

Traceability for impact analysis & co-evolution

Done By,
Cynthia Aludogbu
Bhargav Bhatt

Topics :

Summary

Approaches

Conclusion

Discussion

Summary :

Software change impact analysis is necessary to be aware of effects caused by the change.

Traceability management can be classified into different approaches in the context of impact analysis.

The challenges of traceability management includes traceability recovery and link evolution.

Requirement Evolution: The goal of this work is to support the maintainer in keeping the requirements specification up-to date when software systems evolve.

Traceability and its dimensions :

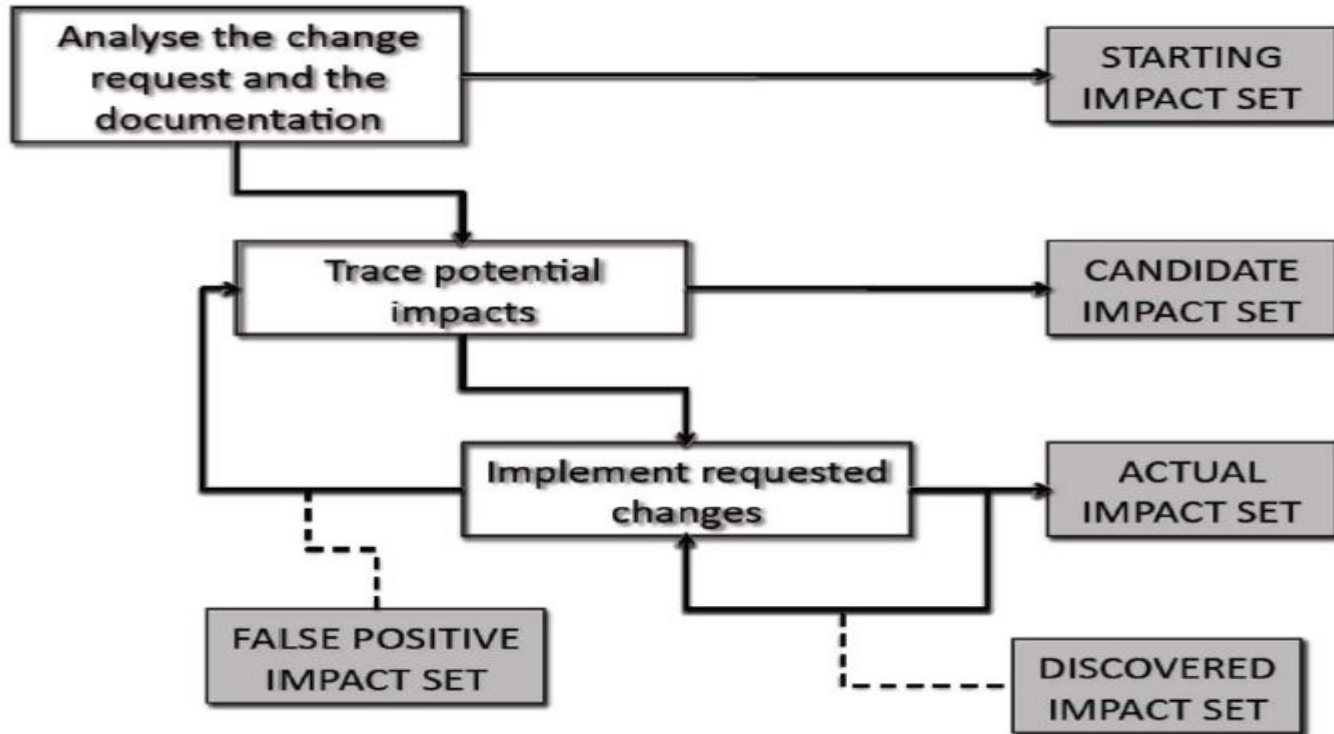
Definition : -

The ability to describe and follow the life of an artefact, in both forward and backward direction.

3 Dimensions : -

- Vertical vs Horizontal Traceability
- Structured vs Knowledge Based Traceability
- Implicit vs Explicit Traceability

Impact process analysis :



Traceability challenges : Traceability recovery

Several methods have been proposed to recover traceability links between software artefacts of different types.

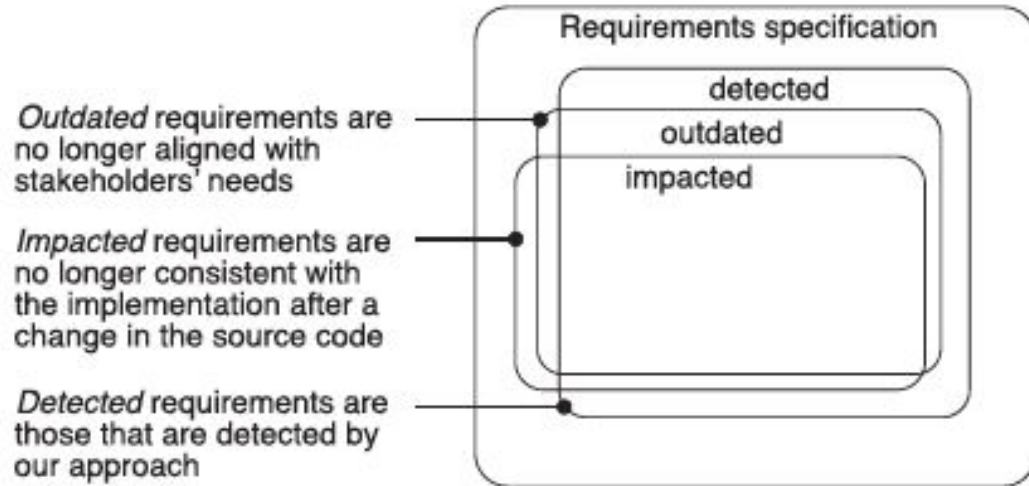
- Heuristic-based approaches
- IR-based approaches
- Data mining-based approaches

Traceability challenges : Link evolution

Another important traceability management issue concerns the evolution of traceability links

- ArchEvol
- Molhado
- A method based on string edit distance to assess the compliance of OO software systems across different releases.

Classification of requirements :



Remarks:

We assume a process where outdated and impacted requirements overlap to a large extent.

The size of the intersection of detected and impacted requirements determines the effectiveness of our approach.

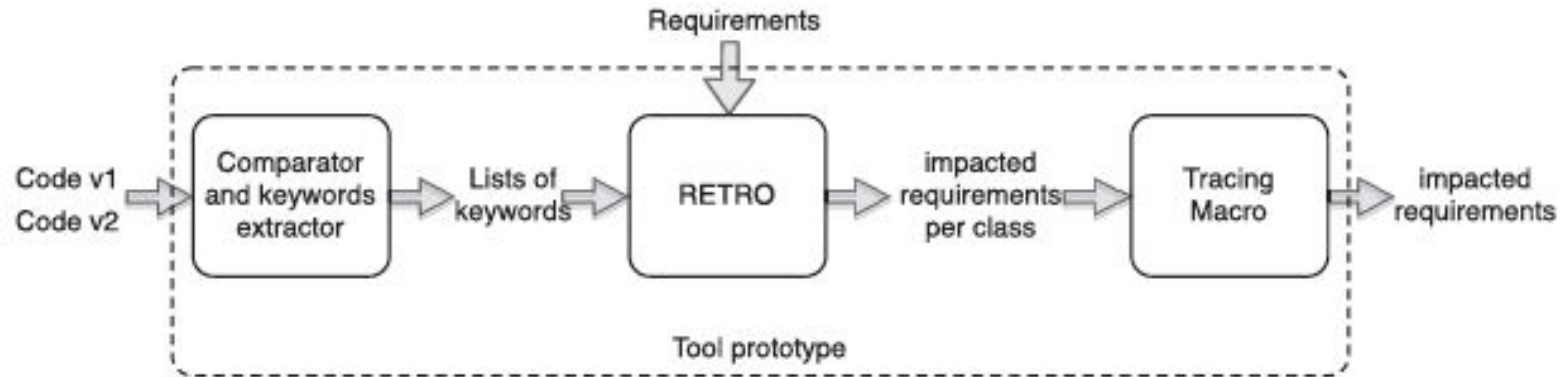
Tracing changes to the requirements :

Changed element	Names	Documentation	Call hierarchy
Package	Package, sub-classes	Package	No
Class	Class, sub-methods, sub-fields	Class	Yes
Method	Method, parent class	Method, parent class	Yes
Field	Field, parent class	Parent class	No

In order to increase the precision of the tracing. For extracting keywords, we consider three sources of information: the names of the elements impacted by the change; the call hierarchy of the changed elements; and the documentation of the changed elements.

By call hierarchy, we mean all the methods/classes that are invoking the changed method or those invoking the constructor of the changed class.

Protocol tool set for Requirement Evolution :



Discussion :

How do you think that airplane manufacturers, like Airbus and Boeing identify relevant and irrelevant changes?

Questions :

What is post-requirements traceability used for?

- Link requirements to design and implementation, documenting responsibility assignment, compliance verification, or impact analysis of a requirement.

How can traceability support the evolution of software systems and their documentation?

- Providing awareness of necessary changes imposed by a change request

Conclusion:

The identification of the set of artefacts that would be impacted as a consequence of processing a change request, is based on domain expert knowledge of the system and artefact traceability.

The given approach for Evolution can be used when an initial requirements specification exists but is not kept up-to-date because of time and cost constraints. For maintainers who think that updating requirements is only a waste of time and does not bring any benefits, our approach is useless.

The current approach detects the requirements that are impacted by the code changes, but it does not detect missing requirements in the specification.

Our approach is expected to work much better for code that is well commented than for code that is uncommented or badly commented.