SERUM - Software Engineering Risk: Understanding and Management

D. Greer and D.W. Bustard
Faculty of Informatics, University of Ulster, Cromore Road, Coleraine, BT52 1SA, Northern Ireland. Email: D.Greer@ulst.ac.uk, D.W.Bustard@ulst.ac.uk

ABSTRACT

Software development is often characterised by problems with projects over-running their schedule or by failure to satisfy user requirements for functionality and quality. It follows therefore that risk management should be part of any effective software development process. One approach is to identify risks explicitly at various stages in the software process and then determine how best to respond to them. An alternative strategy is to handle some (perhaps most) of the significant risks covertly by building risk reduction techniques into the development process. This paper argues that the overall risk management process should make use of both forms of risk management, with initial emphasis placed on implicit risk management as that is where the greatest return for effort is likely to be obtained. A particular approach, SERUM, is described which considers risk from initial business analysis up to and beyond software delivery. The approach makes use of two particular techniques: (i) Checkland and Wilson's Soft Systems Methodology (SSM); and (ii) Gilb's Evolutionary Delivery. In each case the implicit risk reduction contribution of the technique is identified and then the further advantage of an explicit risk treatment explored.

1. Introduction

Risk management research seems particularly relevant to computing projects because of the relatively high instance of failure in the industry. It is perhaps surprising, therefore, that in practice few organisations currently use formal risk analysis[1]. Part of the reason may be that software engineers are by nature optimists[2], tending to overestimate the likelihood of success and, by implication, underestimate the likelihood of failure[3]. Also, project managers may tend to play down risk to some extent in an attempt to develop a climate of optimism and enthusiasm that helps drive a project forward.

Another factor inhibiting the uptake of risk management is that often there are so many risks in a software development project that their investigation can become overwhelming. Standard advice is to address only the high impact risks (e.g. the 'top-4') but in practice it can be difficult to decide where to draw the line - indeed it can be uncomfortable to ignore any risk once it has been identified and documented. The situation is made worse by the presence of general risks that are acknowledged to be important but extremely difficult to handle because they affect so much of the development life cycle. One example is the risk of software not meeting client requirements adequately.

This paper presents a technique, SERUM, for handling risk management within a defined software development process. It combines the use of *implicit* risk

management, in which the software development process is designed to reduce risk, with more traditional, *explicit* risk management techniques, in which risks are addressed directly by a process of identification, assessment, prioritisation, planning, resolution and monitoring[5]. SERUM places particular emphasis on implicit risk management. By doing so, a generic risk that is common across all projects, such as late delivery, need not be dealt with explicitly as it is managed implicitly by the defined process. Factoring out the generic risks then reduces the number of risks that need to be handled explicitly, making it easier to ensure that they are handled adequately.

The backbone of SERUM is provided by the integration of two well-established techniques: (i) Checkland and Wilson's Soft Systems Methodology (SSM)[6,7] ; and (ii) Gilb's Evolutionary Delivery[9]. In this paper the implicit risk reduction contribution of each technique is discussed and then the further advantage of an explicit risk treatment considered. In the next two sections, SSM and Evolutionary Delivery are summarised, illustrated, and their risk reduction contributions identified. A following section then demonstrates how these implicit risk reduction techniques are integrated and extended with explicit risk management techniques, defining SERUM, an evolutionary, business-driven, software development strategy.

2. Soft Systems Methodology

Soft Systems Methodology (SSM) has been developed as an engineering approach to management problems[6,7]. It is, however, a general problem solving technique and can be used for any system where a process is involved. The methodology is based on an assumption that while some computing problems are 'hard', i.e. they are well defined with exact solutions, more commonly they are 'soft', suffering from a lack of agreement about the nature of the problem, its cause, or its solution.

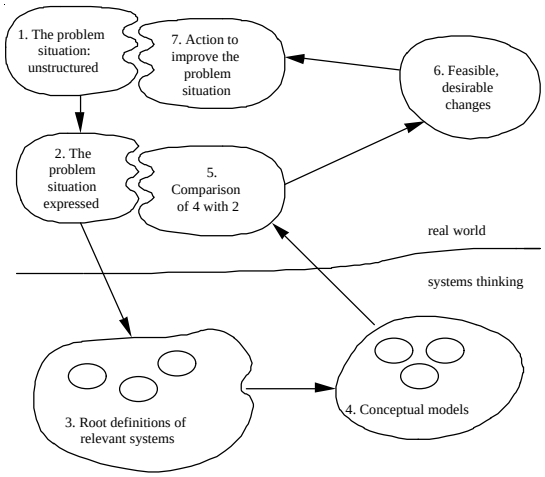

Figure 1: Soft Systems Methodology Summary Diagram

SSM supports a goal driven approach to computing systems analysis and ensures that the investigation starts with a business analysis. This is achieved by first identifying the intended purpose or purposes of the system in which the computing facilities are to be used. Each such purpose is developed into a behavioural model of the activities necessary to achieve that purpose. Classically, the phases of SSM are summarised by the diagram shown in Figure 1. This divides analysis into *Real World* activities, dealing with tangible objects, and *Systems Thinking* activities, which involve the building and

are subsequently used for comparison with the real world to identify desirable changes.

To illustrate the use of SSM, consider the case of a manufacturing division of a company, PrecisionParts. The division manufactures aircraft parts for the parent company and for outside customers, on a competitive basis. The analysis has been started by a request for an investigation into how the division might achieve a more effective organisational structure making full use of information systems.

The first step of the analysis is to generate *root definitions* for the system under investigation. For example, one root definition for the manufacturing division might be: *A manufacturing division owned system which responds to PrecisionParts for the effective manufacture of a range of aircraft parts and which seeks to be a major supplier for both PrecisionParts and the market, within the performance constraints defined by PrecisionParts.*

A root definition identifies a central goal or purpose of a system and the transformation it performs (seeks to be a major supplier). It may also, optionally, include the customer for the transformation (PrecisionParts), the actors of the transformation (not stated here), the owner of the system (the manufacturing division) and any environmental constraints (within the performance constraints defined by PrecisionParts). A *conceptual model* can be constructed (Figure 2), identifying the activities stated or implied in the root definition, and indicating dependencies among them.

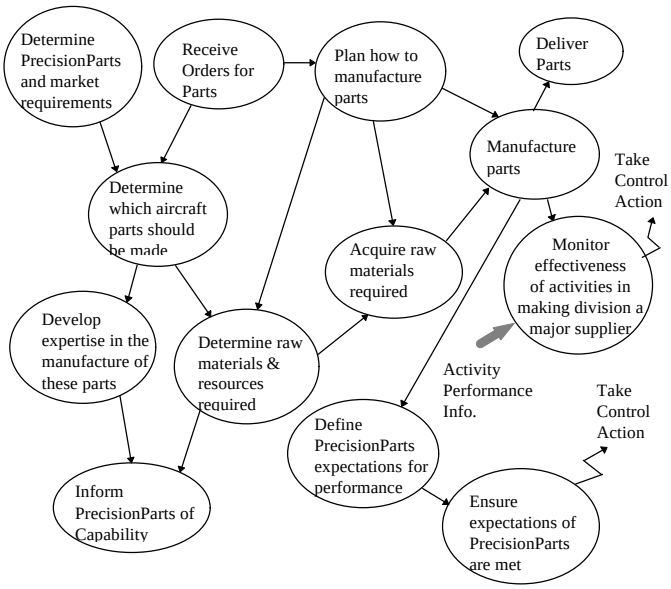

Figure 2: Sample Soft Systems Activity Model for a Manufacturing Division.

Systems typically have several root definitions. For example, from the parent company's point of view, the purpose of the manufacturing division is to make parts for it as inexpensively as possible. This perspective then requires another root definition to be produced, together with a matching conceptual model. Conceptual models are later merged to form a single system model, introducing activities to resolve conflicts where they occur.

In phase 5 (Figure 1) a comparison is made between the system models and the real world to test the adequacy of the models and identify real world improvements. One systematic approach is to build a table (Table 1), examining each activity in turn.

Table 1: Sample Activity Table

(1) Activity	**(2) Exists ?**	**(3) Current Mechanism**	**(4) Measure of Performance**	**(5) Proposed Change**	**(6) Comments**
Determine which aircraft parts should be made	Yes	Decision on which parts to make is based on manager experience	% orders filled	Build costing system and inventory of tools to aid decision	May be possible to implement expert system
Determine raw materials & resources required	Yes	Assumption that raw materials can be ordered and resources are available	None	Bill of Materials is drawn up for every part ordered and resources and materials identified	In the past, delays have often resulted while materials were ordered
etc.					

The first column of the table identifies the activities concerned, the second indicates whether or not that activity is currently performed; if so, then the third column describes the mechanism involved. The fourth column defines how the performance of the activity is measured; the fifth column summarises any proposed changes and the final sixth column can be used to make clarifying comments or to add notes.

2.1 Implicit Risk Reduction in SSM

In terms of risk reduction, the SSM approach offers a number of distinct advantages over other systems analysis techniques:

1. *Risk: Undue reliance placed on a computing system to bring about improvement*
   There is a tendency in traditional systems analysis to assume at the outset that computing facilities are needed and that they will largely be responsible for bringing about business improvement. SSM instead focuses on business needs, without any initial assumptions about how those needs will be met. In this way the business can be improved in whatever way is considered appropriate and may not involve computing development at all.
2. *Risk: The system has not been fully defined.*
   One of the key requirements for effective risk management is that there is a clear, unambiguous understanding of all of the relevant key aspects of a project[8]. SSM in starting with an investigation into the problem situation, and in obtaining a comprehensive root definition of the system, ensures that the required understanding is developed, including an awareness of any environmental constraints.
3. *Risk: System changes do not achieve potential benefit*
   Another problem with traditional analysis techniques is that they tend to focus on improving the current situation and so miss opportunities for innovation. SSM also starts with an initial investigation of the current situation but only to help identify its purpose. The first models developed (root definitions and conceptual models)

should be carried out rather than those that are currently performed. This approach eliminates a common fault of merely implementing a computerised version of the current system.

4. *Risk: System changes fail to satisfy those in the problem situation*
   The SSM approach, because it considers multiple perspectives on the purpose of the system, tends to take more viewpoints into account and so reduce the risk of ignoring the needs of any group or individual in the system. The multiple viewpoint approach also helps to expose differing views and encourage the resolution of conflict (detected when separate models are merged).
5. *Risk: System changes do not have the effect predicted or the system deteriorates over time.*
   It is an established practice in soft systems analysis to include activities that relate to monitoring and control. Monitoring activities measure the effectiveness, efficacy and economy of other activities and take control action as necessary. Hence, these activities reduce the risk of a system change failing to bring about improvement or problems arising later because of changing circumstances.

As well as reducing risks, development techniques can also *introduce* risks so these need to be identified to ensure that there is a net gain. In the case of SSM, the associated risks are relatively small, although a problem that occurs commonly in practice is that the analyst fails to escape from the influence of the current system (ŧeality seepage!) and ends up largely following a traditional analysis approach in an SSM style. SSM analysts need good creative skills.

Another risk concern is that the SSM approach typically identifies a large number of potential system improvements and implementing these at the same time could affect the stability of the system or simply be impractical because of the associated cost. One way to reduce this risk is to use a variation of Gilbŝ evolutionary approach to software delivery[9] in phases 6 and 7 of SSM, where beneficial changes are defined, implemented and evaluated. The effect of Evolutionary Delivery on risk is discussed in the next section.

3. Evolutionary Delivery

The Evolutionary Delivery technique for software production is simple, at least in concept. Instead of implementing and installing a software system in one step, parts are developed and delivered in phases. Each delivery is a complete system that is of value to the client. The delivered system is evaluated by the client and the results fed back to the developers who then take that information into account when implementing subsequent phases.

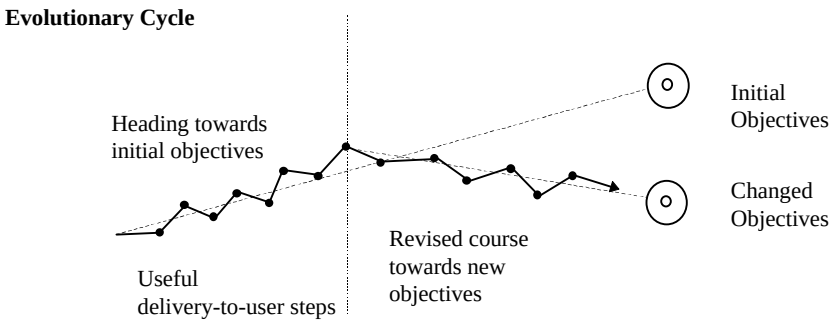

Figure 3: Evolutionary Delivery

The same approach can be taken when making any type of system change. An evolutionary plan is produced initially, based on the objectives for the system being developed. The plan is executed in stages, each with its own objectives, deliverables and system changes. After each stage the plan is reassessed based on an evaluation of the stage delivered. Figure 3 illustrates the evolutionary delivery cycle[9], implying that there can be large or small adjustments to objectives after each delivery.

3.1 Implicit Risk Reduction in Evolutionary Delivery

The Evolutionary Delivery technique reduces risk in a number of ways:

1. *Risk: System change fails to satisfy those in the problem situation*
   Proposed changes are prioritised according to their user-value to development cost ratio[9] so the initial changes at least should provide a good return for the costs involved. Also, if the criteria for delivery are accurately established, the risk of losing user support is reduced; it should also help that users are directly involved in the development process and have a chance to comment at each stage. Finally, as changes are made in relatively small steps, any unsatisfactory changes can be repaired relatively quickly anyway.
2. *Risk: Late delivery*
   Usually there is some flexibility in the changes implemented at each stage so if an overrun seems likely some aspects of the change can be deferred to a later release. Thus late delivery need never occur.
3. *Risk: Budget overruns*
   Budget risk is reduced because the stages are relatively small and consequently easier to analyse and cost.
4. *Risk: Environmental changes invalidate the system changes planned*
   Since staged changes are scheduled relatively frequently, environmental changes can be detected and responded to relatively quickly.

These potential advantages are offset to some extent by a matching number of risks associated with using the technique:

1. *Risk: Overhead costs are too high*
   Each change step is a project carrying with it various overhead costs. The steps need to be reasonably large to ensure that sufficient benefit is gained for the costs expended.
2. *Risk: Funding is difficult to obtain for a succession of changes*
   Because of the way that many organisations manage their budgets it may be much easier to obtain support for one major change than a succession of small changes.

3. Risk: *The change process deteriorates into 'code & fix'*
   Without sufficient controls there is a danger that changes will not be sufficiently well worked out because corrections can be made relatively quickly.
4. *Risk: People tire of frequent changes*
   It is likely that almost any change to a system will cause disruption and therefore lead to some dissatisfaction, regardless of the eventual benefit. A promise of on-going frequent changes may, therefore, be considered unacceptable.

Despite these threats, the Evolutionary Delivery technique is an appealing concept and represents a helpful way of prioritising and planning change even if a monolithic implementation is attempted. When using Evolutionary Delivery its associated potential risks should be examined explicitly as part of the development process, possibly as part of a formal plan review.

In general, having defined a development process to reduce risk inherently, an explicit risk treatment is still necessary in each project to deal with issues directly related to the techniques being used and the particular type of change being proposed. The next section illustrates one use of explicit risk management in connection with the combined application of Soft Systems Methodology and Evolutionary Delivery, summarising and illustrating earlier work in this area[10,11]. In doing so it effectively illustrates a combined use of implicit and explicit risk management.

4. SERUM: Combining Implicit and Explicit Risk Management

When applying SSM with its Evolutionary Delivery extension, the associated risks that need to be considered explicitly are as follows:

1. *Risks in the application of the SSM process*.

The quality of the models depends on the experience of the consultant/analyst. There may be a risk that the situation under investigation, its scope and its boundaries, may not have been properly identified that the models are not validated or accepted by all of the parties concerned. The Evolutionary Delivery risks, listed earlier, should also be examined.

2. *Risks in the system proposed as a result of the SSM study*.

Given that SSM may result in the recommendation of an innovative, 'evolutionary' system, the proposals need to be carefully examined for operating risks.

3. *Risks in the implementation of the system resulting from the SSM study*.
   These are basically risks of acceptance, risks to budget or schedule, or technical risks where the system uses new technology, or uses technology in novel ways.

These risks can be handled as part of the development of an evolutionary delivery plan. The approach is based on a combined cost-benefit and risk analysis of the current system, the proposed system and the change process. It is summarised in Figure 4.

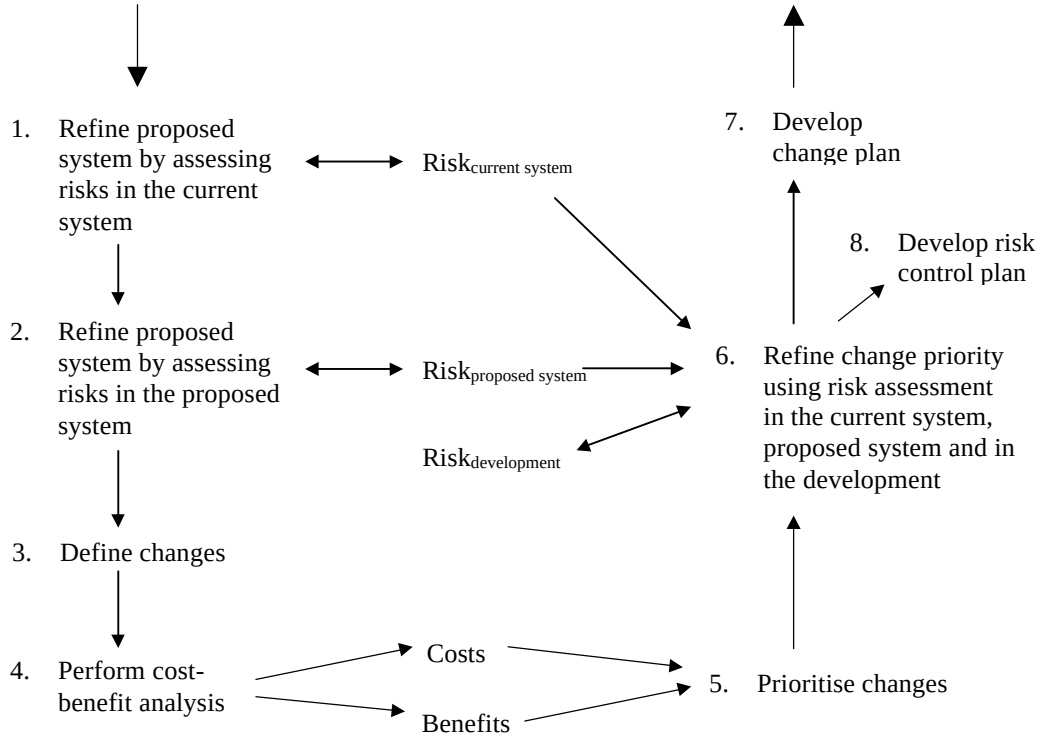

Figure 4: Overview of SERUM

This is a particular way of implementing step six of SSM as shown in Figure 1. Its input is a proposed new system and its output is a change plan, with associated risk control plan.

The basis of the SERUM method is that the order in which a proposed system should be delivered may, initially, be considered as a function of five variables.

$$\textit{Implementation Priority} = f(\ \textit{costs, benefits, risk}_{\textit{current system}}\ ,\ \textit{risk}_{\textit{proposed system}}\ ,\ \textit{risk}_{\textit{development}}\ )$$

The derived priority values can then be used to rank the changes, taking account of inter-dependencies among them.

The input to the SERUM method is a model of the proposed system. This will identify activities required in the system (not necessarily those in existence) along with a set of recommendations for change summarised in tabular form as illustrated earlier in Table 1. (The description also, in effect, defines the current system.) SERUM involves building up risk information in columns added to the activity table. To save space in the discussion that follows only the relevant portion of the extended table is shown in each case, with column numbering indicating what has been selected.

Step 1: Refine the proposed system by assessing risks in the current system.
Risks are identified explicitly for each activity in the current system by considering how each could produce an unfavourable outcome or adversely affect another activity. In addition, established risk identification techniques, such as questionnaires[12] may be used to ensure that all risks have been identified. Columns (7) to (9) of Table 2

illustrates how the risks in the current system are documented.

Table 2: Identification of risks in the current system

(1) Activity	(3) Current Mechanism	(7) $Risk_{current}$	(8) Consequence	(9) Risk Assessment $_{current}$
Determine which aircraft parts should be made	Decision on which parts to make is based on manager experience	Management make a mis-judgment and are overoptimistic, assuming a part can be made but in fact cannot.	Cost of sub-contracting, loss of reputation, delay	Medium
		Part is incorrectly specified.	Cost of rework	Low
etc.				

The method of assessing the risks is open to the analyst. Either a risk exposure calculation[13] may be performed or risks may be classified as in the SEI's SRE method[14] where risks are assessed by a magnitude (high, medium or low) derived from a description of the severity or consequence of the risk and its probability. The latter approach has been used in this example, but the method is easily adjusted to accommodate a richer assessment scheme.

Having identified and assessed risks in the current system, the original changes proposed may now have to be refined and indeed, other risks identified for those refinements. For example, for the activity "Determine raw materials and resources required," there is a risk of assuming that the resources are available to complete a job. This may lead to a refinement of the SSM model to include an activity: "Check availability of resources." Hence, the process is iterative and ensures that the models are adequate.

Step 2: Refine the proposed system by assessing risks in the proposed system.
For each proposed change the risks are identified and assessed in a similar manner to that for the current system (Table 3).

Table 3: Identification of operational risks in the proposed system

(1) Activity	(5) Proposed Change	(10) Operational $Risk_{proposed}$	(11) Operational Risk Assessment $_{proposed}$
Determine which aircraft parts should be made	Build costing system and inventory of tools to aid decision	Estimating skills are lost	Low
		Costing system data is incorrect	Low
etc.			

A consideration of the risk assessment may now necessitate a refinement to the proposed changes to reduce the risk. For example, the risk of the development team

necessary training. This will have a cost and so must be added to the proposed changes column and the risk assessment adjusted accordingly.

Step 3: Define Changes

The previous two steps have identified changes to the current system. Since these where derived by looking at the activities in isolation, it is possible that some changes have been documented twice, with the same or different descriptions. It is also possible that certain groups of changes are more meaningful and manageable if aggregated. Hence, since the aim is to derive an evolutionary plan, all of the changes are examined and defined into tasks of suitable size (Table 4).

Table 4: Proposed changes

Proposed Changes
1. Build a costing system
2. Establish Inventory of Tools
3. Automate job booking mechanism
4. Etc…

Step 4: Perform cost-benefit analysis on the proposed changes.

Since a categorisation method has been used for the risk assessment, it is appropriate to take a similar approach to cost and benefit. The following matrix (Table 5) has been developed[11] to illustrate how a relative measure of cost-benefit ratio may be obtained. For example, a cost considered "Low" and with a "Medium" benefit is classified as II. A very simple banding of costs and benefits has been used here for illustration purposes but can be set at whatever level of detail is considered desirable.

Table 5: Matrix for obtaining the cost-benefit assessment of a proposed change

Cost	**Benefit**		
	Low	*Medium*	*High*
Low	III	II	I
Medium	IV	III	II
High	V	IV	III

This process is carried out for each proposed change (Table 6).

Table 6: Assessment of cost-benefit for proposed changes

Priority	**Proposed Change**	**Cost-benefit assessment**
1	Build costing system	II
2	Automate job booking mechanism	III
3	Establish inventory of tools	IV
4	etc	

Step 5: Prioritise the proposed changes using the cost-benefit assessment.
From step 5 it is then possible to group the proposed changes into, in this case, five classes. Gilb[9] expresses the view that in planning a project the potential steps with the highest "user-value to development-cost ratio" should be selected for earliest implementation. Although user-value may be something other than a high return of investment, a first step towards this would be to implement those with the lowest cost-benefit ratio first.

Step 6: Refine the change priority using the risk assessment for the current system, the proposed system and for the development process.
Specific development risks must also be identified and assessed. Again, established risk identification techniques such as the SEI's taxonomy approach[15] may be used. Table 7 shows a small part of the risk assessment for the PrecisionParts manufacturing division.

Table 7: Identification of development risks

Priority	**Proposed Change**	**Development Risk$_{proposed}$**	**Development Risk Assessment$_{proposed}$**
1	Build costing system	Inexperience in this type of application may lead to slippage	Medium
2	Automate Job Booking Mechanism	Non-cooperation of relevant personnel	Low
3	Establish inventory of tools	Tools may prove very difficult to classify	Medium
	etc		

Having prioritised the proposed changes according to cost-benefit ratio, the analyst may then review the priority taking account of the risk assessment information that has been accumulated. If the current system were found to have a "High" risk exposure associated with one of its activities, then it would be preferable to implement the change that reduces this first before the associated problem occurs. Similarly, if a change is given a high priority in step 6, but has a high risk associated with it, it may be advisable to delay that change somewhat.

The tediousness of this review process may be reduced by using an automated tool that allows the analyst to examine and manipulate the relevant columns of the table easily.

Step 7: Develop Change Plan.
This step provides a link with the project management activities required for the implementation phase, and includes the production of a development schedule.

Step 8: Produce risk control plan for accepted risks.
The process up to now will have produced a prioritised list of system changes which hopefully will reduce the risks in the current system. Inevitably, some changes will have been rejected leaving risks in the current system unanswered. Other risks will have been introduced in the new system and there will be risks associated with the development of the system. Risk control consists of monitoring these risks and preparing contingency plans. Thus for each risk not resolved, a risk control plan with a

means of monitoring and taking possible corrective action is produced.

Conclusion

This paper has introduced SERUM (Software Engineering Risk: Understanding and Management), a risk management method which has implicit and explicit risk management elements. The underlying notion is that any systems development process should be designed in a way that reduces risk inherently to reduce the amount of explicit risk management required. SERUM has been defined as an extension of Checkland and Wilsonś Soft Systems Methodology approach to identifying desirable system changes. This has been coupled with Gilbś evolutionary development method as a way of implicitly reducing risks in implementing changes. SERUM also defines explicit risk management extensions to the basic process to help prioritise and plan changes. Tool support for the method is essential and is currently under development. Further work will include more a explicit treatment of the identification of risks and of risk responses. In the meantime, however, parts of the technique are being refined through their trial application in the Northern Ireland Civil Service, where SSM is a standard business analysis technique and where explicit risk management is performed in many projects.

ACKNOWLEDGEMENT

This paper has been developed from a presentation at the 5th Software Engineering Institute Conference on *Software Risk: Managing Uncertainty in a Changing World*, in April 1997. The work is also contributing to an EPSRC funded project, RIPPLE, GR/L60906.

REFERENCES

1. Griffiths, C. And Willcocks, L., *Are Major Information Technology Projects Worth the Risk?*, Oxford Institute of Information Management/IC-PARC, Imperial College, 1994.
2. Brooks, F. *The Mythical Man-Month*, Addison-Wesley, MA, 1975.
3. Sage, A, *Behavioural and Organization Considerations in the Design of Information Systems and Processes for Planning and Decision Support*, IEEE

   September/October.
4. Boehm, B.W., *A Spiral model of Software Development and Enhancement*,

5. *Software Risk Management*, IEEE Computer Society Press, 1989.
   Checkland P.B. & Scholes, J: , Wiley, 1990.
7. Wilson B: *Systems: Concepts, Methodologies and Applications*
   Wiley, 1990.
8. Chapman, C., Project Risk Analysis and Management, International Journal of Project Management, 1997, vol. 15, no. 5, pp 273-281.
   Gilb, T., , Addison-Wesley, 1988.
10. Bustard, D.W., Greer, D. and Tate, G., *Enhancing Soft Systems Methodology with* , in Software Quality Management II, 1994, vol. 2, Ch. 53, pp. 145-147, Elsevier, Southampton.
    Greer, D. And Bustard, D.W.
    on Soft Systems and Risk Analysis, Proceedings of the IEEE Symposium and

Workshop on Engineering of Computer Based Systems, 1996, pp. 126-133.
12. CCTA (Central Communications and telecommunications Agency - UK Civil Service), *Introduction to the Management of Risk*, HMSO Books, London, 1993.
13. Boehm, B.W., *Software Risk Management: Principles and Practices*, IEEE Software, January 1991.
14. Sisti, F. and Joseph, S., *Software Risk Evaluation method v 1.0*, Software Engineering Institute Technical Report, CMU/SEI-94-TR-19, Dec. 1994.
15. Carr, M.L., Konda, S.L., Monarch, I., Ulrich, F.C, and Walker, C.F., *Taxonomy-Based Risk Identification*, Technical Report CMU/SEI-93-TR-6, Software Engineering Institute, Pittsburg, PA 1993.