EVALUATING FLEXIBILITY AND RELIABILITY IN EMERGENCY RESPONSE INFORMATION SYSTEMS

Edward Mahinda, Brian Whitworth
New Jersey Institute of Technology,University Heights, Newark New Jersey, USA
Email: egm3@njit.edu, bwithworth@acm.org.

Keywords: Emergency response information systems, flexibility, reliability, evaluation, questionnaire instrument

Abstract: Flexibility, variously known as adaptability, tailorability, and customizability, has long been recognized as important in information system (IS) success. Reliability has known value in IS for the resulting predictability it bestows on a system. However increasing flexibility can increase possible paths for system breakdown, and so contribute to failure, i.e. increasing flexibility can reduce reliability. Reliability and flexibility seem in design "tension", as one creates change and the other resists it. The combination of reliability and flexibility has been called "robustness", and it seems a desirable integration particularly for emergency response systems. However typically these two areas are studied separately. Our approach to evaluating the combination of reliability and flexibility is to define two distinct requirements that neither overlap nor contradict, and can be assessed by system users. A questionnaire instrument for users is proposed for measurement of the flexibility and reliability of a system.

1 INTRODUCTION

Most analyses of flexibility and reliability focus on one or the other, yet they are related as both involve responding to change (El Sawy and Nanus, 1989). That software does not fail when presented with a variety of input data is normally seen as reliability, but can be seen as flexibility (Knoll and Jarvenpaa, 1994). Conversely software that operates on a variety of platforms is normally considered flexible but this can be seen as reliability (Wittaker and Voas, 2000). Such conceptual confusion arises when either aspect is considered in isolation and over-generalized. These two aspects of modern information systems must be considered together to be properly discriminated and integrated. For example, (El Sawy et al, 1989) combines both into the concept of robustness. Flexibility is seen as external robustness (relative to external environment changes), and reliability is seen as internal robustness (relative to internal system changes). Robustness itself seems part of the higher concept of system performance, proposed to also include functionality, usability, security, extendibility, connectivity and confidentiality (Whitworth and Zaic, 2003). This paper analyses reliability and flexibility as distinct but connected system concepts, in order to develop conceptually valid statements for system evaluation.

In doing this, four levels of information system will be recognized (El Sawy et al, 1989; Whitworth et al, 2003), each with a distinct environment, as shown in Table 1 below. This paper uses examples from all levels. Deciding on the level is critical as it determines whether the flexibility or reliability is achieved by hardware, software, user interface or the social level, or perhaps a combination.

System Level	Environment
Mechanical	Hardware peripherals and networks
Information	Software data and programs
Cognitive	Users and agents
Social	Groups and societies

Table 1. Information Levels and Environments

The circumstances of September 11, 2001 brought into focus the issue of how to deal with events unexpected both in nature and timing by the organizations who deal with them (Arens et al, 2003). Unanticipated events, of low probability but high impact, must be expected in the future, but given limited resources, it is not feasible or even desirable to prepare for every unlikely possibility. Hence the need for a general-purpose IS infrastructure that can adapt and respond to any threat, i.e. flexible emergency response information systems. Flexibility (or synonyms like adaptability, tailorability and customizability) has long been recognized as important in information systems (El Sawy et al, 1989). Furthermore, a recent analysis of IS performance includes flexibility as a critical design requirement of any advanced information system (Whitworth & Zaic, 2003). This paper analyzes how flexibility can be increased in the sort of advanced information systems used in emergency responding. It concludes that flexibility and reliability are intrinsically related, and suggests an assessment tool that designers could use to improve emergency system response.

2 FLEXIBILITY

The need for flexibility in an emergency responses mirrors a general need for performance flexibility in business information systems. A survey by Information Week found that the creation of a strong and flexible IT infrastructure was the top priority among 150 IT managers (InformationWeek, 1999). This seems in part due to global competition and compressed cycle pressure for new products, i.e. a rapidly changing and turbulent environment (Lee et al, 1992]). In addition, large social applications, such as the World Wide Web, email, DNS and multi-user file systems have flexibility demands placed upon them (Montz et al, 1998). It has been argued that tailorability is the modification and adaptation of systems after they have been put into use by (Kjaer and Madsen, 1995):

- constructing new behaviors from existing ones, e.g. by writing macros
- choosing between alternative anticipated options, e.g. by switches
- effecting a fundamental change in the system, e.g. by modifying code

These can be seen as three basic ways to achieve flexibility in the three common aspects of any information system, namely its input, output and processing. For example in the context of supply chain management, product output flexibility has been defined as " the ability to handle difficult, non-standard orders, to meet special customer specifications, and to produce products characterized by numerous features, options, sizes and colors." (Vickery, Calantone and Droge, 1999). Organizational information systems must also be capable of assessing market demands and environmental changes in general, requiring:

"… an associated scanning component that alerts it to environmental changes and enables switching to another configuration quickly and inexpensively to take advantage of the environmental change." (El Sawy et al, 1989, p. 36)

One benefit of the World Wide Web is its ability allow people to scan and assess trends in the social/business environment. Just as supply chains strive to flexibly respond to changing target markets, so flexible software should be able to respond to different user demands. In this sense, flexibility is the ability to both predict and sense environment change and to respond to it appropriately. Flexible software can handle a wide variety of input demands and give output that is equally varied in nature. Finally if the software code itself can be easily modified to fit changes, any changes can be accommodated. In general flexibility is a system's ability to not be rendered ineffective by changing circumstances and to take advantage of environment opportunities (Whitworth et al, 2003). This review suggests that IS flexibility has three distinct aspects:

Flexibility by detection: the system can detect, recognizes or predict a wide variety of environment input changes.

Flexibility by response: the system has a wide variety of output responses to unpredicted situations that can be quickly activated (e.g. by reconfiguring or recombining responses from its response repertoire).

Flexibility by adaptation: the system can modify or reconfigure itself using environment feedback.

The above aspects are distinct, but not alternatives. Most cases of flexibility involve some combination of the above three types. In biological terms, flexibility by detection is awareness of situational change, flexibility by response is agility, and flexibility by adaptation is the ability to learn new things. A system advanced in all three would be maximally flexible (in input analysis and prediction, in response variety and agility, and in adapting itself to new environments). It can be argued that rather than building flexibility into the system, one could adapt and maintain the software as and when needed. However when changes are numerous and frequent, this approach can be expensive, unless the life of the system is expected to be short. Insufficient flexibility can restrict the scope and application of any system.

For an information system, if it can predict, by system logs or records, a future environment change, then users or programmers can adapt it to fit those changes. If a system cannot predict, but is easily modified, then it can be modified to fit, even though the events were unexpected. Finally, even if a system is neither aware nor agile, if it can learn (or change its own processing), then both these can be achieved given time. An example can illustrate the three aspects. For a system to run on a variety of hardware or operating system environments is generally considered flexibility. To do this it needs a switch to change its mode of operation in the new environment (flexibility by response). If the software also has some way to recognize a new environment (flexibility by detection), then the change can be automatic rather than user generated. Microsoft's Plug 'n Play illustrates this type of flexibility, which requires both detection and reaction. Finally if a system could truly learn, it could even adapt to a new operating system by trying various options (flexibility by adaptation).

3 RELIABILITY

The importance of software reliability seems linked to the increasing dependence by society on technology (Littlewood and Stigini, 2000). Reliability is a major concern of traditional corporate information management systems where the consequences of downtime are felt directly as disrupted operations, lost profits and dissatisfied customers who may shift allegiance elsewhere. It is also important in software growth areas like embedded systems which can control critical operations, whose failure can have dire consequences (Rinard, 2003).

Reliable systems operate even under conditions of high stress. If performance is affected, it degrades rather than collapses. If they fail, it is possible to repair them quickly. Reliable systems can be trusted to perform, and since predictability is essential for planning, this makes reliability important in computer performance. Reliability can be defined as a system attribute that gives confidence it will perform as required. Conventionally software reliability is the probability a software system will operate without failure for a specified time under specified operating conditions (Whittaker and Voas, 2000). Reliability has also been defined as internal robustness, a software property of having low sensitivity to changes in the values of internal parameters and functions (El Sawy et al, 1989).

A common theme underlying the above reliability definitions is the idea of continued performance despite some internal failure, of either a necessary system part or necessary interconnection. Hence, in general reliability can be defined as a system's ability to continue performance despite internal variations (such as part failure) (Whitworth et al, 2003). The ability to maintain key internal structural relationships when environmental disturbances occur, gives a system a resilience to absorb environmental shocks and bounce back (El Sawy et al, 1989). While flexibility responds to environment change by creating new internal equilibrium states by structural changes, reliability is about maintaining the existing equilibrium.

One way to achieve reliability is to keep the program sufficiently simple that the standard reliability method of testing all possible performance paths can cope. However, as systems increase in performance, internal complexity increases, and the full specification of all internal interactions becomes an infeasible solution to software reliability (Rinard, 2003). In general, increasing any aspect of system performance may unavoidably reduce reliability.

Another way to increase reliability is the classic design advice to design systems with loosely coupled parts, each with high cohesion. Reducing the interdependence of a system's constituent parts can increase its reliability, because then when one part fails the error does not "cascade" to other parts. This general approach can be called reliability by modular design.

The probability of failure of a component in a system can be halved by connecting it in parallel to an alternative one (Prayurachatuporn. and Benedicenti, 2001). The two components are redundant, but if one fails, the other can take over. When such systems are damaged they gradually and gracefully degrade in performance, rather than crashing instantaneously. This approach can be called reliability by redundancy.

Finally while one can design and duplicate to prevent failure, an alternative is to allow failure to occur, but ensure quick recovery. Hence a recommended means to reliability is the deployment of error detection techniques (Littlewood et al, 2000). Monitoring can include not only detecting and responding to errors, but also logging them for future processing. This approach can be called reliability by recovery.

The above analysis suggests three distinct aspects to reliability:

Reliability by modularity: System is designed to minimize coupling so if one part fails others still continue.

Reliability by redundancy: System is designed to have independent means to the same end, so if one fails another can take over.

Reliability by recovery: System is designed to deal with any failure by some recovery technique.

Again these aspects are distinct, but are not alternatives, and most examples of reliability will involve more than one aspect. A system advanced in all three would be maximally reliable (in being modular so no failure cumulated, in being redundant so no failure was critical, and in being recoverable so no failure was permanent.)

4 APPLICATION TO EMERGENCY RESPONSE INFORMATION SYSTEMS

The IS emergency response system requirement is for both flexibility and reliability, where flexibility is a system's ability to perform differently given external changes, and reliability is its ability to perform the same despite internal changes. The two distinct system design "layers" of flexibility and reliability need particular attention in emergency response IS.

The first requirement is for reliability, often equated with error response code. In describing an ideal emergency response system, Turoff suggests that a system that only operates when exceptions occur is not acceptable [Turoff, 2002]. He argues that the system that responds in an emergency must also be one that is ordinarily used for everyday operations. One reason is that the people in the response network have operational familiarity with it. The emergency would then be an exceptional situation in a familiar context. Using any system in day to day circumstances is one way to ensure a degree of reliability, because breakdowns will be discovered and corrected. Hence this proposal is a general means to increase reliability in emergency response systems.

This paper suggests more detailed aspects of reliability that can be consciously built in. For example the error response subsystems of any information system, whose job is to recognize errors and recover from them, form a distinct code layer that can be enhanced. The error can occur in input (e.g. data input error), in output (e.g. one active process attempts to access a record in use by another), or in processing (e.g. an infinite loop). Reliability can also be increased by modular design, and system components can be deliberately failed to test that this does not cause the whole system to fail entirely. Finally redundancy could be built into critical system functions that could fail. For software, this usually means redundant data input channels, so if for example a scanner doesn't work, data can be input manually. However it could also involve data redundancy, usually considered inefficient, but from a reliability perspective storing the same data in more than one place can increase reliability. In communication, a mesh network topology can be seen as redundant communication links, which is thus more reliable. .

The second requirement is for flexibility, often corresponding to code that provides system options or preferences. Emergency situations are by definition outside of the norm. Hence normal input and output processes may not work or fit the demand. A reliable system may continue to operate but be ineffective because the demand has changed and it cannot be adapted. Inflexible information systems can easily be rendered useless by an emergency. Again this paper suggests detailed aspects of flexibility that can be designed into systems. To respond to change one must first be aware of it, and so system components that scan the environment and report changes increase flexibility. This could range from knowledge of power outages in other areas to knowledge of who is online at the present. Secondly a flexible system must be able to change its output to fit the need, which again could range from printing at remote computers to the ability to sort and filter data reports in any way required. Finally, and especially in an emergency, a flexible system needs to be changeable at the core level. One option is a self-modifying system, with artificial intelligence (AI). Another is to give human access to basic code functions, the equivalent of manual override of automated systems. For example a plane's control system normally prevents certain maneuvers as unacceptably dangerous, but in some cases they may be appropriate, so pilots need an override method. The objective should be to have flexibility by detection, response, and adaptation built into the system. These three aspects of flexibility have the effect of reducing the dependence of the system on a particular external environment. Thus for example, the system should be able to work with data/information from various different internet sources, from a CD, or from direct keying in.

If flexibility and reliability impose contradictory design requirements on a system's internal structure, these layers must be designed to work together. While customizability can improve technology, it can also reduce reliability by increasing possible paths for system breakdown. Hence flexibility has been extended into the concept of "robustness", as the ability to adapt to change both inside and outside the information system (El Sawy et al, 1989). However the WOSP model suggests that flexibility and reliability are only two of many system requirements in tension (Whitworth and Zaic, 2003, p267). Thus while flexibility of information systems is usually desirable, it can have unintended and unwanted consequences, e.g. the ability to override safety mechanisms can be misused. Flexibility can also reduce usability, which can affect performance. For example, the flexible automation of cockpits requires results in a large number of functions and options for flights under different circumstances (Woods, 1992). To utilize this flexibility, the pilot must learn all these options, and when and how to use them. These new user burdens and overheads can cause failure, unless usability is also increased along with the flexibility advance. Similarly, too much flexibility in a system can result in the loss of a strategic direction, i.e. threaten organizational functionality (El Sawy et al, 1989).

5 EVALUATION

Software evaluation is one way to assess emergency preparedness. Using the above review and analysis to provide content validity, a tool is proposed to evaluate an information system's flexibility and reliability. A questionnaire of 30 items has been developed that could be used to upgrade systems for emergency response use. Experienced system users could use the statements below to assess the flexibility and reliability of their systems by rating each statement on a 5 to 7 point scale. The statements are categorized as per the flexibility or reliability aspect they pertain to, and the system level they are applicable to. The source is given in brackets.

5.1 Flexibility

5.1.1 General flexibility

1. The software can run on a wide variety of computers and operating system platforms (mechanical level; Montz et al, 1998; Knoll et al, 1994 and Rivard, Poirier,Raymond, Bergeron, 1997).
2. The software has a preferences "control panel" that allows me to change a wide variety of system settings. (information level, from pilot study)
3. The software can be easily changed to fit the task for which it is being used. (cognitive level; Knoll et al, 1994)
4. The application could be used in many organizational settings, other than the one it is currently used in, without major modification. (social level; Rivard et al, 1994)

5.1.2 Flexibility by detection

5. The software automatically recognizes if it is online or offline, in a "plug 'n play" manner, and changes its functions accordingly. (mechanical level, Pilot Study)
6. The software regularly scans its environment and advises of relevant changes (e.g. new messages, new users online, new updates). (information level; el Sawy et al, 1989)
7. The software keeps a log of user activity that allows me to predict usage trends, e.g. to select off-peak access times. (cognitive level, Pilot Study)
8. The software enables me to predict global business trends (social level; el Sawy et al, 1989)

5.1.3 Flexibility by response

9. The software output, or what it does, can be changed easily and quickly (Knoll et al, 1994).
10. The software is ready and able to participate in a new use which was previously not recognized. (information level Knoll et al, 1994)
11. The software can perform vaguely defined user tasks which change over time. (cognitive level, Montz et al, 1998)
12. It is easy to adapt the software to fit different people's needs (cognitive level, from pilot study)

5.1.4 Flexibility by adaptation

13. The software allows my preference configuration to be saved and loaded as needed. (information level, Pilot Study)
14. The software can change how it works according to what I do, i.e. it learns from me. (Knoll et al, 1998).
15. The software "remembers" my previous choices so I don't have to keep typing them in. (cognitive level, Pilot Study)
16. The software can easily record a sequence of commands as a macro, and repeat them when I want (Pilot Study)

5.2 Reliability

5.2.1 General reliability

1. The software performs as required, without failure, over a wide variety of operating conditions, such as users, input data and operating systems. (Wittaker et al, 2000)
2. The software continues to work even when under heavy load. (Whitworth et al, 2003)
3. The software's performance is predictable over time (Whitworth et al, 2003)

4. The software can operate for a long time without breaking down (Pilot experiment)

5.2.2 Reliability by modular design

5. Even if one part of the system fails, the other parts keep going (Pilot experiment)
6. If an error occurs, the rest of the system still works fine (Pilot experiment)
7. When one software function fails, the entire system does not crash (Whitworth et al, 2003)
8. When doing a multi-step task, if one step fails I don't need to redo all the steps from the beginning. I can just redo the mistake, e.g. I re-enter the erroneous data and rerun it.

5.2.3 Reliability by redundancy.

9. The software gives several ways to direct it to do the same task (Prayurachatuporn et al, 2001)
10. If a task cannot be done one way, this software usually offers an alternative means (Prayurachatuporn et al, 2001)
11. I am never stuck, because this software has many ways to do the same thing (Pilot experiment)
12. Every software command can be given by mouse or by keyboard.

5.2.4 Reliability by recovery

13. The software has no problem recovering from errors. (Knoll et al et al 1989)
14. The software gives useful error messages that help users recover (Littlewood, 1978)
15. The software can run an error check, and repair internal data or file problems (Pilot experiment)
16. Under heavy load, software performance degrades gradually and gracefully, rather than crashing catastrophically. (Whitworth et al, 2003)

REFERENCES

Arens, Y., and Rosenbloom, P. S., 2003. Responding to the Unexpected. *Communications of the ACM*. 46(9), 33-35.

El Sawy, O., A., and Nanus, B., 1989. Toward the Design of Robust Information Systems. *Journal of Management Information System.,* 5,4, 33-54.

InformationWeek. Bond the new and the old: Enterprise architecture, January 11, 1999, pp. 108-109.

Kjaer, A., and Madsen, K., H., 1995 Participatory Analysis of Flexibility. *Communications of the ACM*. 38,5, 53-60.

Knoll, K., and Jarvenpaa, S., L., 1994. Information Technology Alignment or "Fit" in Highly Turbulent Environments: The Concept of Flexibility. *Proceedings of the 1994 computer personnel research conference on Reinventing IS : managing information technology in changing organizations*.

Lee, S., Leifer, R., S., 1992. A Framework for Linking the Structure of Information Systems with Organizational Requirements for Information Sharing. *Journal of Management Information Systems*. 8,4, 27-45.

Littlewood, B. and Lorenzo Strigini, 2000 L. Software reliability and dependability: a roadmap *Proceedings of the conference on The future of Software engineering* . May

Montz, A., B., and Peterson, L., 1998. Controlled Flexibility in Systems Design. Proceedings of the 8th ACM SIGOPS European workshop on Support for composing distributed applications.

Murray, T., 2002. Past and Furture Emergency Response Systems. *Communicatioins of the ACM*. 45(4), 29-32.

Prayurachatuporn, S. and Benedicenti, L. 2001. Increasing the reliability of control systems with agent technology, ACM SIGAPP Applied Computing Review. 9, 2, July

Rinard M., 2003 OOPSLA onward! track: Acceptability-oriented computing *ACM SIGPLAN Notices*. 38,12 December

Rivard, S., Poirier, G.,Raymond, L. and Bergeron, F. 1997. Development of a Measure to Assess the Quality of User-Developed Applications. *ACM SIGMIS Database.* 28, 3 June.

Vickery, S., Calantone, R. and Droge, C.,1999. Supply chain flexibility: An empirical study, *Journal of Supply Chain Management*. 35,3, Summer

Whittaker, J. A. and Voas, J. 2000 Towards a More Reliable Theory of Software Reliability, *IEEE Computer Society*. December

Whitworth, B., and Zaic, M., 2003. The WOSP Model: Balanced Information System Design and Evaluation. *Communications for the Association for Information Systems*. 12, 258-282.

Woods, D., D., 1993. The Price of Flexibility. Proceedings of the 1st international conference on Intelligent user interfaces.