contract account receives a message its code activates, allowing it to read and write to internal storage and send other messages or create contracts in turn.

Messages and Transactions

"Messages" in Ethereum are somewhat similar to “transactions” in Bitcoin, but with three important differences. First, an Ethereum message can be created either by an external entity or a contract, whereas a Bitcoin transaction can only be created externally. Second, there is an explicit option for Ethereum messages to contain data. Finally, the recipient of an Ethereum message, if it is a contract account, has the option to return a response; this means that Ethereum messages also encompass the concept of functions.

The term "transaction" is used in Ethereum to refer to the signed data package that stores a message to be sent from an externally owned account. Transactions contain the recipient of the message, a signature identifying the sender, the amount of ether and the data to send, as well as two values called STARTGAS and GASPRICE. In order to prevent exponential blowup and infinite loops in code, each transaction is required to set a limit to how many computational steps of code execution it can spawn, including both the initial message and any additional messages that get spawned during execution. STARTGAS is this limit, and GASPRICE is the fee to pay to the miner per computational step. If transaction execution "runs out of gas", all state changes revert - except for the payment of the fees, and if transaction execution halts with some gas remaining then the remaining portion of the fees is refunded to the sender. There is also a separate transaction type, and corresponding message type, for creating a contract; the address of a contract is calculated based on the hash of the account nonce and transaction data.

An important consequence of the message mechanism is the "first class citizen" property of Ethereum - the idea that contracts have equivalent powers to external accounts, including the ability to send message and create other contracts. This allows contracts to simultaneously serve many different roles: for example, one might have a member of a decentralized organization (a contract) be an escrow account (another contract) between an paranoid individual employing custom quantum-proof Lamport signatures (a third contract) and a co-signing entity which itself uses an account with five keys for security (a fourth contract). The strength of the Ethereum platform is that the decentralized organization and the escrow contract do not need to care about what kind of account each party to the contract is.