Alistair McKinnon

Abstract

This report examines the construct of information systems quality. It is measured with the use of four questionnaire items that elicit perceptions of quality from the end users of the information system. These items query the level of functionality, information sufficiency, response time, flexibility, ease of use and overall quality of the system. The representational assumptions of the instrument are examined, with emphasis placed on the need for user experience of the system under examination. The evidence of construct validity is observed, which includes measures of unidimensionality, reliability validity, convergent and discriminant validity, with all except convergent validity holding true. Other studies using information system quality constructs are then discussed, with the conclusion that there exists a utility for the construct in both the practitioner and academic worlds.

Construct description

The construct under review in this report is information system (IS) quality (also known interchangeably as product quality). IS quality can be measured a number of ways. Ravichandran and Rai suggest that quality "reflects users' subjective evaluation of the features and functionality of the information system" (2000, pg 132). They have developed a number of questionnaire items that are used to measure the construct (figure 1).

Figure 1: Questionnaire items for the product quality construct (taken from Ravichandran and Rai, 2000, pg 154)

Users perceive that the system meets intended functional requirements.

The information provided by the systems meets user expectations.

Systems meet user expectations with respect to response time, flexibility and ease of use.

Users are satisfied with the overall quality of the system.

Functional requirements

Before an information system is developed, a requirements elicitation stage defines exactly what the system will do for the end user. The quality of the software will be high when all requested functionality exists. If functionality is missing, there should be a negative perception given by the questionnaire respondent.

Information sufficiency

This item attempts to elicit users perceptions on the quality of the information that the IS system provides. If the type or amount of information provided is of limited use to the user, the quality of the system will be lower than if all of the necessary information was available.

Response time, flexibility and ease of use

Response time is the time taken by the system to do a task and then become available to the user again. Very long response times are likely to frustrate the user and lower perceptions of quality. Also, if the system is excessively rigid in its requirements of use, perceptions of quality may suffer. Software with a steep learning curve or an unintuitive interface may also result in user perceptions of quality that are low.

The above descriptions of software quality and the questionnaire items for the construct will be used as the basis for this report. The questionnaire items use a seven-point Likert scale with the mapping of respondent opinion to numerical values as shown in table 1.

Table 1Mapping of user perceptions to numerical values (seven-point Likert scale)

Numerical value	User perception
1	Strongly agree
2	Disagree
3	Disagree slightly
4	Neutral
5	Agree somewhat
6	Agree
7	Strongly agree

The target of the questionnaire items is the end user of the software that is having its quality level evaluated. In an experimental study where the software is specifically developed for that study, the end user is likely to be an expert judge. In an industrial setting, the end user will be the customer, or in some cases the developer.

Relation to other constructs

IS quality is related to a number of other constructs. Ravichandran and Rai (2000) identify eleven quality management constructs and two quality performance constructs (of which one is product quality) that can be used in IS research. As part of the validation process of these constructs the authors performed an assessment of criterion-related validity. This established that all but two quality management constructs were related to the product quality construct. Table 2 shows these constructs and their relationship significance to the product quality construct (*t*-value).

Table 2: Criterion-related validity of quality management constructs to product quality (adapted from Ravichandran and Rai, 2000, pg 139)

Construct	**Significance of relationship to product quality construct (*t*-value)**
IS management commitment to quality	10.9
Quality policy and goals	20.4
Commitment to skill development	20.0
Quality orientation of reward schemes	40.1
Formalisation of analysis/design	20.5
Fact-based management	40.5
Process control	40.4
User participation	30.4
Programmer/analyst empowerment	50.1

Team ability

Farr-Wharton (2003) suggests that capacity, competency and capability are three important characteristics that, in the right mix, should bring about project success when using teams. The capacity of a team can be seen as the frequency of delivery of the resources in a project to generate an outcome (Cheng, Simmons and Richie, 1997, cited in Farr-Wharton, 2003). As software development teams are working with computers rather than materials as in a manufacturing setting, resources can be seen as computing hardware and system developers. If a team has poor capacity, the quality may suffer due to an inability to meet the requirements initially developed.

Competency is defined as the specification of knowledge and skill, whereas capability is the tacit knowledge built up from experience (Farr-Wharton, 2003). Both of these can be based at the organisational or individual level. An experienced team (high capability) that does a particular task well (high competency) should be expected to generate higher quality than one that does not.

Representational assumptions

Stablein (1996) provides a number of representational assumptions for Likert-style research. These include truthful responding, item understandibility, the ability to make the required judgement, and the ability to relate a judgement to a numeric scale (Stablein, 1996, pg 514). All of these points are relevant to the product quality construct. In addition to these general representational assumptions, a number of more specific assumptions in relation to the construct questionnaire items are now explored. The following points detail some assumptions of participants in questionnaires that have the product quality construct as a measure. If these assumptions do not hold, the validity of the results is likely to be in question.

- In rating the functionality of a system, users will know what the original scope of the system was intended to be.

- In rating the quality of the system information and functionality, a user will have had adequate opportunity to explore the entire system.

- The user's perception of what constitutes adequate information is based on norms in the particular application area rather than an expectance of absolute perfection.

- When considering the ease of use of a system, the user will be adequately trained in the system and have a reasonable degree of computer literacy.

Validity evidence

Ravichandran and Rai (2000) provide validity evidence for the product quality construct. Unidimensionality is the degree that measurement items represent an underlying trait. A goodness of fit index of 0.90 was calculated for the product quality construct, which is at the recommended level for unidimensionality (Ravichandran and Rai, 2000).

Reliability validity indicates the level of error and the ability for that measure to provide consistency. Cronbach's alpha can be used to test for reliability, and Ravichandran and Rai (2000) calculated this to be 0.82 for the product quality construct, which is larger than the required level of 0.70 for reliability. A composite reliability measure using the Werts, Linn and Joreskog ρ_c was also found, with a result of 0.83 which is well above the 0.5 threshold (Ravichandran and Rai, 2000).

Convergent validity was examined using the Bentler-Bonnet coefficient, which attempts to determine if differing approaches to construct measurement give the same results. A value of 0.87 was calculated, which is lower than the required threshold value of 0.90 (Ravichandran and Rai 2000). Although this is close to the threshold, it indicates that each questionnaire item may not be measuring product quality to the same degree.

Finally, Ravichandran and Rai (2000) examine discriminant validity, which evaluates the uniqueness of a construct. Confirmatory factor analysis on pairs of scales with construct correlation was performed, followed with a repeat of this but with the construct correlation constrained to 1 (Ravichandran and Rai, 2000, pg 137). A chi-square difference test was found to be significant for the product quality construct, signifying that discriminant validity holds.

Ocker, Hiltz, Turoff and Fjermestad (1996) is one of a limited number of studies examining product quality in information systems in a qualitative sense. Whilst they do not use the same measurement instrument as used by Ravichandran and Rai (2000), a number of similarities exist. A significant part of the quality construct in this case is functional requirements analysis and design, which includes system functionality and the systems interface (Ocker et al., 1996). Information on the validity of the quality construct in this case is not given, however it is noted that the approach used was developed by Olson, Olson, Carter and Storrosten (1993, cited in Ocker et al., 1996) after a comprehensive study including software development academics and professionals.

Parzinger and Nath (2000) measured software quality by questioning developers on their perceived level of customer satisfaction, capability maturity model level and compliance with ISO 9000-3 software standards. There is a limited relationship between how quality is measured here and how it is measured in Ravichandran and Rai (2000). The main similarity is the user satisfaction item, which is simply a yes/no question to user satisfaction, and a 1 – 5 Likert scale for the level of change in quality perception due to the implementation of a total quality management system. Again, no validity information was provided.

Coupe and Onodu make use of a number of measures for software quality, including capability, reliability, usability, efficiency, accuracy, reusability, security and interoperability (1996, pg 177). These items were used in a questionnaire to developers to build up an understanding of software quality using differing processes of software creation. Capability, usability, and elements of efficiency and accuracy are covered in the construct defined by Ravichandran and Rai (2000), however the target of the questionnaire in this case, as with the last case is the developer, rather than the end user. This limits the comparisons that can be made between them, as developers often have different focuses than end users (i.e. technical quality as opposed to user interface quality). Validity evidence was not provided by Coupe and Onodu (1996).

Conclusion

The construct and measurement instrument provided by Ravichandran and Rai (2000) provides good evidence of validity. However, there is no single definition of what constitutes 'information system quality' or 'software quality'. This lack of focus has resulted in a dearth of research to fully test the utility and flexibility of such a qualitative instrument. Therefore, it is uncertain how valid the construct will be outside of IS research. Quality can also be measured in a quantitative manner, by looking at reliability (faults per day), average response times and so forth. An improvement in the measurement of software quality might include some of these quantitative measures.

Managers of organisations that in some way deal with software development may find this construct useful when examining how customers perceive product quality. It is in the interest of those managers to maintain a high level of quality to generate positive word-of-mouth and to reduce maintenance expenses. Because this construct is measured from responses from the customer, a more accurate evaluation of quality should result, as they are the group that must actually use the software.

This construct is also useful for researchers in an experimental setting or in the field. Expert judges can be used to answer the questionnaire when software is developed specifically for a study (i.e. students at a university building an information system for a research study). Researchers looking in on an organisation can either use system customers if they are available, or alternatively the developers of the software as subjects in their research.

References

Coupe, R. T. & Onodu, N. M. (1996). An empirical evaluation of the impact of CASE on developer productivity and software quality. *Journal of Information Technology, 11,* 2, 173-182.

Montoya-Weiss, M. M., Massey, A. P. & Song, M. (2001). Getting it together: temporal coordination and conflict management in global virtual teams. *Academy of Management Journal*, *44*, 6, 1251 – 1263.

Ocker, R., Hiltz, S. R., Turoff, M. & Fjermestad, J. (1996). The effects of distributed group support and process structuring on software requirements development teams: results on creativity and quality. *Journal of Management Information Systems, 12,* 3, 127-153.

Parzinger, M. J. & Nath, R. (2000). A study of the relationships between total quality management implementation factors and software quality. *Total Quality Management, 11,* 3, 353-382.

Ravichandran, T. & Rai, A. (2000). Total quality management in information systems development: key constructs and relationships. *Journal of Management Information Systems, 16,* 3, 119-155.

Stablein, R. (1996). Data in organizational studies. In S. R. Clegg, C. Hardy & W. R. Nord (Eds.). *Handbook of Organizational Studies.* London: Sage, 509-525.