Design Decisions

CS6.401 Software Engineering

Rudra Dhar

PhD Student

SERC, IIIT-H

SERC

Software Engineering Research Centre

INTERNATIONAL INSTITUTE OF
INFORMATION TECHNOLOGY
HYDERABAD

Let's design ChatGPT

- Key functionalities
- Tech-stack, (programming language)
- Hardware limitations
- Team skill
- Financial Constraint
- Socio-political
- Data

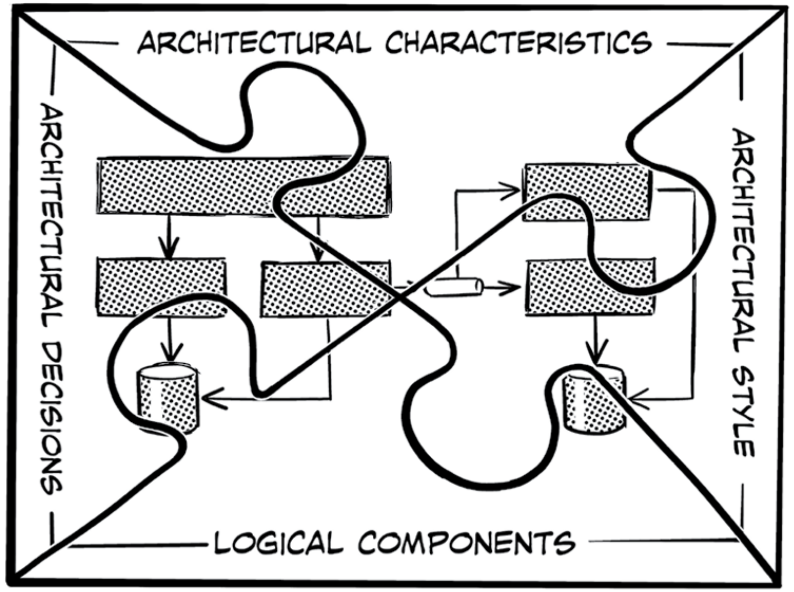
Architectural Design Decision

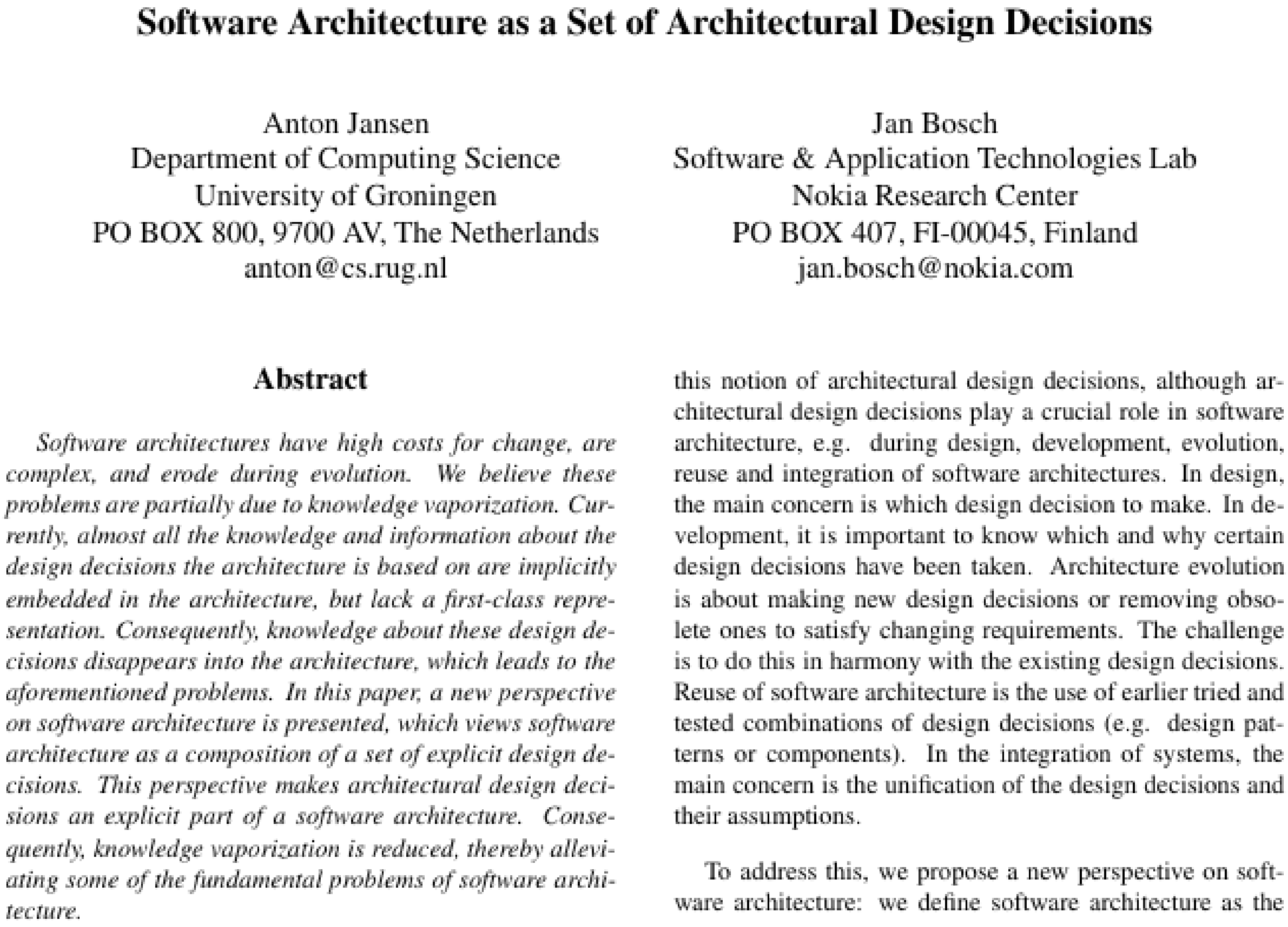
Software Architecture

Software Architecture is the stuff
that's hard to change.
- Martin Fowler

Software Architecture =
Components and Connectors
- Garlan and Shaw

Software Architecture =
{ Elements, Form, Rationale }
- Perry and Wolf
(Foundation of Software Architecture)





Software Architecture as a Set of Architectural Design Decisions

Anton Jansen
Department of Computing Science
University of Groningen
PO BOX 800, 9700 AV, The Netherlands
anton@cs.rug.nl

Jan Bosch
Software & Application Technologies Lab
Nokia Research Center
PO BOX 407, FI-00045, Finland
jan.bosch@nokia.com

Abstract

Software architectures have high costs for change, are complex, and erode during evolution. We believe these problems are partially due to knowledge vaporization. Currently, almost all the knowledge and information about the design decisions the architecture is based on are implicitly embedded in the architecture, but lack a first-class representation. Consequently, knowledge about these design decisions disappears into the architecture, which leads to the aforementioned problems. In this paper, a new perspective on software architecture is presented, which views software architecture as a composition of a set of explicit design decisions. This perspective makes architectural design decisions an explicit part of a software architecture. Consequently, knowledge vaporization is reduced, thereby alleviating some of the fundamental problems of software architecture.

this notion of architectural design decisions, although architectural design decisions play a crucial role in software architecture, e.g. during design, development, evolution, reuse and integration of software architectures. In design, the main concern is which design decision to make. In development, it is important to know which and why certain design decisions have been taken. Architecture evolution is about making new design decisions or removing obsolete ones to satisfy changing requirements. The challenge is to do this in harmony with the existing design decisions. Reuse of software architecture is the use of earlier tried and tested combinations of design decisions (e.g. design patterns or components). In the integration of systems, the main concern is the unification of the design decisions and their assumptions.

To address this, we propose a new perspective on software architecture: we define software architecture as the

A. Jansen and J. Bosch, "Software Architecture as a Set of Architectural Design Decisions," 5th Working IEEE/IFIP Conference on Software Architecture (WICSA'05), Pittsburgh, PA, USA, 2005, pp. 109-120, doi: 10.1109/WICSA.2005.61.

Architectural vs Design Decision

Low <-------- Significance -------> High

Low <-------- Effort-------> High

Tactical <---------------> Strategic

<-->

Renaming a method or function

Deciding which programing language to use

Migrating your system to cloud

Why record ADDs ?

- Lack of clear visibility into the reasons behind the current architecture.
- Implicit embedding of design decisions leads to **Knowledge Vaporization**.

- Violation of design rules and constraints during system evolution leads to architectural drift.
- Failure to remove obsolete design decisions accelerates system erosion
- High cost of change.

- **Informed** Decision making
- Improved **traceability** of design decisions

Architectural Decision Record

Simple ADR

Status

Accepted

Context

We need to decide on whether to use Python as a programming language for our project. Our project involves data analysis, machine learning, and web development.

Decision

We have decided to use Python as our primary programming language for our project.

We chose Python as it is a versatile programming language that is widely used in the areas of data analysis, machine learning , and web development.

Consequences

Python streamlines development with rich libraries and community support for data analysis and machine learning.

But it may suffer from slower runtime performance and higher memory overhead.

How to write proper ADRs

Where to learn from

- https://adr.github.io/
- https://github.com/joelparkerhenderson/architecture-decision-record

Standard Templates

- Nygard - https://cognitect.com/blog/2011/11/15/documenting-architecture-decisions
- MADR - https://github.com/adr/madr?tab=readme-ov-file

Sample repositories

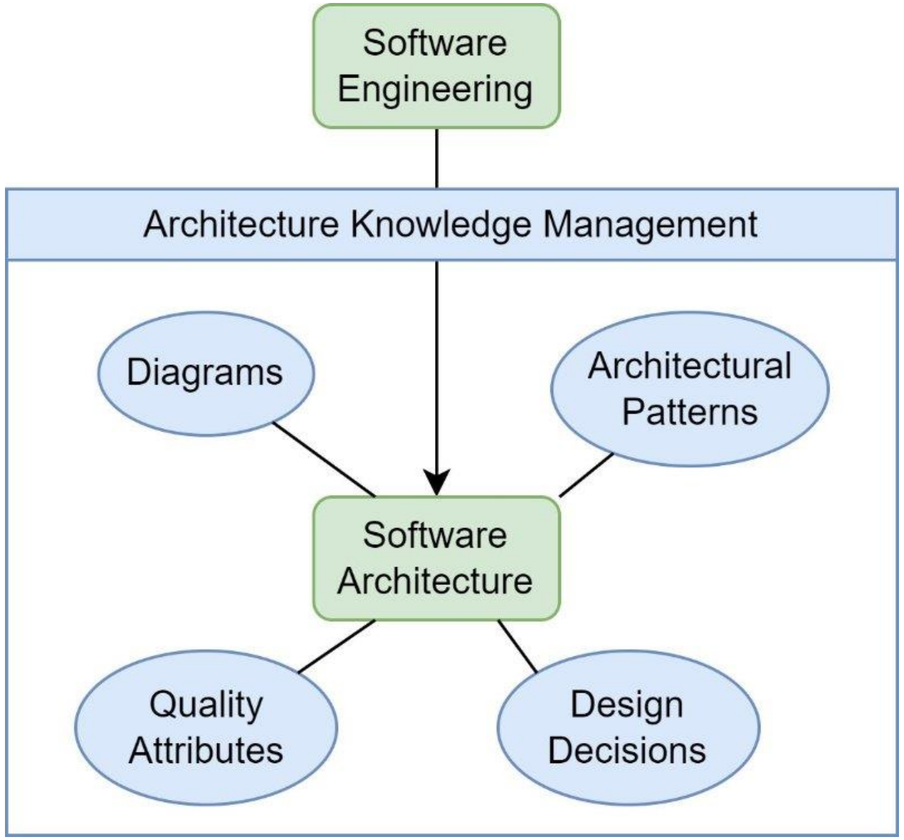
- https://github.com/arachne-framework/architecture/tree/master
- https://github.com/npryce/adr-tools/tree/master/doc/adr
- https://github.com/mozilla/fxa/tree/main/docs/adr
- Bad example - https://github.com/actions/toolkit/blob/main/docs/adrs/0381-glob-module.md

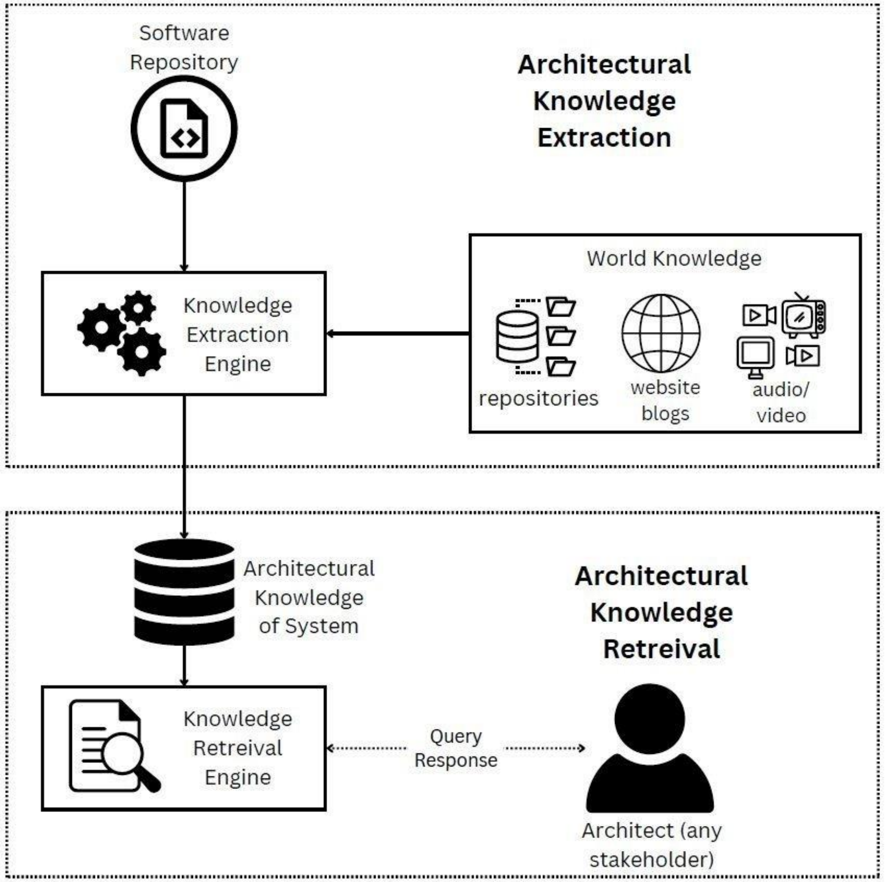
Recommended best practices

- https://docs.aws.amazon.com/prescriptive-guidance/latest/architectural-decision-records/adr-process.html
- https://learn.microsoft.com/en-us/azure/well-architected/architect-role/architecture-decision-record

My Research

Research Area





Software Repository
Architectural Knowledge Extraction
Knowledge Extraction Engine
World Knowledge
repositories
website blogs
audio/ video
Architectural Knowledge of System
Architectural Knowledge Retreival
Knowledge Retreival Engine
Query
Response
Architect (any stakeholder)

Generating Design Decision using LLMs

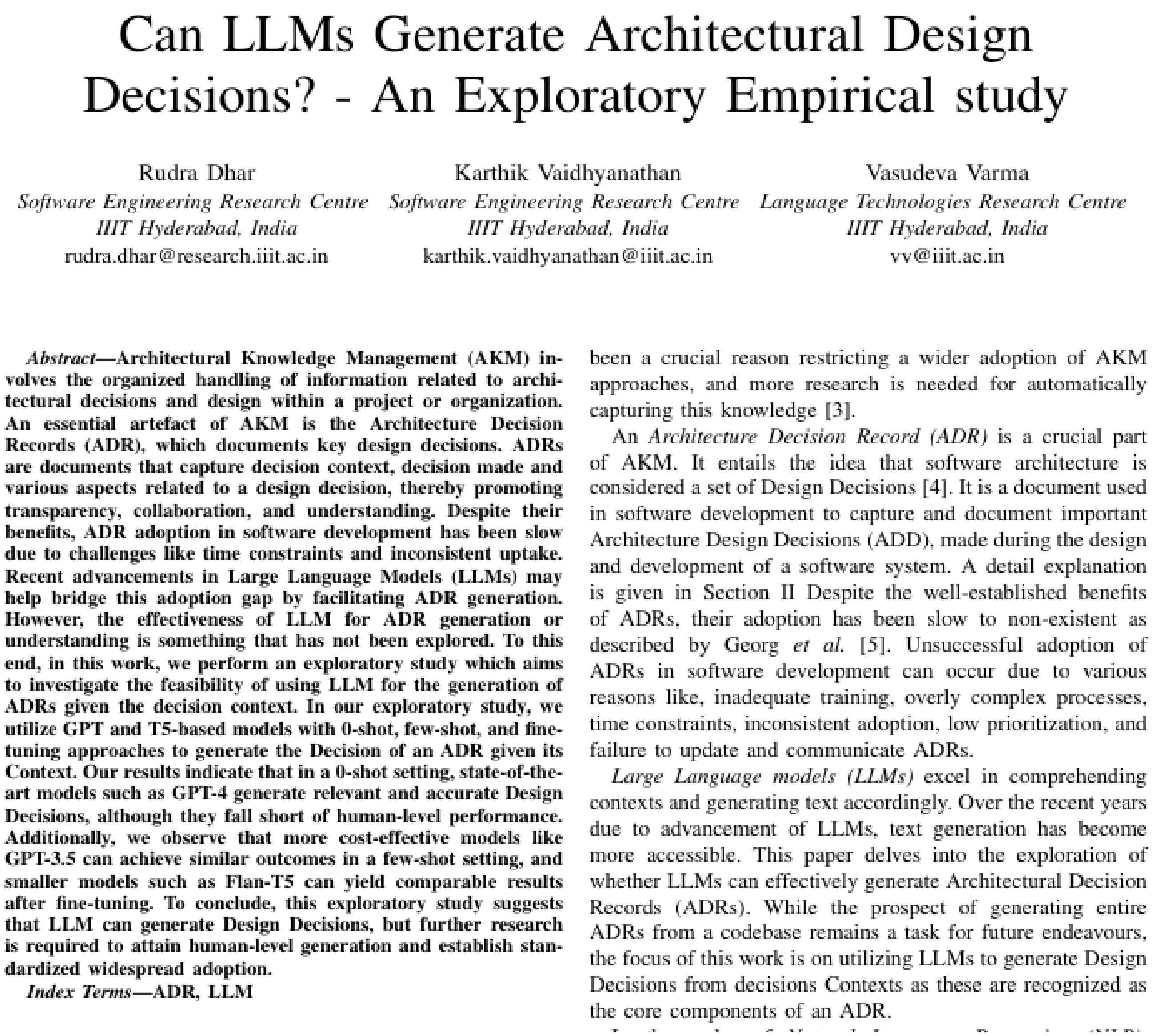
Context

We need to decide on whether to use Python as a programming language for our project. Our project involves data analysis, machine learning, and web development.

Decision

We have decided to use Python as our primary programming language for our project.

We chose Python as it is a versatile programming language that is widely used in the areas of data analysis, machine learning, and web development.



Can LLMs Generate Architectural Design Decisions? - An Exploratory Empirical study

Rudra Dhar
Software Engineering Research Centre
IIIT Hyderabad, India
rudra.dhar@research.iiit.ac.in

Karthik Vaidhyanathan
Software Engineering Research Centre
IIIT Hyderabad, India
karthik.vaidhyanathan@iiit.ac.in

Vasudeva Varma
Language Technologies Research Centre
IIIT Hyderabad, India
vv@iiit.ac.in

***Abstract*—Architectural Knowledge Management (AKM) involves the organized handling of information related to architectural decisions and design within a project or organization. An essential artefact of AKM is the Architecture Decision Records (ADR), which documents key design decisions. ADRs are documents that capture decision context, decision made and various aspects related to a design decision, thereby promoting transparency, collaboration, and understanding. Despite their benefits, ADR adoption in software development has been slow due to challenges like time constraints and inconsistent uptake. Recent advancements in Large Language Models (LLMs) may help bridge this adoption gap by facilitating ADR generation. However, the effectiveness of LLM for ADR generation or understanding is something that has not been explored. To this end, in this work, we perform an exploratory study which aims to investigate the feasibility of using LLM for the generation of ADRs given the decision context. In our exploratory study, we utilize GPT and T5-based models with 0-shot, few-shot, and fine-tuning approaches to generate the Decision of an ADR given its Context. Our results indicate that in a 0-shot setting, state-of-the-art models such as GPT-4 generate relevant and accurate Design Decisions, although they fall short of human-level performance. Additionally, we observe that more cost-effective models like GPT-3.5 can achieve similar outcomes in a few-shot setting, and smaller models such as Flan-T5 can yield comparable results after fine-tuning. To conclude, this exploratory study suggests that LLM can generate Design Decisions, but further research is required to attain human-level generation and establish standardized widespread adoption.**

***Index Terms*—ADR, LLM**

been a crucial reason restricting a wider adoption of AKM approaches, and more research is needed for automatically capturing this knowledge [3].

An *Architecture Decision Record (ADR)* is a crucial part of AKM. It entails the idea that software architecture is considered a set of Design Decisions [4]. It is a document used in software development to capture and document important Architecture Design Decisions (ADD), made during the design and development of a software system. A detail explanation is given in Section II Despite the well-established benefits of ADRs, their adoption has been slow to non-existent as described by Georg *et al.* [5]. Unsuccessful adoption of ADRs in software development can occur due to various reasons like, inadequate training, overly complex processes, time constraints, inconsistent adoption, low prioritization, and failure to update and communicate ADRs.

Large Language models (LLMs) excel in comprehending contexts and generating text accordingly. Over the recent years due to advancement of LLMs, text generation has become more accessible. This paper delves into the exploration of whether LLMs can effectively generate Architectural Decision Records (ADRs). While the prospect of generating entire ADRs from a codebase remains a task for future endeavours, the focus of this work is on utilizing LLMs to generate Design Decisions from decisions Contexts as these are recognized as the core components of an ADR.

Rudra Dhar, Karthik Vaidhyanathan, Vasudeva Varma. Can LLMs Generate Architectural Design Decisions: An Exploratory Empirical Study, ICSA 2024

Outcome and future Work

- LLM can be used to generate Design Decisions. Next we should generate full ADRs

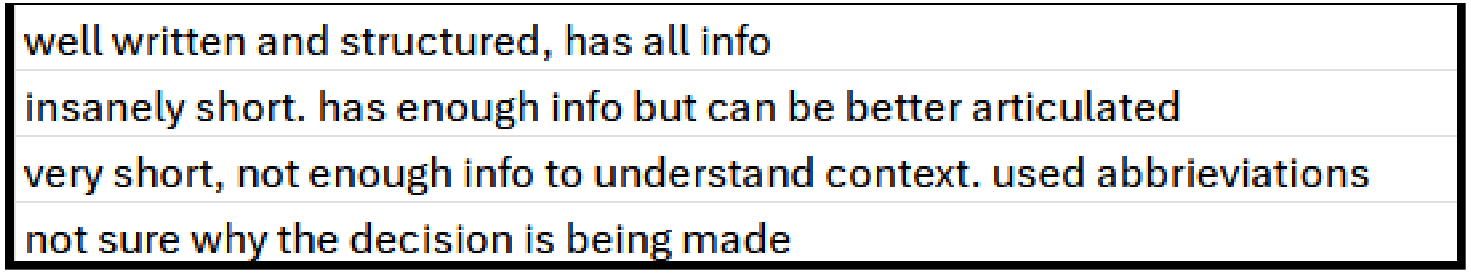
- Many people in the real world don't write proper ADRs
  - SE students should

- Influenced (finetune, or just few shot) by real people (data) :

  LLMS often generate bad Design Decisions

Class Activity – Determine if ADRs are good or not

- You will be given 6 ADRs and a google form.
- Copy the file name of the ADR as is from the drive to your form response, along with the ".md"
- Choose if the ADR is good, bad, or you're unsure
- Write a short ~one sentence remark of why YOU think it is good/bad/unsure. Do not use an LLM for this part, it is essential to write your observations
- If it is bad/ugly, improve it. Add the section necessary, add more text, etc. You can use an LLM in this step. Upload this improved .md or .txt file in the form.



well written and structured, has all info
insanely short. has enough info but can be better articulated
very short, not enough info to understand context. used abbrieviations
not sure why the decision is being made

Example of the remarks you need to write

Class Activity – Determine if ADRs are good or not

Folder containing your files

https://drive.google.com/drive/folders/1b74dv4hI57LWQWdOk7-oRrzWOltljrEa?usp=sharing

Form for your responses

https://docs.google.com/forms/d/e/1FAIpQLScO4L095Bw4eolU_qVyX6WTQOQpR2gBi-bN6Gdmqyn8mtvElg/viewform?usp=sharing

Thank You

Rudra Dhar
rudra.dhar@research.iiit.ac.in