AN INTEGRATED, PROBABILISTIC FRAMEWORK FOR REQUIREMENT CHANGE IMPACT ANALYSIS

Simon Lock and Gerald Kotonya
Department of Computing, Lancaster University, Lancaster, UK
Tel: +44 1524 65201 ext 93140 Fax: +44 1524 593608
E-mail: {lock,gerald}@comp.lancs.ac.uk

ABSTRACT

Impact analysis is an essential part of change management. Without adequate analysis it is not possible to confidently determine the extent, complexity and cost of proposed changes to a software system. This diminishes the ability of a developer or maintainer to make informed decisions regarding the inclusion or rejection of proposed changes. The lack of coherent impact analysis can also hinder the process of ensuring that all system components affected by a change are updated. The abstract nature of requirement level entities has meant that current impact analysis techniques have focused largely on design and code level artifacts. This paper proposes a novel approach which integrates traditional impact analysis with experience based techniques to extend current approaches to support requirement level impact analysis. Central to this approach is the use of probability to assist in the combination and presentation of predicted impact propagation paths. An Auto Teller Machine (ATM) example is used to illustrate the approach.

Keywords: Requirement, Change, Impact Analysis, impact propagation probability

INTRODUCTION

Most requirements are defined during the early stages of system development and evolve throughout the system life cycle to reflect the changing needs of the system stakeholders, the customer organisation and operational environment. Changes to single requirements may ripple through a system and impact on other requirements and broader organisational goals. Requirement changes have to be managed and assessed to ensure that they are feasible, make economic sense and contribute to the business needs of the customer organisation. Most development approaches pay insufficient attention to the problems of uncertainty and change (Harker et al 1993). A number of approaches address these problems by freezing requirements at fixed points within the life cycle so that system development can be undertaken in an orderly and controlled manner. This, however, often leads to systems which fail to meet the real business needs of the system procurer. Other approaches seek to identify the minimum set of stable requirements which must be satisfied and which will lead to perceived benefits for the user (Sommerville 1996). It is hoped that by providing incremental delivery, which assists developers in meeting subsequent requirements, it is possible to cope with changes arising from requirements. However, this only prevents certain types of change by focusing on the correctness of the design and implementation of the requirements, rather than supporting changes to the requirements themselves.

The problem of requirement change management is further compounded by size of the information and complex relationships between requirement artifacts. This can make the process of assessing the effect of change expensive, time consuming and error-prone. However, without proper assessment it is impossible for developers and maintainers to fully appreciate the extent and complexity of proposed changes. This makes cost estimation, resource allocation and change feasibility study impractical. In addition to this, a lack of adequate impact analysis can lead to difficulties in ensuring that all affected artifacts are updated for all changes made to a system.

In order to perform impact analysis, extensive lateral traceability information regarding the system must first be obtained (Bennett 1996). Lateral traceability links indicate potential relationships between the artifacts, which make up a system and the system's environment. These relationships can cause changes to be propagated from one entity to another. Current impact analysis techniques (Ajila 1995, Goradia 1993, Li et al 1996, McCrickard et al 1996, Moriconi et al 1990, Han 1997) have focused mainly on design and code level artifacts. The main reason for this is that these artifacts are less abstract than requirements level entities, thus making the identification of lateral traceability links simpler (Bennett 1996).

In this paper we propose a novel new approach which integrates traditional impact analysis techniques with our own experience based approach in order to support change impact analysis at the requirement level. This is supplemented by features to reflect impact certainty and probability values derived during analysis. The use of probabilities in the impact analysis process provides us with useful information regarding the likelihood of occurrence of impact propagations. This allows use to gain a certain degree of contrast between the often large number of predicted propagation paths which are a feature of any realistically sized system.

A prototype tool has been developed to support the impact analysis approach described here. A brief description of the tool is given in a later section of this paper.

Throughout this paper, we will use of the word 'impactable' to denote any previously specified entity, which is affected by a proposed change. For the purposes of this paper the set of impactable entities is limited to the requirements of a system and to any existing or proposed changes to those requirements. By including changes as impactable entities, we can support their evolution in the change management process, as well as predicting the impact of other changes upon them. In addition to this, we can use the experience of previously enacted changes to try to shed light on the impact effect of new changes.

The notion of impactables provides a mechanism for extending the approach beyond the requirements stage to include design level and code level artifacts. This would allow the approach to support impact analysis of development artifacts at all stages of the software lifecycle.

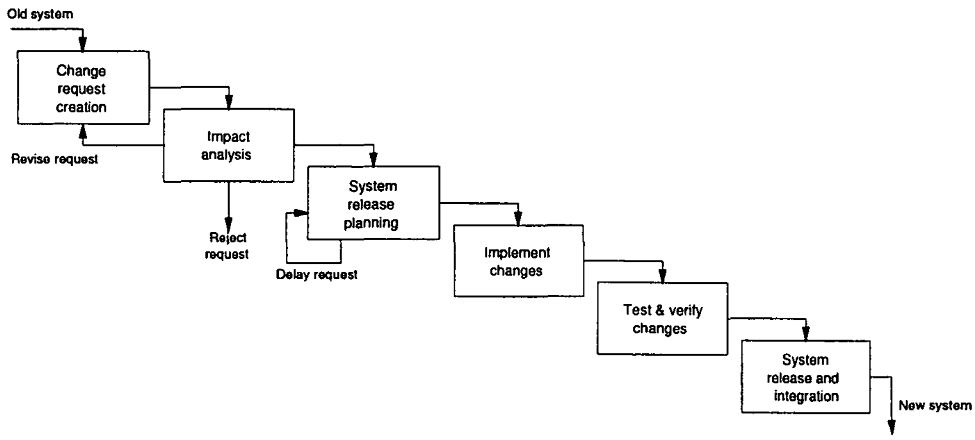


Fig. 1 Change management process

Change management process

The process of impact analysis is dependent on the availability of a mechanism for change management that provides the information required for assessing proposed changes. The main activities of a management process are illustrated in Figure 1 (Moreton 1996).

The work described in this paper is limited to the impact analysis stage of the change management process.

Impact analysis process

Accurate impact determination is crucial for effective impact analysis. Because the information produced is often large and complex, care must be taken to ensure that the information generated is both relevant and adequate. A careful balance must be maintained between producing impact analysis results with too much information and not supplying enough to allow proper appreciation of change impact. The main stages of the impact analysis process are illustrated in Figure 2 (Moreton 1996).

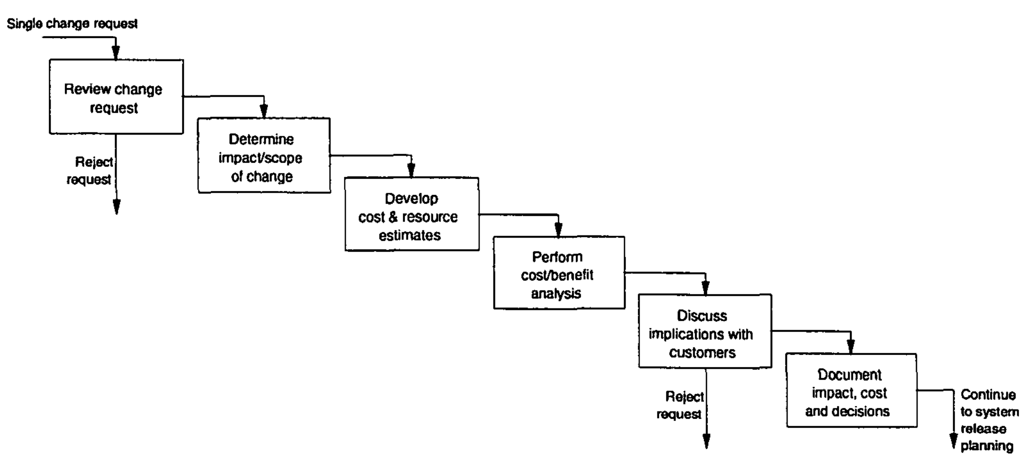


Fig. 2 Impact analysis process

Impact determination

The aim of impact determination is to predict the probable effect of a proposed change on a system. Impact determination uses lateral traceability information to determine impact propagation paths between impactables. These paths are then composed into a single structure, which may then be used as a basis for direct investigation, or for the production of a visualisation of the effect of the proposed change (Lock et al 1998). Figure 3 shows the impact determination process.

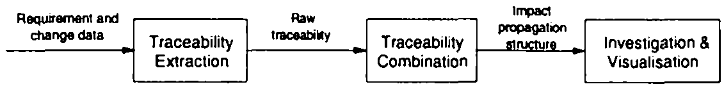


Fig. 3 Impact determination Process

i) *Traceability Extraction* Denotes the extraction of individual traceability relationships from the representation of the proposed system.
ii) *Traceability Analysis* Involves the processing of the raw traceability information extracted previously to produce a representation of the impact of a change on a model of the system.

Example

We will use the example of an automated teller machine (ATM) to illustrate our approach to impact analysis. The ATM contains an embedded software system to drive the machine hardware and to communicate with the bank's customer database. The initial ATM requirements are as shown in Table 1. In the remainder of the paper we will refer to these requirements by their short labels rather than descriptions to conserve space.

Id	Label	Description
1	*Cash*	The system will allow customers to withdraw cash
2	*Balance*	The system will allow customers to display and print their current balance
3	*Databa se*	The system will access a remote database in order to obtain customer PIN and account information
4	*Valid*	A customer will require a valid cash-card and personal identification number (PIN) issued by the bank, to access the ATM services
5	*Printou t*	The system will print a paper record of all transactions
6	*Encrypt*	Bank standard encryption algorithms must be used for all transactions
7	*Risk*	The system security risks must be explicitly identified, analysed and minimised according to the ALARP principle
8	*Avail*	The cash-withdrawal service should have an availability of 99%
9	*Failure*	The system failure rate should be no more than 1 in a year
10	*Simple*	The system should be simple to use for all customers. Customers should spend no more than 10 secs to locate required information

Table 1 ATM requirements

Table 2 shows four possible requirement changes that may be proposed even as the initial ATM requirements are formulated.

Id	Label	Description
1	*Affiliat e*	In addition to its own customers the bank would also like customers from affiliated banks to access the ATM for cash withdrawal
2	*New PIN*	The bank would like its customers to be able to change their PINs interactively on the ATM
3	*Mainta in*	To help ensure fault-free operation, the bank proposes to do complete ATM maintenance once a month
4	*Transfe r*	The bank would like to add a funds transfer facility to the ATM for its customers to transfer funds between accounts.

Table 2 Requirement change list

In this example, that the first three change requests (*Affiliate, New PIN* and *Maintain*) are assumed to have already been enacted and their total resultant impact is known. The last change (*Transfer*) has yet to be enacted and must be assessed for its impact on the system.

Traceability extraction

The first step in impact determination is to extract traceability information from a model of the proposed system. Our approach uses 5 different schemes to extract traceability information at requirements level, namely:

- Pre-recorded traceability analysis
- Dependency analysis
- Plain Experience analysis

- Extrapolation analysis
- Certainty analysis

We will use the ATM example to illustrate each of these techniques, and later demonstrate how we have integrated them into a single framework that provides a more sensitive and informed impact analysis.

Pre-recorded traceability analysis

Pre-recorded traceability analysis is an existing technique which uses information recorded by developers, maintainers and end users to identify potential traceability links between impactables (Bohner et al 1996). Pre-recorded traceability analysis can be done throughout the system life cycle, from initial requirements formulation through to implementation.
This information captures the relationships between the entities that make up the requirement problem space. Each recorded relationship has the potential to cause an impact to propagate from one impactable to another. Thus all relationship types, including all inverse relationships, must be considered when assessing a proposed change. The information used as a basis for performing pre-recorded traceability analysis must be recorded by the requirements engineer when the requirements are being specified. Early traceability information can be extracted from user requirement documents by manually scanning or automatically parsing them for specific keywords representing relationships. The identifiable pre-recorded traceability relationships, which are present in the ATM example, are illustrated in Figure 4.

Transfer	affects	Database	Database	affected by	Transfer	Avail	constrains	Database	Database	const'ed by	Avail
Transfer	affects	Printout	Printout	affected by	Transfer	Failure	constrains	Database	Database	const'ed by	Failure
Transfer	affects	Simple	Simple	affected by	Transfer	Valid	impedes	Simple	Simple	impeded by	Valid
Transfer	affects	Risk	Risk	affected by	Transfer	Avail	supports	Printout	Printout	supported by	Avail
Valid	impedes	Cash	Cash	impeded by	Valid	Failure	supports	Avail	Avail	supported by	Failure
Avail	supports	Cash	Cash	supported by	Avail	Risk	constrains	Cash	Cash	const'ed by	Risk
Simple	constrains	Cash	Cash	const'ed by	Simple	Risk	constrains	Balance	Balance	const'ed by	Risk
Valid	impedes	Balance	Balance	impeded by	Valid	Risk	constrains	Database	Database	const'ed by	Risk
Avail	supports	Balance	Balance	supported by	Avail	Risk	constrains	Valid	Valid	const'ed by	Risk
Simple	constrains	Balance	Balance	const'ed by	Simple	Risk	constrains	Encrypt	Encrypt	const'ed by	Risk

Fig. 4 Pre-recorded traceability relationships

Dependency analysis

Dependency analysis is an existing technique which aims to extract traceability information from behavioural models of the system in order to predict possible impact propagation paths (Bohner et al 1996). The basis for extracting such information is normally a call graph representing dependencies between the system entities. Both forward (depends on) and backward (is depended on by) calling relationships are extracted from the models to ensure that all possible impact propagation paths are identified (Paakki et al 1996). Once this has been done it is possible to determine potential paths by tracing altered functional requirements (impact sources) to impacted functional requirements (impact targets) via the call graph. This is illustrated in Figure 5, where a change (C1) requires an alteration to the source requirement (FR5) which could propagate an impact to the target requirements (FR1,FR2,FR3,FR4,FR8,FR9) via the model calling graph.

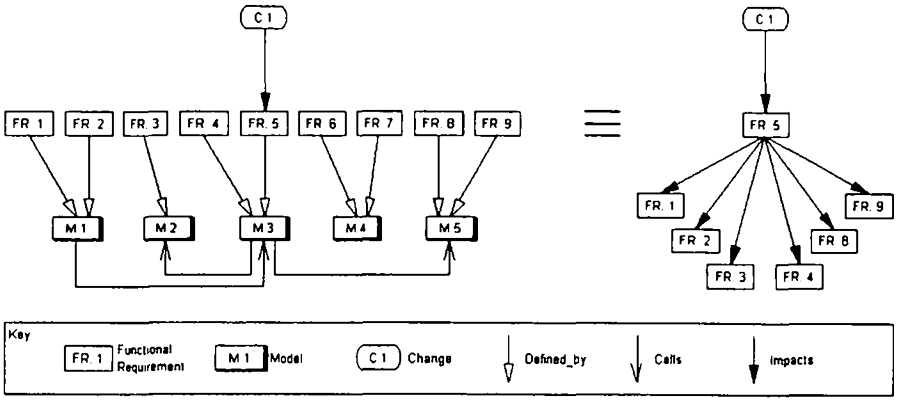


Fig. 5 Identifying propagation paths

The models used to extract call graphs range from structured textual descriptions through use-case scenario diagrams to pseudo code. The extraction of such call graphs may be performed manually, although improvements in efficiency and accuracy can be attained if automated analysis tools are available.
The dependency relationships, which can be extracted from the ATM example are illustrated in Figure 6.

Cash	depends on	Database	Database	dep'ed on by	Cash
Cash	depends on	Valid	Valid	dep'ed on by	Cash
Cash	depends on	Printout	Printout	dep'ed on by	Cash
Cash	depends on	Encrypt	Encrypt	dep'ed on by	Cash
Balance	depends on	Database	Database	dep'ed on by	Balance
Balance	depends on	Valid	Valid	dep'ed on by	Balance
Balance	depends on	Printout	Printout	dep'ed on by	Balance
Balance	depends on	Encrypt	Encrypt	dep'ed on by	Balance
Valid	depends on	Database	Database	dep'ed on by	Valid
Valid	depends on	Encrypt	Encrypt	dep'ed on by	Valid

Fig. 6 Dependency relationships

Affiliate	impacts	Database	Database	impacted by	Affiliate
New PIN	impacts	Database	Database	impacted by	New PIN
Affiliate	impacts	Valid	Valid	impacted by	Affiliate
New PIN	impacts	Valid	Valid	impacted by	New PIN
Affiliate	impacts	Printout	Printout	impacted by	Affiliate
Affiliate	impacts	Encrypt	Encrypt	impacted by	Affiliate
Maintain	impacts	Avail	Avail	impacted by	Maintain
Affiliate	impacts	Simple	Simple	impacted by	Affiliate
New PIN	impacts	Simple	Simple	impacted by	New PIN
Affiliate	impacts	Risk	Risk	impacted by	Affiliate

Fig. 7 Plain experience data

Plain experience analysis

Existing plain experience analysis approaches use a record of the effect of previous changes to the system as a basis for extracting traceability relationships. For a given change, traceability relationships can be assumed to exist between the change and impactables which it has previously effected. Thus each previous change provides us with a cluster of impacts which represent a small segment of the entire system traceability structure. This technique is known as plain experience analysis because previous impact propagation paths are used to construct a path map which is then directly applied to the current change to the potential impact propagation patterns. No attempt is made to identify similarities between the current change and the previously proposed changes and no additional analysis or intelligent filtering takes place. Although simplistic, such an approach is quick and reliable and provides a sound basis for impact analysis.

Figure 7 shows the effect of three currently enacted changes (*Affiliate, New PIN* and *Maintain*) on the ATM example system.

Extrapolation analysis

Extrapolation analysis is a new technique which utilises past experience data to perform analysis. The aim of this type of analysis is to project the small number of direct impacts specified for a proposed change into a complete set of both direct and indirect impacts. Extrapolation analysis is achieved by comparing the direct impacts specified for the proposed change with the actual total impacts of all previous changes. In this way it is possible to identify a number of previously enacted changes which contain the direct impacts of the proposed change as a sub-set of their total direct and indirect impacts.

It is possible to combine the total impact effect of a number of similar, previously enacted changes to develop a composite prediction of the impact effect of the proposed change. This not only maximises the set of projected impacts, but allows us to identify the most common, and therefore most likely, impacts. The direct impacts of a proposed change are compared with the set of all impacts (direct and indirect) of previously enacted changes. The previous changes can then be ranked in order of similarity with the new change. The impact effects of most similar (i.e. highest ranking) previous changes are combined to produce a final set of impacts with which to extrapolate the effect of the proposed change. The number of previous changes used in this combination is configurable, so that the predicted effect could be based on any number of previous change effects. Each of a the given number of most similar previous changes contributes a number of potential impacts. The more previous changes that agree on a particular impact , the higher will be the estimated probability given to that impact. This forms the basis for a scoring system in which the most popular and most likely impacts is given the highest probabilities, and conversely, impacts which are unpopular given lower probabilities.

The proposed change (*Transfer*) best matches the impact effect of *Affiliate*. This is because the direct impacts of *Transfer* are a subset of the total impact of *Affiliate*. We can therefore designate both changes to be of the same class. As a result, we can infer that the total impact of *Transfer* is equal to that of *Affiliate*. This is illustrate in Figure 8.

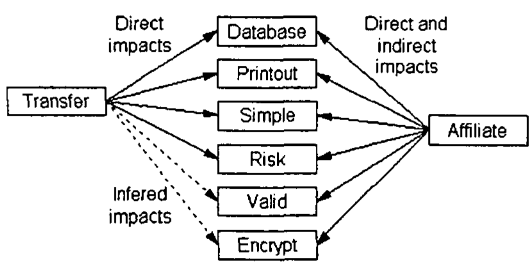


Fig. 8 Extrapolation data

Certainty analysis

We have developed a certainty analysis technique as part of this work to try to minimise the effect of the inherent incompleteness and uncertainty involved in performing impact determination with requirement level artifacts. This should not be confused with the concept of certainty factors as used in the field of expert systems. Our only aim is to try to measure the reliability of the information held about the requirement artifacts which make up a system. Certainty analysis provides us with additional information to offset the deficiencies of the recorded information. To this end, the following additional items of information must be collected for each impactable by the developers specifying the system:

- Degree of definition - the extent to which the impactable has been completely specified (Eckland et al 1996).
- Certainty of definition - the confidence with which the developers believe the entered information is correct

Using these two measures, it is possible to derive a quantitative estimation of the reliability of impactable information and the traceability relationships between them. This additional information makes it possible to gain an appreciation of which impact predictions are most likely to be correct. Conversely, it is also possible to identify those predictions, which are less likely.

The assignment and maintenance of values for 'degree of definition' and 'certainty of definition' relies to some extent on the judgement and experience of individual engineers. However, provided that a coarse scale (e.g. 3 or 4 point) is employed the assessment of the impactables is relatively straightforward. The values that these measures can hold range from 0 for complete uncertainty to 1 for complete certainty. To produce a total certainty value for a particular impact, the two measures are combined using the following formula:

$$C_{total} = DoD * CoD$$

Where C_{total} is the total certainty value, DoD is the degree of definition and CoD is the certainty of definition. We shall investigate how certainty values are used to adjust propagation probabilities in a later section.

The total certainty values for each requirement and change specified for the ATM system are present in Figure 9.

Impactable	Total certainty
Cash	0.7
Balance	0.8
Database	0.6
Risk	0.7
Avail	0.8
Failure	0.8
Simple	0.5

Impactable	Total certainty
Valid	0.7
Printout	0.9
Encrypt	0.8
Affiliate	0.6
New PIN	0.8
Maintain	0.7
Transfer	0.7

Fig. 9 Certainty values

Integrated approach

As mentioned earlier, the abstract nature of the entities makes it difficult for any one of these techniques to extract enough traceability information to allow for viable impact analysis. This lack of adequate information can adversely affect the accuracy of impact determination and result in:

- The failure to distinguish between relationships, which propagate change, and those which do not
- The prediction of impact effects that are larger than would actually be caused by a change. This is due to the difficulty in separating relevant propagation paths from large sets of heterogeneous traceability information.

Our proposed solution to these problems is an integrated framework, which combines a number of individual techniques. This is underpinned by a consistent structure of probability and certainty analysis as a basis for the analysis technique. We believe such a framework provides a basis for better impact analysis by:

- Providing a more sensitive analysis than the single paradigm methods. This results from the fact that although each individual approach may predict spurious propagation, those on which the methods agree will be presented as the most likely impacts.
- Ensuring completeness through improved system analysis coverage.
- Counteracting the abstract nature of requirements through a scheme that integrates the individual strengths of traditional techniques.
- Providing potentially extensible approach that be used to support change impact analysis across the entire system lifecycle.
- Providing a mechanism for distinguishing between potential impacts due to the fact that propagation paths identified by more than one analysis technique are more probable than those identified by a single technique.

Impact propagation structure

Once the raw traceability information has been extracted from the system by one or more of the traceability extraction techniques it is combined into a single propagation structure. Disparate traceability links are combined to form a concrete structure showing all direct and indirect propagation paths for the particular change being analysed. This structure is known as an 'impact propagation structure' and constitutes the end product of the impact determination process. Figure 10 shows an example of an impact propagation structure.

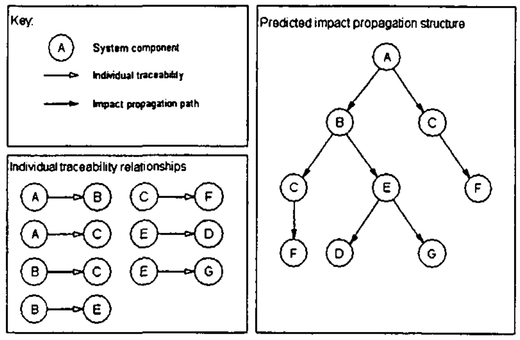


Fig. 10 An impact propagation structure

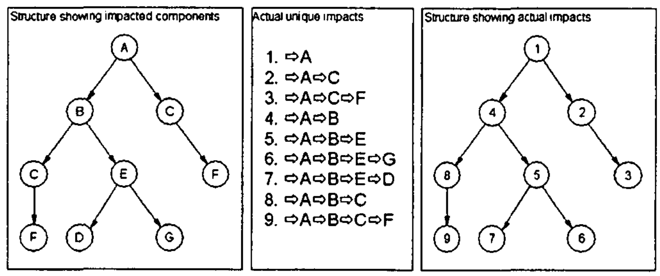


Fig. 11 Unique structure nodes

A single impactable may be associated with any number of nodes within the structure. However, every node is unique because the impact which it represents is defined by the total path to that node. This is shown graphically in Figure 11.

Propagation probability

Due to the potentially large size of the produced propagation structure, the hybrid approach utilises estimated probability values to provide some distinction between the many predicted propagation paths. In addition to this, probabilities are used during the combination of individual extraction techniques to reflect the differing reliability of the results produced by each.

Each propagation path identified is allocated a relative probability by the extraction method which identified the path. This probability value reflects the confidence with which the method has suggested the path. The value indicates the likelihood that, if the impact source was altered, the target would be impacted (Ajila 1995). These probability values are independent of each other in that the occurrence of an impact propagating between two impactables does not effect the probability of propagation between any other impactables. It is however important to take other probabilities into account when composing the final propagation structure, but this is a different matter and will be discussed in more depth in a later section.

It is not possible to differentiate between the significance of relationships derived from pre-recorded and dependency analysis. This is because such links are derived from simple entity relationships. Thus the probability of each impact propagation relating to a single call path will all be equal. However, if multiple relationships exist between two impactables, then the probability assigned to the impact path will be correspondingly higher.

Probability values given to propagations identified by past experience techniques depend on the number of times each impact path has been previously experienced. For example, if one change causes many impacts to a particular impactable, then the predicted propagation path which is produced will be associated with a relatively high probability. Conversely, if an impactable is rarely effected by a change then the predicted path will have a relatively low probability.

It is obvious that much of the generation of probability values is based on previous experience. When analysing previous experience, it is not possible to distinguish between the probabilities of impact propagation through an impactable and from one impactable to another. These two components of impact propagation are illustrated in figure 12.

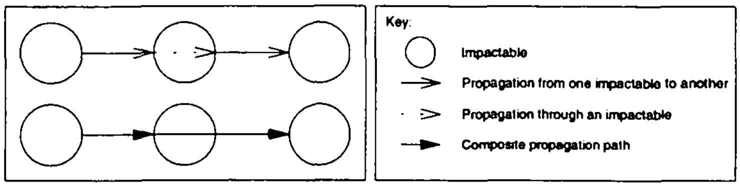


Fig. 12 Composite propagation paths

We cannot distinguish between the propagation probabilities of these two components because, when recording the impact effect of changes, only a single probability is actually observable and we cannot easily decompose it into its component parts. This is not a problem in its self in that such decomposition is not essential to the generation of probability values, however it should be borne in mind that all impact propagation paths actually represent such composite probabilities.

The production of probability values is particularly useful in providing answers to 'what if' questions. That is, when a developer wishes to observe the effect of a theoretical or experimental change, without the need for the enactment of that change. The tool developed to support the hybrid approach is particularly useful in such cases, as it provides the engineer with a rapid and accurate way of experimenting with different change scenarios.

The probability values for the ATM past experience relationships are given in Figure 13.

Affiliate	impacts	Database	with a probability of 0.7	Database	impacted by	Affiliate	with a probability of 0.7
New PIN	impacts	Database	with a probability of 0.8	Database	impacted by	New PIN	with a probability of 0.8
Affiliate	impacts	Valid	with a probability of 0.5	Valid	impacted by	Affiliate	with a probability of 0.5
New PIN	impacts	Valid	with a probability of 0.7	Valid	impacted by	New PIN	with a probability of 0.7
Affiliate	impacts	Printout	with a probability of 0.6	Printout	impacted by	Affiliate	with a probability of 0.6
Affiliate	impacts	Encrypt	with a probability of 0.9	Encrypt	impacted by	Affiliate	with a probability of 0.9
Maintain	impacts	Avail	with a probability of 0.7	Avail	impacted by	Maintain	with a probability of 0.7
Affiliate	impacts	Simple	with a probability of 0.8	Simple	impacted by	Affiliate	with a probability of 0.8
New PIN	impacts	Simple	with a probability of 0.5	Simple	impacted by	New PIN	with a probability of 0.5
Affiliate	impacts	Risk	with a probability of 0.6	Risk	impacted by	Affiliate	with a probability of 0.6

Fig. 13 Probability values

Certainty adjustments

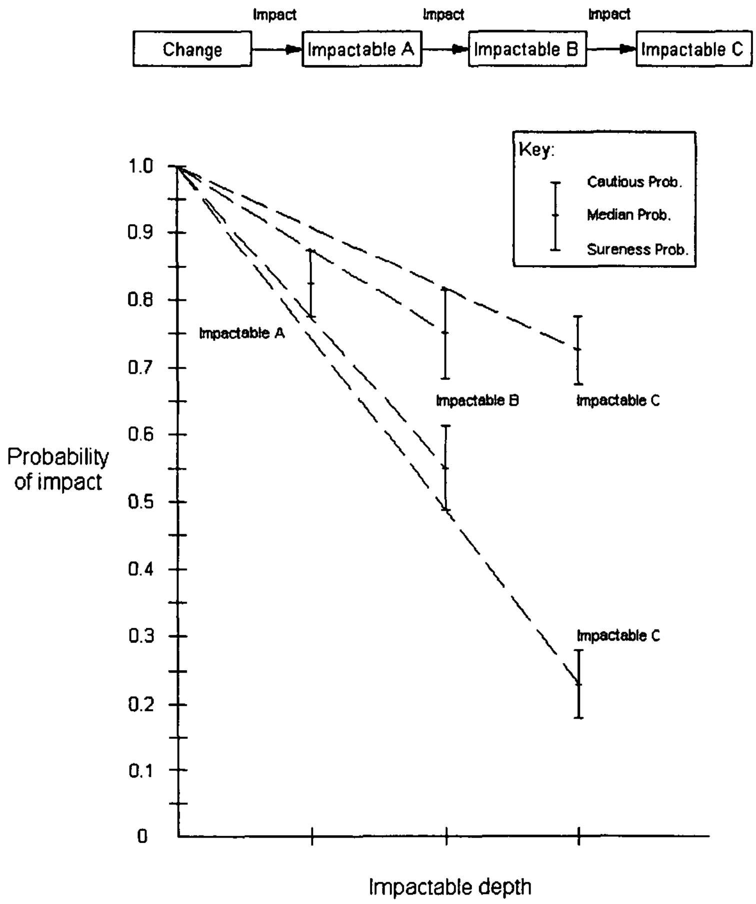


Figure 14 The effect of certainty adjustment

The certainty values generated by the tool can be used to adjust the calculated propagation probabilities in two distinct ways. The two types of probability values produced by the application of the certainty information are as follows:

- Sureness probabilities - These represent the estimated likelihood of impact calculated from the information available. Such values indicate the sureness with which the analysis technique predicts that impacts will occur. Thus, with low certainty values, predicted probabilities are reduced due to the high uncertainty of the information on which the predictions were made.
- Cautious probabilities - These represent the risk of impact calculated from the information available. Such values indicate an estimation of the danger of an impact occurring. Thus with low certainty values, predicted probabilities are increased due to the high uncertainty of the information on which the predictions were made.

The effect of these two types of probability value are illustrated in figure 14 which shows the two probability estimations for each of three hypothetical impactables. The diagram shows the effect of the different probability values on the values of the 'downstream' (indirectly impacted) impactables. We can see from figure 14 the large range of values that the impact probability of impactable C may have, depending on how certainty is used.
The cautious and sureness probability values for the three impactables A, B and C are show in figure 15

Impactable	Cautious	Sureness
A	0.875	0.775
B	0.825	0.475
C	0.775	0.175

Figure 15 Cautious and sureness probability values

In order to achieve either cautious or sureness probability values, we must apply the results of certainty analysis to the propagation path probabilities in a certain manner. The formulae used to calculate probability values is as follows:

For sureness probabilities, we reduce the probability of the propagation by scaling it using the calculated certainty value:

Adjusted probability of impact = Probability of impact * Certainty value of path

(1) $P_{adjusted} = P_{path} * C_{path}$

For cautious probabilities, we reduce the probability **against** propagation by scaling it using the calculated certainty value:

Adjusted probability of NO impact = Probability of NO impact * Certainty value of path

(2) $\neg P_{adjusted} = \neg P_{path} * C_{path}$

However, standard probability theory allows us to state that:

(3) $\neg P_{path} = 1 - P_{path}$

And that:

(4) $\neg P_{adjusted} = 1 - P_{adjusted}$

Substituting (3) and (4) into (2) we get:

$1 - P_{adjusted} = (1 - P_{path}) * C_{path}$

$P_{adjusted} = 1 - (1 - P_{path}) * C_{path}$

$P_{adjusted} = 1 - C_{path} + (C_{path} * P_{path})$

Where P_{path} is the probability calculated by the extraction mechanism, C_{path} is the certainty of the target impactable of the extracted relationship and $P_{adjusted}$ is the final probability assigned to the path.
For simplicity's sake, we will concentrate on the generation of sureness probabilities for the ATM example.

Traceability analysis

Traceability analysis involves the processing of the extracted traceability information to produce a single impact propagation structure. Once the traceability data has been extracted from the system representation, the individual traceability links are combined into a single structure. This does not simply mean combining the results of each method for the entire system, but rather interleaving them so that they can work together in unison. A diagram showing the process used is illustrated in Figure 16.

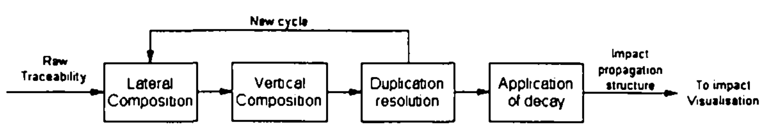


Fig. 16 Propagation structure generation

The generation of the impact propagation structure involves four distinct stages, the first three of which are interleaved and performed repetitively. We will first investigate the cyclic nature of the process and then move on to discuss the four key stages in more detail.

Analysis cycles

In order to identify the total potential impact of a change on a particular system, any analysis technique must be repeatedly applied to the system, forming a number of analysis cycles. This is true for the hybrid technique and in fact forms an important aspect of the combination process.
Each analysis cycle takes a set of impact sources (entities which must be altered to accommodate the change proposed) as inputs and produces a set of impact targets (entities which are impacted as a result of the alterations made to the sources) as outputs. Repeated analysis is required because the targets of the impacts can propagate the change impact to other entities and must therefore be fed into the next analysis cycle as impact sources (Thompson 1994). This is illustrated in Figure 17.

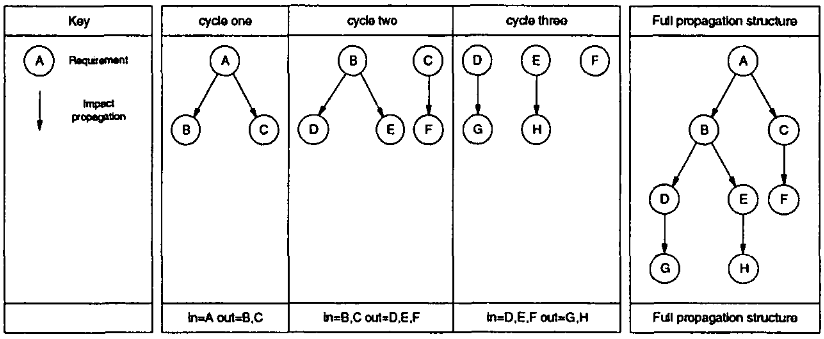


Fig. 17 Analysis cycles

Lateral composition

Lateral composition is the stage in which the outputs of individual methods are combined. This is achieved by summing all the propagation paths detected by the individual methods at each cycle of analysis. Figure 18 shows the combination of the results of two separate methods for a single cycle of analysis.

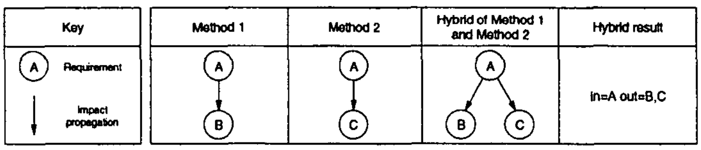


Fig. 18 Lateral composition

In order to allow propagation paths identified by different methods to be combined in this way, the results of each method must be weighted appropriately. This is done because the accuracy with which impacts can be predicted may vary depending on the reliability and focus of the method being used.
It is important to note that these weightings are relative and could thus have any range. The suggested method weightings for each of the techniques used in the framework are given below:

- Pre-recorded traceability analysis - 0.8
- Dependency analysis - 0.8
- Plain experience analysis - 1.0
- Extrapolation analysis - 0.6

By weighting the results of each individual method it is possible to adjust them to reflect the relative accuracy of each approach. In this way the results of disparate techniques may combined while still ensuring approximate parity between predicted impacts. Thus the probability of predicted impacts identified by dependable methods will be generally higher than impacts identified by less reliable methods.

Vertical composition

Vertical composition involves the chaining together of the results of each cycle of analysis in order to construct the complete propagation path structure. Each cycle of vertical composition adds an additional layer of predicted impacts to the propagation tree. This additional layer is the total set of impacts identified by the lateral composition stage of the current cycle. Figure 19 shows how lateral composition combines the results of each method and how vertical composition then takes these impacts to construct the next layer of the propagation structure.

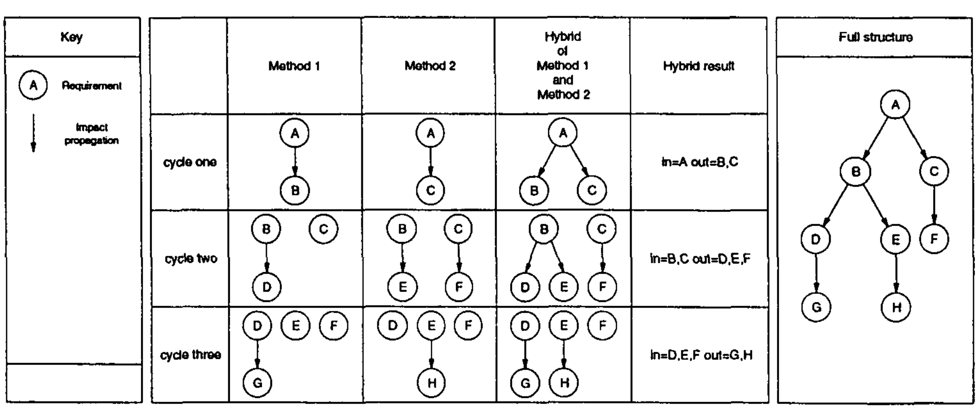


Fig. 19 Lateral and vertical composition

Figure 33 in the appendix shows the interleaved vertical and lateral composition stages as applied to the ATM example. The figure illustrates the predicted impact of a newly proposed change, referred to as *transfer*, on the ATM system. The impact propagation structure produced for the example system after lateral and vertical composition is illustrated in Figure 34 in the appendix.

Duplication resolution

It is possible for duplicate propagation paths to be identified between two requirements. This results from the same impact being predicted by two different extraction mechanisms or multiple predictions from the same method. During duplication resolution these are converted into a single propagation path in order to minimise the size of the final propagation structure. The probability of the single path is calculated from a combination of the values for the two original propagation paths. The probability value assigned to the predicted impact will be relatively high, reflecting the fact that the propagation has been detected by two independent mechanisms. This duplication resolution is done at the end of each cycle to minimise the number of inputs to the next cycle of analysis and prevent the replication of analysis.

A number of duplicate paths are present in the structure shown by Figure 34 in the appendix. In order to resolve these duplicate paths, an adjustment formula must be used for combining multiple probabilities. We shall now describe the adjustment formula which is used and in particular examine the purpose and rational behind that formula.

Let us imagine that we have a impactable 'A' and a impactable 'B' and that two impact propagation paths exist between them as shown in figure 20.

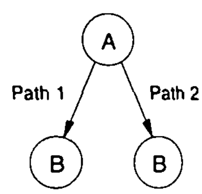


Fig. 20 Impactables and propagation paths between them

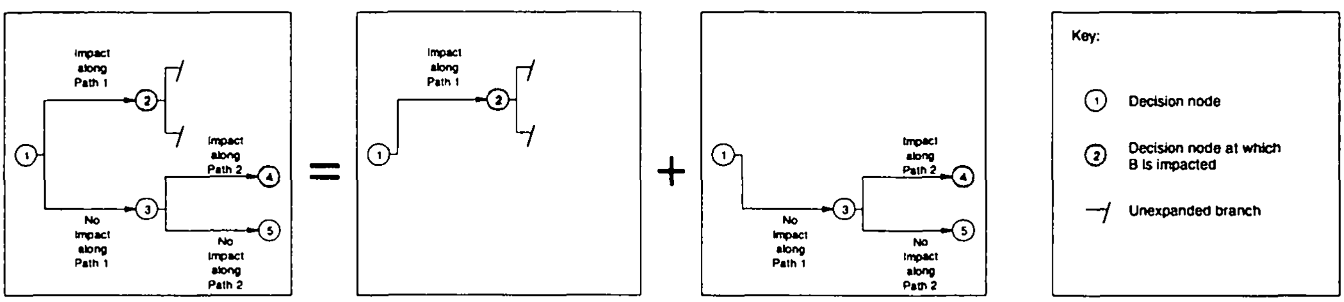


Fig. 21 Decision tree for an impact on impactable B

In order to combine both paths into a single link, we must integrate the impact propagation probabilities of the two paths into one. If we examine the decision tree for the impact on B (see figure 21), we can see that an impact on B may occur if:

- The impact propagates along path 1 (node 2 on the decision tree) or
- The impact does not propagate along path 1, but does propagate along path 2 (node 4 on the decision tree)

We can illustrate the probabilities of these conditions using the Venn diagrams shown in figure 22

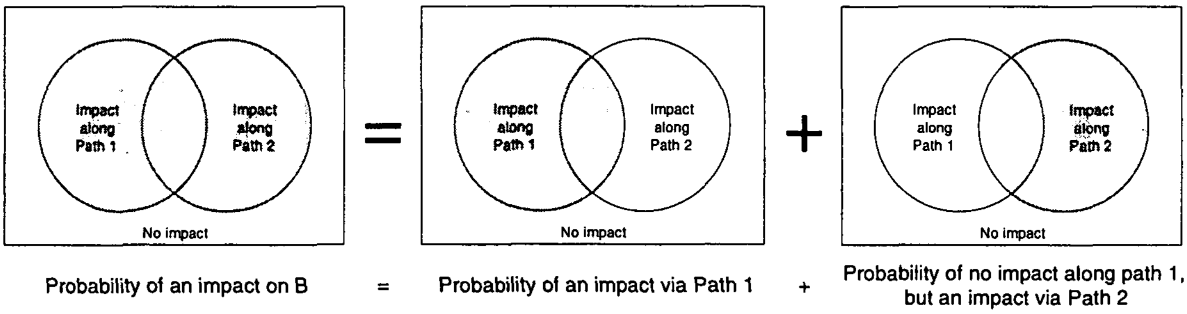


Fig. 22 Venn diagrams showing the probability of impact on B

The shaded areas of the Venn diagrams represent occurrences in which B is impacted. The diagram illustrates how the total probability of impact on B may be broken down into two separate areas. If we further decompose the rightmost Venn diagram from figure 22, we reach a stage where we may formalise the discussion and derive a formula for calculating the probability of impact on impactable B. This is illustrated in figure 23, which shows equivalent Venn diagram, natural language and formula for calculating the probability of impact on impactable B.

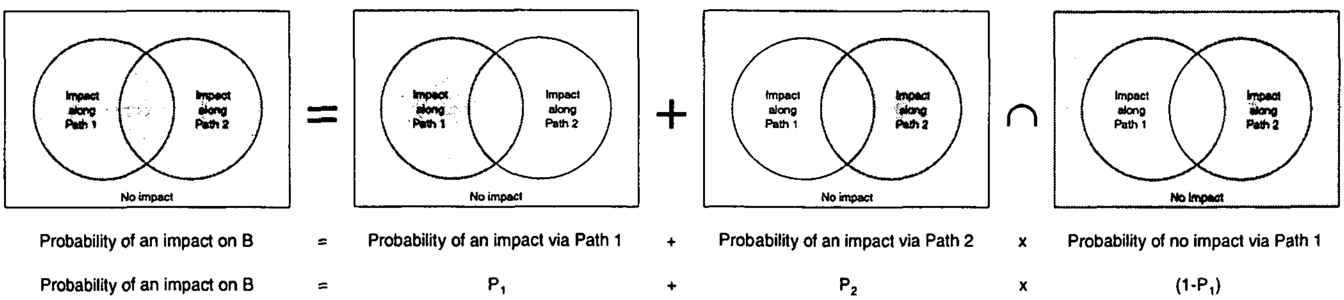


Fig. 23 Equivalent Venn diagram, natural language and formula

We can generalise this case and state the following formula for integrating the probabilities of any two impact propagation paths:

$$P_{adjusted} = P_{old} + P_{new}\,(1-P_{old})$$

Where P_{new} is an additional probability, P_{old} is the current probability before the assimilation of the additional probability and $P_{adjusted}$ is the total probability after adjustment and integration has taken place. If many propagation paths are to be integrated together, then the formula is applied incrementally, combining each additional path in turn to the composite path.

The resolved paths and adjusted probabilities for the ATM system are illustrated in Figure 35 in the appendix.

Application of decay

As mentioned before, the impact propagation probabilities associated with each propagation path are independent of all other probabilities. The occurrence of one impact along a propagation path will not effect the probability of the propagation paths between any other pairs of impactable. Despite this fact, when analysing the total effect of a given change, we cannot simply combine all the previously extracted traceability relationships without first considering their relative position in the final impact propagation structure. As we move through the layers of propagation in the generated structure, the probability of reaching a particular point decreases. This is because each propagation in a transitive path will usually have a probability of less than 1. Thus, when calculating the probability of a single propagation path, it is important take account of the probability of reaching the path in the first place. This process is known as probability decay and is illustrated in Figure 24.

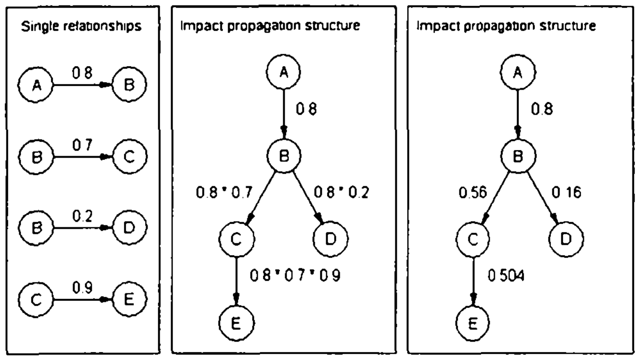


Fig. 24 Probability decay

In the diagram, the probability of an impact on impactable 'C' propagating to impactable 'E' is 0.9. However, this value needs to be adjusted when the propagation path is part of the full propagation structure because impactable 'C' has a less than one probability of being reached. Thus, for the given change, the probability of an impact upon impactable 'E' is the probability of an impact upon 'C' (i.e. 0.56) multiplied by the probability of that impact propagating to 'E' (i.e. 0.9) which gives a final result of 0.504.

Figure 36 in the appendix shows the effect of applying probability decay to the ATM propagation structure.

COMBATING COMPLEXITY

The size of the impact propagation structure shown in Figure 36 is significant, despite the relatively small size of the example system. It is thus obvious that the impact propagation structure produced from any non-trivial system will potentially be very large. A number of mechanisms can be employed to help minimise the extent of this structure. These include filtering methods such as:

- The prevention of recursive paths
- The use of a probability cut-off level
- The application of behavioural thread filtering
- The employment of viewpoint partitioning

These four mechanisms will now be discussed in more detail in the following sections. In addition to mechanisms which aim to reduce the physical size of the structure, it is also possible to ease the process of construction and investigation. This can be done by providing tool support for the impact analysis process, which can include facilities for collection and analysis of lateral traceability, as well as the visualisation of the produced impact propagation structures. An overview of the tool support for the integrated approach is provided in section 4 of this paper.

Prevention of recursive paths

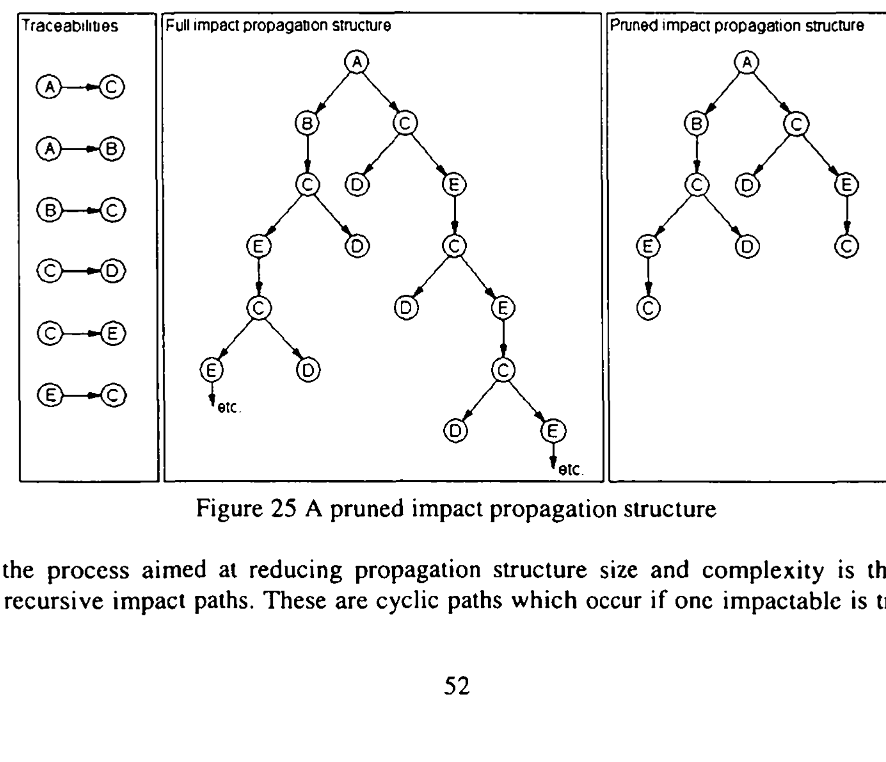


Figure 25 A pruned impact propagation structure

A feature of the process aimed at reducing propagation structure size and complexity is the detection and prevention of recursive impact paths. These are cyclic paths which occur if one impactable is traceable back to itself, either directly, or indirectly via a transitive path. If these cyclic paths are incorporated into the generated propagation structure, it can result in structures of infinite depth. Preventing these recursive paths involves recording which impactables have previously been visited when performing traceability analysis. This ensures that no impactable is expanded more than once in any single transitive path. By preventing these cyclic paths from appearing in the structure, it is possible to remove all recursive paths, and thus all duplicate information. This is illustrated in figure 25 which shows the effect of 'pruning' an impact propagation structure in this manner.

Probability cut-off

One possible way to help minimise the size of the generated impact propagation structure is to allow the specification of a probability cut-off value. This value represents a level below which the probability of impact is considered to low to be of significance. Only propagations with probabilities above the cut-off level will be included in the propagation structure. In this way we can minimise the size of the final impact propagation by removing propagations which are unlikely to occur and thus do not contribute significantly to the assessment of impact. Figure 26 shows the effect of two different probability cut-off values on a complete propagation structure.

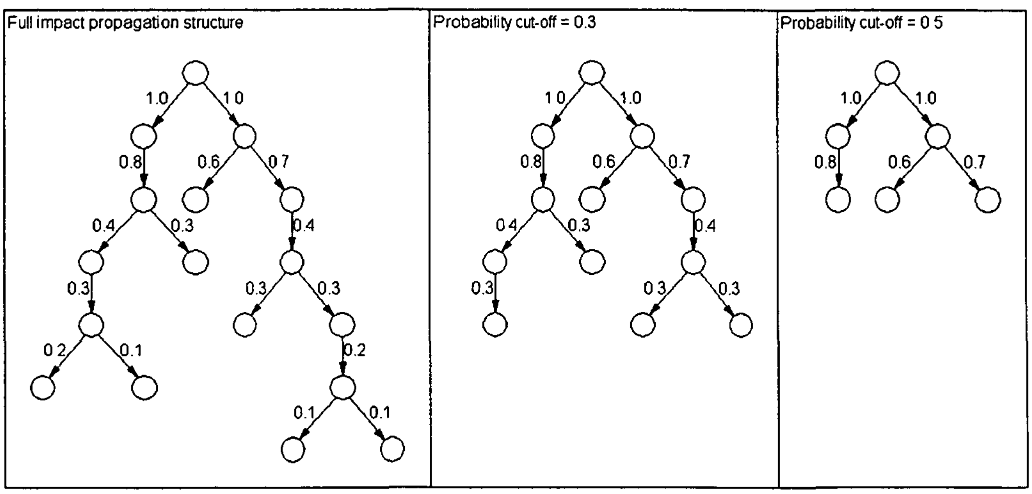


Fig. 26 Probability cut-off values

Behavioural thread filtering

Behavioural thread filtering is an abstraction technique which allows us to focus on a particular aspect of system behaviour when assessing the impact of change. This leads to a reduction in the complexity of the final impact propagation structure produced which can ease investigation and appreciation. In addition to this, utilising behavioural threads can provide a useful tool for investigating the impact of change upon a system. Such analysis can assist the requirements engineer in:

- Assessing the relative effects of different, potential changes on the same portion of system functionality
- Comparing the effect of a number of different changes on a number of different portions of system functionality
- Evaluating the resilience of different implementations of the same functionality to given changes
- Identifying potential impact 'hotspots' in the implemented functionality of the system
- Experimenting with a number of potential changes to try to identify which ones have the most significant effect on system functionality
- Determining which of a number of different change request implementation alternatives will most effect the critical operations of the proposed system.

To perform behavioural thread filtering, the requirements engineer must first specify a pattern of execution of the proposed system. This pattern of operation is referred to as a single 'behavioural thread' and represents a single path of flow control through the impactables which make up the proposed system. The behavioural thread is actually made up of a single slice through the behavioural models used to represent the operation of the system.

Once a behavioural thread has been specified, we must then identify the system impactables associated with that thread. Each model representing the system helps define one or more functional requirements. The functional requirements defined by the models in the behavioural thread are then all collected. This subset of all system functional requirements is then used as a basis for filtering the generated propagation structure. All requirements not present in this selected set are then filtered out of the propagation structure, so that only those associated with the behavioural thread are present. It is important to note that a complete analysis takes place and all possible transitive paths are derived, but only impacts on the identified subset of requirements will be present in the final structure.

Viewpoint partitioning

A useful technique for reducing the size and complexity of the generated impact propagation structure is the use of viewpoints to partition the system. Viewpoints are individual perspectives of a system and can include end users, external systems, managerial staff and other system stakeholders. Each viewpoint is associated with a subset of the entire collection of impactables of a system. By limiting the impactables present in the impact propagation structure to those of a particular viewpoint, we can produce a assessment of the impact of a change from a single perspective of the system. This can not only greatly reduce the complexity of the propagation structure, but provides us with a valuable tool for isolating desirable segments of impact information. When assessing the effect of change upon a system, we may often only be interested in how a change impacts a single viewpoint of that system. For example, it may be desirable for the user to assess how a particular change will effect the security viewpoint of a system. Even if we need to identify the effect of the change from all perspectives of a system, the use of viewpoints can assist in breaking the total impact into logically grouped, more easily digestible segments. These can then be explored sequentially in order to investigate the impact of the change on the system in it's entirety. It is useful to classify the viewpoints of a system exist into a hierarchical structure such as that show in figure 27.

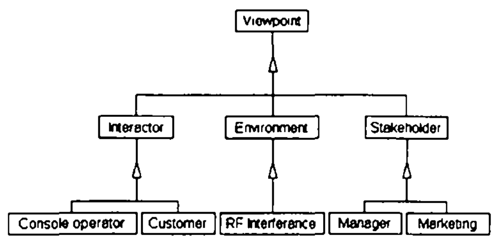


Fig. 27 A Viewpoint hierarchy

It then becomes possible to select specific viewpoints (such as console operator), abstract viewpoints (such as interactor), or even the root viewpoint. This gives us the flexibility to focus on individual viewpoints or alternatively general viewpoints, as well as the ability to generate a single propagation structure of the whole system.

Tool support

We have implemented an extensive prototype tool to support the integrated approach described here. The tool provides facilities for:

- Recording, managing and analysing requirement artifacts
- Abstract system modeling
- Recording, managing and analysing proposed changes
- Automated traceability extraction
- Automated propagation structure creation
- Advanced impact visualisation
- Impact investigation and appreciation features

Figures 28 to 30 in the appendix show screenshots of the implemented systems. These include the interface components for the specification, maintenance and examination of functional requirements (figure 28), non functional requirements (figure 29) and changes (figure 30). In addition to these interface elements, figures 31 and 32 in the appendix depict the impact propagation structure visualisation interface and magnification facilities respectively.

CONCLUSIONS

This paper has investigated the important process of impact determination and analysis within the general context of requirements change management. We have presented the various traceability extraction mechanisms available for performing requirements level impact analysis. We have discussed shortcomings of individual traceability extraction techniques and proposed an integrated approach to addresses these problems.

Our proposed solution achieves the composition of multiple traceability extraction approaches while maintaining approximate parity between predicted probabilities. The result of this work has been the development of a novel framework, which successfully integrates past experience based approaches with more standard dependency and pre-recorded techniques. The issue of uncertainty has also been addressed by the framework, which uses certainty values to adjust generated probability values.

To ensure the scalability of the approach a number of filtration mechanisms have been developed. These can be used to both minimise the set of impacts predicted by the process as well as providing important techniques for revealing different aspect of the system. In addition to this, the analysis of large systems is also eased by the availability of an integrated tool which supports all of the features described in this paper.

We are currently exploring ways of extending the change impact visualisation to integrate use cases and event scenarios. By using scenarios we hope to incrementally explore the change process and the wider context in which the change occurs. We intend to integrate static and dynamic visualisation. Dynamic visualisation will show how control ripples through the system and how change affects exceptions.

We are also investigating ways of extending our approach to support impact analysis in the later stages of system development. In this respect we are exploring ways of extending the notion of 'Impactables' to include artifacts from the design, implementation, testing and maintenance stages of the lifecycle.

REFERENCES

Ajila, S. , (1995), Software Maintenance: An Approach to Impact Analysis of Objects Change, **Software-Practice and Experience** 25(10) October, Comer, D. E. ; Wellings, A. J., pp1155-1181

Bennett, K. , (1996), Software evolution: past, present and future, **Information and software technology** 38(11) November, Dyer, M. ; Shepperd, M., pp673-680

Bohner, S. A. , Arnold, R. S. , (1996), **Software Change Impact Analysis**, ix-xii, IEEE Computer Society, Los Alamitos

Ecklund, E. F. , Delcambre, L. M. L., Freiling, M. J. , (1996), Change Cases: Use Cases that Identify Future Requirements, **SIGPLAN notices** 31(10), October , Berman, A. M., pp342-357

Goradia, T. , (1993), Dynamic Impact Analysis: A cost-effective Technique to Enforce Error-propagation, **Proceedings of the 1993 International Symposium on Software Testing and Analysis,** June, pp171-181

Han, J. , (1997), Supporting Impact Analysis and Change, Propagation in Software Engineering Environments, **Proceedings of the Eighth IEEE International Workshop on Software Technology and Engineering Practice**, July, pp172-182

Harker, S. , Eason, K. , Dobson, J. , (1993), The Change and Evolution of Requirements as a Challenge to the Practice of Software Engineering, **Proceedings of the 1993 Requirements Engineering Symposium**, January, pp266-272

Li, L. , Offutt, A. J. , (1996), Algorithmic Analysis of the Impact of Changes to Object-Oriented Software, **Proceedings of the International Conference on Software Maintenance**, November, pp171-184

Lock, S. , Kotonya, G. , (1998), **Requirement Level Change Management and Impact Analysis**, internal technical report no. CSEG/21/98, October, Lancaster University Computing Department

McCrickard, D. S. , Abowd, G. D. , (1996), Assessing the Impact of Changes at the Architectural level: A case study on Graphical Debuggers, **Proceedings of the International Conference on Software Maintenance**, November, pp59-67

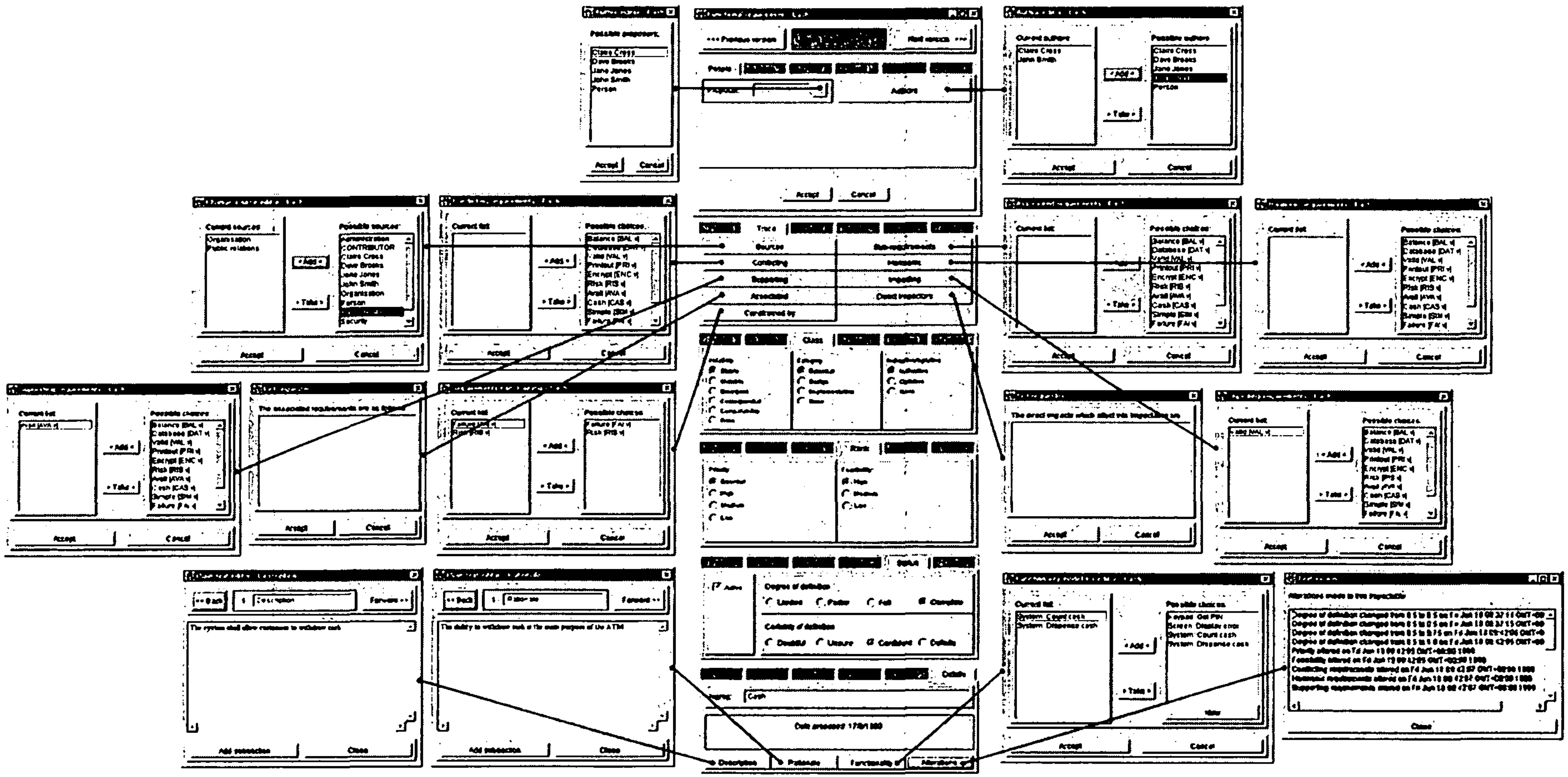
Moreton, R. , (1996), A Process Model for Software Maintenance, **Software Change Impact Analysis**, Bohner, S. A. , Arnold, R. S., pp29-33, IEEE computer society, Los Alamitos

Moriconi, M. , Winkley, T. C. , (1990), Approximate reasoning about the effects of program changes, **IEEE Transactions on Software Engineering** 16(9), September, Basili, V. R., pp980-992

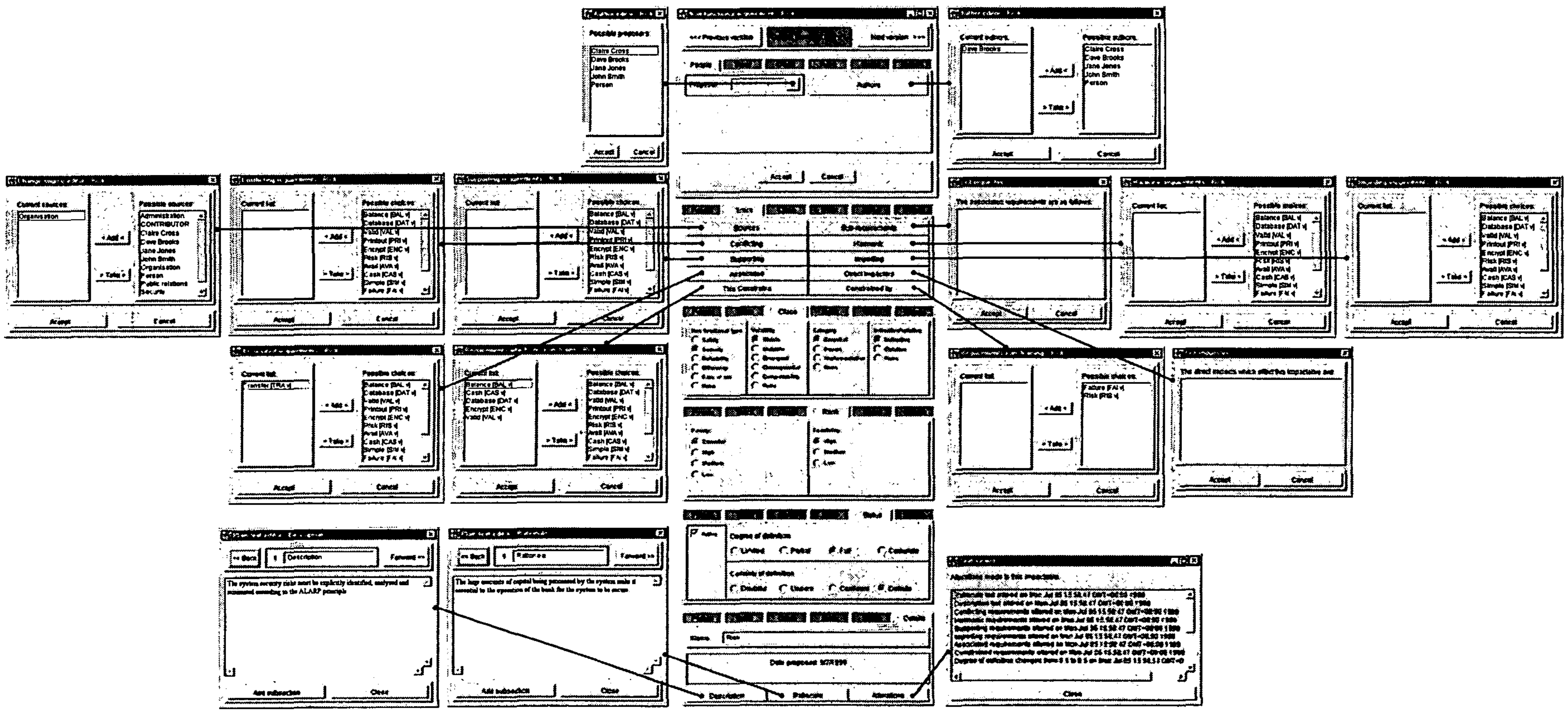
Paakki, J. , Salminen, A. , Koskinen, J. , (1996), Automated Hypertext Support for Software Maintenance, **The Computer Journal** 39(7), August, van Rijsbergen, C. J., pp577-597

Sommerville, I., (1996), **Software Engineering**, Addison-Wesley, Wokingham

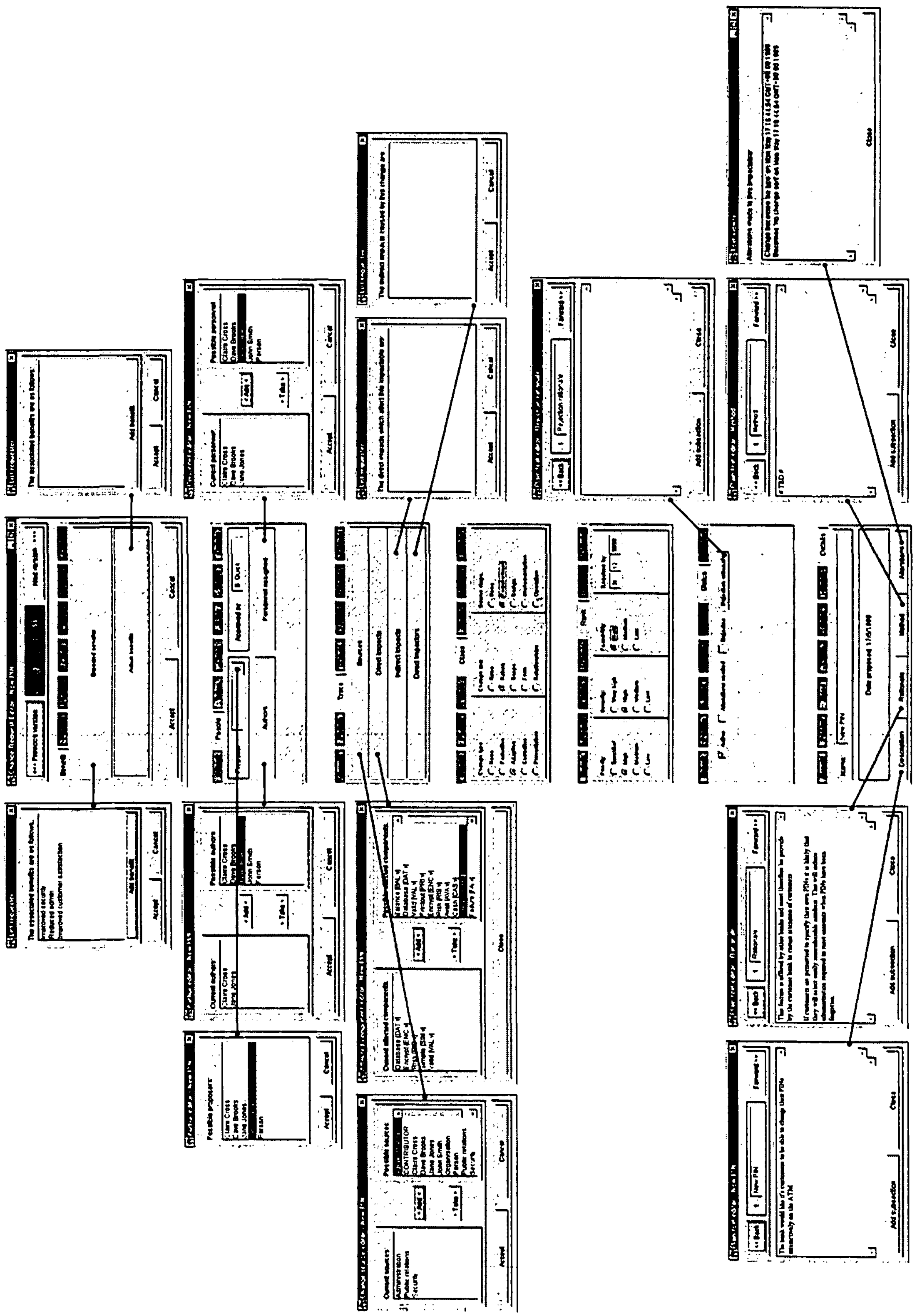
Thompson, J. , (1994), What you need to manage requirements, **IEEE Software** 11(2), March, Sharon, D., pp115-118



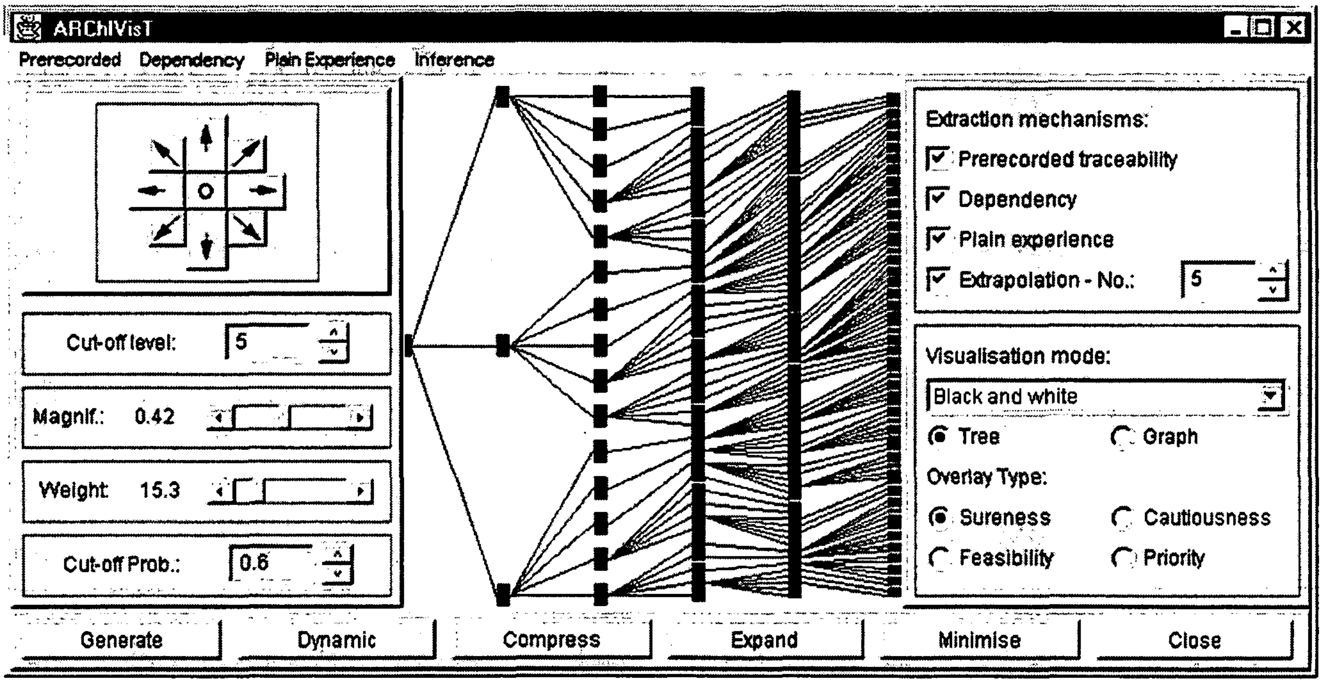
Appendix - Fig. 28 Interface for functional requirements



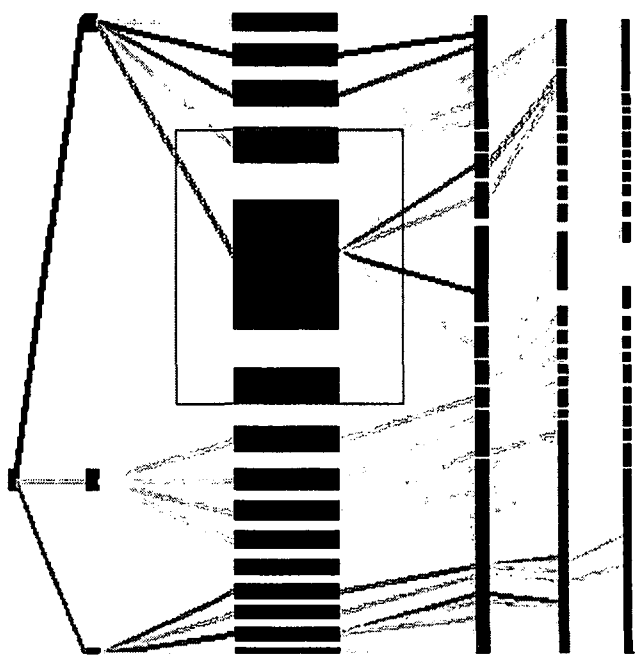
Appendix - Fig. 29 Interface for non functional requirements



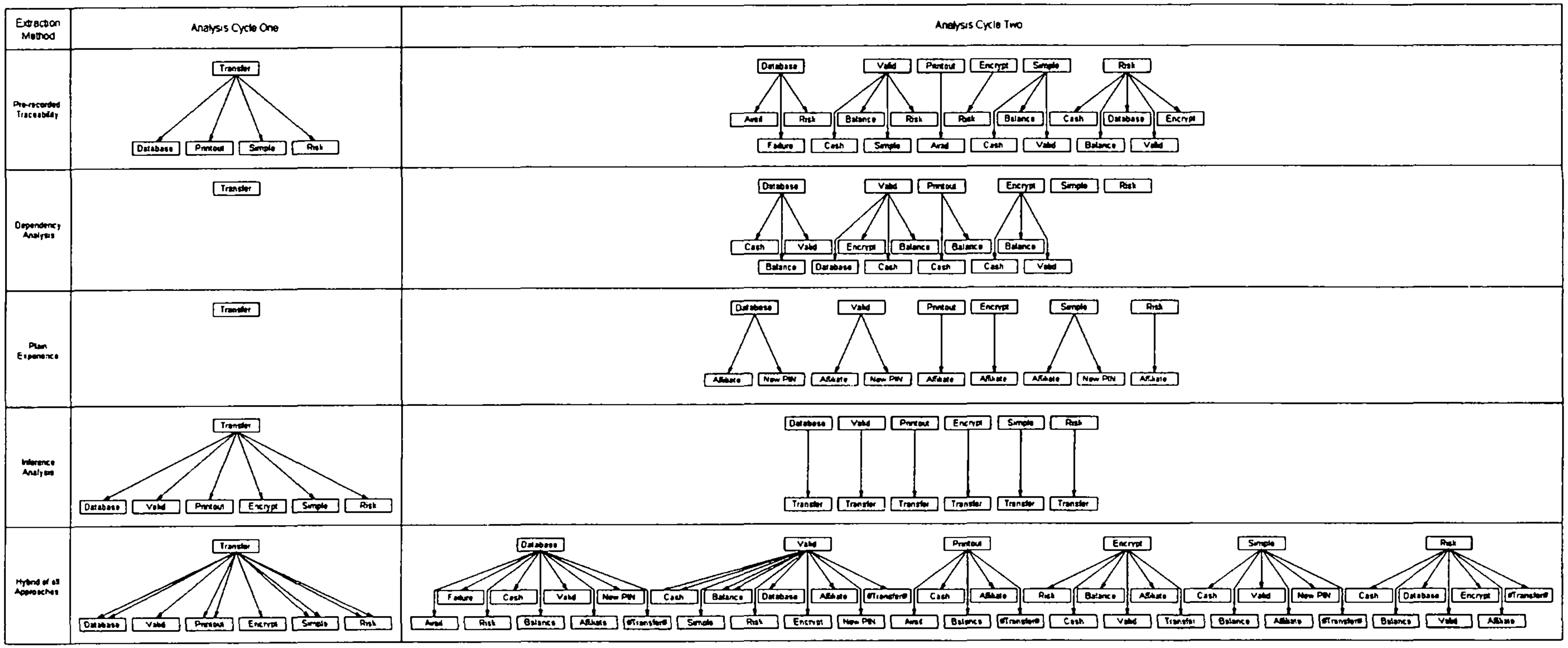
Appendix Fig. 20 Interfaces for proposed changes



Appendix - Fig. 31 Visualisation interface



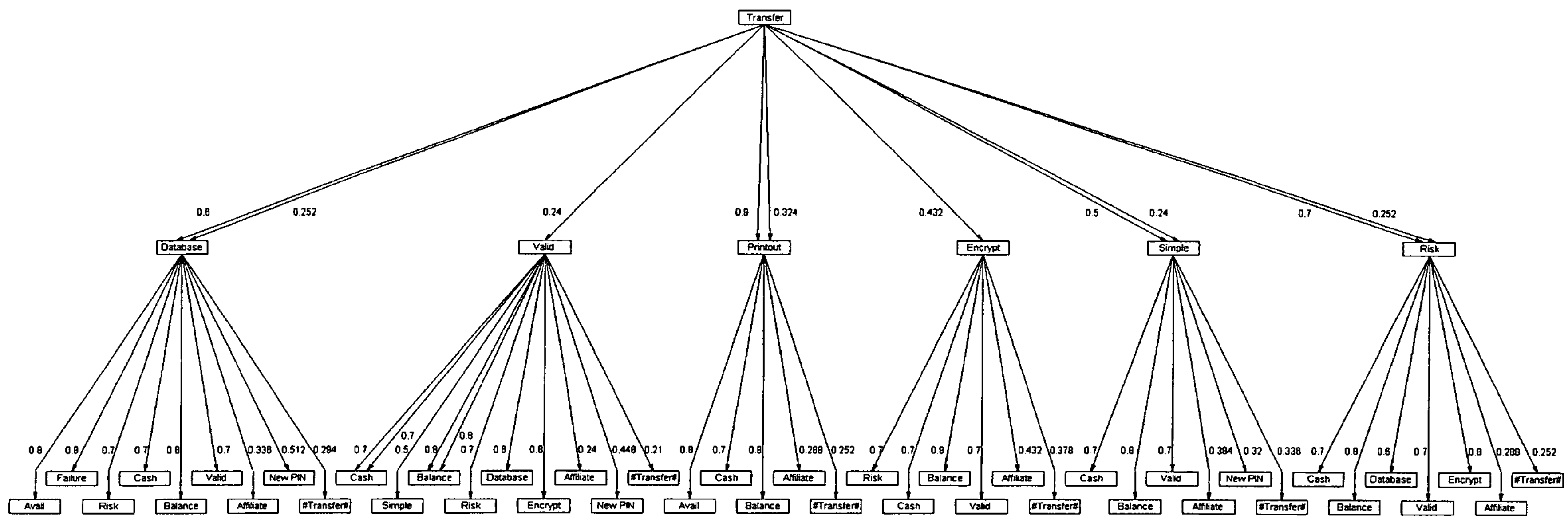
Appendix - Fig. 32 Magnified propagation structure

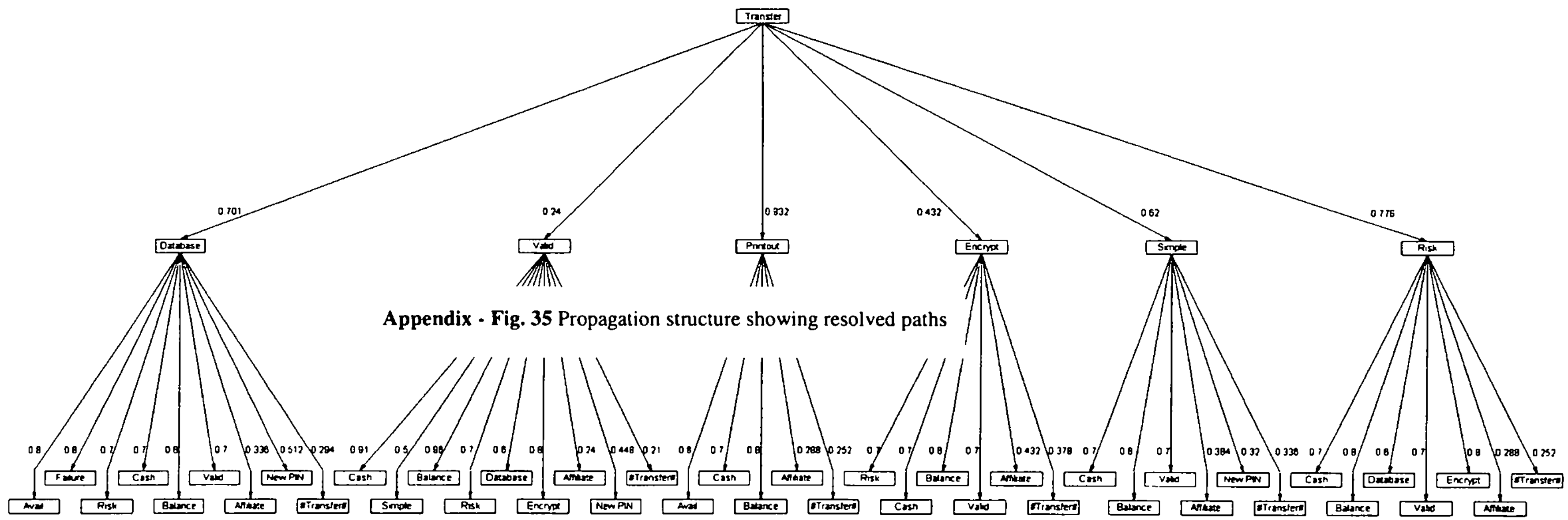


N B #<name># indicates a node which has been pruned and will thus not be expanded further

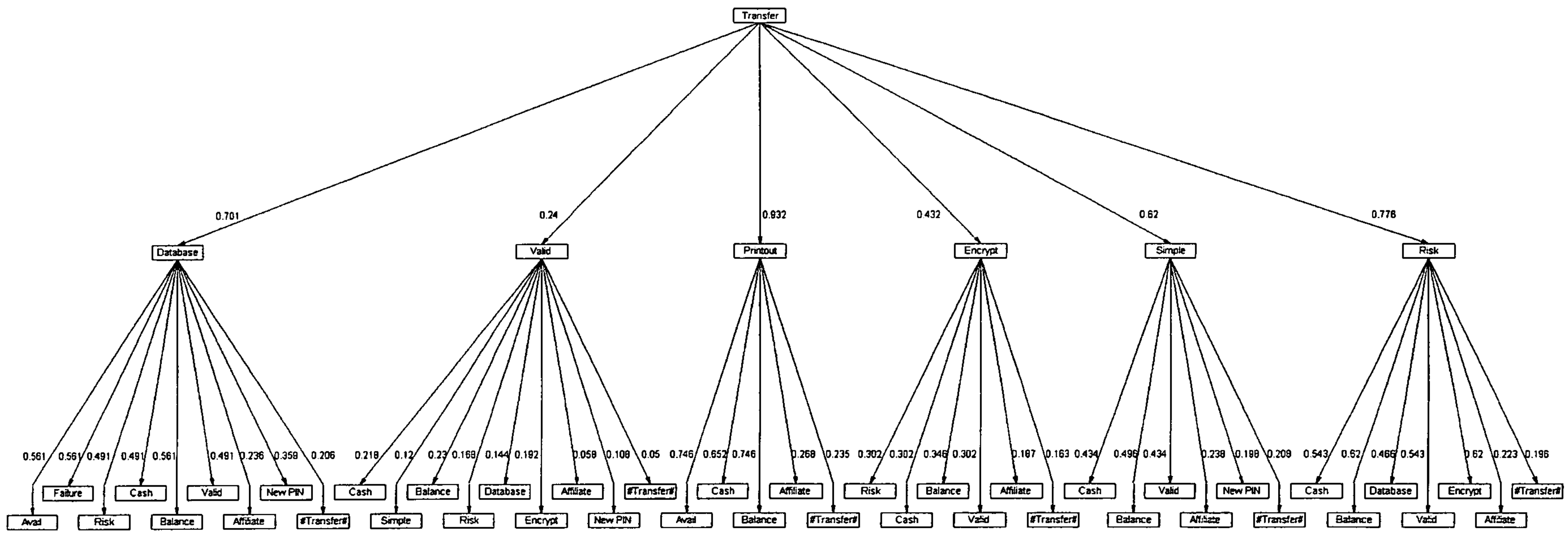
Appendix - Fig. 33 Lateral and Vertical Composition

Appendix - Fig. 34 Propagation structure showing duplicate paths





Appendix - Fig. 35 Propagation structure showing resolved paths



Appendix - Fig. 36 Propagation structure showing probability decay