

온톨로지 가스(Ontology Gas)

해당 번역본은 이용자가 디지털자산 관련 정보를 편리하게 확인할 수 있도록 단순 참고용으로 번역된 것이며, 투자 권유를 목적으로 하지 않습니다. 그 내용이 정확하지 않을 수 있으므로 정확한 정보 습득을 위해서는 원문을 참고하시거나 원문 작성 주체 측에 문의하시기 바랍니다. 두나무는 해당 번역본 내용의 진실성 및 정확성을 보장하지 않고, 제공된 정보에 의한 투자 결과에 대하여 어떠한 책임도 지지 않으며, 투자 손실은 투자자에게 귀속됩니다.

소개 (Introduction)

온톨로지(Ontology)는 차세대 퍼블릭 블록체인 네트워크이자, 원장 계정과 스마트 계약을 통합 지원하는 분산 신뢰 기반 협업 플랫폼입니다.

온톨로지는 자체 퍼블릭 체인에서 실행되는 서비스와 애플리케이션을 지원할 뿐만 아니라, 온톨로지 프레임워크를 기반으로 개발된 애플리케이션의 퍼블릭 체인과 다양한 프로토콜 그룹과의 협업도 지원합니다.

보다 근본적인 수준에서 온톨로지는 **분산 신원 프레임워크(DID)**, **분산 데이터 교환 프레임워크(DDXF)** 및 기타 분산 신뢰 협업 패키지와 모듈과 같은 폭넓은 범용 모듈을 계속해서 제공합니다.

이 **문서 센터**는 Ontology로 dApp 개발을 처음 시작하는 개발자를 위한 가이드 역할을 하며, Ontology 프레임워크의 다양한 구성 요소에 대한 정보를 제공합니다.

또한, 개발자가 애플리케이션 개발 과정에서 활용할 수 있는 다양한 리소스를 탐색하고 이해할 수 있는 장소로서 기능합니다.

문서와 관련된 문제나 부정확한 내용은 **링크**(<https://forms.gle/STBynyuEPU1VXAiv9>)를 통해 공유해 주실 수 있습니다.

온톨로지 살펴보기

우리가 하는 일과 제공하는 것을 알아보세요.

테스트넷 토큰 (TestNet Tokens)

온톨로지 테스트넷(Ontology TestNet)은 개발자들의 dApp 개발 과정을 지원하기 위해 공개적으로 제공됩니다. 이는 **메인넷(MainNet)**에 실제로 배포하기 전에 스마트 계약을 **배포하고 테스트 실행**하는데 사용할 수 있습니다.

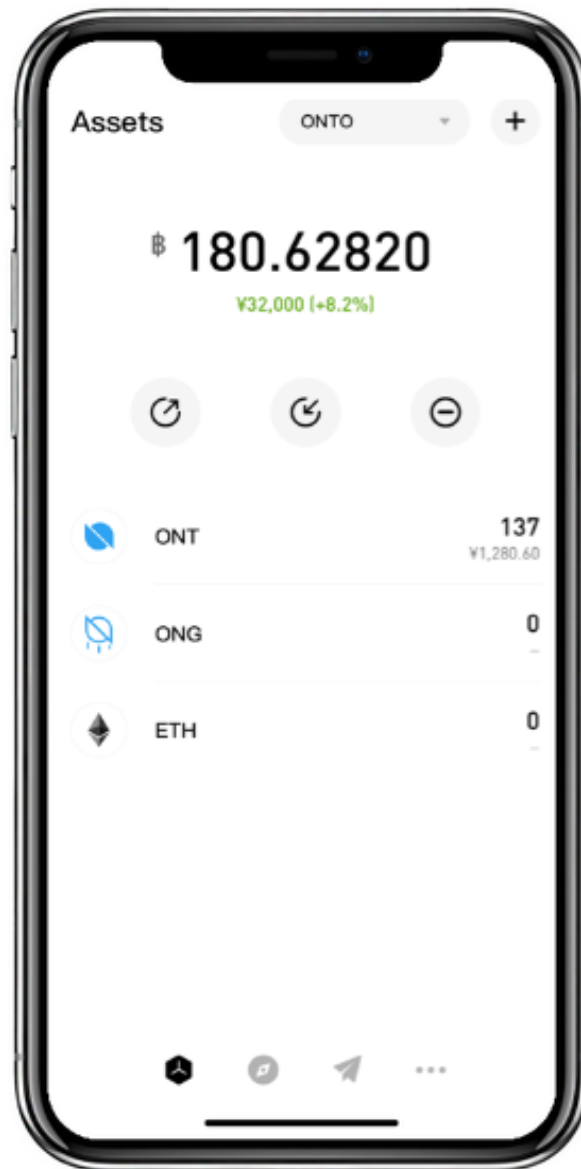
테스트넷에서도 배포 및 실행 과정에는 가스가 소모됩니다. **테스트넷 토큰**은 이 링크(<https://developer.ont.io/applyOng>)를 통해 신청할 수 있습니다.

온톨로지 트러스트 생태계 (Ontology Trust Ecosystem)



ONTO

ONTO는 온톨로지의 주요 **디지털 지갑 애플리케이션**입니다. 현재 **자산 관리** 및 **신원 관리**를 지원합니다. 사용자들은 향후 업데이트에서 **클레임 관리** 기능도 ONTO의 일부가 될 것으로 기대할 수 있습니다.



현재 출시 기능:

- 노드 스테이킹 – ONT 를 스테이킹하고 ONG 를 획득
- Candy Box 에어드롭 이벤트 에 접근
- ONT, NEO, ETH, Kakao 등 여러 플랫폼의 자산 지원
- dApp 스토어 에서 다양한 dApps 이용
- 새로운 ONT ID 등록

ONTO 다운로드는 이 링크(<https://onto.app/en/download/>)를 따라주세요.

OWallet

OWallet 은 온톨로지의 디지털 지갑 데스크톱 클라이언트입니다.



최신 릴리스의 기능:

- 멀티시그(multi signature) 기술 을 사용한 일반 지갑 관리 및 공유 지갑 지원
- 계정 관리 및 레저(ledger) 지원
- 노드 스테이크 관리
- ONT ID – 신원 관리

OWallet 다운로드 는 이 링크(<https://github.com/ontio/OWallet/releases/>) 를 따라주세요.

Explorer

온톨로지의 공개 신뢰 검증 시스템 으로, 블록체인 활동 모니터링을 지원합니다.

Explorer는 트랜잭션 해시, ONT ID, 컨트랙트 해시, 블록 높이 등의 식별자를 사용하여 트랜잭션, 엔티티, 스마트 컨트랙트, 블록 관련 데이터를 가져올 수 있습니다.



Explorer

Explorer 접속은 이 링크(<https://explorer.ont.io/>) 를 따라주세요.

노드 배포 (Deploy a Node)

온톨로지 노드에는 세 가지 분류가 있습니다:

Triones 노드 | Ontology MainNet

Triones 노드는 네트워크 합의에 참여하는 부기(bookkeeping) 노드이며, 합의 노드로 지원한 후 사용자들의 **ONT** 스테이킹 투표를 통해 선출되어야 합니다.

인센티브 및 합의 관련 정보는 [여기\(https://medium.com/ontologynetwork/triones-node-incentive-model-dbc175f4728\)](https://medium.com/ontologynetwork/triones-node-incentive-model-dbc175f4728)를 클릭하세요.

동기화 노드 | Ontology MainNet / Polaris TestNet

동기화 노드는 네트워크 합의에 참여하지 않으며, Triones(부기) 노드가 생성한 블록만 동기화합니다.

단일 노드 | Ontology Privatenet / Polaris TestNet

온톨로지는 프라이빗 테스트 환경에서 개발을 위한 단일 노드 네트워크 배포를 지원합니다.

노드 배포 가이드는 이 링크(<https://docs.ont.io/ontology-node/node-deployment>)를 따라주세요.

백서 (Whitepapers)

소개 백서 (Introductory White Paper)

- 영어(<https://ont.io/wp/Ontology-Introductory-White-Paper-EN.pdf>)
- 중국어(<https://ont.io/wp/Ontology-Introductory-White-Paper-ZH.pdf>)
- 한국어(<https://ont.io/wp/Ontology-Introductory-White-Paper-KR.pdf>) 번역: @Sunghwan Kim
- 네덜란드어(<https://github.com/ontio/documentation/blob/master/nl-NL/Introductory%20White%20Paper%20-%20Dutch%20V1.pdf>) 번역: @Mark Westerweel
- 일본어(https://github.com/ontio/documentation/blob/master/jp_JP/Ontology%20Introductory%20White%20Paper%20JP.pdf) 번역: @Miho Nakauchi
- 스페인어(<https://github.com/ontio/documentation/blob/master/es-ES/Introductory%20White%20Paper%20-%20Spanish.pdf>) 번역: @Alejandro Garcia
- 터키어(https://github.com/ontio/documentation/blob/master/tr_TR/Introductory%20White%20Paper%20-%20Turkish.pdf) 번역: @Hakan

기술 백서 (Technical White Paper)

- 영어(<https://github.com/ontio/Documentation/blob/master/Ontology-technology-white-paper-EN.pdf>)
- 중국어(<https://ont.io/wp/Ontology-technology-white-paper-ZH.pdf>)
- 한국어(<https://ont.io/wp/Ontology-technology-white-paper-KR.pdf>) 번역: @Sunghwan Kim
- 네덜란드어(<https://github.com/ontio/documentation/blob/master/nl-NL/Techno>

[logy%20White%20Paper%20-%20Dutch%20V1.pdf](https://ont.io/wp/Ontology-White-Paper-%20Dutch%20V1.pdf)) 번역: @Mark Westerweel

- 일본어(https://github.com/ontio/documentation/blob/master/jp_JP/Ontology%20Technology%20White%20Paper%20JP.pdf) 번역: @Miho Nakauchi

생태계 백서 (Ecosystem White Paper)

- 영어(<https://ont.io/wp/Ontology-Ecosystem-White-Paper-EN.pdf>)
- 중국어(<https://ont.io/wp/Ontology-Ecosystem-White-Paper-ZH.pdf>)
- 한국어(<https://ont.io/wp/Ontology-Ecosystem-White-Paper-KR.pdf>) 번역: @Sunghwan Kim
- 네덜란드어(<https://github.com/ontio/documentation/blob/master/nl-NL/Ecosystem%20White%20Paper%20-%20Dutch%20V1.pdf>) 번역: @Mark Westerweel
- 일본어(https://github.com/ontio/documentation/blob/master/jp_JP/Ontology%20Ecosystem%20White%20Paper-JP.pdf) 번역: @Miho Nakauchi

프레임워크 백서 (Framework White Paper)

- 영어(<https://ont.io/wp/Ontology-Framework-White-Paper-EN-2.0.0.pdf>)
- 중국어(<https://ont.io/wp/Ontology-Framework-White-Paper-CH-2.0.0.pdf>)

인프라 백서 (Infrastructure White Paper)

- 영어(<https://ont.io/wp/Ontology-Infrastructure-White-Paper-EN-2.0.0.pdf>)
- 중국어(<https://ont.io/wp/Ontology-Infrastructure-White-Paper-CH-2.0.0.pdf>)

트러스트 프레임워크 백서 (Trust Framework White Paper)

- 영어(<https://ont.io/wp/Ontology-Trust-Framework-White-Paper-EN-2.0.0.pdf>)
- 중국어(<https://ont.io/wp/Ontology-Trust-Framework-White-Paper-CH-2.0.0.pdf>)

트러스트 앵커 되기 (Become a Trust Anchor)

트러스트 앵커(Trust anchors)는 온톨로지의 분산 신원 네트워크의 일부이며, 트러스트 네트워크 내 다른 사용자들에게 **자격 증명 검증 서비스**를 제공합니다. 트러스트 앵커의 역할은 **개인부터 정부 기관, 조직까지** 누구나 수행할 수 있습니다.

dApp 제출하기 (Submit a dApp)

온톨로지는 **혁신**을 높이 평가하고 지원합니다. **dApp**을 개발했거나 개발 중이며 **커뮤니티 dApp 스토어**에 등록을 원한다면, 이 링크(<https://oodapp.io/>)를 따라주세요.

비즈니스 협업 (Business Collaboration)

블록체인 기반 솔루션을 비즈니스에 검토하고 있으며 온톨로지와 협업을 희망하는 기업은 [여기\(https://ont.io/contact/\)](https://ont.io/contact/)로 연락할 수 있습니다.

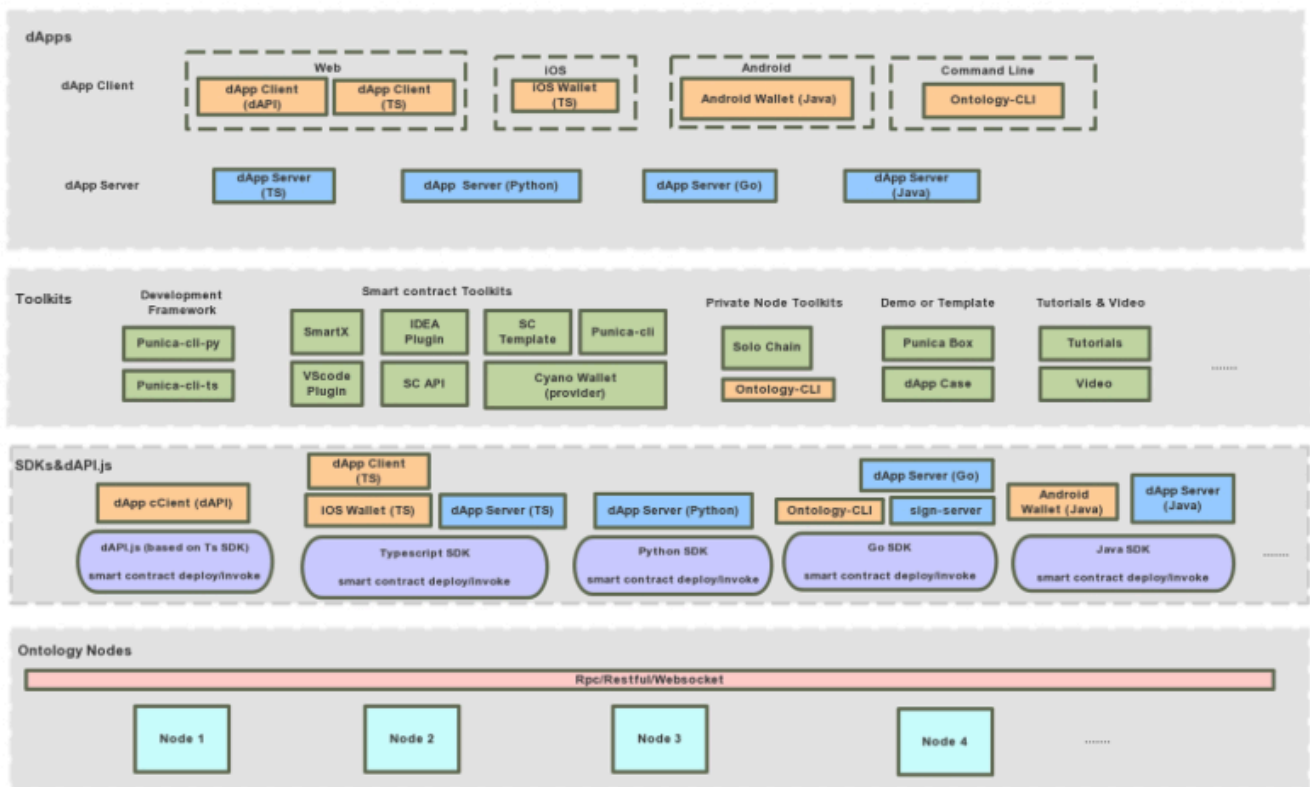
합류하기 (Join Us)

온톨로지에 관심 있는 개인은 이 링크(<https://ont.io/join>)를 통해 채용 지원을 할 수 있습니다.

시작하기 (Getting Started)

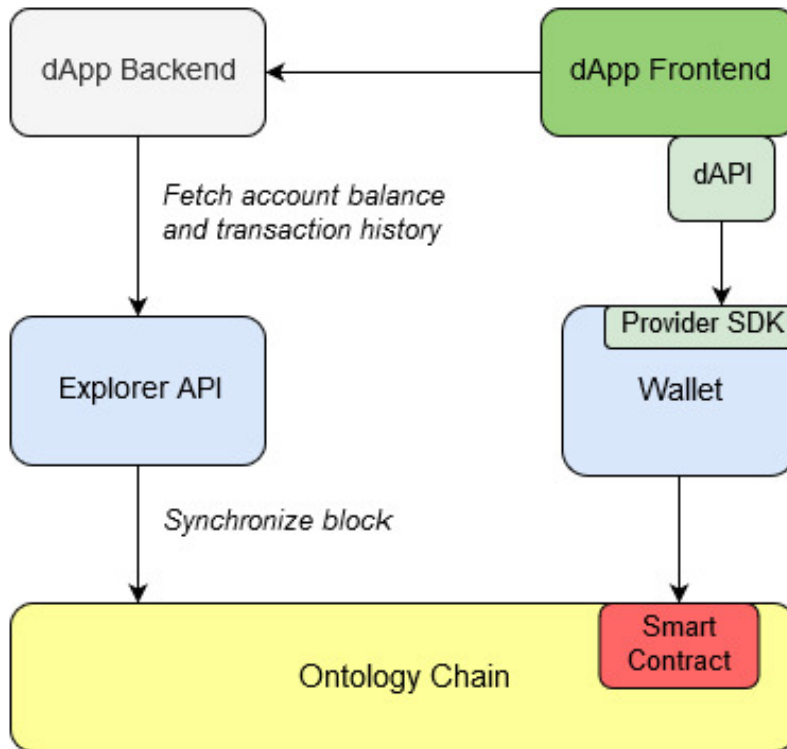
dApp 개발 과정에 대한 간략한 개요

온톨로지는 dApp 개발 과정을 염두에 두고, 개발자가 초기 학습 곡선을 극복하고 dApp 개발에 익숙해지도록 돕기 위해 원스톱 솔루션을 제공하고 있습니다.



dApp 기본 (dApp Fundamentals)

dApp 개발은 크게 두 요소로 구성된다고 이해할 수 있습니다 – 스마트 컨트랙트 개발과 dAPI 통합.



- dApp이 체인에서 데이터를 동기화하고 업데이트할 필요가 있을 때, 온톨로지의 SDK를 사용하거나 Restful 인터페이스 또는 RPC 인터페이스를 호출하여 이를 수행할 수 있습니다. 앞서 언급한 구성 요소들에 대한 세부 사항을 확인해 보세요.

*dApp 백엔드는 반드시 필요한 전제 조건이 아닙니다. 일부 dApp은 백엔드를 필요로 하지 않을 수도 있습니다.

- 특정 dApp의 경제 모델은 핵심 기능 구현을 위해 토큰을 필요로 할 수 있습니다. 그런 경우, 사용자는 필요에 따라 토큰을 발행할 수 있습니다. 스마트 컨트랙트의 OEP4 기능과 스마트 컨트랙트 샘플 코드를 참고할 가치가 있습니다.
- 일부 특수 목적의 dApp은 데이터를 체인에 저장해야 할 수도 있습니다.

일반적으로, 대부분의 dApp의 기본 로직은 스마트 컨트랙트 범위 내에서 완성됩니다. 온톨로지가 제공하는 다양한 도구들은 개발 및 배포 속도를 높여주며, 경우에 따라 다른 체인에서 온톨로지로 컨트랙트를 마이그레이션할 수도 있습니다. 또한 온톨로지는 스마트 컨트랙트와 상호작용하기 위한 표준 인터페이스(dAPI)를 제공하여, 사용자가 보안을 해치지 않고 편리하게 dApp을 사용할 수 있도록 합니다. 개인 키를 잃어버릴 위험이 없습니다. dApp 개발자는 dApp의 핵심 로직 개발에만 집중할 자유를 갖습니다.

- 개발 가이드 (<https://docs.ont.io/guides-and-tutorials/development-guides>)
- 통합 가이드 (<https://docs.ont.io/guides-and-tutorials/integration-guides>)

스마트 컨트랙트 기본 (Smart Contract Fundamentals)

스마트 컨트랙트는 블록체인 생태계에서 사용되는 근본적인 논리 단위입니다. 스마트 컨트랙트는 특정한 계약 조항이 충족되거나 충족되지 않았을 때, 비즈니스 로직과 아키텍처에 기반하여 단일 또는 다중 체인에서 자산을 발행함으로써 기능을 구현합니다. 따라서 스마트 컨트랙트는 산업에 제약 없이 복잡한 로직과 기능을 프로그래밍하는 데 사용될 수 있습니다.

온톨로지는 **SmartX** 라는 IDE를 제공하며, 이는 **Python, C#, JavaScript(곧 지원 예정)** 을 사용한 스마트 컨트랙트 개발을 지원합니다. SmartX에는 내장 컴파일러가 있어 온톨로지 **테스트넷(TestNet)**, 로컬에 배포된 **프라이빗넷(privatenet)**, 그리고 **메인넷(MainNet)** 에 스마트 컨트랙트를 배포하고 테스트할 수 있습니다. 또한 다양한 복잡성을 가진 템플릿을 포함하고 있어, 개발자와 엔지니어들이 활용할 수 있습니다.

- 스마트 컨트랙트 (<https://docs.ont.io/ontology-elements/smart-contracts>)

로컬에서 노드 배포 (Deploying a Node Locally)

온톨로지 노드는 로컬 머신에 배포하여 dApp과 스마트 컨트랙트 로직을 테스트할 수 있습니다. 온톨로지가 제공하는 **punica suite** 는 개발 프레임워크로서, 모든 필수 도구를 포함합니다.

- dApp 개발 프레임워크 (<https://docs.ont.io/developer-tools/punica-suite>)

용어집 (Glossary)

온톨로지와 관련된 주요 용어와 개념

아래는 문서 전반에서 자주 언급되는 용어와 개념의 목록입니다. 이 목록은 온톨로지 프레임워크를 처음 접하는 사용자들에게 참고 자료 또는 출발점으로 사용될 수 있습니다.

관련된 일부 용어와 개념은 이해와 접근을 용이하게 하기 위해 함께 그룹화되었습니다.

*이 용어집에 언급된 모든 용어와 개념 정의는 온톨로지의 범위 내에 한정됩니다.

블록체인 (Blockchain)

합의(consensus) 와 **신뢰(trust)** 에 기반해 작동하는 노드들의 **원장 기반(ledger-based)** 네트워크. 모든 기능과 애플리케이션은 복잡한 **알고리즘**과 **암호학(cryptography)** 을 사용해 구현된 위의 특성에 기반해 작동합니다.

노드 (Node)

노드는 노드 기반 **P2P** 네트워크의 인스턴스 또는 가장 기본적인 요소입니다. 동일한 정의는 일반적으로 **P2P** 특성을 가진 블록체인 네트워크에도 적용됩니다.

온톨로지 프레임워크 맥락에서, 노드는 온톨로지 코어의 공식 Go 기반 구현을 실제로 배포해 실행하는 인스턴스를 의미합니다.

블록 (Block)

블록은 **트랜잭션 기록**이 해당 **트랜잭션 해시**와 함께 등록되는 **원장 단위**입니다. 모든 데이터는 모든 노드에서 **열려 있고(open)**, **투명하며(transparent)**, **접근(accessed)** 및 **동기화(synchronized)** 될 수 있습니다. 블록은 체인에서의 **높이(height)**로 참조할 수 있습니다. 시간 동기화와 연결된 블록들의 집합은 본질적으로 블록체인을 형성합니다.

*온톨로지 블록체인은 약 6초마다 새로운 블록이 생성되는 빠른 블록 생성 속도를 제공합니다.

트랜잭션 (Transaction)

전통적으로, 트랜잭션은 금융 자산과 상품의 전송이나 이전으로 이해됩니다. 하지만 블록체인에서는 네트워크 또는 블록체인의 두 노드 간의 데이터 전송이나 메시지 전달 행위가 모두 트랜잭션으로 간주됩니다. 각 트랜잭션에는 특정 트랜잭션에 대한 세부 정보를 확인할 수 있는 트랜잭션 해시가 있습니다.

토큰 (Tokens)

ONT와 **ONG**는 온톨로지 체인의 네이티브 토큰입니다.

ONT

ONT는 디지털 화폐 시장에서 인기를 얻은 암호화폐 토큰입니다. 이를 지원하는 애플리케이션 내에서 일반적인 교환 목적으로 사용할 수 있습니다. **ONT**의 dApp을 탐색해 보세요. **ONT**는 다른 디지털 화폐(예: 비트코인, 이더리움)나 법정화폐(예: USD)와 교환할 수 있으며, [Binance](#), [Huobi](#)와 같이 **ONT**를 지원하는 거래소 플랫폼에서 거래할 수 있습니다.

ONT의 특징:

- 온톨로지 체인에서 스테이킹을 위한 거버넌스 토큰으로 사용할 수 있습니다.
- 분할 불가능합니다. 즉, 토큰의 최소 단위는 1로 제한됩니다.

토큰 소수점(decimal):

- 원래 **ONT**는 소수점 0자리로 설계되었습니다.
- 블록 높이 **13920000** 업그레이드 이후, 소수점 **9자리**로 업데이트되었습니다.

ONG

ONG 는 ONT 와 유사한 암호화폐입니다.

ONG의 특징:

- ONG는 온톨로지 블록체인에서 연산을 수행하기 위한 가스로 사용됩니다. 모든 온체인 동작 또는 트랜잭션은 소량의 가스 수수료를 소비합니다.
- 분할이 가능합니다. ONG의 최소 단위는 **0.000000001** (소수점 9자리 정밀도)입니다.

토큰 소수점(decimal):

- 원래 ONG는 소수점 9자리로 설계되었습니다.
- 블록 높이 **13920000** 업그레이드 이후, 소수점 **18자리** 로 업데이트되었습니다.

스마트 컨트랙트 (Smart Contracts)

스마트 컨트랙트는 블록체인 생태계 내에서 사용되는 **논리 단위**의 기본 요소입니다. 스마트 컨트랙트는 특정한 계약 조건이 충족되거나 충족되지 않았을 때, 비즈니스 **논리**와 **아키텍처**에 따라 단일 혹은 다중 체인에서 자산을 발행하여 기능을 구현합니다. 따라서 스마트 컨트랙트는 산업의 제약 없이 **복잡한 논리**와 기능을 프로그래밍하는 데 사용될 수 있습니다.

NeoVM

Neo 가상 머신(NeoVM) 은 온톨로지 스마트 컨트랙트를 컴파일할 때 생성되는 **AVM 바이트 코드** 를 처리하고 실행하는 엔진입니다. 현재 NeoVM은 Python 및 C# 으로 작성된 컨트랙트를 지원합니다. 온톨로지의 온라인 IDE SmartX(<https://smartx.ont.io>) (**Google Chrome**을 통해 접속)를 확인해 보세요.

웹어셈블리 프레임워크 - WASM (Web Assembly Framework - WASM)

NeoVM과 마찬가지로, **웹어셈블리(WASM)** 프레임워크는 이식 가능한 바이너리 코드를 처리할 수 있는 기능을 가지고 있습니다. 애플리케이션은 온톨로지 프레임워크를 사용하여 개발된 후 최적화된 바이너리 코드로 컴파일되어 웹어셈블리 엔진에서 실행될 수 있습니다. 현재 WASM 프레임워크는 **Rust** 와 **C++** 를 지원합니다.

디지털 지갑 (Digital Wallet)

사용자가 디지털 화폐를 저장하고 활용할 수 있도록 하는 지갑입니다. 온톨로지는 현재 사용 가능한 세 가지 디지털 지갑을 가지고 있습니다. **Cyano** 는 크롬 플러그인 지갑이고, **ONTO** 는 모바일 애플리케이션 형태로 제공되며, **OWallet** 은 데스크톱 지갑입니다.

dApp - 탈중앙화 애플리케이션 (Decentralized Application)

탈중앙화 애플리케이션(dApp)은 비즈니스 로직을 달성하기 위해 분산 시스템이나 네트워크의 **기능과 백엔드 메커니즘**을 구현합니다. 온톨로지는 **프레임워크**와 **SDKs**를 제공하여 **dApp 개발**을 지원합니다.

브리지 (Bridge)

온톨로지 **브리지(Bridge)**는 OEP-4와 ORC-20 토큰 간 교환과, 온톨로지 주소(a-)와 이더리움 주소(0x-) 간 ONG 전송을 지원합니다. 향후에는 **NFT 전송**도 지원할 예정입니다.

익스플로러 (Explorer)

익스플로러는 온톨로지의 신뢰 검증기(trust verifier)입니다. 블록, 트랜잭션, 컨트랙트 관련 정보는 트랜잭션 해시, 블록 높이, 컨트랙트 해시 등의 식별자를 사용하여 편리하게 조회할 수 있습니다.

OEP - 온톨로지 개선 제안 (Ontology Enhancement Proposals)

OEPs는 온톨로지 프레임워크와 네이티브 토큰 **ONT**, **ONG**를 기반으로 하는 토큰 및 토큰 프로토콜에 대한 새로운 표준을 정의합니다.

SDKs

SDK는 개발자가 쉽게 접근할 수 있도록 도구들을 패키지화한 모음입니다. 온톨로지는 주요 프로그래밍 언어 모두에서 SDK를 제공하여 **dApp 개발**을 지원합니다.

dAPI

dAPI는 온톨로지가 제공하는 **API**들을 총칭하는 용어입니다.

Restful API

- **HTTP 요청**을 보내고 개별 응답을 받아 온체인 데이터를 가져오는 데 유용합니다.

RPC API

JSON-RPC 프로토콜을 사용하여 구현된 **원격 프로시저 호출(RPC) API**입니다. **HTTP 요청/응답**을 사용해 데이터를 통신합니다.

Websocket API

웹소켓(Websocket) 프로토콜을 사용하여 구현됩니다. 노드와 체인 사이에 **양방향 연결** 을 설정하여 자유로운 데이터 흐름을 가능하게 합니다.

엔티티 (Entity)

분산 네트워크의 맥락에서 엔티티(entity)는 개인, 신뢰 기반 클레임을 검증하는 기관, 정부 기관 등 제한 없이 될 수 있습니다. 온톨로지의 **ONT ID** 와 **DDXF 프로토콜** 은 엔티티가 신원을 검증하고 신뢰와 신용에 기반한 네트워크에서 자원을 교환하는 다양한 기능을 수행할 수 있도록 합니다.

ONT ID

온톨로지 네트워크의 **엔티티** 에 할당된 **디지털 신원** 으로, **디지털 식별자** 역할을 합니다.

DDXF

DDXF 는 **Distributed Data Exchange Framework(분산 데이터 교환 프레임워크)** 의 약자로, 온톨로지가 설계하여 **토큰화(tokenization)** 와 **데이터(data)** 및 **클레임(claims)** 의 거래를 마켓플레이스를 통해 가능하게 합니다.

리소스 (Resource)

DDXF 의 맥락에서, 리소스(resource)는 마켓플레이스에서 제공되는 모든 것을 의미합니다. 리소스는 가치 있는 물품과 같은 **물리적 대상** 일 수도 있고, 창작물의 **디지털 사본** 일 수도 있으며, 클라우드 플랫폼의 **연산 능력** 이나 **스토리지** 일 수도 있습니다. 교환 시스템을 통해 이러한 리소스에 대한 **교환(exchange)** 과 **전송(transfer)** 활동을 수행할 수 있습니다.

클레임 (Claim)

클레임(Claim) 은 신뢰 네트워크 내 특정 상황에서 가치를 가지는 **검증 가능한 자격 증명(verifiable credential)** 이나 **소유권과 관련된 주장(ownership related assertion)** 입니다. 클레임은 **개인 데이터(personal data)**, **디지털 서명(digital signatures)**, **속성(attributes)**, **물리적 객체에 연결된 소유 상태(ownership status)** 등과 같은 정보를 나타내는 데 사용될 수 있습니다.

신뢰 네트워크 (Trust Network)

신뢰 네트워크(trust network)는 분산 네트워크(decentralized network)로, 검증 가능한 자격 증명(verifiable credentials)과 신뢰 메커니즘(trust mechanism)을 사용하는 주체들이 참여하여 디지털 토큰(digital tokens)이나 기타 토큰화된 자산(tokenized assets)과 관련된 트랜잭션을 수행합니다.

논스 (Nonce)

논스(Nonce)는 암호학적 통신(cryptographic communication)에서 사용되는 임의의 수(arbitrary number)입니다. 단 한 번만 사용할 수 있습니다.

ONT ID

온톨로지의 탈중앙화 신원 프레임워크

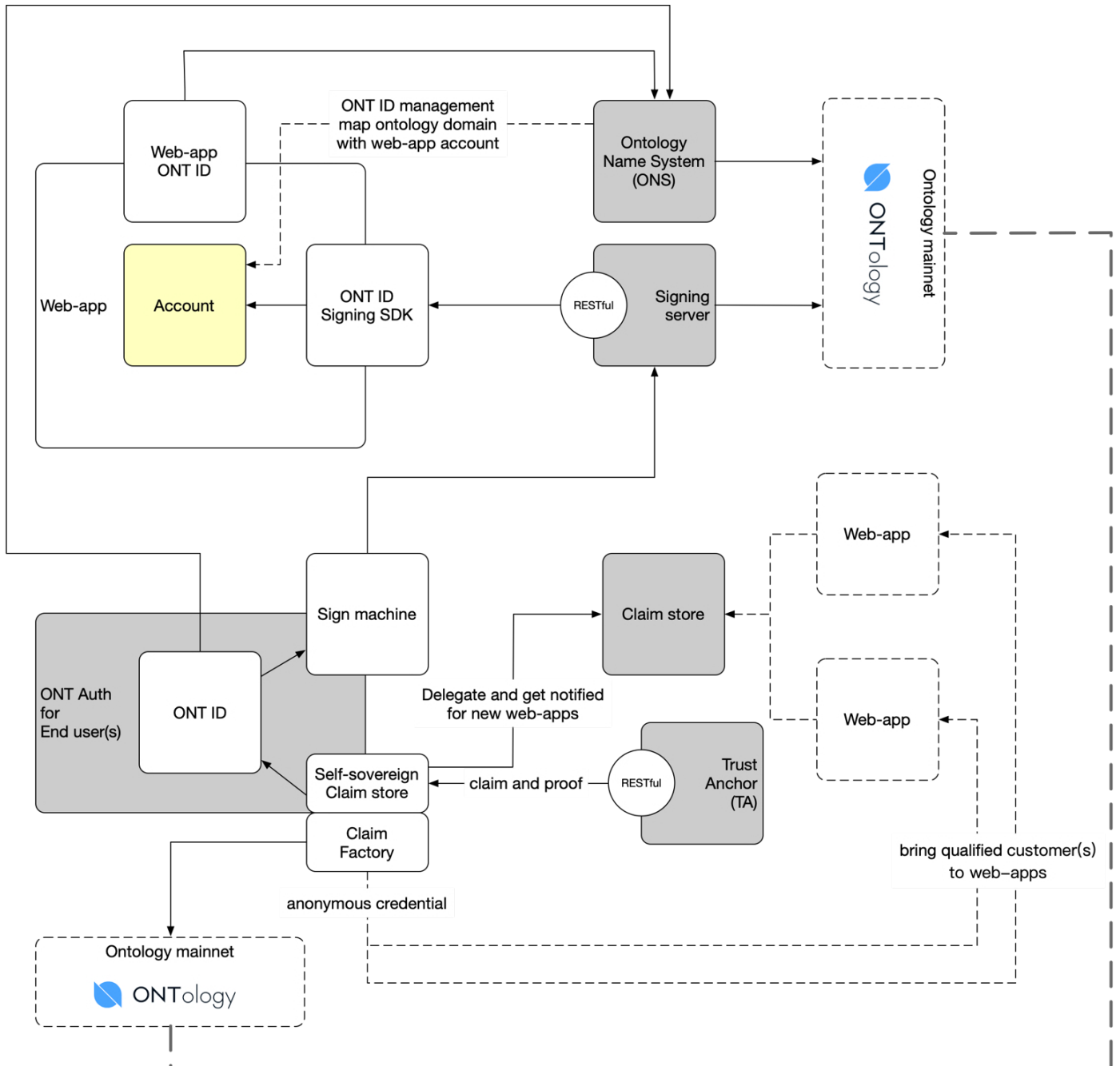
온톨로지(Ontology)의 탈중앙화 신원 프레임워크 ONT ID(<https://ontid.ont.io>)는 W3C 권고안에 따른 탈중앙화 식별자와 검증 가능한 자격 증명에 기반한 탈중앙화 신원 프레임워크입니다. 이는 개인, 기관, 객체, 자산 등 다양한 엔티티를 탈중앙화 네트워크와 중앙화 네트워크 모두에서 식별하고 상호 통신할 수 있도록 설계되었습니다. 개인, 기관, 객체, 콘텐츠 등 다양한 엔티티에 대해 디지털 및 암호 기반 신원을 생성하고 부여하는 데 사용될 수 있습니다.

ONT ID 프레임워크는 온톨로지 체인에만 국한되지 않습니다. 이 방식은 **이더리움(Ethereum)**과 **바이낸스 스마트 체인(Binance Smart Chain)**에도 구현되어 동일한 기능을 해당 네트워크들에도 확장합니다.

ONT ID는 데이터 권한 부여와 소유권 확인을 위한 자기주권(self-sovereign) 시스템을 제공하여 사용자에게 진정한 통제권을 부여합니다. ID 생성, 저장, 업데이트 및 기타 핵심 작업의 전체 과정은 완전히 자동화되고 탈중앙화된 방식으로 이루어지며, 이를 통해 생태계 내에서 운영하는 사용자는 자신과 연관될 수 있는 모든 데이터에 대한 완전한 통제와 소유권을 가질 수 있습니다. 다양한 생태계에 연결된 개별 ID는 여러 대리인(delegate)과 속성(attributes)을 가질 수 있습니다. 검증 가능한 자격 증명의 형태로, 엔티티는 데이터 소유권, 접근 권한, 유효성에 대한 클레임을 생성하고 검증할 수 있습니다. 네트워크에 속한 어떤 엔티티든 이를 수행할 수 있기 때문에, ONT ID 프레임워크는 높은 수준의 탈중개(disintermediation)를 갖춘 계정 메커니즘을 구축하기 위한 강력한 후보가 됩니다.

ONT ID의 몇 가지 특징은 다음과 같습니다:

- 탈중앙화(Decentralization)
- 자기주권(Self-sovereignty)과 관리
- 프라이버시 보호(Privacy protection)
- 보안(Security)
- 사용 용이성(Ease of use)



탈중앙화된 ONT ID 프레임워크

주요 특징 (Salient Features)

아래는 ONT ID 프레임워크의 주요 기능들입니다.

계정 메커니즘 (Account Mechanism)

지갑 주소와 연결되며 온톨로지, 이더리움, 바이낸스 스마트 체인 생태계의 애플리케이션 레이어에서 지원되는 계정 메커니즘.

크로스체인 신원 집계 (Cross-chain Identity Aggregation)

이더리움, 온톨로지, BSC 체인에 배포된 해당 컨트랙트와 연결하여 오라클(Oracle)과 같은 오프체인 서비스 형태의 상호운용성을 지원합니다. 또한 온톨로지와 대상 체인에 각각 배포된 ONT ID 컨트랙트 간의 크로스체인 통신도 지원합니다.

멀티 자산 지원 (Multi-asset Support)

비트코인, 이더리움, 온톨로지 등 여러 체인의 자산을 지원합니다.

신용 점수 (Credit Score)

자산 보유, 스마트 컨트랙트와의 트랜잭션 이력 등 트랜잭션 데이터를 기반으로 주소에 부여되는 평판 점수.

ONT ID 는 사용자가 신뢰성 증명을 요구하는 플랫폼(예: 무담보 대출)에서 서비스를 이용할 수 있도록 선택적으로 활성화할 수 있는 또 다른 기능인 *OScore* 를 제공합니다. *OScore* 에 대한 자세한 내용은 [여기\(https://ont.io\)](https://ont.io)에서 확인할 수 있습니다.

자기 주권 신원(Self-sovereign identity)

ONT ID 맥락에서, 엔티티(entity) 라는 용어는 다음을 의미합니다:

- 개인(Individuals) - ONT ID의 개별 사용자
- 법적 엔티티(Legal entities) - 기관, 조직, 기업 등
- 객체(Objects) - 휴대전화, 자동차, IoT 기기 등
- 콘텐츠(Content) - 연구, 창작 콘텐츠 등

신원(identity) 이라는 용어는 온톨로지 신뢰 네트워크 내에서의 엔티티의 신원을 엄격히 의미한다는 점이 중요합니다.

ONT ID 는 기본적으로 시스템 범위 내에서의 식별자(identifier)입니다. 네트워크의 모든 엔티티는 자신의 신원을 식별하고 관리하는 **온톨로지 식별자(Ontology Identifier, ONT ID)** 를 가지게 됩니다.

사용자는 온톨로지의 **ONTO** 지갑을 사용해 자신의 ONT ID를 등록할 수 있습니다. 다운로드 링크는 [여기\(https://ont.io\)](https://ont.io)에서 확인할 수 있습니다.

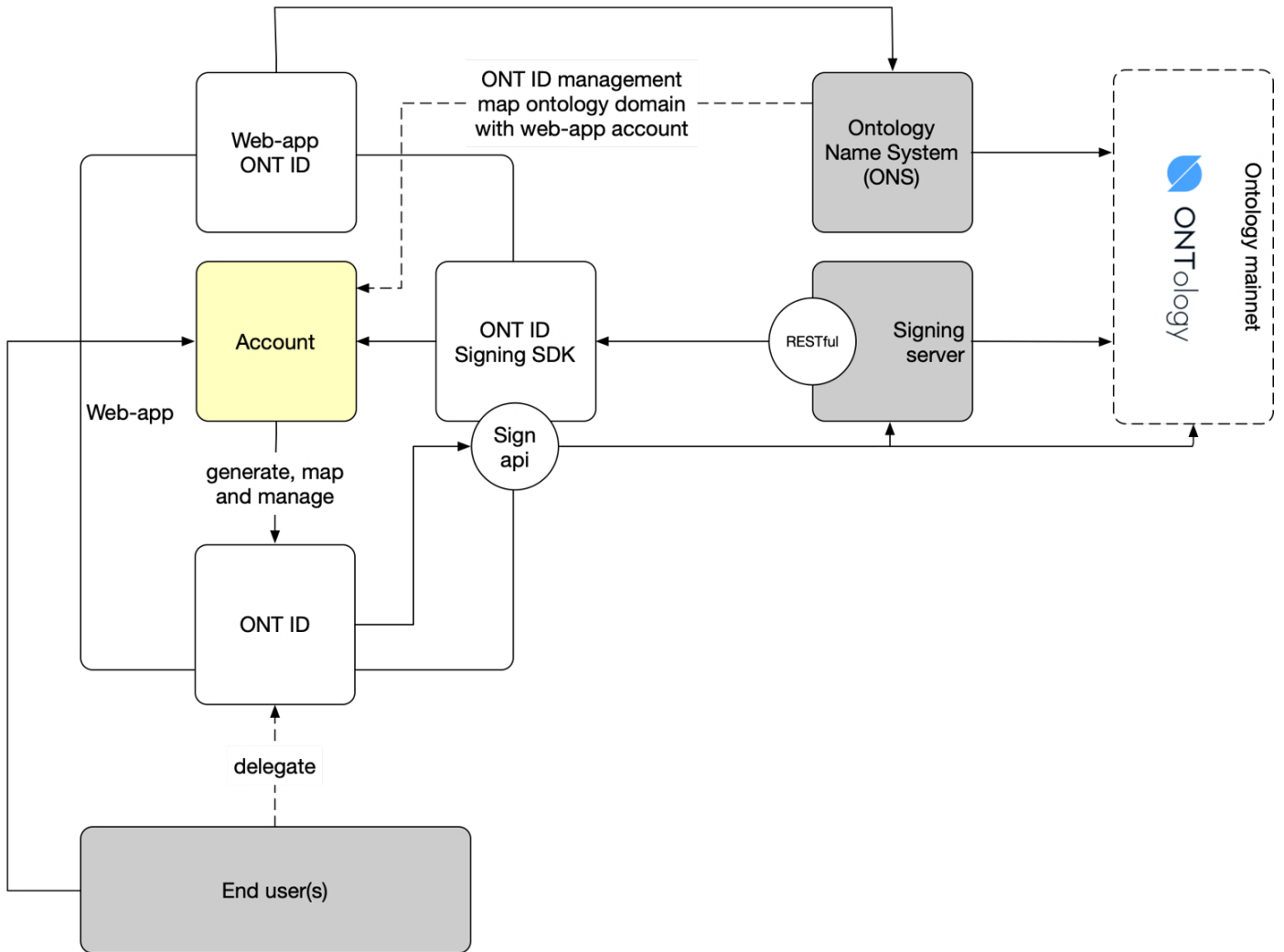
온톨로지 블록체인에서, 하나의 엔티티는 여러 개의 개별 ID에 대응할 수 있으며, 이 여러 ID들 사이에 어떠한 관계도 존재할 필요가 없습니다.

ONT Auth 와 Signing server 서비스는 온톨로지 메인넷에 배포되어 있습니다.

서명 서버(signing server)는 최종 사용자의 ONT ID를 애플리케이션 내 계정 시스템과 연결하는 백엔드 서비스입니다. 서명 서버에 대한 더 많은 정보는 [여기\(https://ont.io\)](https://ont.io)에서 확인할 수 있습니다.

전통 시스템을 위한 워크어라운드 (Workaround for Traditional Systems)

전통적인 기업들은 부분적으로 혹은 완전히 중앙화된 시스템을 보유하는 것이 일반적입니다. 따라서 이러한 시나리오와 사용 사례를 염두에 두고, 온톨로지는 중앙화된 시스템이 **ONT ID** 를 통합하고 사용할 수 있도록 워크어라운드를 개발했습니다.

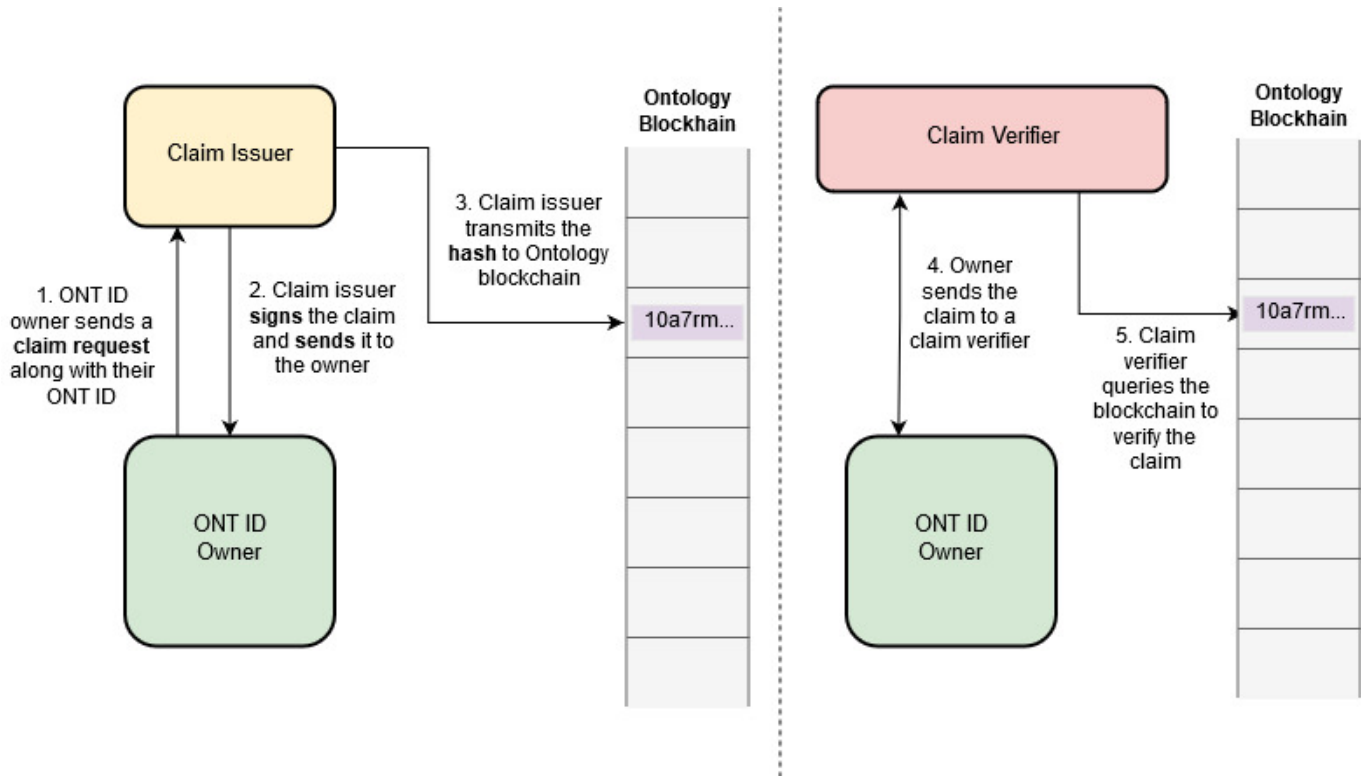


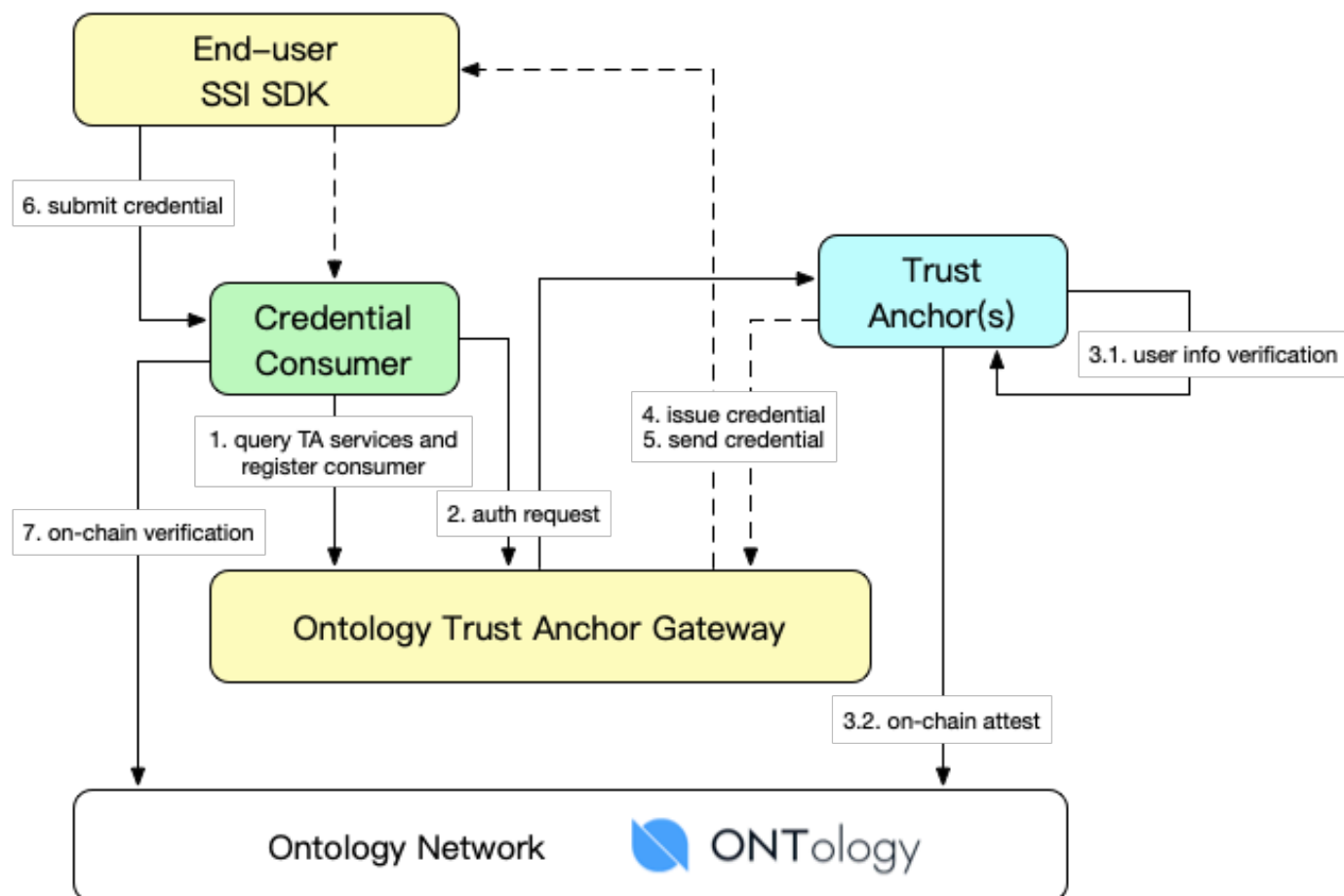
전통 시스템을 위한 워크어라운드

ONT ID의 **owner** 속성을 애플리케이션에서 최종 사용자로 변경함으로써, ID는 사용자에게 반환될 수 있습니다. 개념적으로, **ONT ID**는 사용자로부터 애플리케이션 서버 호스트로 위임됩니다. 이를 위한 조건들은 사전에 준비되며, 사용자는 해당 조건을 수락함으로써 이러한 ‘위임’을 승인하게 됩니다.

기술적으로, 위임된 **ONT ID**는 다음과 같은 작업을 수행할 수 있습니다:

- 사용자 대신 시스템 내에서 등록(registration) 및 로그인(login)과 같은 작업을 수행하고, 결과를 온톨로지 메인넷에 게시할 수 있습니다.





탈중앙화 식별자 (Decentralized Identifiers)

탈중앙화 식별자(DID, Decentralized Identifiers)는 검증 가능하고 자기주권(self-sovereign)적인 디지털 신원을 가능하게 하는 새로운 유형의 식별자로, W3C 탈중앙화 식별자 워킹 그룹에 의해 개발되었습니다. 해당 사양은 여기(<https://www.w3.org/TR/did-core/>)에서 확인할 수 있습니다.

현재 우리는 다음 체인에 대한 DID 메서드를 개발했습니다.

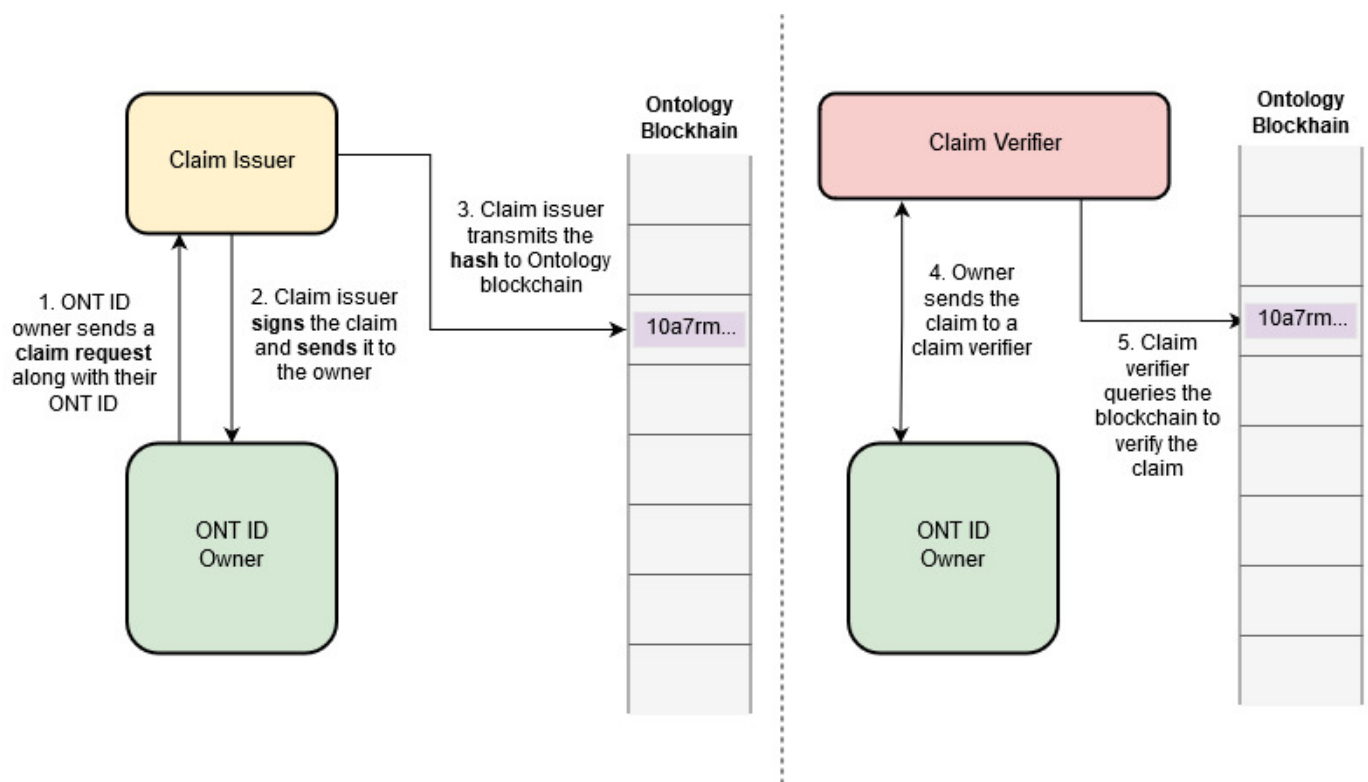
네트워크 (Network)	메서드 (Method)
온톨로지 (Ontology)	did:ont
이더리움 (Ethereum)	did:etho
바이낸스 스마트 체인 (Binance Smart Chain)	did:bnb

검증 가능한 자격 증명 (Verifiable Credentials)

신뢰 모델 개요 (Outline of the trust model)

온톨로지의 신뢰 메커니즘의 기반은 검증 가능한 자격 증명 시스템(verifiable credential system)에 있습니다.

엔티티들은 자격 증명을 발급하고 이를 고객에게 판매하며, 이를 통해 검증 시나리오가 형성됩니다. 자격 증명의 발급 요청, 생성, 소비로 이루어진 폐쇄 루프(closed loop)가 바로 신뢰 메커니즘을 구성합니다. 모든 엔티티는 클레임을 생성하고 검증할 수 있습니다.



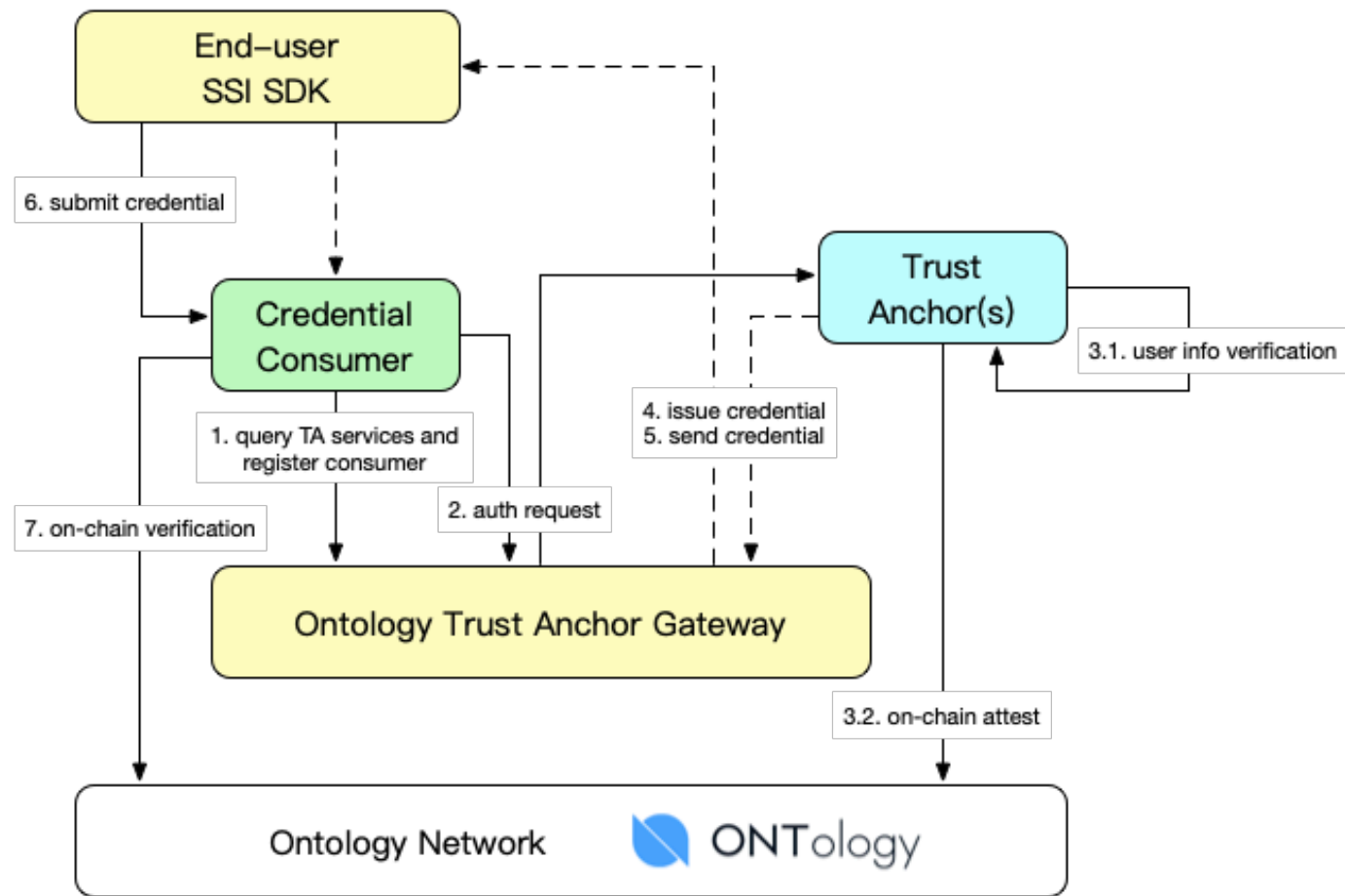
다음은 이 과정에 참여하는 주체들입니다.

자격 증명 소유자(Credential owner) – Ontology ID를 가지고 있습니다. 이 엔티티는 다른 엔티티(자격 증명 발급자라고 불림)가 발급한 검증 가능한 자격 증명(verifiable credential)을 획득합니다. 이 엔티티는 익명 자격 증명 기술(anonymous credential technology)을 통해 자격 증명을 조작(manipulate)하고, 자격 증명을 자격 증명 소비자에게 제공합니다. 따라서 자격 증명 소유자는 신뢰 판매자(trust seller)의 역할을 합니다.

자격 증명 발급자(Credential issuer) – Ontology ID를 가지고 있습니다. 이 엔티티는 특정 자격이나 인증을 위해 대상 엔티티를 보증(endorse)하기 위해 자격 증명을 발급합니다. 자격 증명 발급자 범주에는 신뢰 앵커(trust anchors), 즉 온톨로지 생태계에서 인증 서비스를 제공하는 파트너 또는 엔티티가 포함됩니다. 신뢰 앵커에는 정부 기관, 대학, 은행, 제3자 인증 서비스, 생체 인식 기술 회사 등이 포함될 수 있습니다. 자

자격 증명 발급자는 신뢰 네트워크의 일부인 엔티티들에 대해 다차원 인증을 제공합니다. 인증 과정과 결과는 데이터 프라이버시 보호와 함께 온톨로지 블록체인에 기록됩니다. 자격 증명 발급자는 자격 증명 소비자가 자격 증명을 검증할 수 있도록 표준화되고 신뢰할 수 있는 인증 방법을 제공합니다. 자격 증명 발급자는 신뢰 보증자(trust endorser)의 역할을 합니다.

자격 증명 소비자(Credential consumers) – 사용자의 검증 가능한 자격 증명을 수락하고, 해당 자격 증명에 대한 검증 프로세스를 시작합니다. 여기에는 다양한 시나리오가 포함되며, 예를 들어 면접자의 신원 정보, 학위, 산업 기술을 확인해야 하는 고용주 등이 있습니다. 자격 증명 소비자는 신뢰 구매자(trust buyer)의 역할을 합니다.



위의 작동 흐름에서 알 수 있듯이, 이 과정에는 세 가지 주요 동작이 포함됩니다:

- 자격 증명 요청 (Credential request)
- 자격 증명 발급 (Credential issue)
- 자격 증명 검증 (Credential verification)

발급 과정에는 두 개의 주체가 참여합니다. 즉, 자격 증명 발급자와 ONT ID를 소유한 엔티티입니다. 검증 가능한 자격 증명(verifiable credential)에는 자격 증명의 내용(시스템에 따라 달라질 수 있음), 디지털 서명, 그리고 블록체인 인증 기록이 포함됩니다. 일부 기록은 다음과 같습니다:

- **자격 증명 ID (Credential ID):** 자격 증명의 고유 식별자
- **자격 증명 내용 (Credential content):** 예를 들어 학위 증명서와 같은 특정 자격 또는 정보
- **자격 증명 메타데이터 (Credential metadata):**

- **생성 시간 (Created time):** 자격 증명에 생성된 타임스탬프
- **발급자 (Issuer):** 발급자의 ONT ID
- **수신자 (Recipient):** 수신자 측의 ONT ID
- **만료 시간 (Expiration time):** 자격 증명에 자동으로 만료되는 UNIX 타임스탬프
- **폐기 메커니즘 (Revocation mechanism):** 폐기 목록을 사용하거나 폐기 정보를 인증 계약(attestation contract)에 직접 기록
- **블록체인 증명 (Blockchain proof)**
- **서명 (Signature):**
 - 발급자의 공개 키 (Public key of the issuer)
 - 서명 값 (Signature value)

직원의 급여 증명서에 대한 검증 가능한 자격 증명 템플릿은 여기에서 확인할 수 있습니다. (https://github.com/ontio/ontology-DID/blob/master/claimtemplate/en/employment_certification_claimtemplate.md)

*중앙화된 ONT ID 시스템의 경우, 워크플로의 첫 번째 단계는 자격 증명 발급자에게 전송되는 요청 측면에서 다를 수 있습니다. 이는 소유자가 자신의 ONT ID와 자격 증명에 접근할 수 있는 다른 기관에 권한을 위임했기 때문입니다. 요청은 반드시 소유자 자신에 의해 전송될 필요가 없으며, 위임된 기관이 자격 증명을 시작하고 승인할 수도 있습니다.

발급 과정 (Issuance process)

발급 과정은 네 가지 주요 단계를 포함합니다:

1. ONT ID 소유자가 자격 증명 발급자에게 요청을 보내면서 프로세스를 시작합니다.
2. 자격 증명 발급자는 검증 가능한 자격 증명(verifiable credential)을 생성하고 안전한 방법으로 수신자에게 전송합니다. 자격 증명에 수신자의 공개 키(public key)를 사용해 암호화됩니다.
3. 소유자(또는 대리인)는 자격 증명에 서명한 후 이를 자격 증명 발급자에게 다시 보냅니다.
4. 자격 증명 발급자는 최종적으로 서명 과정을 완료하고 자격 증명의 상태를 '인증됨(attested)'으로 설정합니다. 이후 자격 증명에 온톨로지 블록체인과 소유자(또는 대리인)에게 전송됩니다.

이 전체 과정은 위에서 설명된 자격 증명 워크플로의 단계 1~3을 기본적으로 포함합니다.

자격 증명 검증 (Credential verification)

자격 증명을 검증하는 과정에는 세 가지 주요 동작이 포함되며, 이는 위에서 설명된 워크플로의 단계 4~5에 해당합니다.

- 자격 증명에 블록체인에 존재하는지 검증
- 서명을 검증하고 만료되었는지 여부 확인
- 자격 증명에 철회(revoked)되었는지 확인

블록체인 기록 검증 (Blockchain record verification)

검증 가능한 자격 증명(credential)의 기록이 블록체인에 존재하는지 확인하는 것이 필요합니다. 노드가 온톨로지 블록체인과 완전히 동기화되지 않은 경우, 머클 증명(merkle proof)을 사용하여 검증 가능한 자격 증명 트랜잭션을 검증할 수 있습니다.

머클 증명은 배열로 구성되며, 각 요소는 두 개의 데이터 항목 — 방향(direction)과 해시(hash) — 을 포함합니다.

- **Direction:** 해당 배열 요소가 속한 머클 트리의 가지(branch)를 나타냅니다. 가능한 값은 두 가지입니다.
- **Hash:** 요소 데이터의 해시 값을 나타냅니다.

머클 증명을 검증하는 알고리즘은 다음과 같습니다:

1. 트랜잭션이 `proof.BlockHeight` 로 인덱싱된 블록에 포함되어 있는지 확인합니다. 포함되어 있지 않으면 `false` 를 반환합니다.
2. `p <- GetBlockHash(proof.BlockHeight)` 를 실행합니다.
3. `proof.Nodes` 의 각 요소에 대해, `p` 를 다음과 같이 갱신합니다.
 - `e.Direction == "Left"` 인 경우, `p <- H(e.TargetHash, p)`
 - 그렇지 않으면, `p <- H(p, e.TargetHash)`
4. `p` 가 `proof.MerkleRoot` 와 같으면 `true` 를 반환합니다. 그렇지 않으면 `false` 를 반환합니다.

또한 자격 증명 인증의 상태(attestation status)를 검증하는 것도 필요합니다. 이는 `proof.ContractAddr` 주소를 사용해 인증 계약(attestation contract)의 조회 인터페이스 `GetStatus()` 를 호출함으로써 수행할 수 있습니다. 상태가 `not attested` 인 경우, 오류가 반환됩니다.

서명 검증 및 만료 시간 (Signature verification and Expiration time)

서명을 검증할 때는 공개 키 ID(public key ID)를 사용하여 공개 키 값과 현재 상태를 가져와야 합니다. 그런 다음 검증 알고리즘이 호출되어 검증 과정을 수행합니다.

공개 키 ID의 형식은 `<ONTID>#keys-<number>` 입니다.

공개 키 ID는 공개 키의 상태를 조회하는 ONT ID 스마트 컨트랙트 메서드를 호출하는 데 사용됩니다.

`GetPublicKeyStatus(byte[] ontId, byte[] pkId)`

응답에는 다음 정보가 포함됩니다:

- **publicKey:** 공개 키 값 (16진수)
- **status:** 가능한 두 가지 값 — `InUse`, `Revoked`

서명 검증 결과는 세 가지 중 하나일 수 있습니다:

- 서명이 유효하지 않음 (Signature is invalid)

- 서명이 유효함 (Signature is valid)
- 서명이 유효하지만 공개 키가 폐기됨 (Signature is valid and the public key is revoked)

다음으로, 만료 시간(expiration time)은 타임아웃 기간이 만료되었는지 여부를 확인함으로써 검증할 수 있습니다.

폐기 검증 (Revoking verification)

현재 두 가지 폐기 모드가 제공됩니다: 폐기 목록(revocation list)과 폐기 조회 인터페이스(revocation inquiry interface).

다음은 폐기 목록 자격 증명 요청의 예시입니다. 이 예시에는 목록의 URL이 포함되어 있습니다.

```
"clm-rev": {
  "type": "RevocationList",
  "url": "https://example.com/rev/1234"
}
```

취소 조회 인터페이스를 예로 들어, 취소 정보가 증명 계약에 포함된 경우, 증명 계약의 조회 인터페이스 GetStatus를 호출할 때 반환된 상태 필드가 증명된 경우에만 취소 확인이 성공적으로 반환됩니다. 증명 상태 필드가 취소된 경우에는 실패로 반환됩니다.

```
"clm-rev": {
  "type": "AttestContract",
  "addr": "8055b362904715fd84536e754868f4c8d27ca3f6"
}
```

취소 목록에는 주로 고유 식별자와 취소된 검증 가능한 자격 증명의 취소 시간이 포함됩니다.

검증 가능한 자격 증명의 형식

발행자(issuer)와 수신자(recipient) 간에 전송되는 자격 증명(credential)의 구조를 구축하기 위해 [JSON Web Token](#) 형식을 확장하여 사용한다.

토큰의 기본 구조는 세 가지 부분으로 구성된다:

- 헤더(Header)
- 페이로드(Payload)
- 서명(Signature)

표준 **JWT** 속성이 광범위하게 사용되며, 특정 특수한 경우에는 **사용자 정의(custom)** 속성이 정의된다.

표준 **JWT** 형식은 블록체인 증명(blockchain proof)을 끝에 추가함으로써 확장된다.
일반적인 검증 가능한 자격 증명은 다음과 같은 레이아웃을 가진다:
`header.payload.signature.blockchain_proof`

*blockchain_proof*는 일부 경우에는 필요하지 않으며, 따라서 선택적(optional)이다.

헤더(Header)

header는 자격 증명에 대한 서명을 검증하는 데 사용되는 형식, 서명 방식(signature scheme), 그리고 공개 키의 ID를 정의한다.

```
{
  "alg": "ES256",
  "typ": "JWT-X",
  "kid": "did:ont:TRAtosUZHNSiLhzBdHacyxMX4Bg3cjWy3zf#keys-1"
}
```

속성(Attribute)	설명(Description)
alg	사용할 서명 방식을 지정한다. 지원되는 값들의 목록은 여기 에서 확인할 수 있다.
typ	"JWT": 블록체인 증명이 자격 증명에 포함되지 않는다. "JWT-X": 블록체인 증명이 자격 증명의 일부로 포함된다.

자격 증명 ID, 자격 증명 콘텐츠 및 메타데이터는 JSON 객체로 패키징되어 페이로드 역할을 합니다. JWT 사양에 명시된 등록된 자격 증명 이름 중 일부인 jti, iss, sub, iat, exp를 사용합니다.

```
{
  "ver": "0.7.0",
  "iss": "did:ont:TRAtosUZHNSiLhzBdHacyxMX4Bg3cjWy3r",
  "sub": "did:ont:SI59Js0zpNSiPOzBdB5cyxu80BO3cjGT70",
  "iat": 1525465044,
  "exp": 1530735444,
  "jti":
"4d9546fdf2eb94a364208fa65a9996b03ba0ca4ab2f56d106dac92e891b6f7fc",
  "@context": "https://example.com/template/v1",
  "clm": {
    "Name": "Bob Dylan",
    "Age": "22"
  },
  "clm-rev": {
```

```

    "typ": "AttestContract",
    "addr": "8055b362904715fd84536e754868f4c8d27ca3f6"
  }
}

```

속성(Attribute)	설명(Description)
ver	따르고 있는 자격 증명 사양의 버전을 지정한다.
iss	발행자의 ONT ID
sub	수신자의 ONT ID
iat	자격 증명이 생성된 시점의 UNIX 타임스탬프
exp	자격 증명이 자동으로 만료되는 시점의 UNIX 타임스탬프
jti	검증 가능한 자격 증명의 고유 식별자
@context	각 필드와 해당 값을 명시적으로 정의하는 자격 증명 콘텐츠 정의 문서의 URI
clm	자격 증명 콘텐츠를 포함하는 객체
clm-rev	자격 증명이 사용하는 폐기(revocation) 메커니즘을 정의하는 객체

지원되는 폐기 메커니즘의 목록은 [여기\(https://ont.id\)](https://ont.id)에서 확인할 수 있다.

다음으로, 자격 증명을 발행하기 위해서는 자격 증명 ID, 콘텐츠, 메타데이터를 포함하는 JSON 객체를 구성해야 한다. 이 JSON 객체는 표준 직렬화(serialization) 방법을 사용하여 **직렬화**될 수 있다. 그 후 발행자의 개인 키 중 하나를 사용하여 페이로드와 헤더의 바이너리 데이터를 **서명(sign)** 한다.

직렬화 후, **페이로드(payload)** 는 다음과 같은 형태가 된다—

```

{
  "ver": "0.7.0",
  "iss": "did:ont:TRAtosUZHNSiLhzBdHacyxMX4Bg3cjWy3r",
  "sub": "did:ont:SI59Js0zpNSiPOzBdB5cyxu80BO3cjGT70",
  "iat": 1525465044,
  "exp": 1530735444,
  "jti":
"4d9546fdf2eb94a364208fa65a9996b03ba0ca4ab2f56d106dac92e891b6f7fc",
  "@context": "https://example.com/template/v1",

```

```

"clm": {
  "Name": "Bob Dylan",
  "Age": "22"
},
"clm-rev": {
  "Type": "Contract",
  "Addr": "8055b362904715fd84536e754868f4c8d27ca3f6"
}
}

```

서명(Signature)

요청의 헤더와 페이로드가 구성된 후, 서명은 **JWS 표준**에 따라 계산된다.

이 표준의 전체 설명은 IETF 문서의 RFC 7515 Section 5.1(<https://datatracker.ietf.org/doc/html/rfc7515#section-5.1>)에서 확인할 수 있다.

절차는 다음과 같다:

1. **JWS** 사양에 따라 헤더와 페이로드를 직렬화(serialization)하여 서명 입력값(signing input)을 계산한다.

```
sig := sign(Base64URL(header) || . || Base64URL(payload))
```

1. 서명 입력에 사용되는 특정 서명 방식에 대해 지정된 방법을 사용하여 JWS 서명을 계산합니다.
2. 서명을 인코딩합니다.

```
signature := Base64URL(sig)
```

블록체인 증명

증명 객체의 일반적인 형식은 다음과 같습니다:

```

{
  "Type": "MerkleProof",
  "TxnHash":
"c89e76ee58ae6ad99cfab829d3bf5bd7e5b9af3e5b38713c9d76ef2dcba2c8eo",
  "ContractAddr": "8055b362904715fd84536e754868f4c8d27ca3f6",
  "BlockHeight": 10,
  "MerkleRoot":
"bfc2ac895685fbb01e22c61462f15f2a6e3544835731a43ae0cba82255a9f904",
  "Nodes": [{
    "Direction": "Right",

```

```

    "TargetHash":
"2fa49b6440104c2de900699d31506845d244cc0c8c36a2fffb019ee7coc6e2f6"
  }, {
    "Direction": "Left",
    "TargetHash":
"fc4990f9758a310e054d166da842dab1ecd15ad9f8f0122ec71946f20ae964a4"
  }]
}

```

속성 (Attribute)	설명(Description)
Type	고정값 'Merkleproof'
TxnHash	증명 계약(attestation contract)에서 자격 증명 ID를 증명하는 트랜잭션의 해시
ContractAddr	증명 계약의 주소
BlockHeight	증명 계약을 포함하는 블록의 높이
MerkleRoot	트리 크기가 BlockHeight 와 같을 때의 머클 트리 루트
Nodes	머클 트리 내 블록의 포함 증명(inclusion proof)

MerkleProof는 다음과 같은 방식으로 인코딩된다:

```

BASE64URL(MerkleProof)

```

따라서 완전한 **검증 가능한 자격 증명**이 생성됩니다. 최종 구조는 다음과 같습니다:

```

BASE64URL(Header) || '.' || BASE64URL(Payload) || '.' || BASE64URL(Signature) '.'
|| BASE64URL(MerkleProof)

```

증명 계약서

확인 가능한 자격 증명의 **증명 계약서**는 **증명 서비스** 및 **기록 가용성** 정보를 제공합니다. 즉, 취소되었는지 여부입니다.

사용 가능한 방법은 아래에 설명되어 있습니다:

- 증명 커밋

```
bool Commit(byte[] claimId, byte[] committerOntId, byte[] ownerOntId);
```

증명 계약서에서 'claimId'는 자격 증명의 고유 식별자 역할을 합니다. 이는 첫 번째 매개변수입니다. 'committerOnTid'는 **증인의 ONT ID**입니다. 'ownerOnTid'는 **owner의 ONT ID**입니다.

이 메서드는 자격 증명이 **증명되지 않은** 인 경우에만 '참'으로 반환되며, **커미터**가 메서드를 호출한 경우에만 '거짓'으로 반환됩니다. 그렇지 않으면 '거짓'으로 반환됩니다.

증명이 완료되면 자격 증명의 상태가 **증명된** 으로 업데이트됩니다.

- 자격 증명 취소

```
bool Revoke(byte[] claimId, byte[] revokerOntId);
```

이 방법은 자격 증명이 **attested**이고 'revokerOnTid'가 **attester의 ONT ID**와 동일한 경우에만 'true'를 반환하며, 그렇지 않으면 'false'를 반환합니다.

- 증명 조회 방법

```
byte[] GetStatus(byte[] claimId);
```

이 메서드는 자격 증명의 상태를 반환합니다. 응답에는 **두 부분**의 정보가 포함되어 있습니다:

- **상태:** '증명되지 않음', '증명됨', '증명됨', '증명됨'
- **행위자의 ONT ID**

서명 방식

현재 지원되는 서명 방식은 다음과 같습니다:

체계	암호화 기술
ES224	ECDSA와 SHA224
ES256	ECDSA와 SHA256
ES384	SHA384와 함께하는 ECDSA
ES512	SHA512와 함께하는 ECDSA
ES3-224	ECDSA와 SHA3 224
ES3-256	ECDSA (SHA3 256 포함)
ES3-384	ECDSA with SHA3 384
ES3-512	ECDSA와 SHA3 512
ER160	ECDSA와 RIPEMD160
SM	SM2와 SM3
EDS512	EDDSA (SHA256 포함)

ONT 로그인(ONT Login)

ONT 로그인은 탈중앙화된 범용 인증 로그인 컴포넌트로, 개발자가 인증 구현의 세부 사항을 숨기고, 엔터프라이즈 및 서비스에 빠르게 Web 3.0 수준의 보안 로그인 경험을 제공할 수 있도록 돕는다.

현재 사용자는 ONT ID를 통해 서비스에 로그인해야 한다.

향후에는 지갑 주소를 사용하여 직접 로그인하는 것도 가능해질 예정이다.

특징(Characteristics)

- **신뢰 불필요(Trustless):** 탈중앙화된 신원을 통해 사용자 데이터를 안전하게 인증한다.
- **자기주권(Self-Sovereign):** 사용자는 신원 정보를 로컬에 저장하고, 필요할 때 서비스 접근을 허가한다.
- **비밀번호 없음>Passwordless):** 사용자는 복잡한 비밀번호를 설정하거나 기억할 필요가 없다.
- **사용자 편의성(End-user Convenient):** 한 번의 개인 정보 검증으로 여러 서비스에서 언제든지 재 사용 가능하다.

- **다중 언어 지원(Multi-Language):** 프론트엔드용 JavaScript SDK와 백엔드용 Golang/Java SDK를 제공하여 빠른 통합이 가능하다.
- **높은 호환성(High Compatibility):** 전통적인 인터넷 서비스와 퍼블릭 체인, 연합 체인을 포함한 다양한 블록체인 시스템을 지원한다. 여러 체인의 DID 프로토콜과 다양한 서명 방식을 지원한다.
- **보안 프로세스(Secure Process):** 챌린지-응답(challenge-response) 인증 방식과 디지털 서명 기술을 통해 인증 과정의 보안을 보장한다.

사용자 참여(Engagement) 향상

사용자명과 비밀번호 기반의 인증 메커니즘은 사용자가 취약점을 방지하기 위해 비밀번호를 유지·관리하는 데 많은 노력을 요구한다. 인증은 일반적으로 사용자의 참여 여정을 설계할 때 디지털 서비스가 거치는 첫 단계이다. 활성 사용자는 제품과 상호작용할 때 주기적으로 로그인을 수행해야 하므로, 인증은 사용자 참여에서 중요한 역할을 한다.

ONT 로그인은 더욱 원활한 사용자 로그인 경험을 제공한다. 사용자는 탈중앙 신원 지갑 내에서 이메일, 전화번호 등을 한 번만 검증하면 검증 가능한 자격 증명(verifiable credential)을 얻을 수 있다. 서비스 로그인 시, 사용자는 해당 자격 증명을 직접 제시함으로써 여러 번의 번거로운 인증 절차를 생략할 수 있다.

사용자 데이터 보안 강화

오늘날 많은 사람들은 다양한 상황에 맞춰 여러 개의 디지털 신원을 보유하고 있으며, 여러 서비스에서 동일한 비밀번호를 사용하는 경향이 있다. 하나의 서비스가 침해될 경우, 다른 서비스들에 대한 접근 역시 위험에 노출된다. 또한 많은 서비스가 사용자에게 전화번호나 소셜 계정 등 개인 정보를 연동하도록 요구하지만, 사용자는 이 정보가 안전하게 보관되는지 알 수 없다.

ONT 로그인은 탈중앙 신원과 디지털 서명 기술을 기반으로 한 챌린지-응답 인증 방식을 채택한다. 이 인증 메커니즘은 서버와 데이터베이스 간 충돌로 인한 위험을 방지하며, 사용자 정보 보안을 향상시킨다. 더 나아가, 사용자만이 자신의 개인 데이터에 직접 접근할 수 있으므로 데이터 유출의 가능성이 크게 감소한다.

데이터 관리 효율 향상

ONT 로그인을 사용하면 민감한 사용자 개인 정보, 사용자명, 비밀번호를 직접 관리해야 하는 번거로움에서 벗어날 수 있으며, 관련 규제에 대응하기 위한 관리 부담도 줄일 수 있다.

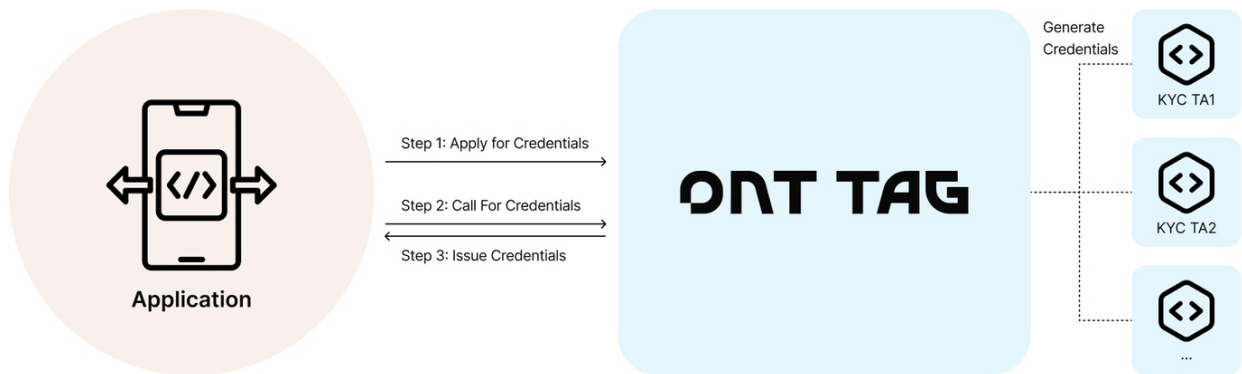
ONT TAG

ONT TAG는 ONT ID와 온톨로지 신뢰 생태계(Ontology trust ecosystem)를 기반으로 한 개방형 탈중앙화 인증 플랫폼이다.

이 플랫폼은 개인, 금융, 사물, 업무 등을 위한 KYC 서비스를 제공한다.

온톨로지 신뢰 생태계는 IdentityMind, CFCA, Shufti Pro 등 전 세계 신원 인증 서비스를 제공하는 신

로 앵커(trust anchor)들과 이메일, 모바일, 소셜 미디어 인증 방식 등을 통합하고 있다.



ONT TAG가 제공하는 기능

자신의 앱이나 플랫폼을 운영 중이라면, ONT TAG는 사용자 인증 과정을 지원할 수 있다. 비즈니스 요구사항에 따라, KYC, 신원 인증 서비스, 로그인 등 기능을 위해 ONT TAG를 플랫폼이나 제품에 통합할 수 있다.

ONT TAG의 주요 장점은 다음과 같다:

- 온톨로지 신뢰 생태계는 이미 전 세계 인증 서비스와 연결되어 있으며, 218개국 및 지역의 사용자를 포괄한다.
- 저비용의 일회성 인증을 제공하며, 여러 번 재사용이 가능해 다중 인증 비용을 절감한다.
- 프로토콜은 종단 간 암호화(end-to-end encryption) 규격을 사용하여 전 과정에서 사용자 데이터를 보호한다.
- 모든 인증 행위와 결과는 온톨로지 블록체인에 증명(attestation)된다.
- 명세는 영지식 증명(zero-knowledge proof) 등 암호학적 알고리즘을 지원하며, 사용자는 자신의 신원 정보를 선택적으로 제시할 수 있어 개인정보 보호 수준을 극대화할 수 있다.

데이터 프라이버시(Data Privacy)

어떠한 형태로도 사용자 데이터는 TAG 서버에 저장되지 않는다. 모든 자격 증명은 사용자에게 의해 직접 저장 및 관리된다. 사용자가 자격 증명에 대한 접근 권한을 잃은 경우, 복구가 불가능하며 새 자격 증명을 발급받기 위한 절차를 다시 거쳐야 한다.

머큐리(Mercury)

머큐리(Mercury)는 신뢰 불필요(trustless)한 P2P(피어 투 피어) 탈중앙화 통신 프로토콜로, 다양한 주체들이 서로 간에 메시지, 검증 가능한 자격 증명(verifiable credential), 검증 가능한 프레젠테이션(verifiable presentation)을 안전하게 전송할 수 있도록 한다.

머큐리를 사용하면 각 주체(entity)는 탈중앙화 신원(decentralized identity)을 보유하게 되며, 탈중앙화 식별자(decentralized identifier, DID)를 통해 식별된다.

머큐리는 DID 방식에 구애받지 않는(identifier-agnostic) 구조로, W3C DID 메서드 등록부(W3C DID method registries)에 정의된 다양한 DID 방식을 지원한다.

현재는 온톨로지 팀이 개발한 ONT ID를 지원하며, 향후에는 더 많은 탈중앙화 식별자뿐 아니라 ENS 및 기타 탈중앙화 도메인 네이밍 시스템도 지원할 예정이다.

머큐리는 암호화(encryption)와 서명(signature) 체계를 적용하여, 보안성과 프라이버시를 동시에 보장하는 통신을 가능하게 한다.

머큐리는 주체 간의 통신 연결, 메시지 전송, 검증 가능한 자격 증명 및 프레젠테이션 전송을 위해 여러 DID 기반 하위 프로토콜(sub-protocol)을 정의한다.

머큐리가 정의한 하위 프로토콜군에는 연결 프로토콜(connection protocol), 일반(암호화된) 메시지 교환 프로토콜(general message exchange protocol), 검증 가능한 자격 증명 및 프레젠테이션 전송 프로토콜이 포함된다.

기본 아키텍처(Basic Architecture)

머큐리에서는 각 주체가 에이전트(agent)를 통해 상호 통신한다.
이 시스템에는 세 가지 유형의 에이전트가 존재한다:

사용자 에이전트(User Agent)

사용자 에이전트는 특정 최종 주체(end entity)에 의해 제어되며, 모바일 앱이나 기타 리치 클라이언트(rich client)로 구현되거나 내장될 수 있다.
사용자 에이전트의 중요한 특징 중 하나는 **항상 온라인 상태(24/7)가 될 수 없다는 점**이다.

주체는 자신의 비밀 키(secret key)를 보관하고, 검증 가능한 자격 증명(verifiable credential)을 로컬 저장소(local storage)에 저장한 후, 사용자 에이전트를 통해 통신을 개시하거나 메시지를 안전하게 전송할 수 있다.

클라우드 에이전트(Cloud Agent)

사용자 에이전트가 지속적으로 온라인 상태를 유지할 수 없는 비즈니스 시나리오에서는 메시지와 자격 증명을 중계(relay), 전달(forward), 일시적으로 저장(store)해야 한다.
이때 클라우드 에이전트는 메시지 라우팅(routing)에 있어 중요한 역할을 수행한다.
클라우드 에이전트는 자체적으로 공공 DID(public decentralized identity)와 서비스 엔드포인트(service endpoint) 속성을 가지며, 해당 비밀 키를 적절히 보호해야 한다.

서비스 에이전트(Service Agent)

서비스 에이전트는 클라우드 에이전트의 일종으로, 특정 검증 가능한 자격 증명(예: 제3기관이 발급하는 학위 증명서 등)을 발행하는 서비스 기능을 제공한다.

서비스 에이전트 역시 공공적이고 인증된 DID를 가져야 하며, 이에 따른 비밀 키를 안전하게 보관해야 한다.

하위 프로토콜(Sub-Protocols)

연결 프로토콜(Connection Protocol)

서로 통신하고자 하는 주체들은 통신 파트너와 연결을 설정해야 한다.

머큐리에서는 이러한 연결이 연결 프로토콜을 통해 설정된다.

일반 메시지 교환 프로토콜(General Message Exchange Protocol)

두 주체 간의 연결이 설정된 후에는 서로 메시지를 주고받을 수 있다.

이때 전송되는 메시지는 프라이버시 보호를 위해 암호화 및 서명 등의 암호학적 방식으로 보호될 수 있다.

검증 가능한 자격 증명 및 프레젠테이션 전송 프로토콜(Verifiable Credential and Presentation Transmission Protocol)

검증 가능한 자격 증명 시스템에서는 보유자, 발행자, 검증자의 세 가지 역할이 있습니다. 발행자는 보유자의 요청에 따라 보유자에게 검증 가능한 자격 증명을 발급할 수 있으며, 보유자는 증명 목적으로 자신의 자격 증명에서 검증 가능한 프레젠테이션을 생성할 수 있습니다. 검증 가능한 프레젠테이션을 얻은 검증자는 프레젠테이션을 암호화하여 검증할 수 있습니다.

검증 가능한 자격 증명 및 프레젠테이션 전송 프로토콜은 위의 세 가지 역할이 서로 어떻게 상호 작용하는지를 정의합니다.

자세한 내용은 Mercury의 GitHub 페이지를 방문하세요. (<https://github.com/ont-id/mercury>)

OScore

크로스체인 평판 및 신용 평가 프로토콜(Cross-chain reputation and credit evaluation protocol)

개요(Overview)

OScore는 온체인 행위(on-chain behaviour)를 기반으로 특정 기준에 따라 계정의 평판(reputation)을 객관적으로 평가할 수 있도록, 트랜잭션 데이터를 분석하여 점수를 부여하는 프로토콜이다.

이 점수는 관련된 ONT ID(또는 다른 DID 기반 구현) 계정 및 주소에 다음과 같은 온체인 활동 데이터를 근거로 산출된다:

1. 자산 교환 규모(Asset exchange volume)
2. 스마트 계약 연관성(Smart contract association)
3. 과거 보유 내역(Historical holdings)
4. 기존 신용 이력(Existing credit history)

OScore 알고리즘이 사용하는 데이터 포인트는 애플리케이션, 플랫폼, 또는 생태계에 따라 수정 및 맞춤화(customization)가 가능하다.

따라서 각 애플리케이션은 자사 생태계 내 주소와 ID에 대해 독자적인 알고리즘을 사용해 맞춤형 점수를 계산할 수 있다.

특성(Characteristics)

통합 및 집계(Consolidation/Aggregation)

여러 체인에서 발생한 자산 및 온체인 활동을 개별 ID 아래에 연결할 수 있다.

유연성(Flexibility)

빠른 통합을 위한 경량 API 기반 플러그인과, 네이티브 통합을 위한 SDK를 제공한다.

보안성과 프라이버시(Security and Privacy)

데이터 및 속성의 선택적·최소한의 공개를 지원한다.

활용 사례(Usecases)

OScore를 활성화한 사용자는 ONT ID를 통합한 플랫폼과 사용자 모두에게 다음과 같은 이점을 제공할 수 있다:

신용 점수(Credit Score)

- 사용자는 DeFi 플랫폼이나 기타 신용 기반 서비스에서 신용 혜택을 받을 수 있다. 이때 사용자는 자발적으로 신뢰 지수를 자격 증명(credential)의 형태로 제시할 수 있다. 플랫폼은 해당 데이터를 알고리즘이 처리하는 공개 데이터로 확인할 수 있으므로, 이 메커니즘은 완전히 신뢰 불필요(trustless)한 방식으로 작동한다.

시빌 방지 시스템(Anti-sybil System)

- 플랫폼은 경량 OScore 웹 플러그인(또는 API)을 통합하여, 사용자가 특정 기능을 잠금 해제하거나 캠페인(예: ICO, 에어드롭)에 참여하기 위해 수행해야 하는 다양한 수준의 실사(due diligence) 검증 절차를 거치도록 유도할 수 있다. 이 과정에서 사용자의 프라이버시는 보호되며, 플랫폼은 필요한 정보만 접근할 수 있다. 이를 통해 플랫폼은 컴플라이언스를 유지하면서 사용자 도달 범위를 극대화할 수 있다.

DDXF

온톨로지 분산 데이터 교환 프레임워크(Distributed Data Exchange Framework)에 대한 간략한 소개

소개(Introduction)

기업, 조직, 기관이 인터넷 시스템에 점점 더 의존함에 따라, 기존 인프라가 최종 사용자(end user)의 요구를 충족하기 위한 압력이 크게 증가하고 있다.

그 결과, 상호 연결된 **사물인터넷(IoT)** 또는 인터넷 시스템에 대한 수요가 지속적으로 높아지고 있다.

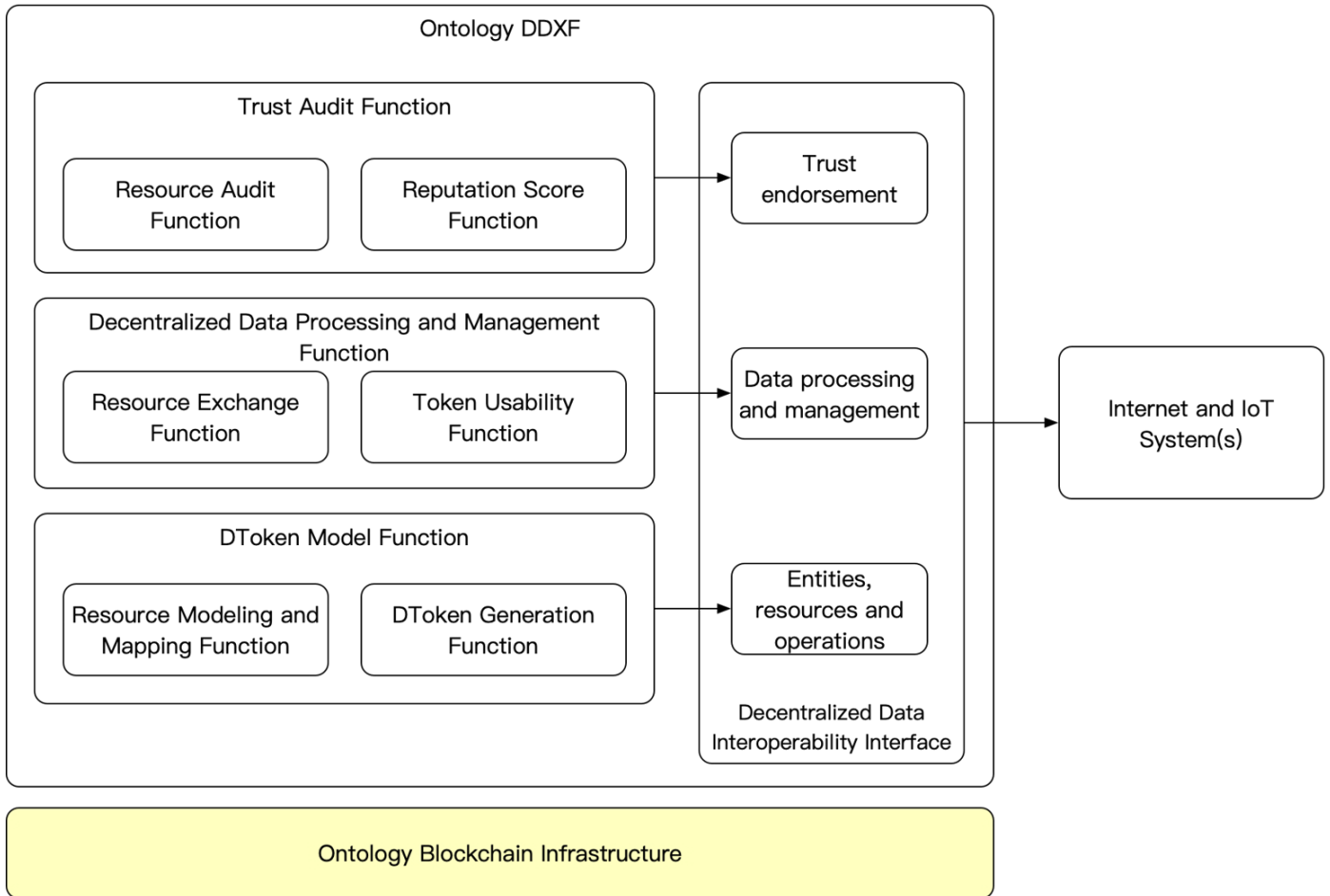
정보 데이터베이스 및 분석 아키텍처·인프라를 포함한 전통적 자원 관리 방식은 여전히 필수적이다.

그러나 데이터 관리 수요가 증가함에 따라, 다양한 출처로부터 복잡하고 방대한 데이터 스트림을 처리할 수 있는 용량과 역량이 필요해지고 있다.

이러한 데이터는 다른 정보원과 결합하여 가치를 극대화하기 위해 안전하게 처리되고 관리되어야 한다.

온톨로지의 분산 데이터 교환 프레임워크(DDXF, Distributed Data Exchange Framework)는 포괄적인 **데이터 처리 및 관리(data processing and management)**를 지원하는 인프라 프레임워크와 가이드라인을 정의하며, **계층적(layered)**이고 **데이터 중심적(data-centric)**인 패러다임을 달성하기 위한 합리적인 방안을 포함한다.

효율적인 데이터 처리 및 관리 시스템의 특성에 대한 광범위한 연구를 바탕으로, **DDXF**는 다양한 이해관계자에게 영향을 미치는 **데이터 상호운용성(interoperability)**, **분류(classification)**, **형식(format)**, **보안(security)** 문제에 중점을 둔다.



DDXF 프레임워크

이 프레임워크는 여러 시스템 간 신뢰(trust) 및 데이터 상호운용성(data interoperability) 요구를 충족하기 위한 일련의 방법(methods) 과 인터페이스(interfaces) 를 제공한다. 이를 통해 다양한 비즈니스 요구와 시나리오 — 특히 데이터 교환(data exchange) 시나리오 — 를 지원할 수 있다.

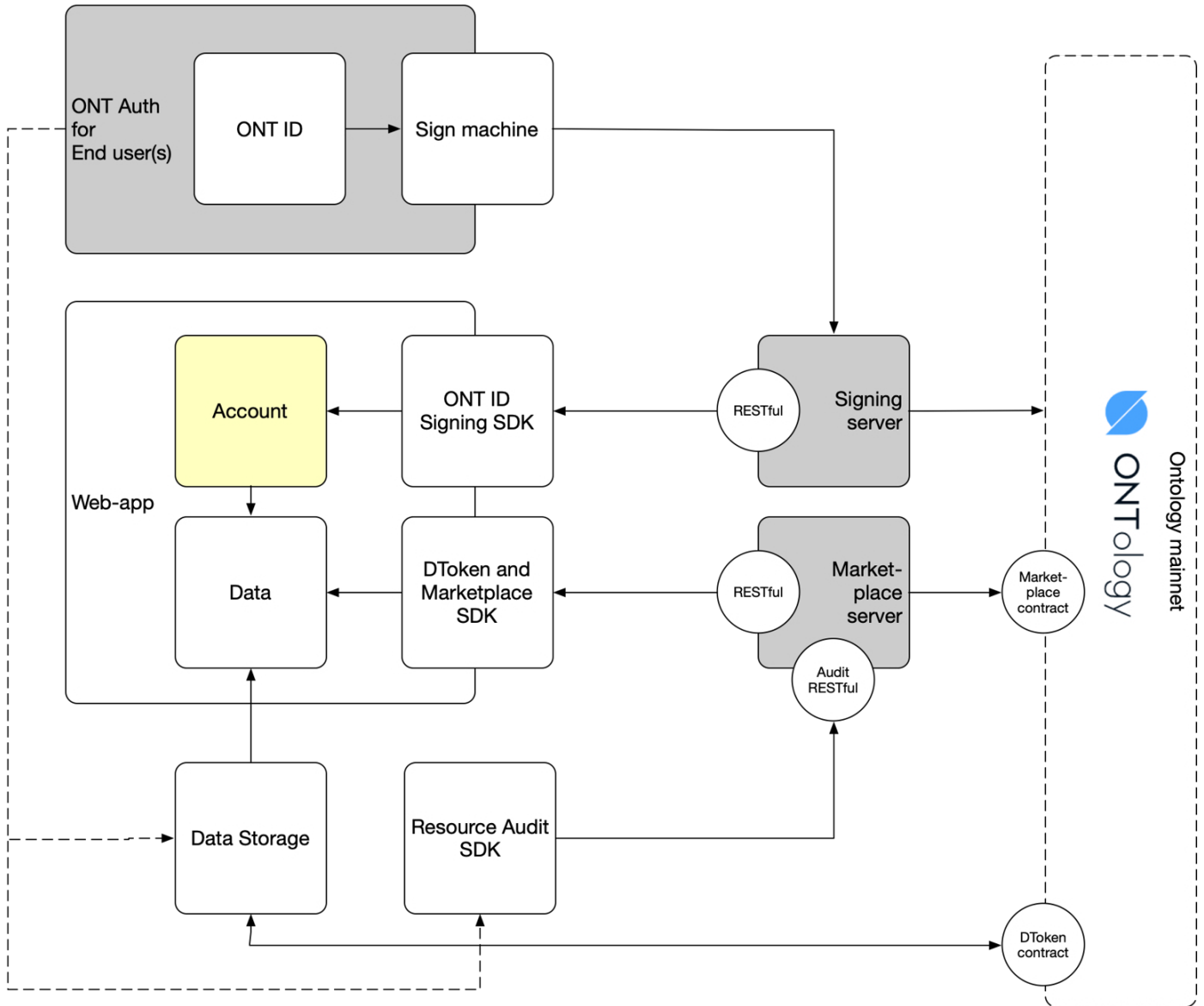
구성요소 및 인터페이스(Components and Interfaces)

DDXF의 다양한 구성요소 및 모듈

구성요소와 인터페이스

DDXF 프레임워크는 다음과 같은 기능을 수행할 수 있도록 지원한다:

- 멀티시스템 오프체인(off-chain) 자원 통합
- 데이터 권한과 접근 권한을 토큰화(tokenize) 하여 크로스체인 자원 및 토큰 거래 수행
- 토큰 전송(token passing) 및 데이터 특성 변화(data characteristics) 형태로 자원 및 데이터 교환 구현
- 오프라인 자원 및 모든 오프체인 중재(off-chain arbitration)에 대해 데이터 처리 및 거래 인증(data processing and transaction authentication) 권한을 지원하여 인센티브 공유(shared incentives)를 보장



DDXF 구성요소

이 프레임워크는 세 가지 기능 모듈을 제공한다.
각각의 모듈을 자세히 살펴보자.

자율적 자원 및 데이터 관리 – DToken 모델(Self-governed Resource and Data Administration – DToken Model)

자기주권적 데이터 관리는 데이터 소유권(data ownership) 과 데이터 접근성(data accessibility) 또는 기타 데이터 처리 권한(data processing privileges) 을 분리(decouple)함으로써, 데이터 소유자와 독립된 수준에서의 데이터 관리가 가능하도록 한다.

DDXF는 다음과 같은 서비스를 결합하여 자기주권적 데이터 관리(data administration) 를 위한 플랫폼을 제공한다:

- **데이터 저장(Data storage):** 보안 기능과 데이터 프라이버시를 갖춘 저장 기능을 비교적 저비용으로 제공한다.
- **ONT ID:** 데이터 자체, 데이터 소유자, 데이터 처리자 등 다양한 주체를 식별하는 데 사용된다.
- **DToken: 데이터 권한 관리(data privilege management)** 를 가능하게 하며, 데이터 처리 및 저장이 온톨로지 블록체인 인프라 위에서 수행되도록 보장한다.
- **온톨로지 블록체인(Ontology blockchain):** 추적성(traceability) 과 신뢰 보증(trust endorsement) 기능을 제공한다.

DToken 모델은 다음과 같은 특징을 정의한다:

- **자원 모델링 및 매핑(Resource modeling and mapping):** 인터넷 또는 **IoT** 시스템의 오프체인 자원을 데이터 모델 형태로 온라인에 저장하고, 해당 온라인 데이터를 온체인 식별자(on-chain identifier)와 매핑하는 정적 모델링 및 매핑 기능을 제공한다.
- **DToken 생성(DToken generation):** 데이터 권한을 **DToken** 형태로 토큰화할 수 있는 정적 식별자 매핑 기능을 제공한다. **DToken**의 사용은 온체인에서 제어되고 기록된다.

DToken 모델은 또한 **DDXF** 내에서 **엔티티(entity)**, **자원(resource)**, **작업(operation)** 을 정의하기 위한 인터페이스를 제공한다.

탈중앙화 데이터 처리 및 자원 교환(Decentralized Data Processing and Resource Exchange)

탈중앙화 데이터 처리 및 관리 모듈은 **자원 교환(resource exchange)**, **데이터 전송(data transfer)**, **데이터 처리(data processing)** 등의 **비즈니스 동작(business actions)** 을 제어하기 위한 런타임 메커니즘을 제공한다.

이 모듈의 두 가지 주요 기능은 다음과 같다:

- **자원 교환(Resource Exchange):** 토큰 기반 자원 전송 및 교환 메커니즘을 제공한다.
- **토큰 활용성(Token Usability):** 특히 DToken을 통한 데이터 접근 제어 등 토큰 사용을 관리하기 위한 메커니즘을 제공한다.

이 모듈은 또한 **DDXF** 내에서 데이터를 처리하고 관리하는 인터페이스를 제공한다.

신뢰 감사(Trust Audit)

DDXF 마켓플레이스는 시스템 간 **자원 교환(cross-system resource exchange)** 기능을 제공한다.

비즈니스 모델의 다양성으로 인해, 데이터는 스마트 디바이스, 최종 사용자, 또는 시스템이 소유할 수 있다. 데이터 처리는 데이터의 형태를 변화시키며, **DDXF**는 데이터 인스턴스의 **소유권(ownership)** 과 **접근성(accessibility)** 을 제어하면서 **데이터 처리, 전송, 교환**을 가능하게 한다.

오프체인 자원과 최종 사용자가 수행하는 거래는 **DDXF**를 위한 **신뢰 보증(trust endorsement)** 역할을 한다.

이 모듈의 두 가지 주요 기능은 다음과 같다:

- **자원 감사(Resource audit):** 오프체인 자원의 **소유권과 접근성**을 보장하기 위한 메커니즘을 제공하며, 여기에는 **자원 품질(resource quality)** 과 **거래 행위(transaction behavior)** 가 포함된다.
- **평판 점수(Reputation score):** 사용 내역과 거래를 기반으로 **DDXF**에 대한 신뢰 보증을 제공한다.

신뢰 감사 모듈(Trust audit module)은 또한 **DDXF** 내 거래, 데이터 자격 증명, 최종 사용자에 대한 신뢰 보증 기능(trust endorsement function)을 구현하기 위한 인터페이스를 제공한다.

GREP

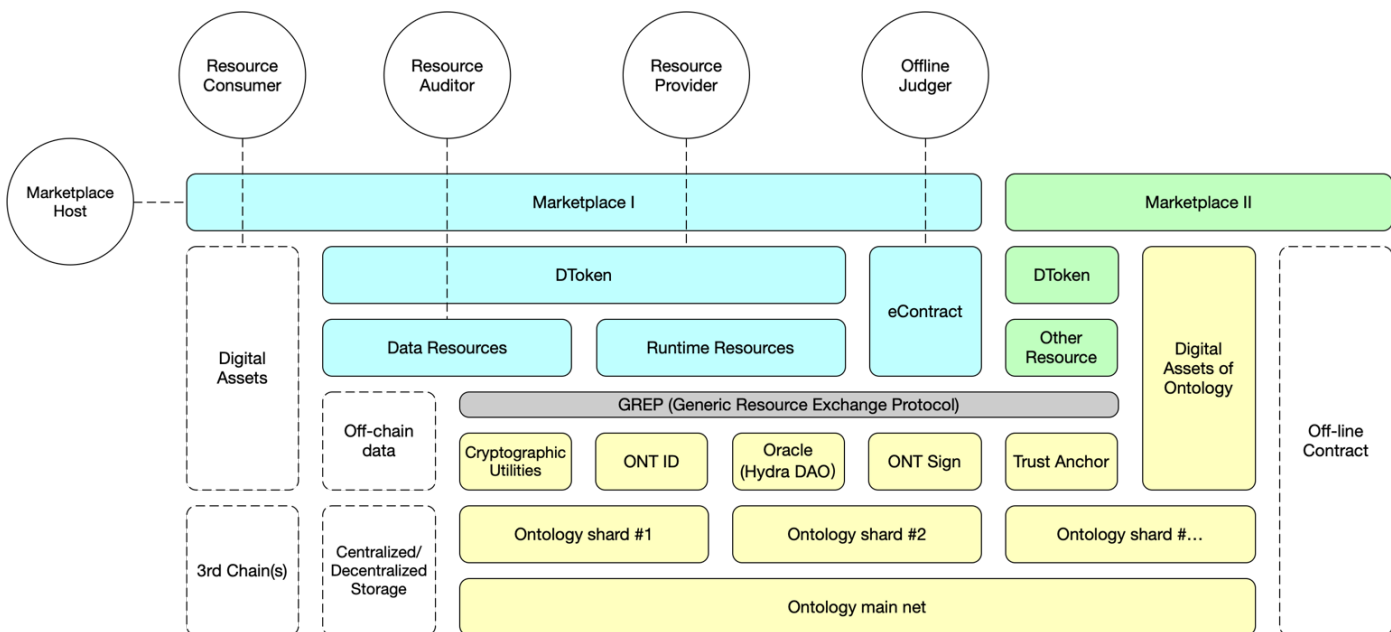
일반 자원 교환 프로토콜 사양(Generic Resource Exchange Protocol Specifications)

초록(Abstract)

일반 자원 교환 프로토콜(GREP, Generic Resources Exchange Protocol)은 온톨로지 메인체인 인프라 위에서 구축된 탈중앙화 자원 교환 프로토콜 세트이다.

GREP을 통해 사용자는 데이터 및 기타 자원을 위한 온체인 증명(attestation) 및 전송 플랫폼을 신속하게 구축할 수 있다.

온톨로지의 포괄적인 신뢰 생태계 인프라—탈중앙화 식별자 **ONT ID**, 탈중앙화 다중 소스 인증 시스템 **Trust Anchor**, 신뢰할 수 있는 오프체인 데이터 커넥터 **Oracle**, 탈중앙 전자 계약 및 서명 시스템 **ONT Sign**—를 기반으로, **GREP**은 탈중앙화 자원 교환을 위한 견고한 신뢰 기반을 제공한다.



GREP 개요도

자원 토큰화 및 자산화(Resource Tokenization and Assetization)

GREP을 통해 누구나 다양한 온체인 자원 증명 및 전송 플랫폼을 빠르고 손쉽게 구축할 수 있다.

이 프로토콜 하에서 자원은 디지털 형태(예: 데이터, CPU 연산력, GPU 연산력, 저장 공간, 온체인 오라클, 신뢰 컴퓨팅 플랫폼)일 수도 있고, 물리적 형태(예: 부동산, 고서화, 회화)일 수도 있다.

플랫폼은 여러 자원이 유통되는 **범용 플랫폼(general-purpose platform)**이거나, 특정 자원의 세분화된 유통을 위한 **특수 목적 교환 플랫폼(specific-purpose exchange platform)**일 수 있다.

자원 유통(resource circulation)은 **ONG, OEP-4 토큰** 등과 교환하거나, **자원 간 직접 교환(resource-for-resource exchange)**의 형태로 이루어질 수 있다.

가능한 자원 전송 형태는 다음과 같다(이에 한정되지 않음):

- **데이터 자원 유통(Data resource circulation)**: 의료 빅데이터(분석 결과)를 ONG와 교환
- **연산 자원 유통(Computing power resource circulation)**: 신뢰된 연산력을 PAX와 교환
- **물리 자원 유통(Physical resource circulation)**: 명화의 소유권을 분할 경매로 거래

자원 유통은 본질적으로 **사용권의 토큰화 및 전송(tokenization and transfer of usage rights)**을 의미한다.

자원에 따라 유통되는 것은 소유권일 수도, 사용권일 수도 있다.

오프체인 실체를 가진 자원은 오프체인에서 인도(delivery)되어야 하며, 인도 방식은 자원의 성격 및 기타 요인에 따라 달라진다.

GREP 내에서 온톨로지 퍼블릭 체인은 탈중앙화된 신뢰 기반을 제공한다.

모든 사용자—자원 제공자, 소비자, 검증자, 오프체인 중재자—는 각각의 **ONT ID**를 생성하고, 마켓플레이스 요구사항에 따라 **KYC 인증**을 수행해야 한다.

자원은 거래 중 반드시 온체인에 등록되어야 하며, 등록 시 자원의 고유 코드가 추출되어 디지털 지문(fingerprint)이 생성되고, 해당 자원에 대한 **ONT ID**가 부여된다.

토큰 기반 교환 메커니즘(Token-based Exchange Mechanism)

데이터 및 자원 교환 과정은 **스마트 계약(smart contract)**을 통해 실행되는 **토큰 교환 및 전송(token exchange and transfer)**으로 간주할 수 있다.

역할(Roles)

GREP은 토큰 교환 과정을 구현하기 위해 다음과 같은 역할을 정의한다:

- **자원 제공자(Resource Provider, RP)**: 자원 관련 권리를 제공 및 이전하며, 이를 보상(예: ONG 또는 기타 자원)과 교환 가능한 형태로 만든다. RP는 자원 소유자일 수도 있고, 데이터 소유자연산력 소유자데이터 수집 플랫폼데이터 관리인 등과 같은 **자원 집계자(resource aggregator)**일 수도

있다.

- **자원 소비자(Resource Consumer, RC):** 특정 유형의 자원이 필요한 자원 제공자의 상대방이다. RC는 (부분적) 소유권 또는 사용권을 얻기 위해 수수료(예: ONG)를 지불한다.
- **자원 인증자(Resource Authenticator, RA):** 자격 및 자원 품질 인증 시스템을 갖춘 제3자 기관으로, 자원 또는 제공자에 대한 검증된 인증을 제공하여 신뢰도를 높인다. 일반적으로 인증된 자원은 더 높은 시장 가치를 갖는다.
- **오프체인 중재자(Off-chain Judge, OJ):** RP와 RC 간 오프체인 분쟁(예: 소비자가 자원을 받지 못한 경우)에 대한 공인 중재자이다.
- **마켓플레이스(Marketplace, MP):** RP와 RC 사이의 중개자 역할을 한다. 자원의 메타정보를 저장하고 유연한 검색 기능을 제공하며, 거래 수수료를 부과한다. 각 마켓플레이스는 고유한 특성을 가질 수 있으며, 여기에는 전자 계약 템플릿, 오프체인 분쟁 해결 도구, 가격 결정 시스템, 규제 투명성을 위한 공개 메커니즘 등이 포함된다.

토큰 거래 절차(Token Transaction Process)

GREP 설계에서 프라이버시는 최우선 고려 사항이다.

GREP은 거래 당사자 모두의 개인 정보 및 거래 데이터 프라이버시를 보호한다.

특히 디지털 자원 및 메타데이터는 **온체인에 업로드되지 않는다**.

또한 GREP은 자원에 대한 **가격 결정(price fixing)** 메커니즘을 제공한다. 가격은 **경매 방식(auction pricing)** 또는 **입찰(bidding)** 등 다양한 방법으로 결정될 수 있다.

일반적인 두 가지 가격 결정 방식은 다음과 같다:

1. **고정 가격(Fixed Pricing):** 자원 제공자가 최초 등록 시 가격을 설정하며, 관심 있는 소비자는 해당 고정 가격으로 직접 구매할 수 있다.

```
{
  pricing: fixed // Pricing method
  price: 10.23 // Price
  currency: ONG // Price unit, e.g. ONT, ONG, etc.
}
```
2. **협상 가격(Negotiatory Pricing):** 자원 제공자가 리소스를 발행할 때 가격을 정의하지 않습니다. 가격은 제공자와 소비자 간의 협상에 의해 결정됩니다. 그런 다음 가격이 설정되고 거래 계약서에 업데이트됩니다.

```
{
  pricing: negotiatory // Pricing method set to negotiatory
}
```

사용자는 자신의 필요에 따라 거래가 이루어질 **마켓플레이스(MP)**를 선택한다.

여러 번 제공 가능한 자원은 여러 MP에서 다양한 방식으로 거래될 수 있다.

예를 들어, 하나의 데이터셋 사용권은 여러 MP에서 거래될 수 있다.

RP, RC, OJ를 포함한 모든 사용자가 해당 MP의 요구사항에 따라 KYC를 완료했다고 가정한다.

자원 유통 과정은 **자원 준비(Resource Preparation)**, **자원 공개(Resource Release)**, **자원 거래(Resource Transaction)**, **이익 분배(Profit Distribution)**, **거래 후 평가(Post-Transaction Review)** 단계로 구성된다.

전체 과정은 다음과 같이 설명된다.

GREP을 통한 자원 및 데이터 교환은 거래 평판(transaction ratings)에 기반한 **평판 시스템(reputation-based system)**을 사용하여, 신뢰 기반의 생태계를 구축한다.

자원 준비(Resource Preparation)

1. **등록(Registration):** **RP**는 자원을 온체인에 등록하고, 자원에 매핑되는 **ONT ID** 및 **ONT ID 문서(Document)**를 생성한다.
2. **인증(Authentication, 선택적):** **RP**는 **RA**로부터 자원에 대한 인증을 획득한다.
3. **가격 결정(Pricing):** **RP**는 **MP**의 가격 시스템에 따라 거래 권리 및 가격 방식을 결정한다.
4. **메타정보 생성(Meta-information generation):** **RP**는 **MP**의 메타정보 템플릿을 사용하여 자원 메타데이터를 생성한다.

자원 공개(Resource Release)

1. **제출(Submission):** **RP**는 자원의 **ONT ID**, 메타데이터, 거래 권한을 **MP**에 제출한다.
2. **정보 처리(Information processing):** **MP**는 온체인 및 자체 데이터베이스에서 해당 자원 관련 데이터를 조회한다.
3. **표시(Display):** **MP**는 자원을 표시하여 **RC**가 원하는 자산을 탐색하고 조회할 수 있도록 한다.

자원 거래(Resource Transaction)

1. **자원 조회(Resource retrieval):**
RC는 자원 메타정보를 기반으로 **MP**에 저장된 필요한 자원을 빠르게 조회하고, 거래를 원하는 자원을 선택한다.
2. **전자 계약 체결(Electronic contract signing, 선택적):**
RP와 **RC**는 **MP**의 전자 계약 템플릿을 사용하여 양 당사자 간 전자 계약을 생성하고, **OJ**를 지정한 뒤 **ONT Sign**을 통해 계약에 서명하며, 이를 거래 스마트 계약에 기록한다.

MP 또는 계약 조건에 따라 **RP**와 **RC**는 분쟁 해결 및 거래 후 이익 분배를 위해 각각 일정량의 **ONG**를 거래 스마트 계약에 예치(stake)해야 할 수 있다.

1. **자원 권리의 토큰화 및 온체인 전송(Tokenization and on-chain transfer of resource rights):**
RP는 전자 계약에 따라 **DToken**을 생성하고, 자원의 일부 소유권 또는 사용권 등 특정 권리를 **RC**에게 부여한다.
2. **오프체인 거래 및 분쟁 중재(Off-chain transaction and dispute arbitration):**
거래가 잠금(lock-in) 기간에 들어가면, **RP**는 **DToken**을 사용하여 자원 인도권과 교환한다.
잠금 기간 중 분쟁이 발생할 경우, 양 당사자는 온체인 또는 오프체인 증거를 제출하며, 오프체인 증거는 **OJ** 또는 **Ontology Oracle**이 판단한다.

이익 분배(Profit Distribution)

잠금 기간이 종료되면 거래 결과에 따라 이익이 분배된다.

중재가 발생할 경우 **OJ** 또는 **Ontology Oracle**이 사전에 이익 분배를 트리거할 수 있다.

거래 후 평가(Post-Transaction Review)

평판 시스템 내에서 **RP**와 **RC**는 서로를 평가(review)하고, 사용자 또는 자원에 평점을 부여할 수 있다. 이 평점은 사용자 평판, 마켓플레이스 내 랭킹, 거래 성공률에 영향을 미친다.

DToken

거래 중 **RP**는 자원 권리를 나타내는 **DToken(스마트 계약)**을 생성한다.

이 DToken에는 자원의 **ONT ID**, 보유자의 **ONT ID**, 유효 기간(validity period)이 포함된다.

DToken은 **균질형(homogeneous)** (예: 부동산 공동 소유권)일 수도 있고, **비균질형(non-homogeneous)** (예: 단일 사용 데이터 접근권)일 수도 있다.

DToken 구조 예시(Example DToken Structure)

```
type DToken struct {
    Name // The name of the DToken
    Symbol // The symbol of the DToken
    Amount // The amount of the DToken
    ResourceID // The ONT ID of the resource corresponding to the resource
    RealContractDigest // The record of the e-contract signed by both parties via ONT Sign
    Expires // Validity period of resource rights
    Exchange // The number of times the DToken can be transferred. The DToken cannot be transferred if the number is set to 0
    Status // The status of the DToken, indicating whether it can be used. The counter method can also be used
}
```

발행 시 **DToken**의 상태는 “Unused(미사용)”이다. 오프체인 접근 권한을 획득하기 위해 사용되면, 그 상태는 “Used(사용됨)”으로 변경된다. 여러 번 사용 가능한 토큰은 감소하는 카운터(Exchange)를 추적하며, 해당 값이 0이 되면 토큰은 만료된다. 접근 권한을 이전하기 전에 **RP**는 DToken의 소유권, 상태, 유효성을 검증한다. 예를 들어, 데이터 접근을 위한 DToken은 **RP**와 **RC**가 서명한 **JWT**를 생성하는 데 사용될 수 있다.

자원 검증 및 감사(Resource Verification and Audit)

소유권과 적법성을 검증하는 등의 오프체인 활동은 실제 환경에서의 행위와 권리를 식별하는 과정을 요구한다. 모든 계약은 탈중앙화 전자 계약(decentralized e-contract)과 **ONT Sign**을 통해 관리되며, 위반 발생 시 계약 조항에 따라 해결 절차가 정의된다.

오프체인 분쟁(예: 자원이 인도되지 않은 경우)에서는 양 당사자가 **ONT Sign** 계약에 서명하며, 그 안에 **OJ**를 통한 중재 절차가 명시된다. **OJ**는 오프체인 중재자로서 판정 결과를 온체인에 제출한다. 분쟁은 온체인 증거를 통해 처리되거나, 필요 시 **Ontology Oracle**을 통해 업로드된 오프체인 증거를 근거로 온체인에서 직접 판결이 내려질 수 있다.

익스텐션(Extension)

GREP은 기술 발전과 생태계 요구에 맞춰 진화하도록 설계된 개방형 공용 프로토콜이다. 이 프로토콜은 가격 기반 자원 교환 및 토큰 결제를 지원한다. 온톨로지의 토큰 자산이 다중 체인에 걸쳐 존재하기 때문에, **GREP**은 본질적으로 **크로스체인 자산 거래 (cross-chain asset transactions)**를 네이티브로 지원한다.

스마트 컨트랙트(Smart Contracts)

논리와 기능의 단위(Units of logic and functionality)



Smart Contract

초록(Abstract)

스마트 컨트랙트(Smart Contract)는 결코 새로운 개념이 아니며, 전통적인 의미에서 블록체인과 직접적으로 연결된 개념도 아니다. 이 용어는 블록체인이 고안되기 훨씬 전인 1990년대에 처음 만들어졌다. 그러나 스마트 컨트랙트가 다양한 블록체인, 특히 **이더리움(Ethereum)** 상에서 일반적인 계산 논리와 메커니즘을 구현하는 데 사용되면서, 이제 블록체인과 분산 원장(distributed ledger)의 맥락에서는 특정 논리를 구현하는 모든 컴퓨터 프로그램이 스마트 컨트랙트로 간주될 수 있다. 온톨로지를 이용한 스마트 컨트랙트 개발 방법에 대한 자세한 내용은 이 링크(<https://ont.io/developer>)를 참조하라.

기능(Functionality)

스마트 컨트랙트(또는 디지털 계약)는 원래 현실 세계의 계약을 모방하기 위해 고안되었다. 하지만 분산 원장과 블록체인이 등장하면서, 스마트 컨트랙트는 금융을 넘어 다양한 산업에서 복잡한 논리와 기능을 실행할 수 있는 기술로 발전했다.

스마트 컨트랙트는 완전하거나 부분적으로 탈중앙화된 블록체인 기술과 결합하여 다음과 같은 방식으로 활용될 수 있다:

- 지식재산권(Intellectual Property, IP) 보호
- 공급망 관리(Supply Chain Management)
- 연구 및 개발(Research and Development)
- 법률 산업(Legal Industry)
- 공공 부문(Government Sectors)

스마트 컨트랙트는 미리 정해진 계약 조항이 충족되거나 충족되지 않을 때, 비즈니스 논리와 시스템 아키텍처에 따라 하나 또는 여러 체인에서 자산을 자동으로 이전한다. 여기서 주목해야 할 점은, 이러한 조항과 모든 과정이 **공개적으로 확인 가능**하다는 것이다. 이는 기존 시스템에서 문제로 지적되던 **불투명성, 책임성, 역할 관리의 어려움**을 극복할 수 있는 새로운 수준의 투명성을 제공한다. 경쟁이 심화되고 블록체인 기술이 발전함에 따라, 스마트 컨트랙트는 더욱 큰 역할을 하게 될 것이다.

온톨로지 스마트 컨트랙트(Ontology Smart Contracts)는 **고확장성(High Scalability)**, **고성능(High Performance)**, **다중 언어 지원(Multilingual Contracts)**, **크로스체인(Cross-chain)**, **크로스 가상 머신(Cross-virtual-machine)** 등의 동적 기능을 하나의 시스템에 통합하여 제공한다.

현재 지원되는 언어는 **Python, C#, Rust**이며, 향후 **Java, C++, Go, JavaScript** 지원이 추가될 예정이다.

또한 온톨로지는 **Solidity**로 작성된 **EVM 컨트랙트(EVM Contracts)**도 지원한다.

보안 조치(Security Measures)

스마트 컨트랙트는 **투명성(Transparency)**, **정확성(Precision)**, **비용 효율성(Cost-effectiveness)**, **실용성(Practicality)** 등의 장점을 지닌다.

스마트 컨트랙트는 특정 자산이 미리 정해진 계약 조건이 충족될 때만 이전되도록 보장함으로써, 안전장치로서의 역할을 수행한다.

그러나 이러한 투명성은 동시에 보안 취약성을 초래할 수 있다.

계약 조항뿐 아니라, 코드 내의 문제나 취약점 역시 공개되기 때문이다.

스마트 컨트랙트는 한 번 배포되면 수정이 불가능하므로,

2016년 DAO 공격(DAO Attack)과 같은 사고를 방지하기 위해

논리를 면밀히 검토하고 현실적인 시나리오에서 철저히 테스트해야 한다.

현재는 이러한 위험을 완화하기 위한 여러 스마트 컨트랙트 감사(auditing) 서비스가 존재한다.

다음은 안전하고 견고한 스마트 컨트랙트를 작성하기 위한 몇 가지 지침이다:

- 사용 언어의 한계를 피하기 위해 일반적인 프로그래밍 모범 사례를 따른다.

- 일반적인 논리 대신, 탈중앙화를 고려한 블록체인 전용 코드를 작성한다.
- 철저한 테스트를 반복적으로 수행하여 다양한 상황에서 컨트랙트가 의도한 대로 작동함을 보장한다.
- 기능을 필수적인 수준으로 제한한다. 복잡한 기능은 일반적으로 더 많은 취약점을 유발한다.

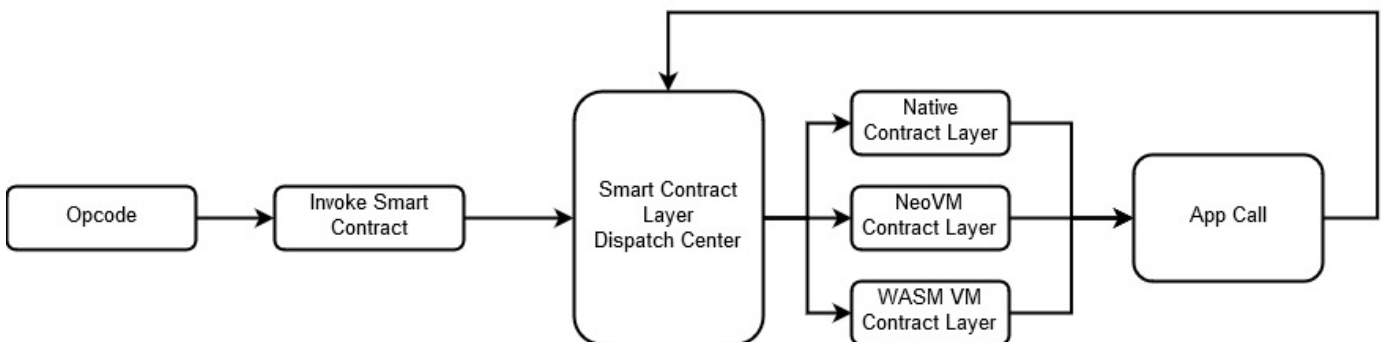
Smart Contract Development(<https://ont.io/developer>)

스마트 컨트랙트의 유형(Types of Smart Contracts)

온톨로지의 다양한 스마트 컨트랙트 유형

온톨로지에는 서로 다른 실행 엔진을 사용하는 세 가지 유형의 스마트 컨트랙트가 있으며, 이들은 모두 디스패치 센터(dispatch center)를 통해 호출된다:

1. **네이티브 컨트랙트(Native Contracts):**
온톨로지의 ONT 및 ONG 컨트랙트로, 모든 OEP 토큰 프로토콜과 그 기능을 관리한다.
2. **NeoVM 컨트랙트(NeoVM Contracts):**
AVM 바이트코드(AVM bytecode)로 컴파일되어 **NeoVM 엔진**에서 실행된다.
현재 **Python** 및 **C#** 언어를 지원한다.
3. **WebAssembly (WASM) VM 컨트랙트:**
이식 가능한 바이너리 코드 형식으로 컴파일되어 **WebAssembly 엔진**에서 실행된다.
현재 **Rust** 및 **C++** 언어를 지원한다.



스마트 컨트랙트 호출 메커니즘(Smart Contract Invocation Mechanism)

스마트 컨트랙트가 **AVM 바이트코드**로 컴파일되고 스택 **opcode**가 생성되면, 해당 컨트랙트는 관련 정보와 함께 온체인에 전송된다. 이 트랜잭션은 일정량의 가스를 소모하며, 이 시점에서 컨트랙트는 온체인에 배포된 상태가 된다.

그 후, 스마트 컨트랙트가 호출되면 디스패치 센터가 컨트랙트의 유형을 식별한다. 세 가지 컨트랙트 유형은 각각 **별도의 엔진에서 실행**되며, 제어 흐름이 해당 컨트랙트 계층으로 전달된다.

이후의 모든 호출도 동일한 절차를 따른다.

EVM 컨트랙트(EVM Contracts)

이더리움 가상 머신(EVM, **Ethereum Virtual Machine**) 환경에서 실행되는 Solidity 스마트 컨트랙트 또한 온톨로지에서 배포할 수 있다.

MetaMask, web3.js, Truffle, Hardhat 등의 개발 및 테스트 도구는 온톨로지 테스트넷과 메인넷 모두에서 호환된다. Solidity 컨트랙트는 바이트코드로 컴파일되어 EVM 오피코드(EVM opcode) 형식으로 실행된다.

이제 스마트 컨트랙트의 개념, 활용 가능성, 그리고 그것이 **온톨로지(Ontology)** 플랫폼에서 어떻게 실행되는지 이해했을 것이다.

스마트 컨트랙트 개발 가이드는 다음 링크를 참고하라: <https://ont.io/developer>

합의 메커니즘 (Consensus Mechanism)

온톨로지의 합의 네트워크와 메커니즘

소개 (Introduction)

검증 가능한 비잔틴 장애 허용(Verifiable Byzantine Fault Tolerance, VBFT) 은 온톨로지 합의 엔진(Ontology Consensus Engine, OCE) 의 핵심 합의 알고리즘이다.

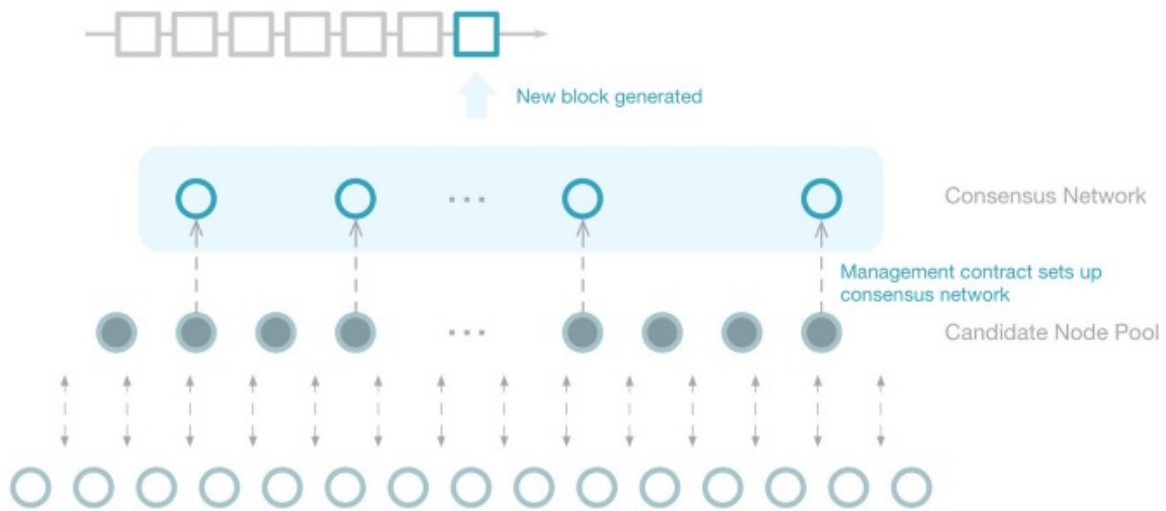
이는 지분증명(Proof of Stake, PoS), 검증 가능한 난수 함수(Verifiable Random Function, VRF), 그리고 비잔틴 장애 허용(Byzantine Fault Tolerance, BFT) 알고리즘을 결합한 하이브리드 알고리즘이다. VBFT는 합의 그룹의 확장을 지원하며, VRF는 합의 그룹 내 무작위성과 공정성을 보장하면서 빠른 최종 합의(final consensus)에 도달하도록 돕는다. 합의 네트워크는 근본적으로 두 부분으로 구성된다.

합의 네트워크 (Consensus Network)

합의 네트워크는 온톨로지 네트워크 내에서 발생하는 이벤트에 대해 합의 과정을 수행하는 모든 합의 노드 (consensus node)로 구성된다. 여기에는 블록 생성, 원장 일관성 유지, 그리고 합의 과정을 통과한 블록을 동기화 노드 네트워크로 전송하는 과정이 포함된다.

합의 후보 네트워크 (Consensus Candidate Network)

후보 노드(candidate node)는 합의 과정에 직접 참여하지 않지만, 합의 노드의 상태를 실시간으로 동기화 하며 자신이 유지하는 원장에 새 블록 정보를 갱신한다. 후보 노드들은 합의 노드와 그 상태를 지속적으로 모니터링하며, 합의 블록을 검증하고 온톨로지 네트워크의 운영을 지원한다.



후보 노드 풀(Candidate Node Pool)

합의 네트워크의 규모는 합의 스마트 컨트랙트를 통해 관리된다. 합의 네트워크의 각 노드는 노드 관리자가 설정한 스테이크(stake)를 보유한다.

합의 네트워크 구축 (Building a Consensus Network)

합의 네트워크는 합의 관리 스마트 컨트랙트(consensus management smart contracts)를 통해 구축된다. 합의 관리 컨트랙트는 온톨로지 네트워크 상에 영구적으로 배포되어 실행되며, 정기적으로 합의 노드 목록과 VBFT 알고리즘의 구성 파라미터를 갱신한다. PoS 테이블은 VBFT의 중요한 파라미터 중 하나이다. VBFT가 실행되면, 모든 노드는 무작위로 합의에 참여할 노드를 선택하여 한 라운드의 합의 과정을 수행한다.

알고리즘 개요 (Algorithm Overview)

VBFT 알고리즘은 전통적인 비잔틴 장애 허용(BFT) 알고리즘을 개선한 형태로, 무작위성을 검증 가능하게 만든다. 한 라운드의 합의에서는 VRF를 기반으로 순차적으로 블록 제안 노드 세트가 선택된다. 노드 세트는 블록 검증 노드 세트와 블록 확정 노드 세트로 구분되며, 이 선택에 따라 합의 과정이 수행된다. VRF로 인한 무작위성 때문에 제안, 검증, 확정 노드 세트는 매 라운드마다 달라지며, 이를 예측하기 어렵기 때문에 악의적 공격에 대한 저항성이 높다. 알고리즘은 다음과 같이 요약할 수 있다.

1. VRF를 기반으로 합의 네트워크에서 교대로 제안 노드를 선택한다. 각 제안 노드는 블록을 제안한다.
2. VRF를 기반으로 검증 노드를 선택한다. 각 검증 노드는 제안된 블록을 수집하고 검증한 뒤, 가장 높은 우선순위를 가진 블록에 투표한다.
3. VRF를 기반으로 확정 노드를 선택한다. 이 노드들은 검증 노드의 투표를 통계적으로 검증하고 최종 합의 결과를 확정한다.
4. 모든 합의 노드는 확정 노드가 산출한 결과를 수용한다. 이렇게 한 라운드의 합의가 종료되며, 이후 새

로운 합의 라운드가 시작된다.

검증 가능한 난수 함수 (Verifiable Random Function)

현재 VBFT 알고리즘에서는 각 블록의 VRF 값이 이전 합의 블록에 의해 결정된다. 알고리즘은 이전 블록의 트랜잭션 정보를 가져와 1024자리 해시 값을 계산하고, 이를 다음 블록의 VRF 값으로 사용한다.

노드 선택 (Node Selection)

VBFT 알고리즘은 이전 합의 라운드의 VRF 값을 인덱스로 사용하여 다음 합의 라운드에 참여할 노드를 PoS 테이블에서 결정한다. PoS 테이블 생성 시 각 노드 운영자의 지분 정보와 합의 네트워크의 거버넌스 정책이 반영된다. VRF 값은 균등하게 분포되지만, 무작위 노드 선택 과정은 온톨로지의 합의 네트워크 관리 정책의 영향을 받는다. 블록이 분기되지 않는 한, 각 블록의 VRF 값은 검증 가능하므로 특정 블록 높이에서 모든 노드는 동일한 VRF 값을 공유한다. 알고리즘 내 VRF는 PoS 테이블에서 노드를 순차적으로 선택하며, 각 VRF 값은 대체 제안 노드 시퀀스를 유지한다. 이 무작위 시퀀스는 합의 과정과 일관된다.

포크 선택 (Fork Selection)

온톨로지는 퍼블릭 네트워크에서 운영되는 퍼블릭 체인이므로, 악의적 공격이나 네트워크 장애와 같은 공용망 특유의 문제에 직면할 수 있다. VBFT 합의 알고리즘은 합의 노드를 무작위로 선택하여 공격의 난이도를 높이지만, 네트워크 분리 시 포크(fork)가 발생할 가능성은 여전히 존재한다. 앞서 언급했듯이 각 블록의 VRF는 노드 시퀀스를 결정하므로, VBFT가 포크 선택을 수행할 때 노드의 우선순위를 해당 시퀀스에 따라 부여한다. 이후 각 포크의 우선순위를 가중치로 평가하고, 각 노드는 이 가중 우선순위에 따라 포크를 선택한다. VRF가 생성한 우선순위에 따라 블록이 선택되기 때문에 악의적 포크가 높은 우선순위를 유지하는 것은 매우 어렵다. 따라서 악의적 포크는 빠르게 사라지며, VBFT는 빠른 최종 합의(finality)를 보장한다.

자동 구성 (Automatic Configuration)

합의 네트워크의 품질을 유지하기 위해 온톨로지의 합의 관리 컨트랙트는 노드 목록을 자동으로 갱신한다. 네트워크 이상이 발생하면 합의 관리 컨트랙트는 스테이크 기반 투표를 통해 합의 네트워크의 노드 리스트를 갱신한다. 새로운 노드가 더 많은 스테이크를 확보하고 성능이 합의 네트워크의 요구 사항을 충족하는 것이 확인되면, 노드 리스트가 갱신될 때 합의 네트워크에 추가된다. 합의 네트워크 갱신 주기는 블록 단위로 관리된다. 합의 네트워크가 일정 개수의 블록 합의를 완료할 때마다 다음 블록의 제안 노드는 관리 컨트랙트 실행 이벤트를 생성하여 이를 첫 번째 이벤트로 포함하고 블록을 제안해야 한다. 검증 및 확정 노드는 이 블록의 유효성을 검증하고, 합의를 통해 해당 블록이 승인되면 네트워크의 모든 노드가 관리 컨트랙트를 실행하여 노드 리스트를 업데이트한다. 이 과정을 통해 합의 네트워크 전체의 노드 목록이 갱신된다.

비교표 (Comparison Table)

합의 메커니즘	적용 시나리오	효율성	합의 확정 속도	합의 확인 샘플 시간	노드 수	악의적 노드 허용 한계	자원 소모율	보안 제어
POW	퍼블릭 블록체인	< 20 tps	느림	비트코인: 60분, 이더리움: 1분	-	50%	높음	낮음
DPOS	퍼블릭 블록체인	> 500 tps	중간	Bitstocks: 10초	< 30	불명	낮음	높음
PBFT	연합/프라이빗 블록체인	> 1000 tps	빠름	FISCO-BCOS: 1초, Fabric: 1초	< 30	합의 노드의 1/3 이하	낮음	높음
VBFT	퍼블릭/프라이빗 블록체인	> 3000 tps	중간	Ontology Testnet: 5-10초	< 1000	구성 가능한 BFT 한계(합의 노드의 1/3 이하)	낮음	높음
Paxos/RAFT	연합/프라이빗 블록체인	> 5000 tps	빠름	FISCO-BCOS: 1초	< 30	없음	낮음	높음

온톨로지 오라클 (Ontology Oracle)

오라클 네트워크와 메커니즘

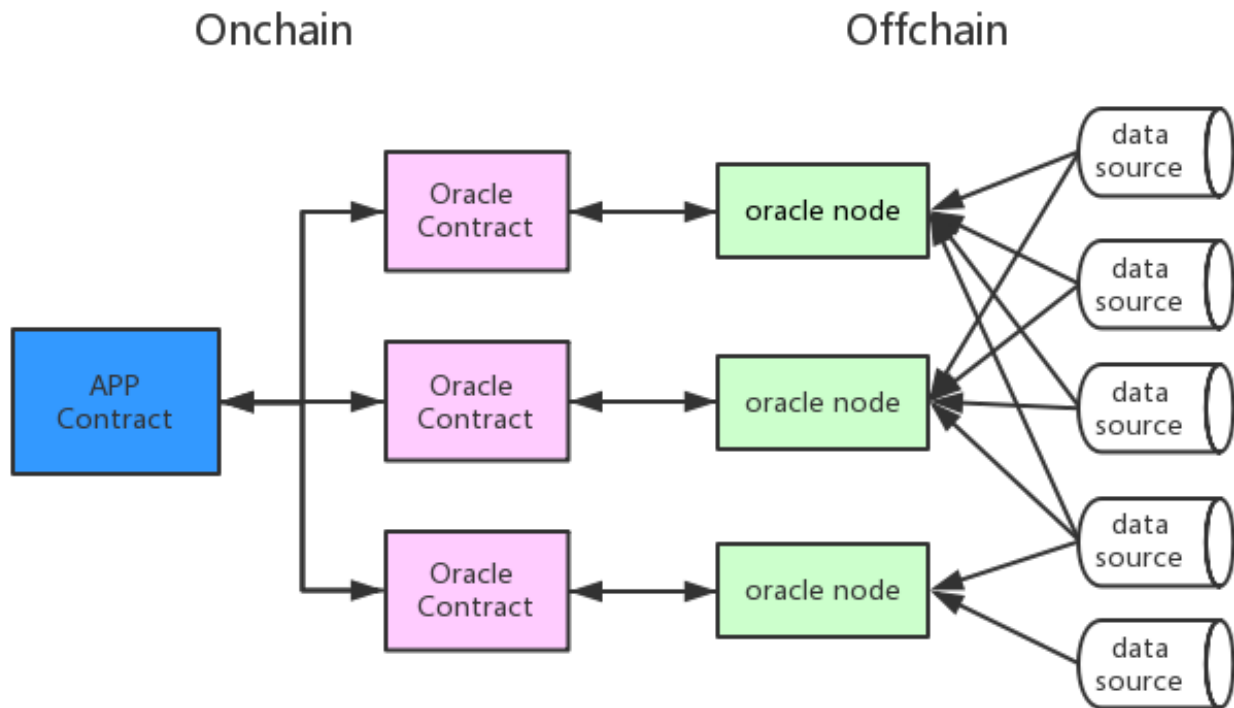
점점 더 많은 dApp들이 외부 세계의 트리거에 의존하게 되고 있다. 다음은 하나의 가상 시나리오이다: 한 항공편이 오전 10시에 도착할 예정으로 설정되어 있다. 항공편이 지연될 경우 자동으로 발동되는 보험 스마트 컨트랙트가 설정되어 있으며, 보험의 모든 수혜자는 보상으로 100개의 토큰을 지급받게 된다. 이러한 시나리오에는 다양한 변수가 존재하며, 이러한 스마트 컨트랙트를 구현하는 dApp은 실제 거래를 수행하기 위해 항공편 정보, 보험 정책 관련 데이터, 계정 정보 등 다수의 외부 데이터를 필요로 한다.

dApp들이 점점 더 많은 기능을 포함하게 됨에 따라 물류 정보, 주가, 날씨 데이터, 스포츠 통계 및 경기 결과 등 더 많은 현실 세계의 데이터가 필요하게 되었다.

온톨로지 오라클(Ontology Oracle)은 이러한 문제 — 즉, 스마트 컨트랙트가 외부 세계와 직접 상호작용할 수 없다는 문제 — 를 해결하기 위해 설계되었다.

오라클은 본질적으로 **데이터 전달자(data transporter)**의 역할을 하며, 스마트 컨트랙트가 외부 데이터를 가져올 수 있도록 한다.

온톨로지 오라클 프레임워크 (Ontology Oracle Framework)



오라클 인프라(Oracle infrastructure)

위의 그림과 같이 인프라는 두 개의 주요 구성 요소로 이루어져 있다:

1. 온체인(On-chain)
2. 오프체인(Off-chain)

오프체인 (Oracle Operator and Data Source)

오라클 노드(Oracle node)와 데이터 소스(data source)는 오프체인에 존재하는 오라클 네트워크의 두 구성 요소이다.

이 노드들은 온톨로지 네트워크에 연결되어 있으며, 오라클 컨트랙트의 요청을 수신 대기한다.

모든 노드는 데이터 요청을 독립적으로 처리한다.

이들 노드는 향후 더 많은 블록체인 네트워크를 지원할 예정이다.

노드가 수행하는 작업은 두 가지 주요 과정으로 구성된다:

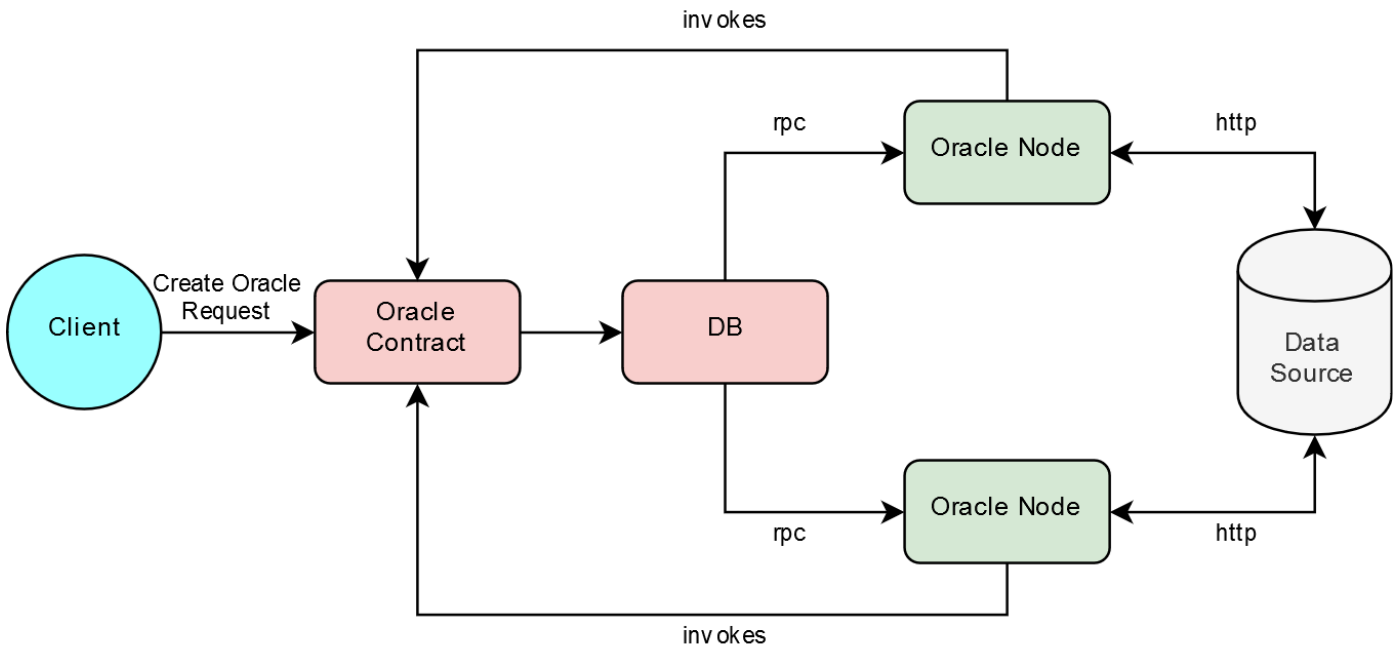
- 데이터 수집(Data crawling)
- 데이터 분석(Data analysis)

노드는 외부 API를 통해 데이터 소스로부터 정보를 가져오고, 응답을 처리 및 분석한 뒤, 사용자가 지정한 형식으로 직렬화(serialization)하여 그 데이터를 오라클 컨트랙트에 기록한다.

온체인 (Oracle Contract)

오라클 컨트랙트는 주로 노드가 전송한 데이터를 수집하고 저장하며, 이를 다른 스마트 컨트랙트가 호출(involve)하고 접근(access)할 수 있도록 한다.

오라클 데이터 및 워크플로 (Oracle Data and Workflow)



오라클 네트워크 데이터 흐름(Oracle network data flow)

위의 그림에서와 같이, 오라클 네트워크 내 데이터 흐름은 다음과 같이 요약된다:

1. 클라이언트는 오라클 데이터 요청(Oracle data request)을 통해 오라클 컨트랙트에 데이터 요구 사항을 전달한다.
2. 오라클 컨트랙트는 수신한 정보를 데이터베이스처럼 동작하는 원장(ledger)에 기록한다.
3. 오라클 노드는 이 요청을 감지하고 RPC 인터페이스를 통해 데이터를 가져온다.
4. 노드는 외부 데이터 소스로부터 HTTP API를 이용해 데이터를 조회한다.
5. 노드는 앞서 받은 클라이언트 요청에 따라 데이터를 직렬화한 후, 오라클 컨트랙트를 호출하여 데이터를 전송한다.
6. 이후 클라이언트 측의 다른 스마트 컨트랙트들은 이 오라클 컨트랙트를 호출하여 해당 데이터에 접근할 수 있다.

온톨로지 노드

초록 (Abstract)

온톨로지 노드 소개 (An introduction to Ontology nodes)

온톨로지 노드 (Ontology Nodes)

네트워크의 유형이 다양한 것처럼, 노드의 유형 또한 여러 가지가 있다. 아래는 그 목록이다.

- **트라이오네스 노드(Triones node) | 온톨로지 메인넷(Ontology MainNet)**
 - 트라이오네스 노드는 네트워크 합의에 참여하는 장부 기록(bookkeeping) 노드이다. 합의 노드로 신청한 후, 사용자가 ONT를 스테이킹(staking)하여 투표를 통해 선출된다. 온톨로지 합의(Consensus)에 대한 자세한 정보는 <https://ont.io/> 에서 확인할 수 있다.
- **동기화 노드(Synchronization node) | 온톨로지 메인넷 / 폴라리스 테스트넷(Ontology MainNet / Polaris TestNet)**
 - 동기화 노드는 네트워크 합의에 참여하지 않으며, 트라이오네스(장부 기록 노드가 생성한 블록을 동기화하는 역할만 수행한다.
- **단일 노드(Single node) | 프라이빗 테스트넷(Private TestNet)**
 - 온톨로지는 프라이빗 테스트 환경에서의 개발을 위해 단일 노드 네트워크 배포를 지원한다.

온톨로지 CLI (Ontology CLI)

온톨로지 네트워크 노드를 생성, 관리, 상호작용하기 위한 완전한 기능을 갖춘 명령줄 인터페이스(Command Line Interface, CLI).

주요 기능 (Features)

- 온톨로지 메인넷, 폴라리스 테스트넷, 또는 프라이빗 테스트넷에서 노드 설정
- 노드의 RPC, REST, WebSocket 하위 서비스를 관리
- 네트워크 가스비(gas fee)와 블록당 트랜잭션 한도 등 노드 파라미터 설정
- 온톨로지 메인넷 합의 노드 실행 및 관리
- 계정 및 자산 관리
- 다중 서명(Multi-Signature) 계정 생성 및 트랜잭션 서명
- 블록 및 트랜잭션 정보 조회
- 스마트 컨트랙트 배포 및 호출

네트워크 환경 (Network Environments)

온톨로지 블록체인 네트워크는 공개 운영 환경(public live environment)과 공개 테스트 환경(public test environment)으로 구성된다.

또한 개발자는 프라이빗 테스트 환경에서 노드를 배포할 수도 있다.

- **온톨로지(Ontology, MainNet):** 공개 운영 환경
- **폴라리스(Polaris, TestNet):** 공개 테스트 환경
- **프라이빗(Private, TestNet):** 단일 노드 네트워크 및 프라이빗 테스트 환경