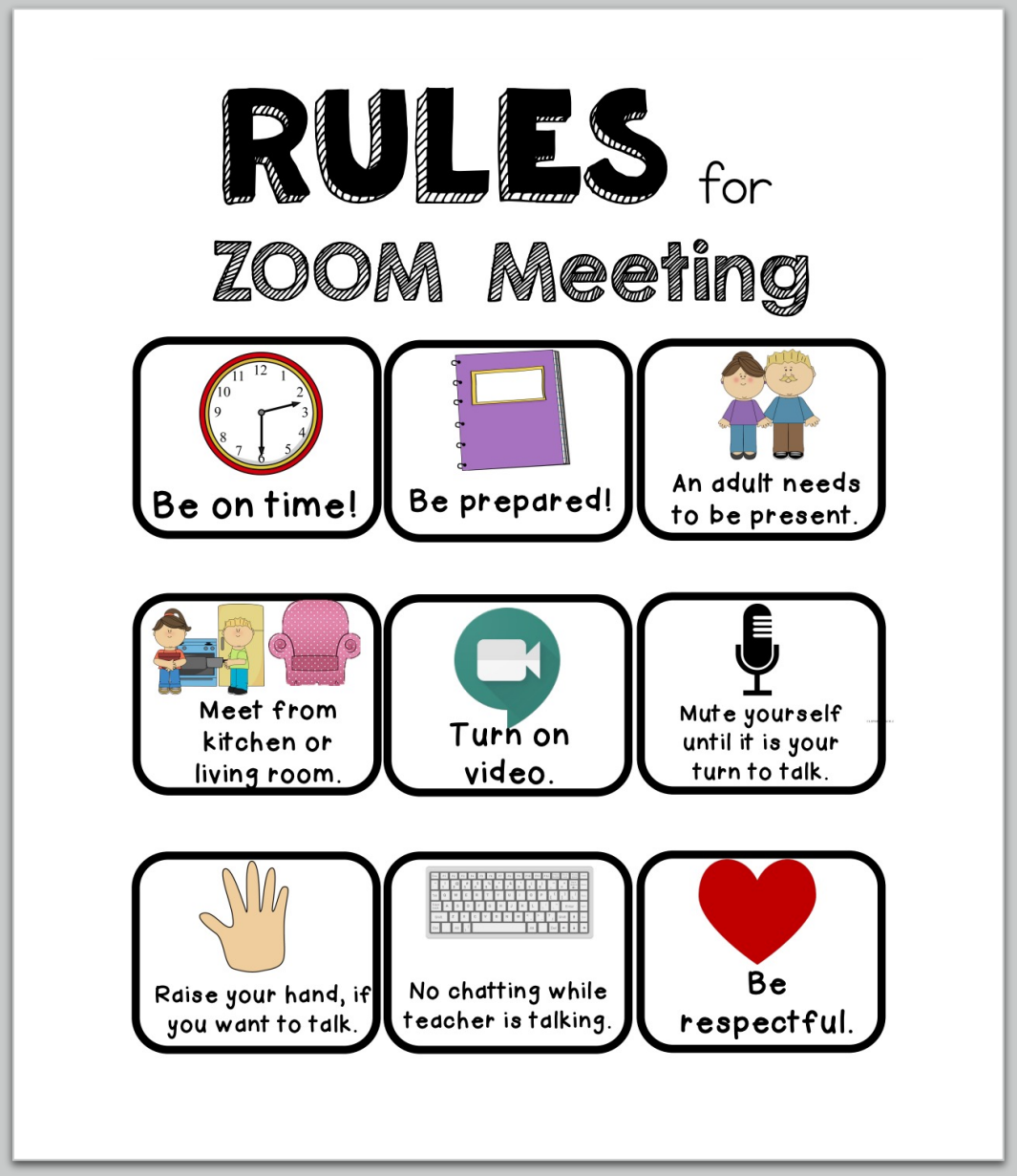
Course Overview and Introduction

SOEN 691: Engineering AI-based Software Systems

Emad Shihab, Hassan Khatoonabadi
Concordia University

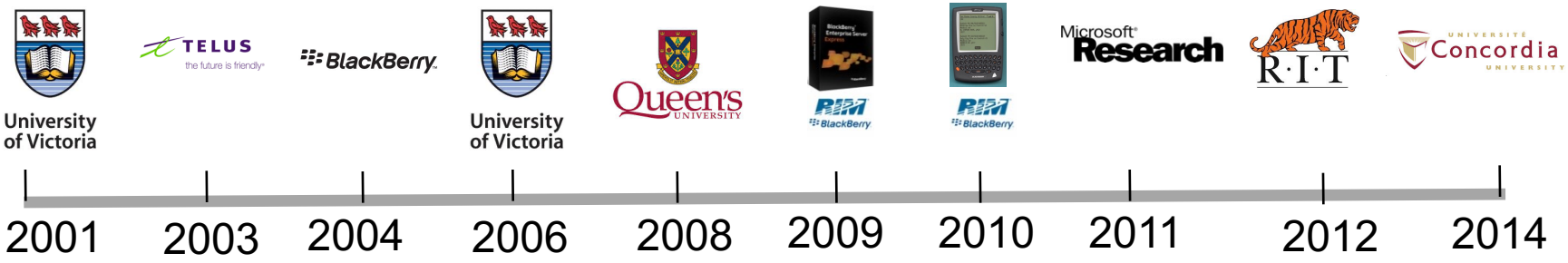
First thing first…



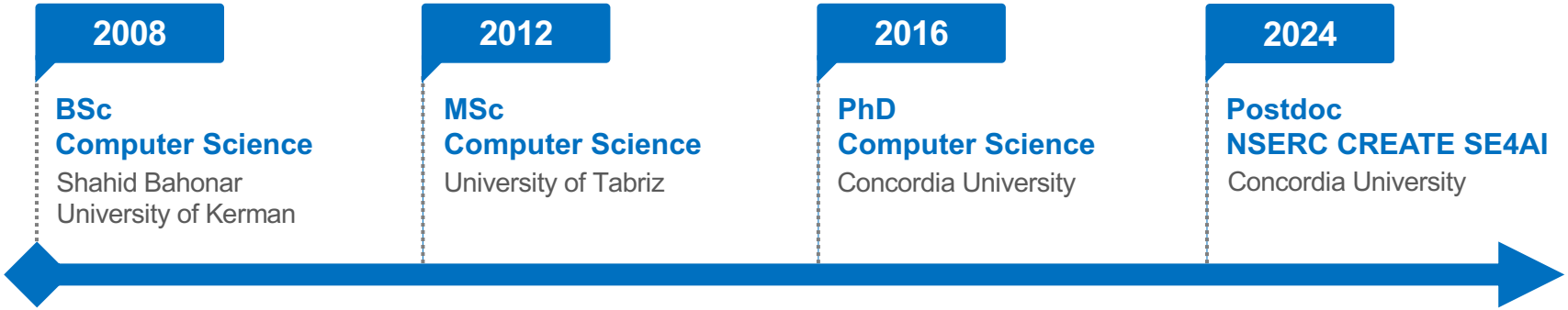
https://www.teacherspayteachers.com/Product/Google-Hangout-and-Zoom-Meeting-Rules-for-Elementary-Students-5431502

Introductions

Who is Prof. Shihab



Who is Hassan



Engineering AI-based Software Systems

What is AI?

Britanica: "… the ability of a digital computer or computer-controlled robot to perform tasks commonly associated with intelligent beings…"

Wikipedia: "… is intelligence demonstrated by machines, as opposed to natural intelligence displayed by animals including humans…"

IBM: "… is a field, which combines computer science and robust datasets, to enable problem-solving…"

Weak/narrow vs strong/general AI

What is AI?

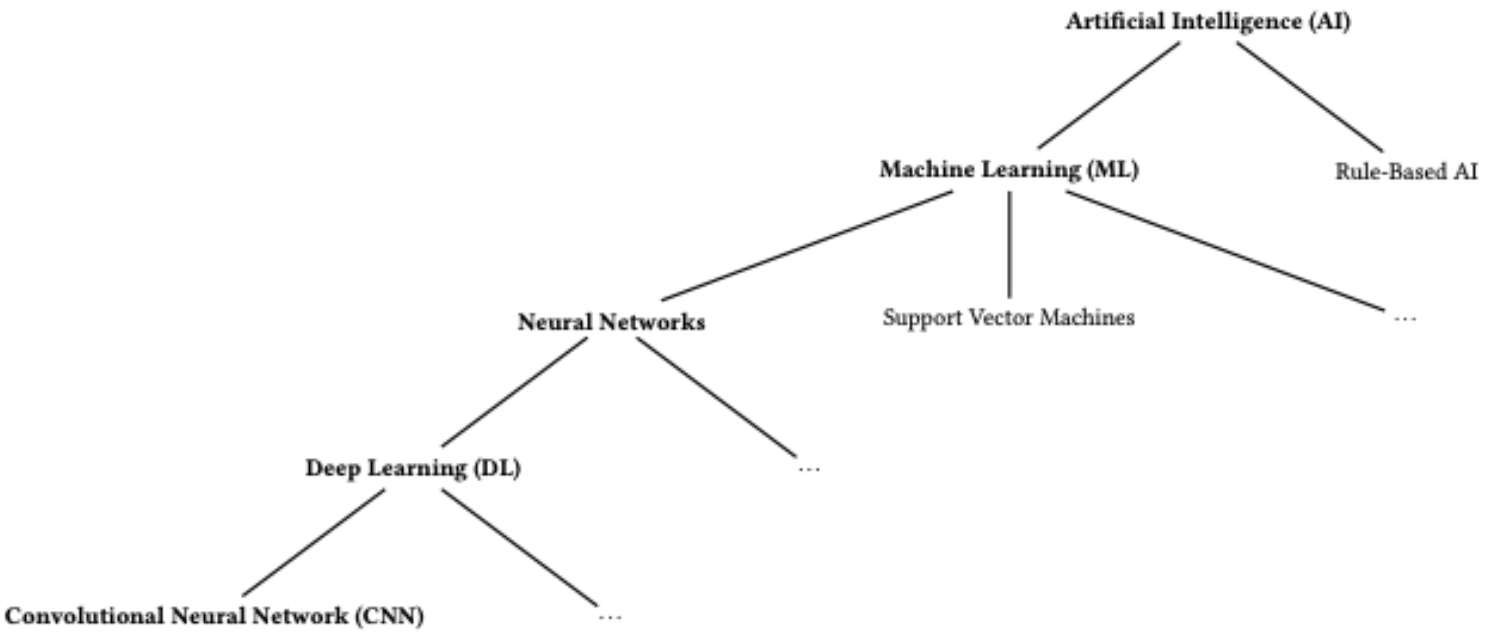


Fig. 17. Example taxonomic classification for paper [32].

https://arxiv.org/abs/2105.01984

What is an AI-based Software System?

- ….systems that learn by **analyzing their environment** and **taking actions** that aim to have **intelligent behavior**

- Systems that integrate AI capabilities
  - Statistical Learning
  - Machine Learning
  - Deep Learning

https://arxiv.org/abs/2105.01984

What is an AI-based Software System?

- AI-based Software Systems are systems that **include one or more AI component** (and other components)

- AI component: a part of the system that uses AI. Examples:
  - Embedded AI code
  - Using an AI library to implement an AI algo

- **Terms used**: AI technologies, AI-based systems, AI-infused systems, AI-enabled systems, AI/ML/DL software/system

The AI System "waves"

According to DARPA, AI is in its third wave.

- First wave: (mostly) **rule-based** systems
- Second wave: **statistical learning** systems
- Third wave: **neural networks**, mostly DL

….at least AI and COVID have something in common…they both come in waves ☺ (outdated joke)

(Refresher)
Software Engineering Basics

Software Engineering

Software engineering is concerned with:

- **all aspects of software production** from the **early stages** of system **specification through maintenance** the system after it has gone into use.

Concerns all aspects of software production

- **Not just technical** process of development. Also project management and the development of tools, methods etc. to support software production.

SE Core Activities

- Requirement (elicitation, analysis, specification, etc.)
- Software design
- Software architecture
- Implementation & Integration
- Testing
- Maintenance
- ….

Fundamental SE Activities

Specification

Development

Validation

Evolution

Software Process Activities

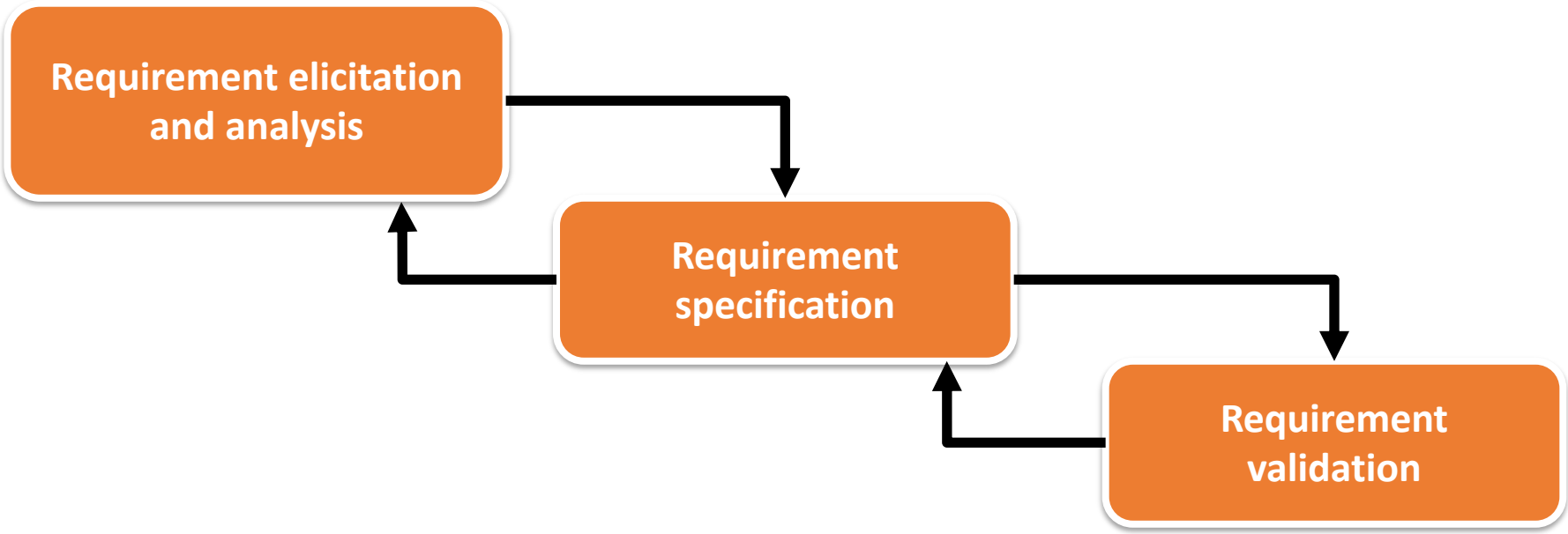
- **Software specification:** customers & engineers **define the software** that is to be produced and the **constraints on its operation**
- **Software development:** the software is **designed** and **programmed**
- **Software validation:** the software is **checked to ensure** that it is what the **customer requires**
- **Software evolution:** modifications done to meet **changing customer and market needs**

Phases and Models of Software Process

- There are many **different software process models**, but they all **share the same basic elements**

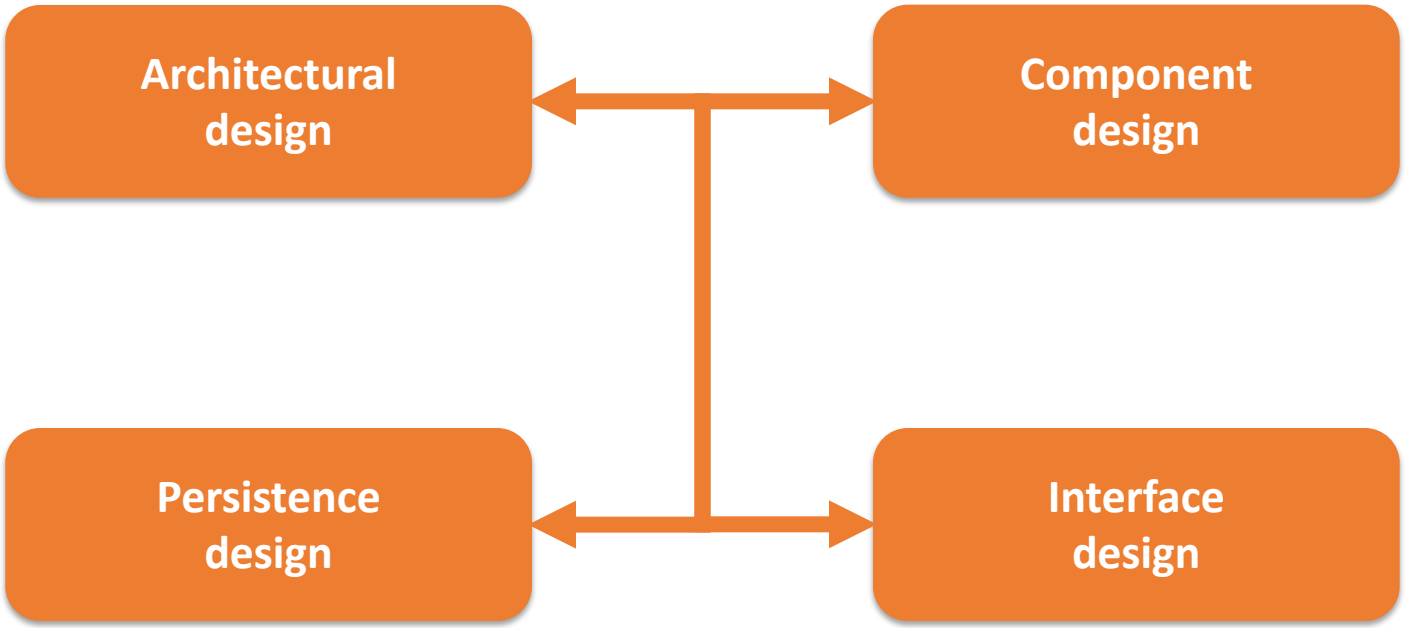
- The difference is in how these elements are organized.

Requirements/Specification

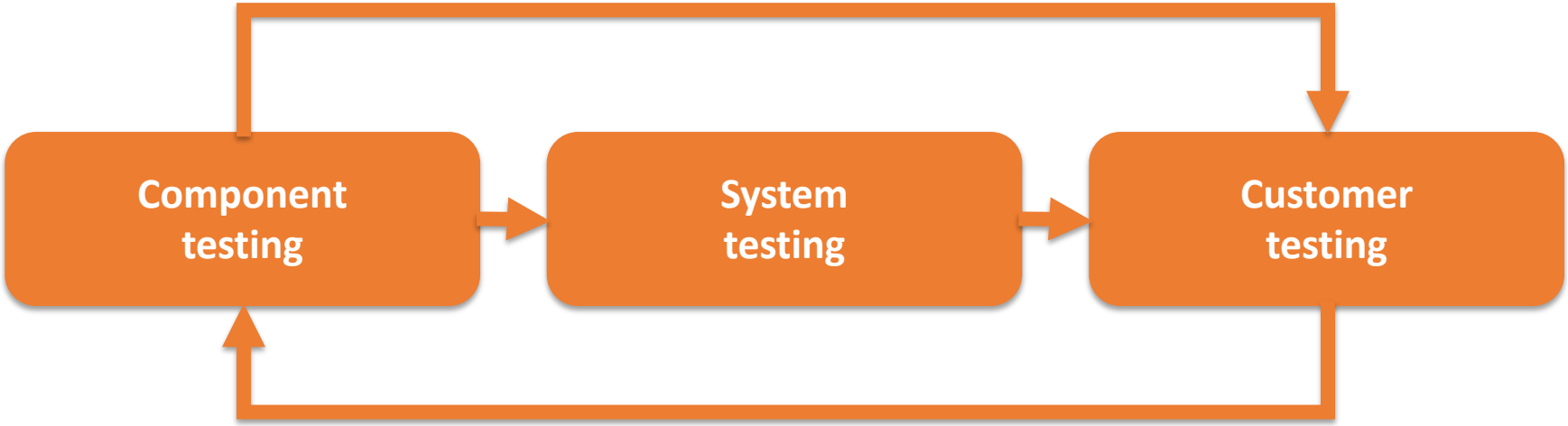


- User vs. System
- Functional vs Non-functional

Design


Architectural design
Component design
Persistence design
Interface design

Validation/Testing

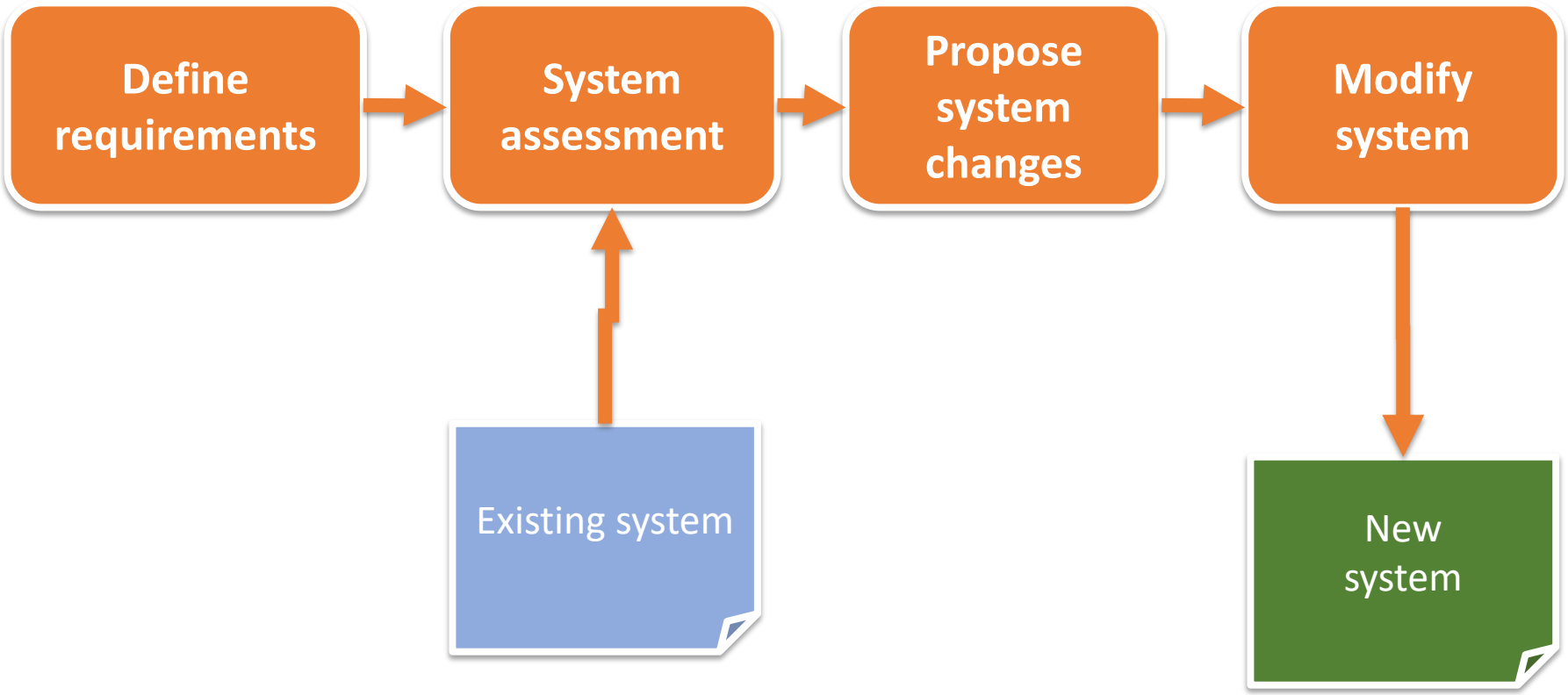


Done by the person who writes the code
Often considered as part of testing

Feature and performance testing
Regression testing

Acceptance testing
Field testing

Evolution


Define requirements
System assessment
Propose system changes
Modify system
Existing system
New system


How the customer explained it

Software Engineering
+
AI-based Software Systems

So… What's the Big Deal?

- In AI-based systems, **rules and system behavior** are inferred from training data (amongst others), rather than program logic and code only

- Moving parts include:
  - Large datasets that play a critical role in system behavior
  - Algorithmic performance
  - Infrastructure
  - Ethics and equity
  - ….

- Because of this, we need SE4AI!

A Meta-Summary of Challenges in Building Products with ML Components – Collecting Experiences from 4758+ Practitioners

Nadia Nahar
nadian@andrew.cmu.edu
Carnegie Mellon University
Pittsburgh, PA, USA

Haoran Zhang
Carnegie Mellon University
Pittsburgh, PA, USA

Grace Lewis
Carnegie Mellon Software Engineering Institute
Pittsburgh, PA, USA

Shurui Zhou
University of Toronto
Toronto, Ontario, Canada

Christian Kästner
Carnegie Mellon University
Pittsburgh, PA, USA

ABSTRACT

Incorporating machine learning (ML) components into software products raises new software-engineering challenges and exacerbates existing challenges. Many researchers have invested significant effort in understanding the challenges of industry practitioners working on building products with ML components, through interviews and surveys with practitioners. With the intention to aggregate and present their collective findings, we conduct a meta-summary study: We collect 50 relevant papers that together interacted with over 4758 practitioners using guidelines for systematic literature reviews. We then collected, grouped, and organized the over 500 mentions of challenges within those papers. We highlight the most commonly reported challenges and hope this meta-summary will be a useful resource for the research community to prioritize research and education in this field.

1 INTRODUCTION

After decades of effort in machine learning (ML) research to build better models, researchers from industry and academia have recently started to shift their attention to improving how to build software products with such models. Incorporating a ML component into a software product is often argued to be harder than incorporating traditional functional components, because of the specific characteristics of machine learning (e.g., based on data, no specifications in the traditional sense, fairness concerns) and how they impact the entire life cycle of the product [41, 49, 58, 83, 111]. While the traditional software development process has challenges of its own, bringing ML into the picture is argued to break a lot of existing software architecture and engineering assumptions [49, 58]. This leads initiatives to rethink existing processes and practices and shift priorities in software teams. As a result, we keep hearing from practitioners on how they perceive building, deploying, and incorporating machine learning in software products as a challenge, even when the initial ML research and model prototypes seemed promising.

While some practitioners give talks on challenges or write experience papers (e.g., examples in academic venues surveyed elsewhere [66, 84]), researchers have also been actively studying the challenges faced by practitioners when building software products with ML components across many projects. In recent years, many researchers have *interviewed* or *surveyed* practitioners to identify what has really changed for them with the introduction of machine learning, often with the goal of identifying challenges, research opportunities, and best practices in a rapidly changing field. While some studies focus on specific aspects, such as challenges regarding

Requirements Engineering: Lack of AI literacy causes unrealistic expectations from customers, managers, and even other team members • Vagueness in ML problem specifications makes it difficult to map business goals to performance metrics • Regulatory constraints specific to data and ML introduce additional requirements that restrict development
Architecture, Design, and Implementation: Transitioning from a model-centric to a pipeline-driven or system-wide view is considered important for moving into production, but a difficult paradigm shift for many teams • ML adds substantial design complexity with many, often implicit, data and tooling dependencies, and entanglements due to a lack of modularity • Difficulty in scaling model training and deployment on diverse hardware • While monitorability and planning for change are often considered important, they are mostly considered only late after launching
Model Development: Model development benefits from engineering infrastructure and tooling but provided infrastructure and technical support are limited in many teams • Code quality is not standardized in model development tools, leading to conflicts about code quality
Data Engineering: Data quality is considered important, but difficult for practitioners and not well supported by tools • Internal data security and privacy policies restrict data access and use • Although training-serving skew is common, many teams lack support for its required detection and monitoring • Data versioning and provenance tracking are often seen as elusive, with not enough tool support
Quality Assurance: Testing and debugging ML models is difficult due to lack of specifications • Testing of model interactions, pipelines, and the entire system is considered challenging and often neglected • Testing and monitoring models in production are considered important but difficult, and often not done • There are no standard processes or guidelines on how to assess system qualities such as fairness, security, and safety in practice
Process: Development of products with ML component(s) is often ad-hoc, lacking well-defined processes • The uncertainty in ML development makes it hard to plan and estimate effort and time
Organization and Teams: Building products with ML components requires diverse skill sets, which is often missing in development teams • Many teams are not well prepared for the extensive interdisciplinary collaboration and communication needed in ML products • ML development can be costly and resource limits can substantially curb/limit efforts • Lack of organizational incentives, resources, and education hampers achieving all system-level qualities

Table 1: Overview of Identified Challenges

https://arxiv.org/abs/2304.00078

Paper coverage

- Examined 50 papers covering 4758 practitioners
- Between 2010 – 2022

- What are the **challenges experienced by industry practitioners** in building software products with ML components?

Requirement Engineering

- **Unrealistic expectations** from ML systems
  - No false positives or very high accuracy
- Not wanting to pay for **continuous improvements** of the model
- ML-specific **system-level qualities** like fairness are **ignored during requirements elicitation**

Requirement Engineering

- It is difficult to **link ML goals to business goals**
  - Responsible AI initiatives, e.g., fairness or explainability, may not contribute to the business goals
  - Many ML projects are exploratory without a business goal
- **Regulatory** ML and data **specifications** impose additional requirements
  - Supporting audits, privacy (right to be forgotten, consent)

Architecture, Design and Implementation

- Moving **from model to full ML pipeline is difficult**
  - Often, these production pipelines are complex and require the integration of many frameworks and technologies
- ML-based systems are **complex to design** due to many **entanglements of data, code, ML code, etc**.
  - This leads to pipeline jungles & CACE
- **Monitoring and planning** for change often **happen after launching**
  - Monitoring for drift and bias is often an afterthought and done in an ad-hoc manner
- ML-software goes through **more revisions compared to traditional software**
  - Due to retraining, model replacement, hyperparameter tuning, change of domain

Data Engineering

- **Data quality** is important but difficult to achieve
  - Difficult to validate and improve data quality
  - Data pre-processing, cleaning and assembly requires lots of effort
- Data **access is often restricted** due to security and privacy concerns
- Data in **production** is often **drifted** and less supported by the (even best trained) models
- **Traceability and transparency is difficult** to achieve, especially when it comes to data

Quality Assurance

- **Testing and debugging ML models is difficult**
  - Very difficult to specify **metrics** to determine quality given that models will never be correct 100%
  - **Amount and types of mistakes** to accept is difficult to determine
  - Having **enough (quantity and quality) data** to represent the production environment is difficult
- **Testing** the **entire pipeline** and the interaction is difficult
  - **Unit tests can never be enough** due to **entanglement** of ML and other components
  - Need to have system level tests beyond just the ML

Quality Assurance

- Testing and monitoring in production
  - Need to have **(online) production tests** (in addition to offline) to handle drift, etc.
  - **Online testing can be very time consuming** since you need **longer observation** periods to obtain meaningful results
  - Very difficult to determine when a model is underperforming in online testing

- No clear criteria on how to **test for fairness, robustness, security, safety**

Process

- There is **no clear process** to developing ML systems
  - Often ad-hoc processes are used to develop ML components and products around them
- **Uncertainty** in ML systems makes them **difficult to develop**
  - Sometimes you **need to iterate many times** to achieve a certain accuracy, hence, it is difficult to plan and set expectations
- **Documentation** is important, particularly for the models and data (but often missing)
  - Final model has been **through many explorations and experiments** to determine parameters, features, etc.

Organization & Teams

- **ML systems** require **many different teams and skills** to be successful
  - HW, engineering, SW, UX, Math and stats, business and ops
- AI literacy is important, for SW engineers and for data scientists
- Because of this, we need SE4AI!

Current State of SE4AI

Software Engineering for AI-Based Systems: A Survey

SILVERIO MARTÍNEZ-FERNÁNDEZ, Universitat Politècnica de Catalunya - BarcelonaTech, Spain
JUSTUS BOGNER, University of Stuttgart, Institute of Software Engineering, Germany
XAVIER FRANCH, Universitat Politècnica de Catalunya - BarcelonaTech, Spain
MARC ORIOL, Universitat Politècnica de Catalunya - BarcelonaTech, Spain
JULIEN SIEBERT, Fraunhofer Institute for Experimental Software Engineering IESE, Germany
ADAM TRENDOWICZ, Fraunhofer Institute for Experimental Software Engineering IESE, Germany
ANNA MARIA VOLLMER, Fraunhofer Institute for Experimental Software Engineering IESE, Germany
STEFAN WAGNER, University of Stuttgart, Institute of Software Engineering, Germany

AI-based systems are software systems with functionalities enabled by at least one AI component (e.g., for image-, speech-recognition, and autonomous driving). AI-based systems are becoming pervasive in society due to advances in AI. However, there is limited synthesized knowledge on Software Engineering (SE) approaches for building, operating, and maintaining AI-based systems. To collect and analyze state-of-the-art knowledge about SE for AI-based systems, we conducted a systematic mapping study. We considered 248 studies published between January 2010 and March 2020. SE for AI-based systems is an emerging research area, where more than 2/3 of the studies have been published since 2018. The most studied properties of AI-based systems are dependability and safety. We identified multiple SE approaches for AI-based systems, which we classified according to the SWEBOK areas. Studies related to software testing and software quality are very prevalent, while areas like software maintenance seem neglected. Data-related issues are the most recurrent challenges. Our results are valuable for: researchers, to quickly understand the state-of-the-art and learn which topics need more research; practitioners, to learn about the approaches and challenges that SE entails for AI-based systems; and, educators, to bridge the gap among SE and AI in their curricula.

CCS Concepts: • **Software and its engineering → Software creation and management**; • **Computing methodologies → Machine learning**;

Additional Key Words and Phrases: software engineering, artificial intelligence, AI-based systems, systematic mapping study

1 INTRODUCTION

In the last decade, increased computer processing power, larger datasets, and better algorithms have enabled advances in Artificial Intelligence (AI) [11]. Indeed, AI has evolved towards a new wave, which Deng calls "the rising wave of Deep Learning" (DL) [46][1]. DL has become feasible, leading to Machine Learning (ML) becoming integral to many widely used software services and applications [46]. For instance, AI has brought a number of important applications, such as image- and speech-recognition and autonomous, vehicle navigation, to near-human levels of performance [11].

The new wave of AI has hit the software industry with the proliferation of AI-based systems integrating AI capabilities based on advances in ML and DL [6, 24]. AI-based systems are software systems which include AI components. These systems learn by analyzing their environment and taking actions, aiming at having an intelligent behaviour. As defined by the expert group on AI of the European Commission, "AI-based systems can be purely software-based, acting in the virtual world (e.g. voice assistants, image analysis software, search engines, speech and face recognition systems)

[1] https://en.wikipedia.org/wiki/History_of_artificial_intelligence#Deep_learning,_big_data_and_artificial_general_intelligence:_2011-present

1

https://arxiv.org/abs/2105.01984

SOEN 691: Engineering AI-based Software Systems

The course

- **Familiarize** you with the **challenges and concepts** of engineering AI-based software systems
- Expose you to **common techniques used to engineer** AI-based software systems
- Get you to think **critically** about research in the area of AI-based software systems

Course Format

- Class:
  - Tuesdays 10:15 AM to 12:45 PM in-person (CU+Poly) and Zoom (UA+QU)
  - We will take a 5-minute break midway

- Classes are expected to be open in nature and include lively discussions of the materials being presented

Course Organization

- **Part 1**: Build necessary knowledge in the area of AI-based software systems

- **Part 2**: Study state-of-the-art in the engineering of AI-based software systems area
- Read and critique emerging research in the area

- **Contribute** to the state-of-the-art in AI-based software systems

Course Outcomes

- **Part 1**: Be able to understand and recognize the **terminology and basic building blocks** of AI-based software systems

- **Part 2**: Understand the main **components, challenges and techniques** pertaining to the engineering of AI-based software systems

- **Improve the state-of-the-art** and communicate these findings so others can benefit

Reference Materials

- We will mostly use a number of research papers that will be posted on the course website
- Course slides
- Reference text:
  - Building Intelligent Systems: A Guide to Machine Learning Engineering, 2018. Geoff Hulten

Tentative Outline

Week	Date	Topic
1	Sep. 3	Introduction & overview
2	Sep. 10	AI for Software Engineers
3	Sep. 17	Software requirements and architectures for AI-based systems
4	Sep. 24	Data validation and management
5	Oct. 1	Interpretation vs. Explanation
6	Oct. 8	**Project updates (student presentations)**
7	Oct. 22	Deployment and testing (MLOps) – Dr. Adams
8	Oct. 29	Technical Debt in AI-based systems
9	Nov. 5	LLMs and AI-based systems
10	Nov. 12	Continuous delivery – Dr. Mujahid
11	Nov. 19	**Quiz**
12	Nov. 26	**Project presentations (student presentations)**

Course Evaluation

Class participation	10%
Paper critiques & activities	20%
Project proposal	10%
Research project	40%
Quizzes and exams	20%

You must pass the quizzes and the course project to pass the course

Course Expectations

- Attend lectures and participate in discussions (Yes, we expect you to talk!)
- Do the assigned readings and assignments!
- Bring your ideas and concerns to class

Asking Questions and Communication

- Ask me or Hassan by email
- Schedule a in-person/Zoom meeting if needed
  - We will stay for 15 minutes after class (most classes)
- Ask in class
- Discuss with your classmates

Paper Critiques

- Most weeks, you will need to submit:
  - A summary of 1 of the papers (< half-page)
  - A critique of the other paper (1 page). The critique should include a summary, at least 3 weak points and at least 3 strong points

- Critiques and summaries are due at **noon (12 PM)** on Mondays **the day before class**

Course Project

- A large portion of the course mark is based on the course project
- Course projects are to be carried out in groups (3-5)
- You are free to do your project on any course-related topic, with instructor's approval

Course Project

- You will present a progress update in week 6
- You will hand in a project proposal in week 6 (3 pages, IEEE format)
- You will present the project in week 12 and hand in the final report at the end of week 12 (10 pages, IEEE format)

Lateness Policy for All Course Deliverables

NO LATE DELIVERABLES!

NO LATE DELIVERABLES!

Academic Integrity and Cheating

- Cheating, plagiarism and other forms of academic fraud are taken very seriously by the University, the Faculty, and the teaching staff
- Examples:
  - Submitting the work of another person as your original work
  - Incorporating others work in your work and not referencing it
  - It is permitted and encouraged to discuss with your peers but **NOT** permitted to copy their solutions. Both parties will be penalized.

Homework

- Have a look at the topic list on the course's Moodle page
  - 1. Submit a list of 3 topics areas that you would like to do your project on
    - For each topic, give a list of 3 relevant papers
  - 2. Submit a list of 4-6 names of people you want to do your project with. The names need to be from at least one other University.
  - Due Monday, Sep. 9 at noon on the course webpage
  - **Make sure to put your name, student id and date on all pages. Submit as a PDF file.**

References

- Textbook
  - Software Engineering by Ian Sommerville (10th edition)
  - Available at Concordia Library
  - https://www.darpa.mil/news-events/2018-07-20a
  - https://www.ibm.com/cloud/learn/what-is-artificial-intelligence
  - https://en.wikipedia.org/wiki/Artificial_intelligence
  - https://www.britannica.com/technology/artificial-intelligence