DIVISION OF COMPUTER SCIENCE

Identifying the Requirements for a Tool to Support the Adoption of SSM within Business Systems Analysis and Design

C Pascoulis

Technical Report No. 277

January 1997

Identifying the Requirements for a Tool to Support the Adoption of SSM within Business Systems Analysis and Design

Chris Pascoulis

Abstract.

SSM has been used successfully in a number of varied and complex problem areas throughout industry (Checkland & Scholes 1990). However, the flexibilty which the method advocates can also be perceived as lacking structure, particularly to those with hard systems backgrounds such as business systems analysts (Oakes 1994). This coupled with its perceived complexity can intimidate non SSM users who may not adopt the methodology due to a lack of confidence (Mingers & Taylor 1992). The SSM research community has been debating whether tool support can address some of the problems mentioned (Avison, Golder & Shah 1992). This report considers the requirements of a tool to support the adoption of SSM by Computing oriented users.

1. Introduction - Context of the Report.

The purpose of this report is to investigate the area of tool support for Soft Systems Methodology (SSM), with particular attention being paid to the adoption of the method in Business Systems Analysis and Design (BSAD).

It forms part of a research programme which examines the impact of tool support on the use of SSM in BSAD. The research will identify issues relating to the acceptance of SSM and how tools influence business analysts using the method.

The objectives of this report are to identify the kind of tool needed to support the adoption of SSM in BSAD.

This is carried out by evaluating existing SSM tools based on evaluation criteria defined during the investigation. The requirements for a proposed new tool are defined after issues relating to the use of SSM are identified from a literature review. Additionally, the practical experience of SSM users is carefully considered alongside academic debate on the issue of tool support for SSM in an effort to identify functional requirements (Pascoulis 1996).

Before embarking on the investigation, it is necessary to consider the relevance of CASE tools in the context of BSAD. This defines a set of criteria which will be used to assess the SSM tools mentioned. These criteria are used in the definition of the requirements of a proposed tool.

2. Overview of CASE Tools.

2.1 Advantages and Disadvantages of CASE Tools.

One route being taken to overcome the challenge of developing quality systems and software rapidly is to increase the use of automated tools. These automated tools are referred to as Computer Assisted Systems Engineering (CASE) tools. Systems engineering involves carrying out a number of tasks beyond that of simply writing software (CCTA 1994). The contributing tasks include - requirements analysis, database and file design, maintenance, configuration management, project management and development of user documentation (CCTA 1994). Due to such a wide variety of possible applications for CASE tools an all encompassing definition could be stated as follows:-

'any software tool to assist a systems engineer with any aspect of the development or maintenance process including management and planning activities.'

(CCTA 1994)

The benefits of using CASE tools include increased productivity, better documentation and improved communication. The disadvantages include high cost, training requirements and a potential lack of flexibility (Brown et Al., 1994).

For the purposes of this report it is useful to state the criteria by which CASE Tools are judged, and in this way define what constitutes a good tool. The criteria are separated into three categories covering the usefulness, usability and functionality of tools. These criteria can be mapped onto efficacy, efficiency and effectiveness respectively, which are known as the 3 E's in SSM and are used to monitor the performance of activities (Checkland & Scholes 1990). Using the criteria it is possible to evaluate the relative performance of tools. This is similar to the way in which SSM is used to identify system areas which fall short of the perceived ideal by comparing conceptual models with reality.

2.2 Usefulness Criteria.

The determining factor of how useful a particular tool is, depends on which application it is attempting to support, and whether it reduces the workload involved in carrying out the necessary tasks. In spite of this there are some generic features which are desirable in areas of analysis and design. The following criteria have been formulated to evaluate existing tools (Gane 1990):-

(i) Graphics

The ability of a tool to accurately document the analysis process as it unfolds is a specific requirement of a tool designed to support a methodology being

followed during this phase. A number of CASE Tools provide graphical editors which can produce Data Flow Diagrams, Entity Relationship Diagrams and many other recognised diagrammatic techniques used for analysis (e.g. Excelerator, Prokit Workbench, Software through Pictures). However, the measure of the success of a tool can be assessed by considering whether the use of these editors actually speeds up and improves the quality of the analysis process. The flexibility to create user defined diagrams which can easily be integrated into the user's existing environment is also desirable.

(ii) Repository.

The ability to provide a central repository where all relevant details, objects and properties of the analysis taking place are kept is a desirable feature of a tool (Pascoulis 1989). A repository can enable consistency and completeness checking of elements within a particular methodology to be carried out. Based on the rules governing the method being used, cross checking of different components may also be a desirable feature. Other obvious advantages of a central repository include the fact that all members of a project team access the latest version of all documentation (IDE 1989).

(iii) Other Features.

It is desirable for tools to facilitate more accurate project management and better quality documentation. In addition assistance during the design process should be provided in the form of templates where appropriate. Some tools have also furthered the design process by continuing into the next stage to generate code (E.G. StP, Excelerator). However, if the analyst is using a tool exclusively for investigation without integrating it into the implementation phase then the code generation functionality is in effect redundant.

2.3 Usability Criteria.

In order for the potential user of a CASE Tool to make full use of the tool's functionality the tool must be considered 'usable'. The tool designer can promote the usability of an interactive tool by following the principles of good interface design. The following sections use IDE's Software Through Pictures (StP) CASE tool to illustrate how these principles can be categorised into three main areas as defined by Dix et al.(1993) :-

(i) Learnability.

The tool must respond in a predictable manner enabling the user to deduce the outcome of a subsequent operation based on their previous experience. This predictability should arise from a consistency of actions within the package as well as exploiting the user's likely familiarity with other computerised tools or interactions. In StP the commands, responses and operations are consistent throughout the various editors of the package (Pascoulis 1989). The command language is self explanatory with few abbreviated commands which helps to contribute to the learnability of the package.

In addition the results of a particular action should be represented immediately through clear feedback. In StP pixel inversion is used to highlight selected menu entry, which is a clear acknowledgement of input (Hearn & Baker 1986).

(ii) Flexibility.

The dialogue should be as user pre-emptive as possible. In other words artificial constraints to the input dialogue imposed by the system should be kept to a minimum allowing the user the maximum flexibility in choosing which commands to enter next. Other principles giving greater flexibility are multi-tasking and customisation, which are both particularly appropriate to windowing systems. The facility to resize, move and close windows whilst a number of tasks are being performed is one of the strengths of windowing systems. Another principle of the interface is to allow the swapping of tasks between the user and the system where appropriate. An example of this within StP is the checking facility which can be invoked to validate the data dictionary of a project database (Pascoulis 1989). This returns errors and warnings which then prompts the user to address particular inconsistencies and then recheck the dictionary. Finally, the principle of substitutivity which allows the selection of equivalent commands to be substituted for each other is another way of adding flexibility to an interface. A CASE tool example of this in StP is the two available ways in which to access a Data Structure Editor. One uses an icon from the menu and the other works by double clicking on a data flow from within a Data Flow Editor (IDE 1989).

(iii) Robustness.

The principle of observability assists the user to determine the condition of the system where the information to do this is hidden in some way. One example of this in StP is on completion of the background task to check the data dictionary a message informing the user of the task's completion appears at the bottom of the screen.

Recoverability is a way of ensuring the user is able to back out of previous actions, once an error has been made. It is a design feature which can save a great deal of effort for the user.

Quick response times are also a desirable feature, but this is can be greatly effected by the hardware, and system software being employed. The interface software can give an immediate response to the user indicating that the requested action has been received and is currently being processed, regardless of whether the action has been completed, thus keeping the user informed of the tool's progress.

Finally, the ability of the software to complete a task adequately enabling the user to carry out all of their necessary functions without faults occurring is a definitive principle of robustness.

2.4 Functionality Criteria.

Gilb (1988) uses the State of California teacher system as an example to illustrate how the real goals of a system can be misunderstood. In this case the specified solution of a 'computerised system' became a goal in itself, where in fact the real goals were speed, reliability and more importantly low cost. Since the resulting automated systems' operational costs were more than seven times more expensive than the original manual one's, the new system was abandoned thus saving the state approximately a million dollars per year. This example illustrates the importance of establishing the absolute requirements of a system and ensuring that an erroneously specified solution is not mistakenly adopted as a goal to be aimed at.

Functional goals are defined as the absolute requirements of a solution (Gilb 1988). For a software tool this means that the functionality criteria should meet the ultimate objectives of a tool. In order to arrive at these objectives it is necessary to investigate the views of the prospective users and, in SSM terms, establish the Weltanschaaung (i.e. why a system exists) (Checkland & Scholes 1990). For a CASE tool the functionality criteria will depend on the problems being tackled by a particular software engineer.

This report seeks to identify the absolute requirements of a tool to support the adoption of SSM by business systems analysts. As a result the views of these analysts is a key factor in establishing the functionality criteria of a tool. The following section addresses issues relating to the use of SSM and concentrates on business analysts where appropriate.

3. Different Views of the Use of SSM.

3.1. Overview of How SSM is Used.

The suitability of SSM at tackling messy ill-structured problems is well documented (Checkland & Scholes 1990). Research carried out into the use of SSM shows that SSM users mainly choose to use it in order to 'ease a problem situation' and 'to develop understanding' (Mingers and Taylor 1992). This evidence supports Checkland's own experiences outlined in his account of the development of SSM which describes how SSM can be used in different modes (Checkland & Scholes 1990). The methodology was originally formulated as a stage by stage process known as Mode 1 which assumed that SSM would be applied by analysts outside a problem situation. Mode 2 developed as a way for managers within a situation to make sense of their day to day problems. However, there is no distinct dividing line between the two Modes, and a large number of the problem areas being tackled will fall between the two. Mingers and Taylor (1992) point to the use of SSM by most users being restricted to developing an understanding rather than as a way of intervention into the problem area. SSM is often complemented by the use of other methodologies in order to bring about change (Pascoulis 96).

3.2 Perceived Problems with SSM.

The research carried out by Mingers and Taylor (1992) into the use of SSM in practice involved sending 300 questionnaires to people with some prior exposure to SSM, of which almost half responded. Of these only 9% of the respondents had occupations within Computing compared to 20% each for Managerial, OR and Consultancy. The proportion of SSM users was approximately 65% of all respondents compared with approximately 50% of Computing respondents. In other words, there was a relatively small number of computing respondents and of those who did respond a lower than average proportion of them used the method. In general, this signifies that people in computing related occupations are less likely to use SSM than those in other occupations within the survey. Although these figures are not conclusive proof they do give an indication that SSM is not widely used by business analysts within the computing industry, a view which is of particular relevance to this report.

There were a number of reasons why respondents did not adopt SSM as a methodology even when tackling problems areas which it was well suited to. The main reason was a lack of knowledge and confidence on their part (Mingers & Taylor 1992). Other reasons included the perception that it was too complex and time consuming. It can be deduced that this perceived complexity stems from the uncertainty which can be associated with the way in which the SSM user is required to think about the problem situation. This therefore also translates to a lack of confidence being exhibited by inexperienced SSM users.

There were also a number of problems expressed by the SSM users' clients when they were introduced to the method. Lack of familiarity with the method led to difficulties when clients attempted to integrate SSM with an organisation's existing systems methodologies. The methodologies and techniques they were used to invariably involved specific checklists and milestones by which deliverables could be assessed.

SSM in its purist Mode 2 form can simply be employed as a means of finding out about the problem situation, as already mentioned. Therefore this finding out stage may not yield any deliverables. In addition, the jargon of SSM can make it difficult to sell SSM at the initial stages, and subsequently it can also make it difficult to establish trust between the SSM user and client (Mingers & Taylor 92). It has been suggested that these types of problems can be overcome by ensuring that the SSM user possesses the appropriate skills to deal with them, through training in consultancy and communication skills (Kreher 1994). In addition the introduction of 'more structure, in the sense of tighter project management' p1302 (Kreher 1994) is also put forward as a remedy.

SSM users also found problems with applying specific components of the methodology. There is not one particular area which all users found difficult, rather different users had problems with different parts of the methodology (Mingers Taylor 1992). In addition it has been found that the intended use of

some techniques within SSM such as rich picture diagramming, for instance, has been misunderstood (Lewis 1992). The drawing of an actual diagram of the problem situation is not an explicit requirement of producing a rich picture. Rich pictures can be used as a way of building a picture of the problem situation during the finding out stages of SSM. These stages could be just as well served by other diagrammatic or written techniques which the SSM user is most comfortable with. This is not always appreciated by users of SSM, and this misunderstanding illustrates the type of difficulty that can be caused by the user's lack of knowledge. Other problems documented about the use of SSM include the large amount of repetition required during the building of conceptual models (Pascoulis 1996), which supports the charge that SSM can be time consuming.

In summary, the main problems cited by users and potential users of SSM, are concerned with the complexity and uncertainty generated by its basic principles. The expectation is that once experience is gained in its use this lack of confidence is no longer a factor. The main problem then becomes the selling of SSM to the clients. Some business analysts using SSM find this task too risky and decide to conceal the methodology from the people within the problem situation (Pascoulis 1996). The main reasons for this are the concern that the people involved would become less forthcoming during the finding out stages, particularly if they felt that they were identifying themselves as causes of a problem situation. Clients may also find it difficult to take seriously a methodology that includes Rich Pictures as a technique, and perhaps may doubt that the methodology is as 'professional' as it might be. In addition, SSM users have found problems in carrying out specific techniques within the methodology.

3.3 Assessing the Need for Tool Support.

There are a number of stages within the methodology that might benefit from tool support, and as a result could perhaps go some way towards resolving some of the problems stated in the previous sections. Avison & Golder (1991) suggest that there is a definite need for tools to support SSM basing their argument on the way in which other support tools have helped information system developers using other systems methodologies. Their suggestion is not sufficiently well argued to be taken as read, particularly since they offer no research to support the assumptions specifically for SSM. In spite of this, they argue that tools to support 'techniques' such as rich pictures, root definitions and conceptual models would benefit SSM (Avison et al., 1992). The reasons put forward to support this view are true for methodologies in general. They include aspects such as validation, cross referencing, project management and other benefits as outlined in section 2. Avison et al. concentrate on rich picture diagramming first since it 'typifies the soft systems approach' p398 and the benefits of drawing aids have long been advocated in all fields of Computer Aided Design.

The view from the purist developers of the methodology is expressed by Kreher when he questions the suggestions outlined above (Kreher 1993). He

points out that rich pictures can include techniques such as data flow diagrams, usually associated with hard methodologies. which is supported by some business analysts in the field (Pascoulis 1996). This agrees with the view that analysts using the methodology in practice will use a wide range of techniques in order to gain an understanding of the problem situation. Kreher objects to the use of tools to support rich pictures, particularly if they are to use default screens or pictures, which he regards as potential constraints to the users creativity (Kreher 1993). Other objections include the appearance that a messy situation may in some way be cleared up by the use of a tool and thus lose the 'climate' of the original thought patterns of an SSM user.

3.4 SSM in Computing Workshop.

The University of Ulster in collaboration with the British Computer Society organised a two day workshop entitled 'SSM in Computing: The Use of Soft Systems Methodology in the Determination of Requirements for Computer-Based Systems' which was held in Ulster on 11-12 April 1994. The first day was simply a tutorial on SSM attended by approximately 20 students. This was led by Raymond Oakes of Brian Wilson Associates and described how the methodology can be applied to computer based systems. The second day examined 'Experiences and Issues' relating to the above title and was attended by approximately 60 participants from a variety of industrial and academic backgrounds. Speakers included Dr Brian Wilson one of the pioneers of the method from the University of Lancaster and Professor David Avison the co-author of the papers mentioned earlier. Speakers from British Telecom and the Northern Ireland Civil Service also gave a view from an industry perspective. Participants were then divided into groups each of which was asked to address a specific question relating to SSM in Computing, after which the conclusions were presented to the whole audience by the chairman of each group.

The question of tool support for SSM was one of the questions addressed during one group discussion which was attended by David Avison, and the author of this report Chris Pascoulis. The following is a summary of the group discussion and of views expressed throughout the two day workshop:-

There was general agreement within the group that tools were not essential and may not always even be desirable.

It was felt that pre-defined symbols and shapes for rich pictures and conceptual models may impose an excessively rigid framework which could inhibit the flexibility and creativity which SSM demands. However, the consensus was that support tools could help in the documentation and presentation of findings, and for consistency and completeness checking in the less dynamic stages of the SSM process. This could cover cross checking of areas such as root definitions with their respective conceptual models.

Many of the negative points made about tool support stemmed from the view that tools were too slow, cumbersome and unwieldy to be able to be useful during a dynamic brain storming finding out stage. Indeed Raymond Oakes

made the point that the rich picture, in whatever form was only a snapshot of a view. As such it should only be regarded as a means to an end, which would serve no-one, once a root definition had been arrived at. The drawing of conceptual models on the other hand, was seen as a laborious exercise, which might benefit from tool support, since it represents documentation of the purpose of a problem area being investigated and would by definition remain consistent. Finally, it was felt that SSM would benefit from tool support in the sense that the method would gain credibility simply by being supported. It was felt that this was particularly the case in the commercial software development community.

3.5 SSM Tutorial Tool.

Another area in which the need for tool support is being considered is in the teaching of SSM (Stowell & Stansfield 1991). It has been suggested an expert system would provide useful support for students being taught SSM. The way that a computerised tool can be used to interact with the user countless times without losing patience is put forward as a valid reason to develop a tool (Stowell, West & Stansfield 1991). The fact that the rule driven process needed to develop an expert system directly conflicts with the dynamic process of SSM is acknowledged. However, since no attempt to model the process of SSM itself is proposed, but rather a form of interactive support in carrying out some of the techniques used in SSM the authors maintain that 'the intellectual framework in which the methodology is seated' is not compromised.

West, Stansfield & Stowell (1994) have also described their development of a method known as Appreciative Inquiry Method (AIM) designed as a framework for iterative inquiry which is useful at tackling ill-structured information systems development problems. The method is based on SSM to an extent, but does not continue beyond the definition of conceptual models since it is not designed to bring about change. The authors go on to advocate the computerisation of this method as a progression of the tutorial tool proposed earlier.

4. Current SSM Tools Available.

4.1. Overview of Tools Available for SSM.

A number of SSM tools have been developed. The tools have concentrated on techniques within SSM and provide features to support the SSM user during the stages of SSM. Although not yet widely used the tools developed have approached the development from a number of different perspectives. As explained in the previous section tools to support rich pictures, root definitions and conceptual models are advocated and have now been developed for the purpose of supporting the SSM user (Avison et Al., 1992). Additionally a tool to support a student attempting to learn the techniques used within all stages of the SSM process has also been developed (Stowell, West & Stansfield 1991). West, Stansfield and Stowell (1994) outline their

more recent development of a computer based version of AIM which was undergoing field studies by 20 'domain experts'. Early results described the limitations of the tool in terms of its constraints on the user. However it was also shown to provide useful support in some more tedious repetitive tasks within the method.

This section concentrates on the evaluation of two specific tools in order to establish their suitability in providing tool support for the adoption of SSM in BSAD. The tools are a Rich Picture editor and a Maltese Cross tool.

4.2 Evaluation of Rich Picture Editor

4.2.1 Introduction.

A rich picture editing tool developed by M.G. Haywood at Aston University as part of ongoing research into the development of a set of CASE tools to support SSM, presents an opportunity to evaluate an SSM tool in terms of usability, usefulness and functionality. As a result it was decided to use various areas of a case study undertaken by Pascoulis (1996) in order to assess the tool. The tool was evaluated by redrawing a rich picture diagram drawn during the case study, and through a stage by stage assessment of its usability.

4.2.2 Description of the Use of the Rich Picture Editor.

The process of drawing the rich picture diagram using the tool is carried out using a relatively simple process. An icon is selected which brings up an entity dialog box. This allows the selection of an appropriate pre-defined picture to signify the entity chosen. There is also a box which allows text to be entered, which subsequently appears in the form of a think bubble when the rich picture diagram is viewed. Lines can be used to link the various entities, and pictures can also be selected at this point to allow a symbol such as crossed swords to be used.

4.2.3 Usefulness of the Rich Picture editor.

One measure of the usefulness of the tool is defined by whether the workload of carrying out the tasks involved in the analysis process is reduced with its use. This and other measures are determined by considering criteria in three main areas as described in section 2.2.

(i) Graphics

In spite of the apparent simplicity of using the editor, it did not speed up the drawing process. In fact it was far more time consuming than it had been when the picture was drawn by hand. The selection of the various pictures from a list of files was required before they could be loaded into the diagram and subsequently viewed. An icon to select each picture would have made it much easier to identify which picture was required. Another difficulty was not

being able to see the whole diagram on one screen, and therefore the scrolling up and down in order to move parts around was very slow, particularly when a P.C. is not powerful enough to give a good response time.

The pictures themselves were reasonably adequate, but there were not always enough different pictures to be able to represent the hand drawn diagram on which the diagram being drawn was based. However, the quality of the final product was satisfactory. It could be argued that a printed rich picture diagram appears much more professional than a hand drawn diagram. In spite of this comparison of the diagrams supports Kreher's view that a computerised diagram is somewhat sanitised and can appear to have solved a problem situation simply by having been printed.

(ii) Repository

Although a number of pre-defined pictures were available for use in drawing diagrams the tool did not possess a central repository of data. This is a significant drawback should communication with later stages of the method require easy access to elements within the diagram.

(iii) Other Features

The printing of Rich Pictures may provide better quality documentation, but the tool itself is very unlikely to have any influence on the quality of project management or system requirements resulting from the SSM process. Templates in the form of the pictures mentioned earlier were very useful in speeding up the drawing process.

4.2.4 Structure of the HCI.

On the initial screen which appears there is a Title Bar at the top and a Menu Bar below it with a Control Menu in the top left hand corner. There is a Minimise and Maximise Box in the top right hand corner and a Help Menu is also provided. Underneath the Menu Bar is an Icon Bar which enables the selection of the main functions within the tool. The initial function to be selected must be an 'Entity', which in effect corresponds to a pre-defined picture. Once this icon is selected a new screen appears called an 'Entity Dialog Box'. Various details can be filled in this box such as the 'Entity Label', 'Thoughts' and 'Comments', as well as the 'Picture Filename' to be selected. A line can also be selected to join two entities. This is again invoked using an icon and allows selection of picture using another screen called a 'Line Dialog Box'. A typical picture could be crossed swords to represent conflict. In this way a complete picture can be built up.

4.2.5 Usability Criteria for the Rich Picture Editor

The Human Computer Interface (HCI) of the package was evaluated during the course of the case study, based on established criteria for good interface design (Section 2.3). Furthermore, the criteria used are tailored towards

assessing a windowing environment since the tool runs under Microsoft Windows and was developed in Visual Basic (Preece et Al. 1994). The components of the package follow the usual Microsoft Windows structure.

(i) Learnability

Generally, the commands, responses and operations are consistent throughout the different windows of the tool. Although, a help facility is provided the help facilities are not very extensive, and are of no real assistance to either experienced or novice users. The absence of a 'demo' to show the user how the package works in a typical application is a notable omission. However, since the tool being tested is a 'freeware' version with some functionality removed, this explains some of the absence of some parts of the tool.

Throughout the tool pixel inversion is used to highlight selected menu entry, which is a clear acknowledgement of input, and as such enables very quick selection of menu items. Further forms of input include Scroll Bars, Push Buttons, Sliders and Radio Buttons which are all clear and simple ways to enter commands.

(ii) Flexibility

Error messages and diagnostics are informative and understandable, in general. Quitting or exiting the tool is quite straightforward and can be done by closing the main window as with most other Windows applications. The validation within the tool does not extend to areas which are specific to SSM, since the tool only supports the drawing of 'Rich Pictures' and no other stages of the methodology.

(iii) Robustness

Response to all input is immediate with partial results displayed as completed, building up a final display piece at a time, keeping the user informed of the system's progress. One further facility which would have been extremely useful is the 'Undo' command to enable the backing out of a previous action. No unexpected faults or errors occurred while evaluation of the tool was taking place which gives an indication of the software's high level of robustness and reliability.

4.2.6 Functionality Criteria for the Rich Picture Editor

In terms of the SSM learning process of enquiry the tool does not contribute very well. The general tasks being carried out during the finding out stages in order to build a rich picture of the problem situation require a great deal of flexibility which the tool does not provide. There is no provision to link the diagram to hard systems techniques such as data flow diagramming, nor does the editor cater for areas such as the three analyses within SSM. The tool can only claim to support the presentation a rich picture diagram to clients. It does not help a user whose knowledge of SSM is limited, nor does it provide

extra support to experienced users in terms of cross checking or validation of techniques. It could also be argued that the finding out stages of SSM are too fluid and diverse to be represented adequately by a tool.

4.3 Evaluation of Maltese Cross Tool.

4.3.1 Introduction.

A second tool, under evaluation was developed by H.Holden to support the Maltese Cross matrix which is used within Brian Wilson's methodology for tackling soft problems (Wilson 1984). Although this methodology is not identical to SSM, evaluation of the tool was considered relevant. In order to assess the tool parts of the case study mentioned earlier were used (Pascoulis 1996), as well as an example used by Brian Wilson to illustrate the use of the Maltese Cross (Wilson 1984). The tool itself is rudimentary, and uses a menu driven Non-Windows based interface which is unwieldy and inflexible. Reporting features are however useful for cross checking and analysis. In terms of functionality the Maltese Cross lends itself much more easily to computerisation than some other soft systems techniques (i.e. Rich Picture diagramming)(Holden 1994), regardless of drawbacks in the tool's usability when compared against established usability criteria (Preece et al. 1994). The evaluation is detailed below.

4.3.2 Usefulness of the Maltese Cross tool

The three sets of criteria used to evaluate this tool are the same as those defined earlier.

(i) Graphics

The tool does not possess a Graphical User Interface, and therefore, it cannot be integrated very well into a Windows environment. However, it was found that the tool speeded up the process of drawing a Maltese Cross matrix in comparison to drawing one by hand. There was an improvement both in terms of the quality of the diagram drawn and the ease with which elements of the matrix were filled in, in spite of the truncation of some lengthy activity names. However, these improvements were dependant on faults not having occurred during the use of the tool.

(ii) Repository

Although not immediately evident the tool does possess an internal repository which it uses for cross checking the elements of the Matrix. This is not available to workgroups within a project team since the tool is not designed to accommodate a number of users in a networked environment. Elements within the matrix cannot be viewed in a tabular format as in a data dictionary application, although a list of elements in various guises is available as part of the reports produced from the tool.

(iii) Other Features

As already mentioned the tool produces an output report detailing various cross checks carried out. This report is unlikely to improve documentation in a particular project because of the poor quality of its layout. This is caused by unclear headings and lines which wrap around leaving some words incomplete. Furthermore, the tool is unlikely to help a project manager to monitor a project team's progress, since the tool is not available within a networked environment, as mentioned earlier. Although menu access of quadrants within a matrix is available, the matrix as a whole cannot be viewed. Another drawback is that the selection of previously stored files is only possible by entering the entire file name.

4.3.3 Usability of the Maltese Cross Tool.

The HCI of the tool was evaluated in terms of its usability and the same criteria as those described earlier were used to assess its performance. The three categories are outlined below.

(i) Learnability

The command set of the tool is menu-driven and very basic, and as such it is easy to follow and consistent throughout. There is no help facility provided and although this is not essential for such a simple tool, some on-line guidance would have been useful, particularly in deciphering the report.

The overall quality of the interface suffers from being a Non-Windows based application. This partly explains the absence of quick and informative feedback and a lack of clarity from some command responses.

(ii) Flexibility

Error messages are limited and diagnostics are non-existent in this tool. Quitting the tool is in theory very simple but can be affected by the lack of robustness mentioned in the next section. Movement around the quadrants of the matrix is not always immediately obvious to the user with some key presses being ignored such as those for the arrow keys. There is also a lack of flexibility in moving between the quadrants when axis headings are being entered. This is particularly true when a new matrix is being built and entering at least one heading is mandatory before quitting the screen is allowed.

(iii) Robustness

The display is quite responsive though not very sophisticated. This is illustrated by the fact that error messages appear quite quickly but then disappear. As with the Rich Picture editor there is no 'Undo' command.

However, the biggest problem with the tool is the number of faults occurring during the evaluation. The main fault occurred when attempting to retrieve a

previously saved matrix. This had the effect of crashing the system and not allowing the user to quit, thus necessitating the machine to be rebooted. Another problem involved ignoring the saving of a file which was then lost. In general, this extremely poor level of robustness rendered the tool almost unusable, at times.

4.3.4 Functionality Criteria for the Maltese Cross Tool

Although the Maltese Cross was not originally used in the case study mentioned earlier, the introduction of the tool does help to add rigour to the cross checking of activities against inputs and outputs. The most useful component of the tool is reporting facility. This enables the production reports detailing "Data Coherence", "Deficient Production of Information Categories" and "Possible Data Base Formation" all of which could help identify shortfalls in performance of a system against the perceived ideal. However since the usability of the tool was so poor this restricted its potential effectiveness. Were the reports better laid out and the system more robust then the functionality would have had a more favourable assessment.

5. Potential Requirements For Tool Support.

5.1 Overview of requirements.

In order to outline the requirements for the tool to be built, it is useful to state the objectives of the research being undertaken for which the tool will provide a central role.

The aim of this research is to establish the ways in which appropriate computer assisted tools can impact on the use of Soft Systems Methodology (SSM) in Business Systems Analysis and Design (BSAD).

The research will identify how tools effect the acceptability of SSM in BSAD and how they modify the process of clarifying and representing soft problems. The research will investigate in what respect tools can be shown to influence the adoption of SSM by non SSM users. The opinions and perceptions of business analysts will be documented during their use of tools in the resolution and understanding soft problem areas.

In order to be able to address the questions above the research must use a tool which meets the usability and usefulness criteria outlined earlier, but more importantly it must meet a number of functionality criteria. The research into the use of the methodology has shown that SSM users find different aspects of the methodology difficult to use, and non-SSM users do not adopt the method largely due to its perceived complexity (Kreher 1994, Mingers & Taylor 1992).

In spite of the inconclusive research into the need for tool support it has become evident that there are a number of desirable requirements for a prospective tool (Avison et Al. 1992, West, Stansfield & Stowell 1994). These include the provision of guidance in the use of techniques within SSM

which could take the form of the tool acting as a tutor and prompting the user where necessary, as advocated by Stowell & Stansfield (1991). Other requirements concern cross checking and consistency validation performed by the tool to ensure that certain rules of the method are adhered to (Avison et Al 1992). These could develop into automated labour saving processes where appropriate. For instance, Root Definitions could be formulated automatically from the CATWOE, and Conceptual Models generated from the root Definitions, as advocated in part for AIM by West, Stansfield & Stowell (1994).

One further issue which has not been considered until now is that of resources. The limitations in terms of the number people developing the tool and the time and budget constraints should also be considered at this stage. Although this is not a tool requirement it is a relevant development issue which needs to be addressed before the tool design stage begins (Gilb 1994).

5.2 Requirements Definition.

Based on case study experience (Pascoulis 1996) and the research outlined earlier, the basic tool requirements can be outlined under the three criteria headings used thus far. The following sections describe the three sets of requirements.

5.2.1 Functionality Requirements of Proposed Tool.

The arguments described in the previous section lead to the following areas of functionality being defined. The functionality requirements of the tool are defined by whether the tool can help support the adoption of SSM in BSAD and as such help to address the research question outlined earlier.

The first functionality requirement is that the tool provides a simple way of translating the elements of the CATWOE mnemonic into a root definition and a subsequent conceptual model. Additionally, the tool should provide guidance when asking the user to enter basic elements. The final functionality requirement is that the tool should allow multiple Root Definitions and Conceptual Models within each 'process of enquiry'.

5.2.2 Usefulness Requirements of Proposed Tool.

The usefulness of the proposed tool should ease the workload during the functional areas explained. To achieve this the provision of a simple graphical editor to support the drawing of conceptual models is a proposed requirement of the tool. An additional requirement is a central data repository designed to facilitate cross checking of different elements within the tool and to support data retrieval across a project team. The facility to enable the documentation of the design process is also a requirement. The documentation could take the form of printed reports showing root definitions and conceptual models from the analysis.

Additional usefulness requirements include the provision of templates at two

stages of the analysis process. One providing a standard wording for the root definition leaving only the elements of a CATWOE to be entered by the user, and the other providing a pre-defined format for the conceptual model dependent on the definition. Once the CATWOE elements have been entered the tool should use a basic template to automatically produce a Root Definition of its own which should be amendable. On completion of the Root Definition a Conceptual Model should be generated automatically, again using a basic template incorporating the elements of the Root Definition which will have been input. Activities within a conceptual model should be movable and amendable on the screen.

5.2.3 Usability Requirements of Proposed Tool.

The usability of a tool is particularly dependent on the quality of its user interface. A primary requirement for the proposed tool is that it provides an interface which possesses a consistent command language within the tool and exhibits predictability by making use of the user's likely familiarity with a Microsoft Windows environment.

Standard Windows facilities such as multi-tasking and interface customisation should be supported. A further requirement is that the speed and clarity of feedback should match that of the Windows environment with which the tool is integrated. Finally, the most important usability requirement is that the tool is robust with no major faults occurring during operation of the tool.

6. Conclusion.

Since the SSM tools which were evaluated did not meet the criteria defined earlier in the report it has become clear that a new tool must be built in order to address the research question. The requirements outlined in section 5 have been formulated by taking into account a combination of factors. The criteria laid out in section 2 outline generic features which are considered desirable in any CASE tool. The particular elements of SSM which benefit from tool support are identified from the literature. In addition, the problems cited in the literature on the use of SSM quite clearly point to the need for some form of guidance by the tool.

In spite of conflicting views on tool support for SSM by academics, SSM is generally accepted as an established method in the academic world. However, it is quite clear that there is limited acceptance of SSM in the computing industry. As a result the tool's requirements have been focused on catering for business analysts with limited prior knowledge of SSM. Particular attention has been paid to integrating the tool into an analyst's likely working environment. It is worth concluding that the requirements are likely to evolve through a number of iterations before being finalised by potential users of the tool.

References.

1. AVISON, D.E., and GOLDER, P.A. (1991). The Need For Tool Support For Soft Systems, Systems Thinking in Europe, Plenum.
2. AVISON, D.E., GOLDER, P.A. and SHAH, H.U. (1992). Towards an SSM toolkit: rich picture diagramming, Eur. J. Inf. Systs. Vol. 1.
3. BROWN, A.W., CARNEY, D.J., MORRIS, E.J., SMITH, D.B. and ZARRELLA, P.F. (1994). Principles of CASE Tool Integration, Oxford University Press.
4. CHECKLAND, P.B. (1981). Systems Thinking, Systems Practice, Wiley, Chichester.
5. CHECKLAND, P.B., and SCHOLES, J. (1990). Soft Systems Methodology in Action, Wiley, Chichester.
6. CCTA (1994). CASE and the Issues for IS Management, HMSO Publications.
7. DIX, A., FINLAY, J., ABOWD, G. and BEALE, R. (1993). Human-Computer Interaction, Prentice-Hall.
8. GANE, G. (1990). Computer-Aided Software Engineering, Prentice-Hall.
9. GILB, T. (1988). Principles of Software Engineering Management, Adison-Wesley.
10. HEARN, D., and BAKER, M. (1986). Computer Graphics, Prentice-Hall.
11. Interactive Development Environments Inc. (1989). Software Through Pictures manual, IDE, San Francisco.
12. KREHER, H. (1993). Critique of two contributions to soft systems methodology - Eur.J.Inf.Systs. Vol 2.
13. KREHER, H. (1994). Some Recurring Themes in Using Soft Systems Methodology - J.Opl.Res.Soc. Vol 45.
14. LEWIS, P.J. (1992). Rich Picture Building in the Soft Systems Methodology - Eur. J. Inf. Systs. Vol 1.
15. MINGERS, J., TAYLOR, S. (1992). The Use of Soft Systems Methodology in Practice - J.Opl Res. Soc. Vol.43.
16. OAKES, R. (1994). Soft Systems Analysis - A Practitioner's Guide, University of Ulster & BCS.
17. PASCOULIS, C. (1989). Investigating Software Tools for Program Design - BSc (Hons) Final Year Project Report - Polytechnic of North London.
18. PASCOULIS, C. (1996). The Problem Area of Invoice Matching at Neon Electrical - A Case Study of SSM in Business Systems Analysis and Design - University of Hertfordshire - Technical Report.
19. PREECE, J., ROGERS, Y., SHARP, H., BENYON, D., HOLLAND, S. and CAREY, T. (1994). Human-Computer Interaction, Addison-Wesley.
20. ROSENHEAD, J. (ed.) (1989). Rational Analysis for a Problematic World, Wiley, Chichester.
21. STOWELL, F.A. and STANSFIELD, M. (1991). A First Step Towards The Automation of SSM ?, Systems Thinking in Europe.
22. WILSON, B. (1984). Systems: Concepts Methodologies and Applications, Wiley, Chichester.
23. WEST, D., STANSFIELD, M. and STOWELL, F.A.(1994). Using Computer Based Technology to Support A Subjective Method of Inquiry.