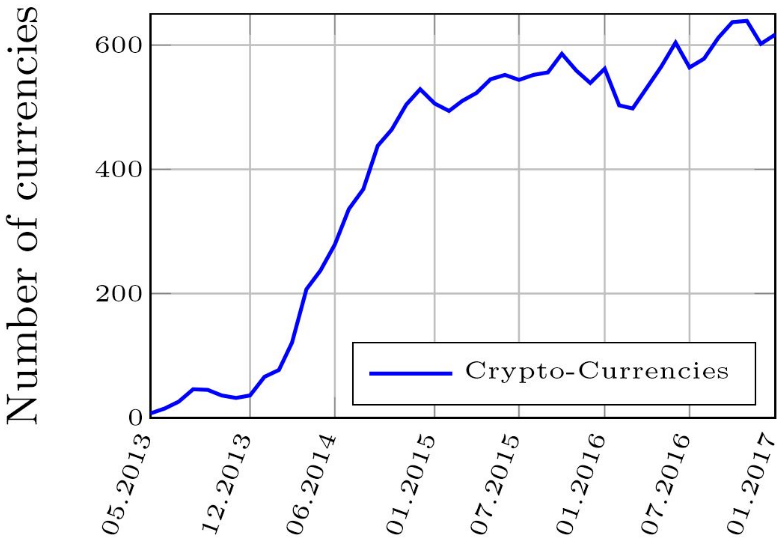
An empirical analysis of smart contracts

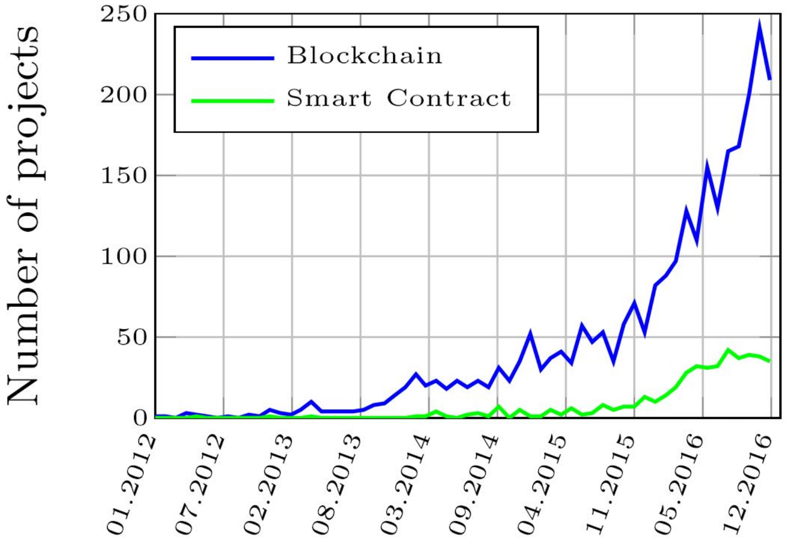
platforms, applications, and design patterns

Massimo Bartoletti **Livio Pompianu**

Università di Cagliari

“Hype” on blockchains and smart contracts





- Increasing interest on cryptocurrencies, blockchain, and smart contracts
- The technology is evolving quickly
- We describe the current situation, by answering to the following questions

An empirical analysis of smart contracts - Questions

- What platforms allow to build and execute smart contracts?
- What applications are developed as smart contracts?
- What design patterns are adopted for writing smart contracts?
- What correlations exist between applications and design patterns?

Platforms for smart contract

Platforms for smart contracts - Methodology

1. We examined all the articles of coindesk.com in the “smart contracts” category: **175 articles** from June 2013 up to the 15th of September 2016

2. We built a first list of 12 platforms by including projects mentioned in the articles

3. We excluded the projects that we could not analyse, i.e. the platforms which do not satisfy one of the following criteria:
    a. have already been launched
    b. are running and supported from a community of developers
    c. are publicly accessible

CoinDesk

Bitcoin	Ethereum	Counterparty
- Contract blockchain - Public	- Contract blockchain - Public	- Contract blockchain - Public
Stellar	**Monax**	**Lisk**
- Contract blockchain - Public	- Contract blockchain - Private	- Contract blockchain - Private

Bitcoin	**Ethereum**	**Counterparty**
- Contract blockchain - Public - **Language - Bitcoin scripting**	- Contract blockchain - Public - **Language - EVM**	- Contract blockchain - Public - **Language - EVM**
Stellar	**Monax**	**Lisk**
- Contract blockchain - Public - **Language - Batch operations + multisignature accounts**	- Contract blockchain - Private - **Language - EVM**	- Contract blockchain - Private - **Language - JavaScript + NodeJS**

Bitcoin	Ethereum	Counterparty
- Contract blockchain - Public - Language - Bitcoin scripting - **Consensus - Proof of Work**	- Contract blockchain - Public - Language - EVM - **Consensus - Proof of Work**	- Contract blockchain - Public - Language - EVM - **Consensus - N/A**
Stellar	**Monax**	**Lisk**
- Contract blockchain - Public - Language - Batch operations + multisignature accounts - **Consensus - Inspired from federated Byzantine agreement**	- Contract blockchain - Private - Language - EVM - **Consensus - Tendermint**	- Contract blockchain - Private - Language - JavaScript + NodeJS - **Consensus - Delegated Proof of Stake**

Bitcoin	**Ethereum**	**Counterparty**
- Contract blockchain - Public - Language - Bitcoin scripting - Consensus - Proof of Work - **Marketcap (M USD) - 18,239**	- Contract blockchain - Public - Language - EVM - Consensus - Proof of Work - **Marketcap (M USD) - 4,144**	- Contract blockchain - Public - Language - EVM - Consensus - N/A - **Marketcap (M USD) - 9**
Stellar	**Monax**	**Lisk**
- Contract blockchain - Public - Language - Batch operations + multisignature accounts - Consensus - Inspired from federated Byzantine agreement - **Marketcap (M USD) - 23**	- Contract blockchain - Private - Language - EVM - Consensus - Tendermint - **Marketcap (M USD) - N/A**	- Contract blockchain - Private - Language - JavaScript + NodeJS - Consensus - Delegated Proof of Stake - **Marketcap (M USD) - 29**

Analysing the usage of smart contracts

Usage of smart contracts - Methodology

Ethereum

- we collect all contracts with “verified” Solidity source code on etherscan.io
- 811 contracts

Bitcoin

- we develop a tool to extract the Bitcoin transactions that:
  1) attach metadata by using the OP_RETURN instruction
  2) have been published by a smart contract
- 23 smart contracts

Extraction date for both Bitcoin and Ethereum platforms: **01/01/2017**

Financial

Manage, gather, or distribute money

- Certify the ownership of a real-world asset (Colu, Omni, Counterparty)
- Crowdfunding (The DAO)
- Ponzi schemes (Government, KingOfTheEtherThrone)
- Insurance on setbacks digitally provable (Etherisc)
- Publish advertisement messages (PixelMap)

Financial	Notary
Manage, gather, or distribute money - Certify the ownership of a real-world asset (Colu, Omni, Counterparty) - Crowdfunding (The DAO) - Ponzi schemes (Government, KingOfTheEtherThrone) - Insurance on setbacks digitally provable (Etherisc) - Publish advertisement messages (PixelMap)	Store some data persistently, and certify ownership - Write the hash of a document on the blockchain (Proof of Existence) - Declare copyrights on digital arts files (Monegraph) - Write messages that everyone can read (Eternity Wall) - Associate users to addresses certifying their identity (Physical Address)

Financial

Manage, gather, or distribute money

- Certify the ownership of a real-world asset (Colu, Omni, Counterparty)
- Crowdfunding (The DAO)
- Ponzi schemes (Government, KingOfTheEtherThrone)
- Insurance on setbacks digitally provable (Etherisc)
- Publish advertisement messages (PixelMap)

Notary

Store some data persistently, and certify ownership

- Write the hash of a document on the blockchain (Proof of Existence)
- Declare copyrights on digital arts files (Monegraph)
- Write messages that everyone can read (Eternity Wall)
- Associate users to addresses certifying their identity (Physical Address)

Game

Contracts implementing games

- Games of chance (Lottery, Dice, Roulette, RockPaperScissors)
- Games of skills (Etherization)
- Games mixing chance and skills (PRNG challenge)

Financial	Notary
Manage, gather, or distribute money - Certify the ownership of a real-world asset (Colu, Omni, Counterparty) - Crowdfunding (The DAO) - Ponzi schemes (Government, KingOfTheEtherThrone) - Insurance on setbacks digitally provable (Etherisc) - Publish advertisement messages (PixelMap)	Store some data persistently, and certify ownership - Write the hash of a document on the blockchain (Proof of Existence) - Declare copyrights on digital arts files (Monegraph) - Write messages that everyone can read (Eternity Wall) - Associate users to addresses certifying their identity (Physical Address)

Game	Wallet	
Contracts implementing games - Games of chance (Lottery, Dice, Roulette, RockPaperScissors) - Games of skills (Etherization) - Games mixing chance and skills (PRNG challenge)	Simplify the interaction with the blockchain: handle keys, send transactions, manage money, deploy and watch contracts	

Financial

Manage, gather, or distribute money

- Certify the ownership of a real-world asset (Colu, Omni, Counterparty)
- Crowdfunding (The DAO)
- Ponzi schemes (Government, KingOfTheEtherThrone)
- Insurance on setbacks digitally provable (Etherisc)
- Publish advertisement messages (PixelMap)

Notary

Store some data persistently, and certify ownership

- Write the hash of a document on the blockchain (Proof of Existence)
- Declare copyrights on digital arts files (Monegraph)
- Write messages that everyone can read (Eternity Wall)
- Associate users to addresses certifying their identity (Physical Address)

Game

Contracts implementing games

- Games of chance (Lottery, Dice, Roulette, RockPaperScissors)
- Games of skills (Etherization)
- Games mixing chance and skills (PRNG challenge)

Wallet

Simplify the interaction with the blockchain:

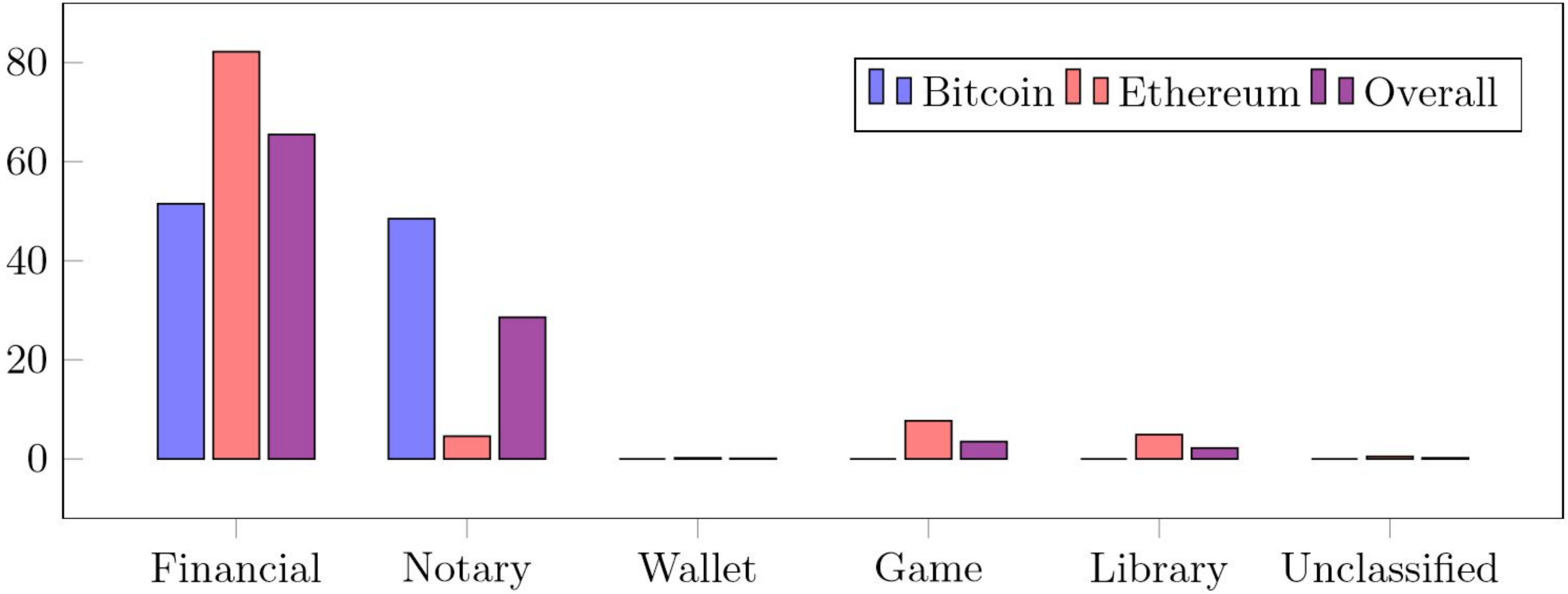
handle keys, send transactions, manage money, deploy and watch contracts

Library

Implement general-purpose operations to be used by other contracts

For instance math and string transformations

Distribution of transactions by category



Design patterns for Ethereum smart contracts

Token		
Distribute some fungible goods (represented by tokens) to users - Track the ownership of a physical or digital property (gold, cryptocurrency) - Crowdfunding systems issue tokens in exchange for donations (Congress) - Regulate authorizations and identities, e.g. vote in a poll (ETCSurvey) Standardization proposal in the ERC20		

Token	**Authorization**	
Distribute some fungible goods (represented by tokens) to users - Track the ownership of a physical or digital property (gold, cryptocurrency) - Crowdfunding systems issue tokens in exchange for donations (Congress) - Regulate authorizations and identities, e.g. vote in a poll (ETCSurvey) Standardization proposal in the ERC20	Restrict the execution of code according to the caller address - Check if the caller is the owner before performing critical operations - Ensuring that each user vote only once per poll (Corporation) - Define a white-list of addresses that can withdraw funds (CharlyLifeLog)	

Token	Authorization	Oracle
Distribute some fungible goods (represented by tokens) to users - Track the ownership of a physical or digital property (gold, cryptocurrency) - Crowdfunding systems issue tokens in exchange for donations (Congress) - Regulate authorizations and identities, e.g. vote in a poll (ETCSurvey) Standardization proposal in the ERC20	Restrict the execution of code according to the caller address - Check if the caller is the owner before performing critical operations - Ensuring that each user vote only once per poll (Corporation) - Define a white-list of addresses that can withdraw funds (CharlyLifeLog)	The Ethereum language does not allow contracts to query external sites Oracles contracts are the interface between contracts and the *outside* Instead of querying an external service, a contract queries an oracle When the service needs to update its data, it sends a transaction to the oracle The most common oracle is Oraclize

Token	Authorization	Oracle
Distribute some fungible goods (represented by tokens) to users - Track the ownership of a physical or digital property (gold, cryptocurrency) - Crowdfunding systems issue tokens in exchange for donations (Congress) - Regulate authorizations and identities, e.g. vote in a poll (ETCSurvey) Standardization proposal in the ERC20	Restrict the execution of code according to the caller address - Check if the caller is the owner before performing critical operations - Ensuring that each user vote only once per poll (Corporation) - Define a white-list of addresses that can withdraw funds (CharlyLifeLog)	The Ethereum language does not allow contracts to query external sites Oracles contracts are the interface between contracts and the *outside* Instead of querying an external service, a contract queries an oracle When the service needs to update its data, it sends a transaction to the oracle The most common oracle is Oraclize
Randomness		
Contract execution must be deterministic: all the nodes must obtain the same value when asking for a random number - Query an oracle to generate the value off-chain (Slot) - Generate the number locally, by using values not predictable a priori (Lottery)		

Token	Authorization	Oracle
Distribute some fungible goods (represented by tokens) to users - Track the ownership of a physical or digital property (gold, cryptocurrency) - Crowdfunding systems issue tokens in exchange for donations (Congress) - Regulate authorizations and identities, e.g. vote in a poll (ETCSurvey) Standardization proposal in the ERC20	Restrict the execution of code according to the caller address - Check if the caller is the owner before performing critical operations - Ensuring that each user vote only once per poll (Corporation) - Define a white-list of addresses that can withdraw funds (CharlyLifeLog)	The Ethereum language does not allow contracts to query external sites Oracles contracts are the interface between contracts and the *outside* Instead of querying an external service, a contract queries an oracle When the service needs to update its data, it sends a transaction to the oracle The most common oracle is Oraclize
Randomness	**Poll**	
Contract execution must be deterministic: all the nodes must obtain the same value when asking for a random number - Query an oracle to generate the value off-chain (Slot) - Generate the number locally, by using values not predictable a priori (Lottery)	Allow users to vote on some question For instance decide whether an emergency withdrawal is needed (Dice) To determine who can vote and keep track of the votes, polls can - Use tokens - Check the voters' addresses	

Token

Distribute some fungible goods (represented by tokens) to users

- Track the ownership of a physical or digital property (gold, cryptocurrency)
- Crowdfunding systems issue tokens in exchange for donations (Congress)
- Regulate authorizations and identities, e.g. vote in a poll (ETCSurvey)

Standardization proposal in the ERC20

Authorization

Restrict the execution of code according to the caller address

- Check if the caller is the owner before performing critical operations
- Ensuring that each user vote only once per poll (Corporation)
- Define a white-list of addresses that can withdraw funds (CharlyLifeLog)

Oracle

The Ethereum language does not allow contracts to query external sites

Oracles contracts are the interface between contracts and the *outside*

Instead of querying an external service, a contract queries an oracle

When the service needs to update its data, it sends a transaction to the oracle

The most common oracle is Oraclize

Randomness

Contract execution must be deterministic: all the nodes must obtain the same value when asking for a random number

- Query an oracle to generate the value off-chain (Slot)
- Generate the number locally, by using values not predictable a priori (Lottery)

Poll

Allow users to vote on some question

For instance decide whether an emergency withdrawal is needed (Dice)

To determine who can vote and keep track of the votes, polls can

- Use tokens
- Check the voters’ addresses

Time constraint

Specify when an action is permitted

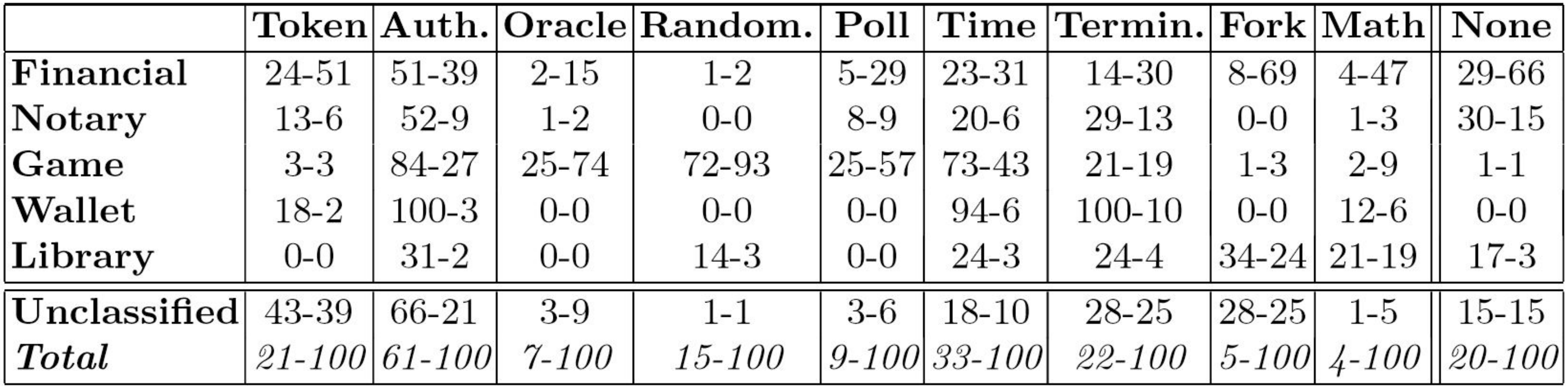
- In notary contracts, prove that a document is owned from a certain date
- Mark different stages of a game (Lottery)
- Allow to withdraw funds after a date (BirthdayGift)

Token	**Authorization**	**Oracle**
Distribute some fungible goods (represented by tokens) to users - Track the ownership of a physical or digital property (gold, cryptocurrency) - Crowdfunding systems issue tokens in exchange for donations (Congress) - Regulate authorizations and identities, e.g. vote in a poll (ETCSurvey) Standardization proposal in the ERC20	Restrict the execution of code according to the caller address - Check if the caller is the owner before performing critical operations - Ensuring that each user vote only once per poll (Corporation) - Define a white-list of addresses that can withdraw funds (CharlyLifeLog)	The Ethereum language does not allow contracts to query external sites Oracles contracts are the interface between contracts and the *outside* Instead of querying an external service, a contract queries an oracle When the service needs to update its data, it sends a transaction to the oracle The most common oracle is Oraclize
Randomness	**Poll**	**Time constraint**
Contract execution must be deterministic: all the nodes must obtain the same value when asking for a random number - Query an oracle to generate the value off-chain (Slot) - Generate the number locally, by using values not predictable a priori (Lottery)	Allow users to vote on some question For instance decide whether an emergency withdrawal is needed (Dice) To determine who can vote and keep track of the votes, polls can - Use tokens - Check the voters' addresses	Specify when an action is permitted - In notary contracts, prove that a document is owned from a certain date - Mark different stages of a game (Lottery) - Allow to withdraw funds after a date (BirthdayGift)
Termination		
Disable a contract when its use has come to an end		

Token	Authorization	Oracle
Distribute some fungible goods (represented by tokens) to users - Track the ownership of a physical or digital property (gold, cryptocurrency) - Crowdfunding systems issue tokens in exchange for donations (Congress) - Regulate authorizations and identities, e.g. vote in a poll (ETCSurvey) Standardization proposal in the ERC20	Restrict the execution of code according to the caller address - Check if the caller is the owner before performing critical operations - Ensuring that each user vote only once per poll (Corporation) - Define a white-list of addresses that can withdraw funds (CharlyLifeLog)	The Ethereum language does not allow contracts to query external sites Oracles contracts are the interface between contracts and the *outside* Instead of querying an external service, a contract queries an oracle When the service needs to update its data, it sends a transaction to the oracle The most common oracle is Oraclize
Randomness	**Poll**	**Time constraint**
Contract execution must be deterministic: all the nodes must obtain the same value when asking for a random number - Query an oracle to generate the value off-chain (Slot) - Generate the number locally, by using values not predictable a priori (Lottery)	Allow users to vote on some question For instance decide whether an emergency withdrawal is needed (Dice) To determine who can vote and keep track of the votes, polls can - Use tokens - Check the voters' addresses	Specify when an action is permitted - In notary contracts, prove that a document is owned from a certain date - Mark different stages of a game (Lottery) - Allow to withdraw funds after a date (BirthdayGift)
Termination	**Math**	
Disable a contract when its use has come to an end	Encode the logic which guards the execution of some critical operations	

Token	Authorization	Oracle
Distribute some fungible goods (represented by tokens) to users - Track the ownership of a physical or digital property (gold, cryptocurrency) - Crowdfunding systems issue tokens in exchange for donations (Congress) - Regulate authorizations and identities, e.g. vote in a poll (ETCSurvey) Standardization proposal in the ERC20	Restrict the execution of code according to the caller address - Check if the caller is the owner before performing critical operations - Ensuring that each user vote only once per poll (Corporation) - Define a white-list of addresses that can withdraw funds (CharlyLifeLog)	The Ethereum language does not allow contracts to query external sites Oracles contracts are the interface between contracts and the *outside* Instead of querying an external service, a contract queries an oracle When the service needs to update its data, it sends a transaction to the oracle The most common oracle is Oraclize
Randomness	**Poll**	**Time constraint**
Contract execution must be deterministic: all the nodes must obtain the same value when asking for a random number - Query an oracle to generate the value off-chain (Slot) - Generate the number locally, by using values not predictable a priori (Lottery)	Allow users to vote on some question For instance decide whether an emergency withdrawal is needed (Dice) To determine who can vote and keep track of the votes, polls can - Use tokens - Check the voters' addresses	Specify when an action is permitted - In notary contracts, prove that a document is owned from a certain date - Mark different stages of a game (Lottery) - Allow to withdraw funds after a date (BirthdayGift)
Termination	**Math**	**Fork check**
Disable a contract when its use has come to an end	Encode the logic which guards the execution of some critical operations	Detect whether a contract is running on the main chain or on the fork

Design patterns for Ethereum smart contracts



	Token	Auth.	Oracle	Random.	Poll	Time	Termin.	Fork	Math	None
Financial	24-51	51-39	2-15	1-2	5-29	23-31	14-30	8-69	4-47	29-66
Notary	13-6	52-9	1-2	0-0	8-9	20-6	29-13	0-0	1-3	30-15
Game	3-3	84-27	25-74	72-93	25-57	73-43	21-19	1-3	2-9	1-1
Wallet	18-2	100-3	0-0	0-0	0-0	94-6	100-10	0-0	12-6	0-0
Library	0-0	31-2	0-0	14-3	0-0	24-3	24-4	34-24	21-19	17-3
Unclassified	43-39	66-21	3-9	1-1	3-6	18-10	28-25	28-25	1-5	15-15
Total	*21-100*	*61-100*	*7-100*	*15-100*	*9-100*	*33-100*	*22-100*	*5-100*	*4-100*	*20-100*

Relations between design patterns and contract categories

A pair (**p**,**q**) at row **i** and column **j** means that

- **p%** of the contracts in category **i** use the pattern of column **j**, and
- **q%** of contracts with pattern **j** belong to category **i**

Conclusions

Since the blockchain is *immutable*, uploaded contracts can not be modified

Even if a vulnerability is discovered, it can not be fixed

In this context, domain-specific languages (DSL) for smart contract could help

DSL allow to write contracts in which some properties can be verified

Verify properties reduce the possible vulnerabilities

Conclusions

We believe that this survey may provide valuable information to developers of new, domain-specific languages for smart contracts

Measuring what are the most common use cases allows to understand which domains deserve more investments

Our study of the correlation between design patterns and application domains can be exploited to drive the correct choice of programming primitives of these DSL

Thank you!