2001

Issues To ConsiderWhen End-Users Develop Their Own Applications

Annemieke Craig
Victoria University of Technology, acraig888@gmail.com

Rhonda Dixon
Victoria University of Technology

Issues To Consider When End-Users Develop Their Own Applications

Annemieke Craig and Rhonda Dixon

School of Information Systems
Victoria University of Technology, .Melbourne, Victoria, Australia
annemieke.craig@vu.edu.au

Abstract

The purpose of the research discussed in this paper is to identify the issues that need to be considered when end-users develop their own applications. A case study was conducted with a semi government department, which explored the reasons why end-users developed their own applications, the advantages this development brought to the organisation and the problems that could result. Some of these problems include support, maintenance, design difficulties, management and the lack of control and documentation. Possible solutions for these issues are also briefly discussed.

Keywords

End-user, application development

INTRODUCTION

The aim of this research was to identify the issues raised when end-users develop their own applications. This area is important to the Information Systems discipline because as technology grows and becomes cheaper and easier to use, so does end-user computing grow within organisations. Organisations, both large and small, need to be aware of the issues that may arise so that they can maximise the benefits of end-user computing and minimise any negative effects.

End-users may become involved in the development of applications for a variety of reasons. In many organisations there is a backlog of application requests. When IT staff cannot meet the demand, end-user development reduces this backlog (Ampaichit 1998). The ongoing need for applications to be developed and changed often left end-users frustrated while they wait for computing staff to develop systems and implement changes. This unfulfilled need was one of the initial catalysts for the end-users to become involved in development (Regan 1989).

An increased level of computer skills amongst employees and the availability of software, which makes development possible, are additional reasons for the increase in development of applications by non-computer professionals. End-user development is most often accomplished by using software such as Financial Software (including spreadsheets, financial modeling, statistical analysis and financial management packages), Word processing, Fourth-generation languages, Data management and Computer Aided Design (Regan 1989).

Ampaichit (1998) described end-user computing as a necessity for the following reasons; the cost of employing people to develop applications is increasing rapidly. End-user development using fourth-generation languages can eliminate some of the labour and programming costs, as well as save time. Business conditions have also become increasingly difficult. Managers and staff need access to more information in a smaller timeframe and this can be accomplished with end-user development.

Benefits of End-User Development

The advantages of end user development are that end-users design systems to meet their needs and schedules. They do not need to interpret their needs for SDLC developers and systems are available more quickly (Dodd and Carr 1994). Edberg & Bowman (1996) carried out an empirical study on application quality and developer activity in relation to user developed applications. They found the main benefits were the improvement in employee productivity and performance.

In an Australian study conducted by Zeffane & Creek (1998), a strong positive relationship was found between conflict resolution, user participation and project success. Conflicts could be resolved easier as people communicated more frequently and had a better understanding of the system. It has also been noted that increased user involvement in system development also increases the level of system acceptance. The levels of user expertise and user involvement also have a relationship with the level of system acceptance. An empirical study conducted by Saleem (1996) found that "users who perceive themselves as functional experts, relative to others, are unlikely to accept a system unless they exerted a substantive influence on its design". Consequently

end-users expect to be consulted and involved during the design phase of an application, even to the extent of developing the system itself.

Other advantages identified included improved strategic planning, initial requirement specification, testing and implementation (Zeffane & Creek, 1998). This study also noted the improvement in how staff perceived the amount of user involvement within their organisation in 1993 and later in 1998. In the 1993 study, 33% felt that there was insufficient user involvement, whereas in 1998 this percentage had decreased to only 5%.

Issues / Disadvantages of End-User Development

The Zeffane & Creek (1998) study also recognised that involvement in application development is 'not without cost and should be carefully managed'. While this study found that end-user involvement improved data quality, Edberg & Bowman (1996) reported decreased quality of the product or application itself. This lack of quality was attributed to the lack of development knowledge. A major risk resulted for the organisation as incorrect business decisions were being made based on information produced by applications created by end-users. This was due to user developed applications which were "incorrect in design, inadequately tested and poorly maintained" (Edberg & Bowman, 1996).

Other disadvantages of end user development as identified in the literature include;

- End-users might use inappropriate tools. The users tend to use the software that they are familiar with, where another type may be more appropriate. For example, using a spreadsheet to do something that could be done more effectively using a database (Ampaichit 1998).
- There is a potential for data and application redundancy (Dodd and Carr 1994).
- The cost of expert users providing peer support to other end-user developers can be hidden and these experts may be sidetracked from their primary responsibilities (Ampaichit 1998).
- Data integrity may be compromised. User developed applications often don't provide audit trails or follow normal quality assurance procedures (Dodd and Carr 1994).
- If end-users do not have the organisational responsibility for maintaining these applications the applications may be out-of-date and be missing vital maintenance revisions.
- End-users do not have the experience of developing logical algorithms required when coding.
- Analysis was one of the areas where end-users proved to be less proficient even though they had a better understanding of what was required (Edberg & Bowman, 1996).
- Security and control measures may not be adequate e.g. lack of validation and quality assurance testing, backup and recovery plans and documentation may be lacking (Edberg & Bowman, 1996).

Responsibility for supporting end-user developed applications

Jenkins (1997) provides guidelines for the procedures and policies required for successful end-user computing. These guidelines include how end-user computing should be supported with assistance provided either externally or internally. Support arrangements would differ however, according to the corporate policy of the organisation.

A Solution – System Development Led by End-Users

According to Dodd and Carr (1994) the solution to the problems caused by applications being solely developed by end-users is to have the end-users lead the development of the application. The end-users are then heavily involved in all parts of the development cycle where their expertise is dominant.

System development led by end-users (SDLU) could involve;

- Users request a new computer-based capability
- Create a team with users and Data Services members, and a user team leader
- Led by users, team expands the request and defines a system
- The team creates a prototype of the system
- Users try the system via the prototype and change the description
- Team refines the system after studying the prototype
- Data Services creates the application with continued user involvement and leadership
- Users accept the system
- Users accept delivery and ownership of the system
- User-lead team retains responsibility for maintenance and change of capability (Dodd and Carr,1994)

The importance of this integration was also recognised by Zeffane and Creek (1998) who advocate that end-users and information systems departments 'must be able to work hand-in-hand with departments and work groups towards strategic goals'.

RESEARCH METHODOLOGY

A small semi government organisation in the Treasury and Financial services area was chosen for the case study because:

- The authors were aware that the organisation encouraged end-user development, and that the organisation had a management strategy for handling end-user developed applications.
- The organisation was easily accessible as one of the authors was employed there in the System Development department as a System Analyst.

The main focus of the investigation was exploring the reasons why the end-users in this organisation, developed their own applications, the advantages this development bought to the organisation, the resulting problems and the strategy the organisation used to manage them.

Interviews were the main research method used and were held with the following people:

- Three end-users currently involved in application development. All three people were Quantitative Analysts in either the Front Office (dealing room, trader support) or Middle Office (risk management). They all held Business & Economics Degrees and had worked in similar positions in the financial sector for three years or more. Each person was currently undertaking post-graduate study relating to financial markets e.g. Securities Institute. One of them had also been employed previously as a contract programmer using Visual Basic.
- The Middle Office Manager who was responsible for the end-user developers. This person held a senior position within the organisation, with expertise in Risk Management issues, and had worked in similar senior positions in larger organisations for 15 or more years (qualifications not available to authors).
- The Systems Development Manager, who was responsible for in-house systems application development. This person held a Degree in IT and had just completed a Masters in Applied Finance. He had worked in the System Development area in the financial sector for over 10 years.
- The Systems Operations Manager, who was generally responsible for computing issues within the organisation. This person had previously worked in an international financial software development house for over 12 years. He was initially employed as a developer and eventually became an expert trouble shooter and client manager. He had only recently moved to this organisation.

The interviewees came from distinct backgrounds, either Financial Markets or Computing, with only a few people having expertise in both areas. The interviews were held late in 2000.

CASE STUDY RESULTS

The organisation interviewed was in the Treasury and Financial Services area. Due to the nature of the industry the main areas of end-user development applications were spreadsheets and databases used in quantitative work. These spreadsheets and databases were used to create applications which would help the organisation identify risk, explain profit and loss, and perform complex analysis.

How did the end-user developers learn to develop applications?

Each of the end user developers (EUDs) learnt the majority of their development knowledge 'on the job' and via 'trial and error'. They had all completed basic courses in MS Excel and / or MS Access, however none of the training included how to code within these applications. Two of the EUDs interviewed felt that their area of weaknesses was VBA coding within these applications. The remaining EUD's first job was in a department where most quantitative analysts performed programming regularly. He later worked as a contract VB programmer, so he was strong in the VBA Code area.

One EUD interviewed was very familiar with MS Excel but had a little knowledge of MS Access and felt uncomfortable using it. He acknowledged that he often used the wrong tool when developing, only because of these deficiencies in coding and application knowledge.

Should end-users develop applications?

Each of the three EUDs interviewed felt that they should definitely be able to develop applications, and saw it as an integral part of their work. They felt that development gave them a better understanding of their work and expanded their knowledge by helping them realise what was possible. They also thought development enhanced their creativity.

The end-user developers recognised that their 'knowledge of the product' made application requirement specification easier and faster as the external developer was taken out of the loop. This 'knowledge of product' also helped them design better prototypes, which in turn led them to produce more applications that were useful to the organisation.

Limited information technology resources could cause delays when changes needed to be made to applications. If the end-users developed the applications, they found they had more flexibility to make the changes as required and that this saved them time as they didn't have to wait for IT.

The managers' responses on the other hand were quite different. The end-user manager made the distinction between an adhoc application compared to an application used for a daily process, or comparing an analytical process to a programming exercise. When it was an adhoc or analytical process he felt that end-user development should be encouraged.

The interview questions related to 'application development' by end-users, but the managers made a clear distinction between 'prototype applications' (development applications) and 'production applications'. All the managers agreed that for 'prototype applications' end-users should be able to develop applications, but when it came to the 'production applications' where it could have major effects on the organisation, end-user development should be limited. The managers had some concerns about applications used in the production environment.

The main concern the managers had was that the applications developed by end-users often did not follow system standards and were of poor quality. This lack of quality meant that the applications were poorly designed and had minimal fault tolerance with insufficient error handling routines. Macros were often used, which don't allow error trapping, so that when the macro had problems the application crashed. Whilst the power of end-user development is capturing their 'knowledge of the product', the weakness of end-user development is their lack of 'development expertise'. Edberg & Bowman (1996) also supported this lack of development knowledge to being the primary reason for a lack of quality in end-user developed applications

A lack of backup and security features in EUD applications can cause problems as well. For example - One EUD application that crashed half way through its processing could not be re-run, so the EUD had to perform some lengthy and complicated manipulation of data. If correct recovery features were in place, the data could have been saved at the beginning of the application and restored when the errors were encountered. This concern was also supported by Dodd & Carr (1994), Ampaichit (1998) and Edberg & Bowman (1996).

The system operations manager, with considerable experience implementing projects, highlighted that in EUD applications, there were often a lot of missing steps from the normal 'development process'. He said that 'when end-users develop applications there is a tendency to miss out some of the steps as they feel that they know the requirements and specifications already'.

Other concerns were:

> The managers felt that often EUD's used the wrong tool for development. The EUD's tended to use tools they were familiar with, rather than ones that could be more applicable.
>
> The end-user's focus may be too narrow in perspective to the application's use e.g. they don't have the 'big picture' of where it fits into the organisation.

Some of these concerns could lead to an audit issue and need to be taken into consideration when end-users are involved in developing applications. All of these issues are supported by the literature.

What would happen if the end-users were not involved in development?

Each end-user interviewed felt that developing applications whether large or small, was part of their job. If they were not involved in development they would not be able to carry out their functions within the organisation to the best of their ability. The job itself would be more difficult and they would not be as productive as they were currently and there would be deterioration in the financial modeling required by the organisation.

The managers were concerned that if the end-users were not involved in development, application specification would be much harder and applications would be less flexible. Problems would have to be spelt out and defined completely 'to the nth degree'. Applications would take longer to code, and testing would be much more cumbersome as mistakes would not be identified until later in the development process. IT and the end-user would likely talk at cross-purposes. The end-user manager also pointed out that end-users being involved in development offered the organisation the 'best of both worlds' i.e. a good combination of end-user expertise and systems expertise.

Benefits of End-User Development

The end-users mainly identified benefits to themselves or their immediate area. They felt that applications developed by them fitted their needs best as the 'communication gap' between themselves and IT did not occur

as it did in System developed applications. The end-users could also create the applications quicker as they did not have to wait on IT resources being available.

The end-users thought that developing applications improved their skills, which in-turn provided ongoing benefits to themselves and the organisation. For example, an end-user may 'develop a financial model for themselves, then realise that this same model can be used for other products within the organisation'. This improvement came about as development helped them gain a much clearer understanding of the processes, data, relationships and algorithms underneath, compared to a reduced understanding when IT had developed the application for them.

The managers identified these points, but also focussed on the overall benefits to the development procedure. The managers reported that EUD improved the processes because the EUD's noticed the mistakes in the specifications a lot quicker and could identify a better way of performing the processes during the development phase. Errors within these processes were also identified quicker. 'This occurs because the end-user reviews the results on an on-going basis during the development phase and identifies errors along the way, compared to checking the result after development has been carried out by systems'.

Not having to rely on the Systems department was a benefit, as EUD could free up some of the very busy system development staff from doing some of the smaller jobs. However the benefit of this might only be short term, as in the long run the applications often required more support. Maquignaz (1994, p20) also recognised the 'short-term' benefits provided by end-user computing but which in the 'long-term' would require improved management and the issues addressed.

Other benefits mentioned were:

- The end-users know exactly what they want. Needs information can often get lost in the communication between the user and the developer. When end-users develop their own applications this loss does not occur. This was also recognised by Saleem (1996) and Dodd & Carr (1994).
- EUD gives the end-user's control over their computing environment.

Overall, the managers agreed with the benefits identified by the end-users, but they also had a wider focus on the benefits to the organisation and the development process. These benefits were supported by the literature.

Issues / Disadvantages of End-User Development

The first response of each of the end-users interviewed when questioned about the disadvantages of EUD was a lack of knowledge and formal training in writing VBA code within the applications used by the organisation. The end-users felt that this lack of coding knowledge could lead to inefficiencies and incorrect design within the applications developed.

Many acknowledged that they did use the wrong tools at times, but they felt they had to use the applications they were familiar with. The managers had also identified this as an issue.

When EUD applications were originally designed, they were done with a particular use in mind for the individual end-user. If the application is to be used by other people within the organisation as well, there may be further adjustments to be made. All of the end-users mentioned the problems encountered with their applications when they were being transferred across to the production environment. One problem was that the end-users did not have a feel for the 'sequence of events' that processing required. For example, the EUD database had links to another database that may not have run as yet.

The end users also mentioned that:

- Development can often take more time than they would like. For example, the end-user who had previous VB programming experience found that when he wanted to learn the syntax to do SQL code in a major Excel project there was insufficient electronic help available. He had to spend a lot of time working out the correct syntax.
- Someone was needed to 'control' the work being developed by the end-users.
- EUD applications may not be up-to-date, if they are not updated to reflect the changes that occur.
- People using EUD applications need to know how to interpret the results and be able to identify incorrect figures and determine where the problem may be e.g. formulas in spreadsheets, SQL statements in queries.

The managers interviewed also identified and expanded the above points as well.

The System Development Manager highlighted the main issues by stating, 'the end-users are not specialists in the development field so issues such as bad design, incorrect logic, inadequate testing and not adhering to development standards occur'. Edberg & Bowman (1996) also support this statement in the literature. The

manager also noted that EUD applications require more support in the long term because of these design issues and because the applications suffered from a lack of discipline and structured programming techniques.

The Middle Office Manager also recognised this lack of quality by stating, 'as the end-users are the owners of the application they tend to be more forgiving of problems within in it'. This lack of quality was also mentioned in the Edberg & Bowman (1996) study that stressed the importance of the quality required when making informed business decisions based upon the information produced.

Another issue identified by the managers was that EUD applications are often not 'complete'. For example, it may 'be 90% right, but the EUD's don't worry about the remaining 10% as they know they can track it down'. This 10% may cause problems if used by other people not familiar with the application, or it goes into the production environment unchallenged. This previous statement ties in with the Middle Office manager's comments about being more forgiving of mistakes in regards to quality.

EUD applications also may have lack of security, backup and recovery features. Ampaichit (1998), Dodd & Carr (1994) and Edberg & Bowman (1996) all identified this lack of security controls and backup facilities as disadvantages of EUD. The Systems Operations manager used an example encountered with an EUD application when a problem was experienced. The application had crashed and could not be restarted. If it had recovery features, a backup would have been taken prior to the run, then put back to its original state when the problems were encountered.

Other issues identified by the managers were:

- Applications may be designed with a 'narrow point of view' that may not take into consideration of how the application is to be used within the organisation e.g. developers do not have the 'big picture'.
- Lack of documentation. End-user applications often do not have proper documentation, which becomes an issue if the end-user developer, who has the knowledge of it, leaves the organisation or if the application later goes into production and becomes critical to the organisation.
- It can be harder to manage resources. The end-user manager used the need to determine staff priorities and the allocation between development time and normal daily processes as an example.
- As there is no 'barrier to introduction', there can be a proliferation of end-user systems which require management.
- Data Inconsistency and Integrity and fragmented databases. For example, the 'end-user may retrieve data from a source that is quick and easy but not the best place'.
- Lack of robustness. Sometimes EUD applications may not have the same level of error checking and handling that occur in Systems Developed ones, which can cause applications to crash or hang.
- Fewer people know how the application works, which may cause difficulties if the end-user developer is not available for support.

Both the end-users and managers agreed with the issues / disadvantages discussed in the literature with particular emphasis on the 'Quality' of the finished application.

Responsibility for supporting end-user developed applications

Jenkins (1997) reported that support arrangements differ according to the corporate policy of each organisation. Within this organisation, when questioned about supporting the end-user-developed applications the replies were fairly consistent between the managers and the end-user developers. Each group made the distinction between prototype (development) and production applications. If the application was a 'production' application such as one within the overnight processing area then Systems provided the support and maintenance. If it was a one-of or single-person application that the end-user developer used for their individual use or a prototype application, then the end-user supports and maintains it.

End-users also write applications for within certain groups not just for their own single use e.g. the 'Quant Group', the 'Dealing Room'. The people within these groups are skilled users and can usually interpret the results. If help is required the end-user developers support these applications. An issue to rise from this situation is 'who is the backup person in case of the EUD being absent, unavailable or leaves the organisation'.

How does the organisation resolve the issues raised by EUD?

End-users are encouraged to develop their own applications. When guidance is required the Systems Operations and System Development departments mainly provide it. The main controlling responsibility for end-user developers are their individual managers.

When an EUD application is changing from 'prototype / single use' to 'production' use, the process at this organisation is that System staff review the application with guidance provided by the end-user developers. The application is then redesigned and modified to systems standards. The application is documented, which also builds the knowledge required to support the application and Systems then take the responsibility for the support and maintenance of the application.

This review process of the end-user developed application is also supported by the literature where O'Brien (1990) states that "the applications that end-users develop and implement must be evaluated for their efficiency and effectiveness in using organisational resources and helping to meet organisational objectives".

In practice this 'productionisation' process does not always occur fully. The end-user developer sometimes makes required changes to a production application, as they know the application better than systems staff. Also the application has not always been re-written according to systems standards, due to the urgency of putting the application into production as quickly as possible.

As well as the dividing line between production and prototype development, the organisation has other means of managing end-user developed application issues. These are the use of an IT Steering Committee, where both users and systems are represented. Projects are discussed and reviewed regularly and standards are often set here. Audit processes are implemented and post implementation reviews are held.

In this case study it was found that because of the distinct division between "prototype" and "production" environments, there were no perceived major problems caused by errors / bugs in the EUD applications as they didn't affect the critical part of the organisation. Issues only arose when EUD applications were rushed through the 'productionising' process and were not standardised correctly.

CONCLUSION

Benefits of End User Development

The literature had cited the main benefits / advantages of end-user development as the ability to manage changing conditions more easily, the reduction of the IT backlog, improved cost savings, communication, increased system acceptance, employee productivity and performance. The research supported the literature cited, but it also highlighted such areas as how the application fits and meets end-users needs better and produces more useful software for the organisation to use, as well as reducing the time spent on development. End-user development improves the control on their computing environment, knowledge, performance, understanding, creativity, flexibility, skills and processes, again all of the above supported by other literature.

Issues / Disadvantages of End-User Development (EUD)

The literature cited many issues that needed to be considered. These issues included reduced data integrity, product quality and functionality, incorrect design, insufficient testing, the use of inappropriate tools and lack of documentation, controls, backup and security provisions. The case study also noted other issues, such as proliferation of end-user applications that require management, decreased robustness, incomplete applications, resource management and, lack of coding knowledge, training, discipline and structured programming techniques. All of these issues have previously been identified, but the case study reinforces the findings of the literature. Some of the literature references were quite old, suggesting that the problems have been around for a long time and still exist in the present time.

Solutions to Issues

One solution discussed in the literature was a 'System Development Led by End-Users' (SDLU) where the end-users were in control of the development process and involved in nearly every step of the development process except for the actual coding. This approach was quite old (1994) but was still relevant as it tried to utilise the areas where the end-users expertise is dominant (Dodd & Carr).

In comparison the case study research showed that the organisation encouraged and supported end-user development, but made a distinct dividing line between single use or prototyping and actual production environment. When an application would enter the 'production' environment the application would undergo a review and rewrite process by the systems department, so that most of the quality issues would be overcome, whilst still gaining the advantage of using the end-user knowledge and expertise. As well as this rewrite process other means such as the IT steering committee, post implementation reviews and audit procedures were in place.

RECOMMENDATIONS

In principle the solution put into place by the organisation works very well. The dividing line between prototype and production environments provides a safety net that reduces the risk the organisation faces from EUD applications.

Recommendations for the organisation are:

- Continue to keep the end-users involved in application development where their expertise can be utilised.
- Make the transition process from 'prototype / development' to 'production' more formal. For example, do not bypass the review, redesign and rewrite process so as to expedite the application into production more quickly. This may mean that the organisation may need to allow more time and systems resources for application implementation.
- Train the end-user developers in the use of different development tools. Training should also cover such things as VBA coding, coding standards within the organisation etc.
- Where the end-user developer provides support e.g. 'Quant Group' or 'Dealing Room' applications, train other personnel to be the backup support.

POST PAPER OBSERVATIONS IN 2001

The interviews in the case study were conducted late in the year 2000. This section contains some post paper comments made by one of the authors (Dixon), who still worked at the organisation one year later.

Of the original people interviewed only two people remained at the organisation. In the meantime, the Systems Operations Manager in the original interviews, had been promoted to IT Manager, which included System Development. The System Development Manager position no longer existed. The new IT Manager has some reservations about End-User development.

Even with the IT manager's reservations, end-users still need to, and do, develop applications the same as before, although the productionising process is more formal with fewer EUD applications slipping through the net.

A presentation to end-user developers on basic development standards and naming conventions is scheduled in the next few months to help ease some of the design issues. Software quality and reliability is becoming even more important to the organisation and is of major interest to the new IT Manager.

End-users have requested new development tools e.g. Visual Basic, but have to put forward an extremely good case to the IT Manager to get them. IT development staff are even more under-resourced, so end-users play a significant role when the need for new applications arise.

The organisation is currently changing its main software from a suite of in-house developed applications to a commercial package. This new system has required taking a new look at the business processes involved and this too needs work-around applications to be developed. The majority of the expertise for the work-arounds comes from End User development and the Project Team.

In conclusion, with a management strategy in place to handle the issues that arise from EUD, the benefits of EUD can be realised. End-user developers play an important and integral role within the organisation due to their knowledge in their areas of expertise, which cannot be replaced by IT staff. What IT can do to help, is provide guidelines on standards etc. and ensure that the productionisation process is followed.

REFERENCES

Ampaichit, N. (1998) "*An exploratory study into the Level of Effectiveness of User-Developed Applications and Commissioned Applications*", STA Thesis, Information Systems, Faculty of Business, Victorian University of Technology. 005.1 NIT

Dodd, J. & Carr, H. (1994) "*Systems Development Led by End-Users: An Assessment of End-user Involvement in Information Systems Development*", Journal of Systems Management, 8/1994, 34-40

Edberg, D & Bowman, B (1996) "*User Developed Applications: An Empirical Study of Application Quality and Developer Productivity*", Journal of Management Information Systems, Summer 96, 13, 1, 167- 186.

Jenkins, G. (1997) "*Information Systems Policies & Procedure Manual*", Prentice Hall, New Jersey, 17.1 – 17.36.

Maquignaz, L. (1994) "*Strategic Planning for End User Computing – A Case Study of End user Computing at the Brotherhood of St Laurence*" Occasional Paper 4-1994, Dept. of Business Computing, Victorian University

O'Brien, J. (1990) "*Management Information Systems – A Managerial End User Perspective*", Library of Congress Cataloging-in-Publication-Data, ISBN: 0-256-07862-9

Regan, E. & O'Connor, N., (1989) "*Automating the Office: Office Systems and End-user Computing*", Macmillan Publishing, USA

Saleem, N. (1996) "*An Empirical Test of the Contingency Approach to User Participation in Information Systems Development*", Journal of Management Information Systems, Summer 96, Vol. 13 Issue 1, 145-167.

Zeffane, R & Creek, B et al, (1998) "*Does User Involvement During Information System Development Improve Data Quality?*", Human Systems Management, 1998, Vol. 17 Issue 2, 115 – 122.

COPYRIGHT