## 1. Topic

**An Evidence-based Approach to Discovery and Assessment of Software Testing Strategies**

## 2. Introduction

Evidence means information that supports a statement. It is gathered whenever a proposition is to be proved. Evidence thus, in specific terms can be defined as "Evidence for a proposition is anything that increases the estimate of the probability of the truthfulness of the proposition."[1] A proposition here is any assumption related to the real world. Evidences are gathered for all available propositions and the best available research evidence is used to draw inferences. To bridge the gap between actual and theoretical decisions, the decisions need to be backed by sufficient evidences. Based upon the real life facts and happenings factual data is collected then inferences are drawn from the data which happens to be the key parameter for any decisions being taken. Evidence based approach [EBA] is followed for the same. The evidence, by itself, does not make any decisions, but it can help support the decision making process. In EBA a conscientious and judicious use of the best available evidence before taking any decision is done. It integrates individual expertise with the best available external practicing approach through systematic research. Thus a lot of thought process is being carried out before reaching to any conclusion. Along with the scientific studies to support the three ways of evidences gathering are: wisdom, expert opinion, and common sense.

EBA is carried out in three steps:

1. It takes into account practical expertise.

2. Need of the system to be developed is evaluated.

3. Best research practices are incorporated into the study or process of decision making.

The full integration of these three components into decisions enhances the opportunity for optimal outcome.

EBA draws its roots from the field of medicine wherein a doctor works on the theory of elimination. He/She gathers evidence from his/her patients and clinical researches and uses these pieces of knowledge for further treatment of patients. While questioning a patient(s) he/she keeps on eliminating those ailments from his/her mind which do not have particular set of symptoms. Thus he/she draws conclusion only after he/she has examined the patient thoroughly and through previous expertise they eliminate all non considerable diseases. The three steps stated earlier are followed in medicinal practice as shown in The EBM Triad Fig 1.and all the other fields that uses EBA also follow the same procedure.

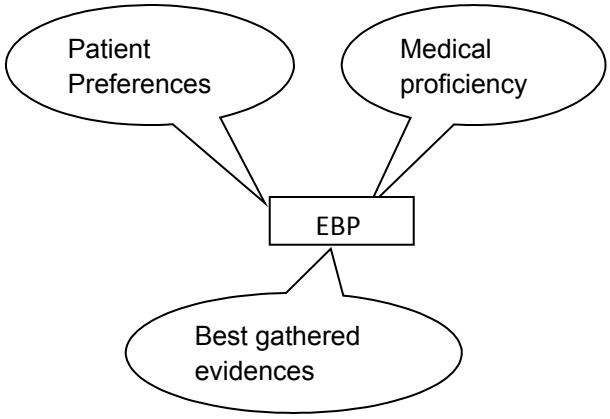


Fig 1. The EBM Triad

It started in medicine as evidence-based medicine (EBM) and is now being used in other fields such as nursing, psychology, education, library and information science also. Its basic principles are that all practical decisions made should,

1) Be based on research studies and

2) That these research studies are selected and interpreted according to some specific norms and characteristics for Evidence Based Practice [EBP].

Research indicates that expert opinion based medical advice is not as reliable as advice based on the accumulation of results from scientific experiments [2]. Along with its applications in medicine Evidence based approach has its roots in management as well. The managerial decisions and organizational policies are formed using the best available scientific evidences. Like its counterparts in medicine, management and education EBM is a widely used tool in speech pathology. In speech pathology the practitioners consider the factors relating to individuals as per their need and values but they also refer to the clinical skills that are underpinned by knowledge gathered through published research and evidences collected over the years.

Since the rise of computing in the 1940s, the applications and uses of computers have grown at exponential rate and software plays a central role in almost all aspects of computing. The number, size, and application domains of computer programs have grown dramatically and so is the need for software engineering. “Software engineering is that form of engineering that applies the principles of Computer Science and Mathematics to achieving cost-effective solutions to software problems. It is the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software [IEEE 1990]” [3]. It is obvious that software engineering is required to create high quality software in an efficient and systematic manner.

Resultantly nowadays more emphasis is laid on analysis, evaluation, specification, design, and evolution of software. In today's era the fast pace of change in computer and software technology needs new and improvised software products. This situation has created customer expectations and competitive forces that strain the developer's ability to produce quality of software within acceptable development schedules.

Evidence based approach is best suited to software engineering as a thorough documentation for every decision taken is done. It can help software developers in following a non conventional development method rather than using the conventional models. For doing so, Software project managers' decisions should be based on solid evidence rather than on common wisdom and EBSE is a methodical and scientific manner upon which managers and practitioners can base their decisions, by forming rational arguments from the evidence they have gathered through a combination of experience and related research. In this approach a unit or a part of software is examined and then gradually proceeded to the whole system. It is possible that EBSE can provide the mechanisms needed to assist practitioners to adopt appropriate technologies and to avoid inappropriate technologies. It is therefore suggested that the goal of evidence-based software engineering (EBSE) should be:

*"to provide the means by which current best evidence from research can be integrated with practical experience and human values in the decision making process regarding the development and maintenance of software"* [4]

Keeping in view the approach followed in medical sciences we compare its Evidence based applicability with that in the field of Software Engineering in the following table 1 [4]:

**Table 1. Five steps used in Evidence-based Medicine and (by analogy) in Evidence-based Software Engineering.**

| Step | Evidence-based Medicine | Evidence-based Software Engineering |
|---|---|---|
| 1 | Converting the need for information (about prevention, diagnosis, prognosis, therapy, causation, etc) into an answerable question. | Converting the need for information (about development and maintenance methods, management procedures etc.) into an answerable question. |
| 2 | Tracking down the best evidence with which to answer that question. | Tracking down the best evidence with which to answer that question. |
| 3 | Critically appraising that evidence for its validity (closeness to the truth), impact (size of the effect), and applicability (usefulness in our clinical practice). | Critically appraising that evidence for its validity (closeness to the truth), impact (size of the effect), and applicability (usefulness in software development practice). |
| 4 | Integrating the critical appraisal with our clinical expertise and with our patient's unique biology, values and circumstances. | Integrating the critical appraisal with our software engineering expertise and with our stakeholders' values and circumstances. |
| 5 | Evaluating our effectiveness and efficiency in executing Steps 1-4 and seeking ways to improve them both for next time. | Evaluating our effectiveness and efficiency in executing Steps 1-4 and seeking ways to improve them both for next time. |

In most cases software is built with technologies for which the developer has insufficient evidence to confirm their suitability, limits, qualities, costs, and inherent risks. Thus the practitioners can have difficulty in making judicious decisions about whether to adopt a new technology because in most of the projects there is little objective evidence to confirm the suitability risks. This in turn can lead to poor decisions about technology adoption, as some researchers suggest that:

*"Software practitioners and managers seeking to improve the quality of their software development processes often adopt new technologies without sufficient evidence that they will be effective, while other technologies are ignored despite the evidence that they most probably will be useful."* [5]

It is therefore required that conclusions drawn for any practice should be based on evidences supported by research. For generating proper evidences rigorous evaluation needs to be done which requires systematic, standardized description of target problems. Evidence base is thus built by summing up observations, careful description and measurement of problem and careful citing of specific results.

### 2.1. Evidence gathering tools

In the evidence-based approach controlled trials are conducted, to get the strongest possible evidence to support a hypothesis. This means that appropriate high quality experimental results are given more weight than those to whom studies judged to be of a low quality. In software engineering domain, the study designs conducted for evidence gathering are [6]:

a) Systematic reviews b) Meta-analyses

c) Mapping studies d) Narrative synthesis techniques

As its name suggests, a **systematic review** requires a systematic and methodical search of the literature in order to present an overall synthesis of results from the highest-quality studies. It is a scientific research method helping the researchers to gather all evidences available on a particular topic and its main objective is to accomplish it in an unbiased manner. **Evidence** is the cumulative data obtained from **'primary' studies** that is judged to be relevant to answer a particular research question. The task of aggregating data to answer a research question is performed through a **secondary study** [7]**.** A rigorous 'secondary' study is thus required to find all relevant data which provides a base for determining what to include and what to exclude. The latest experimental finding reported in Conference Proceedings and Journal papers is the primary data. The appropriateness of the experimental method used in the experiment marks the quality of it.

In **meta-analysis** one has to take the results of several related studies and pool the data, with the effect of creating one large study which can then be analyzed. It combines the results of several studies that

address a common set of research hypotheses. Both systematic reviews and meta-analyses are important techniques for data collection and analysis. They are not considered as empirical scientific studies in themselves but they synthesize the entire body of empirical work that has been done on a topic.

**Mapping study** or systematic mapping study [SMS] is also done for collecting evidences. This technique is used when researchers have little evidence on a particular field. In many researches it is also used as a tool to enhance SLR based inferences.

**Narrative' synthesis** is another approach followed in systematic review when findings include multiple studies that are based on the use of words and text to summarize and explain the findings of the synthesis. This method adopts a textual approach which tells the crux of all the findings to a process of synthesis focusing on a wide range of questions.

### 2.2. EBA in Software Engineering

A direct research conducted to identify the requirements of the software industry, helps practitioners to make more rational decisions about technology adoption, enable better choice of development technologies and increase the acceptability of software intensive systems that interface well with the users. The procedure followed in Evidence based approach can be summarised as shown in Fig 2:

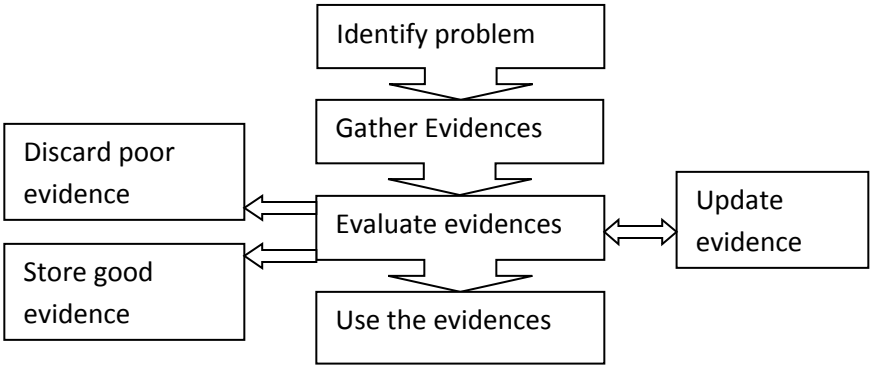


Fig 2: Working with evidences

This approach can be used in software engineering since many a times software developers adopt a new technology because of their novelty or anecdotal evidence. In Evidence-based software engineering (EBSE) all the preliminary requirements of the preliminary phases of Software Development Life Cycle (SDLC) are fulfilled, here all the experiences are properly documented in order to inform software professionals to adopt a particular decision. In EBSE, the study factor may be the technology of interest. The technological specifications should be very detailed and not at a very high level of abstraction that is the software lifecycle and all the design methods should be properly read and documented and only then should the engineer collect evidences on it and design the software generation model or identify or

evaluate various strategies for the phases in a SDLC. Use of EBSE in the majority of relevant empirical studies, for example OO methods, Agile methods, or Cost estimation methods can be accomplished. To implement this following steps of behavioral practices are followed, they are shown in Fig 3:

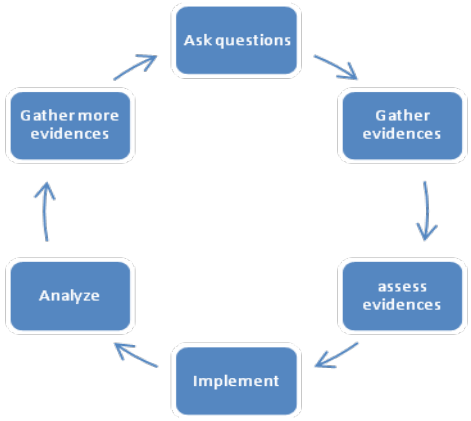


Fig 3. Steps of behavioral practice

1. Convert a relevant problem or information need into an answerable question.
2. Search the literature for the best available evidence to answer the question.
3. Critically appraise the evidence for its validity (closeness to the truth), impact (size of the effect) and applicability.
4. Integrate the appraised evidence with practical experience and the customer's values and circumstances to make decisions about practice.
5. Evaluate performance and seek ways to improve it.
6. Gather further evidences if required.

The first three steps require a systematic review of the literature, conducted in order to provide a balanced and objective summary that is relevant to meeting a particular need for information. Evidence-based approaches [EBA] demand that, among other things, practitioners systematically track down the best evidence relating to some practice, critically appraise that evidence for validity, impact, and applicability; and carefully document it.

The EBA thus, may be applied at all the stages of SDLC, with its applicability having been studied for maintenance [8], cost and effort estimation [9], testing and test case generation [10]. Feasibility analysis is evidence based as well, thus the researches done reflect that evidence based approach is a viable

technique for software engineering practices and a lot of motivation is drawn from the research papers on using it as a tool for gathering data on test cases and testing strategies.

### 2.3. EBA in Software Testing

Testing is an important mechanism both to identify defects and assure that completed products work as specified. Software testing is a fundamental component of software quality assurance and represents a review of specification, design and coding. Even though extensive research has been done in the Testing field, it is necessary to assess the current state of research and practice, in order to provide practitioners with evidence that enable fostering its further development.

To do testing in an effective manner it is required to design appropriate test cases followed by testing strategies questions which should be formulated in an answerable manner, and then the search for relevant evidence shall begin. Evidence Based Practice utilizes the latest testing techniques available through literature and on-line testing related literature search strategies.

Thus, the research problem may be stated as: How can evidence or empirical based data help researchers in evaluating testing techniques in SE? What should be considered in order to gather relevant evidences? Based on these questions, researchers try to investigate the use of evidence as a tool to conduct robust empirical studies in SE. Mainly, the research topics investigated through the use of evidence has been identified as: Software product line testing strategies, Software test case generation, and Generating test beds.

#### 2.3.1 Software product line testing strategy

A software product line (SPL) is a set of software intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way. A software product line is a unified representation of a set of conceptually similar software systems that share many common features and satisfy the requirements of a particular domain. In SPL, the best practices and techniques in Software Engineering will be articulated and applied. Functional testing and variability testing with UML models and use cases are the most popular topics.

#### 2.3.2 Software test case generation

Software testing is an essential yet expensive activity in software development, a lot of effort goes in automation of testing as much as possible. Thus case studies are picked from the literature available and are implemented on software. To get a better picture of the current practice in evaluations in software

engineering research, the researchers survey the literature on test generation for a specific type of software and generate test cases [11].

### 2.3.3 Generating test beds

It becomes a bit costly to test a software extensively thus test beds are generated which are an alternative to offer a good set of empirical tools to evaluate technologies in controlled environments.

## 3. Review of literature

In order to meet the objectives of the research various research papers are studied. The papers were focused on the different approaches to gather evidences and to find feasible testing strategies for a software product line. In the course of evaluating testing strategies empirical investigation of test cases are also carried out. The research papers referred lead path for fulfilling the objectives of this research.

Tore Dybå et al.[12] suggest that EBSE aims to improve decision making related to software development and maintenance by incorporating current best evidence from research with practical experience and human values. EBSE provides mechanisms to support various parts of Software Process Improvement. In particular, EBSE focuses on finding and appraising an appropriate technology for its suitability in a particular situation. There are several information sources that can be used. For example, getting viewpoints from the customers or the software's users, asking a colleague or an expert, using own experience, or search for research-based evidence. In research-based evidence reports, articles, and other documents that describe a study conducted and reported according to certain guidelines are included. The researchers here have used only the textual material available which does not prove to be sufficient enough.

Data sources like the Internet, Open forums and electronic databases should also be used for gathering data and seeking evidences. A large amount of data can be collected through these sources and it will then be classified under various heads and then will it be used for inference drawing.

Sackett *et al*.[13] recommend that individual doctors review the way in which they practice and teach Evidence Based Medicine in order to improve their individual performance. For EBSE, this would involve propagating successful technologies throughout a company and preventing the spread of technologies that are unsuccessful. In this paper adequate research was done in the field of medicine and the author suggests the same practices to be carried out in Software Engineering where engineers should first evaluate his decision based upon the evidences gathered and then bring it into practice. Adoption of new technology before the commencement of a new project should be based on the evidences gathered.

Though it is feasible in the field of medical science since individual doctors are responsible for reporting unanticipated side-effects of drugs, but when compared to software engineering in a competitive industry, barely any company will assist their competitors by reporting good and bad experiences with new technologies. Thus evidences should be gathered using the available literature and data gathered through primary and secondary data collection tools.

Juristo et al [14] paper on Unit Testing Techniques states that testing is an invariable part of all systems engineering project. The human efforts that are spent on testing sometimes even exceed half the project total cost. Inadequate testing has been the source of many systems failures. Thus if test cases are designed keeping in view the real world practices then analysis of these can produce useful insights. Apart from mitigating defects in programs, testing is performed in peripheral areas such as performance of software, reliability and security. Testing does not simply focuses on the technology alone but on socio-technical issues such as acceptability, usability and fitness of software. It focuses on the whole systems rather than software in isolation. The authors did not attempt to create a new definition of testing in this paper, but rather to navigate this fuzzy concept by looking at what testing is and how it is meaningful from a practitioner point of view. For this study, the authors chose to use the definitions provided in the IEEE's Software Engineering Body of Knowledge—usually referred to as the Swebok. Juristo et al. refined and extended the Swebok classification system to include more detailed categories for code coverage and test set classification. The source of evidence however was confined to papers published in the IEEE and ACM electronic databases.

Wasif Afzal et al [15] use Search-based software engineering (SBSE) as a tool for the application of optimization techniques in solving software engineering problems. This paper discussed the applicability of optimization techniques in solving software engineering problems. The growing interest in evidence based or search based search technique can be attributed to the fact that generation of software tests is generally considered as an undeniable problem, since there are many possible combinations of a program's input. The researchers here performed a SLR and came to the conclusion that metaheuristic search techniques have been applied for non-functional testing of execution time, quality of service, security, usability and safety. A variety of metaheuristic search techniques were found to be applicable for non-functional testing including simulated annealing, tabu search, genetic algorithms, ant colony methods, grammatical evolution, genetic programming and swarm intelligence methods.

Martin Johansen et al [16] suggested that evidence gathering is most suitable for software products which cater to similar type of software requirements. These type of softwares share a considerable amount of code and testing products individually is redundant for product lines. Since the products in a product line share a large amount of code, it should be possible to improve testing by utilizing the specification of the

similarities and differences between the products. A strategy for testing a software product line is a description of which steps a test engineering should follow to reduce the effort on testing the product line significantly below the effort required to test each product delivered to a customer.

Sarvnaz Karimi and Falk Scholer [17] credit systematic review as the key tool used in evidence based policy. The SLR synthesize available research on the topic of investigation. Though many researches hold their base by searching the literature but a SLR is carried out to agreed standards: using clear protocols in carrying out the process, focusing on specific questions, identifying as much of the relevant literature as possible, critically appraising the quality of the research included in the review; synthesizing research findings from included studies; being as objective as possible to remove bias, and, updating the review so that it remains relevant.

Mostafa Kandil[18] et al draw a line between testing done in software and testing a web application. The functional requirements of a web application are different thus separate testing strategies should be followed to give a proper coverage to WA's functional requirements. Due to the heterogeneous nature and different quality criteria of Web Applications, its components and user expectations, new demands emerge for testing of those systems to ensure a high reliability level. The most important aspect of WA's is that it deals with large numbers of users, clients and stakeholders around the world. Many WA's require high quality security checks such as banking systems, governmental, ticketing system, e-commerce etc. The researchers here only contribute in WA's and GUI applications, which can be scaled for other aspects of web applications as well like data flow, user acceptability and so on.

Experiments carried out by Ciupa [19] et al show that the relative number of faults which are detected through random testing is certain that is their numbers can be predicted, but different runs of the random test case generator detect different faults. The experiments carried out suggest that to find faults quickly random testing is supposedly the best suitable technique. The nature of faults was also evaluated here and different strategies were applied in the data sets. The researchers reached to the conclusion that number of faults can be predicted and they also vary from one technique to another but the nature of faults detected are more or less the same in all techniques.

To reduce the cost of testing software it is advisable to automate the testing process as much as possible, and automating test case generation is an important phase of this automation. A possible strategy in the automation of test case generation is the application of metaheuristic search (MHS) algorithms.

Considerable amount of research has been carried out in MHS algorithms and there is still a large potential in this area as search based testing proves to be a cost effective and workable option in testing field [20].

To get more test cases a mapping study can be done by the researchers and they will have to systematically classify the research papers and from them conclusions can be drawn. Evidence based datasets are also generated which can also be referred.

There is a severe decoupling between research in the computing field and the state of the practice in the field. The gap between empirical software testing strategies and software testing practice might be lessened if more attention were paid to two important aspects of evidence. The first is that evidence from case or field studies of actual software testing practice are essential in order to understand and inform that practice. The second is that the nature of evidence should fit the purpose to which the evidence is going to be put.

Thus a study of all the mentioned papers indicates that there is a lot of potential in the field of software testing using the EBSE approach. The research focuses on the use of evidence based approach for testing which includes test case generation and also determination of the best approach for testing using various techniques. The main intention of this research proposal is primarily on testing strategies.

## 4. Motivation

After extensive literature search, evidence based software engineering approach appears to be a viable alternative for many software testing practices. It is a very current and innovative approach which is being followed and is able to eliminate some of the defects that the software developers are facing.

For instance testing each and every product individually is a lot time and effort consuming task since similar kinds of product share a considerable amount of code. If a testing strategy is backed up with empirics and evidences it can result in reusable component testing. Testing all the products based upon the empirics in isolation will reduce the effort required on testing the whole product line compared to testing each product that is delivered to a customer from scratch, to achieve the same defect detection level.

Research indicates that testing being a cumbersome as well as integral part of software development when combined with EBSE leads to more optimum results.

## 5. Objectives

The primary objectives of the proposed research include:

- To find efficient software testing strategies for specific problem domains, using evidence based approach.
- To apply effective evidence gathering techniques for compiling evidence on application of software testing strategies.

- To analyze the gathered evidence so as to classify the testing strategies on the basis of applicability and types of testing.
- To perform a systematic review of the application and Empirical investigation of test cases in the specific software domain.
- To evaluate the performance of the identified strategies by applying them to relevant software domains.
- To evaluate and validate the findings by implementing the identified strategies vis–a–vis other testing approaches.

  Along with all the evidence gathering tools Literature review will be the primary source of evidence gathering backed up by hands on experience on testing softwares. Besides these testing strategies applied on open source softwares will also be taken into account.

## 6. Plan of work and Methodology

By reading relevant research papers it is being concluded that for evidence based approach the most feasible approach is systematic literature review, mapping studies and evidence gathering as they provide the researcher an unbiased review. In due course of time Narrative synthesis and Meta analysis tools will also be explored and the optimum evidence gathering tool will be selected for further collection.

The process of **Systematic Literature Review** is widely used to aggregate the results of primary studies. Conducting such a review involves an exhaustive search of the relevant literature, selection and data extraction. As shown in Fig 4 [7] all of this is conducted in accordance with a **review protocol**, which is composed ahead of performing the study and specifies in detail how the study is to be performed and the reasoning behind any choices.

In analyzing the available evidence the notion of '**best available evidence**' is used. This means that appropriate high quality experimental results are given more weight than those studies judged to be of a low quality [21].

It is suggested to follow a review protocol, consisting of several steps. Where, in order to gather the evidences of applicability of testing techniques in a software product line use, a metaheuristic search technique is used for searching the literature, for which we suggest the following research questions:

Q 1. In which testing areas have evidence based techniques been applied?

Q 2. What are the current challenges or limitations in the application of these techniques for testing?

Once the research questions have been identified the next step is generation of search strategy followed by Study selection criteria and procedures for including and excluding primary studies. Quality data can be used to devise a detailed inclusion/exclusion criterion and/or to assist data analysis and synthesis.

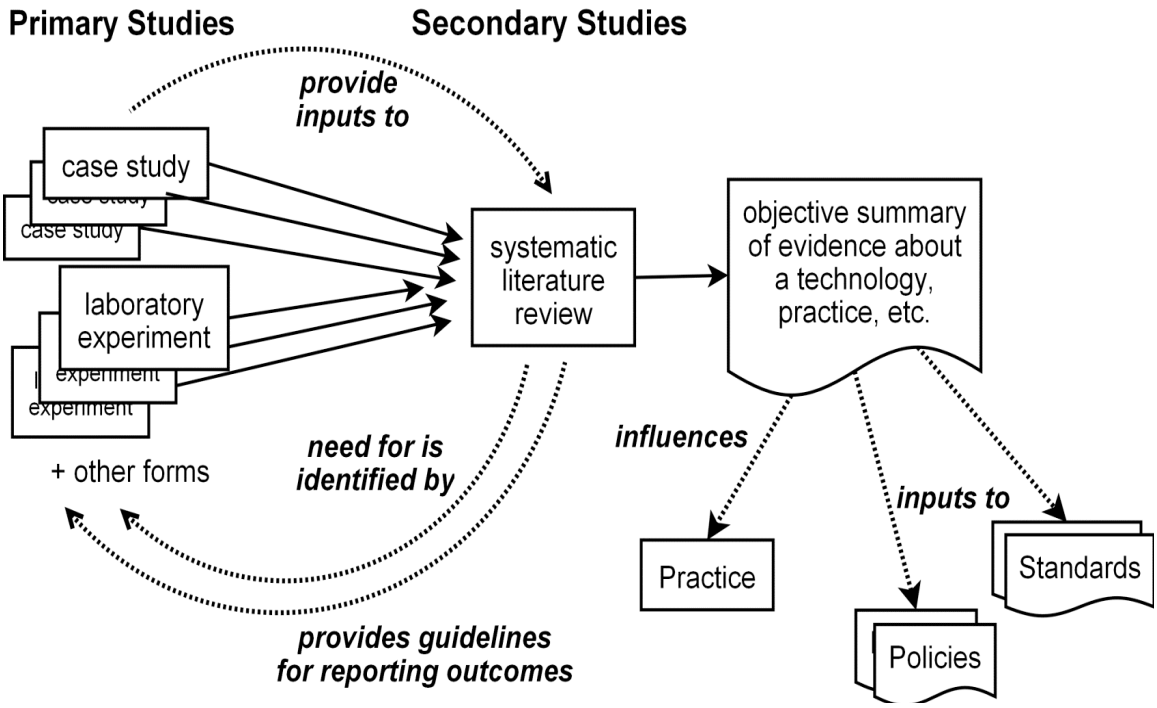


Fig 4. Working of SLR

The result of a review is an objective summary of evidence about the technology or practice that was studied. The results of Systematic Literature Reviews can be used to inform, practice, policy, standards and to identify where new primary research is needed. Using evidence to make decisions about the adoption of technology and techniques ensures that those decisions are justified and can be shown to be based on sound research findings. Fig 5 [22] represents the methodology which is to be followed.

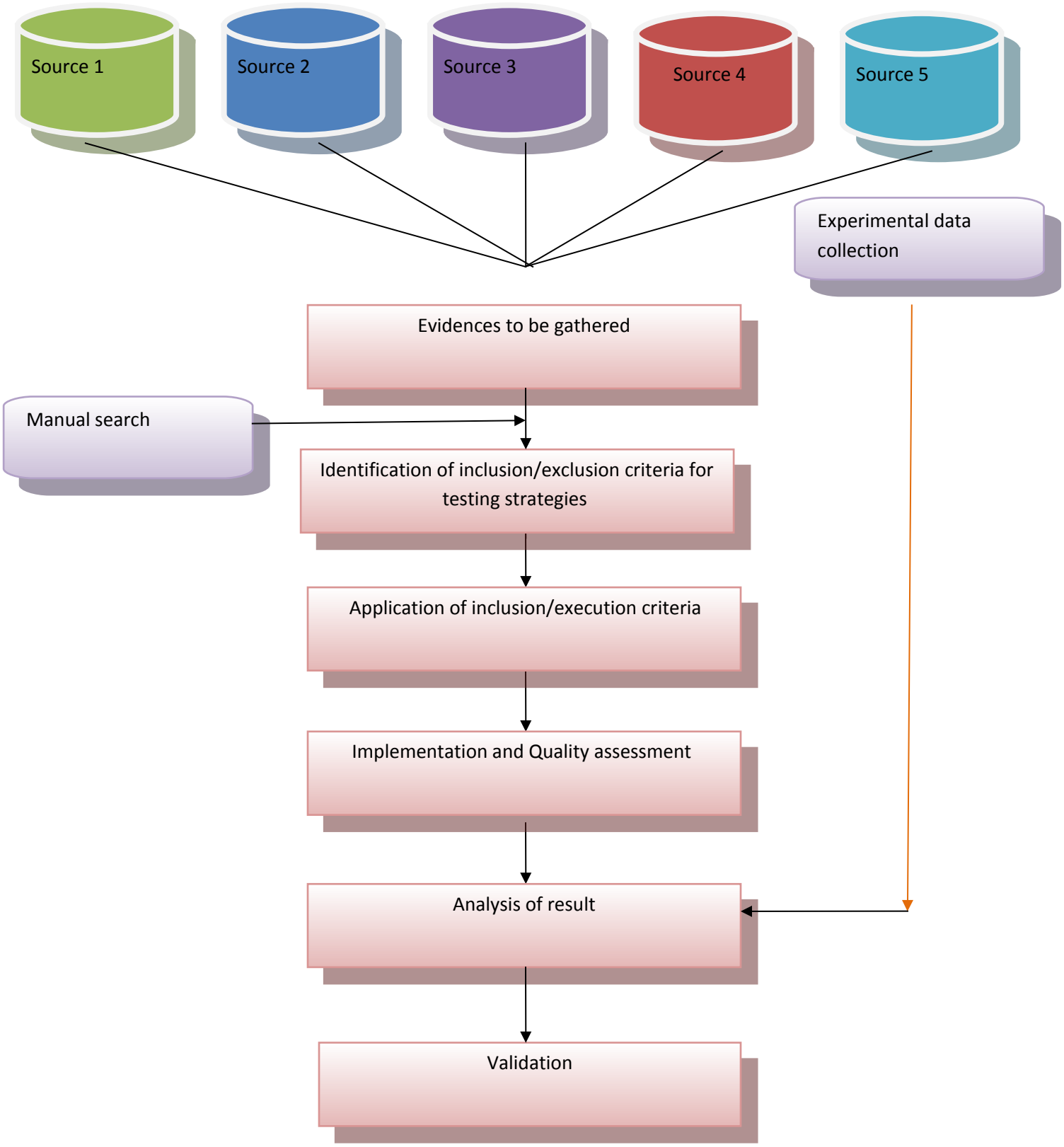


Fig 5 Methodology

**Work Plan:**

The research plan is divided into following seven phases:

| Phase | Activity | Approx time (in months) |
|---|---|---|
| 1. | Literature survey on evidence based software engineering. And its classification | 8-12 |
| 2. | Gathering primary data of software projects and Searching empirical databases | 4-5 |
| 3. | Refining and implementing search strategies for system properties | 3-5 |
| 4. | Assessment of drawn inferences from the proposed study | 3-4 |
| 5. | Develop new / refine existing methods for improvement in testing strategy. | 2-3 |
| 6. | Implementation, comparison with existing techniques and drawing conclusions | 4-5 |
| 7. | Paper/ Thesis Writing | 3-5 |

The timeline chart for research plan is as follow:

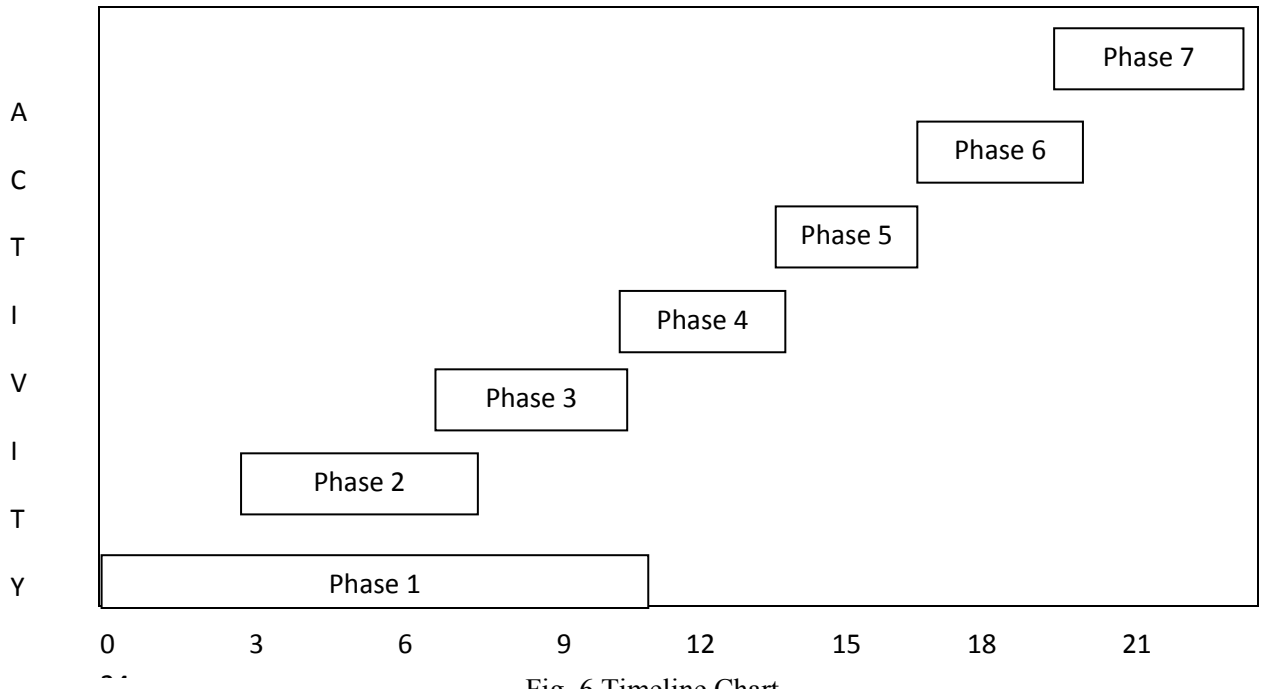


Fig. 6 Timeline Chart

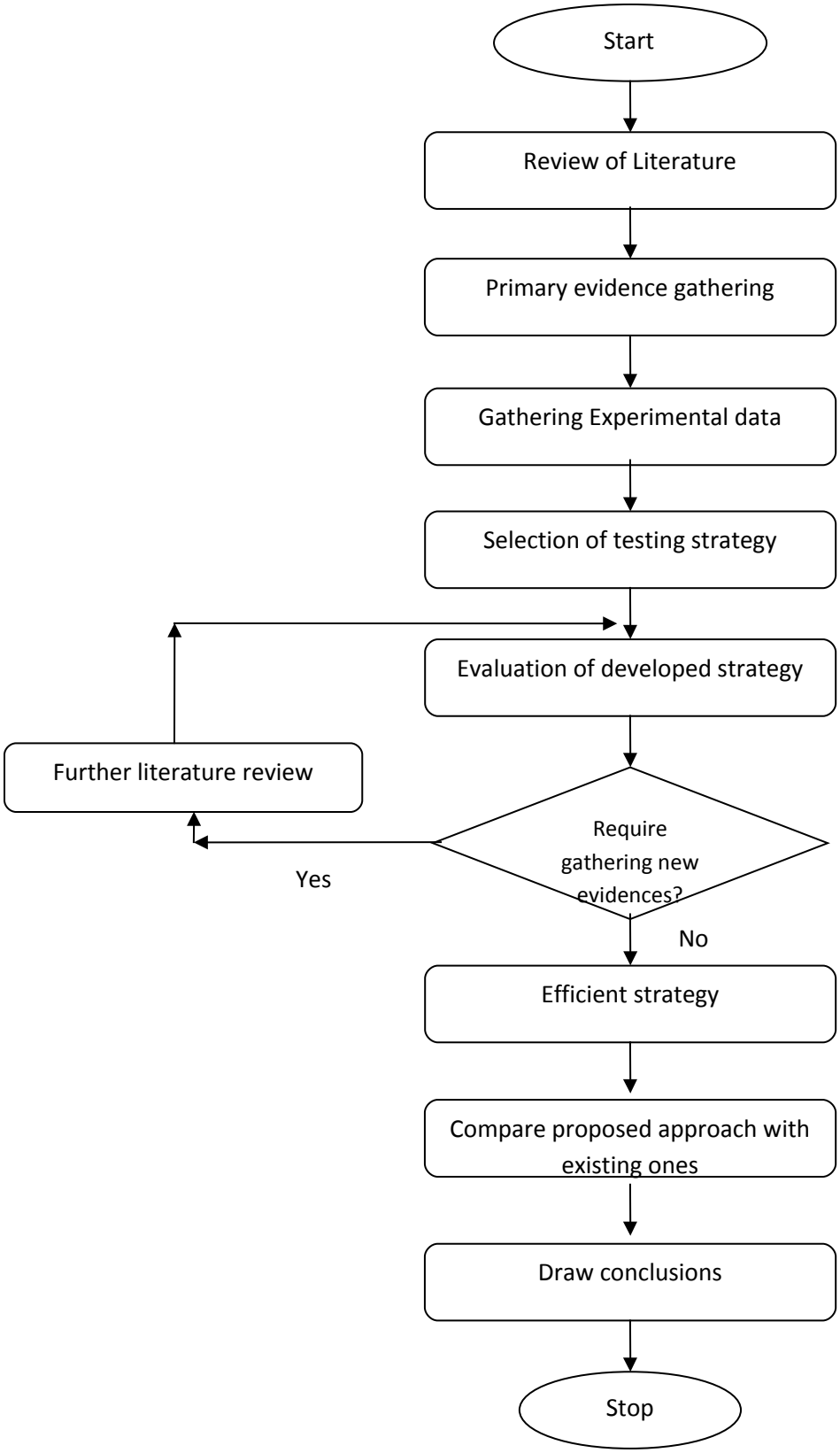


Fig.7 Working Methodology

REFERENCES


1. *"Evidence". Knowledge Management Dictionary. World Health Organisation.*
2. *Sackett, D.L., Straus, S.E., Richardson, W.S., Rosenberg, W., and Haynes, R.B. Evidence-Based Medicine: How to Practice and Teach EBM, Second Edition, Churchill Livingstone: Edinburgh, 2000*
3. *www.agile.csc.ncsu.edu*
4. *B.A. Kitchenham, T. Dybå, M. Jørgensen, Evidence-based software engineering, in: Proceedings of the 26th International Conference on Software Engineering, (ICSE'04), IEEE Computer Society, Washington DC, USA, pp. 273–281, 2004.*
5. *M.V. Zelkowitz, D.R. Wallace, and D.W. Binkley, "Experimental Validation of New Software Technology," Lecture Notes on Empirical Software Engineering, World Scientific, pp. 229–263, 2003.*
6. *Means, Barbara and Toyama, Yuki and Murphy, Robert and Bakia, Marianne and Jones, Karla Evaluation of Evidence-Based Practices in Online Learning: A Meta-Analysis and Review of Online Learning Studies, 2009.*
7. TheCaseForEvidence, dur.ac.uk/ebse, 2008.
8. *Mehwish Riaz, Emilia Mendes, Ewan Tempero , Towards Predicting Maintainability for Relational Database-Driven Software Applications: Extended Evidence from Software Practitioners, International Journal of Software Engineering and Its Applications Vol. 5 No. 2, April, 2011*
9. *Wille, C. Fiegler, A., Neumann, R., Dumke, R.R., Evidence-Based Evaluation of Effort Estimation Methods, Joint Conference of the 21st Int'l Workshop on and 6th Int'l Conference on Software Process and Product Measurement, 2011*
10. *Ali, S, Briand, L.C., Hemmati, H., Panesar-Walawege, R.K. A Systematic Review of the Application and Empirical Investigation of Search-Based Test Case Generation , Software Engineering, IEEE Transactions 2010.*
11. *Gordon Fraser, Andrea Arcuri, Sound Empirical Evidence in Software Testing. ICSE 2012.*
12. *B.A. Kitchenham, T. Dybå, M. Jørgensen, Evidence-based software engineering: Proceedings of the 26th International Conference on Software Engineering, (ICSE'04), IEEE Computer Society, Washington DC, USA, pp. 273–281, 2004.*
13. *Sackett, D.L., Straus, S.E., Richardson, W.S., Rosenberg, W., And Haynes, R.B. Evidence-Based Medicine: How to Practice and Teach EBM, Second Edition, Churchill Livingstone: Edinburgh, 2000.*
14. *Juristo, N., Moreno, A., Vegas, S. & Solari, M. , 'In search of what we experimentally know about unit testing', IEEE Software 23(6), pp72–80, 2006.*
15. *W. Afzal et al., A systematic review of search-based testing for non-functional system properties, Inform. Softw. Technol. 2009*
16. *Martin Fagereng Johansen, Øystein Haugen, Franck Fleurey , "A survey of empirics of strategies for software product line testing" Fourth International Conference on Software Testing, Verification and Validation Workshops, 2011*
17. *Sarvnaz Karimi, Falk Scholer , Supporting Complex Search Tasks, SIGIR Workshop Beijing 2011,.*

**18.** *Kandil, Mostafa, Hassanein, Ehab, Mazen, Sherif A.: Cross-View of Testing Techniques Toward Improving Web-Based Application Testing. In: CoRR, abs/1205.6677, 2012.*

**19.** *I. Ciupa, A. Pretschner, M. Oriol, A. Leitner, B. Meyer, On the number and nature of faults found by random testing. ICST 2008, the First IEEE International Conference on Software Testing, Verification and Validation Volume 21, Issue 1, pages 3–28, March 2011.*

**20.** *S. ALI, L. C. BRIAND, H. HEMMATI, and R. K. PANESAR-WALAWEGE, A Systematic Review of the Application and Empirical Investigation of Search-based Test-Case Generation, Future of Software Engineering: IEEE Computer Society, 2008.*

**21.** *B.A. Kitchenham, Guidelines for performing systematic literature reviews in software engineering, Tech. Rep., EBSE-2007-001, UK ‚July 2007.*

**22.** *Adapted from "A systematic review of search-based testing for non-functional system properties" Wasif Afzal , Richard Torkar, Robert Feldt*