Impact Analysis - Basics and Process

Idaho State University | Computer Science

Isaac Griffith

CS 4423 and CS 5523
Department of Computer Science
Idaho State University

ROAR

Outcomes

After today's lecture you will:

- Be able to understand and describe the general idea of impact analysis
- Be able to understand and describe the process of impact analysis
- Be able to understand and describe the basic measures used in impact analysis

Impact Analysis

CS 4423/5523

ROAR

General Idea

- The maintenance process is started by performing impact analysis.
- Impact analysis basically means identifying the components that are impacted by the Change Request (CR).
- Impact of the changes are analyzed for the following reasons:
  - to estimate the cost of executing the change request.
  - to determine whether some critical portions of the system are going to be impacted due to the requested change.
  - to record the history of change related information for future evaluation of changes.
  - to understand how items of change are related to the structure of the software.
  - to determine the portions of the software that need to be subjected to regression testing after a change is effected.

General Idea

- Implementation of change requests impact all kinds of artifacts, including the source code, requirements, design documentation, and test scenarios.
- Therefore, impact analysis traceability information can be used in performing impact analysis.
- Gotel and Finkkelstein define traceability as the ability to describe and follow the life of an artifact in both the forward and backward directions.
- Bohner and Arnold define traceability as the ability to trace between software artifacts generated and modified during the software product life cycle.

General Idea

- Thus, **traceability** helps software developers understand the relationships among all the software artifacts in a project.
- There are two broad kinds of traceability:
  1. **horizontal** (external) traceability.
  2. **vertical** (internal) traceability.
- **Traceability** of artifacts between different models is known as external traceability, whereas internal traceability refers to tracing dependent artifacts within the same model.

General Idea

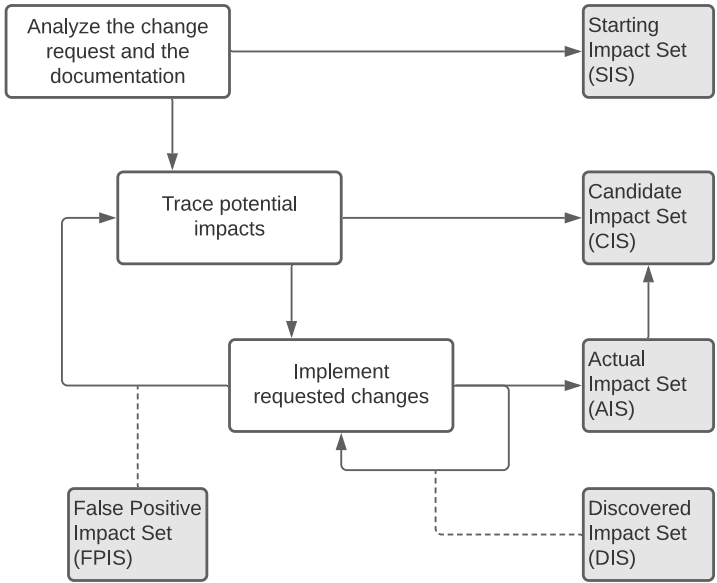
- A topic related to impact analysis is ripple effect analysis.
- Ripple effect means that a modification to a single variable may require several parts of the software system to be modified.
- Analysis of ripple effect reveals what and where changes are occurring.
- Measurement of ripple effects can provide the following information about an evolving software systems:
  1. between successive versions of the same system, measurement of ripple effect will tell us how the software's complexity has changed.
  2. when a new module is added to the system, measurement of ripple effect on the system will tell us how the software's complexity has changed because of the addition of the new module.

Impact Analysis Process

CS 4423/5523

ROAR

Impact Analysis Process



Impact Analysis Process

- **Starting Impact Set (SIS):** The initial set of objects (or components) presumed to be impacted by a software CR is called SIS.
- **Candidate Impact Set (CIS):** The set of objects (or components) estimated to be impacted according to a certain impact analysis approach is called CIS.
- **Discovered Impact Set (DIS):** DIS is defined as the set of new objects (or components), not contained in CIS, discovered to be impacted while implementing a CR.
- **Actual Impact Set (AIS):** The set of objects (or components) actually changed as a result of performing a CR is denoted by AIS.
- **False Positive Impact Set (FPIS):** $FPIS$ is defined as the set of objects (or components) estimated to be impacted by an implementation of a CR but not actually impacted by the CR. Precisely, $FPIS = (CIS \cup DIS) \setminus AIS$
- where $\cup$ denotes set union and \ denotes set difference.
- In the process of impact analysis it is important to minimize the differences between AIS and CIS, by eliminating false positives and identifying true impacts.

Idaho State University
Computer Science

Impact Analysis Process

Two traditional information retrieval metrics:

- **Recall:** It represents the fraction of actual impacts contained in CIS, and it is computed as the ratio of $|CIS \cap AIS|$ to $|AIS|$. The value of recall is 1 when DIS is empty.
- **Precision:** It represents the fraction of candidate impacts that are actually impacted, and it is computed as the ratio of $|CIS \cap AIS|$ to $|CIS|$.
  - For an empty $FPIS$ set, the value of precision is 1.
- Note that if AIS is equal to CIS, both **recall** and **precision** are computed to be equal to 1.
- **Adequacy** and **effectiveness** are two key aspects of any impact analysis approach.

ROAR

Adequacy

- Adequacy of an impact analysis approach is the ability of the approach to identify all the affected elements to be modified. Ideally, $AIS \subseteq CIS$.
- Adequacy is represented in terms of a performance metric called inclusiveness, as follows:

$$Inclusiveness = \begin{cases} 1 & \text{if } AIS \subseteq CIS \\ 0 & \text{otherwise} \end{cases}$$

Adequacy

- The concept of adequacy is essential to assessing the quality of an impact analysis approach.
- An inadequate approach is in fact useless, as it provides the maintenance engineer with incorrect information.
- For example, if inclusiveness is 0, the approach cannot be used because it does not provide the maintenance engineer with all the components to be analyzed.

Effectiveness

- The ability of an impact analysis technique to generate results, that actually benefit the maintenance tasks, is known as its effectiveness.
- Effectiveness is expressed in terms of three fine grained characteristics as follows:
  - Ripple-sensitivity
  - Sharpness
  - Adherence

Ripple-sensitivity

- **Ripple-sensitivity** implies producing results that are influenced by ripple effect.
- The set of objects that are directly affected by the change is denoted by $DISO$ (directly impacted set of objects).
- Similarly, the set of objects that are indirectly impacted by the change is denoted by $IISO$ (indirectly impacted set of objects).
- The cardinality of $IISO$ is an indicator of ripple effect.
- The software maintenance personnel expect that the cardinality of $IISO$ is not far from the cardinality of $DISO$.

Amplification

- Amplification, as defined below, is used as a measure of Ripple-sensitivity.

$$Amplification = |IISO|/|DISO| \rightarrow 1$$

- where $|\cdot|$ denotes the cardinality operator.

Sharpness

- Sharpness is the ability of an impact analysis approach to avoid having to include objects in the CIS that are not needed to be changed.
- Sharpness is expressed by means of Change Rate as defined below:

$$ChangeRate = |CIS|/|System|$$

- $ChangeRate$ falls in the range from 0 to 1.

Adherence

- Adherence is the ability of the approach to produce a CIS which is as close to AIS as possible.
- A small difference between CIS and AIS means that a small number of candidate objects fail to be included in the actual modification set.
- Adherence is expressed by $S - Ratio$ as follows:

$$S - Ratio = |AIS|/|CIS|$$

- $S - Ratio$ takes on values in the range from 0 to 1.

Starting Impact Set (SIS)

- Impact analysis begins with identifying the SIS.
- The CR specification, documentation, and source code are analyzed to find the SIS.
- It takes more efforts to map a new CR's "concepts" onto source code components (or objects).
- In the "concept assignment problem," one discovers human oriented concepts and assigns them to their realization.
- It is difficult to fully automate the concept assignment problem because programs and concepts do not occur at identical levels of abstractions, thereby necessitating human interactions.

Identifying the SIS

- There are several methods to identify concepts, or features, in source code.
- The “grep” pattern matching utility available on most Unix systems and similar search tools are commonly used by programmers.
- However, the tool has some deficiencies:
  - it is based on the correspondence between the name for the concept assigned by the programmer and an identifier in the code.
- The technique often fails when the concepts are hidden in the source code, or when the programmer fails to guess the program identifiers.

Identifying the SIS

- The software reconnaissance methodology proposed by Wilde and Scully is based on the idea that some programming concepts are selectable, because their execution depends on a specific input sequence.
- Selectable program concepts are known as features. By executing a program twice, one can often find the source code implementing the features:
  1. execute the program once with a feature and once without the feature.
  2. mark portions of the source code that were executed the first time but not the second time.
  3. the marked code are likely to be in or close to the code implementing the feature.

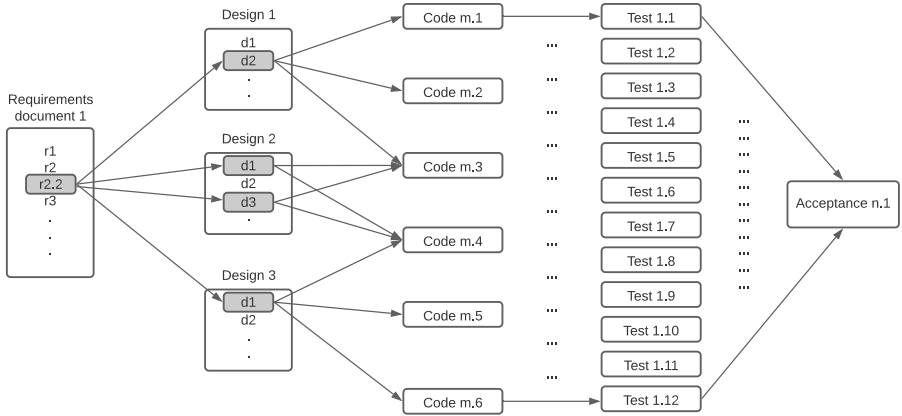
Identifying the SIS

- Chen and Rajlich proposed a dependency graph based feature location method for C programs.
- The component dependency graph is searched, generally beginning at the main().
- Functions are chosen one at a time for a visit.
- The maintenance personnel reads the documentation, code, and dependency graph to comprehend the component before deciding if the component is related to the feature under consideration.

Idaho State University
Computer Science

Analysis of Traceability Graph

- The graph shows the horizontal traceability of the system.
- The graph that is so constructed reveals the relationships among work products.

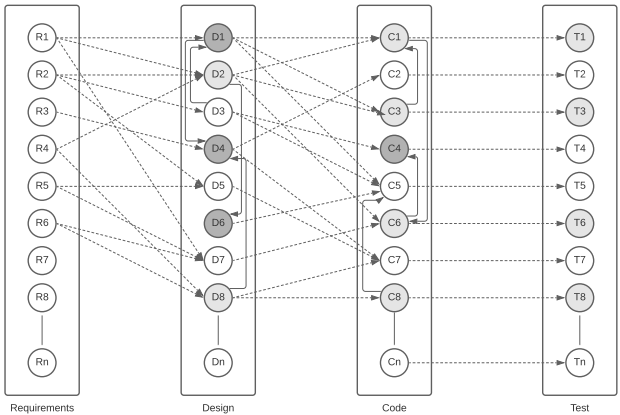


ROAR

Idaho State University
Computer Science

Analysis of Traceability Graph

- The graph has four categories of nodes: requirements, design, code, and test.
- The edges within a silo represent vertical traceability for the kind of work product represented by the silo.

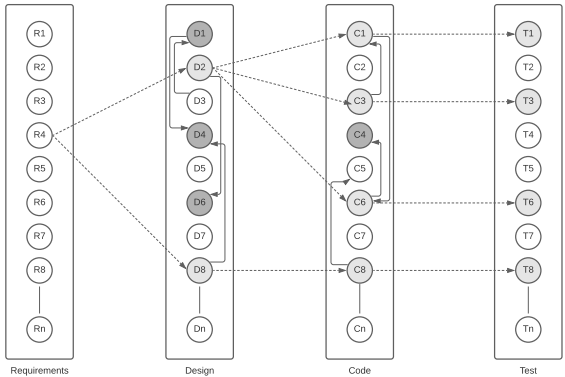


ROAR

Idaho State University
Computer Science

Analysis of Traceability Graph

- If some changes are made to requirement object “R4,” the results of horizontal traceability and vertical traceability are shown
- The horizontally traced objects have been shown as lightly shaded circles, whereas the vertically traced objects have darkly shaded circles.



ROAR

Analysis of Traceability Graph

- For a node i in a graph, its in-degree in(i) counts the number of edges for which i is the destination node, and in(i) denotes the number of nodes having a direct impact on i.
- Similarly, the out-degree of node i, denoted by out(i), is the number of edges for which i is the source.
- Node i being changed, out(i) is a measure of the number of nodes which are likely to be modified.
- A common measure of complexity of a graph is the well-known Cyclomatic complexity.

Analysis of Traceability Graph

- In addition, node count is a measure of size.
- It may be noted that vertical traceability metrics are product metrics – and those metrics reflect the effect of change on each product.
- To minimize the impact of a change, out-degrees of nodes need to be made small.
- For nodes with large out-degrees, one may partition the nodes to uniformly allocate dependencies across multiple nodes.
- Low in-degrees of nodes are an indication of a good design.

Idaho State University
Computer Science

Analysis of Traceability Graph

- Process metrics are useful in examining horizontal traceability.
- To understand changes in horizontal traceability, it is necessary to understand:
  - the relationships among the work products.
  - how work products relate to the process as a whole.
- There exist three graphs:
  - the first one relating requirements to software design.
  - the second one relating software design to source code.
  - the third one relating source code to tests.
- Next, size and complexity metrics are obtained for those three graphs to know about the work products and the effects of changes.
- One might conclude that if a proposed change results in increased size or complexity of the relationship between a pair of work products, the resulting system will be more difficult to maintain.

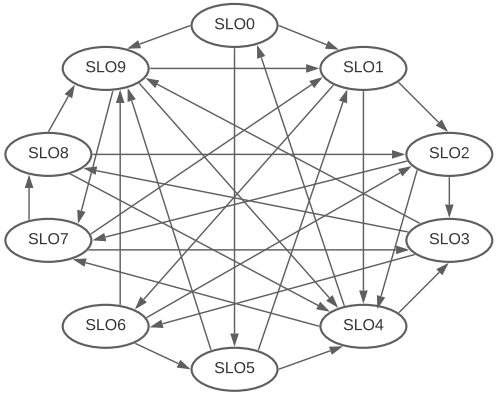
Identifying the CIS

- A CIS is identified in the next step of the impact analysis process.
- The SIS is augmented with software lifecycle objects (SLOs) that are likely to change because of changes in the elements of the SIS.
- Changes in one part of the software system may have direct impacts or indirect impacts on other parts.
- Both direct impact and indirect impact are explained in the following.
  - **Direct impact:** A direct impact relation exists between two entities, if the two entities are related by a fan-in and/or fan-out relation.
  - **Indirect impact:** If an entity A directly impacts another entity B and B directly impacts a third entity C, then we can say that A indirectly impacts C.

Idaho State University
Computer Science

Identifying the CIS

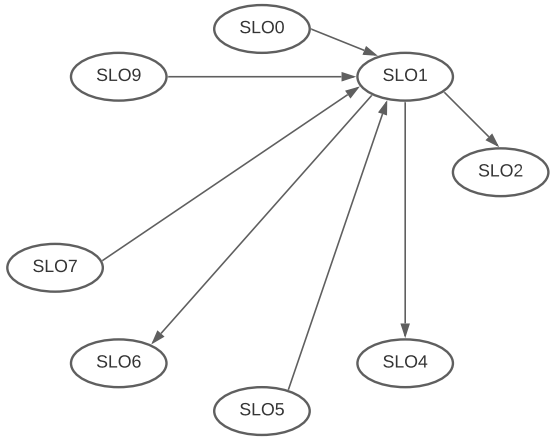
- Let us consider the directed graph, shown, with ten SLOs.
- Each SLO represents a software artifact connected to other artifacts.
- Dependencies among SLOs are represented by arrows.
- In the figure, SLO1 has an indirect impact from SLO8 and a direct impact from SLO9.
- The in-degree of a node i reflects the number of known nodes that depend on i.



ROAR

Identifying the CIS

- The figure shows the four nodes – SLO0, SLO5, SLO7 and SLO9 – that are dependent on SLO1, and the in-degree of SLO1 is four.
- In addition, the out-degree of SLO1 is three.



Identifying the Candidate Impact Set

- The connectivity matrix of Table 6.1 is constructed by considering the SLOs and the relationships shown in Figure 6.5.
- A reachability graph can be easily obtained from a connectivity matrix.
- A reachability graph shows the entities that can be impacted by a modification to a SLO, and there is a likelihood of over-estimation.

SLO	0	1	2	3	4	5	6	7	8	9
0		X				X				X
1			X		X		X			
2				X	X			X		
3							X		X	X
4	X			X				X		
5		X			X					X
6			X			X				X
7		X		X			X		X	
8			X		X					X
9		X			X			X		

Identifying the Candidate Impact Set

- The dense reachability matrix of Table 6.2 has the risk of over-estimating the CIS.
- To minimize the occurrences of false positives, one might consider the following two approaches:
  - Distance based approach.
  - Incremental approach.

SLO	0	1	2	3	4	5	6	7	8	9
0		X	X	X	X	X	X	X	X	X
1	X		X	X	X	X	X	X	X	X
2	X	X		X	X	X	X	X	X	X
3	X	X	X		X	X	X	X	X	X
4	X	X	X	X		X	X	X	X	X
5	X	X	X	X	X		X	X	X	X
6	X	X	X	X	X	X		X	X	X
7	X	X	X	X	X	X	X		X	X
8	X	X	X	X	X	X	X	X		X
9	X	X	X	X	X	X	X	X	X	

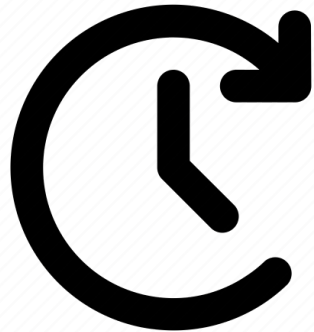
Identifying the Candidate Impact Set

- **Distance based approach:** In this approach, SLOs which are farther than a threshold distance from SLO i are not be considered to be impacted by changes in SLO i.
- **Incremental approach:** In this approach, the CIS is incrementally constructed. For every SLO in the SIS, one considers all the SLOs interacting with it, and only SLOs that are actually impacted by the change request are put in the CIS.

SLO	0	1	2	3	4	5	6	7	8	9
0		1	2	3	2	1	2	2	3	1
1	2		1	2	1	2	1	2	3	2
2	2	2		1	1	3	2	1	2	2
3	3	2	2		2	2	1	2	1	1
4	1	2	3	1		2	2	1	2	2
5	2	1	2	2	1		2	2	3	1
6	3	2	1	2	2	1		2	3	1
7	3	1	2	1	2	3	2		1	2
8	3	2	1	2	1	3	3	2		1
9	2	1	2	2	1	3	2	1	2	

For Next Time

- Review EVO Chapter 6.1 - 6.3
- Read EVO Chapter 6.4 - 6.6
- Watch Lecture 15



Are there any questions?