

HeyElsa Korean Lite Paper- ver 1

HeyElsa — 크립토 에이전트 레이어

(Litepaper | 한국어 번역본)

1. 개요 (Overview)

TL;DR

HeyElsa(이하 "Elsa")는 사용자의 의도(Intent)를 실제 온체인 실행(Action)으로 전환하는 크립토 에이전트 레이어입니다.

사용자 및 파트너는 목표를 제시하고, Elsa의 다중 에이전트 시스템은 이를 계획·검증·실행하여 안전하고 자율적이며 확장 가능한 방식으로 멀티체인 환경에서 실행합니다.

2. Elsa란 무엇인가?

Elsa는 DeFi를 메시지를 입력하듯 사용할 수 있게 만드는 실행 레이어입니다.

Elsa는 하나의 엔진 위에 다음 세 가지 실행 인터페이스를 제공합니다.

(1) Copilot (B2C)

- 자연어 기반 채팅을 통해
스왑, 브리지, 스테이킹, 헤징, 파밍, 자동화 설정 등을 실행

(2) Widget & SDK (B2B)

- 모든 지갑, dApp, 콘텐츠 앱에
"AI로 거래하기" 기능을 임베드할 수 있는 위젯 및 SDK 제공

(3) AgentOS

- Agent-to-Agent(A2A) 조정 버스 위에서
특화된 에이전트를 구축·호스팅·운영할 수 있는 에이전트 운영체제
-

3. 본 라이트페이퍼의 범위

본 문서는 다음 내용을 다룹니다.

- **비전:**
온디맨드 실행을 넘어,
자율적으로 수익 실현·헤징·리밸런싱·APY 이동을 수행하는 Autonomous DeFi
 - **멀티 LLM 오케스트레이션:**
작업의 복잡도·지연시간에 따라 적절한 모델로 라우팅하며,
검증 가능하고 근거 기반의 결과를 생성
 - **A2A 커뮤니케이션 레이어:**
전문 에이전트들이 작업을 조율하고, 입찰·협업하며, 컨텍스트를 공유
 - **실행 및 안전성:**
시뮬레이션, 가드레일, MEV 인지 라우팅, 영수증 및 증명 제공
 - **Cognitive Cache:**
정보 유출 없이 개인화를 가능하게 하는 **프라이버시 보존 메모리**
 - **Elsa 위에서 구축하기:**
AgentOS, SDK, 임베디드 위젯을 통해
에이전트를 호스팅하고 수익화 가능
-

4. 기술 아키텍처 개요 (레이어 구조)

Elsa는 하단에서 상단으로 계층화된 아키텍처를 채택합니다.

4.1 데이터 및 근거 레이어 (Data & Grounding)

- 온체인 상태 (노드/인덱서)
 - 가격·유동성 오라클
 - 검증된 지식 베이스
 - 데이터 최신성 검증 및 이상 탐지
-

4.2 실행 레이어 (Execution Layer)

- 사전 감사된 실행 스크립트
 - 트랜잭션 시뮬레이션
 - DEX·브리지 메타 라우터 기반 경로 선택
 - MEV 안전 제출
 - 영수증 및 텔레메트리 기록
-

4.3 에이전트 레이어 (Agent Layer)

- 조합 가능한 전문 에이전트:
 - 스왑
 - 브리지
 - 수익(Yield)
 - 리스크/헤징
 - 파생상품(Perps)
 - NFT
 - 스나이핑
 - 알림
 - 1st party 에이전트 및
권한·SLA가 제한된 외부 에이전트 호스팅 가능
-

4.4 A2A 버스 (Agent-to-Agent Coordination)

- Pub/Sub 메시징
 - 계약 기반 입찰 (Contract-Net)
 - 계획 집계
 - 평판 및 공정 스케줄링
-

4.5 오케스트레이션 레이어 (멀티 LLM 플래너)

- 작업 분류
 - 모델 라우팅 (소형 / 중형 / 대형)
 - 전략 합성
 - 제약 조건 해결
 - 폴백 및 헤지 추론
-

4.6 의도 레이어 (Intent Layer)

- 자연어 인터페이스
 - 의도 파싱
 - 제약 조건 수집
 - 다중 턴 컨텍스트 처리
-

4.7 안전 및 정책 레이어 (Safety & Policy)

- 시뮬레이션 게이트
 - 허용/차단 리스트
 - 최대 거래 금액 제한
 - 지역·KYC 연동
 - 검증된 추론 (zkTLS / MPC-TLS)
-

4.8 Cognitive Cache

- TEE 기반 개인 메모리
 - 사용자 선호·행동·결과 저장
 - 계획 속도 및 개인화 향상
-

4.9 관측성 (Observability)

- 트레이스
- 메트릭

- 감사 로그
- 이상 탐지
- 서킷 브레이커

5. 의도 레이어 (Intent Layer)

역할

의도 레이어는 사용자의 자연어 입력을 ****기계가 해석 가능한 구조화된 의도(Intent)****로 변환하고, 관련 제약 조건을 캡처하는 인터페이스입니다.

주요 책임

- 자연어(NL) 입력(텍스트/음성)을 구조화된
`Intent { verb, objects[], params{}, constraints{}, riskHint }` 형태로 파싱
- 금액, 자산, 체인, 기간, 목표 등 **슬롯 채우기(Slot Filling)** 및 불확실성 점수 산출
- **제약 조건 수집:**
 - 리스크 한도(최대 슬리피지, LTV 상한)
 - 컴플라이언스(지역/ KYC)
 - 예산, 마감 시한
- **Cognitive Cache**로부터 컨텍스트 주입
 (선호도, 과거 승인 내역, 선호 체인/폴 등)

입력 → 출력

- **입력:** 자연어 프롬프트, 지갑 컨텍스트, 세션 정책, 캐시 프로필
- **출력:** 검증된 Intent JSON(신뢰도 포함) 및 필요 시 누락 슬롯 질의

핵심 구성요소

- **NER(개체 인식)** 및 타입화된 엔터티 레지스트리(토큰, 체인, 프로토콜)
- 정규 스키마 및 버저닝
 (예: `v3.intent.swap` , `v2.intent.yield.optimize`)

- **가드레일:** 정규식, 허용 리스트, 위험 동작 거부 규칙

실패 모드 및 처리

- 신뢰도 낮음 → **명확화 질문** 요청
- 자산/체인 모호성 → **상위 3개 후보 제시**
- 정책 위반 동작 → **사유를 제시한 안전한 거절**

6. 오케스트레이션 레이어 (Orchestration Layer)

역할

의도를 **어떻게 충족할지** 결정합니다. 적절한 모델로 라우팅하고, 실행 계획을 합성하며, 에이전트를 배정하고 A2A 메시징을 설정합니다.

주요 책임

- **작업 분류:** 복잡도, 도메인, 요구 지연시간, 리스크 등급
- **멀티 LLM 라우팅:** 소형/중형/대형(또는 톨포머) 모델을 비용·지연 예산 내 선택
- **계획 합성:** 단계 분해, 사전/사후 조건 및 성공 지표 정의
- **에이전트 배정:** 역량 매칭 및 협업을 위한 A2A 계약 생성

입력 → 출력

- **입력:** Intent JSON, 캐시 프로필, 실시간 데이터 포인터(오라클/인덱서)
- **출력:**

`Plan { steps[], dependencies[], SLAs, fallbackPaths[], evalFns[] }` + 에이전트 로스터

핵심 구성요소

- **분류 특징:**
 - 작업 복잡도
 - 리스크 등급(R0~R3)
 - 지연시간 예산

- 체인/프로토콜 접근성
- 지갑 승인 필요 여부
- MEV 노출도
- 상태 유지 필요성
- 라우팅 정책: 아래 섹션 참조
- 플래너(STRIPS 유사) + 제약 해결기
(수수료, 가스, 유동성, 브리지 리스크)

실패 모드 및 처리

- 실행 가능한 계획 없음 → 대안 제시
(저렴한 체인, 다른 풀)
- SLA 위반 예상 → 모델 다운그레이드, 계획 단순화, 또는 지연 실행

7. A2A 버스 (Agent-to-Agent 커뮤니케이션)

역할

다중 에이전트 작업을 위한 **계약 네트워크(Contract-Net)** 방식의 조정 레이어로, Pub/Sub 메시징, 스케줄링, 평판 관리를 제공합니다.

주요 책임

- **매칭:** 하위 작업을 브로드캐스트하고 에이전트가 가격/SLA로 입찰
- **평판 및 스케줄링:** 성공률·지연·정확도 추적, 신뢰 에이전트 우선
- **메시지 의미론:** 멍등 명령, 재시도, 정확히-한-번(Exactly-once) 영수증

핵심 구성요소

- **토픽 예시:**

`swap.quote` , `bridge.route` , `stake.position` , `risk.hedge` , `perps.open` , `alerts.set`

- **메시지 타입:**

`rfq` , `proposal` , `award` , `work.start` , `work.done` , `work.fail`

- **보안:**

작업별 스코프, 최소 권한 자격증명, 시간 제한 임대(lease)

8. 에이전트 레이어 (Agent Layer)

역할

계획의 각 단계를 수행하는 **조합 가능한 전문 에이전트** 집합입니다.

핵심 에이전트

- **스왑 에이전트:** DEX 견적, 분할 라우팅, 슬리피지 제어
- **브리지 에이전트:** 경로 선택(리스크/수수료/시간), 증명 검증, 정체 거래 복구
- **수익(Yield) 에이전트:** APY 스캔, IL/리스크 모델링, 자동 복리, 볼트 이동
- **리스크/헤지 에이전트:** 파생상품/옵션, 손절·이익실현, 델타 헤징
- **Perps/NFT/스나이퍼/알림:**
거래소별 실행, 바닥가 스윙, 지정/취소/IOC, 이벤트 감시

인터페이스

- **역량 매니페스트:**
지원 체인·토큰, 최대 거래 한도, 기대 지연시간
- **결정적 시뮬레이션** **혹:** 사전 거래 검증
- **텔레메트리 계약:** 단계 결과, 가스 사용량, PnL, 신뢰도, 이상 징후

9. 실행 레이어 (Execution Layer)

역할

승인된 계획을 **안전하게 온체인 트랜잭션으로 전환**합니다.

주요 책임

- **프로토콜별 사전 감사된 실행 스크립트** 사용(파라미터 범위 제한)
- **트랜잭션 시뮬레이션**(포크 또는 RPC 시뮬) 및 실패 사유 캡처

- **메타 라우팅:**

DEX(0x/CoW/UniswapX/1inch)·브리지 집계자 선택 및 배치 실행

- **MEV 안전성:**

프라이빗 릴레이, 백런 방지, 비차익 번들 규칙

- **정산 및 영수증 수집, 논스/가스 조정 재시도**

출력

- `ExecutionReport { txids[], status, receipts[], realizedSlippage, fees, proofs }`

10. Cognitive Cache (인지 캐시)

역할

속도와 개인화를 위한 **프라이버시 보존형 개인 메모리**입니다.

저장 내용

- 선호도(리스크 성향, 수수료 민감도, 선호 체인/풀)
- 행동 특성(활동 시간대, 자산 선호)
- 허용/승인 내역 및 신뢰 프로토콜
- 지갑 요약 이력(포지션, 원가, PnL 구간)

제어

- TTL 및 감쇠(Decay)
- 사용자 내보내기/삭제
- 스코프별 동의
- 저장 데이터 암호화

효과

- 명확화 질문 감소
- 더 나은 기본값

- 라우팅 속도 향상
-

11. 안전 레이어 (Safety Layer)

역할

무엇을 알고 있는지 증명하고, 무엇을 허용할지 강제합니다.

근거(Grounding)

- 실시간 온체인 상태(인덱서/풀 노드)
- 오라클(가격·유동성)
- 검증된 지식 베이스(감사, 리스크 공지)
- 모든 추천/검증은 `GroundingEngine.validate()` 를 통과

정책 엔진 (Policy Engine)

- 지역/ KYC
- 프로토콜 허용 리스트
- 거래 금액·리스크 상한
- 레버리지 상한
- 카운터파티 필터

검증된 추론 (Verified Inference)

- **zkTLS / MPC-TLS** 증명 어댑터를 통해
"해당 인사이트가 특정 시점 T에 특정 소스로부터 도출되었음"을
원시 로그 노출 없이 증명

시뮬레이션 & 드라이런

- 사전 실행 시뮬레이션
 - 가드 조건 및 중단 규칙(Stop-if) 적용
-

12. 관측성 (Observability)

역할

의도 → 실행 전 과정을 추적 가능하고 운영 가능하게 만듭니다.

추적 항목

- 의도 파싱 → 계획 → A2A → 실행 → 영수증까지의 분산 트레이스
- 단계별 SLA/SLO(p50/p95 지연, 성공률, 비용)
- 정책 결정 및 에스컬레이션에 대한 감사 로그
- 이상 탐지(브리지 정체, 비정상 슬리피지, 오라클 드리프트)

제공 방식

- 운영자 대시보드
 - 파트너 웹훅
 - 사용자 가시 영수증
-

13. 분류기 특징 (확장)

- 작업 복잡도: 단일 단계 vs 다단계, 크로스체인, 승인 필요 여부
 - 리스크 등급(R0~R3):
읽기 전용 → 저위험 스왑 → 레버리지/파생 → 신규·고위험 프로토콜
 - 지연 예산:
서브초(견적) / 수분(브리지) / 지연 실행(리밸런싱)
 - 체인 접근성: 승인, 허용량, 지원 거래소/브리지
 - 시장 상황: 변동성, 유동성 깊이, MEV 리스크 구간
 - 사용자 프로필: 수수료 민감도, 리스크 허용도, 선호 거래소
-

14. 라우팅 정책 (멀티 LLM)

- 고속/트랜잭션성 작업

(잔액, 가격, 파라미터 추출) → **소형 파인튜닝 모델**

- **분석적 작업**

(수익 비교, 경로 탐색, 리스크 점수) → **중형 모델 + RAG**

- **전략/자율 작업**

(다단계 계획, 포트폴리오 관리, 헤징) → **대형 모델 + 계획·제약 해결**

비용/지연 가버너: 요청별 예산 설정, 부하 시 점진적 성능 저하

15. 근거 파이프라인 (Grounding Pipeline)

- **수집:** 온체인(노드/인덱서), 오라클(가격/TVL), 검증 KB
 - **검증:** 교차 소스 일관성, 최신성 임계치, 정족수 규칙
 - **부착:** 모든 추천/행동에 groundingRef 포함
 - **증명:** 선택적 zkTLS/MPC-TLS 증명 산출
-

16. 검증된 추론 (Verified Inference) — 상세

- **무엇:** 모델 응답이 명시된 입력/로그로부터 도출되었음을 암호학적으로 증명
 - **방법:** zk/MPC 기반 TLS 프록시; 다이제스트 저장; 검증 가능한 영수증 발행
 - **적용처:** 고위험 의사결정, 파트너 컴플라이언스, 분쟁 해결
-

17. 폴백 및 헤징

- **모델 타임아웃:** 더 작은 모델로 재라우팅, 요약 제공, 신뢰도 임계치 미달 시 사용자 확인
- **실행 헤징:** 보조 경로/거래소, 주문 분할, 변동성 구간 시간 분할
- **이중 실행:** 대형 모델로 계획 수립 후 중형 모델로 교차 검증

18. 의도 → 실행 예시 (간결)

사용자:

“ETH 2개를 솔라나로 브리지하고, 가장 안전한 수익에 스테이킹해줘.”

처리 흐름:

1. 의도 레이어:

`intent.yield.optimize`

- amount = 2 ETH
- dstChain = Solana
- risk = low

2. 오케스트레이션:

리스크 등급 R2로 분류 → 중/대형 모델 라우팅

- 계획: (스왑 없음) Wormhole 브리지 → Solana Pool A 스테이킹
- SLA 설정

3. A2A 버스:

브리지 에이전트·수익 에이전트에 RFQ 발송 → 최적 제안 선정

4. 실행:

브리지 + 스테이킹 시뮬레이션 → 프라이빗 제출 → 영수증 수집

5. 안전:

금리/APY 근거 검증 → 정책 상한 적용 → 증명 첨부

6. 관측성:

트레이스 기록 → 사용자에게 영수증 및 실현 APY 기준 제공

7. 인지 캐시:

선호 업데이트("Solana 허용", "저위험 수익")

19. 스키마 (요약)

Intent (v3)

```
{
  "verb": "yield.optimize",
  "objects": [{"asset": "ETH", "amount": "2"}],
  "params": {"dstChain": "solana", "risk": "low"},
  "constraints": {
```

```
"maxSlippageBps":50,  
"deadlineSec":900,  
"profileRef":"cache://user/123",  
"policyScope":"retail.low",  
"version":"v3"  
}  
}
```

Plan (v2)

```
{  
  "steps":[  
    {"id":"s1","type":"bridge","route":"wormhole","from":"ethereum","to":"solana"},  
    {"id":"s2","type":"stake","venue":"solend","asset":"SOL","policy":"lowRisk"}  
  ],  
  "dependencies":[["s1","s2"]],  
  "sla":{  
    "p95LatencyMs":30000,  
    "fallbacks":[{"on":"bridgeFail","route":"portal"}]  
  }  
}
```

20. SLA / SLO (권장 기준)

- 의도 파싱 p95: $\leq 400\text{ms}$
- 계획 합성 p95: 분석형 $\leq 1.5\text{초}$, 전략형 $\leq 4\text{초}$
- 견적 최신성: $\leq 3\text{초}$
- 시뮬레이션 성공률: $\geq 99\%$
- 실행 성공률: $\geq 98\%$ (재시도 포함)

- **최대 실현 슬리피지:** 견적 + 정책 버퍼(예: +15bps) 이내
 - **A2A 태스크 할당 p95:** $\leq 800\text{ms}$
 - **에이전트 실패율:** 1,000건당 $\leq 1\%$
-

21. SDK 및 AgentOS

SDK (TypeScript / Python)

- 의도 스키마
- 계획 API
- 시뮬레이션
- 트랜잭션 제출
- 영수증 조회
- 정책 템플릿
- 샌드박스 스코프 및 쿼터 관리
- 테마/톤 오버라이드

```
import {Elsa }from"@elsa/sdk";

const elsa =newElsa({apiKey: process.env.ELSA_KEY });

const intent = {
  type:"yield.optimize",
  amount:"500",
  asset:"USDC",
  constraints: {chain:"base",maxSlippageBps:30 }
};

const plan =await elsa.plan(intent);
await elsa.simulate(plan.id);
```

```
const receipt = await elsa.execute(plan.id);
```

22. AgentOS 수명주기

- 등록 → 역량 광고 → 태스크 수신
- 계획 조각/견적/트랜잭션 페이로드 생성
- 실행 → 결과 보고

샌드박스

- 에이전트별 런타임
- 리소스 쿼터(CPU/메모리)
- 스코프 기반 최소 권한
- 시크릿 격리

스케줄링

- 역량 매칭
- 실시간 부하
- 평판 기반 우선순위

23. A2A 메시징 및 마켓

- 메시징:

Pub/Sub 토픽(`quote` , `plan` , `risk` , `exec` , `status`)

서명된 엔벨로프 사용

- 컨트랙트 넷:

CFP 브로드캐스트 → 에이전트 입찰 → 코디네이터 선정

- 합의:

평판 × 최신성 × 스코프 적합도 가중치 투표

- **먹등성:**

(intent_id, step_id) 기준 재실행 안전

24. 보안·MEV·계정·지갑

- **경로 탐색:**

0x / CoWSwap / UniswapX / 1inch 메타 라우팅, 브리지 셀렉터, 주문 분할

- **시뮬레이션:**

풀 상태 기반 드라이런, 최소 수량/가스 상한/마감 검증

- **MEV 보호:**

프라이빗 릴레이/번들, 가격 영향 상한, 백런 탐지

- **계정:**

AA/ERC-4337, 세션 키, MPC/HW 지갑 지원

- **리스크:**

동적 슬리피지, 조건부 스탑, 청산 버퍼

- **영수증:**

트랜잭션 해시, 로그, 실행 슬리피지, 수수료, 경로, 견적 대비 체결

- **재현 안전:**

(user_id, intent_id, step_id) 기반 재실행 보호

25. 데이터 및 프라이버시 (Data & Privacy)

데이터 구성

- **콘텐츠:** 프롬프트 이력, 지갑 행동(체인·프로토콜·리스크), 전략 결과
- **저장:** 벡터 DB(임베딩) + 키/값 특성 저장소
- **활용:** 모호성 해소, 기본값 추론, 제약 조건 도출, 소수 샷(few-shot) 예시
- **프라이버시:**
 - TEE 기반 인지 캐시

- 사용자별 스코프 분리
- 연구 목적 공유는 옵트인
- 집계 데이터에 차등 프라이버시 적용

26. 성능 지표 (Performance Metrics)

- 라우터 지연(p95): 저/중 난이도 $\leq 300\text{ms}$, 고난이도 $\leq 900\text{ms}$
- 계획→시뮬레이션(p95): 단일 체인 $\leq 1.2\text{초}$, 크로스체인 $\leq 2.5\text{초}$
- 견적 정확도: 정책 범위 내 **99.0%**
- 실행 성공률: 재시도 포함 **99.2%**
- 처리량: 초당 **200–500** 의도(수평 확장 가능)

27. 플라이휠 (Flywheel)

더 많은 사용자 → 더 풍부한 프롬프트·결과 데이터 →

더 나은 파인튜닝·정책 업데이트 →

더 높은 계획/실행 품질 →

더 많은 파트너 전환 →

더 많은 사용자

28. 위젯(드롭인) & 임베드

위젯

```
<iframe src="https://app.heyelsa.ai/widget?partner=XYZ&theme=dark" />
```

- 컨텍스트 프리필(자산/체인/플로우)
- 콜백: `onQuote`, `onTx`, `onError`, `onComplete`

SDK

- 의도 스키마, 계획 API, 시뮬레이션, 실행, 영수증
- 정책 템플릿(거래 한도, 허용 라우터, 지역 제한)
- 테마/톤 커스터마이징

29. 비즈니스 모델 (Business Model)

Elsa는 투기 중심이 아닌 사용량 중심 모델입니다.

수익원

- 실행 트랜잭션 수수료
- AI 에이전트 커미션
- B2B SDK 및 위젯 라이선스
- 추론(Inference) 및 실행 수수료

실행 볼륨이 증가할수록 수익은 자연스럽게 확장됩니다.

30. 토큰 개요 (Token Overview)

항목	값
토큰	\$ELSA
총 공급량	1,000,000,000
소수점	18
네트워크	Base

\$ELSA는 실행, 추론, 접근 권한을 구동하는 유틸리티 토큰입니다.

31. \$ELSA의 유틸리티 (Utility of \$ELSA)

- 수수료 할인:
보유자 및 스테이커 대상 실행 수수료 인하
- 기능 접근:
프리미엄 에이전트(카피 트레이딩, 스나이핑, 인지 캐시) 해금

- **가스 추상화:**

\$ELSA를 사용한 가스 프리 트랜잭션

- **추론 & 실행:**

AI 모델 추론 비용 및 온체인 에이전트 실행

- **B2B 사용:**

SDK 및 위젯 사용

유틸리티는 플랫폼의 실제 사용량과 함께 확장됩니다.

32. 토큰 배분 및 베스팅 (Token Allocation & Vesting)

구분	배분 비율	베스팅 조건
팀 (Team)	7%	12개월 클리프 후 24개월 선형
재단 (Foundation)	34.490%	TGE 20% 언락, 6개월 클리프 후 48개월 선형
커뮤니티 (Community)	40%	TGE 20% 언락, 48개월 선형
프리시드 (Pre-Seed)	1.4%	12개월 클리프 후 24개월 선형
시드 (Seed)	9.11%	12개월 클리프 후 24개월 선형
유동성 (Liquidity)	8%	TGE 시 100%

커뮤니티 중심의 배분과 ****장기적 정렬(alignment)****을 통해 지속 가능한 생태계를 지향합니다.

33. 로드맵 스냅샷 (Roadmap Snapshot)

2025년 2분기까지 (완료)

- 퍼블릭 베타 출시
- 핵심 DeFi 코파일럿 공개
- 일일 거래량 **\$1M+**
- **MAU 30만+**, 가입자 **70만+**
- 수익화 개시

2025년 3-4분기

- Hyperliquid 기반 영구선물(Perps) 지원
- 카피 트레이딩 및 스나이핑 에이전트
- 수익 볼트 및 자동화
- 모바일 앱(iOS/Android)
- 멀티 에이전트 오케스트레이션 고도화
- \$ELSA 토큰 생성 이벤트(TGE) 및 유틸리티 활성화

2026년

- 완전 자율 포트폴리오 관리
- 토큰화된 크립토 지수
- 에이전트 빌더 SDK
- 자동화 전략 마켓플레이스
- 크로스체인 실행 키트
- 지갑·앱 전반의 B2B 확장

34. 디자인 철학 (Design Philosophy)

- 실행이 투기보다 우선 (Execution > Speculation)
- 거버넌스보다 유틸리티 우선
- 수요 기반 토큰 이코노미
- 조언이 아닌 '행동하는' AI

35. 문제 정의 (Problem)

크립토는 강력하지만, 대규모 사용에는 부적합합니다.

- 너무 많은 체인·도구·워크플로우
- 기본 작업에도 높은 인지 부담

- 수동 실행으로 인한 비효율적 결과
- 자동화는 존재하지만 접근성 부족

대부분의 사용자는 더 많은 프로토콜이 아니라 더 나은 실행을 필요로 합니다.

36. 해결책 (Solution)

Elsa는 **AI 중심 실행 레이어**를 도입합니다.

- 사용자는 **의도를 표현**
- Elsa는 **이를 실행**
- 자연어 → 온체인 행동
- 체인·프로토콜 전반의 **최적화된 라우팅**
- AI 에이전트가 거래·수익·리스크 관리
- **통제력을 유지한 자동화**
- 투명성을 보존하면서 복잡성을 추상화

37. 제품 스택 (Product Stack)

B2C: Elsa Copilot

- 스왑, 브리지, 스테이킹, 차입
- 영구선물 거래 및 카피 트레이딩
- 수익 최적화 및 포트폴리오 자동화
- 스마트 알림 및 에이전트 기반 실행

B2B: Elsa 인프라

- 위젯 및 iframe 통합
- 앱·지갑용 SDK
- 에이전트 실행 API
- 수익 공유형 유통 모델

38. 비즈니스 모델 요약 (Business Model — Summary)

Elsa의 비즈니스 모델은 **투기 중심이 아닌, 실행과 사용량 중심**으로 설계되었습니다.

플랫폼 상에서 발생하는 실제 활동과 실행 볼륨이 증가할수록 수익이 자연스럽게 확장됩니다.

주요 수익원

- 온체인 실행 트랜잭션 수수료
- AI 에이전트 커미션
- B2B SDK 및 위젯 라이선스
- 모델 추론(Inference) 및 실행 비용

이 모델은 **지속 가능성**과 **실사용 기반 성장**을 핵심으로 합니다.

39. \$ELSA 토큰 유틸리티 (Utility — Consolidated View)

\$ELSA는 Elsa 생태계의 **경제적 중추**로서, 다음과 같은 실질적 유틸리티를 제공합니다.

- **수수료 할인**
\$ELSA 보유자 및 스테이커에게 실행 수수료 인하 제공
- **기능 접근 권한**
프리미엄 에이전트(카피 트레이딩, 스나이핑, 고급 인지 캐시 등) 이용 가능
- **가스 추상화**
Elsa 내에서 \$ELSA를 활용한 가스 프리 트랜잭션
- **AI 추론 및 실행 비용**
AI 모델 추론과 온체인 에이전트 실행에 사용
- **B2B 인프라 사용**
SDK 및 위젯 기반 파트너 통합 비용 결제

| \$ELSA의 유틸리티는 플랫폼의 실제 사용량 증가와 함께 비선형적으로 확장됩니다.

40. 장기 비전 (Long-Term Vision)

Elsa는 단순한 인터페이스나 분석 도구가 아니라,

의도를 이해하고 실행까지 책임지는 자율형 크립토 에이전트 레이어를 지향합니다.

- 사용자는 목표를 제시하고
- Elsa는 이를 계획·검증·실행하며
- 안전성과 투명성을 유지한 상태로 자동화를 제공합니다.

궁극적으로 Elsa는:

- 자가 주행 포트폴리오(Self-driving portfolios)
- 에이전트 기반 금융 인프라
- 지갑·앱·프로토콜 전반에 내장되는 실행 표준

으로 진화하는 것을 목표로 합니다.

41. 결론 (Conclusion)

암호화폐의 문제는 기능의 부족이 아니라 **실행의 복잡성**입니다.

Elsa는 이 문제를 **AI 기반 실행 레이어**로 해결합니다.

- 자연어 기반 의도 입력
- 멀티 에이전트 협업
- 검증 가능한 추론
- 안전한 온체인 실행
- 실제 사용에 기반한 토큰 이코노미

Elsa는 조언하는 AI가 아니라, 실행하는 AI입니다.

마무리 문구

Elsa는 **확장성, 자동화, 실사용**을 위해 설계되었습니다.

크립토의 다음 단계는 더 많은 프로토콜이 아니라, **더 나은 실행**입니다.