold accountable activities diverging from the organization's strategic direction.[3] As such, placing them at more tactical levels within the organization will have little positive impact and, indeed, can lead to a high degree of complexity and incoherence of technical approaches to business problems. Enterprise architects are typically best served reporting as individuals (or as a Principal EA-managed group) to the organization's head of technology or information services.

Develop design principles library

As individuals, enterprise architects have a multiplicative effect on the value and volume of work produced by technical and aligned business practitioners. They do this by effectively baselining the known and the accepted within an organization. Effectively, this reduces ramp times for product work, creates guidelines for evolutionary design practices such as Agile, reduces complexity, makes onboarding of new employees and partners faster and of higher fidelity, and reduces risk while improving quality. While enterprise architects must have extremely strong verbal, written, and visual communication skills, they are **not the sole author** of design principles within an organization. They are the editor, arbiter, and curator of the library where the content exists. To successfully implement an organization's strategic digital innovation and foundational systems' goals, it must develop a well-curated design principles library.

Curate existing technical documentation

In the author's experience, most organizations have scattered and unaligned repositories of technical documentation. Much of the content in these stores is outdated, fragmentary, contradictory, or non-authoritative. As is, this content poses more risk than it does reward individuals persistent enough to discover it. This content must be triaged, curated, and discoverable to business and technical practitioners.

Form studio (product-based) and practices (expertise-based) communities

As mentioned above, one highly successful technique to ensure the dissemination and collaborative adoption of cultural principals and patterned behaviors is ensuring practitioners belong to multiple communities aligned not only with discipline (practices) but also with cross-cutting work efforts (studios). In our experience, placing responsibility for success of this approach in the hands of enterprise architecture leads to sustainable and valuable outcomes.

Drive adoption of technology rather than invention

Of particular concern to any organization should be the strong preference of technical practitioners for creating new techniques or approaches to well-known or previously solved solution spaces. When questioned about these approaches, velocity or cost are often pointed to as drivers. This is contrary to the authors' own experience as well as research. Speaking only to cost, Ernst et. al. points out that:

> *"Software maintenance takes between 50% to 80% of total project cost...[the majority] of the research... agrees...it is at the high end of the range."*
>
> – ERNST, KAZMAN, & DELANGE, 2021

Hence, whatever the measured cost of getting a custom approach to production, four times that cost will be accrued over its lifecycle. Add to this the well-documented risk and low reward of innovative efforts discussed earlier; adopting existing products, frameworks, or components within the organization's existing portfolio or externally sourced would be far more productive.

Develop clear guidance for build vs. buy decisions

Correlated to the above and particularly critical when considering any kind of innovation at scale, organizations must develop clear guidance for when to purchase technologies or services that clearly and explicitly consider full lifetime cost comparison based on substantiated and demonstrable data rather than aspirational accounting.

Use different architectural approaches for Foundational, Incubation, and Innovation products

It should be readily apparent that Foundational products (high-value, low rate of change), Incubation products (adoption and growth, low to moderate rate of change), and Innovation products (speculative, high rate of change) must be designed and governed in different ways. In practice, we do not see this happening at most organizations and strongly encourage its adoption.

Adopt technologies/ideas fully before basing innovation upon them

Martin Fowler rather adroitly addresses this in a speech regarding the adoption of microservices:

> *"You want to move something early on that's not too critical to the overall flow of the organization because you are going to mess things up the first time. Everybody messes up everything the first time. Get that working, and then move quickly to the ones where you will get the biggest return on value. But all of these decisions are based on the business value of things."*
>
> – FOWLER, THE ELEPHANT IN THE ARCHITECTURE, 2020

He precedes that by stating that organizations should not even consider microservices before they are highly competent and mature with basal skills preceding adoption of them, such as DevOps. However, we consistently see innovative and business-critical technology products based on new (or new to the organization) technologies where it is demonstrably clear that neither practitioners nor the organization have the deep and practical engineering experience necessary to drive production or maintain and support the system once there.

Develop a regular communications strategy for detailing changes, alignment, new tech vs existing tech vs outdated tech, etc

The simple fact is that governing principles and practices will and must change over time. It is not sufficient to expect this information to simply trickle down to the level of practitioners. One proven approach to this was outlined above in the discussion of studios and practices; however, that is not sufficient in and of itself. Enterprise architects must regularly evangelize the state of their work directly and, at a minimum, by maintaining change logs and versioning of the library products they create.

Actively audit patterns and principles in practice

There is an aphorism stating that “One should trust. One must verify.” This is absolutely true of domains like architecture and engineering, where complexity is high, and so too is the rate of change internal and external to the systems governed. Detailed, pointed, and measurable audit practices must be in place not only to find points where the guardrails need reinforcement but also where benefit might be gained by expanding them.

References

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., . . . Thomas, D. (2001). Twelve Principles of Agile Software. Retrieved from The Agile Manifesto: http://agilemanifesto.org/iso/en/principles.html

Brown, N., Ozkaya, I., Sangwan, R., Seaman, C., Sullivan, K., Zazworka, N., . . . Nord, R. (2010). Managing Technical Debt in Software-Reliant Systems. FoSER, 1-5.

Carroll, L. (1865). Alice's Adventures in Wonderland. London: Macmillan.

Cohen, W. M., & Levinthal, D. A. (1990). Absoptive Capacity: A New Perspective on Learning and Innovation. Administrative Science Quarterly, 35, 128-152.

Dingman, A., Kralapp, K., Kreitz, C., Scott, C., & Bush, E. (2023). Devops: A deep dive into Change Control and Monitoring Practices. Grand Rapids: OST.

Ernst, N., Kazman, R., & Delange, J. (2021). Technical Debt in Practice. Cambridge: The MIT Press.

Fowler, M. (2004, May). Is Design Dead? Retrieved from MartinFowler.com: https://www.martinfowler.com/articles/designDead.html

Fowler, M. (2020, 2 28). The Elephant in the Architecture. Retrieved from https://www.youtube.com/watch?v=u_LnubpBDCA

Gartner. (n.d.). Gartner Glossary. Retrieved from Gartner.com: https://www.gartner.com/en/information-technology/glossary/enterprise-architecture-ea

Maki, N., Penttinen, E., & Rinta-Kahila, T. (2020). A Domino Effect: Interdependecies Among Different Types of Technical Debt. Proceedings of the 53rd Hawaii International Conference on System Sciences, (pp. 5949-5958).

Ohr, R.-C. (2020, 6 27). Digital Innovation Units: Setting-Up for Scaling-Up. Retrieved from Integrative Innovation: https://integrative-innovation.net/?p=2448

Porter, M. E. (1980). Competitive Strategy: Techniques for Analyzing Industries and Competitors. New York: The Free Press.

Roeder, H. (2018, 7 18). A Strategist's Guide to Digital Innovation (Part 1): Mastering Digital Innovation. Retrieved from LinkedIn: https://www.linkedin.com/pulse/strategists-guide-digital-innovation-part-1-mastering-holger-roeder/

Vieru, D., Klein, S., Bourdeau, S., & Ntakirutimana, J.-B. (2021). Developing Ambidextrous Routines in the IT Service Provider Industry. Proceedings of the 54th Hawaii International Conference on System Sciences, (pp. 6452-6461).

Winter, R. (2022). Designing Informal EAM Interventions - A Complementary Approach for Managing Enterprise Architecture Complexity. Proceedings of the 55th Hawaii International Conference on System Sciences, (pp. 7244-7253).

Worsley, P. (1957). The Trumpet Shall Sound - A Study of 'Cargo' Cults in Melanesia. Macgibbon & Kee.

Yates, L. K. (2022). The Unicorn Within. Boston: Harvard Business Review Press.