Five Fundamental Distinctions of Enterprise Software

by Mike Bergman - Monday, January 13, 2014

http://www.mkbergman.com/1703/five-fundamental-distinctions-of-enterprise-software/

Consumer Trends are Manifest, but Enterprise Software Has Its Own Imperatives

The end of the year is always the silly season for technology pundits. To gain attention, it is often the "end of this", the "death of that" or new paradigms or revolutions. Granted, it is hard to get attention when everyone is pontificating on this or that. But hype and hyperbole do not serve helping users understand fundamental changes in the marketplace.

This year's silliness award goes to Brian Profitt of ReadWrite Web who opined 2014 as the death of the distinction between consumer and enterprise software, stating, "legacy enterprise vendors need to serve business and consumers alike, or risk becoming roadkill." Balderdash. (And bunkum, BS and brimborion if one wants to be alliterative.)

PCs thirty years ago, local networks twenty years ago, the Web ten years ago, or cloud computing or smartphones today did or will not "kill" enterprise software. Consumer applications and technology will continue to point the way to important new trends. But the fundamental distinctions of enterprise software will also live on. Let's look at five of these distinctions [1].

1. The Buying Process

Consumer software is an individual purchase; enterprise software purchases are on behalf of a group. That means enterprise sales need to involve many more decisionmakers, some or all of whom may not be the actual users, as when IT is the *de facto* purchasing agent. Multiple perspectives need to be brought to bear on the enterprise acquisition. Often a single negative voice is sufficient to scuttle a sale. On the other hand, consumer software may be free, notably lower cost, or acquired on a whim.

Traditionally this has led to longer decision cycles and the need to employ dedicated reps for enterprise sales. Though SaaS (software as a service) or PaaS (platforms as a service) can lower initial acquisition costs and improve the fundamental business model, adoption of enterprise software still is a group decision in the enterprise. Enterprises well know that initial adoption carries longer-term costs in integration, interoperability, training and documentation. Software may be "legacy" in the enterprise because of these lifecycle realities and costs.

2. Enterprise "-ilities"

Enterprises, then, in representing the interests of groups or organizations, also have requirements that extend beyond what an individual consumer requires. Many of these correspond to the well-known "-ilities" -- reliability, scalability, operability, interoperability, maintainability, and availability. An individual consumer is inconvenienced when there may be failures along any of these dimensions. The enterprise experiences costs, risks or lost opportunities when they occur. In other words: money.

These "-ilities" place a premium on testing and documentation, as well as lead to often requiring a longer-term relationship with the software vendor or its representatives. Because of the financial impacts from failures in "-ilities" it is often necessary to have support agreements or contracts in place to insure risks. The "-ilities" also place additional code and testing requirements upon the software.

3. Security

Though often lumped in with the "-ilities", security is an additional enterprise requirement that warrants its own distinction. Whether profit or non-profit, all enterprises are unique, with potential proprietary information both internally and externally (with the public or possible competitors). Though individual consumers also have requirements for privacy and confidentiality, these information flows are strictly between the individual and outside entities. In an enterprise, access may occur and be between many internal individuals and all of their external contacts.

The nature of individual consumer security is more like a ring or protective shell. In enterprises, security must be built fully "into the cake", capturing distinctions between applications, databases, datasets, and access and modification rights or not at all levels. Like the other "-ilities," the enterprise security requirement leads to a much different development and coding model than consumer software. And, frankly, it leads to higher development costs.

Initially, these hurdles were some of the causes for slower adoption of open source within enterprises. We are also learning better architectural designs and reliance on APIs that are aiding fulfilling these enterprise requirements at lower costs with greater sustainability. But the importance of security to the enterprise remains.

4. Governance

Security, the "-ilities", and ongoing reliance on legacy enterprise systems also mean that repeatable workflows and governance need to be at the core of enterprise software use. Are things working well? Where are they breaking down? Need improvement? How can we incorporate a constant influx of new users? How can we manage actual costs and effectiveness?

Any enterprise that needs to maintain a competitive or sustainability edge must be able to address these questions. For software, this means versioning, documentation and training of same, and means to track use and misuse. (Not to mention the additional workflow software to manage these processes.) Every effective enterprise understands that what is not measured can not be managed.

These training, versioning and logging requirements are essential to effective governance of software and the information upon which it operates. These, requirements, too, are different than what an individual needs or wants. They, too, add to costs and requirements above normal consumer software demands.

5. Business Model

These enterprise distinctions help bound the kind of business models that may be applied to enterprise software. Enterprise software requirements are higher and more demanding (and take longer to bring to fruition) than consumer software requirements. Support, longevity, reputation and quality are important factors for software vendors to fulfill in order to overcome legitimate risk questions enterprises ask when contemplating a new (potential legacy) commitment to a new enterprise software adoption.

Fortunately, as systems have become more open with a new architectural model based on the distributed Web, many older enterprise hurdles can now be more readily overcome. These advances are unalloyed goodness. But enterprise imperatives still remain.

We can hope with less risky SaaS or PaaS that much can be done to reduce initial acquisition costs and risks. Open source software is also lowering the cost of initial enterprise software development by orders of magnitude [2]. Nonetheless, higher costs with support commitments distinguishes enterprise software business models from any of the consumer kind. I expect that fundamental distinction to remain.

Consumer Trends DO Affect the Enterprise

These five factors, or other splits that could be reasonably made, are not meant to deny the importance of consumer software. Merely, the point is that enterprise software has its own set of imperatives. Enterprise software is certainly more conservative and slower-paced for the exact reasons of its distinction from consumer software. Talk of convergence or the "consumerization" of enterprise software misses these distinctions and what will continue to be the fundamental differences between the two software categories.

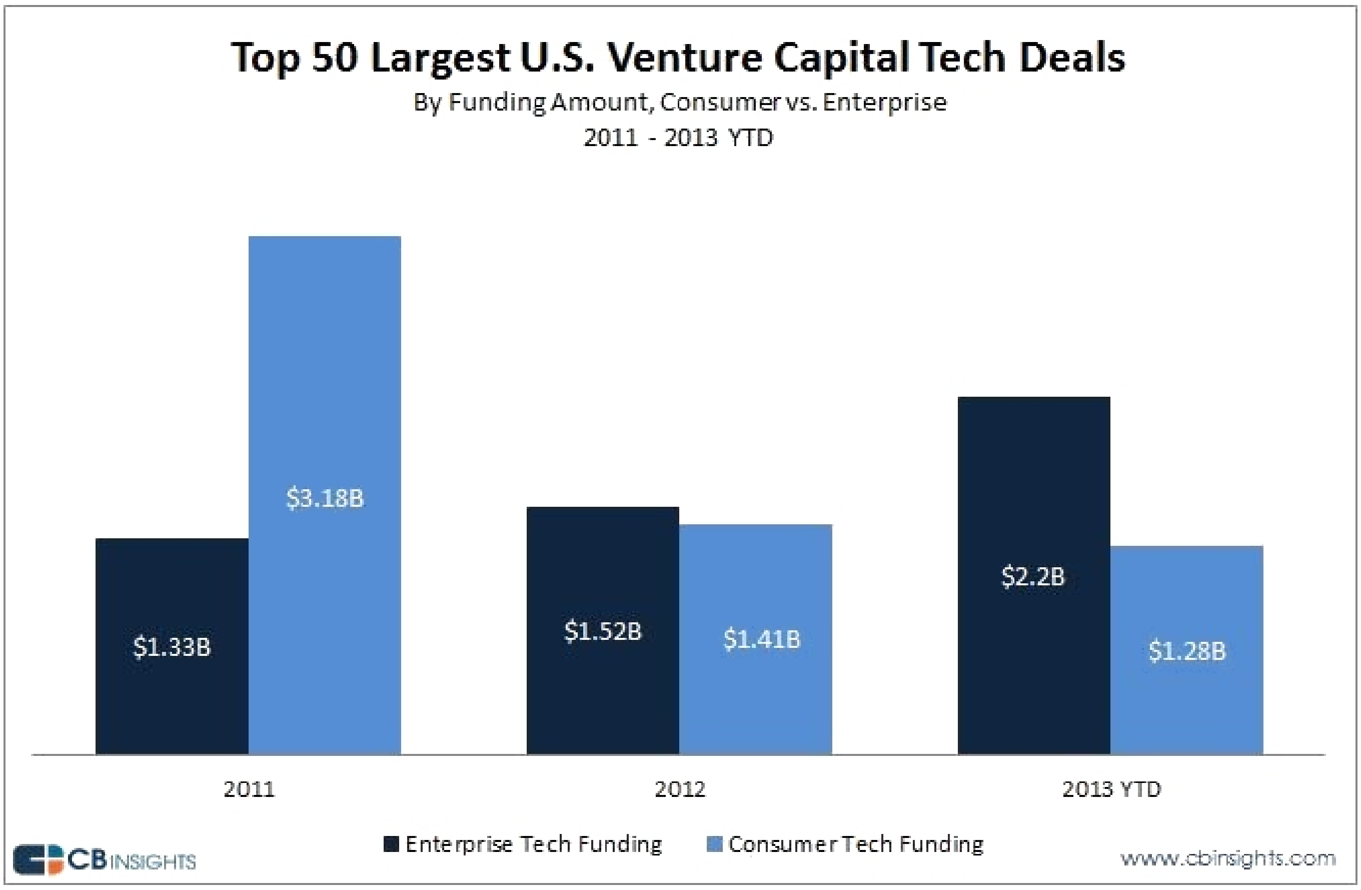
Because of its lesser requirements, meaning in economic terms "lower barriers to entry," we will also see that consumer software and its devices will be the lodestar for innovation. In my own thirty years in this space, we have seen consumer leadership in device form factors (PCs to smartphones), architecture (Web, APIs and distributed networks), user interfaces (browsers and HTML), data and data models (RDF, XML, JSON), programming languages (scripting, Ruby, Python), business models (open source, cloud computing), software models (apps, SaaS, PaaS), etc. Enterprise software is, by and large, a sink for consumer innovations, not a source.

But to be successful in the enterprise, those innovations must also meet more stringent requirements. And, some of those requirements, such as interoperability, are clearly driven more from the enterprise side of things.

Thus, silly talk about consumer versus enterprise markets, framed as either "death" or "convergence," really misses the point. Ultimately, they are different markets with different imperatives. Yes, there is a synergy and natural relationship -- after all code and devices may be shared in either realm -- but the roles and contributions of each differ. Though I don't deny that some innovations may work equally well in either the consumer or enterprise markets, most innovation will occur in the consumer sector, while higher revenues and income are to be derived from the enterprise sector.

Today's Enterprise Picture

Despite the silly punditry noted above, major industry analysts and the venture capital community are signalling a shift from the consumer to the enterprise market. Gartner, for example, sees a doubling in enterprise software growth to 5.8% in 2014 over other IT expenditures. CB Insights points to a dramatic shift in venture capital support for enterprise software versus consumer over the past two years [3]:



In 2013, about 70% of VC software funding went to startups building tech for businesses. (Actually, the shift was much greater in that $450 million of the consumer total went to just two consumer companies, Uber and Pinterest.) VC funding for enterprise software has risen 65% in the past two years; meanwhile, funding of consumer software by VCs has dropped 60%.

Besides the crowded consumer space and perhaps steam being lost behind social networking, these trends suggest that consumer innovations of the past few years are now ripe for "enterprising" within the enterprise market. What can be taken from the consumer side now must be looked at for incorporation and adoption on the enterprise side. This is not the "consumerization" of the enterprise, but the "enterprising" of consumer innovations.

This distinction is important. Adoption of prior consumer innovations will not occur via osmosis ("consumerization"), but by purposeful re-packaging and modification of those innovations to meet enterprise requirements ("enterprising"). That is, those successful in leveraging consumer innovations into the enterprise will do so purposefully by adapting to enterprise imperatives. The target-rich environment of the next couple of years will be adopting prior consumer innovations to the enterprise.

[1] For some recent further reading on this topic, see 5 Best Kept Secrets About Enterprise Software, The Difference Between Consumer and Enterprise Software, The Difference Between Enterprise and Consumer Software, Enterprise Web vs Consumer Web [2.0]: Top Six Differences, and Forget Virality, Selling Enterprise Software Is Still Old School.

[2] My first enterprise software company from the early 1980s required more than $1 million in start-up software development funds; more recent experience has been on the order of $50,000 to $100,000.

[3] Other recent commentaries on the shift to enterprise software include Enterprise Software Spending Will Set Pace of IT Spending in 2014, VCs Are Tripping Over Themselves To Fund Enterprise Startups, Research Shows, and As the Pendulum Swings—Back to Enterprise Software?

[4] Opening image courtesy of Willow and Stone, UK.