Software Engineering vis-à-vis Artificial Intelligence: Whither Research

FTIR Research Week 2019

Shamsul Sahibuddin

Universiti Teknologi Malaysia

Agenda

- **Prologue**
- **Software Engineering**
- **Artificial Intelligence**
- **SE and AI**
- **Summary**

vis-à-vis

/ˌvēzəˈvē/

- *preposition*

  in relation to; with regard to.

  "many agencies now have a unit to deal with women's needs vis-à-vis employment"

- *adverb*

  in a position facing a specified or implied subject.

  "he was there **vis-à-vis with** Miss Arundel"

- *noun*

  a person or group occupying a corresponding position to that of another person or group in a different area or domain; a counterpart.

whither

[**hwi*th*** -er, **wi*th*** -]

- *adverb*

  to what place? where?
  to what end, point, action, or the like? to what?

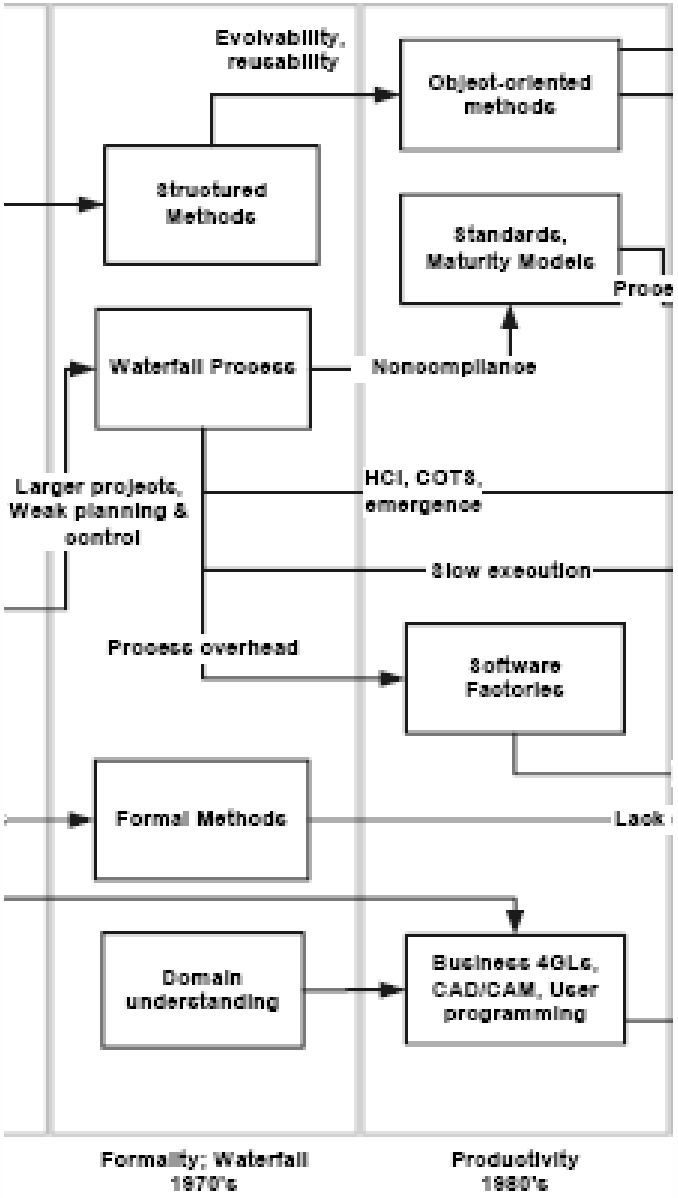
- *conjunction*

  to which place.
  to whatever place.

Agenda

- **Prologue**
- **Software Engineering**
- **Artificial Intelligence**
- **SE and AI**
- **Summary**

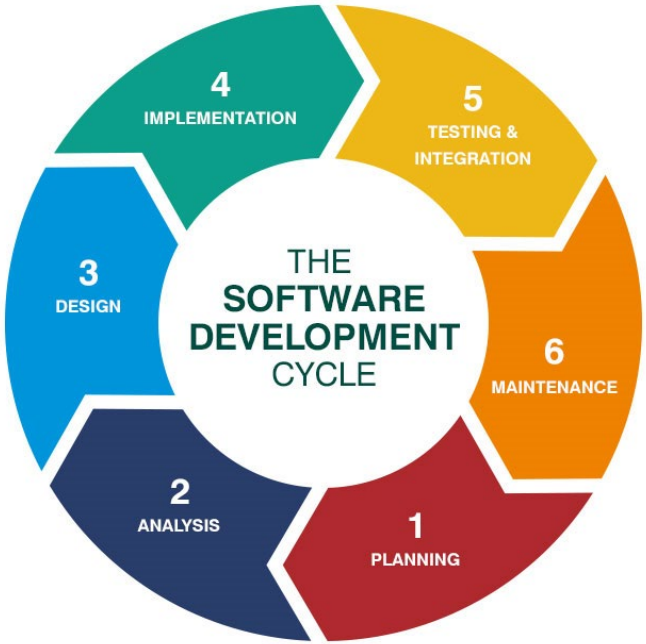
Trend

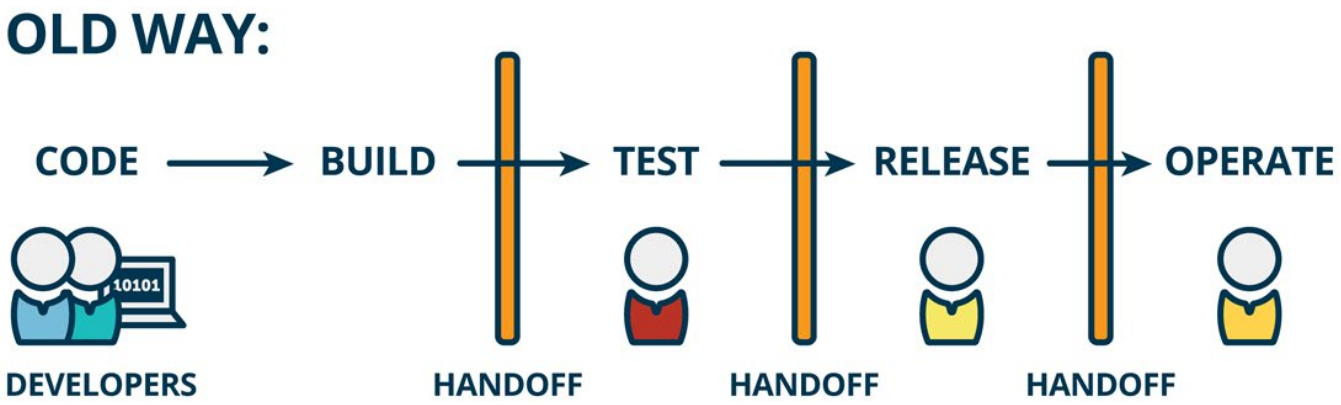

Evolvability, reusability
Object-oriented methods
Structured Methods
Standards, Maturity Models
Proce
Waterfall Process
Noncompliance
Larger projects, Weak planning & control
HCI, COTS, emergence
Slow execution
Process overhead
Software Factories
Formal Methods
Lack
Domain understanding
Business 4GLs, CAD/CAM, User programming
Formality; Waterfall 1970's
Productivity 1980's

70s and 80s

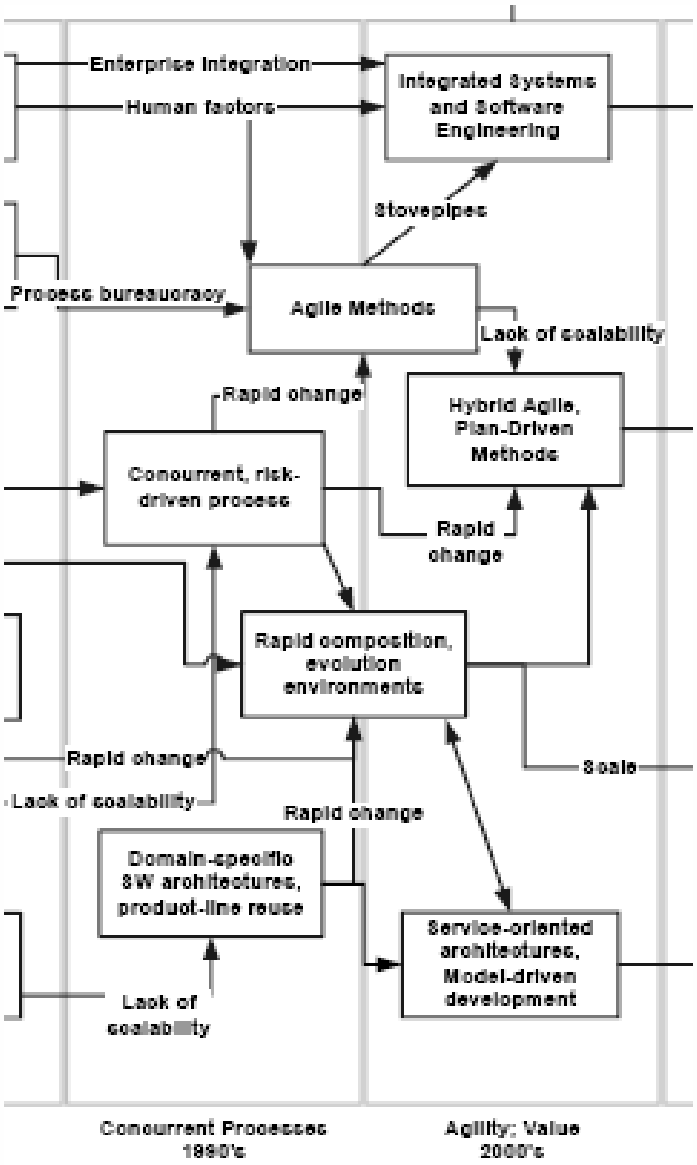
- Formality
  - Structured methods
  - Waterfall process
  - Formal methods
  - Domain understanding
- Productivity
  - Object oriented methods
  - Standards, maturity models
  - Software factories
  - Business 4GLs, User programming

Old Way


4
IMPLEMENTATION
5
TESTING &
INTEGRATION
3
DESIGN
THE
SOFTWARE
DEVELOPMENT
CYCLE
6
MAINTENANCE
2
ANALYSIS
1
PLANNING


OLD WAY:
CODE
BUILD
TEST
RELEASE
OPERATE
10101
DEVELOPERS
HANDOFF
HANDOFF
HANDOFF

Trend


Enterprise Integration
Human factors
Integrated Systems and Software Engineering
Stovepipes
Process bureaucracy
Agile Methods
Lack of scalability
Rapid change
Hybrid Agile, Plan-Driven Methods
Concurrent, risk-driven process
Rapid change
Rapid composition, evolution environments
Rapid change
Lack of scalability
Rapid change
Scale
Domain-specific SW architectures, product-line reuse
Service-oriented architectures, Model-driven development
Lack of scalability
Concurrent Processes 1990's
Agility; Value 2000's

00's

Agility

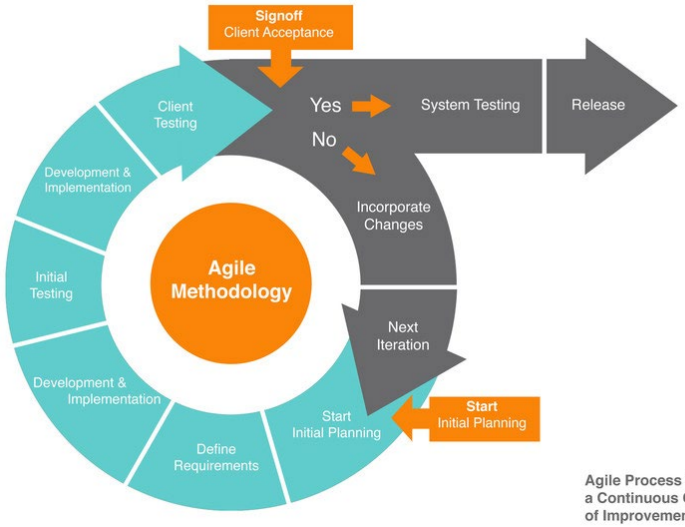
Agile methods

Rapid composition, evolution environment

Integrated systems and software engineering

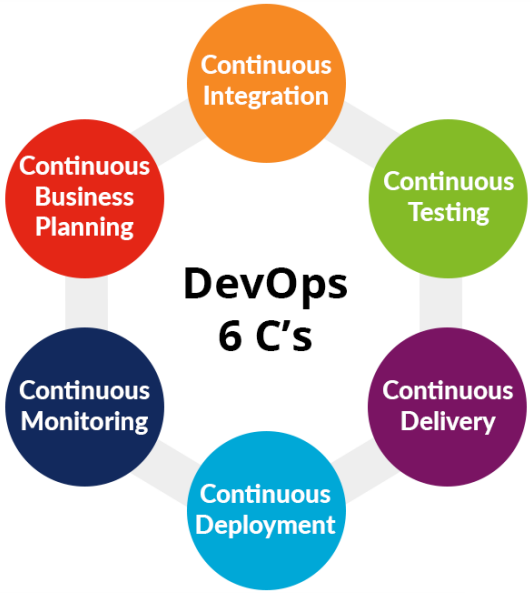
Hybrid agile, plan driven methods

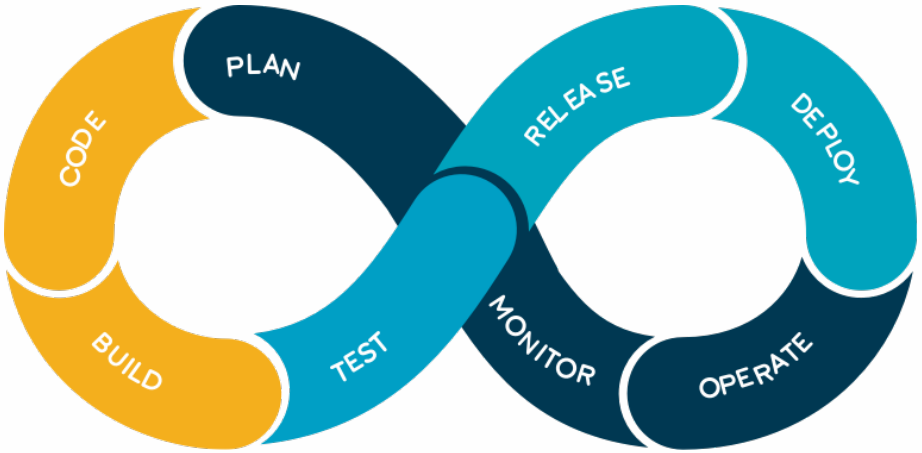
Service oriented architecture, model driven development

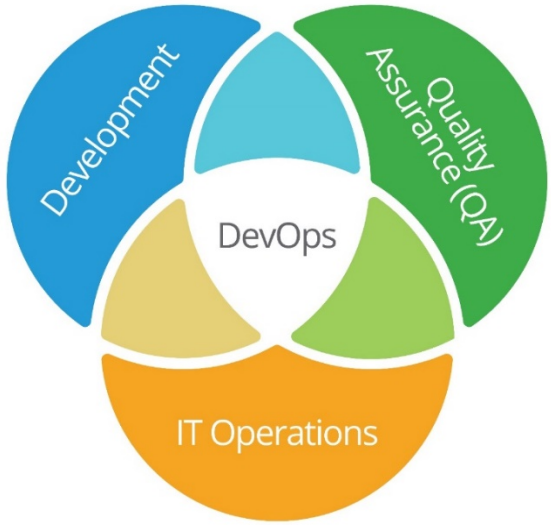


Agile Process is a Continuous Cycle of Improvement

DevOps


Continuous Integration
Continuous Testing
Continuous Delivery
Continuous Deployment
Continuous Monitoring
Continuous Business Planning
DevOps
6 C's


PLAN
CODE
BUILD
TEST
RELEASE
DEPLOY
MONITOR
OPERATE


Development
Quality Assurance (QA)
DevOps
IT Operations

Continuous Software Engineering:

Lean Startup - method

DevOps

Continuous Delivery

Continuous Experimentation

Defense Innovation Board Ten Commandments of Software

Version 0.14, last modified 15 April 2018

1. Make computing, storage, and bandwidth abundant to DoD developers and users.

2. All software procurement programs should start small, be iterative, and build on success – or be terminated quickly.

3. Budgets should be constructed to support the full, iterative life-cycle of the software being procured with amount proportional to the criticality and utility of the software.

4. Adopt a DevOps culture for software systems.

5. Automate testing of software to enable critical updates to be deployed in days to weeks, not months or years.

6. Every purpose-built DoD software system should include source code as a deliverable.

7. Every DoD system that includes software should have a local team of DoD software experts who are capable of modifying or extending the software through source code or API access.

8. Only run operating systems that are receiving (and utilizing) regular security updates for newly discovered security vulnerabilities.

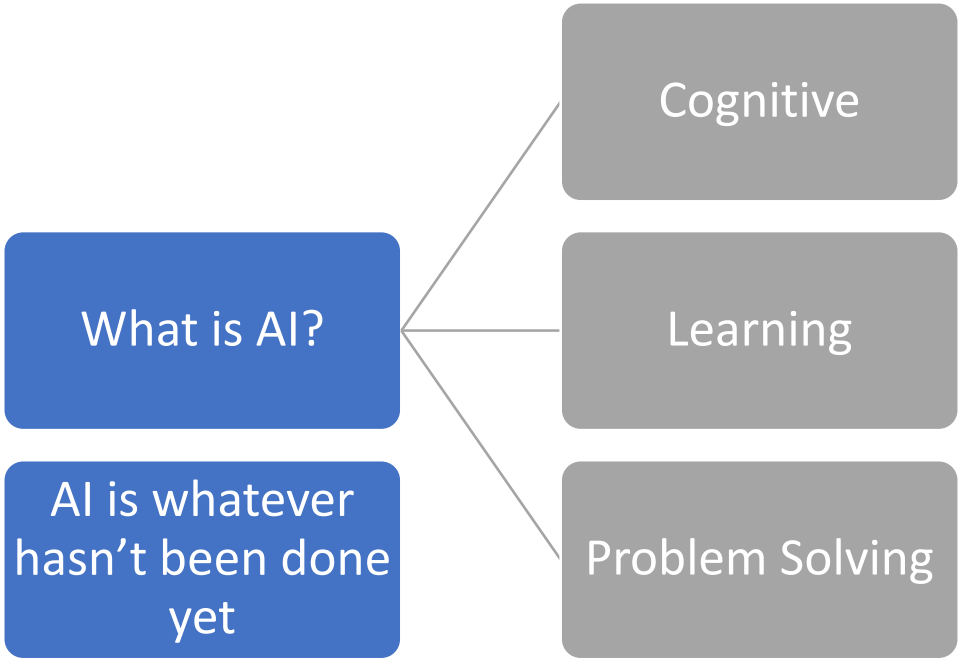
9. Data should always be encrypted unless it is part of an active computation.

10. All data generated by DoD systems - in development and deployment - should be stored, mined, and made available for machine learning.

Agenda

- **Prologue**
- **Software Engineering**
- **Artificial Intelligence**
- **SE and AI**
- **Summary**

AI


What is AI?
AI is whatever hasn't been done yet
Cognitive
Learning
Problem Solving

AI

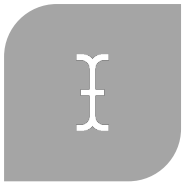
- Types
  - Analytical
    - Cognitive
  - Human-Inspired
    - Cognitive, EI
  - Humanized AI
    - Cognitive, Emotional, Social

Problems

- Reasoning, problem solving
- Knowledge representation
- Planning
- Learning
- NLP
- Perception
- Motion and manipulation
- Social intelligence
- General intelligence

Approaches

CYBERNETICS AND BRAIN SIMULATION



SYMBOLIC

SUB-SYMBOLIC

STATISTICAL LEARNING

INTEGRATED APPROACHES

Tools

- Search and optimization
- Logic
- Probabilistic methods for uncertain reasoning
- Classifiers and statistical learning methods
- Artificial neural networks

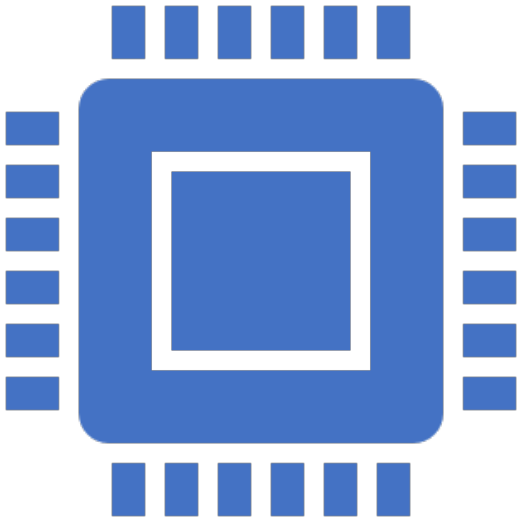
Agenda

- **Prologue**
- **Software Engineering**
- **Artificial Intelligence**
- **SE and AI**
- **Summary**

SE and AI

Horneman, Mellinger, and Ozkaya, 2019

- The availability of ML libraries and off-the-shelf solutions sometimes gives the impression that implementing AI-enabled software systems (AI systems) is easy.
- Developing viable and trusted AI systems that are deployed to the field and can be expanded and evolved for decades requires significant planning and ongoing resource commitment.



SE and AI

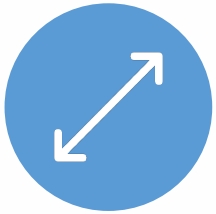
- AI system is a software-intensive system.
- The established principles of designing and deploying quality software systems that meet their mission goals on time apply to engineering AI systems.
- Teams should follow modern software and systems engineering practices

SE and AI

Engineering teams will need to architect AI systems for inherent uncertainty in their components, data, models, and output.

Engineering teams will need to adapt to managing different rhythms of change. The rate of change in AI systems is not consistent throughout the system. Data and models may change very frequently, which may or may not imply changes to the rest of the system.

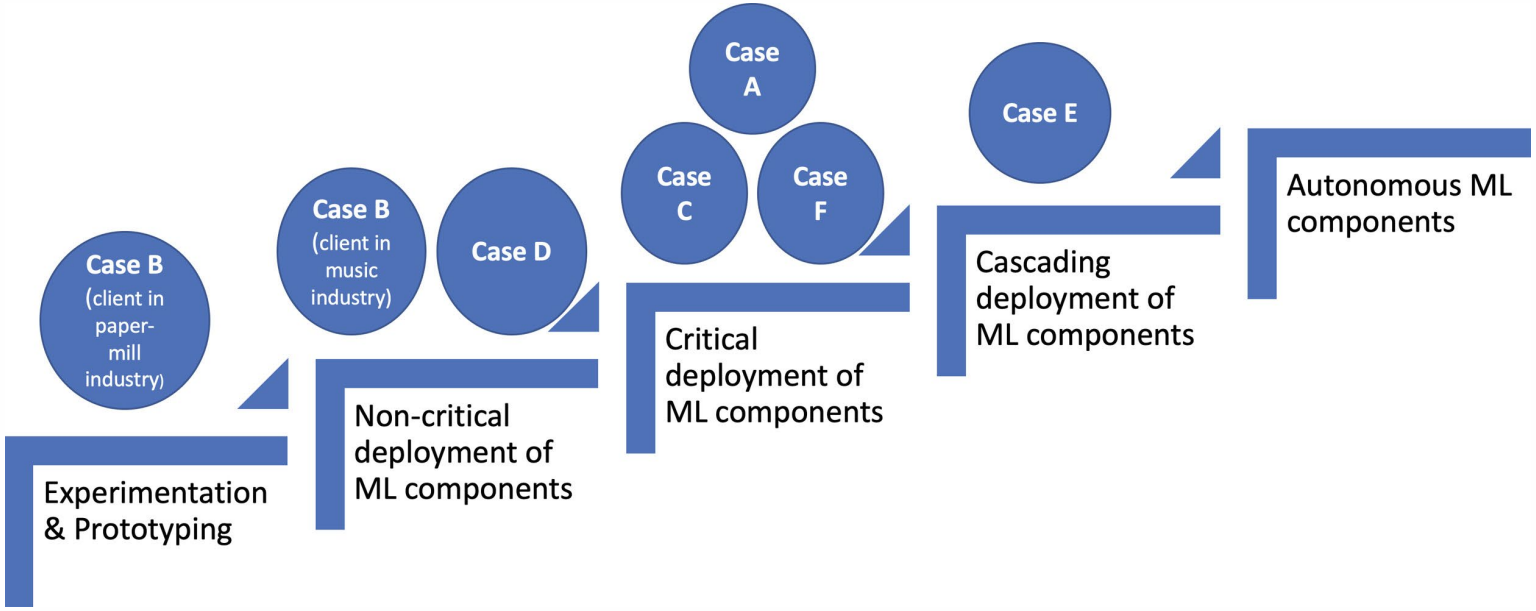
Managing data will require more resources both up-front and throughout the life of the system.



Verifying, validating, and securing AI systems will need to account for ambiguity as well as increased attack surface due to frequently changing data and underlying nature of models.

- Re-engineering software technologies
  - Software composability
    - Composability of systems involving AI-based components will introduce a new layer of complexity
  - Software and data lifecycles
    - ML components are dependent on data. System changes to code, data and models are interdependent and are likely to require new types of source and dependency control. As the number of possible ML hyper-parameters is exponential, difficulties may arise in maintenance and updates.
  - Quality assurance
    - AI-enabled software systems introduce a host of new quality assurance and security challenges

Software and AI
NESSI (Networked European Software and Services Initiative), 2019

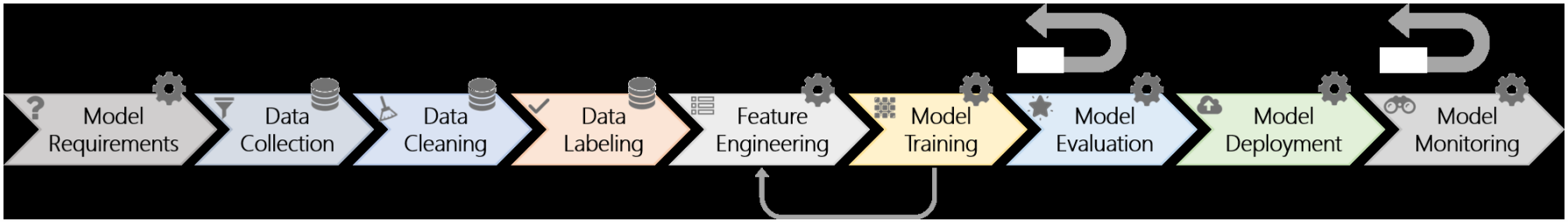


SE Challenges for ML Systems

Lwakatare, Raj, Bosch, Holmstr¨om Olsson, Crnkovic, 2019

Software Engineering for Machine Learning: A Case Study

Amershi *et al.*, 2019



Issues in Deep Learning Engineering

John, 2019

- Rapidly evolving deep learning frameworks
- Data management
- Configuration management
- Testing deep learning systems
- Deployment of deep learning systems

Agenda

- **Prologue**
- **Software Engineering**
- **Artificial Intelligence**
- **SE and AI**
- **Summary**

AI Engineering

Horneman, Mellinger, and Ozkaya, 2019

1

Ensure you have a problem that both can and *should* be solved by AI.

2

Include highly integrated subject matter experts, data scientists, and data architects in your software engineering teams.

3

Take your data seriously to prevent it from consuming your project.

4

Choose algorithms based on what you need your model to do, not on their popularity.

AI Engineering

- **Secure AI systems by applying highly integrated monitoring and mitigation strategies.**
- **Define checkpoints to account for the potential needs of recovery, traceability, and decision justification.**
- **Incorporate user experience and interaction to constantly validate and evolve models and architecture.**
- **Design for the interpretation of the inherent ambiguity in the output.**

AI Engineering

- **Implement loosely coupled solutions that can be extended or replaced to adapt to ruthless and inevitable data and model changes and algorithm innovations.**
- **Commit sufficient time and expertise for constant and enduring change over the life of the system.**
- **Treat ethics as both a software design consideration and a policy concern.**

Thank You

Prof. Dr. Shamsul Sahibuddin
Pro-Vice-Chancellor
(UTM Kuala Lumpur Campus)
shamsul@utm.my