Evolving Technology and the New Risk and Reward Picture in Information Systems, Part III: The Transition from Legacy Architectures-Applications to Services

Description

In the previous post, we examined some of the characteristics of early application systems through the lens of the tools employed to build and operate them and their impact on the organizations that used them. Clearly, limitations of the available technologies imposed constraints on how and how fast companies could adapt and transform to reposition themselves in the markets in which they operated, exploit opportunities and respond to business threats. The systems and the way that they were designed, implemented and integrated into the organization and operations of the companies that use them should be looked at as elements of a complex ecosystem of causes and effects. In such ecosystems changes in the business environment engender the need for revised systems and evolving technologies create opportunities to which the enterprise must respond. The ability to understand and manage the resonance between outside-in business drivers (evolution of business models and changes in the competitive landscape) and inside-out drivers (evolution of technology-driven capabilities) is a critical success factor for todayâ€™s enterprises.

The ability to transform rapidly is gaited by the ability to revise and automate processes, which depends on the ability to design and build systems quickly. Rapid transformation brings with it the potential for errors and omissions so, therefore, there is a need to balance prospective competitive benefits with the costs and risks of building the wrong things or building them in such a manner that they ultimately do not operate reliably or serve the strategic imperatives of the business. Overall, what has happened over the past 25 years or so has impacted every aspect of the relationships among these considerations.

Systems: Under the Hood Now

As opposed to the architecture that I described in the previous post, todayâ€™s applications are composed of innumerable components, distributed independent data repositories and complex communications networks and protocols. *Separation of responsibilities* and *statelessness*, design characteristics that minimize interdependence, or *coupling,* among components are common goals. *Orchestration*, coordination among the components to assure that services are provided reliably and coherently, is a requirement that must be accommodated in design and implementation. Incorporation of external services and capabilities from *open-source libraries,* enabled via *Application Program Interfaces* (APIs) facilitates rapid delivery of rich functionality, much of it data-driven and some of it providing advanced functionality, such as *artificial intelligence* and *machine learning*-based capabilities.

The evolution of technology and methodologies for implementing and delivering application systems have created substantial competitive issues for enterprises and their ICT organizations. Some companies still have an enormous investment in and dependence on systems developed *more than 20 years ago.* No one wants to write off and replace depreciated assets that are still performing needed functions and no one wants to take on the risk of replacing them, especially when they may be poorly

documented, not particularly well-understood and dependent on a shrinking pool of talent with experience in the technology in which they were implemented. Such systems will certainly need to be updated from time to time and the risk of operating them only increases as they grow older and older.

This TechTarget article addresses some of the issues facing companies that continue to operate mainframe systems, which constitute many of the â€˜dinosaursâ€™ still running, indicating a general trend toward replacing them. When monolithic systems were first built, *vertical scaling* was almost all that was available to add capacity when it was required. Vertical scaling simply means â€œget a bigger computer with more memory and faster processor(s).â€• It is, in general, a discontinuous and expensive way to grow. If a system can support n users, then n + 1 will require a larger machine, say one twice as capable as the one it replaces. This raises the cost of servicing them to twice what it was until the user base grows toward the limit of the larger configuration.

Todayâ€™s componentized architectures inherently favor *horizontal scaling,* which is accomplished by adding capacity in parallel and sharing the load across multiple processors. In the cloud environment capacity can be managed automatically and dynamically, spinning new servers up as needed and shutting them down as demand ebbs. The user pays only while servers are running, producing significant cost savings, and the ability to scale automatically provides resilience as failing processors can be bypassed and replaced quickly, automatically and transparently. In fact, *serverless computing*, known as Function as a Service (FaaS), in which no particular server (virtualized or actual) is dedicated to performing stateless computing tasks is becoming increasingly common. AWS Lambda is a prime example of FaaS.

Usage Scenarios and Context Today

Disintermediation and user self-service are primary requirements of modern application systems. It is simply unacceptable, today, for a company to fail to remove unnecessary human impediments between itself and its customers, vendors, business partners, employees and associates and their goals and requirements.

A common option is to provide information or services that would traditionally be serviced through a remote application as an API that an external user can call and incorporate into his environment as best suits his companyâ€™s needs. ***Thus, what were applications may now be services, a seminal theme of todayâ€™s architecture.***

The lean startup process of establishing and growing a business, even a new business unit within a larger company, is predicated on, among other things, minimizing unnecessary investments, such as defining and building operating enablement. The focus for startups must be to develop and mature the business model and diverting attention and resources to operational issues not directly related to delivering products and services is antithetical to that. Simply put, a minority of startups succeed for any length of time and none that do look like what they were originally expected to after they undergo a few inevitable rounds of evolution.

Ultimately, two very important aspects of what is possible today drive approaches to technology-driven business evolutionâ€”***velocity*** and ***granularity.*** The ability to erect systems and attach them to enormous sources of data with remarkable speed, coupled with the ability to put them into production in a nascent but usable state and evolve them in place in discrete steps result in being able to change the

tires while the car is driving down the highway. This has profound implications and impacts across almost any product or service delivered in the economy of the US and world markets.

Juxtaposed with this are many corporate infrastructures rife with now-outmoded, yet still-operable foundational components in companies that had focused relentlessly on minimizing redundancy for many years. The result of this was large shared-services organizations with substantial pools of dedicated assets and core applications that are relied on by many different business units. This can become a substantial drag on their ability to reorganize or effectuate acquisitions or divestitures, events which are increasingly common today.

The need to keep things running under the pressure of doing more with less has resulted in outsourcing, loss of in-house expertise and eliminating staff needed to effectuate any transition, to say nothing of something as important and broad in scope as the digital transformations that most established corporations face today. A number of industry analysts and strategy consultancies have opined on the issue of how to deal with this and one thing some of them have advised is *bimodal IT*, an approach in which some ICT resources are focused on older technologies and systems built in them while the remainder is focused on mastering the latest and greatest and cutting a new path. To put it bluntly, many analysts think bimodal IT is a terrible idea (see this Forbes article) and I agree with them, for reasons I will address later.

The Challenges

In the end, the scope of the challenges includes opportunities and threats, risks and rewards:

- Aging systems represent a significant drag on agility, engendering excessive costs and requiring an aging and diminishing pool of support staff to run.
- Past efforts to minimize redundancy and share resources among business units has made it more difficult to separate them from one another and complicates reorganization or divestiture when it is advantageous.
- The advantages in speed, agility and cost reduction available through adopting modern technology, architecture and cloud-based implementation are creating increasing strategic pressure on enterprises that donâ€™t employ them. Companies not burdened with outdated information technology are at a significant advantage.
- Determining how to execute a transformation from legacy infrastructure to embrace current tools and practices is a fraught decision process. Facing the challenge, deciding how much to bite off and how quickly to proceed can be an existential challenge.

In the two final posts of this series, we will explore these issues.

Date Created
2019/06/11
Author
howardmwiener