Contracts, Parachains and Parathreads

Contributions by: Tahgi D. Jones

Smart Contract Or.....
Parachain...?

Key Differences

- Smart contracts are immutable once deployed, whereas parachains built with substrate are customizable and upgradeable.
-
- When developing a smart contract, there is a trade-off between optimizing for usability and readability versus optimizing for gas fees and maintainability.
-
- Parachains are block-based software that can perform forkless upgrades due to the modularity of the substrate framework.
-
- Substrate-based technologies require a significant amount of education and financial resources to bring a product to market, giving smart contracts an advantage.
-
- Parathreads offer a happy medium between parachains and smart contracts, with lower capital requirements than parachains. However, they still require a deep understanding to implement.

Application Design

Technologies used

Frontend-Python & Streamlit

Backend-Solidity/ERC721/Ganache

Breakdown of tasks and roles - Creation of smart contract

Ganache to act as RPC endpoint & Private Key Vault

Streamlit to provide a way for users to interact with our smart contract.

Challenges - Working through compile errors and limiting contract functionality in order to optimize for gas fees while maintaining contract use ability.

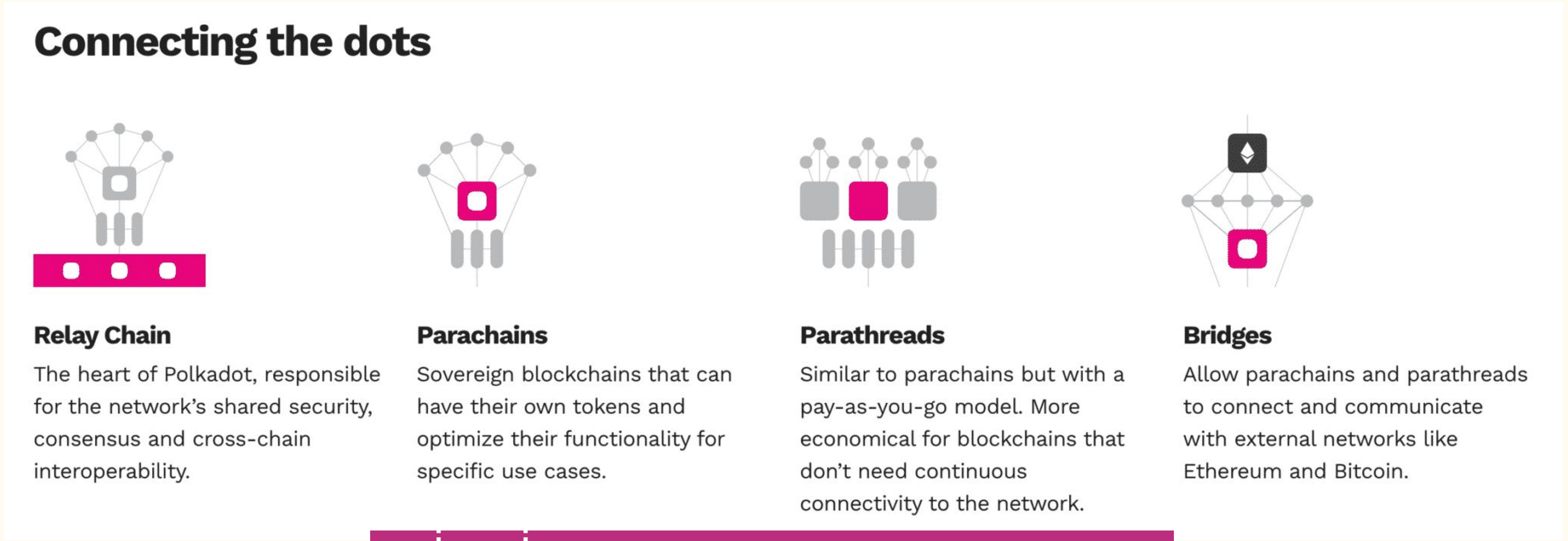
Successes - Striking a balance of usability and gas optimization.

Smart Contracts

- Smart contracts are block-based software that are like parachains.
- Once deployed, smart contracts are immutable.
- There is a trade-off between optimizing smart contracts for usability and readability versus optimizing for gas fees and maintainability.
- Smart contracts are written in programming languages such as Solidity.

Parachains

- Parachains are block-based software that are similar to smart contracts.
- Parachains built with substrate have the ability to perform forkless upgrades.
- Substrate-based technologies require a significant amount of education and deep pockets to bring a product to market.
- Parathreads offer a happy medium between parachains and smart contracts and require less capital to implement.



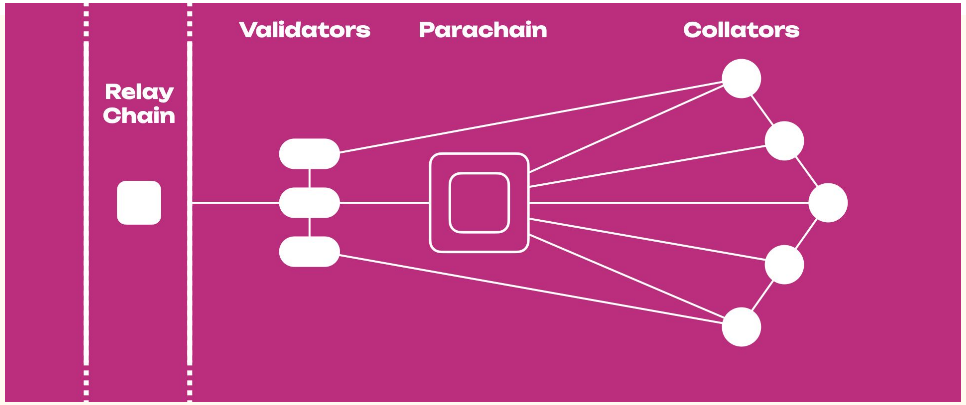
Connecting the dots

Relay Chain
The heart of Polkadot, responsible for the network's shared security, consensus and cross-chain interoperability.

Parachains
Sovereign blockchains that can have their own tokens and optimize their functionality for specific use cases.

Parathreads
Similar to parachains but with a pay-as-you-go model. More economical for blockchains that don't need continuous connectivity to the network.

Bridges
Allow parachains and parathreads to connect and communicate with external networks like Ethereum and Bitcoin.



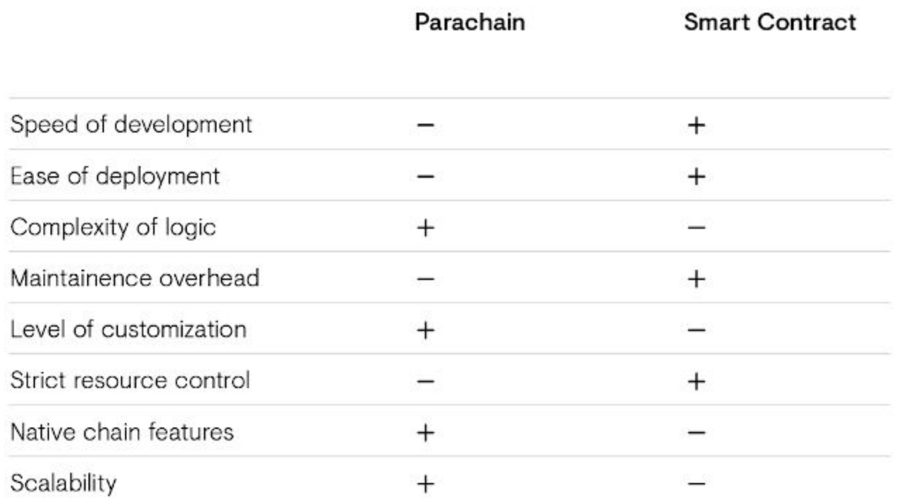
Parathread (The best of both)

- Parathreads offer a lightweight and efficient option that allows multiple projects to share a single parachain.

- Parathreads require less capital to implement than parachains, making them more accessible to small and medium-sized projects.

- Flexibility and scalability are important for long-term viability and may be attractive to investors.

- High capital requirements for substrate-based technologies may limit investment opportunities for smaller projects.



	Parachain	Smart Contract
Speed of development	–	+
Ease of deployment	–	+
Complexity of logic	+	–
Maintainence overhead	–	+
Level of customization	+	–
Strict resource control	–	+
Native chain features	+	–
Scalability	+	–

Construction

This section details how smart contracts, parachains and parathreads are constructed.

Smart Contract Construction

Solidity

•Solidity is a programming language used to write smart contracts for Ethereum and other blockchains.

•Developers write code in Solidity using a text editor or integrated development environment (IDE).

•The Solidity compiler is used to compile the code into bytecode, which is then deployed onto the blockchain.

•Once deployed, the smart contract is immutable and cannot be changed.

Substrate

•Substrate is a blockchain development framework that enables the creation of customized blockchains with specific features and functionalities.

•Substrate-based smart contracts are typically written in Rust, a systems programming language.

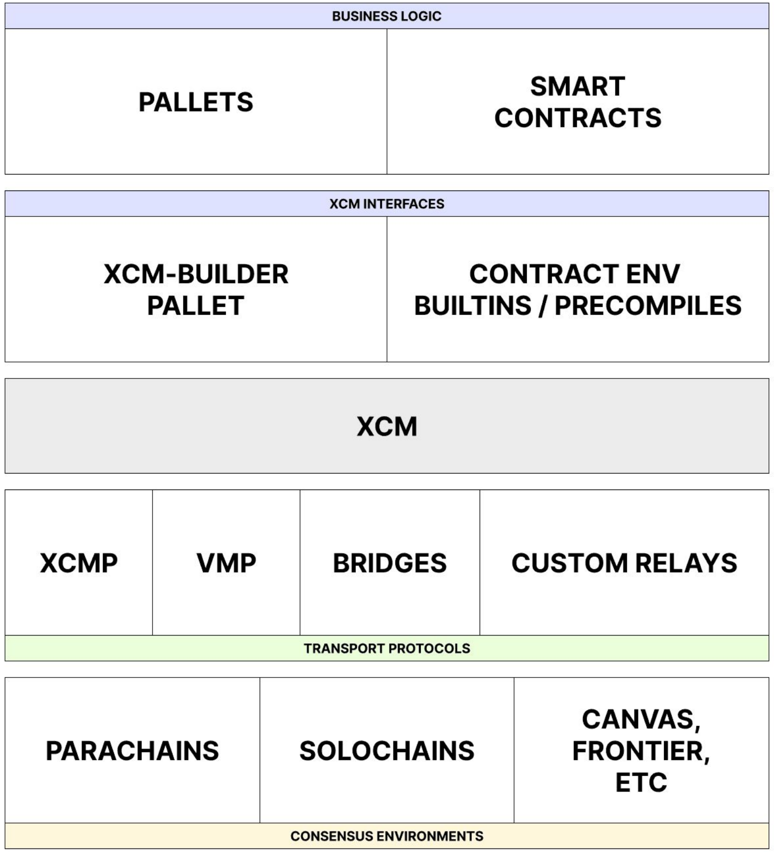
•Developers use the Substrate framework to create custom modules that define the smart contract's features and functionalities.

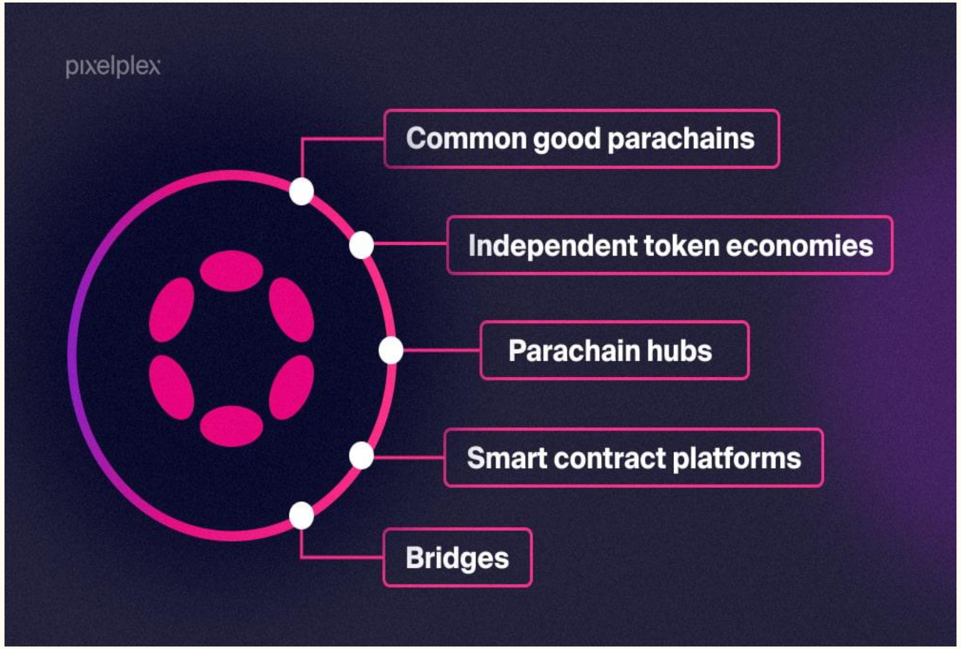
•The smart contract is compiled into WebAssembly (Wasm) bytecode and deployed onto a Substrate-based blockchain.

•Substrate-based smart contracts can be upgraded without requiring a hard fork, due to the modularity of the framework.

Construction of a Parachain

- Parachain can be seen as sovereign nations states, due to the fact that they implement their own State Transition Function.
- The only constraint the relay chain place on parachain is that the STF must be easily verifiable.
- Collators are parachain network maintainers and have the responsibility of remaining available for the relay to validate.
- They also have the responsibility of passing XCMP messages from parachain to parachain or parachain to relay chain.


BUSINESS LOGIC
PALLETS
SMART CONTRACTS
XCM INTERFACES
XCM-BUILDER PALLET
CONTRACT ENV BUILTINS / PRECOMPILES
XCM
XCMP
VMP
BRIDGES
CUSTOM RELAYS
TRANSPORT PROTOCOLS
PARACHAINS
SOLOCHAINS
CANVAS, FRONTIER, ETC
CONSENSUS ENVIRONMENTS


pixelplex
Common good parachains
Independent token economies
Parachain hubs
Smart contract platforms
Bridges

Construction of a Parathread

- Parathreads are constructed similarly to parachain.
- The key difference is that they only need a fraction of Polkadot’s computational due to the fee structure they implement.
- This means the parathread will need to make the desire to produce a block known and pay for polkadot’s computation to do so.

Smart Contract Demo

Directions for Future Development

Endless usability but will require a large amount of community outreach and education programs. (Polkadot Academy)

Potential for Solidity to benefit from adopting some substrate framework practices.

Links

- Deployed
- GitHub repo