By BRIAN WHITWORTH, JERRY FJERMESTAD, *and* EDWARD MAHINDA

THE WEB OF SYSTEM PERFORMANCE

Because information system performance is multidimensional, specialist theories of performance dimensions must be integrated into a model of system design.

Modern information systems face demands so diverse some computer science researchers postulate a virtual evolution, where only the fittest applications survive [4]. What is IS fitness? Successful life, measured by its continuance, varies from simple viruses to powerful predators. All are fit but not in the same way. Indeed, the strongest need not be the fittest, as one-sided "excellence" can precede extinction. IS progress seems equally nonlinear, as laptops deliver less power than PCs for more cost yet sell well. The variability of modern technology leads some to argue the need for a multi-goal model of system performance [2]. Here, we introduce such a model that applies synthesis-of-form concepts to information systems [1].

The model derivation described in [12] assumes system performance affects fitness, along with external factors like marketing in IS. It has no time dimension, describing only performance at a particular point in time while ignoring biological reproduction (used by only a few programs, excluding viruses). Performance is defined as "how well a system interacts with its environment to gain value and avoid loss. The model suggests that advanced systems include four elements: a boundary, a supporting internal structure, output effectors, and input receptors [12]. Likewise, biological cells have a membrane boundary, internal support (nucleus), flagella to move (effectors), and photoreceptors. People have a skin boundary, internal brain and organs, acting muscles, and sensory input. Computers have a physical case, motherboard architecture, printer/screen effectors, and keyboard/mouse "receptors." Software has a memory boundary, an internal program structure, and specialized input/output modules.

Each element plays a natural role in system performance and must interact successfully with its environment. If we accept that successful interactions generally maximize the opportunity of the system for gain and minimize the risk of system hurt, then com-

ILLUSTRATION BY SERGE BLOCH

bining four general elements with two general environment interaction types gives eight general performance goals of the web of system performance (WOSP). They are outlined (in parentheses) in the following list:

Boundary manages system entry to enable useful entry (extendibility) and deny harmful entry (security);
Internal structure controls and sustains the system to accommodate external change (flexibility) and internal change (reliability);
Effector manages changes on the direct environment to maximize external effects (functionality) and minimize internal effort (usability); and
Receptor manages sensing of the environment to enable meaning exchange (connectivity) and limit meaning exchange (privacy).

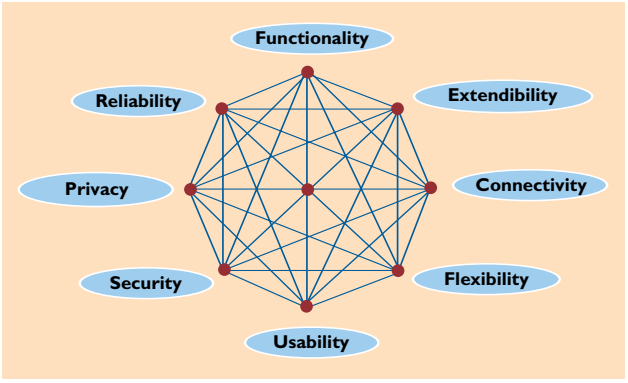

Figure 1. The Web of system performance dimensions.

Subgoal	Similar Terms
Extendibility	Openness, interoperability, permeability, compatibility, scalability
Security	Defense, protection, safety, threat resistance
Flexibility	Adaptability, portability, customizability, plasticity, agility, modifiability
Reliability	Stability, dependability, robustness, ruggedness, durability, availability, maintainability
Functionality	Capability, effectualness, usefulness, effectiveness, power, utility
Usability	Ease of use, simplicity, user friendliness, efficiency, accessibility
Connectivity	Networkability, communicativeness, interactivity, sociability
Privacy	Confidentiality, secrecy, camouflage, stealth, social rights, ownership

Table 1. System performance dimension terminology.

The eight WOSP goals can be illustrated as a web (see Figure 1) where a point's distance from the center is the degree of that performance dimension. This web of performance has an area, a shape, and goal tensions. The web area represents the system's overall performance, so a larger area has more fitness potential. The web shape is the system's performance profile, which varies with the environment; for example, a threat environment might require more security. The web lines are the goal interactions, or tensions. One can imagine them as rubber bands of different tensions connecting the performance dimensions, so increasing one may suddenly pull back another. As Table 1 suggests, none of the WOSP goals are new to computer science researchers; what is new is their conceptual integration into a common framework.

A system's boundary determines what is allowed to enter and exit it and can be designed to repel external threats (security) and accept external opportunities (extendibility).

Extendibility is a system's ability to make use of outside elements (such as the way a car with a tow hitch might add a trailer and software can have extensions and plug-ins). Human tool use extends performance the same way, and programs can use third-party plug-ins, given the equivalent of an open human hand. However, for a car to extend itself via a trailer, its tow hitch must match the trailer's link, so extendibility requires a known link form. IS open standards create this benefit and represent the value of open source code. In the 1980s, software businesses focused on copy protection and were surprised when Netscape made its source code public yet thrived. It was successful (at the time) because openness encouraged third-party development, which increased performance. A similar argument could explain why the early IBM PC 25 years ago outperformed a more reliable, secure, usable Apple Macintosh. One was a propriety black box; the other had standard slots for third-party cards. Extendibility is a critical factor in IS performance.

Security is a system's ability to protect itself against unauthorized entry, misuse, or takeover (the way a car has locks and keys and applications have logons and passwords). Secure hardware is sealed and tamperproof, and distributors prefer compiled to interpreted software because users cannot alter it. The entry-denial principle is the same for hardware and software. Virus and hacker threats make boundary firewalls and logon checks critical to system survival. A security breach is a system failure and thus a performance failure. Security is a key part of IS performance.

A system's internal structure can be designed to manage internal changes (reliability) or external changes (flexibility). Flexibility is a system's ability to work in new environments (in the same way tracked vehicles that operate in difficult terrain are flexible); similarly, mobile devices can receive signals in difficult

network areas. CSMA/CD (Ethernet) protocols outperformed more-reliable but less-flexible polling protocols. Flexible relational databases displaced more efficient but less-flexible hierarchical and network models. Most modern software has a preferences module (such as the Windows control panel) to configure it for hardware, software, or user environments. Flexibility is another critical aspect of IS performance.

Reliability means a system keeps operating despite internal changes (such as part failure); a car, like software, that always goes is a great thing. Reliable systems are almost always available, survive stress or load, and if affected degrade gracefully rather than crash catastrophically. In IS, mean time between failure measures the probability of failure-free operation over time. Equally important is fast recovery, whether by error code or state rollback. Computer companies like Dell Computer that provide better after-sales support and warrantees, succeed because reliability is critical to IS performance.

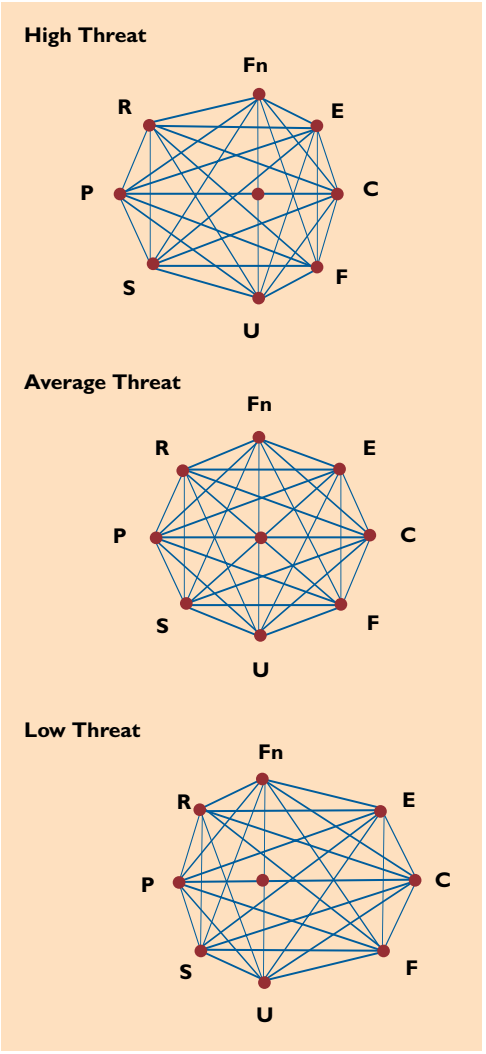

Figure 2. Performance shapes by threat environment.

System effectors change the external environment and can be designed for maximum effect (functionality) or to minimize the cost of producing that effect (usability). Functionality (capability) is a system's ability to act directly on its environment to produce a desired change (such as the way a car's speed changes its position and a word processor's power changes documents). Focusing on functional requirements produces feature-laden software that gets the job done, so the effort to use it is never wasted. People upgrade for new capabilities, so functionality is important in IS performance.

Usability is a system's ability to minimize the relative resource costs of action (in the same way a usable car is easy to drive and easy on gasoline). Reduced instruction set computing outperforms complex instruction set computing by using less code for the same work. "Light" software runs well in background because it uses little CPU/memory. In human-computer interaction, graphical user interfaces replaced command user interfaces in the 1980s because they reduced users' cognitive effort. In today's online world, where unhappy users just click on to another Web site, usability is a critical part of IS performance.

Social interaction, which adds a social dimension to system performance, can both enable information exchange (connectivity) and limit it (privacy). Connectivity is a system's ability to communicate with other systems; for example, connected cars can detect other cars to avoid collisions, and connected software can download updates or help people communicate. We earlier linked actions to effectors, even though actions occur in a sensory-guided feedback loop. Likewise, we link meaning to receptors because receptor processing creates meaning, even though communicative acts also require effectors. The communication end result of meaning comes from receptors (and the ensuing processing) in the same way effectors create the end result of actions. For modern software, connectivity is critical to IS performance.

Privacy is a system's ability to control the release of information about itself (such as the way a car's tinted windows hide the occupants or software lets one browse the Web anonymously). Confidentiality is the name engineers give privacy from a software perspective, rather than from a user perspective. The military values stealth airplanes for the same reason animals camouflage themselves. In society, not giving information is important because public ridicule or censure can have physical consequences. In social-technical environments, privacy is a critical part of IS performance.

The eight WOSP goals are conceptually modular; the definitions do not overlap. In theory, any performance level on any dimension can combine with any other dimension (such as the way a sealed and bullet-proof plexiglass house can provide 100% security but no privacy). In the practice of design, the dimensions interact, as all system goals do, because the system

must meet all demands. However, conceptual modularity means there is not necessarily a relationship between any two such demands that might arise from their definitions.

Practical Implications

The WOSP model can be used in system design or and in system evaluation. While performance can be considered absolute, the WOSP model views performance relative to the environment, so performance has no "perfect" form. Of the eight WOSP goals, four are generally success-creating—functionality, flexibility, extendibility, connectivity—and four are failure-avoiding—security, reliability, privacy, usability. This is useful, as environments can vary:

Opportunistic. Actions can yield benefits to systems that are able to reap them;
Hazardous. Actions can harm systems that cannot handle hazards; and
Dynamic. The effects—loss and gain—of risk and opportunity of action can change quickly, favoring systems that do the same.

Goal	Detail	Weight %
Extendibility	Use outside component/data add-ins?	
Security	Resist outside attack/take-over?	
Flexibility	Predict/adapt to external changes?	
Reliability	Avoid/recover from internal failure?	
Functionality	What task functionality is required?	
Usability	Conserve system/user effort or training?	
Connectivity	Communicate/connect with other systems?	
Privacy	Manage self-disclosure and privacy?	
Performance	**Interact successfully with the environment.**	**100%**

Table 2. System performance dimension weights.

If performance has a shape, as well as an area, different shapes may be more suitable in different environments (see Figure 2). The WOSP model helps developers achieve the performance shape that fits their environments by allocating weights to the performance dimensions (see Table 2).

In traditional IS development, functionality is the primary goal, and nonfunctional requirements (NFRs) are second-rate needs. However, many software systems end up with many more lines of error or interface code than functional code, failing often for unexpected nonfunctional or quality reasons [3]. If NFRs can cause system failure, they define performance, as well as modify it. In the WOSP model, functionality differs from other performance goals only by being more obvious. Poor usability can nullify functionality, just as poor functionality can make usability irrelevant.

Modern communications technology illustrates the many dimensions of performance. Cell phones let people talk anywhere, anytime (flexibility), but must still be functional (voice quality), usable (keys not too small), reliable (if dropped), secure (if stolen), extendable (earphones, phone covers), connected (reception), and private (to prevent snooping). Each criterion may have a different weight, but any of them might be critical. Ubiquitous wireless software must flexibly use different devices/networks but also be resilient to data-entry errors and power failures. It must be scalable yet secure against virus attack and connect yet maintain user privacy. An information system is not high performance if it is:

Ineffectual. Cannot do the job;
Unusable. Users cannot make it work;
Unreliable. Breaks down often;
Insecure. Succumbs to viruses;
Inflexible. Fails when the technical or business environment changes;
Incompatible. Cannot use standard plug-ins or data;
Disconnected. Cannot communicate; and
Indiscreet. Reveals personal or corporate information.

The WOSP model is a useful checklist for new technology designers; while success needs many causes, failure may need only one.

The WOSP model suggests that a genuine performance advance in one dimension may not succeed if the advance also significantly reduces other performance dimensions. In Figure 1, only a total area increase is progress, and increasing one dimension at the expense of another may not produce such an increase. For example, in 1992, Apple Computer's then-CEO John Sculley introduced the handheld Newton, saying portability (flexibility) was the wave of the future. We now know he was right, but the Newton's small size made data entry difficult, and its handwriting recognition was poor. The flexibility advance was neutralized by a usability reduction, and in 1998 Apple dropped the line due to poor market performance. Later, when Palm's Graffiti language solved the usability problem, the PDA market revived, though today, PDAs are under threat from cell phones with better connectivity. The conclusion is that breakthroughs may require advances on more than one WOSP dimension.

For new systems, the WOSP begins "slack" and involves few tensions, so increasing a performance

goal usually also increases overall performance. However, as systems evolve and the WOSP area increases, so do goal tensions. As specialty goals pull the system in different design directions, one purpose can cut across another [9]; for example, upgrading connectivity from one-to-one 1G circuit switching to 2.5G always-on packet switching created security problems for networked PCs. Likewise, feature creep can result in complex bloatware that is difficult to use, maintain, and defend, and unidimensional "progress" can bite back [10]. The interaction of performance goals explains a strange paradox: that later versions of successful products, after much effort and many additional features, might perform less well than the original.

Developers can expand the web by "pulling" two or more sides at once; for example logon subsystems (security) might welcome users by name and recall their preferences (increased usability). Flexibility need not deny reliability, nor functionality reduce usability, nor Internet connectivity abuse privacy [11]. In the WOSP model, apparent opposites (such as security and openness) can be reconciled through innovation.

Goal	Analysis	System Layer	Testing
Extendibility	Interoperability analysis	Import/Add-in	Compatibility
Security	Threat analysis	Security/Log-on	Penetration
Flexibility	Contingency analysis	Configuration/Control	Situational
Reliability	Error analysis	Fault Recovery	Stress/Load
Functionality	Task analysis	Application	Output
Usability	Usability analysis	Interface	User
Connectivity	Network analysis	Communication	Channel/Network
Privacy	Legitimacy analysis	Authorization Rights	Social

Table 3. System performance design layers.

Reconciling goal conflicts occurs at the intersection of specialties. Traditional projects define system requirements by specialty (such as interface and database design) because they require different skills. Designing for add-ins requires standards knowledge, while security design requires virus knowledge. The WOSP model welcomes such goal specialization. Depending on the project, as many as eight project specialty teams can design different system layers, with separate specifications, code, and testing (see Table 3). However, it also recognizes that specialization alone, however good, can produce what has been called the Frankenstein effect [10].[1] This suggests a design requirement over and above those specified, namely their innovative integration.

While efficiency increases as specialists specialize, integration may decrease, and with it system performance. Agile and extreme project methods address this problem by encouraging everyone to be involved in all aspects of a system's design. Their popularity and success suggests integration is as important in system design as specialization. Integration can be combined with specialization in two main ways:

Cross-disciplinary integrators. They mediate among specialties, chair common meetings, reconcile crossover conflicts, and create synergies; and

Cross-specialist groups. There might be four groups, one for actions (functionality and usability), one for interactions (security and extendibility), one for contingencies (reliability and flexibility), and one for sociability (connectivity and privacy).

Designers can expect advanced new projects to need more integration than simple older ones, so developing complex social-technical systems may be as much about innovative integration as traditional specialization.

Theoretical Implications

The WOSP model addresses a system's performance properties, not outside influences (such as marketing, politics, and distribution), on system performance. System cost is also outside the scope of WOSP, as it assumes buyers expect to pay more for more performance. Finally, the WOSP logic assumes the system "world" is defined at one of four levels: mechanical, informational, cognitive, and social [12]. Each depends on the previous one, as software depends on hardware but also "emerges" with both greater demands and greater potential. For example, the social level requires legitimate interaction in order to vastly increase system productivity, as illustrated by cooperative societies [11]. To apply the WOSP model, the world level must first be defined, as the model applies to any IS level but not to all at once. For example, a system may be hardware-reliable but software-unreliable; or both hardware-and software-reliable yet operator-unreliable; or work reliably with individuals but break down when scaled to a social level.

System design theories (such as the waterfall method) suggest ways to achieve known goals. However, the WOSP model addresses the goals and discovers a web of system performance. When research specialties (such as security, usability, and flexibility) develop performance models, they tend to subsume

[1] Dr. Frankenstein stitched together the "best" of each body part he found in a graveyard. The result was a (low-performance) monster.

other goals within their specialty. For example, the European general security model includes availability, integrity, reliability, and confidentiality under a general security concept.

However, mechanisms that increase fault-tolerance (reliability) can reduce security but are illogical if reliability is part of security. Reliability and security differ in the same way the engineers who maintain society differ from the police who protect it, one aiming to provide services, the other to deny them [6]. Likewise confidentiality (privacy) is not part of security, as one can be private and not secure or secure but not private (such as a prisoner in a jail cell under video surveillance). So, it is illogical for each specialty to include the other specialities within itself.

The Technology Acceptance Model (TAM) argued in 1989 that usability could be as critical as usefulness in IS system acceptance. It questioned the performance value of a powerful system that was too difficult to use. Yet today, security and privacy criteria may be more important to Web users than the now-traditional functionality and usability criteria [8]. One could expand the original TAM theory to make security part of "usefulness," as an insecure system is not useful. However, the same argument could make usability part of usefulness, so the argument collapses. Expanding TAM's usability concept fairs little better. When usability measures include suitability for task (functionality) and error tolerance (reliability) [5], then usability, like security, becomes a confusing catch-all term for performance. When the proponents of flexibility suggest that scalability and connectivity are aspects of flexibility [7], specialist concepts expand to fill the available theory space and create confusion. The WOSP approach is not to conceptually expand but to conceptually contract, or "modularize," such concepts as usability, security, and flexibility, placing them all under the general rubric of system performance.

Theorists easily forget how they previously viewed today's progress. The Internet was for techno-geeks until virtual reality became real. Email was socially inept lean communication until text became rich. Tim Berners-Lee's Web idea was ignored by his employer the European Organization for Nuclear Research, by the academic hypertext community, and by Microsoft, before MIT took it up to help create today's online society. Cell phones were yuppy toys, until everyone got one.

These cases, and many more, illustrate that performance, and hence progress, is multidimensional. The Internet provides massive connectivity; text email is easy to use; the Web is scalable; the cell phone is flexible. Each adds a different web of system performance factors, so progress may seem unpredictable, but such variety is the nature of multidimensional progress. While some view progress as a train moving forward on a single track, the WOSP model views it as a train on many tracks, switching among them to increase the covered area as progress occurs.

Implicit in this model is that today's trends will not necessarily be tomorrow's innovations. A decade ago multimedia was hot, but *Star Trek*'s vid-phone, though technically feasible, is not commercially viable. Moreover, videoconferencing did not boom, nor did people take up virtual reality goggles in computer gaming. Instead, games became connected through virtual social worlds (such as The Sims and various massively multi-player online role-playing games). Meanwhile, game editors made games more extendible, as users could add maps and scenarios, as in id Software's Doom WAD files. Progress in one IS performance dimension it seems tends to be followed by progress in another.

Experts are, by nature, experts in the past, so the progress they predict is not always the progress that occurs. The WOSP model suggests, somewhat counterintuitively, that developing a system's weaker aspect(s) may yield greater performance increases than developing its stronger aspect(s), even though the latter usually creates its success. If performance is the WOSP area, the greatest area increase is achieved by extending the shortest dimension. For example, perhaps the future of online gaming will involve exclusive gaming groups (privacy). We (the authors) are developing a WOSP instrument to help designers determine a system's performance profile.

Conclusion

If something works, developers and users alike want to do it again and again. But if software is evolving the way life is evolving, IS progress will take many forms. It is interesting that killer applications (such as email and chat) are functionally simple, at least initially. Perhaps less capability creates the WOSP slack needed for all-round performance expansion. Users not only want functionality, they also want usability, reliability, flexibility, security, extendibility, privacy, and connectivity, as all are aspects of performance.

As information systems become more complex, performance-integration issues will become more critical. An information system is a synthesis of form in a multidimensional performance space where each design choice affects each dimension [1]. Researchers must recognize what designers face: that the system whole is more than the sum of its parts. Hence putting only advanced specialists into cross-discipli-

nary teams may be as useful as putting people who speak different languages together in the same room. Cross-disciplinary research without cross-disciplinary people to translate among specialties is not really cross-disciplinary. IS academic researchers must reward generalists who struggle to cross-train, as well as specialists who struggle to focus; specialized conferences, journals, and departments often do not do this. Practice and theory are both needed to advance integration and specialization and to achieve balance and excellence. C

References

1. Alexander, C. *Notes on the Synthesis of Form.* Harvard University Press, Cambridge, MA, 1964.
2. Chung, L., Nixon, B., Yu, E., and Mylopoulos, J. *Non-functional requirements in Software Engineering.* Kluwer Academic, Boston, 1999.
3. Cysneiros, L. and Leita, J. Non-functional requirements: From elicitation to modelling languages. In ICSE (Orlando, FL, May 19–25). ACM Press, New York, 2002.
4. David, J., McCarthy, W., and Sommer, B. Agility: The key to survival of the fittest. *Commun. ACM 46,* 5 (May 2003), 65–69.
5. Gediga, G., Hamborg, K., and Duntsch, I. The IsoMetrics usability inventory: An operationalization of ISO9241-10 supporting summative and formative evaluation of software systems. *Behaviour & Information Technology 18,* 3 (1999), 151–164.
6. Jonsson, E. An integrated framework for security and dependability. In NSPW (Charlottesville, VA, Sept. 22–25). ACM Press, New York, 1998, 22–29.
7. Knoll, K. and Jarvenpaa, S. Information technology alignment or 'fit' in highly turbulent environments: The concept of flexibility. In SIGCPR (Alexandria, VA, 1994). ACM Press, New York, 1994.
8. Mahinda, E. and Whitworth, B. The web of system performance: Extending the TAM model. In Americas Conference on Information Systems (ACIS) (Omaha, NE, Aug. 11–14, 2005).
9. Moreira, A., Araujo, J., and Brita, I. Crosscutting quality attributes for requirements engineering. In Software Engineering and Knowledge Engineering (SEKE) (Ischia, Italy, 2002). ACM Press, New York, 2002.
10. Tenner, E. *Why Things Bite Back.* Vintage Books, Random House, New York, 1997.
11. Whitworth, B. and deMoor, A. Legitimate by design: Towards trusted virtual community environments. *Behaviour & Information Technology 22,* 1 (2003), 31–51.
12. Whitworth, B. and Zaic, M. The WOSP model: Balanced information system design and evaluation. *Communications of the Association for Information Systems 12* (2003), 258–282.

Brian Whitworth (bwhitworth@acm.org) is a senior lecturer in the Institute of Information and Mathematical Sciences at Massey University, Auckland, New Zealand.

Jerry Fjermestad (fjermestad@adm.njit.edu) is an associate professor in the School of Management of the New Jersey Institute of Technology, Newark, NJ.

Edward Mahinda (egm3@njit.edu) is a Ph.D. student in information science in the College of Computing of the New Jersey Institute of Technology, Newark, NJ.

While some view progress as a train moving forward on a single track, the WOSP model views it as a train on many tracks, switching among them to increase the covered area as progress occurs.