Towards an Evolutionary Software Delivery Strategy based on Soft Systems and Risk Analysis

D. Greer and D.W. Bustard
University of Ulster
Coleraine, Northern Ireland, UK

Abstract

RACE is a requirements engineering method which is currently under development. This paper describes broadly the techniques developed so far, reviews earlier work on how risk analysis might be incorporated in RACE and proposes an extension of the method to include evolutionary delivery of proposed changes derived from the method.

Proposed changes arising from RACE are often software related and tend to be radical and so are by nature high risk. Hence, these changes are well suited to evolutionary delivery. A means of deriving an evolutionary delivery plan based on cost-benefit analysis and on a risk assessment of the current system, the proposed system and the development of the proposed system is described.

Introduction

> "Whoever digs a pit may fall into it, Whoever breaks through a wall may find a snake"
>
> King Solomon - Book of Ecclesiastes

Software engineering is a high risk discipline. This fact can be confirmed by even a cursory examination of the popular computing magazines, which regularly report examples of severe losses incurred during computer system development or because systems fail to meet customer requirements. It is very difficult to determine in advance exactly what a computer system should provide and so it is difficult to estimate the costs involved and plan a project accordingly. The problem is made worse because software developers are by nature 'optimists' [1] and tend to ignore potential problems that are highly likely to occur, such as software containing errors that will take a substantial time to remove. Further, those commissioning software often have inflated expectations of what can be achieved and may assume that a computer system will solve all their problems whereas in fact it will often create other difficulties. It follows, therefore, that explicit risk management *should* be a mandatory, prominent and integral element of any software development process. In practice, however, this is not the case and risk tends to be considered adequately only in security or safety-critical applications.

The purpose of this paper is to introduce and explain a software development model that addresses risk adequately. The model is part of the RACE [2] requirements engineering method being developed at the University of Ulster.

The net effect of this approach is to define an "ideal" way of running the business that is often quite different from its present operation. Bridging the gap in one step is usually impractical, partly because of the costs involved but more importantly because of the high associated risks. It is assumed, therefore, that evolutionary development is necessary [3]. This paper proposes an approach to evolutionary delivery that is based on desirable changes identified through the RACE method and prioritised using a combination of cost-benefit and risk analysis. The first section of the paper discusses the need for risk management in more detail. The second section describes the relevant parts of the RACE method and summarises earlier work [4] on risk analysis as a refinement technique for assessing and improving RACE system models. The final section presents an evolutionary development method that extends the current RACE work and discusses its strengths and limitations.

The need for risk management

Risk basics

Risk is essentially the possibility of failure. This involves two elements: *loss* and *chance*. *Risk Exposure* can be introduced as the product of the probability of failure and the loss associated with that failure [5]:

Risk Exposure = probability(risk) * loss(risk)

Since risk is a function of probability and of loss incurred, and since both of these may be estimated, the temptation is to treat risk entirely in mathematical terms. In practice, however, it is impossible to obtain a precise numeric value for something as intangible as the likelihood of an event occurring in the future. Fortunately, exact figures for probability are not needed since they are mostly used for comparison purposes. In fact being too exact carries the risk of placing too much reliance on a figure whose accuracy cannot be justified.

The loss associated with a failure must also be estimated. This is probably even more difficult as losses differ greatly in nature. Losses may have units as diverse as dollars, days, failures per month, defects - even lives lost. In other cases the units are unclear such as loss of reputation, loss of data or loss of employees to other companies. One obvious solution is to reduce everything to a common unit such as time or money.

Several authors take a more qualitative approach and attempt to classify risks into categories [6], [7]. For example, the European Riskman methodology classifies Risk Exposure as being "Unacceptable", "Critical", "Significant" or "Minor". These classes are obtained by considering both the loss due to failure and the probability of the failure occurring as being "Low", Medium" or "High". A matrix (Table 1) is used to obtain the classification. For example, a "High" loss risk with "High" probability of occurrence gives an "Unacceptable" risk exposure while a "Low" probability coupled with a "High" loss gives a "Critical" risk exposure.

Table 1: Matrix for obtaining the class of a risk

Loss	**Probability**		
	Low	*Medium*	*High*
Low	Minor	Significant	Critical
Medium	Significant	Critical	Unacceptable
High	Critical	Unacceptable	Unacceptable

Having identified and assessed risks, the next step is to decide what to do about them. Boehm [5] lists *avoidance*, *transfer* and *acceptance* as valid approaches to handling identified risks. Avoidance may mean choosing an alternative route which does not involve the identified risk. Transferring the risk means to let someone else deal with it, e.g. by subcontracting a task. Acceptance of a risk necessitates the monitoring of the identified risk and the creation of contingency plans should a failure occur.

What is being done about risk management?

The principles of risk management are covered in almost all popular software engineering textbooks and many books on software project management include chapters on the subject. Complete textbooks (e.g. [5-8], have been devoted to risk management and it has been the subject of many research papers in computing journals and at computing conferences (e.g. [9]) . There can be no doubt that "academically" the treatment of risk management is becoming increasingly more visible.

Further, some large organisations have brought risk management to a prominent position. In the US, for example, the Software Engineering Institute (SEI) has devoted a programme to Risk Management [10], in Europe, the Riskman methodology has been developed [7] and in the UK the government have a risk management methodology, CRAMM [11].

However, despite the apparent high profile of risk management, industry seems to have been slow in its adoption of risk management as an integral part of software development. This is despite risk identification being a level 2 activity in the SEI's Capability Maturity Model and risk assessment and control being at level 3 [12]. Other process assessment models, including those based on the SEI model and ISO9000, feature risk management techniques prominently (e.g. [13]). Given the increasing moves by government organisations to give preference to contractors who have a given level of process maturity, risk management must be seen as more and more important.

In researching for this paper, representatives from 20 companies, each developing software as an integral part of their business, were interviewed. With regard to risk identification, almost all claimed to do it informally, based on the experience of project managers. Typically, risks associated with software projects were identified before commitment to a project and these remained on the agenda for the lifetime of the project. One respondent concentrated on areas of the project that were 'new' and identified risks associated with these areas as being the ones to be monitored. Only a very small percentage actually assessed risk using a risk exposure calculation or even a qualitative ranking exercise.

When asked to give an opinion on whether risk identification is or would be useful in planning a software project, all the respondents strongly agreed. Almost all supported the idea of calculating risk exposure or at least trying to assess the risks in some manner. Reasons given for not carrying out risk identification and assessment where that it was too time consuming or that they were simply not familiar enough with the techniques. Interestingly, some considered that risk assessment

calculations were unhelpful in that reliance on them might cause an additional risk (rather like the general opinion in 1912 that "The Titanic" was unsinkable).

On risk reduction, all those interviewed had measures in place which they felt reduced software risk. Sub-contracting, quality systems including adherence to ISO9000, prototyping techniques, thorough testing, design proving, use of metrics and contingency planning were included in actions that were taken to reduce risk.

All of those interviewed expressed an interest in risk management and almost all stated that they thought their organisation should be doing more about risk management.

The case for explicit risk management?

Evidently, almost everyone concerned is in agreement that Risk Management is important and that it should be an integral part of the software development process. So why are risk management techniques not widely adopted in practice? After all, cost-benefit analysis is well established in assessing the feasibility of a computing system. Why then is risk, which after all is the possibility of an increase in cost or decrease in benefit, not taken more seriously?

One conclusion that is reached from this exercise is that while software risk is present in the development process *and* in the software product, most documented approaches to risk management concentrate on one of these aspects to the exclusion of the other. However, risk is present before a software project begins, during its development and during the lifetime of the end-product software. Hence, there is a need for an approach that addresses risk at all stages and uses it as a basis for planning software delivery. In particular, current approaches to software risk management, do not explicitly state a need to carry out risk analysis in the current system before any software development is carried out. It is suggested here that this is where risk analysis needs to start and a method is described that initiates risk analysis at the business analysis stage and carries this through to the implementation stage and beyond.

RACE

RACE (Requirements Acquisition and Controlled Evolution) is a requirements engineering method currently under development at the University of Ulster. RACE makes use of Checkland and Wilson's Soft Systems Methodology (SSM) [14], [15] as a basis for a business analysis before any computing analysis is carried out. The aim of this approach is to determine how a business should operate, to compare this with how it currently operates and from this to make recommendations as to where computing support may be beneficial. Soft Systems Methodology as used in RACE comprises four stages of business analysis.(*Fig* 1).

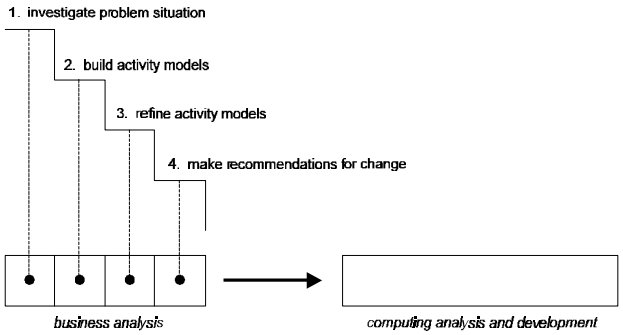

Figure 1. Four Stages of Business Analysis in RACE

The first stage is to carry out a broad investigation into the problem area. The purpose of this is to allow the analyst to gain knowledge about the nature of the business, the constraints under which it operates and some of the problems that exist. For example, in a much simplified version of a university's student accommodation service, a university assigns students to university-managed accommodation. An initial investigation would reveal broadly defined activities, the structure of the organisation and might also show up some of the problems such as legal restrictions, inadequate records, uncollected debt from tenants and so on.

The second stage is concerned with the definition of one or more distinct purposes of the system being analysed, each expressed as a *root definition*. A root definition states this purpose along with the *transformation* taking place (students without assigned accommodation become students with assigned accommodation), the *owner* of the system (the university), and any *environmental constraints* (such as the need to keep within budget). The root definition may also contain the *customer* (the students) and the *actors* in the system (accommodation staff). In the university accommodations system, the purpose could at one extreme be solely to make profit for the university while another might be to provide an environment in which students can socialise. Wilson [15] deals with this by allowing different models and at a later stage merging them into a *consensus* model. For the purpose of this example, we will simplify things by having just one root definition and removing most of the environmental constraints that exist in the real system. the following is a

simplified root definition for the university accommodation system.

A University owned system to accommodate students by assigning them to university accommodation while remaining within budget.

Each root definition is then expanded into a conceptual model (See Appendix, *Fig 2*). This defines the activities that are necessary to fulfil everything in the root definition and defines the relationships between these activities.

The third stage of RACE is to refine the activity models. At this stage, the activities derived are analysed to determine precisely what interaction is taking place, what processes are linked to it and what data is input/output from the activity. This results in so-*called interaction models*. A small part of the university accommodation system is shown in Table 2. Such a description is produced for each activity in the model and may subsequently be used to obtain other models such as dataflow diagrams [16] .

Table 2 - Sample Interaction Table

Process			
A1	Obtain List of eligible students: Obtain List of eligible students from the academic registrar system (I1) and pass on list for processing (I2)		
	Interaction	***Linked Process***	***Data/ Material***
	I1 Obtain list of eligible students	E1 academic registrar system	D1 Eligible students
	I2 Pass on list	A2 Inform students of accommodation service	D2 Rules for eligibility

The activities in the interaction table are labelled to aid their identification (A1, A2, etc.). Each activity is described briefly in terms of its purpose, the sequence of interactions that occur to achieve that purpose, the data involved and the processes with which it interacts. References are used, again, to aid identification (e.g. I1, I2, etc. for interactions, D1, D2, etc. for data items).

The fourth stage of RACE is that of integrating the separate interaction models into one merged model that represents all the required behaviour in the system. It is this merged model that is then compared with the current business processes and used as a basis for recommending change. For each activity in the model its current mechanism is examined in collaboration with the client using a pre-defined measure of performance. Wilson [15], describes a table similar to that in table 3 for this purpose.

The activities in the merged interaction model are compared with those in the current system and based on this analysis, changes are proposed.

Risk in RACE models

The identification of risk in RACE models was the subject of an earlier paper [4]. In summary, risk may be identified:

- At the root definition stage as a means for ensuring its completeness with regard to environmental constraints,
- At the conceptual model stage, after each model has been formed as a means of ensuring its completeness, and
- In the merged model, to ensure that the model deals adequately with the separate risks and to examine risks resulting from interactions and conflict among the models.

The first two of these are largely related to ensuring an accurate model. We will later show how the identification of risk in the merged model can be used to establish the risks in the current and proposed systems

Table 3: Sample Activity Table

(1) Activity	**(2) Exists?**	**(3) Current Mechanism**	**(4) Measure of Performance**	**(5) Proposed Change**	**(6) Comments**
A1: Obtain List of Eligible students	Yes	Printout received from academic registrar system at a given date in the summer	Statistics on rooms rented	Build interface between accommodation system and academic registrar system.	Printout sometimes arrives late and has been know to be inaccurate

and how these along with a cost-benefit analysis and a risk analysis of the development process can be used in deciding the order of an evolutionary delivery of system changes.

Evolutionary Delivery

Normally, the order of evolutionary delivery is effectively established by cost-benefit analysis. This paper expresses the view that such analysis should be further refined to include an assessment of risks: in the current system, in the proposed system and in the development of the proposed system. These expose, in effect, hidden costs (risk increase) or hidden benefits (risk reduction). Thus, priority of implementation may be considered a function of five variables.

Implementation Priority
= *f* (costs, benefits,
$risk_{current\ system}$, $risk_{proposed\ system}$,, $risk_{development}$)

This method for planning evolutionary delivery provides a basis for determining the priority of proposed changes. Note that, in practice this is used to rank the changes and that dependencies between proposed changes also need to be taken into consideration.

The proposed method starts with the recommended changes from the RACE analysis and uses the merged interaction model that has been developed. The process of arriving at an evolutionary delivery plan can be described in terms of the following steps.

Step 1: Identify the risks for each activity in the merged interaction model derived from RACE stage 4.

The output from stage 4 of the RACE analysis is a merged interaction model that represents the required system.

Risks are then identified using standard techniques (e.g. [5], [6], [11]) may be used. In addition, each interaction in the model is examined and the conditions necessary for the interaction to fail are identified as risks.

These risks must be individually specified for the current and proposed systems. Thus at the end of this step, we have for each activity, a list of risks and their consequences (Table 4).

Table 4: Identification of risks from Interaction tables

(1) Activity	(7) Risk	(8) Consequence
A1: Obtain List of eligible students	List arrives too late	Loss of income due to no-rental of rooms
	List is wrong	Loss of income due to cost of sending out information in vain and due to non-rental of rooms

This risk identification may itself reveal weaknesses in the models that have been produced and show up the need for a separate activity to be added to deal with that risk. For example, for the activity "A9: Obtain payment for accommodation", there is a risk of unpaid accounts going unnoticed. This may lead to a refinement to the SSM model and in turn the interaction model extended to include an activity: "Check that tenants make due payment". Hence this process is iterative and ensures that the RACE models are adequate.

Step 2: Identify and assess risks in the current system.

The risks in the current system can easily be derived by considering the risks identified in the required system model in the previous step. Column (9) of Table 5 illustrates how the risks in the current system are documented. Having obtained the required system model and compared it with the current system there will be activities that are present in the current system but not in the required system. Risks must be specified for these also.

Table 5: Identification of risks in the current system

(1) Activity	(3) Current Mechanism	(9) $Risk_{current}$	(10) Risk Assessment current
A1: Obtain List of eligible students	Printout received from academic registrar system at a given date in the summer	Printout arrives too late	Minor
		Printout is wrong - operator error	Significant

The method of assessing the risks is open to the analyst. Either a risk exposure calculation may be carried out or the method of classifying risk into categories could be used. Both of these approaches have been described above.

We have used the latter approach. For example, the

risk of a printout being wrong due to operator fault may rate a "Low" probability but have a "Medium" loss and is therefore judged "Significant" (See Table 1).

Step 3: Refine proposed changes to current system to reduce risks in the current system.

Most descriptions of the application of SSM concentrate on problems with the present system. A problem could be defined as a risk that has already occurred. Charette [17] describes problems as "yesterday's risks" while current risks are potential "future problems". Hence, the investigation from which proposed changes were derived may be incomplete in that it may ignore important risks. It may be argued that these should be considered at stage 3 of the RACE analysis. However, since stage 3 is carried out in collaboration with the user and other interested parties, this may be an unnecessary complication. As a compromise, we suggest that proposed changes are refined in the light of the risk assessment of the current system.

Step 4: Identify and assess 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 6). Again, the risk is classified rather than calculated. In the example, the risk of the interface system failing due to a break in the link between the accommodations system and academic registrar is judged "Significant" (see Table 1). Table 6 shows a sample of the risk assessment from the proposed changes in the accommodation system example.

Table 6: Identification of operational risks in the proposed system

(1) Activity	**(5) Proposed Change**	**(11) Operational Risk$_{proposed}$**	**(12) Operational Risk Assessment$_{proposed}$**
A1: Obtain List of eligible students	Build interface between accommod-ation system and academic registrar system.	Link to academic registrar system is lost	Significant
		Interface misses some eligible students	Minor

Development risks must also be assessed. This can be achieved using well established checklists/ questionnaires [5], [18] . Table 7 shows a small part of this risk assessment for the university accommodations system.

Again, risks are assessed by using categories. For example, the risk of the team not being familiar enough with academic registrar system resulting in overtime costs is rated "Significant".

Table 7: Identification of development risks

(1) Activity	**(5) Proposed Change**	**(13) Development Risk$_{proposed}$**	**(14) Development Risk Assessment$_{proposed}$**
A1: Obtain List of eligible students	Build interface between accommod-ation system and academic registrar system.	- overtime costs	Significant
		Interface misses some eligible students	Minor

Step 5: Refine proposed changes to reduce risk in the proposed system.

The risk assessment of the previous step may now lead to a refinement of the proposed changes. This refinement will be such as to reduce the risk. For example, the risk of the team not being familiar enough with academic registrar system could be reduced by providing the necessary training. This will have a cost and so must be added to the proposed changes column and the risk assessment adjusted accordingly.

Step 6: Carry out cost-benefit analysis on the proposed changes.

Cost-Benefit analysis is a well established technique for determining the feasibility of a system and is well documented in systems analysis books (e.g. [19]). At this stage the recommended changes to the system have not

been fully accepted and a full cost-benefit analysis may only be considered worthwhile for changes that have actually been “signed-off”. Since a categorisation method has been used for the risk assessment, it is be appropriate to carry out a similar exercise in relation to cost and benefit. We have developed the following matrix to classify a proposed change in terms of its cost-benefit assessment Table 8).

Table 8: 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

Cost and benefit are in turn rated as "Low", "Medium" or "High". The assessment is then read off from the table. The meaning of "Low", "Medium" and "High" is left to the analyst/user, but since these assessments will be compared, the important attribute is consistency not accuracy. The assessments themselves are from I to V, although alternative ratings may be used. This exercise is carried out for each proposed change (Table 9).

Table 9: Identification of risks from Interaction tables

(1) **Activity**	**(5)** **Proposed Change**	**(15)** **Cost-benefit assessment**
A1: Obtain List of eligible students	Build interface between accommod-ation system and academic registrar system.	II

Step 7: Prioritise the proposed changes using the cost-benefit assessment.

The cost-benefit assessment will allow the proposed changes to be grouped into, in this example, five classes, which in turn are ranked. Gilb [3] 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. An approximation of this is to implement those with the lowest cost-benefit ratio first. Using our assessment technique those with an assessment of I would be top priority.

Step 8: Refine priority of the proposed changes using risk assessment.

If the current system is found to have an "Unacceptable" risk exposure associated with one of its activities, then it would be preferable to implement the change that reduces this first before the risk is realised. If a change is given a high priority in step 7, but has a high risk associated with it, it may not be advisable to perform it first.

Cost-benefit ratio is clearly not a perfect attribute for determining the correct order of implementation of proposed changes. We suggest that a better means to establish the order of delivery is look at cost-benefit assessments alongside the increase or decrease in operational risk resulting from the change and the development risk in making the change. By looking at the relevant columns of the table, the analyst can adjust the order. This may sound like a painstaking task, but with automation it should be relatively easy. With such a tool it would be possible to rank the changes according to a cost-benefit assessment. Some changes would come out equal and could be manually ranked given more detailed knowledge of costs and benefits. This list could then be adjusted by comparing proposed changes and their respective operation risk reductions or their development risk assessments. The net result would be a prioritised list of changes.

Step 9: Produce risk control plan for accepted risks.

During the analysis, changes will have been proposed and it is hoped that these change reduce risks in the current system. However, some of he changes may not take place for some time or perhaps will be rejected. Further, new 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 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.

Conclusions

We have presented a method for evolutionary delivery of changes to a system which have been derived from a RACE analysis. It has been assumed that changes arising out of a RACE analysis are by nature radical and consequently best implemented using an evolutionary delivery paradigm. Traditionally, cost-benefit calculations are the basis for planning evolutionary delivery. We have put forward the case that this is not sufficient and that rather, the priority of delivery is a function of cost, benefit, risk in the current system, risk in the proposed system and development risk.

Although no easy way of calculating this priority has been stated, the main parameters and their relationship to each other has been described and a guide given on how an evolutionary delivery plan can be derived.

This method remains to be tested in a real-life situation. Further, manually, the method may be quite repetitive but this problem can be removed by the

development of tool support. Together these two tasks constitute the direction of further work.

References

1. Brooks, F. *The Mythical Man-Month*, Addison-Wesley, MA, 1975
2. Bustard, D.W., *Progress Towards RACE, a 'Soft-Centred' Requirements Definition Method*, 1st IFIP/SQI International Conference on Software Quality and Productivity, Hong Kong, pp 29-36, Chapman & Hall, Dec. 1994.
3. Gilb, T., *Principles of Software Engineering Managemen*t, Addison-Wesley, 1988.
4. Bustard, D.W., Greer, D. and Tate, G.: *Enhancing Soft Systems Methodology with Risk Management Techniques*, in Software Quality Management II, vol. 2, Ch. 53, pp. 145-147, Elsevier, Southampton, 1994.
5. Boehm, B.W. , *Software Risk Management*, IEEE Computer Press, Washington D.C., 1989.
6. Down, A., Coleman, M, Absalon,P., *Risk Management for Software Projects*, McGraw-Hill, London, 1994.
7. Carter, B., Hancock, T., Morin, J., Robins, N., *Introducing Riskman Methodology*, NCC Blackwell, 1994.
8. Charette, R., *Software Engineering Risk Analysis and Management*, Mc Graw-Hill, NY,1989.
9. Fairley, R., *Risk Management for Software Projects*, IEEE Software, vol 11, no. 3, pp 57-67, 1994.
10. Sisti,.F.J. and Joseph, S., *Software Risk Evaluation Method version 1.0*, CMU/SEI-94-TR-19, Software Engineering Institute, Dec. 1994.
11. Scarff,F.; Carty, A. and Charette, R., Introduction to the Management of Risk, CCTA Publication, 1993
12. Humphrey, W.S. *Managing the Software Process*, Addison-Wesley, Reading, Mass.,1989
13. Koch, G.R., *Process Assessment: the 'Bootstrap' Approach*, Information and Software Technology, vol. 35, pp. 387-403, 1993.
14. Checkland P.B. & Scholes J: *Soft Systems Methodology in Action*, Wiley, 1990
15. Wilson B: *Systems: Concepts, Methodologies and Applications*, 2nd Edition, Wiley, 1990.
16. Bustard, D.W., Oakes, R. and Heslin, E., *Support for the Integrated Use of Conceptual and Dataflow Models in Requirements Specification*, Colloquium on Requirement for Software Intensive systems, DRA Malvern, May 1993, pp. 37-44.
17. Charette, R., *Essential Risk Management: Notes from the Front*, Proceedings of The second SEI conference on Software Risk, SEI Pittsburg, PA, 1993.
18. 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.
19. Wu, M.S.and Wu, S., Systems Analysis and Design, West, 1994.

Appendix

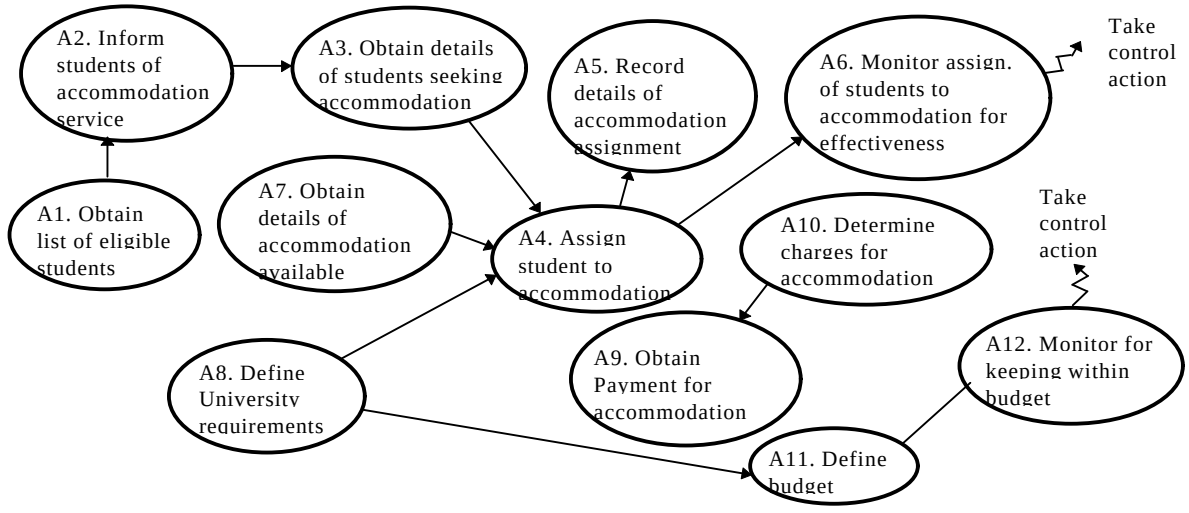

Figure 2. Conceptual model for university accommodation service