Managing and Developing Modern Software

Saeed Siddik

Assistant Professor

IIT University of Dhaka

Evolution on Software Engineering

- SE 1.0 – The Code-Centric Era
  - Classical Software Engineering
- SE 2.0 – The Data, Service and Collaboration Era
  - DevOps and ML driven training-prediction
- SE 3.0 – The AI-Driven and Autonomous Era
  - LLMs and AIware in software automation

SE 1.0 – The Code-Centric Era (Codeware)

- Core Idea: Software is programmed manually by humans using traditional languages such as C, C++, and Java.
- Process: Developers explicitly write rules, algorithms, and logic that define how a system behaves.
- Examples: Website, databases, mobile apps and desktop applications.
- Characteristics:
  - Logic-based, deterministic, and rule-driven.
  - High manual effort and limited learning capability.

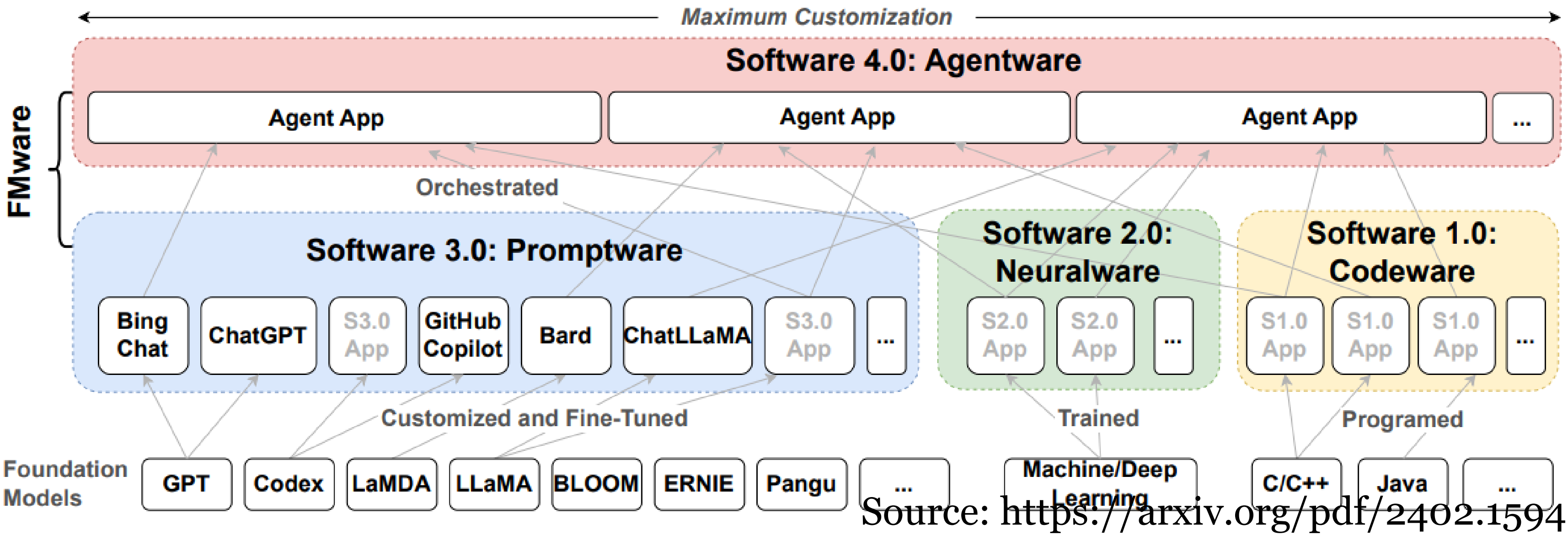
SE 2.0 – The Process, Service and Collaboration Era (Neuralware)

- Core Idea: Software is trained rather than explicitly programmed.
- Process: Systems learn patterns and behaviors from data through machine learning and deep learning models.
- Examples: Image recognition, speech recognition, and recommendation systems.
- **Characteristics:**
  - Learning-based, probabilistic, and data-driven.
  - Developers design neural architectures and provide datasets instead of direct code logic.

SE 3.0 – The AI-Driven and Autonomous Era (Promptware)

- Core Idea: Software is customized and fine-tuned through foundation models (FMs) such as GPT, using prompts instead of training from scratch.
- Process: Developers and users interact with pre-trained AI-models through prompt engineering and fine-tuning.
- Examples: ChatGPT, GitHub Copilot, Bard, and LLaMA.
- Characteristics:
  - Built on top of massive pre-trained models (foundation models).
  - Bridges AI with software engineering via prompt-based design.

Agentware and its relation to prior software generations.



Source: https://arxiv.org/pdf/2402.15943

Software Engineering for Machine Learning: SE2.0

- Data-Centricity: Focus shifts from managing code versioning to data quality, provenance, and pipelines.
- Specialized Workflow: Iterative ML workflow to be integrated into existing Agile or DevOps processes.
- New Requirements: Unique challenges of ML, such as model evolution/debugging, component entanglement

Fundamental Differences from Traditional SE

- Data is Primary and Complex
  - SE: Managing and versioning software *code* is the focus.
  - ML: Discovering, managing, and versioning the data is much more complex and difficult than with code. Data provenance is critical for model reproducibility.

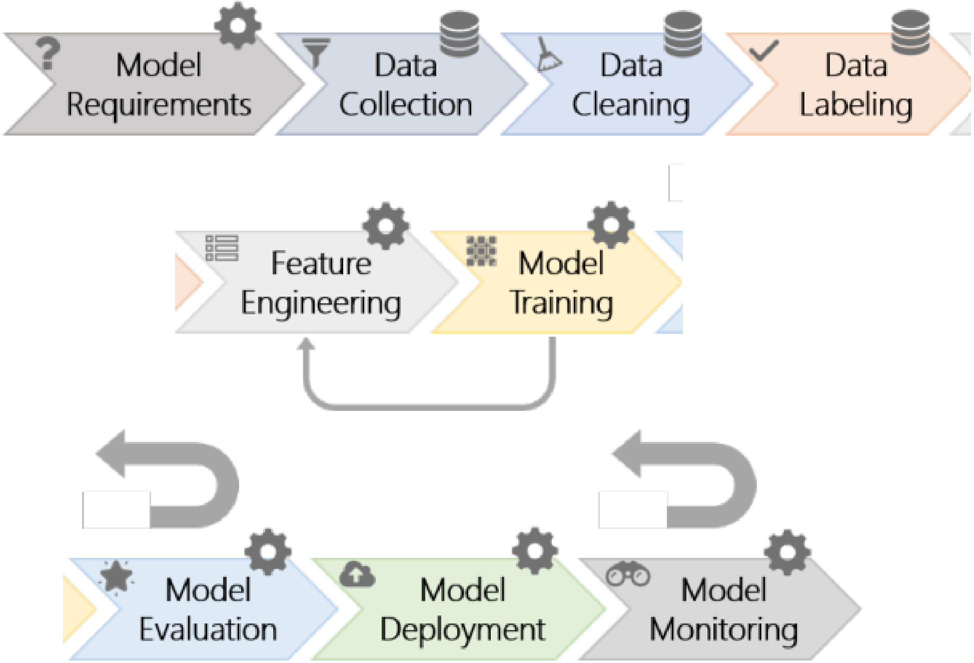
Fundamental Differences from Traditional SE

- New and Different Skill sets Required
    - SE: Traditionally relies on Program Managers, Developers, and Testers.
    - ML: Model customization and reuse require not only software engineering skills but also deep machine learning knowledge to build, evaluate, and tune models from scratch.

Fundamental Differences from Traditional SE

- Component Entanglement
  - SE: Components are handled as distinct, modular units.
  - ML: AI components are more difficult to isolate. Models can be "entangled" in complex ways, causing them to affect one another during training and tuning, even if they were intended to be isolated. This leads to problems like non-monotonic error behavior.

The machine learning workflow



The ML Workflow (The Nine Stages)

Preparation

1. Model Requirements: Define feasible and useful features; select appropriate model types

Data

2. Data Collection: Discover, source, and integrate available datasets or collect new ones
3. Data Cleaning: Remove inaccurate or noisy records
4. Data Labeling: Assign ground truth labels (often manual or crowd-sourced)

The ML Workflow (The Nine Stages)

Modeling

5. Feature Engineering: Extract and select informative features
6. Model Training: Train and tune the chosen model on clean data and labels
7. Model Evaluation: Evaluate the model on test/safeguard datasets using metrics and human evaluation

Operations

8. Model Deployment: Deploy the inference code to the target device(s)
9. Model Monitoring: Continuously monitor for errors during execution

Essential Challenges and Best Practices (SE 2.0)

- Data Availability, Collection, Cleaning, and Management
  - Data Quality and Availability
  - Data Provenance and Evolution
  - Data aggregation
  - Feature extraction and synthesise

Essential Challenges and Best Practices (SE 2.0)

- End-to-End Pipeline Support
  - Iterative and Experiment
  - Integration and Automation
  - User Experience and Monitoring
  - Rich Dashboards to display the value the model

Essential Challenges and Best Practices (SE 2.0)

- Model Evolution, Evaluation, and Deployment
  - Rigorous Testing and Evaluation
    - Systematic Evaluation
    - Experiment Score Cards
    - Automated Test Sets
  - Ensuring Quality and Reliability
    - Human in the Loop
    - Integration and Versioning

Essential Challenges and Best Practices (SE 2.0)

- Model Debugging and Interpretability
  - Understanding Model Failures
  - layered, and tiered software architecture modularization
  - visualization techniques

Essential Challenges and Best Practices (SE 2.0)

- Compliance (Fairness, Accountability, Transparency, Ethics - FATE)
  - Ethical AI Development
  - align with team practices and behavior

FATE (Fairness, Accountability, Transparency, and Ethics)

- **Fairness**: Focuses on mitigating algorithmic bias to ensure that ML models do not discriminate against specific individuals or groups (e.g., race, gender, or age) and provide equitable outcomes across populations.
- **Accountability**: Defines the clear ownership and responsibility for an AI system's actions, including failures and potential harms.
- **Transparency**: Relates to making the process of AI decision-making understandable to stakeholders by documenting the data sources (provenance) and model architecture.
- **Ethics**: Encompasses broader moral principles guiding AI use, ensuring the system's deployment adheres to an organization's and society's values, particularly concerning user well-being, privacy, and prevention of misuse.

End of SE2.0