

Impact Analysis - Towards A Framework for Comparison

Robert S. Arnold

Software Evolution Technology
12613 Rock Ridge Rd.
Herndon, VA 22070
703-450-6791
r.arnold@compmail.com

Shawn A. Bohner

MITRE Corporation
7525 Colshire Dr.
McLean, VA 22102
703-883-7354
bohner@mitre.org

ABSTRACT

The term "impact analysis" is used with many meanings. We define a three-part framework for characterizing and comparing diverse impact analysis approaches. The parts correspond to how an approach is used to accomplish impact analysis, how an approach does impact analysis internally, and the effectiveness of the impact analysis approach. To illustrate the framework's application, we classify five impact analysis approaches according to it.

1. Introduction

1.1. Purpose

Many activities are termed "impact analysis," yet it is difficult to relate them. Impact analysis (IA) approaches should be characterized so that IA approaches can be understood, compared, and assessed. This paper presents a framework for doing this.

The framework aids comparing IA approaches, assessing the strengths and weaknesses of individual IA approaches, and unifying the widely varying IA technology within a single conceptual framework. We present the framework, justify it, and use it to compare five IA approaches.¹

1.2. How the Reader Can Use These Results

If the reader is interested in understanding, evaluating, or using IA technology, reading this paper will be helpful. By understanding the parts of the IA framework, the reader will see several features of IA that could be in a given IA approach, but may not be. This will help the reader assess the potential value of an IA approach. The reader can use the parts of the framework to critique claims of IA made by software tool vendors or by researchers.

Vendors and researchers may use the framework to redefine their work in terms comparable to other approaches and

¹ By "approach" we mean tools, semi-automatic procedures, and manual procedures.

tools. This will help them track IA technology improvements.

1.3. New Results

This paper has several new results. First, the framework for understanding and classifying IA approaches is new. We are unaware of any other such paper in the IA literature. The comparison of the five IA approaches systems using the framework is new. We have not seen different types of IA approaches compared according to a common framework.

1.4. Paper Structure

Section 2 discusses why an IA classification framework is needed and the issues such a framework should address. Section 3 presents the framework. Section 4 applies the framework to compare five IA approaches and tools. Section 5 relates our work to others'.

2. The Need for an Evaluation Framework

2.1. Definition

Impact analysis (IA) is the activity of identifying what to modify to accomplish a change, or of identifying the potential consequences of a change. Examples of IA are:

- using cross reference listings to see what other parts of a program contain references to a given variable or procedure,
- using program slicing to determine the program subset that can affect the value of a given variable [Gallagher1991],
- browsing a program by opening and closing related files,
- using traceability relationships to identify changing artifacts,
- using configuration management systems to track and find changes, and
- consulting designs and specifications to determine the scope of a change.

IA precedes, or is used in conjunction with, change. It

provides input to performing the change. Normally, nothing changes except our understanding of what may be involved with the change.²

2.2. No Consensus Definition

IA has been practiced in various forms for years, yet there is no consensus definition. For example, IA does not appear in the IEEE Glossary of Software Engineering Terminology [IEEE1983]. [RADC1986] defined IA as "an examination of an impact to determine its parts or elements." (They defined an impact as the "effect or result of making a change to a system or its software.") [Pfleeger91] defined IA as "the evaluation of the many risks associated with the change, including estimates of the effects on resources, effort, and the schedule." (p. 433)

2.3. Related Terms

There are other IA-related terms. An **impact** (noun) is a part determined to be affected, and therefore worthy of inspection. **Traceability** is the ability to determine what parts are related to what other parts according to specific relationships. A **side effect** is an "error or other undesirable behavior that occurs as a result of a modification" [Freedman1981]. **Stability** is "...the resistance to the potential ripple effect which a program would have when it is modified" ([Yau1980], p. 28). **Ripple effect** is the "effect caused by making a small change to a system which affects many other parts of a system." [Stevens1974]

2.4. Problems with Impact Analysis Divergence

The lack of a common view of IA, and the proliferation of related terms, has led to several problems:

- It is hard to decide what is meant by IA. People rarely give explicit definitions.
- There is a lack of dimensions for comparing one IA approach with another.
- It is hard to know if enough information is available for significant comparison.
- It is hard to discern when different work on IA is related.
- It is hard to discern what work contributes to IA and what does not, according to a basic framework for assessing the technology.

This paper presents a conceptual framework for resolving these problems.

²Some IA approaches, for specialized applications, have the option of actually performing a change once impacts are found. We consider this an added feature and not part of the basic impact analysis definition.

3. The Impact Analysis Framework

In this section we present the framework intuitively. First we summarize the major parts of the framework. Next we discuss each part in more detail. Then we summarize the collected features of the framework as a way to compare IA approaches.

3.1. Overview

Figure 3-1 outlines how to use the framework. The framework can be used to guide understanding of an IA approach, to compare or evaluate IA approaches, or to structure analyses of IA approaches. The framework provides several points for assessing an IA approach. This

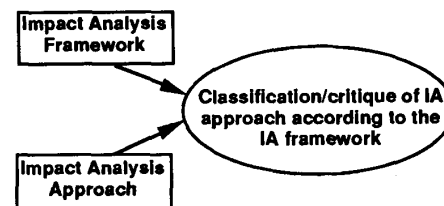


Figure 3-1. How to Use the Impact Analysis Framework

will result in a critique or assessment of the IA approach according to the framework.

Figure 3-2 summarizes the three parts of the IA framework: IA Application, IA Parts, and IA Effectiveness. IA Application examines how the IA approach is used to accomplish IA. It looks at the features offered by the IA approach interface. IA Parts examines the nature of the internal parts and methods used to actually perform the IA. IA Effectiveness examines properties of the resulting search for impacts, especially how well they accomplish the goals of IA.³

The following sections, describing each part of the framework, are structured as follows: First, the purpose of the framework part is given. Then a diagram is given to frame its context. The parts of the diagram are discussed. Finally, a table is given that summarizes the framework elements resulting from this part.

³It is common, in evaluations of tools and technology, to create evaluation criteria for technology-specific and technology-generic factors. The latter include reliability of the vendor, user-friendliness of the tool interface, level of customer service, etc. In discussing this framework, we just focus on the impact analysis-specific items. Non-functional criteria are not discussed in this paper, but often do form a part of a technology evaluation.

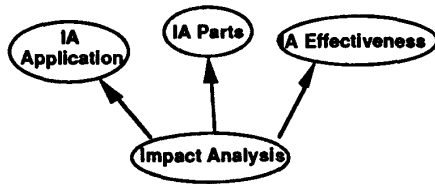


Figure 3-2. Parts of the Impact Analysis Framework

3.2. IA Application

IA Application examines how the approach is actually used to perform IA. To accomplish IA, we must have a proposed change, something to be changed, and a way to estimate what must be done to do the change.

Figure 3-3 pictures a generic IA process. A change is conceived in the real world, then reduced to a change specification.⁴ The change specification uses objects and relationships familiar to the change specifier. These objects and relationships are drawn from the artifact object model. The change specification and knowledge of the item to be changed are used to specify what is initially impacted to the IA approach.

The IA approach then determines what else may be affected. These results are then translated, if necessary, back into real world terms. The results are then used to plan, to scope, or to accomplish the change.

The IA approach may also provide other features, such as

- explanations of why items are estimated to be impacted,
- measures of the IA itself,
- animation illustrating how the impacts ripple,
- access to change histories,
- suggested change strategies,
- actually performing the change,
- ways to test the change, and
- graphical views of impacts.

An example of IA is when a programmer is given an engineering change report and asked "what is involved" to do the change. The programmer should provide a difficulty assessment (how hard it will be for the programmer to do the change), a level of effort estimate (how long the

⁴ Often many intermediate steps are done before a change specification is reached. We focus here just on the key conceptual elements of the technical impact analysis process.

change should take) and perhaps a risk assessment (how complicated the change will be to perform). IA—here browsing program code and forming a strategy for accomplishing the change—helps the programmer to answer all three points.

The key elements of IA application are given in Table 3-1.

3.3. IA Parts

This part of the framework concerns the functional parts of the IA approach—what the approach does, and how it does it, and the duties of the agents or tools involved.

Figure 3-4 illustrates the elements of IA Parts. To express a specific change, the IA approach has its own model of objects and relationships at its interface. The input, expressed in terms of the interface object model, is translated into the IA approach's internal object model.

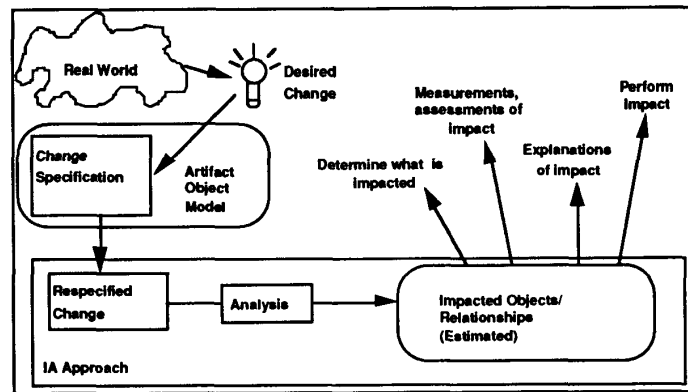


Figure 3-3. IA Application: Performing from the User's Viewpoint

The internal object model defines the objects and relationships (or dependencies) the approach uses to accomplish IA.

The internal object model is normally stored in a repository of some kind. The repository has its own features for loading, browsing, and modifying objects and relationships. The repository is loaded by decomposing the artifact into objects and relationships conforming to the internal object model.

The impact model defines the rules or embedded assumptions reflecting the semantics about what affects what. It defines the classes of objects and relationships used by the IA approach, and ways (rules, algorithms) for determining when a change to one object will affect another object. These may be embedded in the internal

object model or the impact calculation algorithms. Sometimes they may appear as a separate rules base.

The tracing/impact approach implements the impact model. The tracing/impact approach defines how objects and dependencies are represented, how impact rules are captured (e.g., programmed), and the specific search algorithms used to find impacted objects and relationships.

Once the results of IA are obtained at the internal object model level, these must be translated back into the interface object model, then further interpreted to determine what parts of the original artifacts are impacted. For some artifacts (e.g., programs), often the directly impacted artifact objects are supplied by the impact analysis approach. For other artifact sets (e.g., requirements), significant manual work is needed to accomplish determine what is impacted.

Each of these parts has many variations that, for brevity, we do not discuss here. Table 3-2 summarizes the elements of "IA Parts."

Element	Explanation	Rating Scale
Artifact Object Model (Domain)	What are the types of objects and relationships captured from the application domain?	Program objects and/or relationships; Predefined domain objects and/or relationships; User specifiable domain objects and/or relationships; None; Unknown
Decomposition	Can the item to be analyzed be automatically decomposed and stored within the IA approach/tool?	Yes, syntax with complete semantics; Yes, syntax with some semantics; Yes, syntax only; No; Unknown
Change specification	How is the change specified for the IA approach?	Yes, with detailed analysis; Yes, with some analysis; No, not applied; Unknown
Results specification	How are the results of IA expressed?	Report; Browsing; Database view; None; Unknown
Interpretation	How much effort by the user is needed to interpret the results (i.e., derive true impacts from IA)?	Significant; Some; None; Unknown
Other features	What other features are available to the user?	<The specific feature>. E.g.: explanations, metrics, impact animation, options to perform the change, access to change histories, suggested change strategies, ways to test the change; None; Unknown

An example illustrating IA Parts is incremental program recompilation. The programmer makes a change to software and the compiler must determine the minimal parts that must be recompiled, and in what order. The change here is specified implicitly: the compiler detects which parts of the code have been modified. The change is then translated into the compiler's compilation graph, which captures compilation dependencies between compilable code units. The compilation graph was produced through earlier compilations of the code. The compiler then applies its impact algorithm to determine what program units must be recompiled and in what order. (Often this is not visible to the programmer, or is just taken for granted.) Though not part of IA, the compiler then often goes ahead and recompiles the affected program units. What the programmer sees is a program that has been recompiled.

3.4. IA Effectiveness

This part of the framework concerns how well the IA approach accomplishes IA. Once IA is done, how

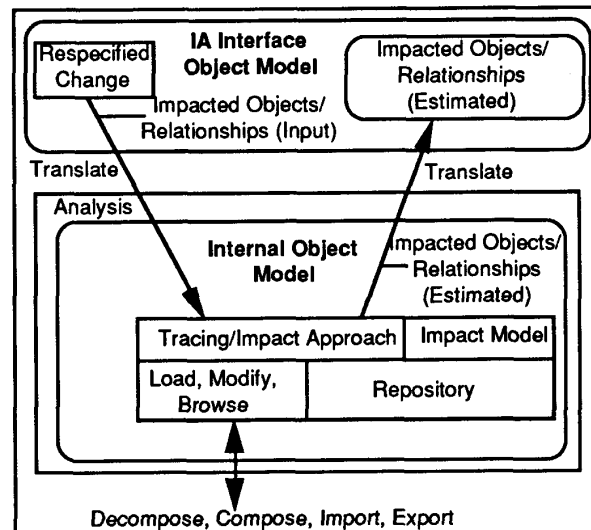


Figure 3-4. IA Parts: Functional parts of an IA approach.

accurate is it?

3.4.1. Definitions

To discuss effectiveness, we introduce the concepts pictured in Figure 3-5.

At the artifact object model level, IA is assumed to take place on a bounding set of objects, which we shall call the *System*. These objects are drawn from an encompassing model called the *Universe*.

At the interface object model level, the analogs of the *System* and the *Universe* are the *System#* and *Universe#*. (We append a "#" to sets that are at the interface object model level.) The *starting impact set* (SIS#) is the set of objects that are thought to be initially affected by a change. The *estimated impact set* (EIS#) is the set of objects estimated to be affected by the IA approach. The SIS# and EIS# are assumed to share the same object model, namely the interface object model. The *impact paths* are the search paths tying objects in the SIS# to objects in the EIS#.

The *actual impact set* (AIS) is the set of objects (in the artifact object model) actually modified as the result of performing the change. The AIS# is the image of the AIS in terms of objects and relationships in the interface object model.

The AIS is normally not unique, since a change can be implemented in several ways. (Nevertheless, in our discussion we will mention "the" AIS, meaning an AIS resulting from a particular impact analysis.) In our examples, we will also assume that the AIS reflects a correct implementation of a change.

It is also possible to characterize the SIS, EIS, and AIS in terms of the internal object model. For simplicity in discussing the framework, we shall discuss them at the level of the interface object model and above.

3.4.2. Effectiveness Concepts and Measures

We consider four areas that should be examined to determine the effectiveness of an IA approach.

3.4.2.1. SIS# and EIS#

This area looks at the relationship of the SIS# with EIS#. By definition, the EIS# always contains the SIS#. Yet the relative size of the EIS# influences the work to be

done later in checking the objects that the IA approach estimates to be affected.

Table 3-3 discusses the possibilities. For each case, a picture of the relationship, with defining conditions, is given. Then a metric with a measurement is given to detect when the case occurs. A point description (i.e., in looking at a single measurement) of the implications of the case is described. Finally, the "Desired Trends" indicates the desired, expected, and goal measurement tends that would be wanted over several applications of the IA

Element	Explanation	Rating Scale
Interface Object Model	What objects and relationships can be expressed at the approach's interface?	Strings; Program objects; Predefined document objects; User specifiable document objects; Unknown
Internal Object Model	What objects and relationships does the approach use to accomplish IA?	Document oriented; Object based; Graph structure; None; Unknown
Impact Model	How are dependencies modeled? Where does the approach take these dependencies into account? How closely does the impact model mirror dependencies of the artifact's model of dependencies?	Data flow; Control flow; String matching; Object dependency; None; Unknown
Tracing/Impact Approach	How does the approach accomplish IA through tracing affected objects and relationships? What algorithms or procedures are used?	Decomposition; Pattern matching; Heuristic search; Stochastic search; Not explicit; None; Unknown
Decomposition	What objects and relationships are captured from the artifact and stored as objects and relationships within the repository?	Compiler; Database entity; Object filter; None; Unknown
Repository	What repository is used to store objects and relationships?	Relational Database Management System (RDBMS); File system; None; Unknown
Load, modify, browse	What features does the repository have for loading objects and relationships into it, modifying them, and browsing them?	Load; Modify; Browse; All three available; None; Unknown

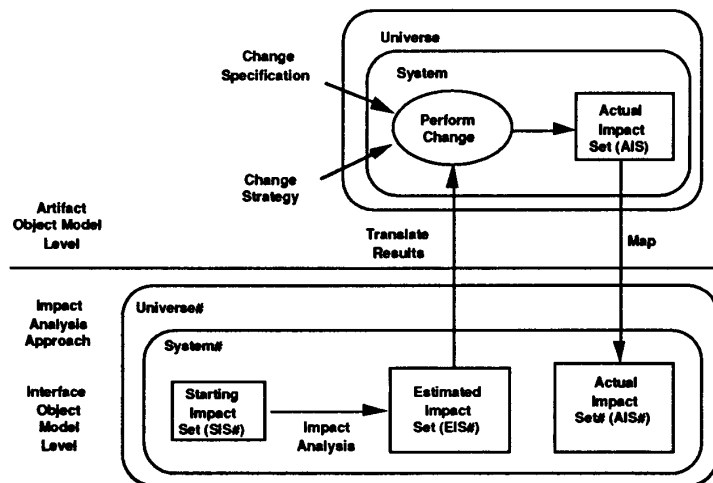


Figure 3-5. Key sets of objects in IA effectiveness.

approach. (The percentages in this column are suggested starting points. They may be tuned as desired.) The percentages in the "Expected" and "Goal" trends sum to 100% across the boxes, rather than within the boxes, because of mutually exclusive cases.

3.4.2.2. EIS# and System#

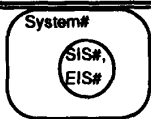
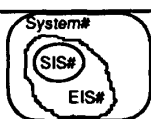
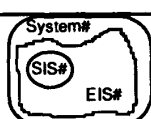
This area looks at the relationship of the EIS# with the System#. In general, we do not want the IA approach to estimate that everything is affected—that is, the EIS# is the same as the System#—unless that is indeed the case. The "distance" of the EIS# from the System# is a way to gauge the sharpness of the IA. Table 3-4 illustrates some

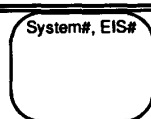
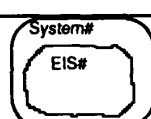
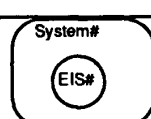
cases and interpretations.

3.4.2.3. EIS# and AIS#

The relationship between the EIS# and the AIS# is also meaningful. We want the AIS# to be contained regularly in the EIS#, and preferably very close or exactly the same. This would give us more confidence that IA approach results in estimated impacts that can be more carefully relied on to give the true scope of a change.

We do not want the AIS# greater than EIS#, since this means that the EIS# is less a reliable indicator of the true scope of a change. Table 3-5 illustrates some cases and

Case/Picture	Defining Conditions	Measurement When Present	Point Interpretation	Desired Trend
1 	$EIS# = SIS#;$ $EIS#, SIS# \subseteq$ System#	$ SIS\# / EIS\# = 1$	Best. Estimated impact restricted to SIS#.	Desired: $SIS# = EIS\#$ always. Expected: $SIS# = EIS\#$ are equal in 5% of a random sample of impact analyses Goal: $SIS# = EIS\#$ in 20% of a random sample
2 	$ EIS\# > SIS\# ;$ $SIS\# \subset EIS\#;$ $EIS#, SIS\# \subseteq$ System#	$K < SIS\# / EIS\# < 1$, for some user-selected K such that $0 < K < 1$	Expected. The estimated impacts are just a little more than the SIS#. "Little" is relative. We suggest $K = .7$ here.	Desired: $1 > SIS\# / EIS\# \geq .70$ never. Expected: $1 > SIS\# / EIS\# \geq .70$ in 40% of a random sample of impact analyses. Goal: $1 > SIS\# / EIS\# \geq .70$ in 60% of a random sample of impact analyses.
3 	$ EIS\# \gg SIS\# ,$ $SIS\# \subset EIS\#;$ $EIS#, SIS\# \subseteq$ System#	$ SIS\# / EIS\# < K$, where K is as in the preceding row	Not good. Big jump from SIS# to EIS# means a lot of things to check in the EIS#.	Desired: $ SIS\# / EIS\# < .70$ never. Expected: $ SIS\# / EIS\# < .70$ in 55% of a random sample of impact analyses. Goal: $ SIS\# / EIS\# < .70$ in 10% of a random sample of impact analyses.

Case/Picture	Defining Condition	Measurement When Present	Point Interpretation	Desired Trend
1 	$EIS# =$ System#	$ EIS\# / System\# = 1$	Default, not so helpful for impact analysis. But may indicate a system with extreme ripple effect.	Desired: $EIS# = System\#$ should never occur. Expected: $EIS# = System\#$ in 30% of a random sample of impact analyses. Goal: $EIS# = System\#$ in 5% of a random sample of impact analyses.
2 	$ System\# > EIS\# ;$ $EIS\# \subset$ System#	$J < EIS\# / System\# < 1$, for some user-selected J such that $0 < J < 1$	Better. Change estimated not to affect entire system.	Desired: $ EIS\# < System\# $ always. Expected: $ EIS\# < System\# $ in 50% of a random sample of impact analyses Goal: $ EIS\# < System\# $ in 70% of a random sample of impact analyses.
3 	$ System\# \gg EIS\# ;$ $EIS\# \subset$ System#	$ EIS\# / System\# < J$, where J is as in the preceding row	Even better. Change estimate is restricted to a relatively small subset of the system.	Desired: $ EIS\# \ll System\# $ always. Expected: $ EIS\# \ll System\# $ in 20% of a random sample of impact analyses. Goal: $ EIS\# \ll System\# $ in 25% of a random sample of impact analyses.

paragraphs, the *paragraph* containing the sentences is specified as the SIS# to the IA approach. The IA approach then does IA, resulting in an EIS# of five paragraphs (including the paragraph in the SIS#). Except for the SIS# paragraph, all four other paragraphs must be inspected, meaning 24 sentences must be inspected (6 sentences/paragraph in the figure). In contrast, if the granularity was at the level of sentences, then potentially only the two actually affected sentences (see figure) could have been found. Thus a finer granularity search would allow a potential search savings of 480% (= 24 predicted impacted sentences / (5 actually impacted sentences, including, in this example, the two sentences in the original SIS)).

Table 3-6 illustrates some possibilities with granularity. Similar comments and observations can be made between the relative granularities of the interface object model and the internal object model.

3.4.3. Summary Table

Table 3-7 summarizes the resulting framework elements from IA effectiveness.

4. Classify Systems According To Framework

To illustrate the use of, and provide some justification for, the IA framework, Table 4-1 uses the framework elements to compare five IA approaches. The first is a program slicer represented by the Surgeon's Assistant developed to investigate decomposition slicing as a software maintenance technique [Gallagher1991]. Decomposition here is effectively an impact analysis of the change at the code level. The second is a generic manual cross referencing that detect all references to a given software object (data field, disk file, flag, module, etc.) and assists in understanding relationships between software artifacts.

The third is a traceability system represented by the Automated Life Cycle IA System (ALICIA) [RAD1986]. ALICIA represents one of the first and most comprehensive attempts to address IA with automated traceability mechanisms. The

fourth is a Documenting System called Software Document Support (SODOS) environment [Horowitz1986]. It supports the development and maintenance of software documentation. Finally, a commercial tool called the Battlemat Analysis Tool™ (BAT) [McCabe1992] represents a control flow analyzer (among other capabilities). Control flow tools identify calling dependencies, logical decisions (conditions such as IF-THEN-ELSE, LOOPS, CASE statements, etc.), and other control information to examine control flow impacts.

We see two distinct approaches to IA here. The first type (represented by the program slicer, cross referencer, and control flow analyzer) is source code oriented and examines dependencies within the same artifact type. The second type (represented by ALICIA and SODOS) is life-cycle-document oriented and examines dependencies between differing artifact types. Generally speaking, the first type is more mature and provides a finer grained analysis of

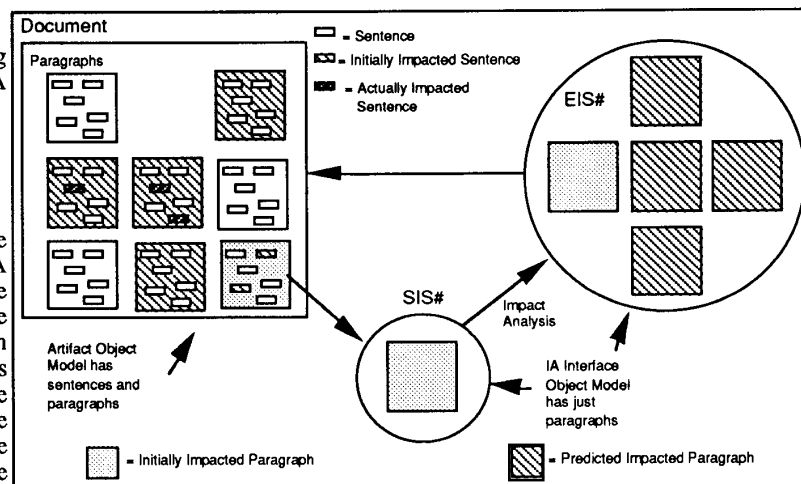


Figure 3-6. How granularity can influence the search for true impacts after impact analysis.

Table 3-6. Granularity Possibilities			
Case	Artifact Object Model	Interface Object Model	Comments
G1			Granularities are about equal. Potentially less work needed to translate impacts into the SIS#. Potentially less work needed to translate results from the EIS#.
G2			Artifact object model has finer granularity than the interface object model. Potentially more work in discovering true impacts (at artifact object model level) from those predicted in the EIS#.
G3			Artifact object model has coarser granularity than the interface object model. Potentially more work is needed to specify fine grained objects in the SIS#.

interpretations.

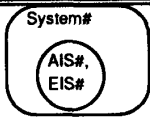



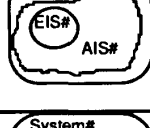
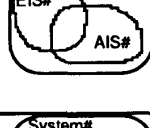

3.4.2.4. Granularity

In practice, the repeated translations from the artifact objects to interface and internal objects, and back, causes several kinds of problems. For example, there is the problem of predicting what artifact objects are actually affected from the EIS# (interface model objects). The translation from the artifact object model to the interface object model and back is usually manual.

model includes sentences and paragraphs as objects, but the interface object model includes only paragraphs as objects, then we must express a change to a sentence in the artifact model as an impacted paragraph in the interface object model. The resulting EIS# would then include impacted paragraphs. This means work is needed to look inside the impacted paragraphs to find the exact sentences that may be impacted. It also means many more spurious impacts would have to be looked at because the impact bandwidth of a paragraph is often bigger than that for a sentence.

Figure 3-6 illustrates this problem. If the artifact object

In this example, the SIS has two sentences. Because the IA interface object model can only express impacts with

Table 3-5. AIS# and EIS# Possibilities				
Case/Picture	Defining Conditions	Measurement When Present	Point Interpretation	Desired Trend
1 	$EIS\# = AIS\#;$ $EIS\# \subseteq$ System#	$ AIS\# / EIS\# = 1$	Best. Estimated impact set matches AIS#. If this happens regularly, usefulness of IA is substantially increased.	Desired: $ AIS\# = EIS\# $ always. Expected: $ AIS\# = EIS\# $ in 10% of a random sample of impact analyses Goal: $ AIS\# = EIS\# $ in 70% of a random sample of impact analyses
2 	$ EIS\# >$ $ AIS\# ;$ $AIS\# \subseteq EIS\#;$ $EIS\# \subseteq$ System#	$H < AIS\# / EIS\# <$ 1, for some user-selected H such that $0 < H < 1$	"Safe". The EIS# contains the AIS#, and the EIS# is not much bigger than the AIS#.	Desired: $ AIS\# < EIS\# $ never. Expected: $ AIS\# < EIS\# $ in 50% of a random sample of impact analyses Goal: $ AIS\# < EIS\# $ in 20% of a random sample of impact analyses
3 	$ EIS\# \gg$ $ AIS\# ;$ $AIS\# \subseteq EIS\#;$ $EIS\# \subseteq$ System#	$ AIS\# / EIS\# <$ H, where H is as in the preceding row	Safe, but not so good. Big jump from AIS# to EIS# means a lot of things to check in EIS# before arriving at the AIS#.	Desired: $ AIS\# \ll EIS\# $ never. Expected: $ AIS\# \ll EIS\# $ in 40% of a random sample of impact analyses Goal: $ AIS\# \ll EIS\# $ in 10% of a random sample of impact analyses
4 	$ AIS\# >$ $ EIS\# ;$ $EIS\# \subseteq AIS\#;$ $EIS\# \subseteq$ System#	$M < EIS\# / AIS\# <$ 1, for some user-selected M such that $0 < M < 1$	Expected. IA approximates, and falls short of, what needs to be changed.	Desired: $ EIS\# < AIS\# $ never. Expected: $ EIS\# < AIS\# $ in 60% of a random sample of impact analyses Goal: $ EIS\# < AIS\# $, in 20% of a random sample of impact analyses
5 	$ AIS\# \gg$ $ EIS\# ;$ $EIS\# \subseteq AIS\#;$ $EIS\#, SIS\# \subseteq$ System#	$ EIS\# / AIS\# <$ M, where M is as in the preceding row	Not so good. Big jump from EIS# to AIS# means extra work to discover AIS#.	Desired: $ EIS\# \ll AIS\# $ never. Expected: $ EIS\# \ll AIS\# $ in 30% of a random sample of impact analyses Goal: $ EIS\# \ll AIS\# $ in 10% of a random sample of impact analyses
6 	$ AIS\# \cap EIS\# >$ 0, $AIS\# \neq EIS\#;$ $EIS\# \subseteq$ System#	$ EIS\# \cap AIS\# >$ 0	Not so good. Extra work to check EIS# objects that aren't in AIS#. Extra work to discover objects in AIS# not in EIS#	Desired: $ EIS\# \cap AIS\# $ near 1 most of a random sample of impact analyses. Expected: $.7 < EIS\# \cap AIS\# < 1$, in 60% of a random sample of impact analyses Goal: $.9 < EIS\# \cap AIS\# < 1$, in 80% of a random sample of impact analyses
7 	$ AIS\# \cap EIS\# =$ 0; $EIS\# \subseteq$ System#	$ EIS\# \cap AIS\# = 0$	Not so good. A worse version of case 6.	Desired: $ EIS\# \cap AIS\# = 0$ never Expected: $ EIS\# \cap AIS\# = 0$ in 20% of a random sample of impact analyses Goal: $ EIS\# \cap AIS\# = 0$ in 5% of a random sample of impact analyses

impacts while the second type is less mature, but provides a broader analysis.

5. Relation to Other Work

This paper presents a variety of concepts relating IA technology. We are aware of other work for characterizing change, but have found no applicable evaluation criteria for IA technology.

In one characterization of change, Madhavji describes a broad perspective on change with respect to people, policies, laws, resources, processes, and results [Madhavji1991]. Madhavji distinguishes changes to described items from changes to the environment that houses these items. The Prism Dependency Structure facility supports describing items and their inter-dependencies as well as identifying possible effects of changes. The Prism Change Structure facility supports classifying, recording, and analyzing change-related data from a qualitative perspective.

Our work differs from Madhavji's in that his work focuses on the change process while ours focuses on IA

Element	Explanation	Desired IA Effectiveness Trends
SIS and EIS	What trend is observed in the relative size of the SIS and EIS when the approach is applied to typical problems? We would like the EIS to be as close as possible to the SIS.	$ SIS / EIS = 1$, (i.e., $SIS = EIS$), or nearly 1
EIS and System#	What trend is observed in the relative size of the EIS and System# when the approach is applied to typical problems? We would like the EIS to be much smaller than the System#.	$ EIS / System\# \leq N$, where N is some small tolerance level
EIS and AIS#	What trend is observed in the relative size of the EIS and AIS# when the approach is applied to typical problems? We would like the EIS to contain the AIS#, and the AIS# to equal to or smaller than the EIS.	$ EIS / AIS\# = 1$, (i.e., $EIS = AIS\#$), or nearly 1
Granularity	What is the relative granularity of the artifact object model vs. the interface object model? We would like the granularities to match, if possible.	G1, granularities match. (It is even better if granularities are fine enough too.)

technology and a characterization of IA applications, parts, and effectiveness criteria.

6. Conclusions

IA has many meanings in industry today and the trend is to use the term for more and more situations. In this paper, we have attempted to bring some order and structure to the discussion of IA technology. We recognize the need for a framework to compare IA approaches. This paper presented a definition of IA and a framework that delineates clearly the basis for comparing IA capabilities. The framework

Table 4-1a. Framework Category: IA Application						
IA Approach	Artifact Object Model	Decomposition	Change specification	Results specification	Interpretation of EIS# to Determine AIS	Other Features
Program Slicer - Surgeon's Assistant [Gallagher1991]	Programs & entities within them	Yes, with some semantics. Knows about programming language objects, control, & data flow relationships.	Yes, with some analysis. User must specify slice criteria & initially affected program, variables, etc	Browsing. Can browse the sliced pieces of the program.	Some effort. Slice approach may compute the slice for the programmer.	Testing, Browsing
Manual Cross Referencing, based on name id & cross reference listings.	Programs, predefined documents	No. Just passively identifies text strings according to match criteria.	No, not applied. User manually reviews cross reference listings	Report. Cross reference listing is the report	Significant. Much effort needed to locate secondary dependencies.	None.
Traceability System - Automated Life Cycle Impact Analysis System (ALICIA) [RADCI1986]	DOD-STD-2167 documents & entities within them	Yes, with little semantics. software life cycle objects (SLOs) stored in a RDBMS with a schema that reflects the relationships.	Yes, with detailed analysis. Uses method. Requirements for a change are analyzed for tracing SLOs.	Report & Database View. ALICIA provides impact report & navigation through a view.	Significant. SLOs are not elaborated nor explained.	Methodology, Document-oriented database schema
Documenting System - Software Document Support (SODOS) environment [Horowitz1986]	Predefined document set & ASCII text.	Yes, with some semantics. Stores decomposed SLOs in a RDBMS based on an object model & predefined relationships.	Yes, with detailed analysis. Req's for change are incorporated in B-spec, then analyzed for tracing to SLOs.	Report, Browsing & Database View. Results of IA in hypertext graph & view for browsing & report.	Significant. SODOS for document management. SLOs not elaborated nor explained thus the user must interpret results	Query - supports traceability with both predefined & user defined relationship Browsing - navigation/edit support.
Control Flow Analyzer - Battlemat Analysis Tool (BAT) [McCabe1992]	Source code programs	Yes, with little semantics. These are limited to decision logic & calling structures.	No, not applied. Control flow analyzers do not support change specification.	Report, Browsing, & Database View. Results stored as a graph & output in a report or editor.	Significant. Control flow impacts not explained by the tool. Interpretation is left to the user.	Call graphs Complexity Metrics Browsing/ editing Execution path slicer

IA Approach	Interface Object Model	Internal Object Model	Decomposition	Impact Model	Tracing/ Impact Approach	Repository	Load, Modify, Browse
Program Slicer - Surgeon's Assistant	Program objects such as variables, statements, etc	Data define-use graph & control flow graph	Similar to compiler	Slicing based on data & control flow dependencies	Decomposition slice, program slicing	File system	All three are available
Manual Cross Referencing	Character Strings, i.e., representing variables	None. Just matches characters	None	Matching between character strings	Pattern matching	File system	Not available
Traceability System - ALICIA	DOD-STD-2167 doc's (req's, design, code, test, etc.) & input templates	Meta-schema based on DOD-STD-2167 objects	Documents decomposed into relational database	Has user-defined & predefined relationships that have dependency info.	Based on traceability relationships. Heuristic & stochastic impact search algorithms	RDBMS	All three available
Documenting System - SODOS	Documents based on predefined object-based software life cycle	Manages SLOs using object-based model & hypermedia graph. Meta-schema based on predefined document & management objects.	Decomposes documents into object model using templates & filters	Object configuration management & navigation. User-defined & predefined relationships that have dependency information	Based on user-defined & predefined traceability relationships. This enables consistency checking of traceability relationships	RDBMS - Smalltalk-80 object-oriented front-end to RDBMS	All three available
Control Flow Analyzer - BAT	Program objects such as variables, statements, etc.	Control flow graph, calling hierarchies, module structure	Decomposes the code into its control flow elements	Based on control flow dependencies	Not explicit - identifies changes associated with control flow	File system	All three available

IA Approach	SIS and EIS	EIS and System#	EIS and AIS#	Granularity
Program Slicer - Surgeon's Assistant	Could not determine from available sources.	Could not determine from available sources.	Could not determine from available sources.	Interface object model is finer-grained. Impacts between programs must be expressed in terms of subprogram entities.
Manual Cross Referencing	Could not determine from available sources.	Could not determine from available sources.	Could not determine from available sources.	Interface object model is finer grained. Impacts between documents must be re-expressed as items that are cross-referenced.
Traceability System - ALICIA	Could not determine from available sources.	Could not determine from available sources.	Could not determine from available sources.	Interface object model is more coarse grained. Impacts between documents must be re-expressed in impact model.
Documenting System - SODOS	Could not determine from available sources.	Could not determine from available sources.	Could not determine from available sources.	Interface object model is more coarse grained. Impacts between documents must be re-expressed in impact model.
Control Flow Analyzer - BAT	Could not determine from available sources.	Could not determine from available sources.	Could not determine from available sources.	Interface object model is finer grained. Impacts between programs must be expressed in terms of subprogram entities.

consists of three parts: IA Application, IA Parts, and IA Effectiveness. To demonstrate the use of this framework, we classified five existing IA technologies according to their criteria.

We believe this work will be helpful to those desiring to investigate the functionality of existing IA approaches. For those hoping to compare approaches, the framework should provide plenty of useful differentiators.

Bibliography

[Freedman1981] Freedman, D. P. and G. M. Weinberg, "A Checklist for Potential Side Effects of a Maintenance Change," In G. Parikh, *Techniques of Program and System Maintenance*, 1981, page(s) 93 - 100.

[Gallagher1991] Gallagher, K.B. and J.R. Lyle, "Using Program Slicing in Software Maintenance," *IEEE Transactions on Software Engineering*, Volume 17, Number 8, August 1991, page(s) 751 - 761.

[Horowitz1986] Horowitz, E. and R. Williamson, "SODOS: A Software Document Support Environment - Its Definition," *IEEE Transactions on Software Engineering*, Volume SE-12, Number 8, August 1986, page(s) 849 - 859.

[IEEE1983] IEEE, "Glossary of Software Engineering Terminology," Std. 729-1983, In IEEE, *Software Engineering Standards*, Third Edition, New York: IEEE, 1989., 1983.

[Madhavji1991] Madhavji, Nazim H., "The Prism Model of Changes,"

Proceedings of the International Conference on Software Engineering, IEEE Computer Society Press, 1991, page(s) 166 - 177.

[McCabe1992] McCabe & Associates, Inc., "Batlimap Analysis Tool Reference Manual," McCabe & Associates, Inc., Twin Knolls Professional Park, 5501 Twin Knolls Road, Columbia, MD, December 1992.

[Pfleeger1991] Pfleeger, Shari L., *Software Engineering: The Production of Quality Software*, New York, Macmillan Publishing Co., 1991.

[RADC1986] Rome Air Development Center, "Automated Life Cycle Impact Analysis System," RADC-TR-86-197, Rome Air Development Center, Air Force Systems Command, Griffiss Air Force Base, Rome, NY, December 1986.

[Stevens1974] Stevens, W., G. Meyers, and L. Constantine, "Structured Design," *IBM Systems Journal*, Volume 13, Number 2, 1974.

[Yau1978] Yau, S.S., J.S. Collofello, and T.M. MacGregor, "Ripple Effect Analysis of Software Maintenance," *Proc. of COMPSAC*, Washington, DC: IEEE Computer Society Press, 1978, page(s) 60 - 65.

[Yau1980] Yau, S.S. and J. Collofello, "Some Stability Measures for Software Maintenance," *IEEE Transactions on Software Engineering*, Volume SE-6, Number 6, November 1980, page(s) 545 - 552.