Sridhar Chimalakonda
Guest Faculty, ***IIT Tirupati, India***
Visiting Faculty, ***IIIT Sri City, India***
Chief Research Advisor, ***FortunaPIX, India***
(+91) 9490705384
ch@iittp.ac.in
researchweb.iiit.ac.in/ sridhar_ch

Research Statement

Fusion of Research in Software Engineering and Educational Technologies

Research Summary

Interdisciplinary Research

Software is omnipresent today! from simple apps in smart phones to mission-critical systems developed by millions of engineers and end users from all walks of life. On the other hand, software is quite effort-intensive, increasingly complex [millions of lines of code?], buggy and never comes with any warranties/guaranties. It is here, I wish to address interesting and significant research challenges in the area of software engineering and computing. Specifically, I am interested to *empirically* and *qualitatively* investigate novel ways for improving *quality* of software while facilitating *reuse* through design of frameworks, platforms and tools. I am also keen to explore computing challenges from the largely unexplored research area of educational technologies and computing education in India. I see *interdisciplinary research* as a critical way forward to drive my research spanning across software engineering, educational technologies and human-computer interaction.

Core Philosophy

Quality, The Foundation

➳ *To me, impactful research stems from great problems that push our limits to go beyond state-of-the-art and state-of-the-practice!* ✺ long-term problems with realistic short-term goals

➳ *"Raise your quality standards as high as you can live with, avoid wasting your time on routine problems, and always try to work as closely as possible at the boundary of your abilities. Do this, because it is the only way of discovering how that boundary should be moved forward."* - Prof. Edsger W. Dijkstra. The foundations for my research stem from Prof. Edsger W. Dijkstra

➳ I construe this to push the *boundary of the field*, especially for an young and emerging discipline like software engineering in collaboration with fellow researchers in the community.

Research Agenda

Core Principles

➳ Promote undergrad research [not at the cost of fundamentals] ➳ Start in Sem III, groom and conduct research from Sem IV onwards. Two modes:
✺ *Research-driven* [Great Research + Good Development] ✺ research papers + some tools
✺ *Development-driven* [Great Development + Good Research] ✺ frameworks + some papers

Software Engineering

➳ *Analyze millions of software repositories qualitatively and quantitatively to assess quality [like correctness, security, reliability, usability and so on] of a diversified range of software artifacts [like code, bugs, logs, tests, patterns, designs and so on]*
➳ *How to instrument software with virtual agents and assistants that can continuously and automatically adapt the software as per evolving requirements?*
➳ *How to support millions of software developers, testers to write quality software?*

Educational Technologies

➳ *How to design and customize educational technologies for all levels of education, adapt them for varied teaching styles, learning styles and different modes of evaluation and for a range of subjects to be delivered in multiple languages?*
➳*Design of Virtual Reality & Augmented Reality environments for story telling, education [such as demonstrating complex procedures in medical domain], elevated experiences for history and culture*
➳ *Provide personalized life long learning for 7.1 billion learners*

Why Software Engineering Research Matters?

5.2 million developers in India by 2018?

Over $3.8 trillion of revenue, 18 million software developers and 7.1 billion users. Software is pervasive today in every aspect of life. Just imagine a world without *Google* and what if all the apps on smart phones stop working? This will only grow further at a much rapid pace. On the contrary, software has a history of failures causing losses of trillions of dollars and even lives. As software pervades natural life with sensors on our bodies, it is immensely critical to understand, analyze and scientifically provide a base for software engineering, which is still an open research problem today. This provides endless opportunities and especially for India where we have massive software industry but with no/limited research.

On Foundations for Software Engineering

Long-term research goal

Why software works sometimes and does not work some other times? Can we assure the quality of software? Can we prove that software is correct, secure, reliable and all other qualities? Can we provide guarantee/warranties for software? Is there a scientific and rigorous way of developing software? What is the science behind software engineering? Why is there a wide gap between industry and academia in software? Is software engineering a discipline or is it part of computer science? Is there a theory for software engineering?

The term software engineering was coined way back in 1968 to figure out answers to some of these fundamental questions and the field has made immense progress in the last 50 years [by 2018] but still a long way to go with most of these questions yet to be answered! This is both a grand challenge as well as a great opportunity to be part of and contribute to the crafting of a discipline. One example project attempting towards these questions is *DeepSpec*, an NSF funded project to prove end-to-end correctness of systems by creating a network of specifications for every aspect of software and hardware.

My quest is to do research in these directions, and yet be realistic and be in line with the community.

The Past - Summary of my existing research

Over the last decade or so, my research work has focused on facilitating software reuse with quality. How to design approaches through which software components can be composed to [semi-] automatically generate software systems? Most of my work has revolved around the idea of discovering patterns of reuse in domain, manifest those patterns in software components and generate systems. Specifically, I focused on facilitating reuse in a family of systems that falls into the area of software reuse and software product lines.

Software Engineering

In essence, my research has influenced creation of several international standards [ISO/IEC 26550, ISO/IEC 26551, ISO/IEC 26552, ISO/IEC 26553, ISO/IEC 26554, ISO/IEC 26555 focusing on aspects like requirements, architecture, verification and validation for product lines] in the area of Software & Systems Engineering and in particular software product lines. *This experience of being involved in the creation of international standards has helped me in getting an international exposure of work culture, collaboration involving several discussions and debates spanning over years. Owing to the stringent and rigorous process during standard creation, I learnt deeply about several areas in the field of software engineering to start and contribute to conversations with international experts in respective areas.*

Educational Technologies

The application of this research in software engineering has eventually led to creation of an approach for design of educational technologies for *scale* and *variety* in the context of adult literacy in India [287 million learners, 22 languages]. Our research and technologies are made public at http://rice.iiit.ac.in and are transferred to *National Literacy Mission Authority* of India for further proliferation across India.

Link

Detailed CV/contributions ✺ http://bit.ly/2mJVppj

The Present - Core Research Statement

Empirical Software Engineering

Over 2.6 million apps are available on *Play Store* and over 2 million apps on *App store* in 2016. The number of source code repositories in *Github* alone is around 50 million with 19 million users contributing to open source software. Every 8 seconds or so, a developer asks a question on *Stack Overflow*, a Q&A website for coding and every 11 minutes, the questions are being answered. There is a massive explosion of software in every field and in all possible forms for a wide variety of emerging contexts. *How can we assess and assure the quality of software in this massive explosion?*

➳ *Design a set of approaches, methods, tools and techniques that can assist a diversified range of stakeholders [such as architects, developers, testers] in software development to qualitatively and quantitatively analyze quality of software artifacts and further facilitate the [automatic] generation of high-quality software as per dynamic evolution of systems across a range of domains.*

Research in Software Engineering

I am making an evolving list of open research problems that are of interest to me available at http://bit.ly/2mGnn4V. *My current research focus is on empirical research in different areas of software engineering.* An initial set of research questions are as follows:

Generic Software Engineering

- How to adapt software engineering methods for massively distributed and parallel computing applications?
- How to adapt conventional software engineering approaches for design of mobile, virtual reality and other kinds of emerging applications?
- How to model incomplete, ambiguous, uncertain and emerging requirements?
- Can we change the programming language used in a software system without affecting the functionality of the system?
- Does existing software engineering methods work for big data applications?
- How to reduce the effort of software development by applying artificial intelligence?
- Does specifying software code using verbal beyond text makes sense?

Software Quality & Metrics

- How to qualitatively and quantitatively analyze quality of software artifacts?
- Can quality of a software system be varied at run time?
- Can we design a quality meter that shows co-relation between code and all qualities?
- What kinds of metrics are required to better predict maintenance of software systems?
- Can we design methods where software developers compete and code for quality?
- How to apply machine learning techniques for better prediction of software estimations?

Software Reuse

- Can we use existing open source components on the web and automatically generate software systems for a given set of specifications?
- Why composition of software components is hard? and How to facilitate their composition while assuring quality?
- How to formally specify composition without burdening software developers?
- How to make tools that can generate tools?
- How to reduce the effort of testing by 50%?

Sofware Architecture

- What kinds of software architecture styles facilitate evolution of systems?
- How to ensure integrity of a system induced by intended/unintended changes?
- Can we automatically verify the architecture of a software system for compliance?
- How to certify software components and software architectures for quality?
- Can we automatically identify design patterns in code?
- Can we refactor a code repository to best practices and design patterns?
- How to design a framework that helps software developers to choose the right set of framework/API by automatically suggesting trade-offs?
- Why software developers use a framework/API without reading documentation?

Software Process
- Can we automatically find why software processes fail and fix them at run time?
- Can software generate customized software development processes for every project?
- What kind of notations are required to model informal, incomplete and ambiguous processes? and Do we need domain-specific process notations?
- How to eliminate waste in software processes?
- What aspects of software development process cannot be fully automated?
- What are the efficient ways for integrating processes in software and domain?

Software Evolution
- How to predict the evolution of software bugs based on version histories?
- How to trace the evolution of a software artifact? Can we establish semantic co-relation between code that is being reused?
- Can version histories help in predicting software bugs?
- What is the lifetime of a software artifact [use cases, designs, code, bugs, test cases]?

Crowdsourcing
- Can we design mechanisms through which thousands of previously unknown software developers can write quality code?
- How to exploit the immense power of crowd to support quality in design of software?
- How to help end users to write programs that can add value to them?

Gamification
- What areas of software engineering can benefit from the use of gamification?
- How to make code review process interesting through gamification?
- Can we design a treasure hunt like game for discovering bugs?
- Can we design a trading game for doing trade-offs of software quality?

Formal Methods
- Can we weave model checking and testing approaches for better verification of systems?
- How can software developers without [or with minimal] mathematical and logical foundations background use formal methods?
- What are the theories for each of the qualities in engineering of software?
- How do we formally specify quality requirements in an informal language?

This list seems to be endless and I believe that once the first set of students start working, I will frame the research agenda towards a specific and concrete direction.

Research in Computing for Education

Personalized Learning

➳ *Provide personalized life long learning for 7.1 billion learners without over-burdening their lives and focusing on design of novel interfaces for varied audience such as house wives, field workers.*

➳ *Pattern-Oriented approach for customization of instructional designs*

➳ *Design of a gesture-based story telling framework and platform for generating apps for virtual and augmented reality experiences* ✺ An instance could be creation of stories in Indian epics

➳ *Games for Computing, Gamification of Computing and Design of domain-specific game engines*

Skill Education

Skill education is undergoing a major revamp in India extensively supported by Government of India through National Skill Development Corporation (NSDC). NSDC has set up 38 Sector Skill Councils pertaining to different industrial skills like automotive, healthcare and so on. Model curriculum for different sectors and roles has been designed. Further NSDC has partnered with 267 training partners in 25 sectors and 2500+ fixed and mobile centers. The partners are advised to customize the model curriculum as per local context, requirements, students and training methodology. However, most of the training today is done manually except using a few online videos. Can computing help? I see that manually developing educational technologies for the scale and variety present in this domain is a herculean task and the work from my PhD thesis can be adapted for this context.

Educational Technologies at all levels of education

How much effort is required to provide customized educational technologies for schooling in India? (KG1 to KG12)? How much effort is required for supporting teaching of engineering subjects (around 800 courses?)? The problems become compounded due to the number of subjects to be learnt, each subject having many topics, the number of languages used as media of instruction, and the number of students as well as teachers. There is a serious debt of ways to accelerate educational technologies even if we borrow ideas of software reuse from software engineering. I see that these challenges can push the limits of software engineering and even computing.

Research in Human Computer Interaction

I do not explicitly work on Human Computer Interaction but however, most of the evaluatons I do either in software engineering or educational technologies involves HCI and hence gained some interest in the recent times. ✺ Every user is a co-designer ✺ Beyond natural user interfaces. A couple of questions that I started with are: ✺ *Can visual annotation of email systems handle email overload?* ✺ *What kinds of gesture based interfaces help improve usability of software?* ✺ *How to evaluate the usability of Government applications?*

Proposed Approach & Strategy

How to address these challenges?

➳ Perform empirical analysis on different datasets [create them as well] and gain insights

➳ Develop approaches/methods to address the challenges based on these empirical studies

➳ Apply the approaches on industry and open source data multiple times and in multiple contexts for assessing the efficacy of the proposed solutions

➳ Repeat this cycle till we get to some core abstractions and theories

A Final Note

➳ *There is an ocean of research yet to be done in software engineering and educational technologies igniting intellectual curiosity and eventually leading to significant impact for society. I wish to strategically address these challenges in a systematic way as I go forward!*