Appendix A: Ethereum Standards

Ethereum Improvement Proposals (EIPs)

The Ethereum Improvement Proposal repository is located at https://github.com/ethereum/EIPs/. The workflow is illustrated in Ethereum Improvement Proposal workflow.

From EIP-1:

> EIP stands for Ethereum Improvement Proposal. An EIP is a design document providing information to the Ethereum community, or describing a new feature for Ethereum or its processes or environment. The EIP should provide a concise technical specification of the feature and a rationale for the feature. The EIP author is responsible for building consensus within the community and documenting dissenting opinions.

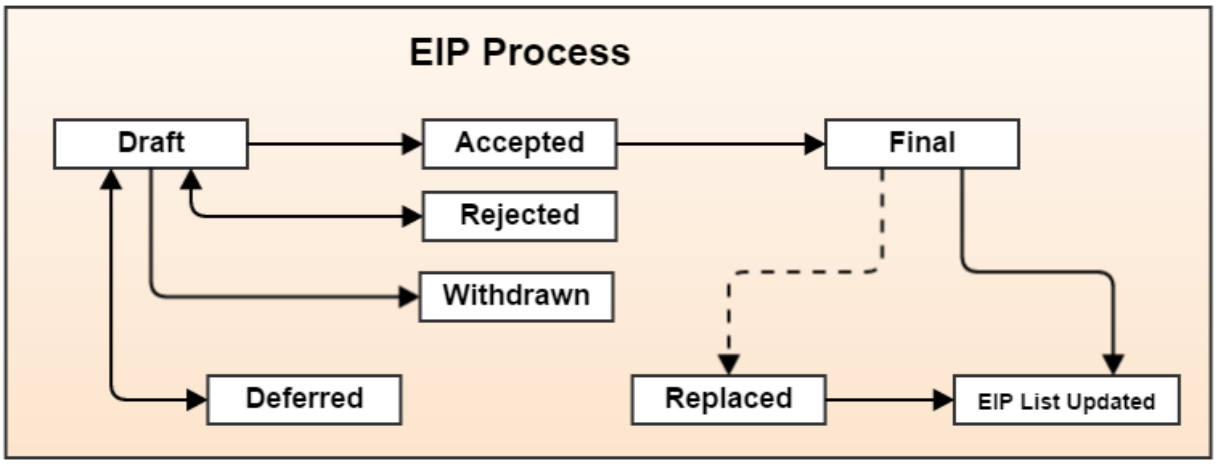

Figure 1. Ethereum Improvement Proposal workflow

Table of Most Important EIPs and ERCs

Table 1. Important EIPs and ERCs

EIP/ERC #	Title/Description	Author	Layer	Status	Created
EIP-1	EIP Purpose and Guidelines	Martin Becze, Hudson Jameson	Meta	Final	
EIP-2	Homestead Hard-fork Changes	Vitalik Buterin	Core	Final	

EIP/ERC #	Title/Description	Author	Layer	Status	Created
EIP-5	Gas Usage for `RETURN` and `CALL*`	Christian Reitwiessner	Core	Draft	
EIP-6	Renaming SUICIDE Opcode	Hudson Jameson	Interface	Final	
EIP-7	`DELEGATECALL`	Vitalik Buterin	Core	Final	
EIP-8	devp2p Forward Compatibility Requirements for Homestead	Felix Lange	Networking	Final	
EIP-20	ERC-20 Token Standard. Describes standard functions a token contract may implement to allow DApps and wallets to handle tokens across multiple interfaces/DApps. Methods include: `totalSupply`, balanceOf(address), `transfer`, `transferFrom`, `approve`, `allowance`. Events include: `Transfer` (triggered when tokens are transferred), `Approval` (triggered when `approve` is called).	Fabian Vogelsteller, Vitalik Buterin	ERC	Final	Frontier

EIP/ERC #	Title/Description	Author	Layer	Status	Created
EIP-55	Mixed-case checksum address encoding	Vitalik Buterin	ERC	Final	

EIP/ERC #	Title/Description	Author	Layer	Status	Created
EIP-86	Abstraction of transaction origin and signature. Sets the stage for "abstracting out" account security and allowing users to create "account contracts," moving toward a model where in the long term all accounts are contracts that can pay for gas, and users are free to define their own security models that perform any desired signature verification and nonce checks (instead of using the in-protocol mechanism where ECDSA and the default nonce scheme are the only "standard" way to secure an account, which is currently hardcoded into transaction processing).	Vitalik Buterin	Core	Deferred (to be replaced)	Constantinople

EIP/ERC #	Title/Description	Author	Layer	Status	Created
EIP-96	Blockhash and state root changes. Stores blockhashes in the state to reduce protocol complexity and need for complex client implementations to process the `BLOCKHASH` opcode. Extends range of how far back blockhash checking may go, with the side effect of creating direct links between blocks with very distant block numbers to facilitate much more efficient initial light client syncing.	Vitalik Buterin	Core	Deferred	Constantinople
EIP-100	Change difficulty adjustment to target mean block time and including uncles.	Vitalik Buterin	Core	Final	Metropolis Byzantinium

EIP/ERC #	Title/Description	Author	Layer	Status	Created
EIP-101	Serenity Currency and Crypto Abstraction. Abstracts ether up a level with the benefit of allowing ether and subtokens to be treated similarly by contracts, reduces the level of indirection required for custom-policy accounts such as multisigs, and purifies the underlying Ethereum protocol by reducing the minimal consensus implementation complexity.	Vitalik Buterin	Active	Serenity feature	Serenity Casper

EIP/ERC #	Title/Description	Author	Layer	Status	Created
EIP-105	Binary sharding plus contract calling semantics. "Sharding scaffolding" EIP to allow Ethereum transactions to be parallelized using a binary tree sharding mechanism, and to set the stage for a later sharding scheme. Research in progress; see https://github.com/ethereum/sharding.	Vitalik Buterin	Active	Serenity feature	Serenity Casper
EIP-137	Ethereum Domain Name Service - Specification	Nick Johnson	ERC	Final	

EIP/ERC #	Title/Description	Author	Layer	Status	Created
EIP-140	New Opcode: REVERT. Adds `REVERT` opcode instruction, which stops execution and rolls back the EVM execution state changes without consuming all provided gas (instead the contract only has to pay for memory) or losing logs, and returns to the caller a pointer to the memory location with the error code or message.	Alex Beregszaszi, Nikolai Mushegian	Core	Final	Metropolis Byzantinium
EIP-141	Designated invalid EVM instruction	Alex Beregszaszi	Core	Final	
EIP-145	Bitwise shifting instructions in EVM	Alex Beregszaszi, Paweł Bylica	Core	Deferred	
EIP-150	Gas cost changes for IO-heavy operations	Vitalik Buterin	Core	Final	

EIP/ERC #	Title/Description	Author	Layer	Status	Created
EIP-155	Simple replay attack protection. Replay Attack allows any transaction using a pre-EIP-155 Ethereum node or client to become signed so it is valid and executed on both the Ethereum and Ethereum Classic chains.	Vitalik Buterin	Core	Final	Homestead
EIP-158	State clearing	Vitalik Buterin	Core	Superseded	
EIP-160	EXP cost increase	Vitalik Buterin	Core	Final	
EIP-161	State trie clearing (invariant-preserving alternative)	Gavin Wood	Core	Final	
EIP-162	Initial ENS Hash Registrar	Maurelian, Nick Johnson, Alex Van de Sande	ERC	Final	
EIP-165	ERC-165 Standard Interface Detection	Christian Reitwiessner et al.	Interface	Draft	
EIP-170	Contract code size limit	Vitalik Buterin	Core	Final	
EIP-181	ENS support for reverse resolution of Ethereum addresses	Nick Johnson	ERC	Final	

EIP/ERC #	Title/Description	Author	Layer	Status	Created
EIP-190	Ethereum Smart Contract Packaging Standard	Piper Merriam et al.	ERC	Final	
EIP-196	Precompiled contracts for addition and scalar multiplication on the elliptic curve alt_bn128. Required in order to perform zkSNARK verification within the block gas limit.	Christian Reitwiessner	Core	Final	Metropolis Byzantinium
EIP-197	Precompiled contracts for optimal ate pairing check on the elliptic curve alt_bn128. Combined with EIP-196.	Vitalik Buterin, Christian Reitwiessner	Core	Final	Metropolis Byzantinium
EIP-198	Big integer modular exponentiation. Precompile enabling RSA signature verification and other cryptographic applications.	Vitalik Buterin	Core	Final	Metropolis Byzantinium

EIP/ERC #	Title/Description	Author	Layer	Status	Created
EIP-211	New opcodes: `RETURNDATASIZE` and `RETURNDATACOPY`. Adds support for returning variable-length values inside the EVM with simple gas charging and minimal change to calling opcodes using new opcodes `RETURNDATASIZE` and `RETURNDATACOPY`. Handles similar to existing `calldata`, whereby after a call, return data is kept inside a virtual buffer from which the caller can copy it (or parts thereof) into memory, and upon the next call, the buffer is overwritten.	Christian Reitwiessner	Core	Final	Metropolis Byzantinium

EIP/ERC #	Title/Description	Author	Layer	Status	Created
EIP-214	New opcode: `STATICCALL`. Permits non-state-changing calls to itself or other contracts while disallowing any modifications to state during the call (and its subcalls, if present) to increase smart contract security and assure developers that re-entrancy bugs cannot arise from the call. Calls the child with `STATIC` flag set to `true` for execution of child, causing exception to be thrown upon any attempts to make state-changing operations inside an execution instance where `STATIC` is `true`, and resets flag once call returns.	Vitalik Buterin, Christian Reitwiessner	Core	Final	Metropolis Byzantinium

EIP/ERC #	Title/Description	Author	Layer	Status	Created
EIP-225	Rinkeby testnet using proof of authority where blocks are only mined by trusted signers.	Péter Szilágyi			Homestead
EIP-234	Add `blockHash` to JSON-RPC filter options	Micah Zoltu	Interface	Draft	
EIP-615	Subroutines and Static Jumps for the EVM	Greg Colvin, Paweł Bylica, Christian Reitwiessner	Core	Draft	
EIP-616	SIMD Operations for the EVM	Greg Colvin	Core	Draft	
EIP-681	URL Format for Transaction Requests	Daniel A. Nagy	Interface	Draft	
EIP-649	Metropolis Difficulty Bomb Delay and Block Reward Reduction. Delayed the Ice Age (aka Difficulty Bomb) by 1 year, and reduced the block reward from 5 to 3 ether.	Afri Schoedon, Vitalik Buterin	Core	Final	Metropolis Byzantinium

EIP/ERC #	Title/Description	Author	Layer	Status	Created
EIP-658	Embedding transaction status code in receipts. Fetches and embeds a status field indicative of success or failure state to transaction receipts for callers, as it's no longer possible to assume the transaction failed if and only if it consumed all gas after the introduction of the `REVERT` opcode in EIP-140.	Nick Johnson	Core	Final	Metropolis Byzantinium
EIP-706	DEVp2p snappy compression	Péter Szilágyi	Networking	Final	

EIP/ERC #	Title/Description	Author	Layer	Status	Created
EIP-721	ERC-721 Non-Fungible Token Standard. A standard API that allows smart contracts to operate as unique tradable non-fungible tokens (NFTs) that may be tracked in standardized wallets and traded on exchanges as assets of value, similar to ERC20. CryptoKitties was the first popularly adopted implementation of a digital NFT in the Ethereum ecosystem.	William Entriken, Dieter Shirley, Jacob Evans, Nastassia Sachs	Standard	Draft	
EIP-758	Subscriptions and filters for completed transactions	Jack Peterson	Interface	Draft	
EIP-801	ERC-801 Canary Standard	ligi	Interface	Draft	

EIP/ERC #	Title/Description	Author	Layer	Status	Created
EIP-827	ERC827 Token Standard. An extension of the standard interface ERC20 for tokens with methods that allow the execution of calls inside transfer and approvals. This standard provides basic functionality to transfer tokens, as well as allowing tokens to be approved so they can be spent by another on-chain third party. Also, it allows the developer to execute calls on transfers and approvals.	Augusto Lemble	ERC	Draft	

EIP/ERC #	Title/Description	Author	Layer	Status	Created
EIP-930	ERC930 Eternal Storage. The ES (Eternal Storage) contract is owned by an address that has write permissions. The storage is public, which means everyone has read permissions. It stores the data in mappings, using one mapping per type of variable. The use of this contract allows the developer to migrate the storage easily to another contract if needed.	Augusto Lemble	ERC	Draft	