January 24, 2011

The Seven Qualities Of Wildly Desirable Software

by Mike Gualtieri

for Application Development & Delivery Professionals

January 24, 2011

The Seven Qualities Of Wildly Desirable Software
To Deliver Extraordinary Software, Design To Balance These Seven Qualities

by Mike Gualtieri
with Mike Gilpin and Adam Knoll

EXECUTIVE SUMMARY

Great software applications seldom happen by accident: Wise design decisions are the key. Yet many application development teams only design for one, two, or three of these qualities, and others have strayed from design altogether as teams, distracted by shiny new technologies, methodologies, and platforms, fail to engage design principles and talent effectively. Is your business demanding more innovation? You must respond by rising above mediocrity to deliver great software, and to achieve this, you must balance your design to provide seven key qualities: user experience, availability, performance, scalability, adaptability, security, and economy. Boost your commitment to design and design for all seven qualities to create better software, faster.

NOTES & RESOURCES

Forrester conducted architecture assessment advisories with dozens of clients and spoke with leading experts in design, design process, and software development, including practitioners and academics. In addition, we reviewed relevant third-party research, books, and articles on software design and design in general.

Related Research Documents
"Best Practices In User Experience (UX) Design"
September 4, 2009

"Use Threat Modeling To Develop More-Secure Applications"
March 10, 2009

"Best Practices: Attaining And Maintaining Blazing Fast Web Site Performance"
February 4, 2009

EXTRAORDINARY SOFTWARE IS THE RESULT OF SMART DESIGN DECISIONS

Great software seldom happens by accident; it is the result of designers and architects' many, many smart design decisions.[1] The design decisions you make — or fail to consider — will determine the success, failure, or mediocrity of your software creation. Many application development teams think they are already doing design, yet development teams are often guilty of one or more of the following offenses:

- **Treating design as merely a required document in the process.** One of development teams' most heinous offenses is perceiving design as just another document they have to complete, one in a long list of documents their organization's software development life cycle (SDLC) process requires. Such teams may assign responsibility for writing the design document to someone who doesn't actually do any design work, as everyone expects that no one will read it.

- **Valuing process more than design.** Process is important, and design is an important part of the process. But, with Waterfall being a perennial disorder and the Agile approach often interpreted as "let's write code as soon as possible," design often gets short shrift. Which is more important: process or design? The answer is both, but too often process has a capital "P" and design gets a lowercase "d."

- **Focusing on only a few aspects of design.** Thankfully, many application development teams do focus on design. But teams sometimes focus on one aspect of the design while ignoring other important design decisions. Classic design myopias include designing services (as in service-oriented architecture [SOA]) but ignoring the user experience design, or designing for user experience but ignoring scalability. At least with design myopia, part of the application will turn out well — although that still may not be enough to avoid overall failure.

- **Tasking the wrong people with design.** Design is about making decisions. Great design comes from people who can enumerate the options, try decisions on for size, and make informed choices that fit within design constraints but that are consistent with the vision for the final product. Many application development teams simply don't have or don't recognize the design talent needed to make great design decisions. Furthermore, some design decisions necessitate consulting multiple experts, requiring a level of design collaboration that is absent from many organizations.

- **Confusing design with architecture.** Architecture is the result of design decisions. Choosing an architecture means making some design choices, but others decisions remain before the design is complete. Application development professionals often confuse choosing platforms such as Java or .NET with design, failing to give adequate attention to the many other design decisions that they must still make, such as user experience and data design.

- **Selecting the wrong design patterns.** Design patterns can be shortcuts if they are relevant to the application you are developing. However, all too often, application development

professionals take inappropriate shortcuts, using design patterns from academic sources or sample code from books or Web sites that is ill-suited to the application they are building. For example, a team might choose a content management design pattern for an eCommerce application, leading to inadequate resilience in managing transactions.
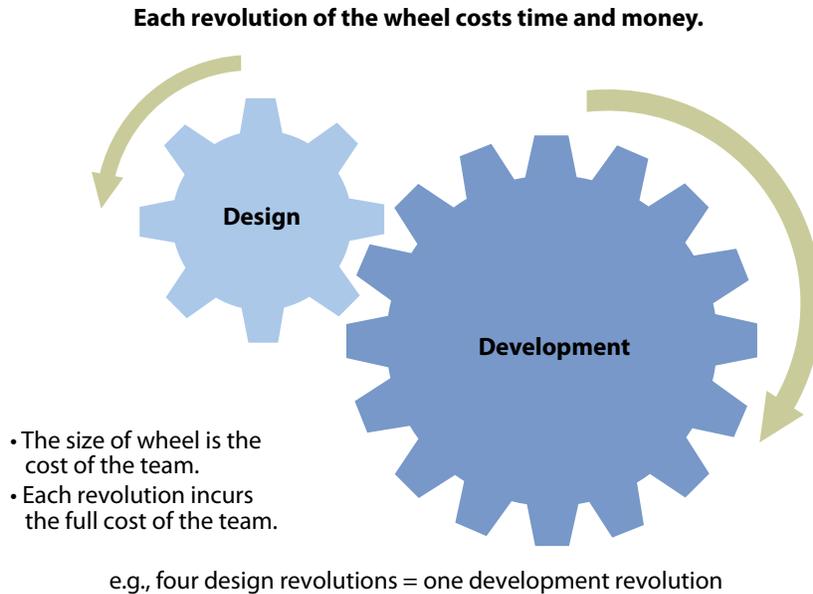
Boost Your Commitment To Design To Improve Quality, Speed Delivery, And Reduce Costs

Gone are the days when application development and delivery teams could cavalierly ask the business to pick two: quality, time, or cost. The business wants and needs all three. Application development teams have responded by experimenting with or adopting Agile development practices and choosing fit-to-purpose development platforms. Now it is time for design to take the driver's seat. Teams that boost their commitment to design will:

- **Improve quality.** Quality is about much more than eliminating defects: Quality software meets requirements, provides a satisfying user experience (including meeting performance goals), and has fewer defects.[2] But quality has to be designed in; it doesn't happen by accident.

- **Speed delivery.** Application development teams are always under pressure to deliver faster without sacrificing quality. Informed design decisions upfront can reduce development process iterations. Iterative development doesn't mean you want more iterations; you want the smallest number of iterations that will deliver a quality result (see Figure 1).

- **Reduce costs.** Poor or nonexistent design decisions can result in gobs of costly rework. It is much less expensive to change your design *before* you implement the application (see Figure 2).

Caution: Don't be lulled into thinking that waterfall automatically means you have strong design or that Agile eliminates the need for design; process is not a substitute for design. If you have ever driven in Boston, then you know that the streets are very difficult to navigate. Why? In colonial Boston, someone built a house. And then someone else built a house. Then they built a path between the two houses. And so on and so forth. Boston's layout was accidental, and the result is a convoluted set of streets that frustrates both residents and visitors. Similarly, application development teams achieve poor results when they develop without a plan, leaving software design to chance.

Figure 1 Good Design Can Result In Fewer Development Iterations

Each revolution of the wheel costs time and money.

- The size of wheel is the cost of the team.
- Each revolution incurs the full cost of the team.

e.g., four design revolutions = one development revolution

58115   Source: Forrester Research, Inc.

Figure 2 Estimates Of Relative Cost Of Rework

Design and architecture	Implementation	Integration test	Customer beta test	Post product release
1X	5X	10X	15X	30X

Note: X is a normalized unit of cost and can be expressed in terms of person-hours, dollars, etc.

58115   Source: Forrester Research, Inc.

BALANCE THE SEVEN QUALITIES OF WILDLY DESIRABLE SOFTWARE

How do you know your software is great? Forrester has identified seven key qualities that all applications must exhibit, basing our choices on research and inquiries on software design, architecture, and process; architecture assessment advisory with dozens of clients; and interviews with leading experts, including both practitioners and academics. After seeing much successful and unsuccessful software, we conclude that designers must balance across all seven qualities to reflect the business priorities the software will serve.

Forrester defines the seven qualities of software as (see Figure 3):

> *The common requirements that all software applications must satisfy to be successful: user experience, availability, performance, scalability, adaptability, security, and economy.*

Figure 3 The Seven Qualities Of Extraordinary Software

Quality	What it means
1. User experience	Users' perceptions of an application's usefulness, usability, and desirability based on the sum of all direct and indirect interactions.
2. Availability	An application's readiness to perform its functions when needed
3. Performance	The speed with which an application performs a function that meets business requirements and user expectations
4. Scalability	An application's ability to handle increasing or decreasing volumes of transactions, services, and data
5. Adaptability	The ease with which an application's functionality can be changed or extended
6. Security	Mitigating the risk of attack and ensuring confidentiality, integrity, authentication, authorization, and nonrepudiation
7. Economy	Minimizing the cost to build, operate, and change an app without compromising its business value or any of the other six design qualities

58115   Source: Forrester Research, Inc.

Quality No. 1: User Experience
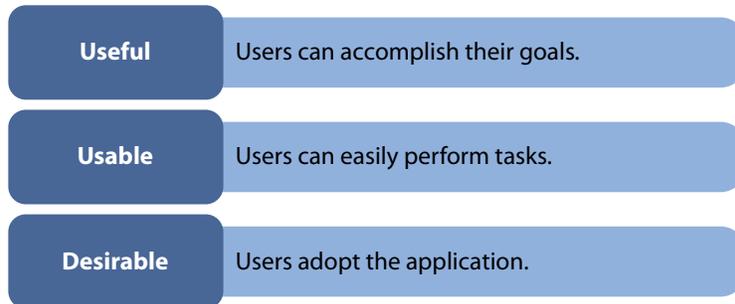
Forrester defines user experience as:

> *Users' perceptions of an application's usefulness, usability, and desirability based on the sum of all direct and indirect interactions.*

All seven qualities are important, but if you get the user experience (UX) wrong, nothing else matters. The UX is the part of your application that your employees and/or customers see and use daily.[3] You can do an exceptional job on project management, requirements gathering, data management, testing, and coding, but if the user experience is poor, your results will be mediocre — or even a complete failure.

You have to get the user experience right to create an extraordinary application, but, paradoxically, it is the one quality that most development teams are least adept at achieving. This is not because they are not genetically capable of designing great user experiences; it's often because no one ever taught them how. Forrester's "Best Practices In User Experience (UX) Design" is written to help application development professionals learn these critical design skills.[4] Extraordinary user experiences must be useful, usable, and desirable (see Figure 4).

Figure 4 Extraordinary User Experiences Are Useful, Usable, And Desirable

Useful	Users can accomplish their goals.
Usable	Users can easily perform tasks.
Desirable	Users adopt the application.

58115   Source: Forrester Research, Inc.
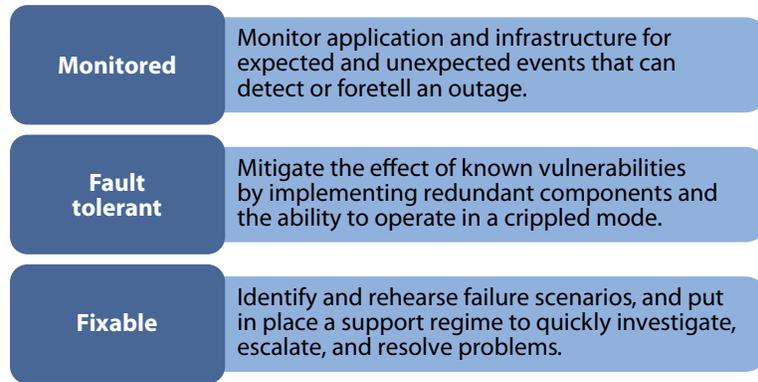
Quality No. 2: Availability

Forrester defines availability as:

An application's readiness to perform its functions when needed.

Applications are susceptible to many kinds of failures that can cause a complete outage or cripple critical functions. The most-devastating failures occur when data centers lose power or cooling. Similarly debilitating are storage crashes, network appliances run amok, and the occasional backhoe that severs your connection to the Internet backbone. Ouch! Adding to inadvertent threats to application availability are malicious attacks such as distributed denial of service (DDoS). However, failures caused by complex platforms and code are more common, with many modes of failure due to bugs, upgrades, and configuration changes. Highly available applications must be monitored, fault tolerant, and rapidly fixable (see Figure 5).

Application managers measure availability as a percentage of uptime (see Figure 6). For example, 99% uptime equates to fewer than 3.65 days of downtime per year. Developers design safety critical applications such as those used by air traffic controllers to achieve zero downtime by provisioning completely redundant systems. An eCommerce site that is down for an hour on Cyber Monday could cost millions of dollars in lost sales.[5] Yet the cost to avoid downtime can be prohibitive for many firms because it requires hardened or redundant data centers.[6] Therefore, system designers must provide as much availability as their business demands but no more than it can afford.

Figure 5 Highly Available Applications Are Monitored, Fault Tolerant, And Fixable

Monitored	Monitor application and infrastructure for expected and unexpected events that can detect or foretell an outage.
Fault tolerant	Mitigate the effect of known vulnerabilities by implementing redundant components and the ability to operate in a crippled mode.
Fixable	Identify and rehearse failure scenarios, and put in place a support regime to quickly investigate, escalate, and resolve problems.

58115   Source: Forrester Research, Inc.

Figure 6 What Does High Availability Mean?

Uptime%*	Downtime per year
99.999% (five 9s)	5.26 minutes
99.99% (four 9s)	52.6 minutes
99% (two 9s)	3.65 days
98%	7.30 days
95%	18.25 days

*Uptime calculations assume no scheduled downtime.

58115   Source: Forrester Research, Inc.

Quality No. 3: Performance

Forrester defines performance as:

> *The speed with which an application performs a function that meets business requirements and user expectations.*

For interactive applications, performance measures the response time of the application in completing transactions from the user's point of view. For other applications, performance measures the ability of the application to process the necessary transaction volume in the required time window. Performance can degrade if designers do not make the application or its infrastructure scalable; with applications that are not scalable, additional load can bring decreased performance.
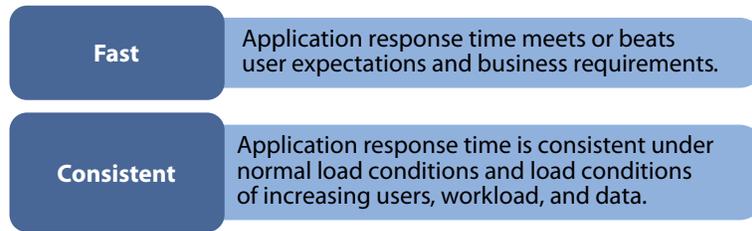
Performance can also be critical for applications that don't interact directly with users. For example, an algorithmic currency trading application must be able to issue buy and sell orders in milliseconds

in order to make or prevent losing big dollars. Well-performing applications must meet or exceed the expectations of users and be consistent under normal and peak load (see Figure 7). Achieving good performance requires not only careful design but also a performance-driven approach to software development.[7]

Figure 7 Performance Must Meet Or Exceed Requirements And Be Consistent

| Fast | Application response time meets or beats user expectations and business requirements. |
| Consistent | Application response time is consistent under normal load conditions and load conditions of increasing users, workload, and data. |

58115   Source: Forrester Research, Inc.
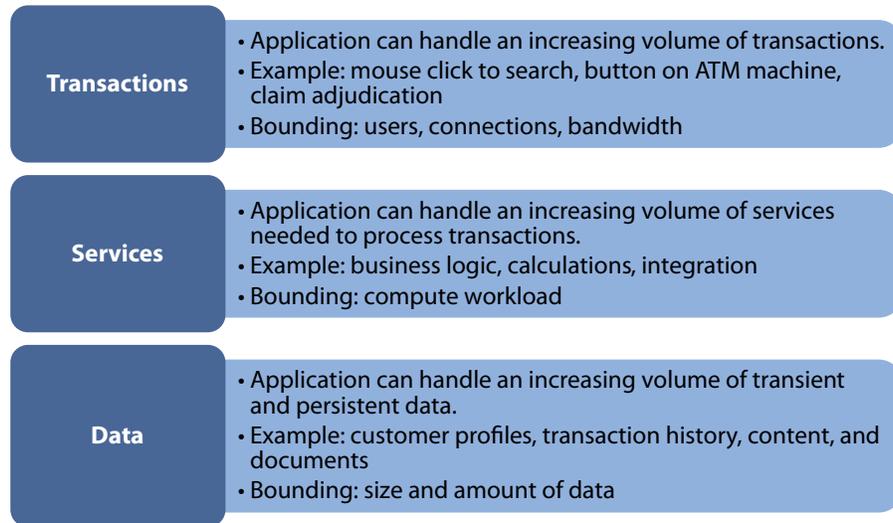
Quality No. 4: Scalability

Forrester defines scalability as:

An application's ability to handle increasing or decreasing volumes of transactions, services, and data.

When teams fail to design applications to scale, they often run into performance or availability problems that in turn can lead to a degraded user experience. Designers must plan for usage peaks and valleys. For example, streaming video volume is greatest during the evening and on weekends, and many eCommerce sites do 70% of their business between Cyber Monday and the end of the year.[8] Applications must scale on three dimensions: transactions, services, and data (see Figure 8).

Scaling down is as important as scaling up. Applications that provision infrastructure to support peak transaction volumes yet do not scale back on resource demands when volumes dwindle waste money. Techniques to scale down range from those appropriate to internal clusters to those relevant in the cloud, including mixed models such as cloudbursting.[9] Savvy designers have to stay abreast of the latest techniques relevant to their business to be sure applications scale up and down cost-effectively.

Figure 8 The Three Dimensions Of Scalability

Transactions	• Application can handle an increasing volume of transactions. • Example: mouse click to search, button on ATM machine, claim adjudication • Bounding: users, connections, bandwidth
Services	• Application can handle an increasing volume of services needed to process transactions. • Example: business logic, calculations, integration • Bounding: compute workload
Data	• Application can handle an increasing volume of transient and persistent data. • Example: customer profiles, transaction history, content, and documents • Bounding: size and amount of data

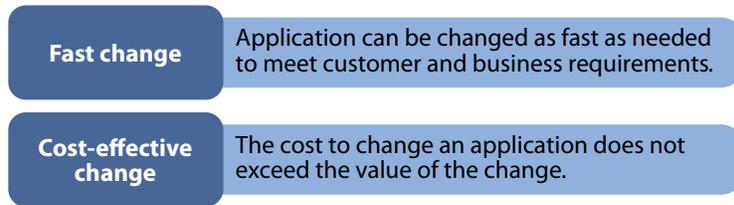58115   Source: Forrester Research, Inc.

Quality No. 5: Adaptability

Forrester defines adaptability as:

> *The ease with which an application's functionality can be changed or extended.*

Software is seldom static. Ongoing changes in customer needs, business operations, competition, and technology motivate the need for adaptability. Smart developers design software applications to allow application development professionals or sometimes empowered business professionals the ability to quickly change applications, with special attention to those areas that change most often (see Figure 9).

Designing for adaptability often requires teams to employ some of the most recent innovations in the architecture of application infrastructure, including business process management (BPM), business rules, and event processing.[10] But not every application will need to use these sophisticated techniques; some applications can use time-honored techniques such as data-driven behavior to meet their need for adaptation. Yet it's surprising how seldom teams think about the need for future change in the applications they are building. If you are building applications for anything other than trivial throwaway uses, make adaptability a key part of your design process.

Figure 9 Adaptability Means Fast, Cost-Effective Change

Fast change	Application can be changed as fast as needed to meet customer and business requirements.
Cost-effective change	The cost to change an application does not exceed the value of the change.

Quality No. 6: Security

Forrester defines application security as:

> *Mitigating the risk of attack and ensuring data confidentiality and privacy, transaction integrity, user authentication and authorization, availability of systems, and nonrepudiation of actions.*

Too many application development professionals have a naive notion of application security that lulls them into thinking they have all the security bases covered. This means that security vulnerabilities are often caught late in the software development life cycle — or, heaven forbid, the vulnerabilities become a feature story on the front page of *The Wall Street Journal*. To build secure applications, designers must first evaluate the risk of the various threats that their application faces and then design strategies to mitigate those risks to an acceptable level.[11] The goal? To ensure data confidentiality, integrity, and privacy and to provide resilience to denial of service attacks, authentication and authorization of users, and auditing of application use (see Figure 10).

Figure 10 When Building Applications, Consider Six Security Properties

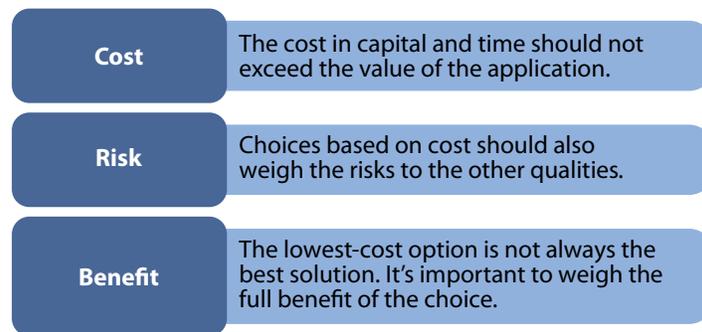| Security property	Description
Confidentiality	Information is available only to the people intended to use or see it.
Integrity	Information is changed only in appropriate ways by the people authorized to change it.
Availability	Applications are ready when needed and perform acceptably.
Authentication	A person's identity is determined before access is granted if anonymous people are not allowed.
Authorization	People are allowed or denied access to use the application or application resources.
Nonrepudiation	A person cannot perform an action and then later deny performing the action.

Quality No. 7: Economy

Forrester defines economy as:

> *Minimizing the cost to build, operate, and change an application without compromising its business value or any of the other six design qualities.*

Every design decision carries a cost. Whether it is the choice of technology, the decision to provision an additional data center or deploy to the cloud, or the choice of application architecture, every option carries different cost implications. For example, the power of cloud economics is elasticity — the ability to pay for resources only when needed and to scale infrastructure up and down, on demand.[12] Architects can design applications to throttle up during normal traffic loads and floor it during unusual peak periods. You don't need to buy or rent additional infrastructure in case traffic volume increases — only if it does increase.

Your design decisions shape the cost/benefit/risk trajectory over your application's useful life. Designers must actively manage application economy to ensure financially sustainable design decisions (see Figure 11).

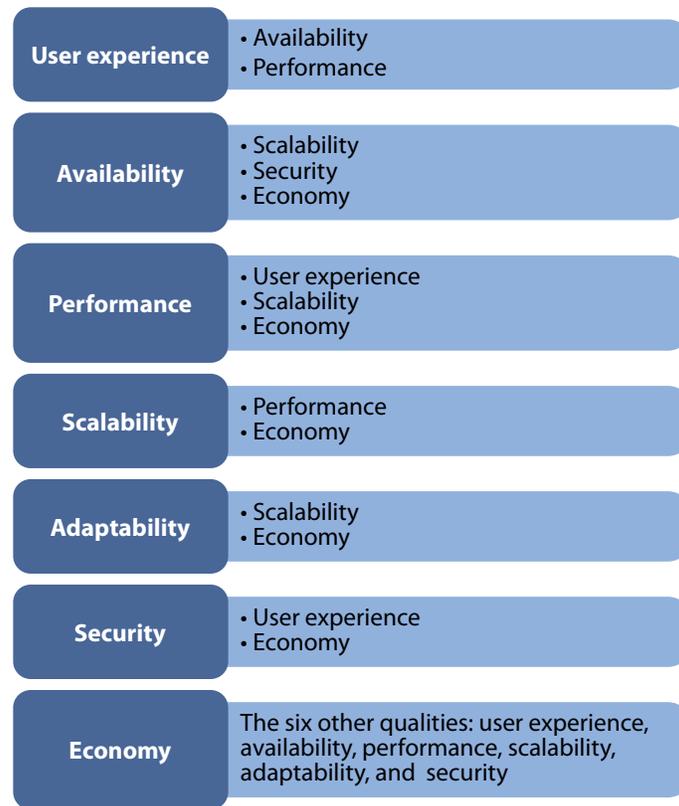**Figure 11** Every Design Decision Has Cost, Risk, And Benefit

Cost	The cost in capital and time should not exceed the value of the application.
Risk	Choices based on cost should also weigh the risks to the other qualities.
Benefit	The lowest-cost option is not always the best solution. It's important to weigh the full benefit of the choice.

58115   Source: Forrester Research, Inc.

EVALUATE YOUR DESIGN TO FIND THE RIGHT BALANCE

The perfect design is the one that achieves the right balance among the seven qualities of extraordinary software. To find the right balance, you must first establish your design goals by deeply analyzing the requirements and knowing your constraints. From these design goals, you can ideate design solutions, evaluate them, and then iterate until you find the right solution or run out of time.[13] To evaluate your design solutions, use an assessment framework to ask a series of questions about the seven qualities, considering each dimension and the balance among them (see Figure 12).

Figure 12 Design Qualities Often Have Tradeoffs

Quality	Tradeoffs
User experience	• Availability • Performance
Availability	• Scalability • Security • Economy
Performance	• User experience • Scalability • Economy
Scalability	• Performance • Economy
Adaptability	• Scalability • Economy
Security	• User experience • Economy
Economy	The six other qualities: user experience, availability, performance, scalability, adaptability, and security

58115   Source: Forrester Research, Inc.

EMPOWER A TWO-PERSON TEAM TO DESIGN FOR THE SEVEN QUALITIES

It takes a rare talent to design extraordinary applications that perfectly balance the seven qualities: user experience, availability, performance, scalability, adaptability, security, and economy. Design is a collaborative effort among many stakeholders and experts. But avoid making final design decisions by committee: Instead, choose two design leads who have the power to make the final design decisions. Why two? Traditional application architects often lack user experience design expertise, and user experience designers often lack technical expertise. Frederick P. Brooks, Jr. says "two-person teams are magical."[14] A team of two designers can edit each other's ideas and bring out the best in each other, resulting in the smartest design decisions. In his memoir *Life*, when discussing his relationship with Mick Jagger, Rolling Stones guitarist Keith Richards also points out the power that teams of two can have, explaining: "I pull things out of him; he pulls things out of me."[15] Consider a two-person design decision team for your software development efforts. Seek for your design leads:

- **A user experience designer.** The user experience designer will take the lead in ensuring that the software is useful, usable, and desirable. This requires a deep understanding of both the business requirements from business analysts and the insights from user research. UX designers' skills run the gamut, including ethnographic research, interaction design, visual design, usability, and sheer creativity. You need an individual who can understand all the aspects of UX design, listen to other team members, and make an informed decision in collaboration with the application architect.

- **An application architect.** The application architect will take the lead on the more technical aspects of the design, including availability, performance, scalability, adaptability, and security. The application architect may only have deep expertise in two or three of these areas, so it is important for her to reach out to other experts. For example, the application architect may be an expert in SOA for adaptability but need to reach out to other team members for security and performance. The important thing is that the application architect is the decision-maker but draws from other experts to help with those decisions.

Most importantly, the design leaders must solicit options and opinions from the extended design team, experts, users, and business stakeholders to make well-informed design decisions. The result? The team will achieve the application's design goals and find the right balance among the seven qualities of extraordinary software, increasing the chances of delivering a successful solution.

Don't Let Process Evangelists Diminish The Value Of Design

Software design professionals should have the most prestige among application development professionals — so boost your commitment to design and find the design talent to staff accordingly.

And don't forget: Wildly desirable means that you must design for all seven qualities — but if you get the user experience design wrong, then nothing else matters. Please see Forrester's report entitled "Best Practices In User Experience (UX) Design" to learn how to design exceptional user experiences.

W H A T   I T   M E A N S

PROCESS IS BANKRUPT WITHOUT DESIGN

Application development teams must not fall into the "process trap": thinking that if they follow a software development process, then magic will happen. Process is very important, but without smart design, even the most flawlessly executed process can result in increased costs, longer time to deliver, and poor quality.

Design Must Set The Course; Process Helps Get You There

Try giving a film director 120 blank pages and say, "make this movie." That is exactly how many firms start application development projects. A great design is like a great movie script — it tells

the entire story. The director must have a script that describes the story — the setting, characters, and scenes. The script does not tell the director how to make the movie. The director interprets the story and collaborates and fills in the details to create the movie. Similarly, application development teams must have a solid design to be successful. And just as scripts often change during the filmmaking process, software design can and should iterate during development as teams discover new requirements or refine design decisions.

ENDNOTES

[1] Forrester uses the terms architect and designer synonymously. Whether you call them architects or designers, both make design decisions, the result of which is an architecture. Oxford Dictionary defines design as: A plan or drawing produced to show the look and function or workings of a building, garment, or other object before it is built or made. It defines architecture as: The complex or carefully designed structure of something. Source: Oxford Dictionaries website (http://oxforddictionaries.com/view/entry/m_en_us1239718#m_en_us1239718 and http://oxforddictionaries.com/view/entry/m_en_us1222777#m_en_us1222777).

[2] Forrester defines quality software as: Software that meets business requirements, provides a satisfying user experience, and has fewer defects. See the August 11, 2010, "Seven Pragmatic Practices To Improve Quality" report.

[3] Many applications do not have or need direct user interaction. For example, an application programming interface (API) provided by a consumer credit bureau may expose application functionality to a bank. The "user" of the application is an application development team at a bank that wishes to include external credit checks in their loan application process. Even though the API might not directly interact with users, at some point it will touch another application that does have direct contact with a user. In this case, the usefulness and usability of the API will determine how the bank's application developers can build an application with an exceptional user experience. Another example is a batch application that calculates investment portfolio risk. Although there is no direct user interface in this application, the real-time portfolio risk is used to make decisions about securities trades that result in profit or loss.

[4] Forrester published a report detailing how to design compelling user experiences. See the September 4, 2009, "Best Practices In User Experience (UX) Design" report.

[5] Cyber Monday is a marketing term for the Monday immediately following Black Friday, or the Friday after Thanksgiving.

[6] Source: Bruns-Pack Data Center Design/Build Solutions (http://www.bruns-pak.com/).

[7] When software automates business processes, software performance is the limiting factor for business performance. A slow order processing engine necessarily means slowly processed orders. Even though software performance matters to the business, it's low on the priority list for most application development organizations. To meet business needs — and to improve the efficiency of their own operations — development shops must mature their performance practices from pure firefighting to performance-driven development. See the February 28, 2006, "Performance-Driven Software Development" report.

8   Source: "After Black Friday comes Cyber Monday," CNNMoney.com, November 28, 2005.

9   USA.gov, one of the busiest US government Web sites, has achieved significant cost savings by embracing cloud computing. The US General Services Administration (GSA) has migrated all of the core resources of the USA.gov Web portal to Terremark's IaaS platform, The Enterprise Cloud. By using The Enterprise Cloud, USA.gov can maintain a small persistent footprint and deploy on-demand scaling as traffic fluctuates. See the September 25, 2009, "Case Study: USA.gov Achieves Cloud Bursting Efficiency Using Terremark's Enterprise Cloud" report.

10 Most business applications are too inflexible to keep pace with the businesses they support. Today's applications force people to figure out how to map isolated pools of information and functions to their tasks and processes, and they force IT pros to spend too much budget to keep up with evolving markets, policies, regulations, and business models. IT's primary goal should now be to deliver a new generation of enterprise software that adapts to the business and its work and evolves with it. Forrester calls this new generation Dynamic Business Applications, emphasizing close alignment with business processes and work (design for people) and adaptability to business change (build for continuous change). See the September 24, 2007, "The Dynamic Business Applications Imperative" report.

11 Many application architects and developers don't know enough about developing secure applications. The solution is to avoid security vulnerabilities as early as possible by employing principles of secure design such as threat modeling. See the March 10, 2009, "Use Threat Modeling To Develop More-Secure Applications" report.

12 It's one thing to say infrastructure and operations (I&O) professionals need to invest in infrastructure-as-a-service (IaaS) cloud computing; it's quite another to justify the financial and resource commitments. This requires a business case that validates the investment on grounds of business empowerment, cost savings, or faster time-to-market. Positive return on investment (ROI) from cloud computing can't be achieved as a blanket business case because the benefits of cloud computing vary based on the application use case. See the November 2, 2010, "Justifying Your Cloud Investment: Web Sites" report.

13 Oxford Dictionary defines ideate as: To form an idea. Source: Oxford Dictionaries website (http://oxforddictionaries.com/view/entry/m_en_us1256915#m_en_us1256915).

14 Frederick P. Brooks, Jr. is the author of the classic computer science book "The Mythical Man-Month". Source: Frederick P. Brooks, Jr., *The Design Of Design: Essays From A Computer Scientist*, Addison-Wesley, 2010.

15 Source: Keith Richards, *Life*, Little, Brown and Company, 2010.

FORRESTER®

Making Leaders Successful Every Day

Headquarters

Forrester Research, Inc.
400 Technology Square
Cambridge, MA 02139 USA
Tel: +1 617.613.6000
Fax: +1 617.613.5000
Email: forrester@forrester.com
Nasdaq symbol: FORR
www.forrester.com

Research and Sales Offices

Forrester has research centers and sales offices in more than 27 cities internationally, including Amsterdam; Cambridge, Mass.; Dallas; Dubai; Foster City, Calif.; Frankfurt; London; Madrid; Sydney; Tel Aviv; and Toronto.

For a complete list of worldwide locations visit www.forrester.com/about.

For information on hard-copy or electronic reprints, please contact Client Support at +1 866.367.7378, +1 617.613.5730, or clientsupport@forrester.com. We offer quantity discounts and special pricing for academic and nonprofit institutions.

Forrester Research, Inc. (Nasdaq: FORR) is an independent research company that provides pragmatic and forward-thinking advice to global leaders in business and technology. Forrester works with professionals in 19 key roles at major companies providing proprietary research, customer insight, consulting, events, and peer-to-peer executive programs. For more than 27 years, Forrester has been making IT, marketing, and technology industry leaders successful every day. For more information, visit www.forrester.com.

FORRESTER®