UWI Universal Wallet Infrastructure

Agentic Trust Layer: Building the Foundation of Trust for the Age of AI Agents

Software Supply Chain Governance through Verifiable Credentials and Agentic AI

CO-AUTHORED BY

docomo Global accenture aws

Executive Summary

The Core Business Challenge

In an era where AI agents execute contracts, write code, and make decisions, there is currently no infrastructure that can record "who authorized that AI, and under what authority it acted" as tamper-proof records that regulators, auditors, and business partners can independently verify. This is not a security problem. It is a business-level challenge: the absence of a trust foundation—an Agentic Trust Layer—required to place AI at the core of business operations.

Most existing AI security and governance solutions focus on model robustness and data input/output control. However, the third layer—a mechanism to prove "what AI actually did" at the workflow level—remains an industry-wide gap. This white paper aims to provide a concrete technical answer to this gap.

What This Initiative Addresses

This initiative proposes an architecture that simultaneously achieves the following three objectives by combining the Verifiable Credential technology of UWI (Universal Wallet Infrastructure), co-developed by NTT DOCOMO GLOBAL and Accenture, with AWS's AI product portfolio.

Provenance Attestation for AI-Generated Code: Records which AI generated the code, under whose instructions, and using which model, in a form that anyone can independently verify after the fact. This resolves the concern that "the origin of code cannot be tracked"—the greatest barrier to enterprise adoption of AI-driven development.

Automation of Continuous Security Assurance: Vulnerability analysis and trust verification are automatically executed with every code change. Rather than a one-time audit, this achieves continuous governance throughout the entire development lifecycle.

Extension to All Future Agent Activities: While AI-driven development serves as the initial implementation domain, the same foundation can be directly applied to Agentic Commerce, recruitment AI, healthcare, financial services, and manufacturing. Building this foundation now creates a competitive advantage in trust infrastructure for the AI era.

Target Market and Regulatory Landscape

Indicator	Figure	Implication
Domestic software projects	Approx. 480,000	By 2027, approximately 70,000 projects may fall within the scope of strengthened software supply chain management, including SBOM-related practices
Domestic SBOM adoption rate	7% ⟶ 15% (2025⟶2027)	Rapidly expanding against the backdrop of the EU Cyber Resilience Act and METI initiatives
Global verification transactions	Approx. 192 million (2027 est.)	As AI-driven development proliferates and build frequency accelerates, market size is projected to double from 2025

With the main obligations under the EU Cyber Resilience Act (Regulation (EU) 2024/2847) scheduled to apply from December 2027, the importance of cybersecurity compliance for products with digital elements, including software, is increasing. SBOMs are merely the first point of contact. In an era where business partners and regulators assume the ability to "prove it," delays in compliance translate directly into competitive risk.

Intended Audience and How to Use This White Paper

The primary intended audience of this white paper includes software engineers, security engineers, architects, CISOs, and compliance officers who are considering or actively pursuing the adoption of AI-driven development.

For executives and CIOs, reading Chapters 1 and 2 will provide an understanding of the Agentic Trust Layer concept and the overall landscape of trust challenges in AI-driven development. For technical details, please refer to Chapter 6 (Technical Architecture Details).

Table of Contents

1. Agentic Trust Layer — The Foundation of Trust for the AI Era

Why "Trust Infrastructure" Has Become a Business-Level Challenge
In a world where AI agents act autonomously, the ability to answer the questions "Did that agent truly have authorization?", "Who was the human who gave the instruction?", and "What decisions were made and how?" becomes a prerequisite for business operations. Regulators demand evidence of compliance, business partners demand supply chain transparency, and shareholders question the reality of AI governance. Organizations that cannot answer these questions risk being deemed as failing to meet the conditions required to place AI at the core of their business.

What Is the Agentic Trust Layer?
The Agentic Trust Layer is a foundational infrastructure that embeds attestation of "who acted, under what authority, and what was done" into every activity of AI agents. Specifically, it is realized by recording agent actions as tamper-proof records and embedding into development and operational workflows a mechanism that enables regulators, auditors, and business partners to independently verify those records.
Most existing AI security and governance solutions focus on model robustness (Layer 1) and data input/output control (Layer 2). The Agentic Trust Layer targets Layer 3: workflow-level trust and accountability for "what AI actually did." This third layer currently remains an industry-wide gap.

Initial Implementation Domain and Extension Path
This initiative selects AI-driven development as its initial implementation domain. This is because it is the domain where AI agents are most actively operating and where governance requirements are beginning to be explicitly demanded. However, the design principles of the Agentic Trust Layer are universal, and the same foundation can be applied to any domain where agents operate. Building this foundation now creates a competitive advantage in trust infrastructure for the AI era.

2. Trust Challenges in AI-Driven Development

The previous chapter presented the concept of the Agentic Trust Layer as a trust foundation and the rationale for selecting AI-driven development as its initial implementation domain. This

chapter provides a concrete analysis of how trust challenges manifest within the domain of AI-driven development.

Software development is a domain where these trust challenges are particularly pronounced. In AI-driven development, AI is positioned as a core collaborator in the development process, executing planning, implementation, and testing. This gives rise to new questions regarding the provenance of AI-generated code, the accountability of agents, and the integrity of software artifacts.

Traditional SBOMs (Software Bills of Materials) provide visibility into software composition, but they reveal new limitations in the context of AI-driven development. While SBOMs can record "which libraries are included," they cannot record "who—or which AI—generated the code, and under what instructions." Furthermore, SBOMs lack a mechanism to cryptographically prove the authenticity of the SBOM files themselves. Although SBOMs are widely adopted, in practice they remain documents created after the fact for auditing and regulatory compliance, rather than mechanisms for verifying trust in real time within development and operational processes. The proof of the issuer's authenticity has also not been operationally established. AI security and governance can be understood across three layers. The first layer is model-level security, the second layer is governance over data inputs and outputs, and the third layer is workflow-level trust and accountability. Most existing solutions focus on the first and second layers, but this initiative focuses on the third layer—embedding verifiable trust and accountability into operational workflows.

To address these challenges, this initiative adopts Decentralized Identifiers (DIDs) and Verifiable Credentials (VCs) as the foundation of trust. This approach enables the following: the content and issuer of an SBOM can be cryptographically verified; compliance with standards allows verification across different organizations without dependence on any specific vendor or system; the expiration and revocation of credentials can be managed; and the same mechanism can be extended to prove the actions of AI agents. Rather than replacing existing SBOMs and SCA tools, it functions as a trust layer that enhances them.

3. Objectives of This Initiative

Now that autonomous AI agents routinely generate and modify code, the inability to prove "who—or which AI—generated the code, and under what circumstances" has become a fundamental barrier to enterprise AI adoption. This initiative aims to provide a concrete technical answer to this challenge.

This initiative seeks to demonstrate the potential for creating a new value domain called "AI Trust" by combining UWI's Verifiable Credential (VC) and trust technologies with AWS's AI product portfolio. As AI technology rapidly proliferates, ensuring the trustworthiness of AI-generated data and decision-making processes has emerged as a critical challenge.

In particular, as AI-assisted and AI-led code generation (Agentic Coding) becomes mainstream in enterprise software development, the provenance, integrity, and security of AI-generated code are increasingly recognized as significant concerns. Without a verifiable mechanism to track which AI model generated the code, under whose instructions, and based on which dependencies, organizations face serious risks in software supply chain security and compliance.

Against this backdrop, this initiative validates a new approach that leverages the Decentralized Identity (DID) and Verifiable Credential technologies provided by UWI to achieve AI agent identity verification, data authenticity assurance, and end-to-end process transparency.

3.1 Core Technical Approach: VC-SBOM

The core technical approach of this initiative is to issue Software Bills of Materials (SBOMs) as Verifiable Credentials (VC-SBOMs). By representing SBOMs—which record software components, dependencies, and their provenance—as cryptographically signed Verifiable Credentials managed by UWI, we establish tamper-evident and auditable records for both human-written and AI-generated code.

Furthermore, this initiative introduces the concept of AI-SBOM to record information specific to AI-generated code. An AI-SBOM is an extended SBOM that, in addition to the component and dependency information recorded in a standard SBOM, additionally records AI-specific metadata such as "who instructed the AI," "which AI agent generated the code," and "which model version was used."

VC-SBOM and AI-SBOM are concepts that extend SBOMs along different axes. VC-SBOM corresponds to the trust mechanism axis (signing and issuing an SBOM as a VC, adding tamper detection and issuer authenticity attestation), while AI-SBOM corresponds to the recorded content axis (additionally recording AI-specific metadata). These two axes are independent, and this initiative combines both to record the provenance of AI-generated code in a cryptographically verifiable manner.

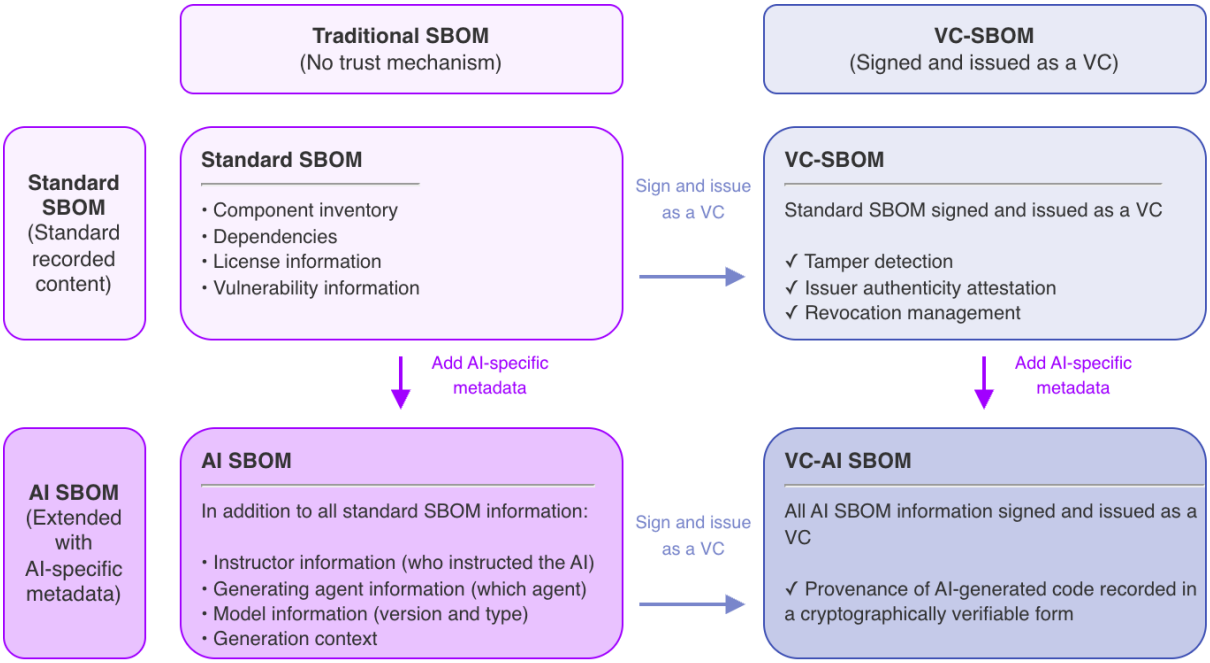

Figure 1: Relationship between VC-SBOM and AI-SBOM

This approach naturally extends to AI-SBOM. An AI-SBOM is a standard SBOM extended specifically for tracking AI-generated code, additionally recording the identity of the AI agent that generated the code, the version and type of the model used, and the chain of instructions from human to AI (who instructed, which model, and under what instructions the code was generated). This ensures that each step from code origin to approval and release is cryptographically attested, establishing accountability and auditability for AI-generated code.

By incorporating AI-SBOM throughout AI-driven development, security verification and trust attestation are automatically coordinated at critical points in the development workflow (code generation, differential updates, and release approval), achieving a highly trustworthy AI development and operations model.

3.2 Assigning Verifiable Identity to AI Agents

Once provenance management of code through SBOMs is established, the next natural evolution is the challenge of "how to prove the identity of the AI agent that generated the code." As autonomous AI agents actively participate in enterprise development workflows—generating, reviewing, and modifying code—a mechanism that can cryptographically prove "which agent took action" has become a prerequisite for governance in AI-driven development.

As an approach to this challenge, assigning DID-based Verifiable Identity to AI agents through UWI is effective. This enables the delegation chain in multi-agent environments (Planner Agent → Coder Agent → Reviewer Agent → Human Approver) to be preserved as verifiable records, allowing each agent's actions to be audited in accordance with the organization's governance policies. Additionally, because agents are assigned Verifiable Identity, it becomes possible to identify agents exhibiting anomalous behavior and update the status of VCs in whose issuance those agents were involved.

3.3 Value Propositions to Be Demonstrated

This initiative concretely demonstrates the following value propositions by combining Verifiable Credentials with AWS's AI services. Specifically, it leverages Kiro, an integrated development environment provided by AWS that supports spec-driven development, and Amazon Bedrock AgentCore, an agent execution platform.

Enhanced SBOM Trustworthiness: By issuing SBOMs as Verifiable Credentials (VC-SBOMs), the integrity of SBOM content and the authenticity of the issuer become cryptographically verifiable. This transforms SBOMs from compliance documents created after the fact into operational mechanisms that can verify trust in real time within development and operational processes. This initiative implements the end-to-end process of automated VC-SBOM generation by the Analyzer Agent and signing/verification by UWI, demonstrating that this transformation is technically feasible.

Provenance Attestation for AI-Generated Code: The ability to record and verify—as tamper-proof VC-SBOMs—who instructed, which AI model, and based on which specification the code was generated. This initiative defines the extended fields of AI-SBOM (instructor information, AI agent information, model information, and generation context) and confirms through implementation that these are correctly recorded and verified as VC-SBOMs.

Continuous Security Assurance: The integration of vulnerability detection and SBOM management into AI-driven development workflows, achieving continuous trust assurance rather than one-time verification. This initiative implements an end-to-end flow in which SBOM generation through vulnerability analysis is automatically executed each time code is generated or modified, confirming that continuous assurance is operationally viable.

The trust chain established in this initiative is intended to evolve in the future toward assigning Verifiable Identity to AI agents themselves (identifying agents via DIDs and recording delegation relationships between agents), and this initiative is positioned as the first step toward that goal.

4. Use Case Overview

Based on the technical approach described in Section 3.1, this initiative envisions the following four use cases. UC1 serves as the core, with UC2 through UC4 each complementing the trust mechanism from different perspectives.

For clarity, this white paper adopts the following terminology conventions. VC-SBOM is the general term for a standard SBOM or AI-SBOM that has been signed and issued as a VC. AI-SBOM is a standard SBOM extended with AI-specific metadata. The relationship between the two corresponds to two orthogonal axes, as illustrated in the conceptual diagram in Section 3.1.

4.1 UC1: Automated Generation and Signing of VC-SBOMs and AI-SBOMs

This use case provides a mechanism for automatically issuing and attesting SBOMs in VC format for software code changes. UC1 consists of the following two sub-use cases.

- UC1a: VC-SBOM — Converting Standard SBOMs to VCs
  Triggered by specific events such as code generation, code changes, dependency updates, pull request creation, and release approval, the software bill of materials (SBOM) for that code is automatically generated or updated and signed as a VC issued by UWI. This transforms the SBOM from a mere list file into a bill of materials accompanied by a tamper-evident certificate. Both human-written code and AI-generated code are covered, and a unified trust mechanism is applied to all code changes.
- UC1b: AI-SBOM — Recording AI-Specific Metadata
  In AI-driven development, AI-specific metadata—such as the model used, associated tools, and prompt information—is individually recorded at each AI code generation event. This information is consolidated into an AI-SBOM when the code is pushed to the repository, and signed and issued as a VC in the same manner as UC1a. As described in Section 3.1, the AI-SBOM additionally records the person who instructed the AI, the agent that generated the code, and the model used, thereby establishing traceability across the entire AI-generated code supply chain.

This use case serves as the foundational core for all subsequent use cases (UC2 through UC4).

4.2 UC2: Code Safety Analysis

Using the VC-SBOMs issued in UC1 as the data source for analysis, the components contained therein are automatically cross-referenced against vulnerability databases (such as NVD and OSV) to visualize and analyze the security risks of AI-generated code.

For AI-generated code, the AI-specific information recorded in the AI-SBOM can be leveraged as additional analysis targets. Specifically, the model information used, the software components constituting the agent, the tools and MCP servers accessed by the agent, and prompt information are analyzed to detect the presence of vulnerabilities or dependencies on untrusted external components. Furthermore, these analysis results can be used as proactive feedback to agent behavior—for example, improving agent behavior by excluding vulnerable libraries from the options during code generation.

4.3 UC3: Continuous Security Verification and Checking of Code

This use case continuously and automatically monitors the security of code throughout the development lifecycle. By managing SBOMs as VCs, continuous security verification is achieved while cryptographically attesting the state of the code at specific points in time. Integration with CI/CD pipelines enables the identification of the impact scope when new CVEs are published and the automatic blocking of deployments that violate security policies.

From the perspective of integration with AI-SBOM, during CI/CD pipeline checks, in addition to standard vulnerability verification, AI governance-oriented verifications can also be automatically performed—such as whether the code was generated by an approved AI agent, whether an approved model version was used, and whether a human review was conducted. This makes it possible to continuously assure code quality and security while also addressing governance requirements specific to AI-generated code. While this initiative uses monitoring triggered by important events in the development lifecycle, monitoring the agent's runtime operation (runtime monitoring) is identified as one of the challenges that will require future expansion.

4.4 UC4: Trust Verification Using Verifiable Identity for AI Agents

AI-SBOMs record which AI agent generated the code. However, for this record to be meaningful, it must be possible to prove that "the AI agent's ID is authentic."

As described in Section 3.2, by issuing a DID (Decentralized Identifier) to an AI agent through UWI and assigning a VC linked to that DID, a Verifiable Identity is established. A DID is a technical mechanism for uniquely identifying an agent, and Verifiable Identity is the concept of

"an identity that can be verified by third parties," realized through the combination of a DID and a VC. The issuance of DIDs to AI agents is envisioned as a future extension of the UWI platform and is included within the scope of this initiative.

By recording the AI agent's DID—issued through UWI—in the "AI Agent Information" field of the AI-SBOM, the agent information within the AI-SBOM becomes not "self-declared" but verifiable information attested by UWI. This enables the identification and immediate response to AI agents exhibiting unauthorized or anomalous behavior, the demonstration of "accountability for AI-generated code" to regulatory authorities and external auditors, and the visualization and management of trust chains between agents.

The overall architecture that realizes these use cases is as follows.

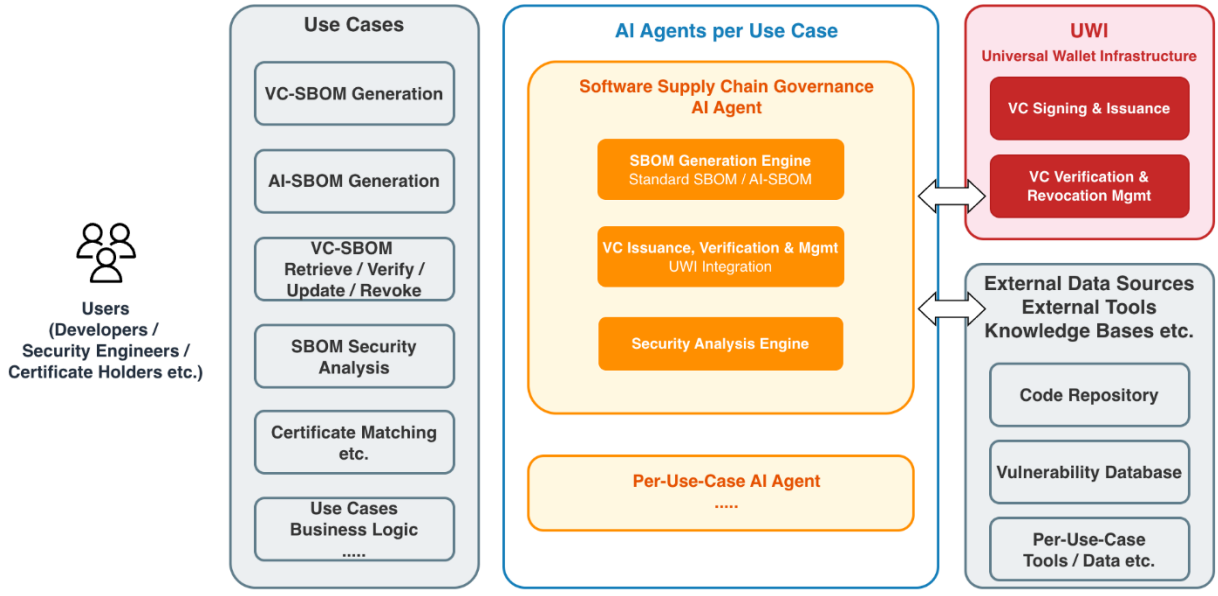

Figure 2: Component Groups That Realize the Use Cases

5. Impact and Business Perspective of This Initiative

5.1 Target Market Size for VC-SBOM

To gauge the scale of the market targeted by this initiative, it is important to note that SBOMs are generated, updated, and verified on a per-build basis, meaning that an increase in build frequency directly translates to an expansion in transaction volume. As AI-driven development accelerates build frequency, this market is considered to have substantial growth potential.

The total number of software projects is estimated by dividing the global IT professional population (approximately 30 million; Human Resocia, 2024) by an assumed team size of three, yielding approximately 10 million projects globally and approximately 480,000 in Japan based on the proportion of IT professionals. SBOM adoption rates in Japan stood at approximately 7% as of 2025 (Veriserve Survey, 2025). Driven by METI's SBOM promotion initiatives and against the backdrop of the main obligations under the EU Cyber Resilience Act (CRA, Regulation (EU) 2024/2847) scheduled to apply from December 2027, adoption is expected to expand to 15% domestically and over 30% globally by 2027 (PwC/Black Duck). Build frequency is assumed at 64 times per year, combining maintenance and operations (once per month) with AI-driven rapid development (once per week).

Based on these assumptions, the domestic market is projected to grow from approximately 2.15 million transactions in 2025 to approximately 4.61 million transactions by 2027, while the global market is projected to grow from approximately 96 million transactions in 2025 to approximately 192 million transactions by 2027. Note that the above figures represent the maximum scale of total SBOM generation and verification transactions; the proportion issued and verified as VC-SBOMs depends on the adoption rate of the VC-SBOM infrastructure.

5.2 Impact by Stakeholder

This initiative creates distinct value for two stakeholder groups involved in the software supply chain: the consumer side, consisting of "organizations and personnel who develop and operate systems," and the provider side, consisting of "organizations that implement and deliver solutions." The impact on each can be organized along two axes: the domain of existing standard SBOMs and the domain of AI-SBOMs that emerge alongside AI-driven development.

- Impact on Organizations and Personnel Who Develop and Operate Systems
  In the standard SBOM domain, establishing SBOMs as tamper-evident and auditable records for both human-written and AI-generated code improves software maintenance and operations, incident response, and regulatory compliance. This is expected to enhance the competitiveness of adopting organizations, and as confidence in reliable SBOM operations grows, the adoption of SBOMs themselves is expected to expand further. In the AI-SBOM domain, the ability to attest the provenance of AI-generated code alleviates security and traceability concerns regarding the use of AI in code generation. Currently, one of the greatest barriers to enterprise adoption of AI-driven development is the concern that "the origin of code written by AI cannot be tracked." AI-SBOM provides a direct solution to this concern, thereby contributing to the broader

adoption of AI-driven development itself.
As these values become widely recognized, demand for VC-SBOM issuance and verification transactions is expected to grow, supporting the realization of the market size projected in Section 5.1.

- Impact on Organizations That Implement and Deliver Solutions
  In the standard SBOM domain, the provision of SBOM generation and verification processes on the UWI infrastructure expands the value proposition of SBOM solution providers. Where previously the ability to "record" SBOM information was considered sufficient value, the addition of cryptographic "verifiability" as a separate axis of added value is now anticipated. This represents an opportunity for feature expansion among existing players and a point of differentiation for new entrants.
  In the AI-SBOM domain, in addition to existing SBOM capability providers, organizations that provide coding agents are expected to consider offering AI-SBOM capabilities. This is because coding agents, as direct participants in the generation process, are in the most natural position to record provenance information. The extent to which third-party verifiability will be demanded by the market depends on future developments, but this initiative aims to establish cryptographically verifiable AI-SBOMs as a standard requirement.
  For UWI, as these solution providers integrate VC-SBOM issuance and verification capabilities into their own services, a structure emerges in which transaction demand for the UWI infrastructure expands across the entire ecosystem.

5.3 The Role of AWS in This Initiative

The solution implementation leveraging AWS's AI infrastructure technically underpins and further amplifies the stakeholder impact described in Section 5.2.

- Value as an AI Agent Implementation Platform
  The solution of this initiative is built on an integrated platform for AI agents provided by AWS. This platform provides an environment for consistently developing, deploying, and managing agents, allowing solution implementers to focus on developing the core logic of the trust mechanism. Because infrastructure construction and operations are provided as managed services, operational overhead is minimized while ensuring scalability in response to demand. The platform is designed to facilitate the integration of agents into AI-driven development workflows, supporting a seamless experience in which developers do not need to consciously manage SBOMs. Because the platform

itself meets fundamental security and reliability requirements, adopters can proceed with deployment without additional concerns about solution safety.

- Value as an Infrastructure Suited for VC-SBOM Workloads
  As estimated in Section 5.1, VC-SBOM generation and verification transactions could reach tens of millions to hundreds of millions per year. AWS's cloud infrastructure provides scalable, low-latency processing capabilities for such high-volume transactions, while storing signed VC-SBOMs in storage with high durability and tamper resistance. This allows adopters to use the solution with confidence in both the availability and integrity of VC-SBOMs. Furthermore, the ability to centrally manage tracing and monitoring of agent operation logs and VC issuance histories serves as a foundation for fulfilling accountability obligations to regulatory authorities and external auditors. Technical details are described in Section 6.

6. Technical Architecture Details

6.1 Architecture Overview

This section describes the technical architecture covering UC1 (Automated Generation and Signing of VC-SBOMs and AI-SBOMs), UC2 (Code Safety Analysis), and UC3 (Continuous Security Verification) among the use cases defined in Section 4. Features related to UC4 (Trust Verification Using Verifiable Identity) are addressed in Section 6.5 (Features Planned for Future Phases).

The architecture of this initiative organizes components into two layers based on their functional roles.

Trust-Related Components (what is achieved): These are the component groups designed and implemented by this initiative. They consist of the following three functions. The Analyzer Agent executes security analysis triggered by code changes and outputs the analysis results in a structured format. The SBOM Generation Engine generates VC-SBOMs (standard SBOMs converted to VCs) and AI-SBOMs (recording AI-specific metadata) based on the output of the Analyzer Agent. The UWI Integration Interface provides the functionality to send generated SBOMs to UWI for signing, issuance, and verification as VCs. These components are responsible for the logic specific to this initiative: VC-based SBOM conversion and traceability recording of AI-generated code.

Execution Environment and Infrastructure Services (where and how it operates): These are the environment and infrastructure services on which the trust-related components operate. They consist of Kiro as the development environment, a scalable agent execution platform, signing key management, access control, and other cloud infrastructure services. This initiative combines these to achieve the secure and efficient operation of the trust-related components. Details of each component are described in Sections 6.2 and 6.3.

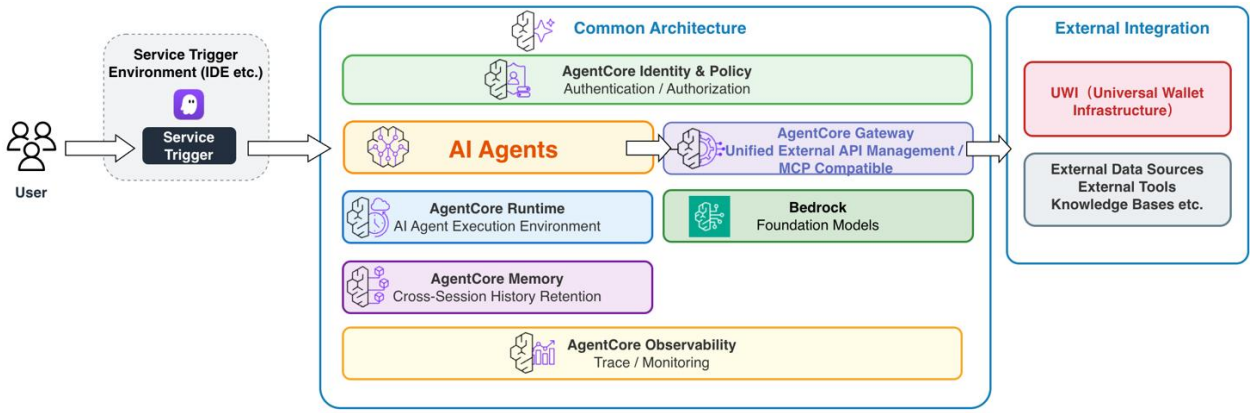

Figure 3: Environment and Infrastructure Services on Which Trust Components Operate

6.2 Trust-Related Components

6.2.1 SBOM Generation Engine

The SBOM Generation Engine is a component that, triggered by code changes, collects and structures component information from the target codebase and generates SBOM artifacts in standard formats (CycloneDX or SPDX). The generated SBOMs are signed and issued as VCs through the UWI integration described in Section 6.2.2 and stored in the VC-SBOM repository. This component is responsible for the implementation of UC1a (VC-SBOM) and UC1b (AI-SBOM).

The SBOM Generation Engine manages two tracks in parallel: standard SBOMs and AI-SBOMs.

- Standard SBOM
  Standard SBOMs are generated for the codebase at the point in time when code is pushed to the repository. The recorded content includes the component inventory (library names, versions, dependencies, and license information). An SBOM corresponding one-to-one with the codebase at the time of each push is generated and stored as a version-controlled history in the VC-SBOM repository.

- AI-SBOM
  AI-SBOMs are managed as an extended track that is typically managed in association with standard SBOMs, recording metadata specific to AI-driven development. At each AI code generation event, the following AI-specific information is individually recorded and consolidated into an AI-SBOM when the code is pushed to the repository.
  The metadata recorded in AI-SBOMs is as follows:

  - Instructor Information: The identifier of the user who instructed the AI, and the date and time of the instruction
  - AI Agent Information: The identifier and version of the AI agent that generated the code (DID is used when Verifiable Identity is established through UC4)
  - AI Model Information: Model name, version, and provider
  - Generation Context: The prompt used, identifiers of referenced files and resources, and information about tools and MCP servers accessed
  - Human-in-the-Loop Information: Whether human review and approval were conducted, and the identifier of the approver (DID is used after UC4 is introduced)

Note that the recording format for prompt information (full-text recording or hash-based recording) is selected based on the intended purpose. If the objective is to confirm identity with past prompts, a hash value is sufficient; however, if retrospective review of prompt content is required, full-text recording is necessary. This choice is determined based on the organization's security policies and data retention requirements. However, in actual operation within large development organizations, it becomes necessary to apply different recording policies to each project or team, so a system is required that allows policies to be set and managed on a project-by-project basis.

6.2.2 VC Issuance, Verification, and Management (UWI Integration)

This component is the interface that sends SBOMs generated by the SBOM Generation Engine or the reference information to UWI (Universal Wallet Infrastructure) for signing and issuance as Verifiable Credentials (VCs) compliant with the W3C VC Data Model standard. This component is also responsible for storing issued VC-SBOMs and AI-SBOMs in the VC-SBOM repository, as well as retrieval and revocation management. UWI handles the secure management of signing keys and the signing, issuance, and verification of VCs.

- VC Issuance Process
  The VC issuance process consists of the following steps:

- Step 1 (Initial Setup): The development organization/project and human developers/approvers register their identifiers with UWI. UWI issues a Verifiable Identity Credential to each entity. Note that DID issuance to AI agents is addressed as UC4 (future extension) in Section 6.5.
- Step 2 (SBOM Submission): The SBOM artifact (CycloneDX/SPDX JSON) itself or its storage location generated by the SBOM Generation Engine is sent to UWI. The request includes a reference to the SBOM artifact or its storage location, the hash of the target artifact, AI-specific metadata in the case of AI-SBOMs, and the requester's identification information.
- Step 3 (Signing and VC Construction): After authenticating the requester, UWI signs the SBOM and constructs a VC compliant with the W3C VC Data Model.
- Step 4 (Storage and Distribution): The issued VC is returned to the SBOM Generation Engine and stored in a version-controlled manner in the VC-SBOM repository. The VC identifier is annotated on the corresponding Git commit or pull request.

- VC Verification Process
  When an authorized verifier invokes UWI's verification function, signature validity confirmation, issuer DID resolution and confirmation, and revocation status confirmation are performed. This enables verification that the target SBOM has not been tampered with, was issued by a legitimate issuer, and is currently valid. Multiple interfaces are envisioned for verifier access, including direct API calls, automated calls from CI/CD pipelines, and manual confirmation from a management console.
- VC Revocation Management
  When it is necessary to revoke the VC of a VC-SBOM or AI-SBOM in which an anomaly has been detected, the relevant VC is updated to a revoked status through UWI's revocation management function, and subsequent verification requests return a revoked status.

6.2.3 Security Analysis Engine

The Security Analysis Engine is a component that detects and visualizes security risks in code by analyzing VC-SBOMs and AI-SBOMs stored in the VC-SBOM repository. This component is responsible for the implementation of UC2 (Code Safety Analysis) and UC3 (Continuous Security Verification).

From the UC2 perspective, the component information contained in VC-SBOMs is automatically cross-referenced against vulnerability databases (such as NVD and OSV) to detect components

with known vulnerabilities. For AI-generated code, the AI-specific information recorded in AI-SBOMs can be leveraged as additional analysis targets. Specifically, the model information used, the software components constituting the agent, and the information about tools and MCP servers accessed by the agent are analyzed to detect the presence of vulnerabilities or dependencies on untrusted external components.

From the UC3 perspective, integration with CI/CD pipelines enables the identification of the impact scope when new CVEs are published and the automatic blocking of deployments that violate security policies. Through integration with AI-SBOMs, AI governance-oriented verifications can also be automatically performed—such as whether the code was generated by an approved AI agent, whether an approved model version was used, and whether a human review was conducted.

Analysis results are utilized for vulnerability notifications and dashboard visualization, as well as proactive feedback to agent behavior. For example, it is possible to improve agent behavior by excluding vulnerable libraries from the options during code generation.

6.2.4 Analyzer Agent (Orchestration)

The Analyzer Agent is an orchestration agent that integrates the components described in Sections 6.2.1 through 6.2.3 and executes a series of processes in response to invocations from the development environment. It operates on Amazon Bedrock AgentCore Runtime.

This initiative uses Kiro as the reference implementation development environment. When a specified trigger condition is met in Kiro, the Analyzer Agent is invoked. As described in Section 6.2.1, the Analyzer Agent invokes the SBOM Generation Engine when code is pushed to the repository, initiating the generation of standard SBOMs and AI-SBOMs. However, the invocation interface of the Analyzer Agent is designed to be independent of any specific development environment, and it can be invoked in the same manner from other coding agents or development environments that meet the technical requirements.

The processing flow executed by the Analyzer Agent is as follows. Upon receiving an invocation from the development environment, it first calls the SBOM Generation Engine (6.2.1) to generate standard SBOMs and AI-SBOMs for the target codebase. Next, through the VC Issuance, Verification, and Management component (6.2.2), the generated SBOMs are sent to UWI for signing and issuance as VCs. The issued VCs are stored in the VC-SBOM repository. Furthermore, the Security Analysis Engine (6.2.3) is invoked to execute security analysis on the

generated VC-SBOMs and AI-SBOMs. Analysis results are forwarded to notifications and dashboards.

In the future, it is also envisioned that the trust-related component group, including the Analyzer Agent, will be published as MCP tools callable from other agents and external systems. This would make the SBOM generation, VC issuance, and security analysis functions available from a broader range of contexts.

6.3 Execution Environment and Infrastructure Services

6.3.1 Amazon Bedrock AgentCore (Agent Execution Platform)

Amazon Bedrock AgentCore is the AI agent execution platform on which the trust-related component group described in Section 6.2 (Analyzer Agent, SBOM Generation Engine, and Security Analysis Engine) operates. It provides an execution environment that is independent of any specific agent framework, and this initiative adopts Strands Agents (https://strandsagents.com/) as the agent framework.

The architecture of this initiative leverages the following AgentCore capabilities.

AgentCore Runtime is a serverless execution environment for AI agents, including the trust-related component group. Session isolation enables the safe parallel processing of SBOM generation and analysis for multiple developers and projects. Support for long-running execution accommodates the analysis of large codebases, while a low-latency execution environment is also provided for lightweight checks. AgentCore Memory retains per-project SBOM histories, vulnerability detection patterns, and past analysis results across sessions. This enables efficient analysis by referencing past analysis results.

AgentCore Gateway is a gateway that centrally manages access to external services such as the UWI API and vulnerability database (NVD/OSV) APIs. It provides integrated management of authentication, authorization, and rate limiting for tool invocations, enabling secure access to external APIs. It also has the capability to convert existing APIs into MCP-compatible tools, enabling future functional extensions through standardized interfaces.

AgentCore Observability traces and monitors the operation logs and analysis results of the trust-related component group. It visualizes agent behavior and is utilized for anomaly detection, performance optimization, and audit compliance.

Strands Agents is an open-source agent framework that enables the construction of agents through a model-driven approach combining three elements: a language model, a system prompt, and a tool set. When combined with AgentCore Runtime, it enables seamless deployment from prototype to production environment. This initiative adopts it as the implementation framework for the Analyzer Agent (6.2.4).

6.3.2 VC-SBOM/AI-SBOM Repository

Amazon S3 (with versioning enabled) is adopted as the VC-SBOM repository for storing and managing the VCs of VC-SBOMs and AI-SBOMs. Issued VCs are stored in this repository in a version-controlled manner. S3 is adopted as the storage implementation due to its history retention through version management and ease of integration with CI/CD pipelines.

6.4 Service Invocation Flow

This section describes the order in which the series of processes are executed after code is generated or modified in AI-driven development. This initiative assumes the use of Kiro as the reference implementation development environment.

1. Code is generated or modified by a coding agent or developer.
2. A specified trigger condition is met in the development environment, and the Analyzer Agent (6.2.4) is invoked.
3. The Analyzer Agent invokes the SBOM Generation Engine (6.2.1) to generate standard SBOMs and AI-SBOMs for the codebase at the time of the push, as described in Section 6.2.1.
4. The SBOM Generation Engine generates SBOM artifacts in CycloneDX/SPDX format.
5. Through the VC Issuance, Verification, and Management component (6.2.2), the generated SBOMs are sent to UWI.
6. UWI signs the SBOMs, and the signed VCs are returned to the SBOM Generation Engine.
7. The VCs are stored in the VC-SBOM repository (6.3.2).
8. Security analysis processing is triggered.
9. The Security Analysis Engine (6.2.3) executes vulnerability analysis on the generated VC-SBOMs and AI-SBOMs.

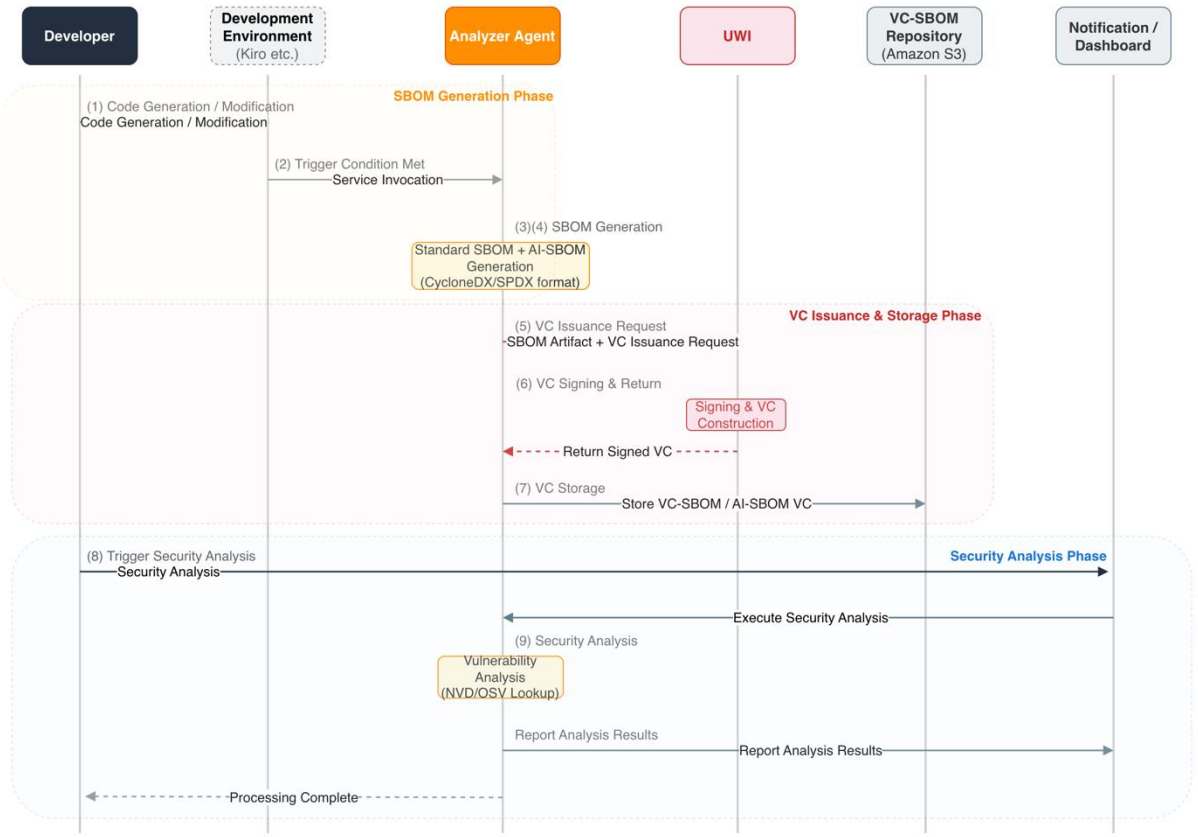

Figure 4: Service Invocation Sequence

Through this flow, each time code is generated or modified, a VC-SBOM is automatically issued for standard SBOMs and a VC of the AI-SBOM is automatically issued for AI-generated code, establishing traceability across the entire software supply chain.

6.5 Features Planned for Future Phases

The architecture of this initiative envisions functional extensions in the following directions beyond the current implementation scope.

- Implementation of UC4 (Verifiable Identity for AI Agents)
  As described in Section 4.4, the issuance of DIDs to AI agents and the establishment of Verifiable Identity are positioned as future extensions of this initiative. When UC4 is implemented, the identifier fields in AI-SBOMs (6.2.1) will transition to DIDs, enabling third-party verification of agent information.
- Integration with CI/CD Pipelines
  The current architecture targets SBOM generation and VC issuance triggered by

invocations from the development environment, but future extensions also envision automation triggered by CI/CD pipeline events (pull requests, merges, releases, etc.).

- Conversion of Trust-Related Component Group to MCP Tools
  As described in Section 6.2.4, it is envisioned that the series of functions provided by the trust-related component group of this initiative—SBOM generation, VC issuance and verification, and security analysis—will be standardized as MCP tools and made callable from other agents and external systems. This would make the trust mechanisms built by this initiative available from a broader range of contexts.

7. Extension to a Common Architecture

7.1 Approach to Extension

The architecture described in Section 6 was designed for the specific use case of software supply chain governance through AI-SBOMs. This section explains how this architecture can be extended, organized into two stages.

The first stage is extension within the domain of AI-driven development. While the AI-SBOM targeted by this initiative represents one aspect of trust mechanisms in AI-driven development, there are other areas within AI-driven development that require the establishment of trust. Examples include the assignment of Verifiable Identity to AI agents as described in UC4 (Section 4.4), and verifiable records of the review and approval processes for AI-generated code. The design patterns of the trust-related component group built in Section 6 (SBOM Generation Engine, VC Issuance/Verification/Management, Security Analysis Engine, and Analyzer Agent) are applicable to these areas as well. Specifically, the design pattern of combining a component that collects and structures domain-specific data with a UWI integration component that issues and verifies it as a VC can be commonly applied to other trust mechanisms in AI-driven development.

The second stage is extension to other domains beyond AI-driven development. The essence of this initiative's architecture lies in the trust mechanism of "issuing and verifying structured data as Verifiable Credentials." This mechanism is applicable to any domain that requires trustworthy data recording and verification. Section 7.2 introduces the Worker Credential (qualification information management) use case as a concrete example.

7.2 Example Extension Use Case: Worker Credential

One extension beyond AI-driven development is the Worker Credential (qualification information management) use case.

In the Worker Credential use case, there is value in issuing and managing qualification information—such as language certifications and professional certifications—as VCs. By recording qualification information issued by a certification authority (Issuer) as VCs, the authenticity of qualifications becomes cryptographically verifiable, and hiring organizations or educational institutions (Verifiers) can independently verify the validity of qualifications without directly confirming with the qualification holder (Holder). It also becomes possible to selectively disclose only the necessary qualification information while protecting the Holder's privacy.

From the perspective of commonality with this initiative's architecture, the following design patterns are reusable. The component that collects and structures domain-specific data (corresponding to the SBOM Generation Engine in AI-SBOM) handles the collection and structuring of qualification information in the Worker Credential use case. The UWI integration component that issues and verifies VCs functions as a fundamental trust mechanism for VC issuance and verification compliant with the W3C VC Data Model standard, regardless of the use case. The agent implementation approach operating on the agent execution platform (Amazon Bedrock AgentCore) is also common, and agents such as a qualification matching agent for Worker Credentials can be built on the same execution platform.

In this way, the architecture established in this initiative can be extended to use cases in different domains by replacing the domain-specific business logic (code analysis and SBOM generation for AI-SBOM; qualification information collection and matching for Worker Credential).

7.3 Design Principles for Extension

The following design principles apply when extending the common architecture to new use cases.

Reusability: The VC issuance and verification pattern through UWI integration, the agent execution platform, and the standardized integration approach through the MCP protocol can be reused without modification for new use cases.

Extensibility: When adding new use cases, integration with the existing infrastructure can be achieved simply by implementing a domain-specific data collection and structuring component and the VC issuance logic for that data.

Interoperability: Compliance with W3C DID/VC standards and A2A/MCP protocols enables agents from different use cases to collaborate using the same protocols. By exchanging information between the AI-SBOM Analyzer Agent and the Worker Credential qualification matching agent, it becomes possible in the future to build cross-domain trust mechanisms—for example, verifying the provenance of "code generated by a developer with specific qualifications using an AI agent" from both the qualification information and code provenance perspectives.

Governance Consistency: By using the same UWI trust anchor across all use cases, trust mechanisms such as issuer authenticity of VCs, data tamper detection, and revocation management are operated under unified standards. This means that even VCs from different domains—such as AI-SBOM and Worker Credential—can have their authenticity confirmed through the same verification process, maintaining a consistent governance framework across the entire organization.

8. Future Outlook

8.1 Direction Indicated by This Initiative

The architecture described in this white paper provides a concrete technical answer to the specific challenge of software supply chain governance in AI-driven development, but it also serves as an entry point to a broader question: "How should trust be structured in a world where AI acts autonomously?"

A key insight that has emerged through this initiative is that trust should not be applied after the fact but should be embedded within workflows. VC-SBOMs and AI-SBOMs automatically generate trust records each time code is generated, modified, or released. This is fundamentally different from the approach of collecting evidence after the fact for auditing purposes.

8.2 Directions for Extension

The common architecture established in this initiative can be extended in multiple directions.

Use Case Extension: Starting with the Worker Credential described in Section 7.2, application to any domain where the combination of structured activity data and Verifiable Credentials provides value is conceivable. Domains with stringent regulatory requirements and a need for auditability—such as medical device software certification, transaction provenance attestation

in financial services, and supply chain transparency in manufacturing—are particularly promising extension targets.

Agent Ecosystem Extension: As the assignment of Verifiable Identity to AI agents envisioned in UC4 becomes widespread, UWI's role as an infrastructure for recording trust chains between agents in a verifiable manner will expand. In multi-agent workflows spanning multiple organizations, a mechanism that can cryptographically prove each agent's actions will become the foundation of enterprise AI governance.

Data Layer Deepening: This initiative centers on structured activity data (SBOM events, vulnerability detection events, etc.), but in the future, integration with unstructured data (design documents, review meeting recordings, conversation logs, etc.) will enable the establishment of trust based on deeper contextual understanding. While structured data alone is limited to pattern recognition of "what was done," integrating unstructured data enables reasoning that understands the context of "why it was done."

8.3 Contributions to Standardization and the Ecosystem

This initiative utilizes a suite of standards, specifications, and protocols that prioritize interoperability, such as W3C DID/VC, CycloneDX/SPDX format SBOM, and A2A/MCP. This will facilitate future ecosystem collaboration and adherence to standardization trends without excessive reliance on specific vendor implementations. Compliance with these standards is an important choice for ensuring interoperability with the broader ecosystem while avoiding dependence on any specific vendor.

Additionally, the reference architecture established in this initiative aims to contribute to the formation of industry standards as a concrete implementation example of software supply chain governance in AI-driven development. In particular, the following two areas are domains where standardization has not yet sufficiently progressed, and we believe the outcomes of this initiative can contribute to future standardization discussions.

Data Model and VC Issuance Flow for AI-SBOM: There is no established cross-industry standard for how to represent AI-generated code-specific metadata (instructor information, AI agent information, model information, delegation chains, etc.) as an SBOM and issue and verify it as a Verifiable Credential. The AI-SBOM data model and VC issuance flow defined in this initiative can contribute to standardization discussions as a concrete implementation example in this area.

DID Assignment to AI Agents: The mechanism for assigning DIDs (Decentralized Identifiers) to autonomous AI agents and making agent actions cryptographically provable is an area where standardization has not yet sufficiently progressed. Many challenges remain to be resolved, including methods for representing delegation chains in multi-agent environments, agent identity lifecycle management, and revocation mechanisms. The UC4 architecture implemented in this initiative can contribute to discussions on extending the W3C DID specification and the Verifiable Credentials Data Model as a pioneering example in this area.

8.4 Conclusion

Now that AI has become a core actor in development workflows, a new approach is needed to ensure the trustworthiness of the software supply chain. The VC-SBOM and AI-SBOM architecture proposed in this white paper provides a concrete and implementable answer to this challenge.

By combining DID/VC—a mature trust technology—with Amazon Bedrock AgentCore—a scalable agent execution platform—provenance attestation for AI-generated code, continuous security assurance, and compliance with enterprise governance can be simultaneously achieved.

This initiative begins with the specific domain of AI-driven development, but its design principles and common architecture are conceived with extension to any domain that requires trustworthy data recording and verification. We are confident that building verifiable trust mechanisms now, as AI becomes part of social infrastructure, will form the foundation for sustainable AI adoption.

UWI Universal Wallet Infrastructure

docomo Global accenture aws

Learn more
universalwalletinfra.com

Get in touch
ndg_uwi@ml.nttdocomo.com