**15. AI-Driven Development**

Shivakumar R Goniwada[**1**](https://learning.oreilly.com/library/view/cloud-native-architecture/9781484272268/html/511610_1_En_15_Chapter.xhtml#Aff2)

(1)

Bangalore, India

During the cloud and cloud native age, organizations are focusing on leveraging AI methodologies, best practices, and enhanced tools and technologies during the software engineering lifecycle and applying them to build or augment enterprise systems.

There is a lot of interest growing in AI-focused methodologies for creating and distributing the development of application solutions.

In modern-day software engineering, you need a separate methodology, process, and toolset with automation for AI development. To improve the process, various industries and enterprises are embedding AI into their software engineering lifecycle to make the process smarter, automated, and efficient. AI-enabled tools can optimize engineering tasks by automating the end-to-end engineering lifecycle.

With these AI-driven development tools, the engineering team can develop an AI-powered automation process without involving specific experts.

In this chapter, I cover the following:

* AI methodologies
* AI solutions to solve industry use cases
* AI tools and best practices
* AI governance
* AI and ML in DevOps cycle

**Introduction**

The evolution of AI has changed the way organizations and engineers are approaching software systems and development. Artificial intelligence (AI) and machine learning (ML) are good options for enhancing the software engineering lifecycle output.

According to Gartner, by 2022, at least 40 percent of new software engineering projects will have an AI-driven virtual engineer on the team. This is because AI and ML-backed tools are already on the market for development, source code generation, and testing, and they can be integrated as part of the DevOps pipeline. IDC estimates that worldwide spending on AI is expected to double in four years, reaching $110 billion in 2024.

Currently, humans focus on everything including solving business problems, identifying quality attributes, testing outcomes, etc. Instead of doing everything, why can’t we just focus on solving business problems and letting the AI- and ML-based tools automate code generation and proactively identify failures?

Adopting an AI strategy helps enterprises to start on a journey toward integrating AI into their fabric, not just by implementing AI solutions but by developing an AI-powered architecture to embrace AI at the core.

AI accelerates the traditional software development techniques and eases your coding, reviewing, and testing process. It creates a scalable and efficient workflow to drive productivity and reduce time to market.

Human engineers are translators and engage in conversation with clients and other stakeholders when testing and with other AI-enabled engineers (using AI-based tools). Before engaging AI-enabled engineers, you need to train them thoroughly. Test-driven development, hypothesis-driven development, and behavior-driven development help you to train your AI colleagues.

Answering the following questions will position your organizations to capture sustainable long-term value through AI:

* What role do we want AI to play in our organization? How will important aspects of our business (for example, skilled resources, customer experience) change?
* How will we get the most benefit from AI? Where are the most impactful opportunities in our business, and will they be enough to justify the investment?
* What action do we need to take to establish a foundation that makes AI practical, effective, and responsible? How do we manage these changes?
* How can our organization sustain the shift toward embracing AI? How do we increasingly make AI core to the performance of our business?

**Unique AI Challenges**

Adopting AI introduces unique challenges that require comprehensive AI thinking to embrace:

* *Risk of unintended consequences*: AI represents a major technological advancement with tremendous potential. However, the universal and standard procedures are not yet available to assure the outcome.
* *Bridge unconventional organization gaps*: Successful AI needs to be a joint effort across every entity of an organization.
* *Specialized talent*: The organization requires specialized talent, training, and an AI career path.
* *Mature data capabilities*: Despite maturity in data and data science, there remains continued gaps in the quality of data to require effective training and operation of AI solutions.
* *Culture*: The organization needs to embrace AI by adopting a culture of AI.

**Why AI-Driven Development?**

The reasons to use AI-driven development are as follows:

* AI-driven development refers to the tools, process, technologies, and best practices used for embedding AI into software applications and for using AI tools to develop AI-enhanced solutions.
* There is a current need for software engineering to be able to operate AI-enhanced technologies independently.
* AI-driven development provides engineering with an ecosystem of AI algorithms and models, as well as development tools tailored to integrate AI capabilities.
* The AI-driven development approach is well suited to minimizing prediction errors, when there are many data points and when there are many alternatives.
* Engineers can infuse AI-powered capabilities into application development without involving a data scientist.
* AI methodologies, reference architectures, and best practices can use as reference for AI-driven development.

**AI-Driven Principles at a Glance**

AI principles set guardrails to help enterprises address the unique challenges associated with pursuing AI responsibly. Google and Microsoft AWS have defined and adopted the following principles:

* *Be socially beneficial*: The expanded reach of new technologies increasingly touches society as a whole.
* *Avoid creating or reinforcing unfair bias*: AI algorithms and datasets can reflect, reinforce, or reduce unfair bias.
* *Be built and tested for safety*: Use strong safety and security practices to avoid untended results.
* *Incorporate privacy design principles*: Incorporate privacy principles in the development.
* *Fairness*: Throughout the lifecycle, AI systems should be inclusive and accessible.
* *Anticipate the future*: AI applications can produce granular insights into what customers and markets want.
* *Act autonomously*: An AI application provides value by automating existing manual processes by enabling the autonomous operation of the business.
* *Detect invisible*: AI can manage operations that humans cannot, and AI application should take advantage of this situation in a complex environment.

**Approach to AI**

There are many approaches to AI. A successful AI adoption can start in many ways.

* *Top down*: Leadership is fully involved in defining the AI value and roadmap and moving quickly from strategy to MVP.
* *Bottom up*: Start with an MVP that will incrementally prove AI’s value as other initiatives are considered and implemented.
* *Part of a bigger picture*: AI is considered as part of the broader organization.
* *Inorganic*: Acquire AI startups or leverage partners to augment the AI.

**AI Governance**

AI governance is about AI being explainable, transparent, and ethical. The AI governance is to define the key mechanisms for executing AI use cases and deploying them across the organization, and the governance framework helps you to drive the partnership between your organization, clients, and other stakeholders. As part of the governance, you need to outline the AI roles and responsibilities of resources within your organization’s structure.

The governance framework outlines the decision-making process for key activities and determines how opportunities are identified, approved, delivered, and scaled across the enterprises and how key decisions are made around AI.

**AI Framework**

The AI framework establishes trust in the AI architecture and helps you to continue monitoring the system. Use the following three-step approach for all your AI deliverables:

* *Govern*: Create an internal governance process, as explained next, which is anchored to industry and societal shared values, regulations, ethical guardrails, and accountability. Promote clarity around decisions.
* *Design*: Architecture and design AI with trust by design. Empower project teams to understand and address bias issues.
* *Monitor*: Monitor and audit regularly against key-value metrics, including concerning algorithmic accountability and cybersecurity.

**AI Governance Measurement**

Lack of measurement will be a weakness of your organization because these cannot be transferred to and incorporated into processes, systems, and platforms. The AI measurement is common for all organizations through regulations from the AI body, but each organization can measure how AI is delivered, what direction AI projects progress in, etc. These measurements are captured by an audit of the AI projects, accountability in AI projects, the time they take to complete, security considerations, etc.

**Governance Process**

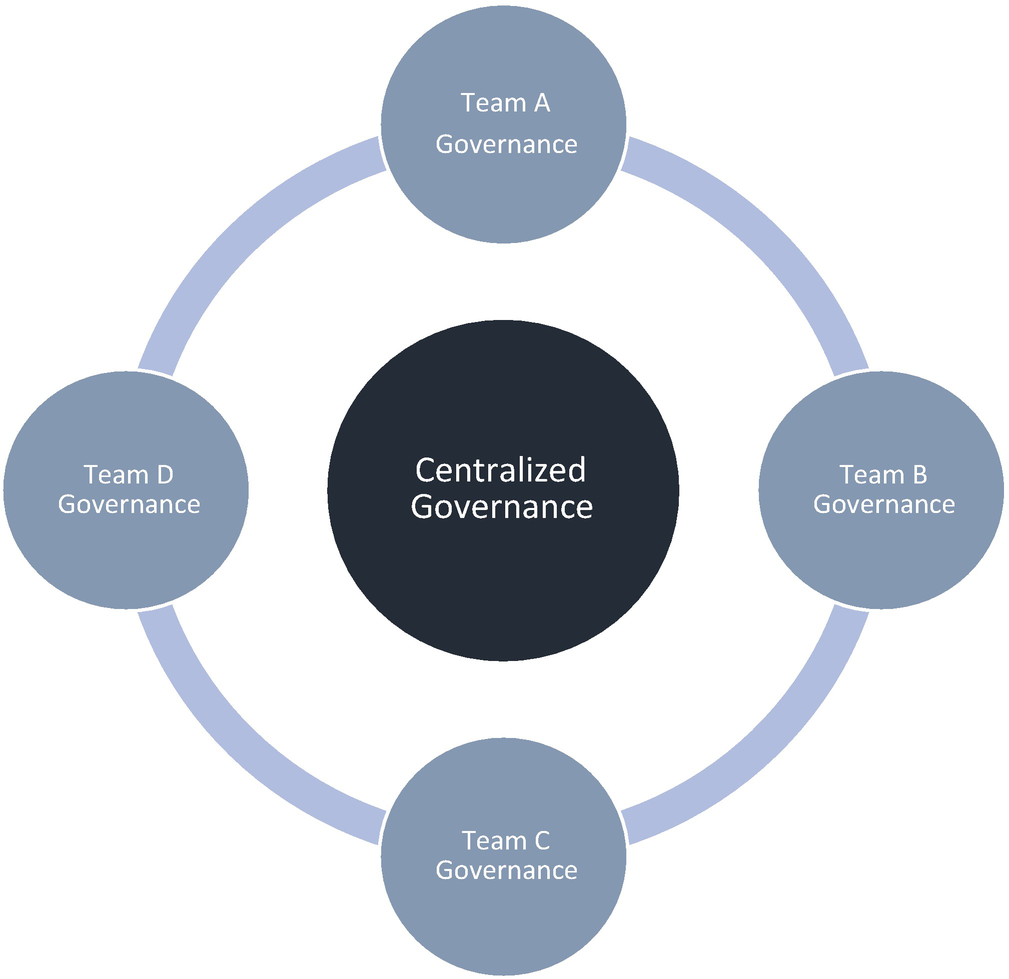
The following list highlights the process that will take place in the event that a deployed AI solution behaves in an unexpected way:

* When monitoring the AI solutions in a project, the center of excellence (CoE) must highlight any red flags or anything out of the ordinary and report it to the responsible AI board.
* A review board will evaluate the AI service that has been flagged to review it and recommend a plan of action.
* The CoE will begin resolving the AI solutions that were flagged according to the approved course of action with support from the AI project team.
* Community and knowledge sharing boards work to analyze and identify the controls in place to prevent repetition.

**Governance Model**

Based on the previous principles, the best model for AI development is the hub and spoke model because it enables controlled growth and encourages autonomy within the business sector. You need innovation at all levels for AI, so this model allows for rapid innovation/sharing of ideas across your organization while centralizing the AI research and best practices. According to the cloud native principles, organizations should adopt a decentralization approach, but AI development requires a centralized decentralization approach. This helps to keep key decisions central and decentralizes the solution implementation.

As shown in Figure [15-1](https://learning.oreilly.com/library/view/cloud-native-architecture/9781484272268/html/511610_1_En_15_Chapter.xhtml#Fig1), the hub and spoke model attracts, develops, and retains scarce talent, allowing for the flexible allocation of resources to keep resources challenging and fresh.



***Figure 15-1***

Hub and spoke governance

In the beginning, the big centralized hub starts to increase the use of AI governance across teams in an enterprise, because there isn’t enough skills and maturity across spokes to decentralize governance. AI matures in an organization when skilled talent and teams begin to empower business and technical teams to contribute to delivering AI solutions. This leads to the allocation of AI talent and a balance of maturity between the hub and spokes. Once the organization reaches the maturity stage, the AI CoE remains a small group that coordinates the AI activities across enterprises.

**How to Train AI-Enabled Frameworks?**

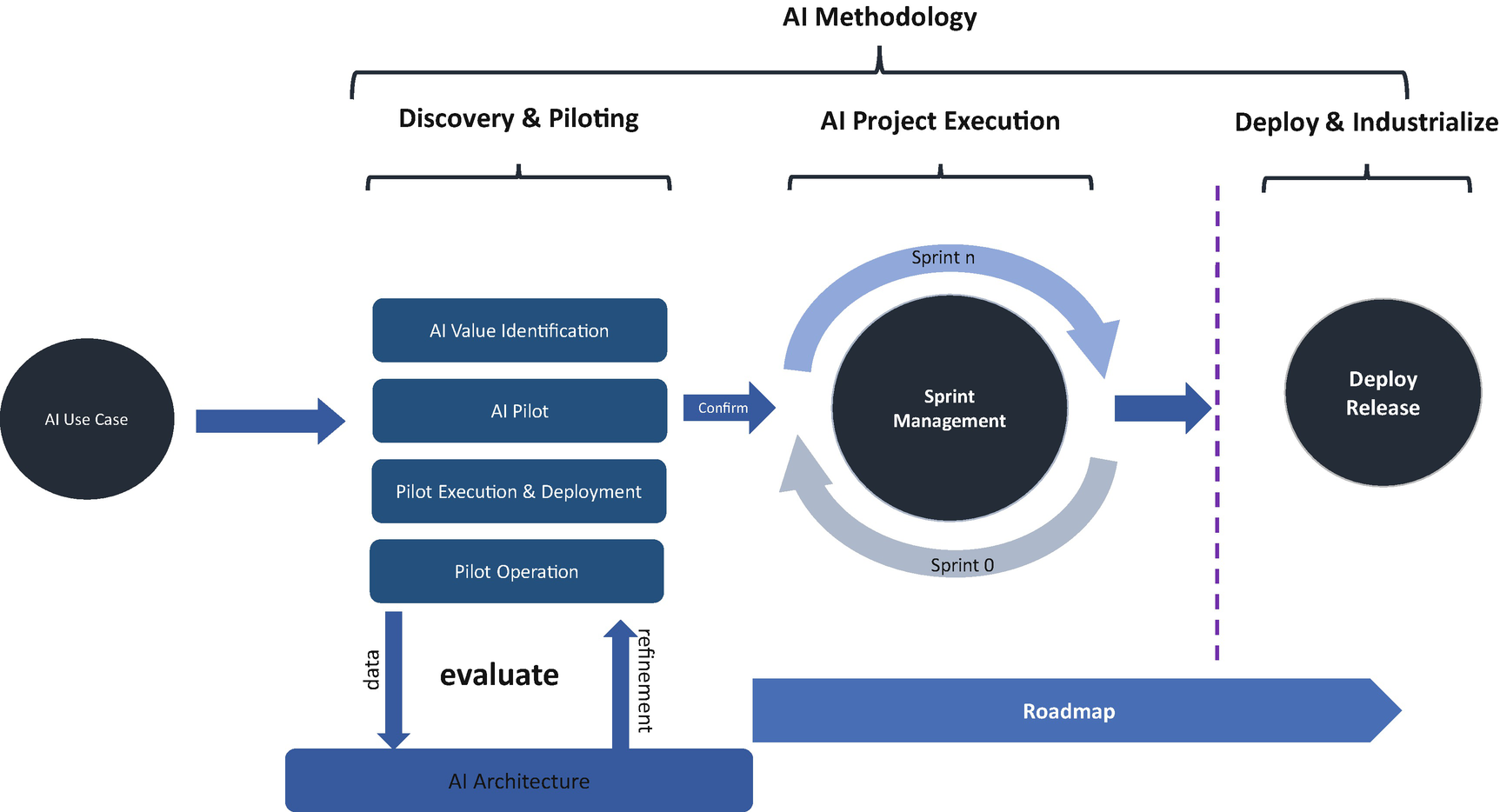
AI and ML enable tools to learn from data rather than explicit programming. AI and ML algorithms ingest data to train the algorithm model and the behavior of these tools is based on the kind and quality of data used for the training. The better-quality data you ingest, the wiser the tool becomes. Once the base model is trained, then you can ingest in real time to learn more precisely. The accuracy of AI and ML tools is based on training processes and automation, which are part of ML.

The TDD, BDD, and HDD techniques provide a framework to train AI in software engineering that can carry out repetitive works.

In TDD, engineers first write a test, and then they write the code to make the test case pass. Use these test cases as data to train AI models. AI enables a system to learn from data rather than through explicit programming. Algorithms ingest training data and produce models based on the test cases and test data; similarly, you can use hypotheses and behavior to train AI models.

**AI-Driven Methodology**

As shown in Figure [15-2](https://learning.oreilly.com/library/view/cloud-native-architecture/9781484272268/html/511610_1_En_15_Chapter.xhtml#Fig2), the AI methodology focuses on identifying use cases and piloting, and it industrializes the AI implementation. It enables bringing all AI projects and execution together with lightweight deliverables and a flexibility of agile practices in an AI environment.



***Figure 15-2***

AI-driven methodology

**AI Use Cases**

AI uses are identified by synthesizing various perspectives from the external research organization, brainstorming workshops, and stakeholder discussions.

* Opportunities can be adapted from well-established AI use cases from different industries.
* Discuss with respective stakeholders the opportunities to apply AI to solve some of its pain points.

You can use these sample questions during your discussion with stakeholders:

* What are your challenges that you wish you could spend less time on?
* What are some repetitive tasks that you and your team do daily?
* Have you identified any AI use cases?
* What types of data do you think you have that no one else does?

**Discovery and Piloting**

In the AI value identification process, you will understand the goals of AI and expectations, analyze the use cases, and select the relevant AI technologies and algorithms. You’ll create a prototype and prepare a benefit-cost analysis with critical success factors. You’ll fine-tune the AI architecture and algorithms during the discovery and piloting phase.

During the analysis of the use case, schedule a workshop with stakeholders to understand the goals and objectives and to collect metrics and data requirements. Data needs to be gathered and analyzed in order to model specific prioritized use cases. The success of AI will be based on the quality of data collection.

Based on the list of requirements and use cases, the next step is to assess the right algorithm and technology. During this phase of development, the critical factor is to identify the right algorithm and technologies. After reviewing the use cases, list the detailed tasks and functionalities that need to be implemented with AI. Decide on the type of algorithm you would like to apply.

* *Supervised learning*: The computer makes a prediction based on general rules for mapping inputs and outputs, and the model is trained from a set of labeled data.
* *Unsupervised learning*: No labeled data is given to the learning algorithm, leaving it on its own to find structure in its input.
* *Reinforcement learning*: The computer interacts with a dynamic environment in which it must perform certain goals.

When choosing an algorithm, you need to understand some extra features for the algorithms, such as accuracy, training time, linearity, and number of parameters.

Once you have finalized these steps, the next step is to create a proof of concept (PoC) by leveraging the AI technology and algorithm. Once the PoC is ready, create a benefits case for industrialization. In the piloting phase, train and test the AI technologies. During this time, the model strength need to be captured in terms of parameters like accuracy, precision, and recall. This might differ for various algorithms. For example, linear regression gives the model strength in terms of the R-Squared and adjusted R-Squared values.

Once the PoC is accepted, you need to create a roadmap to outline the key initiatives required to support industrialization.

**AI Project Execution**

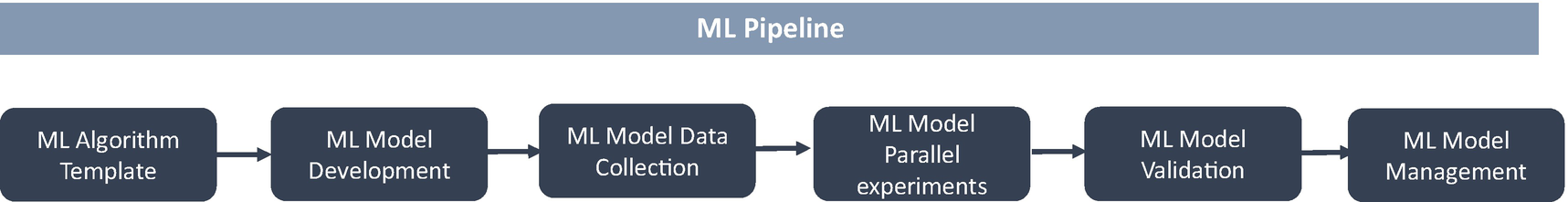
The sprint management discipline involves executing user stories and tasks and tracking their progress. Generally, you will follow a similar approach as a normal agile methodology like grooming user stories for sprints, creating a user story backlog, daily standups with sprint team and, end of sprint demonstration, etc.

**Deploy and Industrialize**

Based on the type of use case, you need to opt for end-to-end testing and full deployment activities at the end of each sprint. During the deploy process, you execute an end-to-end DevOps process. During the process, you develop the AI code, execute tests, and deploy and industrialize the use cases.

**AI and ML in DevOps**

In the new age, modern technologies like cloud native bring considerable change and complexity to how modern systems are created and released. These systems require more than agile; they need to be adaptive and capable of responding dynamically to frequently changing conditions. Automation in regular DevOps is limited to scripting and orchestration. Such scripts sometimes create a bottleneck, and the application and environment can change rapidly. As shown in Figure [15-3](https://learning.oreilly.com/library/view/cloud-native-architecture/9781484272268/html/511610_1_En_15_Chapter.xhtml#Fig3), you need automation that can adapt dynamically, is testable, and can self-heal based on the requirements. The automation solution needs to be able to look at past data, keep learning from recent data, and make flexible, intelligent forecasts about the right course of action.



***Figure 15-3***

ML model pipeline

Checking enormous quantities of information to find an important problem as part of a daily routine is time-consuming. Here, AI can play a significant role in processing, evaluating, and making instant decisions that can take a human hours.

AI and ML integrations can power DevOps by automating routine and repeatable tasks, offering enhanced effectiveness, and minimizing the time spent on procedure code, test, and delivery.

The following types of automation are defined as part of AI in DevOps. There are many ways you can use AI and ML in your software engineering lifecycle.

* *The solution that helps in requirements*: These kinds of standard tools help to advance requirements engineering by applying AI.
* *The solution that helps engineers*: This kind of standard tool helps engineers in programming and reviewing the code.
* *The solution that does quality checks*: With a detailed evaluation of testing outputs, AI performs efficient quality results, and these kinds of standard tools help to solve the authoring, initialization, and generation of automated testing.
* *Environment management*: Improve the range of automation in an environment, including automating many routines and repeatable jobs, using resources, and predicting the load on containers.
* *Early discovery*: AI tools can provide the operations teams with the ability to detect an issue at an early stage and ensure faster response time.

**AI and ML in Code Management**

In your day-to-day development, you might have to use static analysis tools to identify problems in your code. The overall effectiveness of these tools is based on the quality and number of rules configured in them. Many companies are working on AI-enabled tools that provide greater and deeper analysis.

**Source Code Progress**

Git and Bitbucket are source code tools. Applying ML to them addresses the irregularities around code quantities, long construct times, delays in check-ins, improper resourcing, etc.

**DeepCode.AI**

DeepCode is an AI-powered programming tool that works as a coding assistant for software development projects. This tool is trained with a massive volume of data with approximately 250,000 coding rules assessed from both public and private repositories. Based on the trained rules and context, the tool suggests to engineers how to fix the code. Along with the suggestion, it warns the engineers about critical vulnerabilities you need to solve in your code. It learns during the usage of the tool and makes suggestions instantly during the code review. This online tool connects to your repository in GitHub, GitLab, or Bitbucket, via either a private or public repository.

DeepCode is based on custom AI and semantic analysis techniques that were specially designed to learn the rules and information from the cloud.

Static code analysis tools require additional capabilities to find vulnerabilities in code, but AI-enabled tools don’t require an understanding of the deep code analysis to identify vulnerabilities and also learn during the analysis. DeepCode is a combination of static analysis and custom machine learning algorithms.

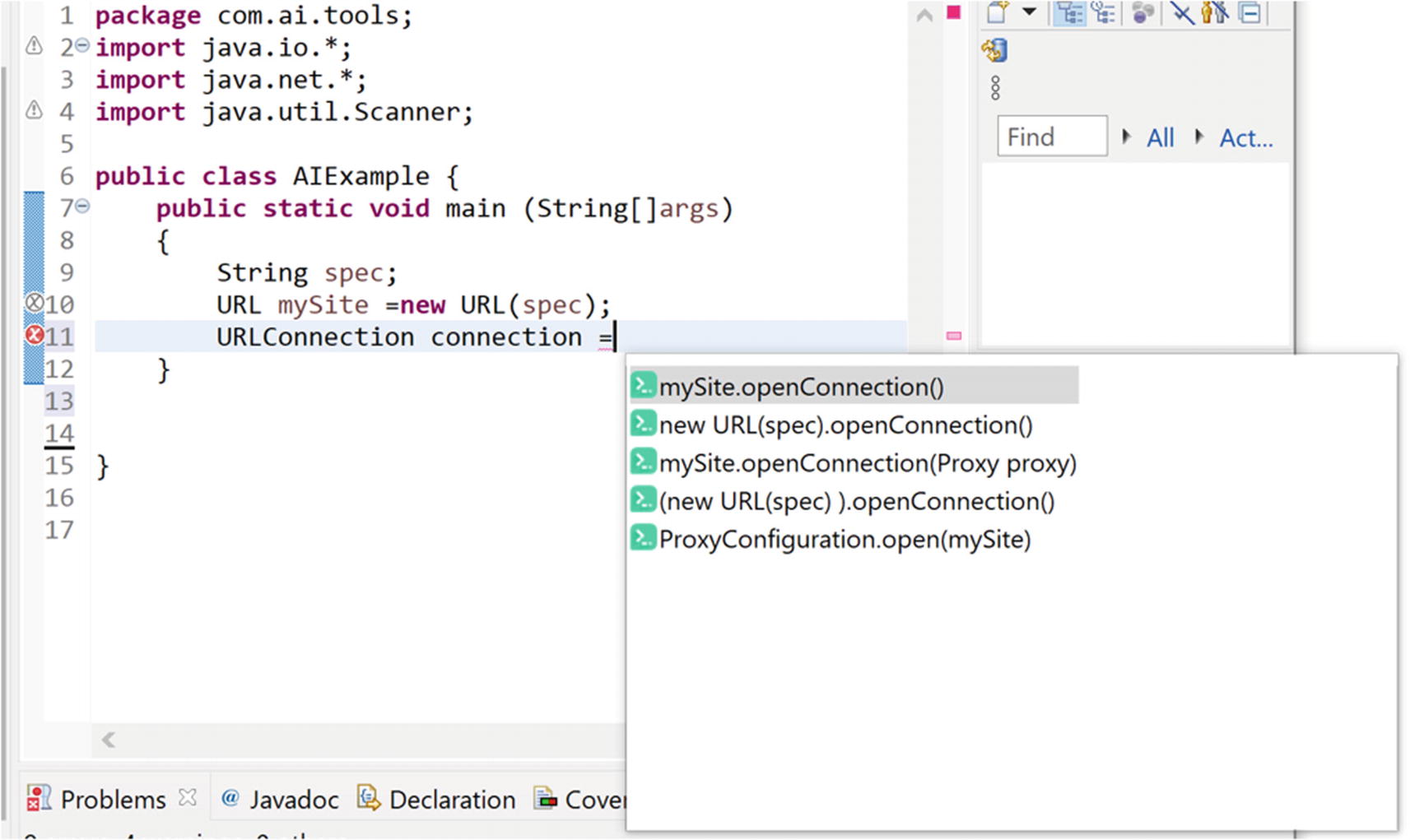
Unlike static code analysis tools, it does not rely on manually hard-coded rules, but learns automatically from data and uses the pre-defined business rules to analyze the program. This concept of never-ending learning enables the system to constantly improve with more data, without supervision.

**Codota**

Codota is AI completion for your Java code in an IDE. It learns as you are writing code to help you code better. It is using AI and ML learnings and gives relevant suggestions to complete the code. It gives suggestions based on a model trained on millions of open source Java programs, which are then modified based on the code you are currently working on.

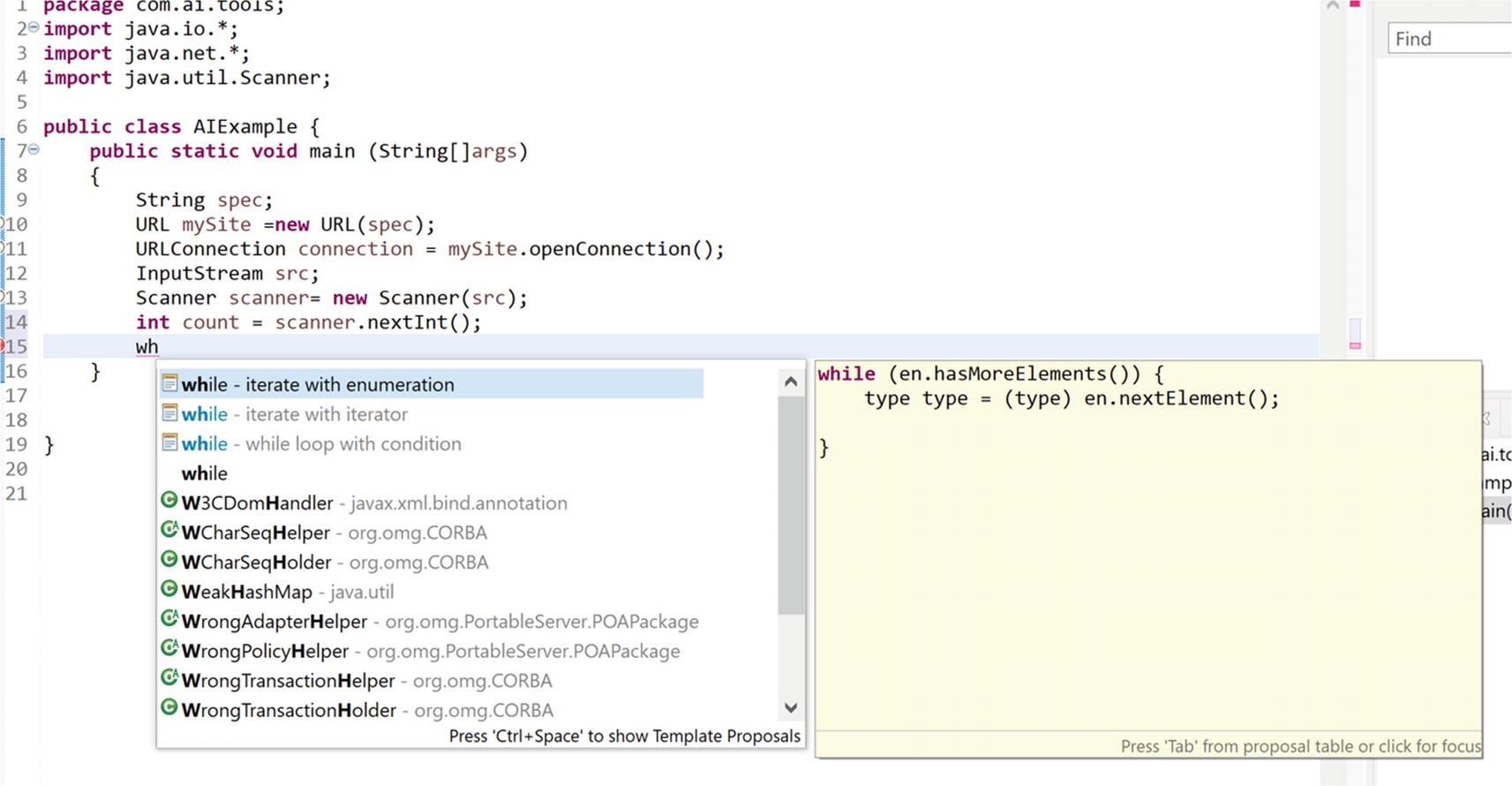
Codota is available for IntelliJ, Android Studios, and Eclipse and you integrate as a plug-in. Codota learned from millions of program lines. With this learning, this tool completes lines of code based on your context, which helps you to code faster with fewer errors. It uses the context of the code you are writing as a required input. If you are in a dilemma to find the best code for your program, this tool is capable of suggesting the best way to complete the code.

Figure [15-4](https://learning.oreilly.com/library/view/cloud-native-architecture/9781484272268/html/511610_1_En_15_Chapter.xhtml#Fig4) and Figure [15-5](https://learning.oreilly.com/library/view/cloud-native-architecture/9781484272268/html/511610_1_En_15_Chapter.xhtml#Fig5) show the code snippets from Eclipse when using Codota.



***Figure 15-4***

Codota suggestion for URL connection



***Figure 15-5***

Codota suggestion for while loop

The following are the benefits of Codota:

* *Code faster*: Codota helps find a reliable code prediction based on an AI-learned code pattern.
* *Prevent error*: ML algorithms detect the code’s intent, not just syntax mistakes.
* *Discover code*: Expand your knowledge and reveal a new and efficient way to leverage open source code.

These are the drawbacks of Codota:

* *Code secure*: Codota sends minimal information on the application’s local context, and the scope is limited to the code that you are currently editing. It doesn’t send full blocks of code outside of your IDE. It sends class names, variable names, and methods that are invoked in secured communication, and it does not use your code to train models.

**Quality Checks**

ML performs efficient quality assessment results and builds test pattern libraries based on the identification of bugs. This helps the team to evaluate results on every launch and thus improves the quality of the application delivered.

Testim.io is a cloud platform that uses AI for fast authoring, execution, and maintenance of automated tests. This tool supports a few test types like end-to-end testing, functional testing, and UI testing. This tool easily identifies any changes like ID names or attributes in a UI by using AI in real time. This tool integrates with the DevOps pipeline, capturing the logs of tests and screenshots of test runs, and provides detailed reporting on test runs.

**Continuous Feedback**

One of the main properties of DevOps is the use of continuous feedback loops at every stage of the process. This includes using the monitoring tools, quality checks tools, etc. ML is already providing detailed monitoring details including performance metrics, log files, and other types. Applying ML in this space will help to identify patterns easily.

**Kubeflow**

Kubeflow is the ML toolkit of Kubernetes. It progressively releases containerized AI microservices over Kubernetes orchestration. It provides a framework-agnostic pipeline for making AI microservices production-ready across multicloud environments. It streamlines the creation of production-ready AI microservices and makes certain the flexibility of containerized AI apps among Kubernetes clusters.

**Alert Monitoring**

AI and ML can manage the monitoring alerts in the systems. The AI-based tools learn and predict the problems of the system and alert proactively.

**Summary**

This chapter provided you with an overview of the end-to-end design, development, and delivery of AI projects. It provided insight into AI requirements and architecture, design, and principles, and we covered best practices. The chapter also covered the details of what methodologies you need to develop AI projects and how to automate the delivery pipeline.

Governance is the most important part of AI projects. Without proper governance, your projects may fail. Therefore, I provided insight into how to govern, manage, and measure your AI projects.

Finally, I provided details about AI-based development, test, and delivery tools to accelerate normal cloud native projects.