# Technical White Paper
# (Zen7 Payment Agent)

## 1. Abstract and Overview

**Zen7** Payment Agent stands as the inaugural practical implementation of the DePA (Decentralized Payment Agent) framework, pioneering the next generation of intelligent payment infrastructure. It not only fully realizes the core functionalities of DePA but has also been successfully deployed with innovative application cases within the field of agent-enabled e-commerce.

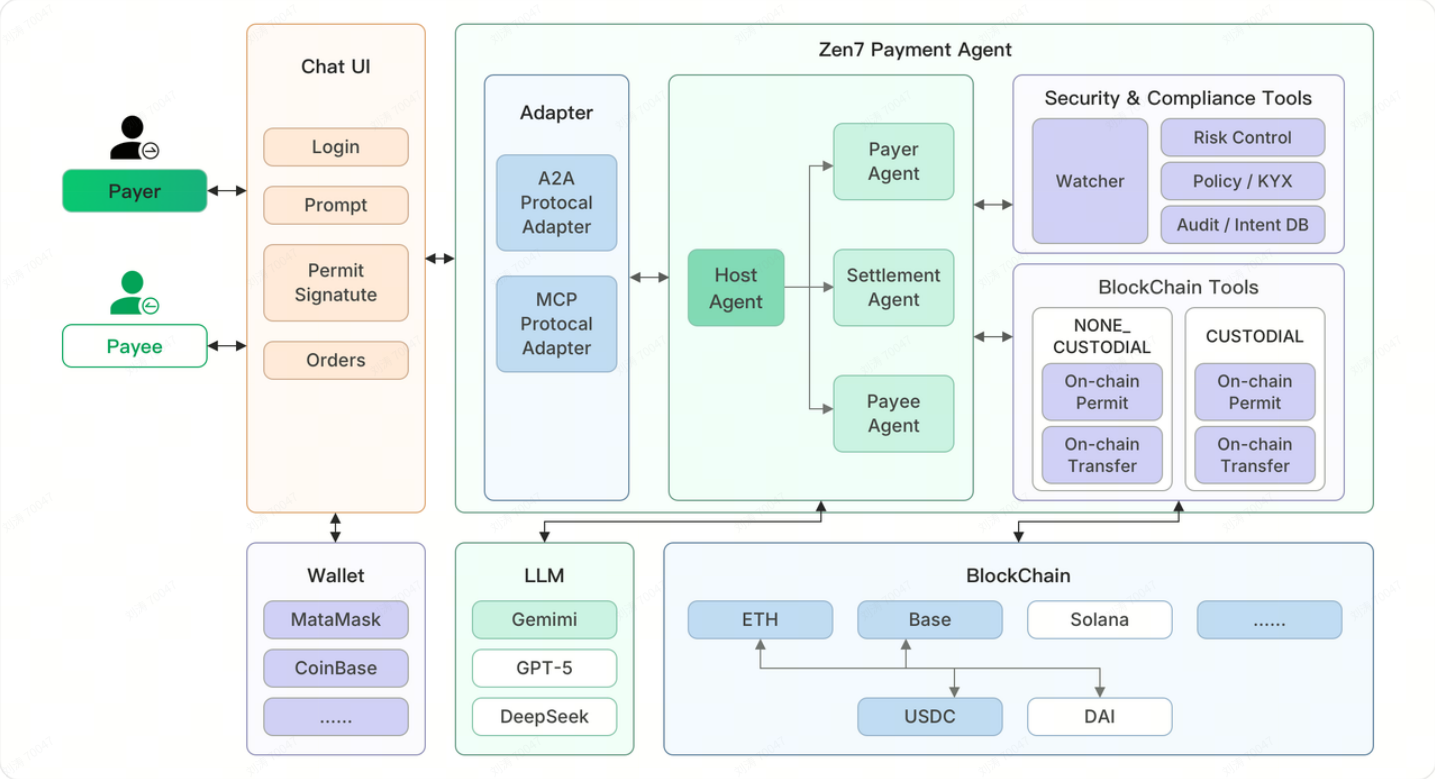
As the first practical project within the DePA ecosystem, Zen7 achieves several key characteristics: automated encrypted payments among intelligent agents, an "authorized password-free" mechanism, and intent recognition and interaction powered by large language models.

By supporting multi-chain, multi-wallet, and multi-currency operations, integrating robust security mechanisms, and delivering an exceptional user experience, Zen7 Payment Agent establishes a seamless "intent-to-result" payment journey.

## 2. Technical Architecture

Zen7 Payment Agent employs a multi-agent collaborative architecture designed to deliver a comprehensive payment solution. It provides support for both A2A (Agent-to-Agent) and MCP (Model Context Protocol) protocols, enabling seamless communication and integration within decentralized ecosystems. The architecture natively supports both custodial and non-custodial fund management models, offering flexibility to accommodate diverse application requirements and user preferences for asset control.

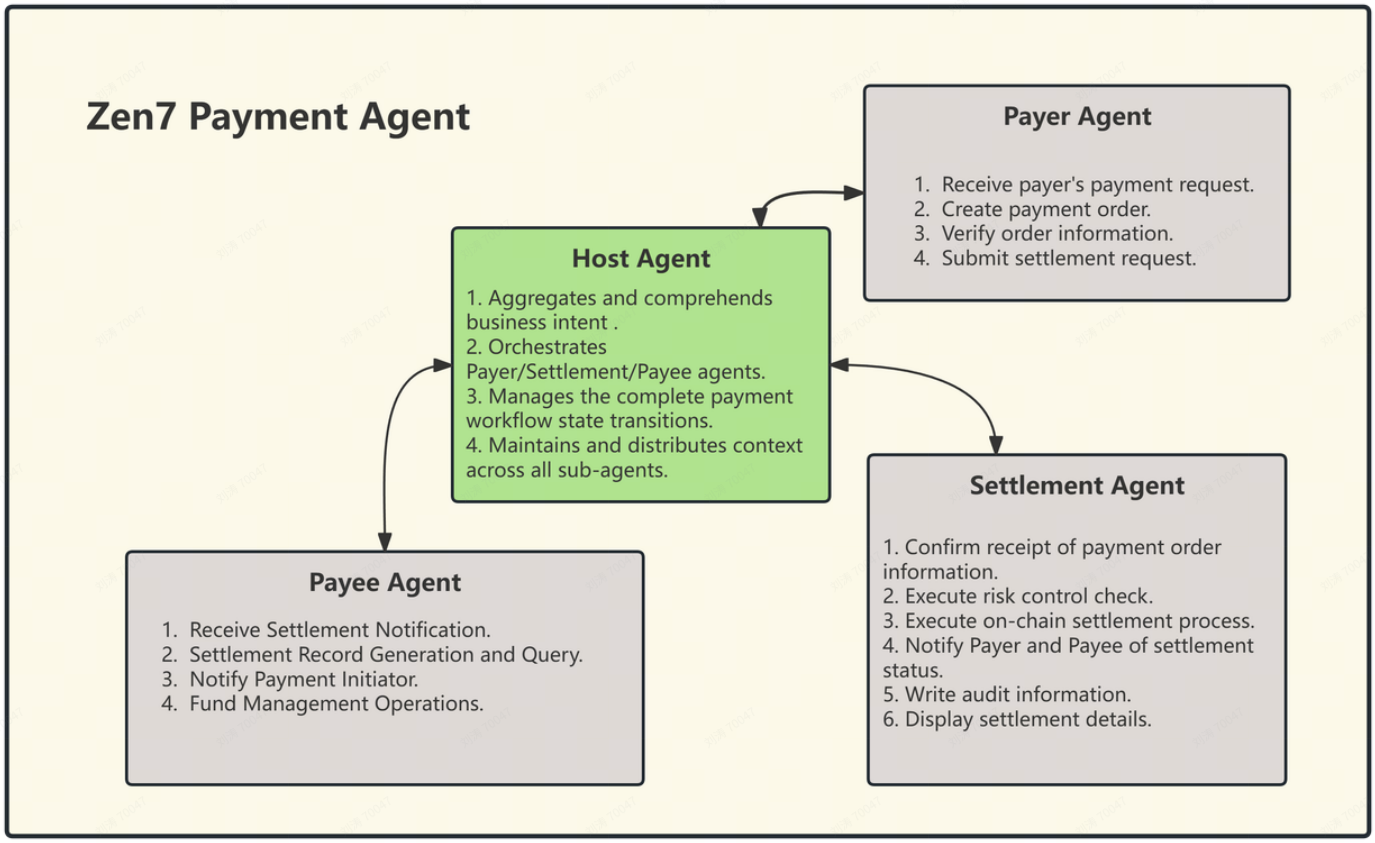
This architecture is engineered to provide a multi-chain, multi-currency, and multi-wallet payment solution specifically for AI Agents and native DApps. It is capable of handling high-frequency transactions while incorporating features such as a "authorized password-free" mechanism and solutions for gas fee abstraction, thereby creating a seamless and efficient payment experience from intent to result.




## 2.1 Multi-Agent Collaborative Architecture

Zen7 Payment Agent employs a "1+3" multi-agent collaborative architecture to implement decentralized payment processes. Within this framework, the Host Agent acts as the core coordinator, responsible for intent comprehension, intelligent routing, state management, risk control, and auditing . Three specialized agents operate with distinct responsibilities: the Payer Agent handles payment capability assessment, credit authorization checks, and the generation of signatures based on the EIP-712 standard; the Settlement Agent functions as the on-chain execution engine, completing processes such as risk control, authorization, and fund transfer, while supporting both custodial and non-custodial models ; and the Payee Agent is tasked with payment receipt confirmation, settlement record generation, and fund management.

The complete payment flow is as follows: the Host Agent routes the payment request → the Payer Agent performs validation and generates signatures → the Settlement Agent executes the on-chain settlement → the Payee Agent confirms receipt of funds. This sequence forms an end-to-end automated closed loop.




Beyond the core multi-agent architecture, Zen7 Payment Agent provides a comprehensive suite of supporting services designed to ensure compliance, observability, and operational efficiency. These services are fundamental to the platform's security, auditability, and manageability in a production environment.

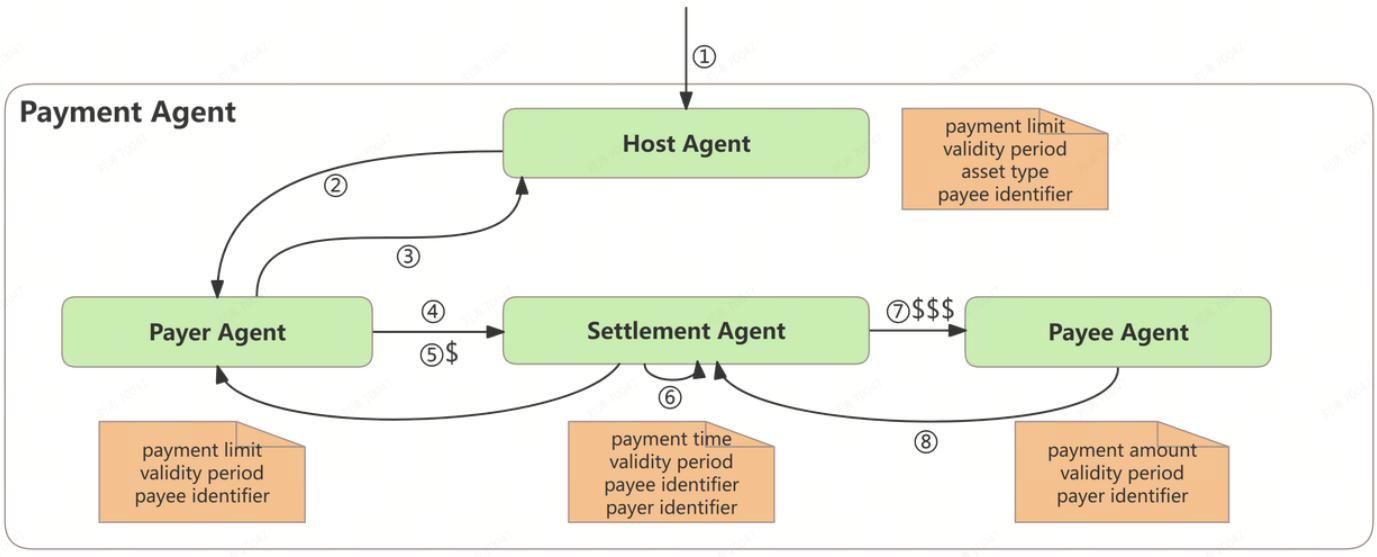
- **Notification & Alerting Service**: This service offers event subscription and push notifications for both end-users and various intelligent agents. It supports multi-channel integration, ensuring timely dissemination of critical information related to transaction status, system events, or potential risks.
- **Risk Control Engine:** The engine performs real-time transaction interception based on a combination of configurable policies, including quota limits, address whitelists, and on-chain risk scoring modules. All enforcement decisions are logged, providing a clear and auditable trail for risk management activities.
- **Event Monitor:** This component continuously monitors both on-chain events and interactions among the various internal agents. It writes these events to immutable audit logs and the internal ledger. The monitor also supports alert triggering, event replay for debugging, and comprehensive compliance tracing.
- **Audit Log & Ledger:** This service meticulously records critical data points, including on-chain transaction hashes, permission usage, approver signatures, and reconciliation vouchers. This creates a dual-ledger system that synchronizes on-chain records with off-chain audit data, ensuring a complete and tamper-evident record of all activities.

- **Policy & KYX Service:** Providing a unified compliance framework that covers KYC (Know Your Customer), KYS (Know Your Supplier), and KYA (Know Your Agent), this service performs identity verification and risk assessments for customers, merchants, and intelligent agents. It underpins transaction policies, manages whitelists and blacklists, and conducts necessary compliance checks.

## 2.2 Custodial Mode

In the Custodial Mode, the operational workflow is as follows: the Host Agent first completes intent validation and subsequently triggers the Payer Agent to perform an on-chain balance check and generate an EIP-2612 Permit signature. The Settlement Agent then verifies the signature, pays the gas fee on behalf of the user to execute the `permit` + `transferFrom` operations, thereby transferring the funds into a designated escrow wallet and recording the transaction in real-time. The funds are protected by risk control policies during the custodial period. Upon reaching the settlement window, the funds are released to the Payee Agent after a review process. Throughout this entire process, the Event Monitor and Audit Log & Ledger maintain comprehensive records. Should any anomalies occur, contingency procedures such as re-authorization, interception, arbitration, or refunds are triggered.

A key feature of the Custodial Mode is that the Settlement Agent uniformly covers the Gas fees, providing users with a genuine "Zero-Gas" payment experience. This encompasses both the custodial and settlement phases, ensuring that both the Payer and the Payee are entirely free from Gas burdens throughout the transaction.



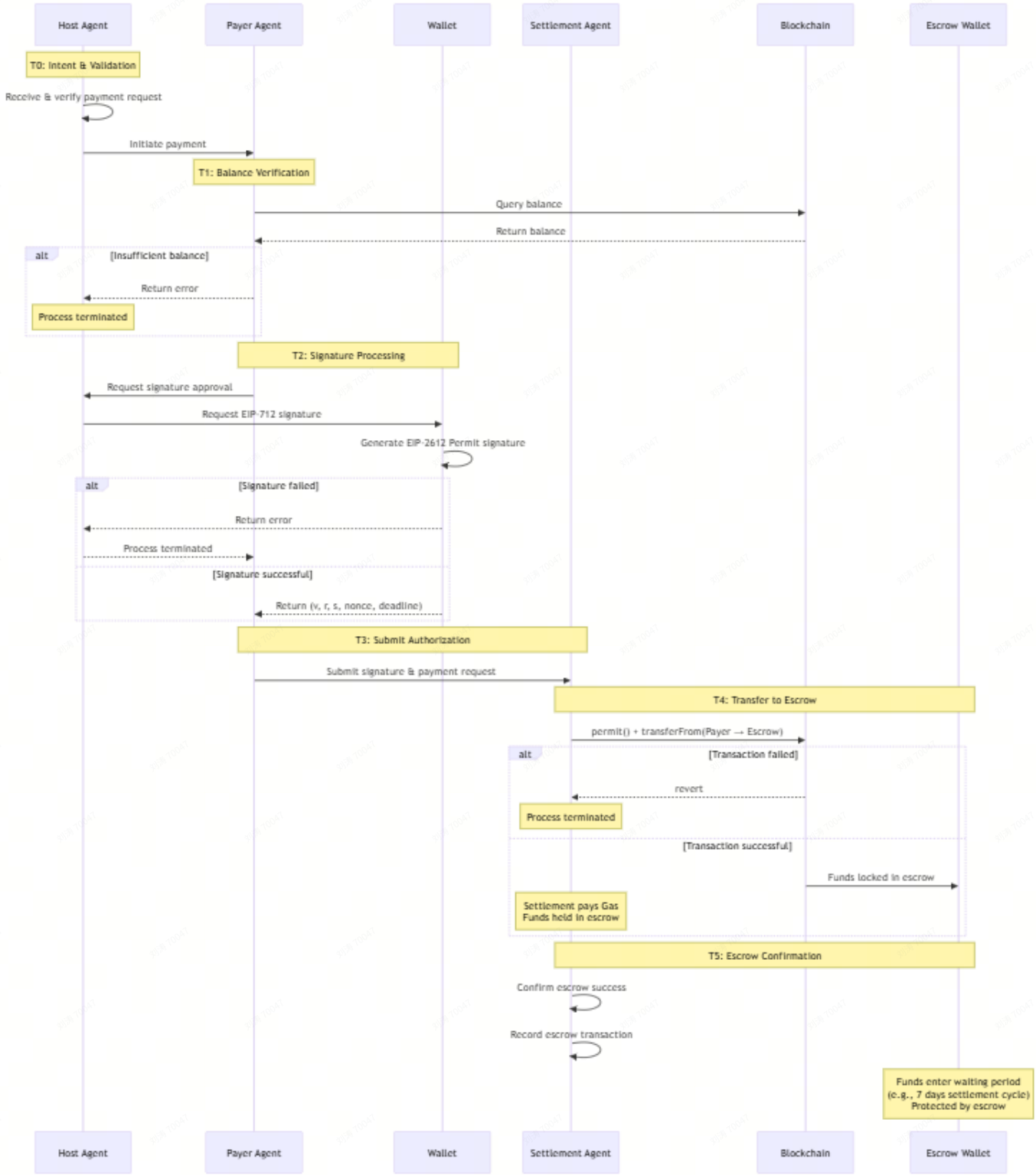

The Zen7 Payment Agent operational workflow is a sequence of discrete, automated steps, meticulously designed to ensure security, reliability, and a seamless "zero-Gas" experience for both Payers and Payees. The following sections detail the flow from intent initiation to final settlement and auditing.

- **T0 Intent & Validation:** The Host Agent receives the payment request. It performs initial validation of the request parameters, including asset type, amount, and validity period. Upon successful validation, it creates a unified Payment Agent Context and routes the request.
- **T1 Balance Check:** The Payer Agent queries the on-chain balance to verify the Payer's fund sufficiency. If sufficient, it proceeds to initiate the payment. If funds are insufficient, the workflow terminates, and an error is returned, prompting the need to re-initiate the payment request.
- **T2 Signature Processing:** The Payer Agent initiates a signature request. It instructs the connected wallet to generate EIP-712 structured signature data compliant with the EIP-2612 Permit standard. If the signature fails, the workflow terminates, and an error is returned.
- **T3 Authorization Request Submission:** The Payer Agent receives the signature data and submits the authorization and payment request to the Settlement Agent.
- **T4 Authorization & Transfer (Escrow Funding):** The Settlement Agent utilizes the EIP-2612 Permit mechanism to execute the authorization. The token contract verifies the signature and establishes the spending allowance. The Settlement Agent then immediately calls the `transferFrom` method, transferring the funds from the Payer's address into the designated escrow wallet, thereby completing the Payer's payment step. The Gas fee for this transaction is covered by the Settlement Agent. If authorization or transfer fails, the workflow terminates with an error.
- **T5 Escrow Confirmation:** The Settlement Agent confirms the successful escrow transaction and records the escrow details. The funds enter a holding period as defined by business rules, during which they are under escrow protection.
- **T6 Settlement Processing:** Upon meeting the pre-defined settlement conditions, the Settlement Agent initiates the settlement process. It performs final risk control and compliance checks before transferring the funds from the escrow wallet to the Payee's address. If settlement fails, an exception handling routine is triggered. The Gas fee for this transaction is also covered by the Settlement Agent.
- **T7 Settlement Confirmation:** The Settlement Agent confirms the successful settlement transaction. The Payee Agent receives the payment confirmation notification, marking the completion of the core payment process.
- **T8 Auditing & Notification:** The Event Monitor captures transaction events from both the escrow and settlement phases. The Audit Log & Ledger records the complete transaction link, ensuring an immutable and auditable trail.

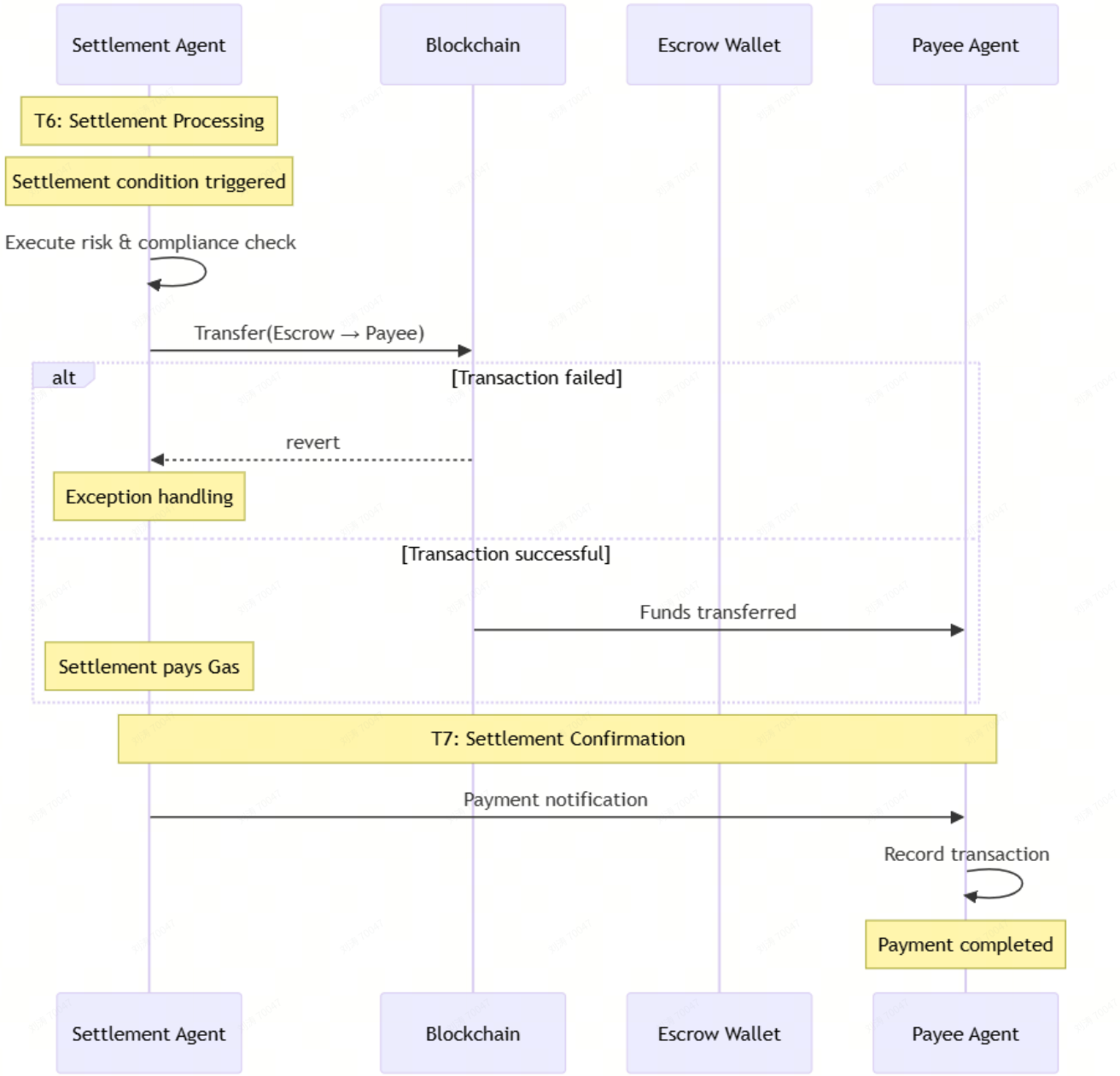
- **Exception & Dispute Resolution:** The system supports robust exception handling mechanisms, including but not limited to, authorization renewal, policy-based interception, dispute arbitration, and fund refunds, ensuring operational integrity and user asset safety throughout the payment lifecycle.

The payment sequence diagram illustrates the message and data flow among the multiple agents across the stages of Intent → Sign → Payment → Settlement.

- Funds Transfer Phase: This phase encompasses the critical steps from intent validation to the successful escrow of funds.



- Settlement Phase: This phase involves the release of funds from escrow to the Payee upon meeting predefined conditions.




All state changes throughout these phases generate corresponding on-chain events. These events are captured by the Event Monitor and immutably written into the Audit Log & Ledger, collectively forming a complete and replayable evidence chain. The key events defining this chain are detailed in the table below.

| Event Name | Trigger Point | Corresponding Workflow Stage | Purpose |
|---|---|---|---|
| IntentCreated | T0 - Payment intent creation | Intent & Validation | Records the origin and parameters of the payment request. |
| PermitSigned | | Signature Processing | |

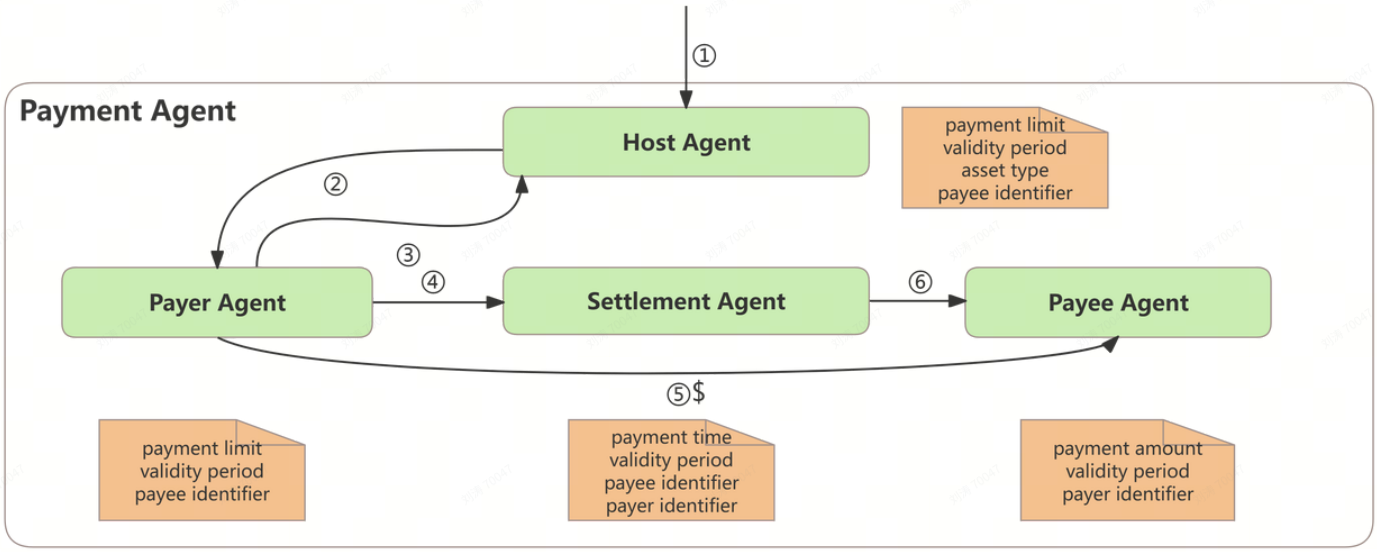
| | | | |
|---|---|---|---|
| | T2 - Signature generation | | Records the EIP-2612 Permit signature data. |
| FundsEscrowed | T4 - Funds successfully placed in escrow | Authorization & Transfer | Marks the transfer of funds into the escrow wallet. |
| FundsReleased | T6 - Funds released | Settlement Processing | Marks the transfer of funds from the escrow wallet to the Payee. |
| SettlementCompleted | T7 - Settlement confirmed | Settlement Confirmation | Confirms the Payee has received funds, marking payment completion. |
| TransactionFailed | Transaction failure | Exception Handling | Records the reason for the failure. |
| DisputeInitiated | Dispute initiated | Exception Handling | Triggers freezing of escrowed funds pending arbitration. |

## 2.3 Non-Custodial Mode

In the Non-Custodial Mode, the operational workflow is as follows: the Host Agent first completes intent validation and triggers the Payer Agent to perform an on-chain balance check and generate an EIP-2612 Permit signature. The Settlement Agent then verifies the signature, pays the Gas fee required for the transaction, and executes the `permit` + `transferFrom` operations. This results in the direct transfer of funds from the Payer's wallet to the Payee's wallet. Upon successful transfer, the transaction is recorded in real-time. Throughout this entire process, the Event Monitor and Audit Log & Ledger maintain comprehensive records. Should any anomalies occur, predefined contingency procedures—such as re-authorization, transaction interception, arbitration, or refunds—are automatically triggered.

A defining characteristic of the Non-Custodial Mode is that the Settlement Agent covers the Gas fees specifically for the on-chain `transferFrom` operation during the fund transfer phase. This mechanism ensures that both the Payer and the Payee experience a seamless, "Zero-Gas" transaction from initiation to completion, eliminating the financial and operational burden of blockchain transaction fees for the end-users.

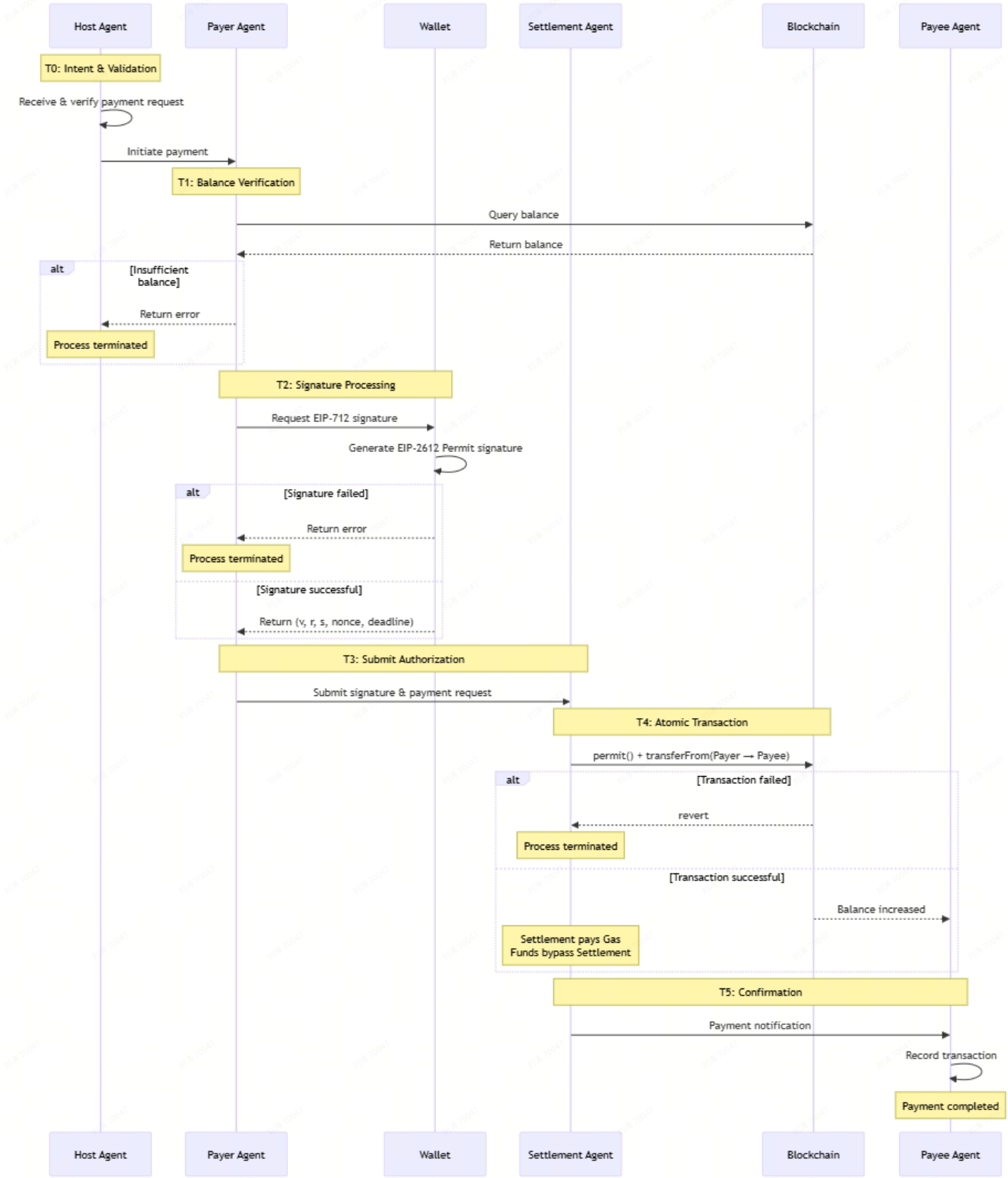
The operational workflow for the Non-Custodial Mode, where funds are transferred directly from the Payer to the Payee, consists of the following sequenced stages:




- **T0 Intent & Validation:** The Host Agent receives the payment request and validates parameters such as asset type, amount, and validity period. Upon successful validation, it creates the Payment Agent Context for the transaction.
- **T1 Balance Check:** The Payer Agent queries the on-chain balance to verify the Payer's fund sufficiency. If sufficient, it proceeds to initiate the payment. If funds are insufficient, the workflow terminates and an error is returned, indicating that the payment request should be re-initiated.
- **T2 Signature Processing:** The Payer Agent initiates a signature request. It instructs the connected wallet to generate an EIP-712 structured signature compliant with the EIP-2612 Permit standard. If the signature fails, the workflow terminates and an error is returned.
- **T3 Authorization Request Submission:** The Payer Agent receives the signature data and submits the authorization and payment request to the Settlement Agent.
- **T4 Authorization & Transfer (Direct):** The Settlement Agent utilizes the EIP-2612 Permit mechanism to execute the authorization. The token contract verifies the signature and establishes the spending allowance. The Settlement Agent then immediately calls the `transferFrom` method, transferring the funds directly from the Payer's address to the Payee's wallet address. The Gas fee for this transaction is covered by the Settlement Agent. If authorization or transfer fails, the workflow terminates with an error.
- **T5 Transfer Confirmation:** The Settlement Agent confirms the successful on-chain transfer. The Payee Agent receives the payment confirmation notification, records the transaction information, and the core payment process is completed.
- **T6 Auditing & Notification:** The Event Monitor captures the on-chain transfer event. The Audit Log & Ledger records the complete transaction trace, ensuring all transaction data is transparent, queryable, and fully supportive of audit trails.

- **Exception & Dispute Resolution:** The system supports robust exception handling mechanisms, including authorization renewal, policy-based interception, dispute arbitration, and fund refunds, ensuring operational integrity under various exceptional conditions.

The payment sequence diagram illustrates the message flow and transaction interactions among the multiple agents across the core stages: Intent → Sign → Payment. The entire process is underpinned by a robust event-tracking mechanism, which captures critical state changes to ensure full auditability. The following table details the key events that form this immutable, replayable evidence chain.




All state changes generate on-chain events, which are written into tracking logs and the audit database to form a replayable evidence chain:

| Event Name | Trigger Node | Key Information | Purpose |
| --- | --- | --- | --- |
| IntentCreated | Intent Creation | | |

| | | | |
|---|---|---|---|
| | | intentId, payment parameters | Marks the starting point of the payment and solidifies the request parameters. |
| PermitSigned | Authorization Signing | Signature digest, deadline | Records the authorization credential and supports replay verification. |
| TransferCompleted | Fund Transfer | payer/payee, amount, txHash | Confirms the funds have been transferred from the payer to the payee. |
| PaymentSettled | Payment Settlement | payee, timestamp | Confirms the payee has received the funds, closing the business process. |
| AuditRecorded | Audit Logging | blockNumber, auditHash | Aligns on-chain and off-chain data, supporting tracing. |
| PolicyBlocked / TransactionFailed / DisputeInitiated | Exception Handling | Blocking rule, error code, dispute reason | Captures risks, failures, and disputes, triggering subsequent processes. |

# 3. Core Technologies and Innovations

## 3.1 Summary of Technical Advantages

Zen7 Payment Agent, as the first implemented practical project of DePA (Decentralized Payment Agent), leads the new generation of intelligent payment infrastructure. It not only fully realizes the core functionalities of DePA but has also successfully deployed innovative application cases within the field of agent-driven commerce.

Zen7 Payment Agent achieves several key characteristics, including automated encrypted payments between intelligent agents, verifiable authorization that safeguards user sovereignty, and intent recognition and interaction powered by large language models.

By supporting multi-chain, multi-currency, and multi-wallet operations, incorporating robust security mechanisms, and delivering an exceptional user experience, Zen7 creates a seamless "intent-to-result" payment journey. It provides a perfect solution for high-frequency, micro-payment scenarios, laying the groundwork for the payment infrastructure of the AI Agent economy era.

### 3.1.1 Technical Advantages

The core competitiveness of Zen7 Payment Agent can be summarized by the following key points:

- Agent-Native Architecture & Automated Payments: Designed from the ground up for AI Agent collaboration, it supports dual A2A/MCP protocols, enabling intent-driven automation of the entire payment process. It allows agents to autonomously negotiate and execute transactions among themselves, establishing payment flows without human intervention. Different Agents can negotiate terms based on predefined rules, execute transactions, and complete settlements autonomously.
- Threefold Technical Breakthrough:
  - Authorization without Passwords (EIP-712/EIP-2612): Provides a Web2-level user experience while maintaining non-custodial security. A single authorization supports batch transactions.
  - Multi-Chain, Multi-Currency, Multi-Wallet Support: Natively supports multiple EVM chains, various stablecoins, and mainstream wallets, offering maximum flexibility.
  - Gas-Free Experience: The Agent covers Gas fees, completely eliminating user payment friction.
- Exceptional User Experience: Zen7 Payment Agent thoroughly shields users from the underlying complexities of Web3, providing a seamless "intent-to-result" experience that greatly meets the needs of mainstream users. Users only need to state their payment goal; achieving the expected outcome does not require their involvement in cumbersome operational processes.
- Enhanced Security: Users are not required to sign each transaction individually, thereby significantly reducing exposure to risks associated with malicious signature prompts. Agent behaviors are strictly confined within authorized scopes. Users can autonomously choose whether to self-custody private keys or opt for a custodial solution.
- Intrinsic Composability: Functioning as a payment layer infrastructure that connects isolated ecosystems such as DeFi, DePIN, GameFi, and SocialFi, Zen7 Payment Agent fosters the emergence and development of complex on-chain application ecosystems.

## 3.2 Multi-Chain & Multi-Currency Strategy

### 3.2.1 Multi-Chain Support Strategy

DePA Chain is designed specifically to serve the high-frequency, micro-payment, and password-free payment scenarios of the AI Agent economy. As the native settlement layer for A2A payments, it possesses the following characteristics:

| Blockchain | Core Advantages | Potential Limitations |
| --- | --- | --- |
| Ethereum L1 | Native issuance of mainstream stablecoins; High security; Strong compliance recognition | High Gas costs; Long confirmation times |

| | | |
|---|---|---|
| Base | Backed by Coinbase ecosystem; Regulatory facilitation; Smooth connectivity with CEX liquidity | Degree of decentralization under development; Relatively fewer assets |
| BNB Chain (BSC) | High throughput; Low fees; Strong liquidity in emerging markets | Relatively lower degree of decentralization; Weaker regulatory reputation |
| Polygon | Low fees; Fast confirmation; Active ecosystem | Security slightly lower than L1; Limited support for some stablecoins |
| Arbitrum | Strong Ethereum ecosystem compatibility; Low Gas fees; Large user base | Relatively concentrated ecosystem; Limited support for some assets |
| DePA Chain | Optimized for A2A: Extremely low Gas, millisecond-level confirmation, native support for authorization without passwords; Deep integration of Agent identity and payment protocols; Self-controlled, customizable governance | Early-stage ecosystem development; Requires cross-chain bridging solutions |

- Ultimate Performance Optimization: Deeply optimized for A2A payment scenarios, supporting second-level confirmation, extremely low Gas fees (near-zero cost), and capable of handling millions of TPS for inter-Agent payments.
- Native Protocol Integration: Natively supports core functionalities at the blockchain level, including the EIP-712/2612 authorization without passwords mechanism, Agent identity management (KYA), policy engines, and audit tracing.
- Dedicated Agent Ecosystem: Provides a customized economic model, incentive mechanisms, and governance structure for AI Agents, fostering a prosperous Agent ecosystem.
- Cross-Chain Interoperability: Connects with major L1s/L2s via cross-chain bridges, enabling free flow of assets; utilizes L1 as the final settlement layer for security, while L2 handles high-frequency transactions.
- Cost Structure Advantage: The cost of Agents covering Gas fees on L2 is controllable, enabling a sustainable business model for the user "Zero-Gas" experience.

### 3.2.2 Multi-Currency Support Strategy

Zen7 Payment Agent employs a layered architecture of "Agent collaboration + smart contracts" to deliver flexible, secure, and scalable multi-currency payment capabilities, providing the payment infrastructure for the AI Agent economy.

| Currency | Core Advantages | Potential Limitations / Risks | Primary Supported Chains |
|---|---|---|---|
| USDC | High regulatory recognition, direct fiat redemption, strong liquidity on major chains | Risk of confusion between native and bridged versions; issuer allowlists/policies require ongoing monitoring | Ethereum, Base, Arbitrum, Polygon |
| USDT | Wide adoption, strong liquidity, abundant trading pairs | Differences in issuance and compliance disclosure; diverse on-chain versions require strict address management | Ethereum, Base, Arbitrum, Polygon, BNB Chain, TRON |
| PYUSD | Strong compliance backing, brand trust, Ethereum ecosystem integration | Relatively limited on-chain coverage and liquidity; early-stage ecosystem | Ethereum |
| EURC | Euro-denominated, clear compliance path | Primarily suited for Eurozone scenarios; limited use in cross-regional contexts | Ethereum |
| ENA | Innovative yield-bearing mechanism, deep integration with Ethereum DeFi ecosystem | Reliance on staking yields and derivative funding rates; smart contract complexity risk; early adoption phase | Ethereum, Arbitrum |
| DAI | High degree of decentralization, DeFi ecosystem friendly | Peg stability pressure during extreme market conditions; parameter/governance change risks | Ethereum, Arbitrum, Base, Polygon |
| LUSD | Pure crypto-collateralization, strong decentralization narrative | Relatively limited liquidity; volatility risk from liquidation mechanisms | Ethereum |
| FRAX | Flexible mechanism, active ecosystem | High mechanism complexity; stability pressure under extreme conditions | Ethereum, Arbitrum, Base, Polygon, BNB Chain |

## 3.3 Multi-Wallet Service (Wallet Execution Service)

Current Wallet Adaptation Achievements

Deep integration has been completed for mainstream EVM-standard hot wallets, such as MetaMask and Coinbase Wallet . Through a unified wallet adapter architecture, core operations—

including on-chain signing, asset authorization, and balance queries—have been standardized across different wallets. This approach not only meets users' daily on-chain interaction needs but also significantly reduces the complexity of multi-wallet compatibility for developers, ensuring consistent and stable operational experience .

Future Wallet Ecosystem Expansion Plan

The future development of wallet capabilities will advance along two dimensions: "scenario coverage" and "security upgrade". The specific plans include:

- Diversifying Wallet Types: Integration of multi-signature wallets (e.g., Safe, Fireblocks MPC multi-signature, Cobo) to cater to more cross-chain scenarios and collaborative team asset management requirements.
- Enhancing Asset Security Level: Plans are in place to support technologies such as MPC (Multi-Party Computation) wallets and multi-factor authentication, providing higher-level security assurance for high-net-worth asset users
- Priority will be given to integrating mainstream MPC solutions like Fireblocks, Safeheron, Coinbase WaaS, ZenGo, Web3Auth MPC, Cobo MPC, and OKX MPC .

## 3.4 Audit and Ledger

The planned "Ledger + Audit" functions work together, utilizing an event-driven dual-ledger system and a replayable audit trail to achieve a full-cycle closed loop from data collection and reconciliation report output to exception alerts and compliance evidence collection. This ensures that payment flows in both custodial and non-custodial scenarios are secure, transparent, and traceable for regulatory purposes.

- Ledger Functionality:

Based on an "Event and Observability" architecture, a dual-ledger model comprising an on-chain ledger and an off-chain business ledger is constructed. The Event Monitor captures core Agent-to-Agent events—such as intent creation, authorization signing, funds escrowed, funds transferred, and settlement completed—to form the off-chain ledger. Transaction hashes, block heights, agent-paid Gas fees, and settlement deadlines are written to the on-chain ledger, achieving synchronization between the on-chain and off-chain ledgers.

- Audit Functionality:

The Audit Service reuses the event stream to build a replayable payment process. It preserves the context of key nodes like transfer completion events and transaction failure events, supporting multi-stage evidence collection and dispute arbitration.

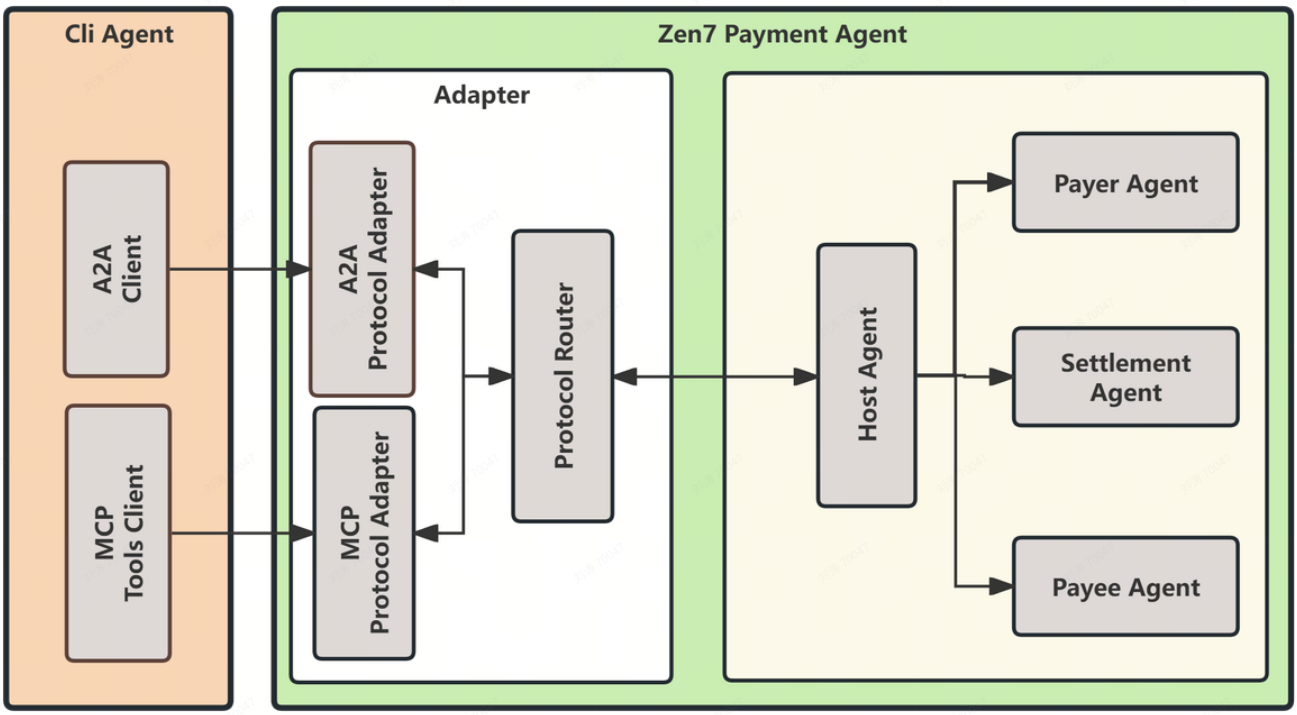
## 3.5 Dual Protocol Support: A2A & MCP

Zen7 Payment Agent adopts the philosophy of "thinnest possible adaptation" to bridge the A2A and MCP protocols. It provides a consistent semantic interface upstream while maintaining a unified invocation path and response format for the core Payment Agent internally. The design adheres to the principles of acting solely as a pass-through with format conversion, not adding business logic, not persisting state, and ensuring explicit failure returns. This is achieved through modular protocol adapters that support horizontal scaling.

### 3.5.1 Protocol Adaptation Layer Positioning

- Role: Positioned between the upstream ecosystems (A2A, MCP) and the core Payment Agent, it is responsible for protocol parsing, authentication validation, invocation orchestration, and response writing. Its input consists of heterogeneous payloads from multiple protocols, and its output includes protocolled or structured execution results and error messages.
- Responsibilities and Functional Components:
  - **Protocol Adapter:** Parses upstream requests, verifies signatures and credentials, and maps them into standard internal instructions.
  - **Invocation Orchestrator:** Uniformly forwards requests to core services, executing retries, circuit breaking, and rate limiting.
  - **Response Formatter:** Transforms core service responses into the upstream protocol format and attaches protocol-specific information.
  - **Observability Layer:** Collects metrics, logs, and traces from the invocation chain and drives alerts.

### 3.5.2 Adaptation Processing Flow




The entire process operates along a closed loop of "Client Initiation → Protocol Adaptation → Protocol Routing → Core Execution → Response Writing."

- In the MCP scenario, the client accesses the adapter with a session token, tool identifier, and payload. The Session Manager isolates multi-tenancy using "Tenant ID + Session Token" and injects quotas. The Protocol Router then schedules the core execution, after which the Response Formatter outputs results containing success/failure status and metadata.
- In the A2A scenario, access occurs via REST, WebSocket, or message queues, carrying chain signatures and context. After completing identity verification, idempotency checks, and session read/write operations, the Protocol Router schedules the core execution based on the business domain. Finally, the Response Formatter generates a response containing status codes, error codes, and trace IDs. Failed requests can be retried using the idempotency key or trace ID. Sessions are automatically cleaned up upon expiration of their lifecycle or time-to-live (TTL).

# 4. Risk Control and Security

## 4.1 Design Principles

- Security by Design: Based on the defense premise that "potential risks always exist," multiple signatures, permission isolation, and real-time auditing mechanisms are mandatorily enforced for all critical paths. A layered defense ensures the core security boundary of funds cannot be breached.

- Scalability and Observability: Leveraging the multi-agent collaborative architecture, the Event Monitor and Invocation Orchestrator, supported by the policy engine and risk control engine, execute circuit breaking, quota limits, and rate limiting to protect the Payer, Settlement, and Payee Agents. On-chain events, metrics, logs, and traces are aggregated into the Audit Log and Ledger, triggering autonomous alerts and anomaly isolation during the payment process.
- Dual Assurance of Compliance and Privacy: KYX data is encrypted and isolated. The Travel Rule is executed through an independent control plane, supplemented by minimal visibility access, ensuring compliance with multi-regional regulations while protecting user privacy.

## 4.2 Key & Signature Management

### 4.2.1 Key Isolation Strategy

- Payer / Payee: The funding account wallet is fully self-custodied. The private key belongs solely to the user and is used only for autonomously signing authorized (Permit) transactions. No third party is involved in custody, ensuring complete user asset sovereignty and supporting offline recovery to guarantee asset control.
- Settlement Agent: Holds only the private key for signing policy instructions (compliant with the EIP-712 standard). This private key is permanently housed within an HSM (Hardware Security Module). Its invocation requires validation of a short-lived, valid token, and all operations are fully audited, preventing key misuse or lateral privilege escalation risks from the source, without involving control over user wallets.
- Escrow Wallet: Serves solely as an intermediate step in the fund flow and does not alter the self-custodied nature of user wallets. In custodial mode, funds are transferred into this wallet by the Settlement Agent, with the entire flow governed by pre-set business policies. In non-custodial mode, funds are directly transferred to the Payee's self-custodied wallet via a `transferFrom` transaction, bypassing any third-party escrow account.

### 4.2.2 Replay Attack Protection

A replay attack occurs when an attacker intercepts a legitimate transaction/signature and resends it repeatedly through unauthorized channels or times, attempting to release funds multiple times or reuse authorizations. Without constraints like terms verification, nonce incrementation, or chain binding, old signatures could be considered valid by the system, leading to duplicate fund transfers or authorization abuse.

Multi-Layer Protection:

- Terms Hash Verification: A unique identifier is generated using a hash algorithm. The formula is `termsHash = keccak(payer, payee, asset, amount, deadline, disputeKey, policyChecksum)`. If any core element in the transaction is tampered with, the `termsHash` changes, causing the corresponding signature to become invalid immediately, preventing "tampered replay" at the source.
- Time Window Limitation: The transaction signature includes a `deadline` (valid until time). Once this time is exceeded, the system outright rejects executing the operation corresponding to this signature. The signature cannot be used after the `deadline`.

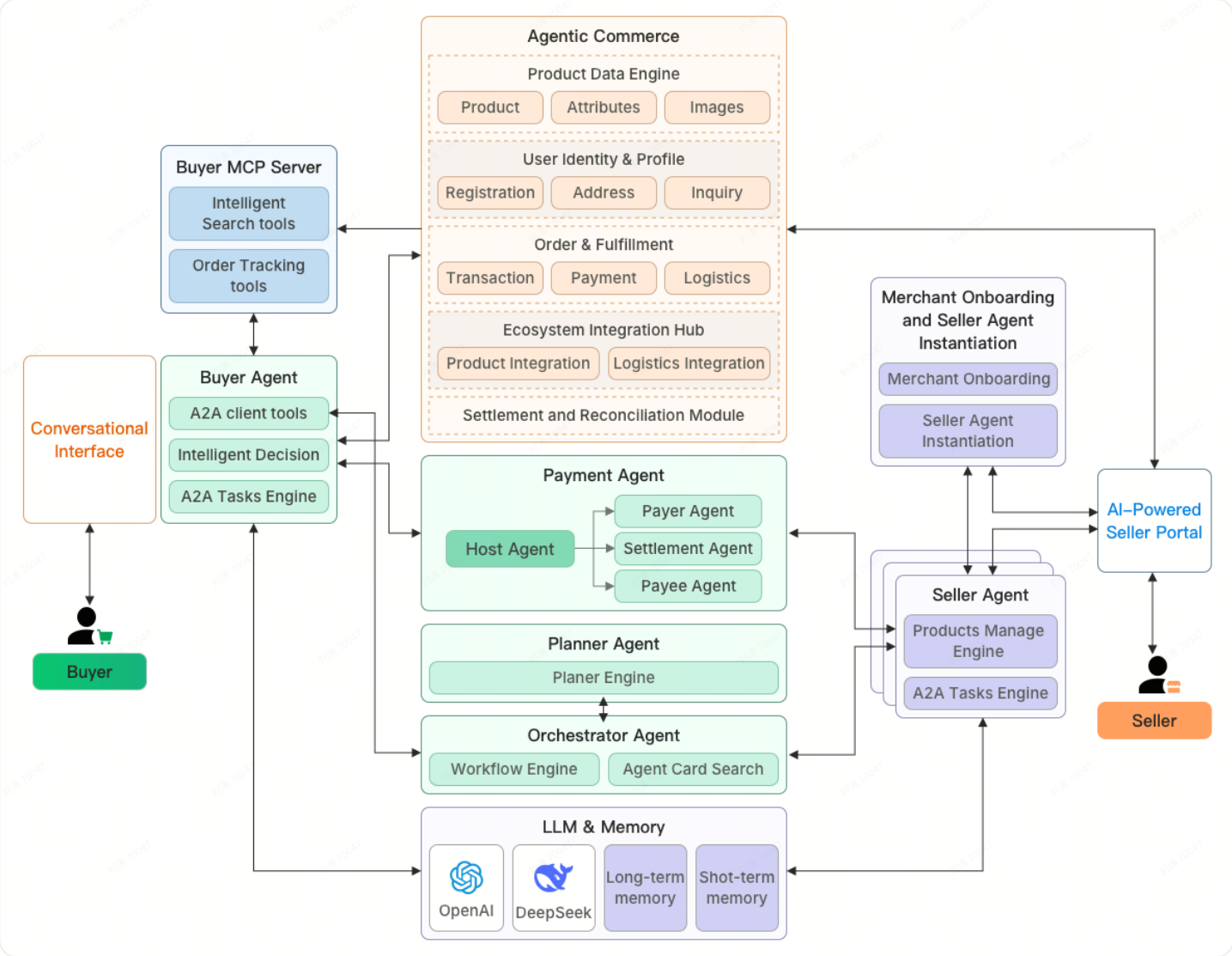
## 4.3 Compliance

The system implements a closed-loop operation for KYC / KYS / KYA, enabling layered compliance management:

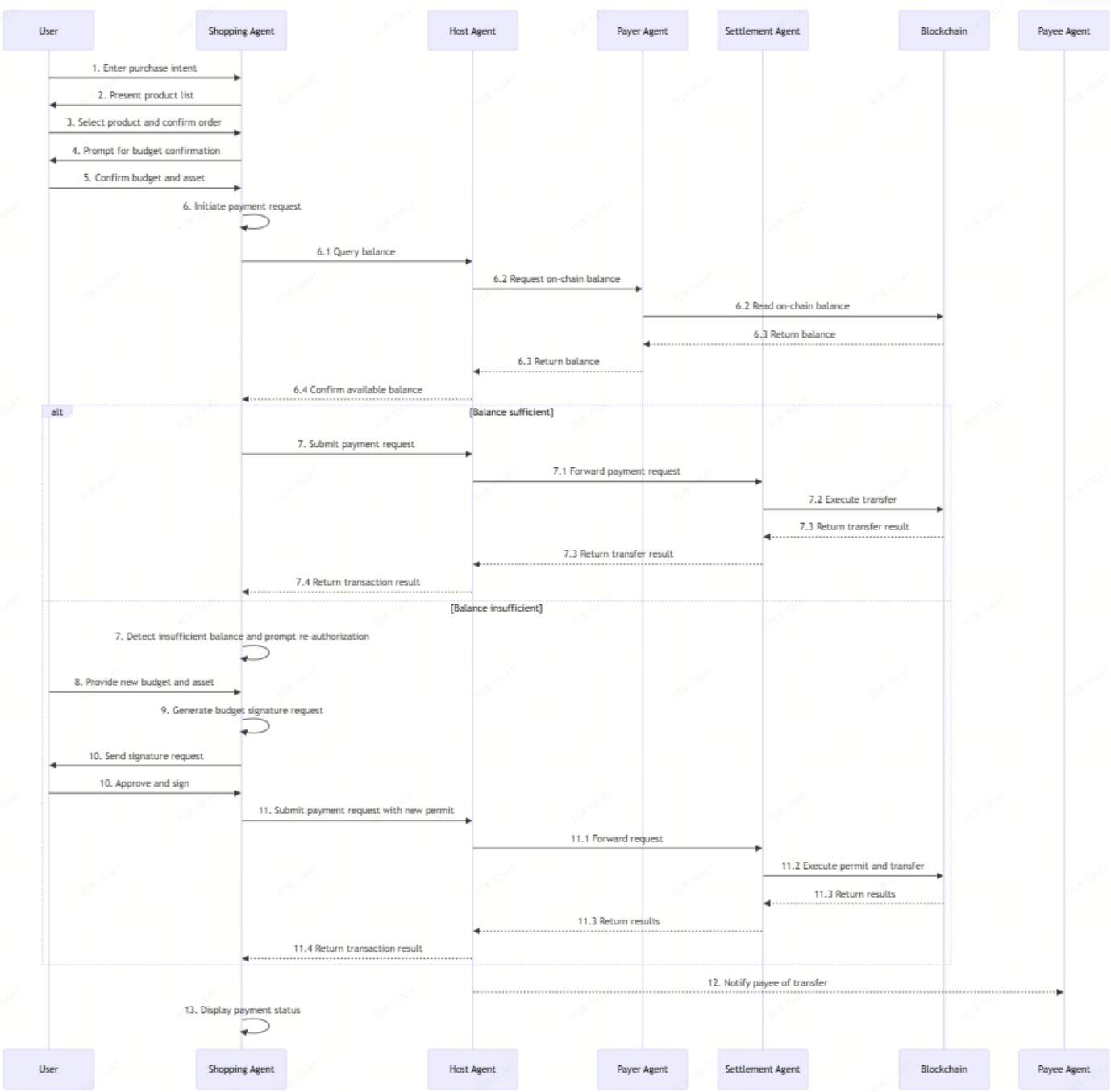
- Define three role types with clear positioning and core responsibilities for layered compliance control:
  - KYC (Know Your Customer): Focuses on the end customer, responsible for customer identity verification and sanctions list screening.
  - KYS (Know Your Seller/Supplier): Targets merchants or suppliers, conducting compliance qualification assessments and risk rating determinations.
  - KYA (Know Your Agent): Covers both human agents and AI Agents, managing their authorization scope and full-process behavior trails.
- Establish automated control mechanisms between roles triggered by risks:
  - When a customer's basic information is incomplete, restrictions are imposed on the initiation of business processes.
  - If a cooperating supplier is identified as higher risk, its risk rating is downgraded, and corresponding adjustments are made to its transaction limits.
  - When anomalous operations are detected for an agent role, corresponding management mechanisms are triggered, which may include the suspension of operational credentials.
- Ensure traceable and reviewable compliance decisions through monitoring and auditing tools:
  - The Event Monitor automatically generates two types of records for all KYX-related events: `KYXStatusChanged` records and `PolicyEscalated` records.
  - The Audit & Ledger system supports tracing the complete compliance decision chain corresponding to a payment intent using the `intentId`.

# 5. Typical Agentic-Commerce Application Case Study

## 5.1 System Architecture Diagram




## 5.2 Business Process Flow Diagram




## 5.3 Agentic-Commerce Business Processing Full Workflow

- **Intent Capture:** The user interacts with an LLM via a prompt or dialogue, which activates the Buyer Agent to drive the Shopping Agent, or directly invokes the Shopping Agent, initiating the e-commerce scenario
- **Order Preparation:** The Shopping Agent parses the user's intent, completes product selection, sets payment amount and validity period, generates the core fields required for the order, and synchronizes the key order points with the Seller Agent
  - The payment process is triggered at this stage. Only payment-related parameters (amount, asset type, validity period, etc.) are sent to the Host Agent, without retransmitting the detailed order items.

    - The Host Agent issues the payment instruction to the Payer Agent, which then performs checks on the amount, validity period, etc., preparing for payment execution.
- **Fund Escrow:** The Payer Agent transfers the digital assets to the Settlement Agent according to the instruction. The Settlement Agent escrows the assets and returns a Callback URL, prompting the Payer Agent to notify the payer of the current status (success/failure)
- **Fulfillment Trigger:** The Settlement Agent synchronizes the successful payment status with the Seller Agent (or Shopping Agent), instructing it to initiate the fulfillment/shipment process
- **Shipment & Notification:** Upon receiving the payment success Callback URL, the Seller Agent pushes a shipment notification to the Buyer Agent and executes the fulfillment action
- **Batch Settlement:** The Settlement Agent aggregates transactions based on threshold conditions (e.g., accumulating 1000 USDC, 20 transactions, or a 3-day cycle) and then initiates a one-time transfer operation to the Payee
- **Payout Confirmation:** After the Payee receives the successful transfer information, it sends a Callback URL back to the Settlement Agent, finally completing the settlement closed loop

## 5.4 Case Highlights

- **D2C Focus:** Serves high-value vertical industries that sell directly to consumers, providing end-to-end automated payment and fulfillment solutions
- **Paradigm Shift:** Upgrades from a manually-operated, search-based e-commerce model to an automated Agent-driven business形态
- **Experience Upgrade:** Enables seamless switching between conversational and embedded interfaces, achieving a more natural and efficient user experience
- **Technology Integration:** Integrates LLM cognition, process automation, data insights, and fintech capabilities to build a trusted payment system
- **Ecosystem Co-creation:** The platform serves as both Agent-driven commerce infrastructure and an incubator for the Agent economy, supporting autonomous negotiation between merchant and buyer Agents
- **Agent Assetization:** Allows for the tokenization of AI Agents and the splitting of revenue rights, leveraging smart contracts to create an incentive-compatible ecosystem model

# 6. Glossary and Appendix

| Term | Full Name / Source | Plain Explanation | Role in Zen7 |
| --- | --- | --- | --- |
| | | | |

| | | | |
|---|---|---|---|
| A2A Protocol | Agent-to-Agent Protocol | A standardized communication method that allows two automated agents to exchange instructions like humans sending emails. | The primary channel for the Host Agent to share intents and terms with Payer/Payee. |
| MCP Protocol | Model Context Protocol (MCP) | A standardized protocol for the discovery, negotiation, and secure invocation of tools and context between models/agents. | Settlement/Host exposes payment tools as MCP Servers, enabling low-friction integration for external Agents. |
| Escrow | Escrow Contract | Funds are first locked in a third-party contract and released only when conditions are met. | The security cornerstone guaranteeing "authorize once, release upon conditions". |
| EIP-712 | Ethereum Improvement Proposal 712 | An Ethereum signature standard that binds off-chain signatures to on-chain data, preventing tampering or replay. | Used for non-repudiation signatures for release/refund requests. |
| Meta-Tx | Meta Transaction | A mechanism where a relayer submits transactions and pays Gas fees on your behalf. | Enables the Settlement Agent to trigger on-chain releases without holding tokens. |
| MPC/KMS/HSM | Multi-Party Computation / Key Management Service / Hardware Security Module | Common enterprise-grade secure signature solutions. | Helps Payer/Payee manage private keys securely, avoiding centralized risks. |
| KYX | Know Your X (Customer/Seller/ Agent) | A unified compliance framework covering KYC/KYS/KYA, for identity verification and risk assessment of customers, merchants, and agents. | Ensures all participants are trustworthy and their permissions are controlled, meeting multi-role regulatory and audit requirements. |
| KYS | Know Your Seller / Supplier | Reviewing the qualifications and compliance risks of upstream merchants, | Assesses the impact of external executors (e.g., liquidity/custody partners) on the fund flow. |

| | | | |
|---|---|---|---|
| | | suppliers, or service providers. | |
| KYA | Know Your Agent | Auditing whether an AI Agent has legitimate authorization and remains compliant. | Manages entities triggering payments in the multi-agent network, preventing unauthorized access or hijacking risks. |
| Travel Rule | FATF Travel Rule | Requires Virtual Asset Service Providers (VASPs) to share sender and receiver identity information compliantly with qualifying transfers. | The compliance control plane interacts with KYX/Policy, supporting cross-border anti-money laundering and regulatory reporting. |
| Policy Engine | Policy Control Service | Aggregates risk control rules, quotas, blacklists, and KYC/KYS/KYA results. | Enables the Settlement Agent to decide whether to approve a payment during the intent stage. |
| termsHash | Terms Hash | A unique identifier generated by encrypting and hashing key parameters of the payment terms. | Ensures term integrity and prevents tampering; serves as the term-binding credential for escrow transactions. |
| | | | |