dApp Development

Overview of Solidity Development	Solidity is a simple programming language that has a lot of benefits to the Ethereum community. It was primarily created to fit the needs Ethereum when it was still an emerging community. It has been a major achievement for the community due to its adoption rate and what has been created. It allows developers to start tinkering immediately and get them hooked on the platform, but it comes with its drawbacks. This language was built from the ground up, so there aren't nearly as many libraries and frameworks compared with more established languages. There are also very few developer tools around the Solidity ecosystem compared to a language like Java. They primarily consist of Truffle, Embark, Zepplin, and Oyente. Both of these factors contribute to slower development, and buggier code. This is still very early and progress has been impressive so far. Solidity also has significant security issues. Some of them revolve around a lack of hand-holding on the part of the compiler. For example, a constructor in Solidity starts with an uppercase character and is only called on creation, but a lowercase constructor would be public and could be called by anyone, leaving critical code exposed to the public.[1] There are also underflow and overflow errors. These errors are worsened by the difficulty to maintain a contract once it is released to the wild. Overall, solidity is a good start but not extensible enough to actually become the language of choice for the "world computer". We are early in the development and complexity of the ecosystem, and Solidity is probably not ready to handle the full weight of the system.
Other Languages for dApp Development	• **Vyper –** Language under development by the Ethereum foundation that removes a lot of the functionality of Solidity to allow for increased security and auditability. • **Simplicity –** Created to eventually replace Bitcoin Script, as a more expressive language. Also works well with MAST to decrease the space

	each contract takes up on the blockchain. It is likely a long ways away from being built on Ethereum because state is not natively supported, and must be creatively established. • **AxLang –** This is the most exciting of all the Solidity replacements. It is a language built on Scala, and can compile to the Java and Ethereum virtual machines. This allows for quick testing locally with the JVM side, and formally verifying code without deploying to the live network. Additionally, AxLang was built to have other languages built on top of it to form powerful network effects. Once released, AxLang will likely be the best language to use for Ethereum development. • **Other Functional Programming Languages –** Functional programming languages allow for simpler programming, so that way they are safer in the perspective of coding errors, and provable correctness. But they are not exactly "more secure". In fact, much of the space and runtime cannot be guaranteed, which can cause important issues. They also do not have the amount of libraries or existing developers and are significantly slower than C or other declarative languages like Java and C++. The largest platforms that use these types of programming are Tezos and Cardano. • **Imperative Programming Languages –** These include languages like C, C++, Java, Python, etc. They are used in implementations of the Ethereum Virtual Machine and for Bitcoin nodes, but they are not the official scripting languages of any of the major networks that are live now.
Turing Completeness	Ethereum's Turing completeness comes at a cost; contracts can take longer to run, be more expensive to execute, and be more complicated to verify. Some languages, like Simplicity, are more restrictive with contracts on Ethereum, and other blockchains don't allow for Turing Completeness at all, like Stellar and EOS. Many applications of distributed consensus will not need Turing Completeness. A Deterministic Finite Automata, which does not

	need Turing completeness, can completely model complicated, real-world contract. Just the outstanding value of worldwide derivatives contracts is 542.4 trillion dollars currently, and the majority of these contracts would not need Turing Completeness. This is just one example from the legacy financial system, and might not matter for new decentralized applications, especially as on-chain governance will advance the need for Turing Completeness.
Ethereum Virtual Machine	• Most of the problems in Ethereum development are not centered in Solidity, but rather the way that the Ethereum virtual machine works. Solidity, like some of its relatives, was specifically made to compile into EVM bytecode. This causes people to misattribute some of the EVM issues with Solidity itself. The EVM was built early in Ethereum's existence and is an admirable project. The EVM was necessary for its nodes agree on computations, and for the platform to get to this point. But it is lacking in functionality and without upgrades Ethereum will be surpassed. Improving languages in Ethereum without improving the Virtual Machine will probably have a limited effect as well. • The Internet has allowed for us to scale our communities to incredible sizes, but as these networks grow, the incremental value of each new user decreases- humans interactions tend to be limited to communities. Public blockchains will allow new types of organizations, communities and markets. In particular, multiparty computation will be a huge area of focus in the future on blockchains as the role of decentralized organizations gets bigger. They will be building blocks for significantly complicated types of organizations and protocols. • Blockchains have particular qualities that give them great properties as cryptographic primitives. They can be used as "bulletin boards", and used for fair Multi-Party Computation. An application of this is auctions on a blockchain. Unfortunately, there is not much support for a significant

	amount of cryptography functions in Solidity or the EVM, and workarounds are prohibitively expensive in the EVM in terms of gas. As a result, there have been some interesting projects that realized these needs and tried to provide off-chain solutions.
Quick Overview of Other Platforms	Ethereum focuses on the thought of "One blockchain to rule them all." There are others that take this approach, like EOS, but some take other approaches. Compared to all of these networks, Ethereum has built a bigger network of developers, and already has dApps in production around the world. **EOS –** EOS is not Turing Complete, and focuses on transaction throughput. EOS uses C++ for their smart contracts. It will try to go head to head with Ethereum, but only the testnet is live. **Stellar** – Focuses on Fiat bridges to the cryptocurrency world. It is not Turing Complete and is tailored primarily transactions rather than true applications. There has been an increase of ICO's on Stellar, and many developers are attracted by the very low transaction costs. Has a unique consensus protocol that could allow for significant boosts in transaction throughput. Having access to the traditional financial world may allow it to bootstrap a very formidable dApp ecosystem. **Cosmos –** Allows for programming in any chain. Like Polkadot, it allows for relaying of tokens between chains (hubs). Cosmos uses the Tendermint consensus algorithm and its Application-Blockchain Interface. It also abstracts the work of spinning up networks. Unlike Polkadot, Cosmos allows private blockchains as well. Overall, it aims for low barriers to entry into development and use of public and private chains. **Dfinity –** Aims to incorporate the legacy IT infrastructure and businesses into the existing decentralized landscape for cost savings to create a

	decentralized cloud. Along with most popular languages will support the EVM and Ethereum, which will run inside Dfinity. Like Cosmos, it also aims to lower the cost of spinning up new blockchains and creating a functioning ecosystem around it. There will also exist a decentralized governance algorithm that will help shut down miscreant contracts. **Polkadot –** Focuses on relaying transactions and contracts between chains, and pooled security. There is almost no focus on building apps on Polkadot, but instead it aims to enable developers to get the benefits of multiple chains, that a single chain cannot adequately provide. Polkadot will be auxiliary to other chains like Ethereum, but might cause an evening out of the chains as interoperability will increase. **Cardano –** Built in Haskell, this project aims to create a blockchain that incorporates existing financial applications and governments along with other blockchains. Their smart contract layer is not available yet, so it remains to be seen what the developer community will come up with. There would be interoperability with Ethereum, so it won't be in pure competition with Ethereum's existing network. **Tezos –** Built on Michelson and Liquidity, a language very similar to the OCaml programming language, for formal verification. Highly focused on on-chain governance. The governance of this project as a whole has suffered setbacks, but the Alphanet is out. Ethereum has a head start, but Tezos' governance capabilities and focus on formal verification could set it apart. **Blockstack –**The focus of Blockstack is on users owning their own data and the blockchain involved allows verifications of id's, namespaces, and incentivizing the networks decentralization and improvement. Blockstack primarily runs in the browser, and allows developers to build applications in any language that they want. It will support mobile clients as well. Blockstack provides the best path for developers to get their apps in the hands of users

	that do not completely depend on blockchain underpinnings. It will be interesting to see if a tech company with existing distribution, like Apple, decides to implement this type of system. Only time will tell if Ethereum incorporates more support an ecosystem that is not as heavily reliant on the chain itself.
Potential Outcomes	**E-WASM –** Ethereum will eventually support more development languages, but it will likely come with the expected adoption of an Ethereum-flavored version of Web Assembly, called E-WASM. Web Assembly is a low-level programming language that will run in browsers. Developers will be able to port/compile their application from languages such as C++ to Web Assembly. This will also decrease the time to develop smart contracts. Web Assembly will have a large support base outside of the Ethereum community, so more languages will be added. **Sharding –** One of the consequences of static gas pricing is that the market forces only concerns the price of Ether. The actual computations for miners are of varying costs. Smart contracts also depend on computations differently. Different shards with different gas prices for computations could emerge and solve this issue to create gas prices optimized for specific needs of the users/contracts. Along the lines of use-case specific shards, there could be chains that use a different VM to call optimized C (or any other language) code. This would likely require a hard-fork of the network, but would allow for specialized chains for on-chain cryptography beyond the capabilities of current network. This would enable new possibilities that were previously unavailable, and adoption of existing frameworks to make developers' lives easier. Most importantly, it will provide a solid counter for Ethereum against competition from Multi-Chain networks for leverage on the developer community. **Multi-Chain Networks(Cardano, Polkadot, Cosmos, Dfinity) –** These

	networks will likely see more support and use from existing businesses with their private chains. Multi-Chain Networks empowers developers in the ecosystem technically and economically with interoperability and reducing the switching cost between different chains. We have a long way to go to this point, but it would be a fascinating landscape. Ethereum is a platform, just like AWS or Twillio, and even though these are not cryptocurrency related, some parts of economics will stay the same. With enough size and infrastructure, companies (and maybe DAO's) will seek to optimize all the way up and down the stack. This will likely be amplified by the cost of replicating software approaching zero with open-sourced code, less real-world resources consumed after proof of stake, and an ability to run more centralized chains on networks like Cosmos. Will lead to the new "firms" and platforms competing up and down the value chain to provide the best combination of economics, developer and user experience, surrounding ecosystem, scalability, latency, and security. These chains could replace Ethereum as the bottom layer of the stack as the actual operating system equivalent while Ethereum would still be a single but important VM. We run many different VM's together in the cloud that are spun up and shut down as needed- it might be possible in the decentralized tech stack that we run multiple different blockchains. Regardless of what happens in the ecosystem, it'll be studied greatly in the future, and will be fascinating to watch in real time.
References:	[1]- In person talk with Matt Green, Founding Scientist of Z-Cash