DELPHI

https://delphi.systems

Contents

Introduction **1**
The Blockchain . . . 1
Ethereum . . . 1
Smart Contracts and Oracles . . . 2
The Oracle Problem . . . 2
Our Approach . . . 3

Context **3**
Augur . . . 3
Oraclize . . . 4
Prediction Markets . . . 4
Existing Approaches . . . 6

Delphi Systems **7**
Pythia . . . 7
Abstract . . . 7
Philosophy . . . 7
Basic Design . . . 8
Beyond the Basics . . . 9
Advantages . . . 9
Summary . . . 10
PHI / sPHI . . . 11
Purpose . . . 11
Minimal Token : PHI . . . 11
Signal Token : sPHI . . . 12
Token Peg . . . 13
Summary . . . 14
Omphalos . . . 15
Core Principles . . . 15
Basic Design . . . 15
Beyond the Basics . . . 17
Benefits . . . 18
Tholos . . . 19
Vision . . . 19
Beyond Surface-Level Aesthetics . . . 19
Tools . . . 19
Long-Term, Moving Target . . . 20
Agora . . . 21
Abstract . . . 21
Platform . . . 21
Strategy . . . 22

Conclusion **23**
Vision of Delphi . . . 23

Introduction

The Blockchain

In 2008, Satoshi Nakamoto published the Bitcoin whitepaper[1] which introduced the concept of the *blockchain*, a distributed global consensus network that enables the flow of both information and value without involving the friction of middlemen or counterparties. Bitcoin and other blockchain systems employ an elegant arrangement of incentives, such that users acting in their own self-interest actually help to strengthen the network. This design allows individual peers on the network to reliably reach a state of consensus in a leaderless protocol with no central arbiter; the system is *decentralized*.

This proved to be a revolutionary breakthrough, with profound implications and extraordinary potential. The technology can eliminate the risks of censorship or counterparty default, remove or minimize reliance on trust, and open up incredible new opportunities and possibilities. When bootstrapped successfully, a blockchain network can grow into a self-sustaining, antifragile, distributed economic engine of prosperity.

Ethereum

The most successful blockchain project after Bitcoin is undoubtedly Ethereum,[2] a Turing-powerful (though gas-bounded) construct which leverages the blockchain to establish a transparent and open-access general state machine. Ethereum can thus function as a host platform and gateway to decentralized applications. Whereas Bitcoin allowed the world to *move value* across the Internet without middlemen or centralized gatekeepers for the very first time, Ethereum allowed the world to *run code* in the same way.

There are many advantages to building an application on Ethereum:

> Since it's run on Ethereum, it cuts out the middlemen and brings costs down to the economic minimum to operate things securely… [with fees of] 1% or less. For the first time people will be able to trade on a censorship resistant, global trading platform without having to trust counter-parties. Liquidity will be truly global because Ethereum **doesn't care** if you are from China *or* the US *or* Russia, you're just a pseudonymous address.[3]

Blockchains like Ethereum's are tools of freedom, profit, and progress. They allow individuals to share their knowledge and wealth online without the risks of third-party interference, and catalyze open and transparent free-market activity.

For certain types of applications (in particular those that would benefit from operating on a cryptographically secure, decentralized, tamper-proof and hyper-resilient, hyper-available network), Ethereum is the best available platform to build and live upon. These applications are generally referred to as *smart contracts*.

Smart Contracts and Oracles

> Smart contracts help you exchange money, property, shares, or anything of value in a transparent, conflict-free way, while avoiding the services of a middleman.[4]

> Oracles provide an indispensable service by connecting smart contracts and distributed autonomous organizations with real world data and events.[5]

The original definition of a "smart contract" was given as "a set of promises, including protocols within which the parties perform on these promises",[6] but another definition and introduction for both smart contracts and oracles is given by Massimo Bartoletti and Livio Pompianu in *An empirical analysis of smart contracts: platforms, applications, and design patterns*:

> **Smart contracts** are computer programs that can be consistently executed by a network of mutually distrusting nodes, without the arbitration of a trusted authority

> **Oracle**. Some contracts may need to acquire data from outside the blockchain, e.g. from a website, to determine the winner of a bet. The Ethereum language does not allow contracts to query external sites: otherwise, the determinism of computations would be broken, as different nodes could receive different results for the same query. **Oracles are the interface between contracts and the outside.**[7]

In other words, smart contracts allow code to govern over economic interactions, and oracles are the services which allow that code to "see" the real world.

The Oracle Problem

For any smart contract that depends on information that is external to the blockchain (i.e. "facts about the real world"), an *oracle* is needed, to provide the contract with data. Because the oracle determines the contract *input*, it also determines the contract *result* or *output*. This means that ultimately, the oracle involved with a smart contract has the power to decide how that contract behaves. In a system designed entirely to disintermediate and disseminate power, allowing a central entity access to this level of influence effectively defeats the entire purpose of the design and approach.

In computer science, there is a principle and phrase: "garbage in, garbage out."[8] This refers to program inputs; a useful or accurate output result cannot be reasonably expected from a program which did not receive quality input in the first place. Smart contracts, being digital executables themselves, are no exception to this rule. It has been argued that "the contract is the basic building block of a free market economy",[6] and any technological progress towards improving smart contract capabilities would be fully undermined by neglecting to ensure the quality of the input data and the source of it.

To date, many different approaches have been taken to address the so-called Oracle Problem, which have each broadly fallen into one of two categories: *centralized* oracles (the use of which completely negates the benefits of a smart contract architecture) and *rigid oracular consensus engines*, which necessarily require:

- extraordinary levels of architectural rigor and fastidious technical parameterization
- precision in economic incentives and the cryptographic balance thereof
- coordination costs (both in terms of time and money) during development and execution

This type of development tends to incur massive costs, and attempts at implementation have been consistently riddled with issues.[9][10] Consensus algorithms in general are notoriously difficult to understand, model, analyze, and refine (and very easy to break).[11][12][13]

Our Approach

To truly solve The Oracle Problem will undoubtedly require input from experts in many different fields. To expect any one development team to furnish the perfect solution alone is unrealistic. Instead, we propose a generalized distributed oracle toolkit, platform, and token.

We propose a compound token design to capitalize on the benefits of multiple different token implementations and enable a ubiquitous platform signaling mechanism and data source.

We present a weighted multisignature framework for the determination of oracular output values, allowing for extreme deployment flexibility and component modularity.

We offer a minimal, understandable and extensible prediction market design to take advantage of this oracle model and demonstrate its utility.

Finally, we commit to a wholehearted focus on user experience and community involvement, with a long-term vision of evolving into an oracle marketplace and ecosystem supplemented with sophisticated metrics and rating systems.

The above approach allows us to meaningfully leverage free market forces to optimize oracle solutions over time, providing a long-term-viable platform on which to build and a foundation for the innovators yet to come.

Context

Augur

Augur is an early prediction markets implementation, originally designed as an extension to the Bitcoin Core source code and using Bitcoin Script-based logic, which later switched to an Ethereum smart-contract based architecture.[14] The project's goal is "to democratize and decentralize finance"[3] and they held an ICO beginning in August 2015 which was preeminently successful (raising thousands of bitcoins worth millions of dollars at the time, which have considerably appreciated in value in the time since the auction took place) and the crowdsale was subsequently greatly acclaimed and celebrated.[15][16][17]

Though the project has experienced a number of directional readjustments since its debut, the basic design is predicated around one particular event resolution model, where the system itself functions as a single rigid and monolithic oracle powered by an iterative commit-and-reveal process that any REP (token) holder is free to participate in. Platform fees are collected on the base layer of the stack and are shared among all voters. This requires a relatively heavyweight design[18] and relatively active participation (e.g. voting and appealing) on behalf of its token holders; ultimately it relies upon a democratic model where the truth is assumed to be a Schelling point upon which voters can be expected to settle, as long as the system minimizes their ability to effectively collude. In cases of dispute, Augur is expected to reach outcome resolution over the course of days or months.

Oraclize

Oraclize act as a data carrier, a reliable connection between Web APIs and Dapps. They use cryptographic proofs to prevent the need for additional trustlines.

They are a data railway and provide valuable tools and services designed to connect oracles (data providers) with distributed applications. In their own words, they are a "platform-agnostic bridge between the blockchain and the Internet".[19]

Oraclize uses TLSNotary proofs[20] to achieve a reliable bridge between smart contracts and Internet servers. They contrast their approach with decentralized oracle networks like so:

> One solution is to accept data inputs from more than one untrusted or partially trusted party and then execute the data-depend[e]nt action only after a number of them have provided the same answer or an answer within some constrain[t]s. This type of system can be considered a decentralized oracle system. Unfortunately, this approach has severe limitations:
>
> - It requires a predefined standard on data format
> - It is inherently inefficient: all the parties participating will require a fee and, for every request, it will take time before reaching a sufficient number of answers.
>
> The solution developed by Oraclize is instead to demonstrate that the data fetched from the original data-source is genuine and untampered. This is accomplished by accompanying the returned data together with a document called authenticity proof. The authenticity proofs can build upon different technologies such as auditable virtual machines and Trusted Execution Environments.[21]

Oraclize services are especially suitable for situations where centralized data servers are acceptable oracle solutions. In other words, if a smart contract needs to ask a question of the form: "What content does a particular webserver $\boldsymbol{S}$ provide at time $\boldsymbol{T}$ when query $\boldsymbol{Q}$ is submitted?" then Oraclize provides excellent mechanisms to provide and verify an answer.

Even in situations where decentralized oracle solutions are needed or preferred, Oraclize can be integrated with and their work can be meaningfully leveraged. They are approaching a different side to the same problem, so their technology can be considered complementary to any distributed oracle platform that incorporates it (e.g. Delphi Systems).

Prediction Markets

> A prediction market is a powerful idea.
> A decentralized prediction market is an even more powerful idea.[22]

Prediction markets are platforms where participants are capable of creating, managing, and exchanging financial shares in *outcomes* or *events*. Put more simply, they are systems which allow people to make bets, and receive compensation if they are correct. The bets could be about anything (the winner of a presidential election, the price of gold at some date, or the final score of a particular sports match, to list a few examples); the important point is that being right earns rewards. "Skin in the game" not only motivates performance,[23] it attaches an incentive directly to honesty and truthful information sharing, which is ultimately the most reliable way to ensure that information becomes publicly available.

> People are fundamentally incentivized by money, and a system that allows people to monetize by betting correctly on outcomes drives better information in the world. This

> information could be used to create better policies, to build better businesses, and to broadly increase aggregate progress by humanity.[24]

It is extraordinary how effective prediction markets can be. They consistently outperform most sophisticated benchmarks,[25] exhibit unlimited scalability and can assist in the aggregation and distribution of unlimited quantities of information,[26] and are highly resistant to manipulation.[27]

In political forecasting, prediction markets have consistently yielded "extremely accurate" predictions and outperformed large-scale polling organizations. They have similarly been used to accurately forecast box office figures, award nominations, and product sales quantities.[25] They beat professional forecasters' measures of macroeconomic performance by an average of 5%.[28]

Prediction markets have also seen resounding success in the private sector, forecasting Google's IPO price better than Google did with its auction mechanisms, outperforming Hewlett-Packard's official forecasts by as much as 70% in some cases, and continuing to be used internally in a myriad of ways by several multi-billion dollar corporations and interests.[28]

Finally, at least in some cases, prediction markets can help save lives. There have been multiple documented cases of prediction markets providing better and sooner forecasts of influenza and fever outbreaks than alternatives were capable of,[28] which is the sort of time-sensitive data that can be critical to the public health sector when it comes to the proper handling of and response to the spread of contagious diseases.

In a nutshell: prediction markets can open the pathways that allow and encourage information to flow freely, make it in everyone's best interest to express and communicate the truth, and grant society newfound powers of rationality and insight.

Paul Sztorc, one of the pioneering philosophers and researchers into cryptocurrency-based prediction markets, expressed it well:

> What if you had access to the combined intellectual powers of all mankind? It would be easier for you to make decisions. You'd know what school to attend (if any), where to live, where to work, what to buy, and how to save/invest. You'd more quickly become aware of new medical treatments, unethical behavior within business/government, terrorist threats, societal problems, and of the consequences of a given law, scientific endeavor, or industrial achievement. The economic-technology that makes this possible is called a Prediction Market.
>
> The printing press helped set the stage for first Scientific Revolution, but it took a new (and heretical) way of looking at information – Empiricism – to make what was printed have the impact that it did. Similarly, we today have the internet, drowning us in information sources. What is broadcast is less useful than it would otherwise be if we could reliably combine Multiple Sources into One Truth. We need a new (and taboo) way of looking at information today! Viva la revolución![26]

Clearly, prediction markets possess an impressive track record, and have already demonstrated great utility in recent history. Even so, we have only begun to scratch the surface of the true promise of this technology. The informational advantage that would be made possible by global, liquid, and low-friction prediction markets would unlock countless possibilities and benefits, perhaps even one day allowing us to upgrade democracy itself.[29][30][31]

Existing Approaches

Multiple projects have begun work on Ethereum-hosted implementations of prediction markets. Augur, Gnosis, Delphy, and Stox represent a few notable examples, each with different characteristics, approaches, and tokens. Augur implements a monolithic network-wide oracle which is designed to converge on true outputs via a cryptographic commit-and-reveal scheme involving the platform's token holders. Gnosis implements ERC-20 tokens to represent event outcomes, and generic oracle interfaces for data input, as well as a bid-quantity-based arbitration fall-back mechanism. Delphy similarly implements oracle interfaces, but also offer a light-client/mobile-centric approach with event filters and social application functions bundled in. Finally, Stox uses Bancor as a token platform for liquidity, pays out and operates exclusively via the STX token, is oracle-agnostic, and allows for event syndication and promotional credit to help incentivize participation.

With the exception of Augur, each of these platforms will be fundamentally compatible with oracles deployed using *Pythia* and other Delphi tools.

Delphi Systems

Pythia

The Oracle Framework

Abstract

The concept of a multisignature smart contract has been explored and modeled extensively,[32] and even basic multisignature contracts possess incredible and revolutionary potential when it comes to maintaining a useful power balance in decentralized systems. Multisignature schemes can provide a means of checks and balances in distributed contexts, preventing any one party from exercising too much control over the others.

We propose an additional innovation beyond a basic multisignature arrangement, when it comes to the determination of the oracle input on a given smart contract. A weighted signature framework called *Pythia* is provided, which provides a host of benefits over naive multisignature majority approaches, including understandable oracle interfaces for developers to work with, quicker estimated input arbitration, and additional flexibility and extensibility moving forward. We consider this to be the most sophisticated and promising distributed oracle framework envisioned to date.

Philosophy

Our approach to the oracle question is based on a few core philosophical tenets:

1) Reliance on a single centralized oracle can, in many cases, entirely defeat the purpose of adopting a smart-contract architecture at all, by significantly reintroducing the need for trust. For any smart contract which relies on external data of any kind, the contract is only as decentralized as its data provider or oracle. Therefore, any oracle "solution" that involves allowing a single entity to unilaterally decide the result is effectively a total regression to centralization; such a regression should be considered irrational, and represents no solution at all.

2) Providing a quality oracle service entails a non-zero cost to the service provider, so any sustainable solution must involve oracle compensation in one form or another. In other words, oracles cannot be expected to work for free, and will necessarily require an incentive or reward.

3) Given the above, the fewer oracle service providers involved for any given event resolution or input determination, the lower the expected cost-of-service should be. To truly minimize oracle-related costs, the system should allow for "localization" of the oracle function; requiring universal (or widespread) coordination or consensus for each oracle event is fundamentally expensive.

4) Open, free-market competition is the best known mechanism for minimizing aggregate system costs over the long-term. The "invisible hand" of the market is an unrelenting force of optimization, and leveraging this phenomenon (to the extent possible) represents the best possible chance of ensuring eventual platform success. It is much better to allow the market to work for you, rather than trying to individually outcompete it.

5) Ideally, modularity and isolation/insulation of oracle instances should be possible. This way, local problems don't necessarily incur global costs. Availability and security are maximized while systemic inertia is minimized, facilitating innovation and optimization.

6) Data formats must be considered carefully. Ultimately, data can come in a variety of formats, so it would be best to provide accommodations for different data formats (and the conversion and processing thereof).

The best way to achieve the above criteria is to build a *toolkit* and *framework* rather than a *specific solution*. We aim to set the stage for rapid prototyping, development, and experimentation. By laying the foundation, making sure the tools are user-friendly, and then letting the world at large explore what is possible from there, we stand a real chance at collaboratively constructing valuable decentralized oracle solutions. We believe that fair and open competition between selfish participants, with well-understood and rigorously expounded contract models and relations, is the most reliable way to ensure that such progress can occur.

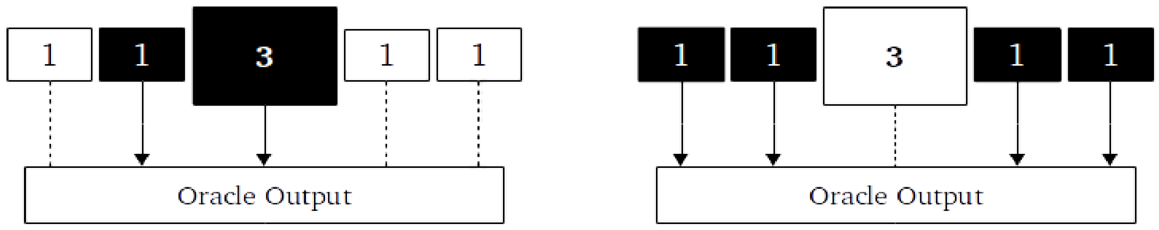
Basic Design

Pythia is a distributed oracle framework, launchpad, and tool-suite. In other words, Pythia is a platform that users can build and deploy distributed oracles from.

At the most basic level, these distributed oracles will rely on multisignature contracts, where authorization from more than one entity is required to generate an oracle output. Beyond the usual M-of-N multisignature arrangements, however, Pythia accommodates signature weights and thresholds. With these simple primitives, an endless variety of oracle arrangements are possible, ranging from straightforward arrangements to incredibly-complex schemes.

To demonstrate the principle in an understandable manner, it can be illustrative to walk through one of the simplest and most minimal oracle schemes possible in Pythia.

We establish an arrangement of **5** oracles total; four of these oracles' signatures are assigned a weight of **1** and the final oracle is ascribed a signature weight of **3**. For an oracle output to be considered valid in this arrangement, it must meet (or surpass) a weight threshold of **4**.



In this arrangement, the "alpha oracle" with the signature weight of 3 is able to cosign an output with *any* of the other oracles to render that output valid. However, the alpha oracle *acting alone* is unable to produce a valid output (it can sign with a weight of **3**, but the weight threshold is **4**). Absent the cooperation of the alpha oracle, the "subsidiary oracles" are still able to produce a valid output, but *only by reaching unanimous agreement*, because it would take all 4 of their singly-weighted signatures to reach the weight threshold.

This simple oracle arrangement example already possesses a few interesting and desirable properties. No single party is able to unilaterally make an output decision, so the arrangement is not totally centralized. There are reasonable checks and balances in place, keeping power meaningfully distributed rather than concentrated. In most cases, valid outputs can be generated with the *bare minimum* number of signatures required in a distributed-oracle context (namely, two), meaning that service costs can be minimized; there is no need to bother or coordinate with more oracles than are strictly necessary for the process, and signature transmissions and verifications are kept to a minimum. There are no consensus-critical complexities to factor in. In fact, the only form of consensus necessary is trivial: as long as sufficient signature weight is provided, the output is considered valid.

The prior example is still relatively limited and by no means represents a perfect and universally-applicable oracle solution, but it does demonstrate some of the power that the Pythia framework can provide, even in minimal-setup contexts.

Beyond the Basics

The Pythia framework is built with extensibility and flexibility in mind. More functions, variables, and even feedback loops can be implemented to suit different needs and use-cases.

Pythian oracles can be programmed such that the balance of weights (and the threshold required) is modifiable under certain conditions. This functionality can be leveraged in many elaborate ways; each successful oracle output could be accompanied by a “re-weight vote” which determines the relative signature weights for subsequent outputs. This scheme would allow the distributed oracle to internally regulate its signature weighting and even “vote out” misbehaving oracle participants. Alternatively, the oracles involved could have *decaying weights* that must be periodically replenished, e.g. via furnishing truthful outputs (and verifiably receiving particular tokens as a result). The determinants of the re-weighting could be internal or external, and meta-oracle contracts could even be devised to handle these processes. In other words, distributed oracles might be deployed *solely to manage other distributed oracles.*

Additional checks and balances can be introduced, to mitigate the externalities of oracular misbehavior or dishonesty. A simple example would be the enforcement of an additional Sybil-resistant democratic approval mechanism to enable an oracle’s output generation. Such checks and safeguards could be phased in (or out) of Pythian contracts’ state, dependent on other factors.

With proper data interfaces and standards established, supplementary contracts for oracle output diffusion would enable blind contract settlement, which could protect outputs and oracles under freedom-of-speech ordinances. Signatures can be handled off-chain, via layer-two technologies, or even using blockchains other than Ethereum’s. Iterative processes (e.g. commit-and-reveal schemes, or the ability for participants to challenge outputs within a certain window of time) can likewise be incorporated. Eventually, systems as complex as Augur could be built and deployed through Pythia (though it is likely that more lightweight and efficient alternatives ultimately come to predominate the oracle ecosystem).

Although the foundation for Pythia has been laid, further research, and in particular a rigorous dissection and exploration of the various possibilities within the framework (e.g. introducing time- or contract-dependent decay functions to the relative signature weights) will be a continued and abiding focus in Delphi.

Advantages

The approach we have taken with Pythia is to concentrate on building an extensible and modular distributed oracle framework, formally model the possibilities within that framework, and continually strive to improve the quality of the modules and tools that it is built on. This approach offers many important benefits.

Contracts produced by Pythia represent independent oracle *instances*, rather than interdependent consensus elements. This means that the specific properties of individual oracles are not *consensus-critical*, which allows incremental development to safely occur; upgrades will be almost entirely frictionless, and much more straightforward to deploy and manage than in systems which rely on continual network-wide consensus. It also means that parallel code branches and modules can be maintained without issues, allowing extreme customizability and providing the ability to target niche use cases. Usage of any given oracle is completely opt-in, so developers and users can choose the specific features they want to take advantage of, without burdening the ecosystem as a whole. In summary, Pythia is built specifically to facilitate and encourage permissionless innovation.

Over the long term, free market competition is expected to elicit more successful optimizations and additional features than any one team or organization ever could. Thus, Pythia is architected around modularity, flexibility, and insularity. This means that both costs and risks can be localized and minimized; it is an inherently decentralized model, where users only bear the risks that they willfully assume. In tandem with robust rating and reputation metrics and algorithms, this lays the foundation for a radically open and free marketplace of ideas and implementations. Some users may only feel comfortable using the simple, minimal oracle deployments that rely on battle-tested and heavily-audited code, while others may want to make use of more advanced and experimental contracts and features. Pythia allows both groups to get what they want, without imposing type-irrelevant risks or restrictions on either.

An open-source development model combined with robust formal analyses of the different possible assumptions, trade-offs, security guarantees, limitations, trust matrices, and feature implications within the Pythia framework will help to mature the industry and push the state of the art forward in a significant way.

Finally, by developing a powerful and modular suite of tools and emphasizing maximal compatibility, capital and effort invested into Pythia will be generally applicable to any system, network, or contract model that would benefit from a decentralized oracle component. Many promising and revolutionary projects built on Ethereum will inevitably depend on such a component in one way or another, and beyond that, Pythia will offer more than just Ethereum support. In principle, the utility provided by Pythia as a platform can be realized on any blockchain or system that supports multisignature outputs. In some cases, potential functionality might be marginally impacted by the capabilities of the host chain and its supported transaction formats, but in most respects, the benefits of Pythia are universal in nature and represent public contributions. As one notable example, the development and maintenance of Bitcoin-oriented tools and technology within Pythia is a definite project priority.

Summary

Our goal with the Pythia framework is not to build *one particular* solution, but instead to provide (and help to explore) tools and models to build from when it comes to the design of and interaction with distributed oracles. The individual oracle deployments can be small and static units involving very few parties (as in the basic example depicted above), or large complex networks that evolve over time; they can be specific and ephemeral (created to generate a particular output) or general and persistent (offering oracle services on the open market and competing with others doing the same thing). This approach avoids commitment to one particular oracle model (which would almost certainly be made obsolete by the inevitable breakthroughs that future generations will make), and instead positions Pythia as a future-friendly oracle-production platform that can fluidly incorporate upgrades and innovations as they are discovered and developed.

Φ

PHI / sPHI

The Compound Token

Purpose

Oracles deployed via Pythia can be paid in PHI, the platform's token. This is an app-coin with specific uses: it is intended to compensate oracles (via fee payments) and to facilitate a flexible platform signaling mechanism. Users can indicate oracle or demand preferences, compositing into an oracle reputation system; future applications built on or around the Delphi stack can make use of the token's signaling mechanism, including Delphi's own *Agora* project.

The token is not intended as an investment vehicle, and we can make no assurances that any development effort we provide will increase the value of the token in any way. Furthermore, the tokens are not representative of equity or a binding governance contract. Instead, PHI tokens represent application-specific artifacts, intended to facilitate the best possible user-experience of our oracle tools and further development and refinement of these tools, and should *not* be viewed or treated as an investment nor as shares in a company. The value will not be pegged to anything, and will thus be susceptible to the laws of supply and demand (over which we, as a team, have no control). We cannot guarantee any particular value for the token to retain, despite our best efforts as a team to build and maintain this project.

For those interested in using and experimenting with Delphi's oracle tools and applications, PHI is the official platform token.

The compound token is implemented via three interlocking contracts: the *minimal token*, *signal token*, and *trustless peg*.

Minimal Token : PHI

The *minimal token* is a standard ERC-20 token contract, implementing only the default token functionality. It implements atomic approvals, protecting against approval-related race-condition vulnerabilities that are present in many existing token deployments.[33] It also incorporates a special-case transfer protection (which does not affect the transfer method signature), to prevent accidental direct transfers to the *peg* contract.

As a basic ERC-20 token deployment, management of PHI will be straightforward and familiar to those who have used Ethereum tokens before. The code is easy to audit and has been subject to considerable and extensive peer-review in the industry. This token contract will be maximally compatible with existing wallets and services, and automatically recognizable and classifiable even by software that uses hard-coded ABI settings.

Beyond the benefits of security and compatibility, the minimal ERC-20 token design is also extremely *efficient* for transfer operations, meaning that gas costs of balance updates are kept as low as possible. Transactional friction is minimized, which means that users who are not interested in taking advantage of the advanced snapshot and functionality of the *signal token* are not burdened by the on-chain costs associated with enabling this functionality. The minimal token contract is therefore the secure, familiar, lightweight, efficient, and inexpensive component in the compound token architecture.

Signal Token : sPHI

The *signal token* contract is based on the MiniMe token design[34], which is an ERC-223[35] (and ERC-20 compatible) token implementation that incorporates advanced functionality including balance-snapshot archives, direct token clone support, and optional token controllers.

This component is a much more flexible and feature-rich token implementation relative to the basic ERC-20 design. Balances are continually tracked via snapshots, which allows for quick and direct balance queries per block height. This introduces a wide range of additional functionality and desirable properties. Because balances can be seamlessly tracked over time, meaningful metrics can be easily generated from these snapshots at any point. The sPHI tokens do not have to be locked up or frozen in order to be used for voting or other forms of token communications, enabling a much more practical and usable signaling workflow. This makes the tokens especially suitable for reviews, rating systems, and polling purposes, providing a flexible and Sybil-resistant mechanism by which to establish and update reputation values and other relevant market indicators.

The signal token contract is also more cleanly extensible and upgradeable than that of the minimal token. The ability to generate a cloned child-token contract (with added or modified features from the cloned parent token) from the balance snapshots available is a powerful feature. This acts as a strong catalyst for permissionless innovation; spin-off tokens are not only possible, but inherently encouraged. The possibilities of this functionality are extensive, and once again the approach serves to insulate the users who are not interested in the extra features provided; all advancements and upgrades are *opt-in* by nature and will not burden existing token holders. Additional risks or complexities introduced by derivatives of the token will not propagate upwards to the cloned parent in any way.

The ERC-223 standard which the signal token adheres to provides efficiency benefits in certain common token processes, prevents some forms of blockchain bloat, allows protections against accidental token loss, standardizes the token transfer functionality considerably, and allows incoming token transactions to be handled elegantly, which is not possible with typical ERC-20 tokens.[35] In simple terms, the standard effectively introduces the functionality of native ether transactions to custom token transactions.

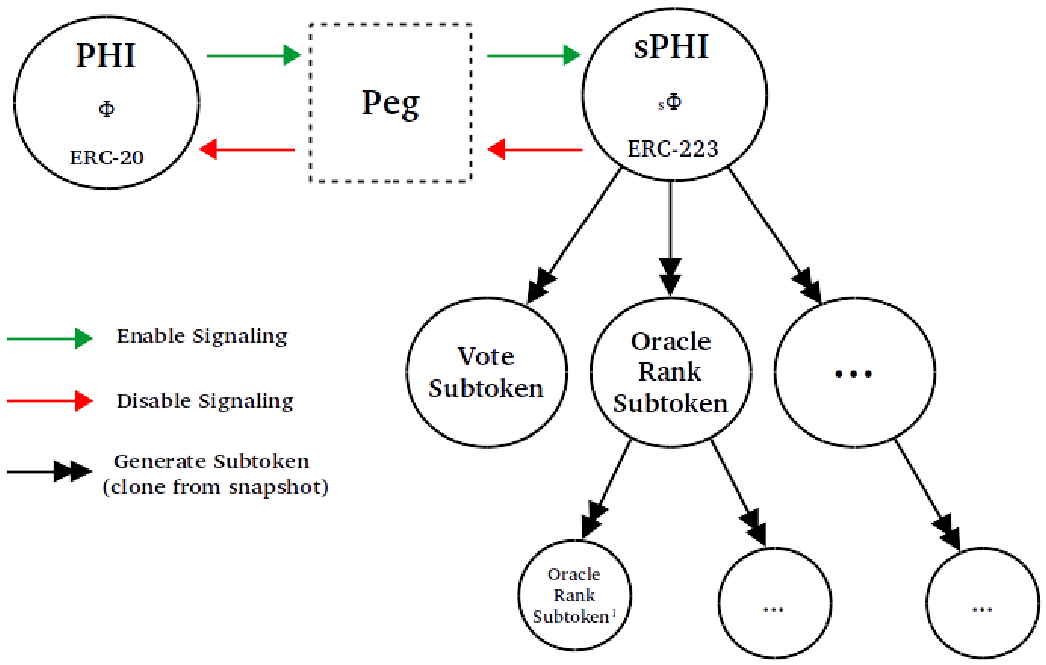
Although the signal token is considerably more powerful than its minimal token counterpart, the added functionality incurs some trade-offs worth mentioning. Most notably, the gas costs of transferring the signal token are significantly higher than those of the minimal token, due to the added overhead of balance snapshot updates. This means that, on average, transfer operations of the signal token will be more expensive, so in situations where large numbers of transfers are expected or needed, the minimal token is a more appropriate tool for the job. The MiniMe token reference implementation also introduces a significant increase in overall code complexity, and has not experienced the same levels of code review and security analysis that the ERC-20 standard has. Finally, because the standard isn't as widely adopted or used, some applications and tools might not be set up to recognize the signal tokens in the same way that they are able to recognize the minimal tokens; there is a trade-off incurred in terms of compatibility with the existing ecosystem.

The signal token is superior to the minimal token in some respects, but inferior in some others. This provides the motivation for a compound token architecture, combining both standards into a single token implementation via a trustless token peg contract.

Token Peg

To realize the full benefits of both token standards, a *trustless token peg* contract is used. This contract enables the best of all worlds, by allowing tokens of either type to be converted to the other on demand.

A compound token is deployed via a factory contract, which will instantiate the minimal token contract, the signal token contract, and the peg contract linking the two. The supply of minimal tokens is distributed normally, while the full supply of signal tokens is provided to the peg contract. The peg contract is programmed to recognize and verify incoming transfers of tokens of either type, and automatically provide the sender with an equivalent quantity of tokens of the other type.



Note: as a protectionary measure against accidental direct-transfers of the minimal token to the peg contract (the receipt of which is not inherently recognizable), these transfers are disabled altogether in the minimal token implementation. This means that the only way to successfully transfer such tokens to the peg contract is via a multi-step approve → transferFrom process (i.e. by invoking the contract itself). This ensures that the peg contract does not mistakenly receive or consume minimal tokens from users without providing a corresponding payout of signal tokens.

This contract serves to unify the two standards into a single *compound token* architecture, which avoids fracturing the economy or forcing commitment decisions on users or developers.

A full compound token deployment provides users the ability to enable (or disable) the token's signaling mechanism through the peg contract. In effect, users can invoke the peg contract to update the *state* of their tokens at any time.

Summary

The PHI compound token construct is designed to actualize the full range of benefits of two different token implementations into a single, flexible system. The *minimal* component is designed to be efficient, lightweight, well-vetted, easy to use and understand, and maximally compatible with existing token solutions. The *signal* component provides the rich functionality necessary for market signaling, seamless and permissionless feature upgrades, and tracking values or rankings over time. Finally, the *trustless peg* component unites the two feature-sets into a single token architecture, allowing users to freely toggle their available token balance, in part or in full, between the two states.

Extensive and profound functionality is possible with this design. Most importantly, the oracle ranking and reputation systems of *Agora* will heavily incorporate data sourced via the token's signaling component. Ad hoc polling initiatives will similarly benefit from this component, allowing users of the platform to identify their preferences without locking up their usable token balances. The token therefore has utility and influence within the contexts of the Delphi platform, and as such, should at minimum enjoy the demand of its competing oracles. This means that while it is up to any given oracle to stipulate what sort of payment is required for their services, payments denominated in PHI make rational sense and will be natively supported, just like normal ETH payments are. Succinctly, PHI can be thought of as the Oracle Coin.

Importantly, the token is designed as a *tool*, rather than as an *investment* or form of *equity*. Any hypothetical value it ultimately enjoys would be a result of its utility (as well as a myriad of factors that we have no control over). While we intend to do everything we can to make the Delphi suite (including the compound PHI token and the derivatives thereof) as valuable as possible to users and to the industry at large, we cannot ensure that our work will increase the price or market value of the token, and we can make no promises or commitments to that effect. In the end, a substantial portion of any value captured into the tokens will depend on the efforts of disparate contributors from around the world, many of which will not be affiliated with Delphi as a team or organization in any direct sense. We are working to build powerful frameworks, tools, and foundations precisely because we believe that open-access free-market competition will yield the best solutions in the long term. This is a critical and defining factor of the general project strategy.

The compound token structure captures all the benefits of two distinct state-of-the-art token implementations, without incurring any downsides or drawbacks. As such, it represents one of the most sophisticated and powerful token designs on Ethereum, and warrants serious consideration as a candidate architecture for other projects in the space that prioritize a "best of all worlds" token solution.

Omphalos

The Prediction Market Framework

Our own prediction market solution, *Omphalos*, has been designed with our oracle tools in mind, and serves as an extensible template by which unique and sophisticated prediction markets can be deployed and used.

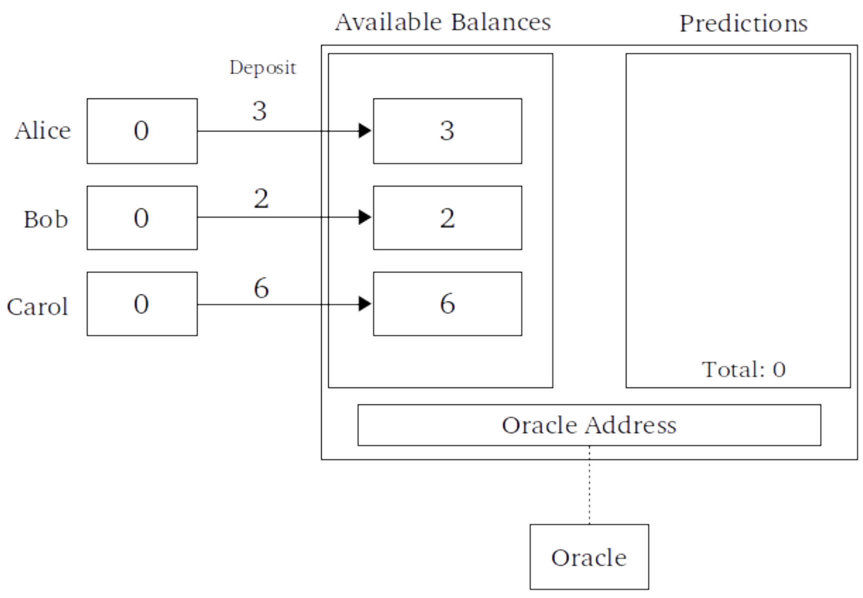
Core Principles

Fundamentally, a prediction market is a platform on which predictions can be made and where compensation is provided to those who predict correctly. Implemented as a smart contract, the core components reduce down to an oracle and three user functions: *deposit*, *predict*, and *withdraw*.

Omphalos represents a basic template for implementing these functions and linking them to a particular oracle contract, with extended (optional) features and functionality maintained in parallel for more advanced or unique needs. In this way, just as Pythia represents a framework for deploying distributed oracles, Omphalos can be viewed as a framework for deploying instances of prediction markets.

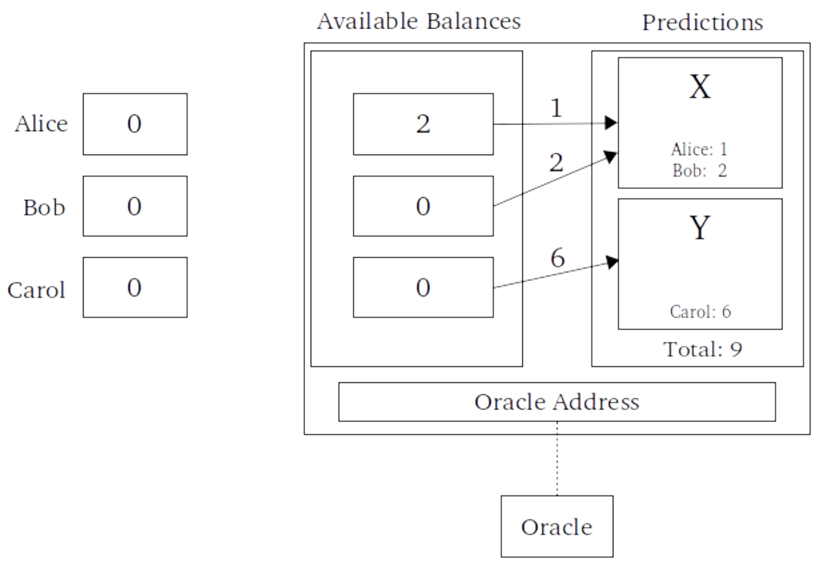
Basic Design

The basic contract flow is straightforward. First, the contract is created, linked to a specific oracle. Users can then deposit funds into the contract:



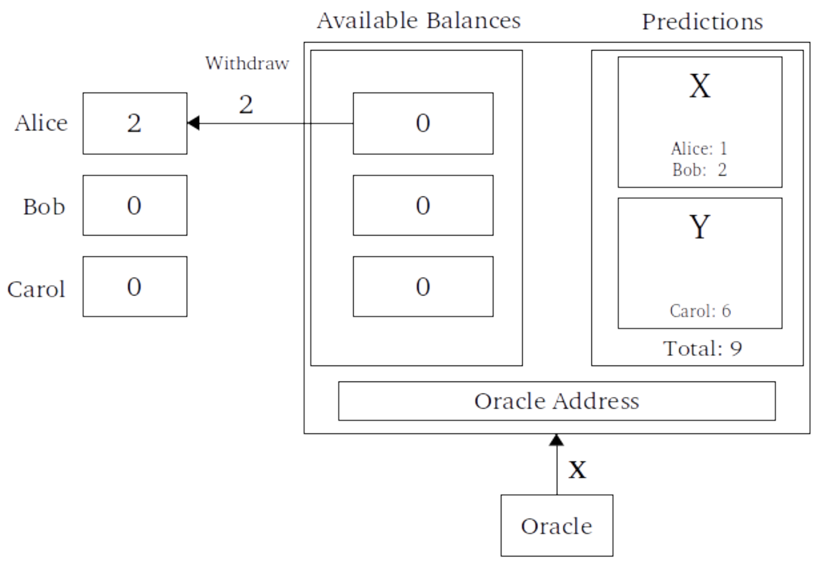
Alice deposits 3 ETH.
Bob deposits 2 ETH.
Carol deposits 6 ETH.
No predictions have been made yet.

Users are free to withdraw their deposited funds, or alternatively to use them stake predictions (which renders the staked balance unavailable until contract resolution). The total predicted amount per user is tracked by the contract.



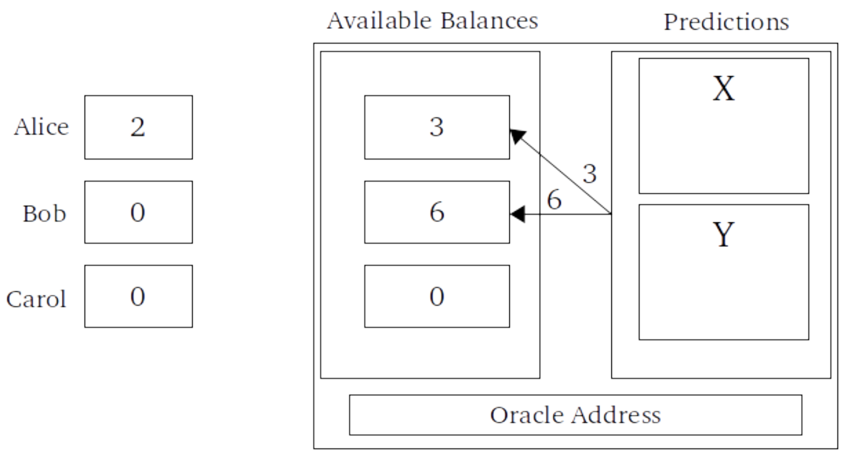
Alice predicts **X** with 1 ETH.
Bob predicts **X** with 2 ETH.
Carol predicts **Y** with 6 ETH.

In the basic example, once the oracle provides outcome data, the contract is considered finalized:



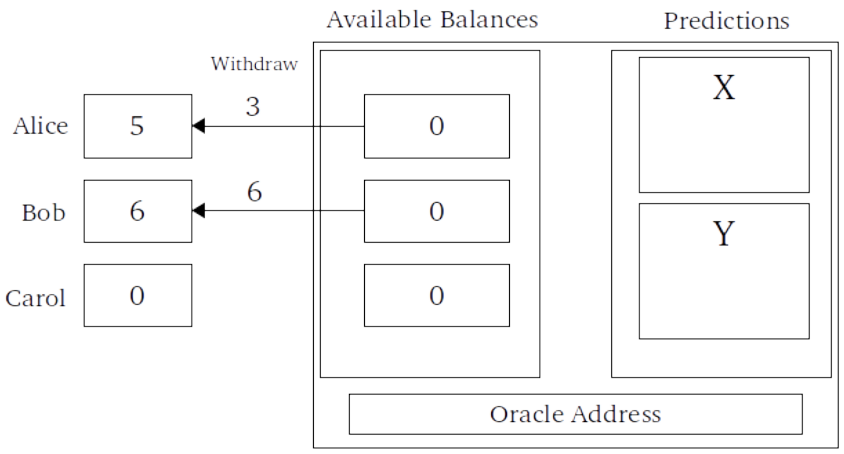
Alice withdraws 2 ETH from her available balance.
Oracle finalizes **X** as the correct outcome.

After the contract is finalized, the cumulative predicted funds can be distributed to the users who issued correct predictions (in the simplest example, in direct proportion to correctly-staked balance).



Alice & Bob receive payouts to their available balances.

These users are then free to withdraw their balances normally:



Alice & Bob withdraw their available balances.

Beyond the Basics

As with Pythia, though the above example demonstrates the core operating design of an Omphalos contract, the template is profoundly extensible. Optional features such as pre-finalization prediction lockdown, temporal-based reward-weighting (and other market scoring approaches), and multi-round or multi-pool instances are all possible, and will be maintained as distinct branches of the master codebase.

The determinants of a particular prediction market's behavior could either be specified internally or provided via an external source, e.g. another oracle. Omphalos deployments can accept a degree of "oracle agnosticism", stipulating criteria to be satisfied rather than a specific linked oracle contract. Additional intermediate contract states and levels of interactivity can be incorporated as needed.

Benefits

Because Omphalos is not tightly coupled to PHI (or any other token), users can deposit, predict with, and withdraw ETH directly. This opens up a massive pool of liquidity that is unavailable to token-dependent prediction market implementations, and simplifies the design considerably. The model is able to reach a state of settlement as quickly as outputs can be furnished by the oracles involved; it is radically efficient.

Consensus-related complexities are once again sidestepped, resulting in an understandable and straightforward high-level design with incredible potential for customization. In this way, Omphalos is another attempt to leverage the advantages of a flexible framework; the individual instances of Omphalos contracts leave unencumbered any users beyond those who choose to use the prediction market in question. Like with Pythia, this approach allows experimentation and optimization by the free market, keeping risks localized while facilitating open innovation.

Tholos

User-Oriented Ideology and Designs

"**User experience is everything.** It always has been, but it's still undervalued and under-invested in. If you don't know user-centered design, study it... **Obsess over it.** Live and breathe it."[36]
~Evan Williams, Twitter Founder and CEO

Vision

The data shows that it is absolutely critical to create an enjoyable experience for the user.[37][38][39][40] Companies like Apple have shown that attention to detail when it comes to the impression that your products make on users makes all the difference. We believe that the field of UX has been woefully under-emphasized in the burgeoning Blockchain economy, and that most efforts and initiatives in the space have dedicated too few resources towards answering the question: "How can we improve the experiences that our users go through?"

Furthermore, we do not believe that this is a secondary or purely-cosmetic concern. The levels of adoption we aim to reach can only be achieved by devoting appropriate attention and emphasis into the dimension of user-interactivity and experience.

Tholos represents a vision of rich end-user experience; every interaction that a user has, at any level of the stack, should leave them feeling satisfied. We want to offer the community our full, honest, continual support. We want our products and tools to be beautiful and to function smoothly, at every step of the way, and we do not believe that this would be possible to achieve if organizational resources and capital are not wholeheartedly dedicated towards this end.

There is clearly a long road ahead before this industry is fully matured, in terms of offering quality end-user experiences. The only realistic way to ensure that this condition is improved is to commit fully towards getting it done. How users feel about the platform and its tool-suite, and how to make them like it even more, should be the organization's guiding principle and top priority.

Beyond Surface-Level Aesthetics

The user experience consists of much more than the pixels that reach users' eyes. Although aesthetics and visual appearances are important in terms of achieving the right impression, other factors can be just as important in determining whether users are satisfied with their experiences and interactions with the technology.

At Delphi, we are committed to building and supporting straightforward and well-explained technologies moving forward. We intend to incorporate community and user feedback as much as we can at every step of the way. We have even worked to make sure our token is capable of communicating user sentiment as appropriate. We are also open to exploring other innovations that may serve to bolster user satisfaction, regardless of whether the improvements or optimizations are visual in nature.

Tools

Tholos will be comprised of multiple different applications and components, ranging from templates and other reusable graphical assets, to full-fledged standalone GUIs. It will include data and visualization libraries, APIs, and extensive developer documentation to accompany these resources.

Existing interfaces and libraries will be leveraged and integrated with, which may necessitate the development of certain types of middleware. Even developer- and researcher-friendly tools built to aid with design, debugging, and analysis will be released under the umbrella of Tholos.

Long-Term, Moving Target

As with most complex design goals (such as the scaling of distributed networks), we will likely never be able to declare the job done, once and for all. Rather, the process is expected to be an iterative one, providing a moving target as the market evolves and matures. This means that in a general sense, Tholos represents the full commitment to improving the experience of users over the long-term, as the platform flourishes and grows and the market circumstances continue to change.

Agora

The Oracle Marketplace and Ecosystem

Abstract

Agora represents the culmination and synthesis of all the components of the Delphi Systems project into a cohesive decentralized oracle marketplace. It is our most ambitious and long-range initiative, combining the individual elements of Delphi into a single unified suite and vision. The foundations we have laid, from the flexibility offered by our framework-based approach, to the signaling and metric-oriented functionality of our compound token, and even our total commitment to a satisfying user-experience, have been oriented around the eventual construction of a full-fledged oracle ecosystem: Agora.

Platform

While Pythia provides the ability for anyone to deploy a distributed oracle contract from anywhere in the world, there are no guarantees regarding the honesty or integrity of such an oracle. This is a serious issue, because without a trustworthy and reliable oracle (whether it consists of a single party or a complex distributed service), a smart contract can be considered totally compromised or even useless. What is needed is a transparent way to review, rate, and rank the different oracles that are produced in the framework, such that reputations and service records can be established and evaluated on demand.

The compound token (PHI) lays the foundation for such rating systems, allowing users to aggregate quantifiable data regarding their oracle experiences and actively communicate their satisfaction (or dissatisfaction) in a trackable way. The token enables a practical mechanism for user-generated feedback to be collected, which can subsequently be parsed and visualized.

The token signaling provides the source of the data, but more tools are needed for the interpretation and handling thereof. The lack of a rigid consensus model at the core of Delphi implies that data analytics will play a crucial role in the system. Self-interested actors will inevitably try to game the system for their own benefit, and to protect against exploitation, high-quality data applications will be employed.

As a straightforward example, consider the scenario in which a well-funded entity decides to deploy a malicious oracle contract, intent on defrauding users by furnishing untruthful outputs on a prediction market contract. This entity might purchase or otherwise acquire a large quantity of PHI before linking the oracle and prediction market contracts, and then use the tokens to self-signal for their oracle contract. In a naive model where instantaneous token signal weight is the only evaluated metric, this attack would potentially prove quite effective. However, it can be protected against through robust time series analysis (which is made trustlessly possible through the sPHI snapshot system) and the inclusion of other variables and factors in the rating system.

Custom user-configurable data filtration will be an invaluable tool of reputation management and assessment. Agora will include filter templates and toolkits, allowing users to develop their own specifications and ranking criteria and to share their work with other users as desired. Our team will maintain repositories of different rating algorithms and presets, designed to be resistant to exploitation or abuse. Agora will include prioritization options reminiscent of search engine ranking systems, and in fact serve as a type of “oracle search engine” itself. Users will have access to oracle listings describing the properties and histories of different service providers, to inform their judgment on the matter.

Beyond the tools centered around signal data management and analysis, Agora will also include compatibility-oriented tools to minimize friction between Delphi and other projects in the industry, like Oraclize. APIs and standalone tools (e.g. for data format conversion or external service integration) will be provided, as well as user-friendly graphical interfaces built to lower the barrier of entry and increase participation as much as possible. A thriving marketplace relies on the network effect, which is maximized through making it easy to work with (and build on) the platform in question. On a high level, Agora represents our attempt to do so.

Strategy

The Pythia oracle framework is a linchpin of our strategy with Agora. By providing robust distributed oracle models, tools, and formal analysis, we expect to lay the foundations for a thriving oracle ecosystem.

Furthermore, we will offer our (reliable and honest) oracle services via Pythia and Agora, not only to bootstrap the distributed oracle network, but as a core revenue stream for Delphi Systems. This will allow us to achieve our project vision (and ensure that there is always at least one useful and difficult-to-compromise oracle service provider available) while also financing our efforts along the way.

This strategy also provides the benefit of continual firsthand experience with our own tools; this will allow us to refine and improve the end-user experience over time, as we go through the same processes that other users of our applications do and are able to generate feedback in-house. A defining characteristic of our vision is that we will be especially responsive to the needs of the community, and by regularly using our own products, we will be able to empathize with and properly understand the suggestions, criticisms, and comments of our users. The token signaling functionality will also provide a practical and straightforward means of community polling; though such polls would not represent binding governance agreements, per se, they would be Sybil-resistant mechanisms to gauge user sentiment and plan or adapt accordingly. This would establish a technical means of "keeping our finger on the pulse of the community" and maintaining an open dialogue between developers and users of the Delphi platform.

Both within the Ethereum economy and beyond, we will attempt to fill the niche of the distributed oracle, offering the "Truth as a Service" and positioning our project such that the majority of smart contracts reliant on external data (regardless of their platform of origin) are ultimately arbitrated via oracle solutions we have developed and provided.

Conclusion

Vision of Delphi

Catalyze the flow of information and harness the power of truth.

Right now, there is a glaring gap in the blockchain frontier (and in Ethereum in particular): no sound distributed oracle solutions exist. This prevents all kinds of interesting projects and applications from being possible, and limits the utility of the blockchain considerably. At Delphi Systems, we aim to fill this gap.

Information has value is a natural corollary to the age-old adage of *Knowledge is power.* In much the same way that mass and energy are equivalent, or two sides of the same coin, so too are information and value. However, there can still be friction associated with state changes between the two. Therefore, although blockchains already serve as very useful tools when it comes to the distributed transfer of value, there remains significant untapped potential in the realm of proper information acquisition, management, and transmission.

By formally modeling the systems we build and use, we can refine our treatment of information and share knowledge with one another like never before. We can truly capitalize on the tools we have while building the next generation of tools along the way. We will take on the oracle problem, establish platforms through which profit can be had via furnishing the truth, and work to leverage these platforms to their fullest potential ourselves. Our oracle technologies will come to power the realm of smart contracts, and we plan to take direct advantage of the opportunities they open up.

Quality oracle systems will unlock the full potential of smart contracts and blockchains, enabling incredible new applications and use cases (e.g. decentralized prediction markets) as well as simplifying or improving existing solutions and processes. The effects and benefits of this cannot be exaggerated. We therefore aim to build, maintain, and refine such truth-reinforcing technologies. By doing so, we can work towards an ideal of ubiquitous transparency and inimitable prescience.

In the immortal words of the Nobel Prize winning economist Friedrich Hayek, from “The Use of Knowledge in Society”:

> The economic problem of society is thus… how to secure the best use of resources known to any of the members of society, for ends whose relative importance only these individuals know.
>
> …practically every individual has some advantage over all others because he possesses unique information of which beneficial use might be made…
>
> …the ultimate decisions must be left to the people who are familiar with these circumstances, who know directly of the relevant changes and of the resources immediately available to meet them. We cannot expect that this problem will be solved by first communicating all this knowledge to a central board which, after integrating all knowledge, issues its orders. **We must solve it by some form of decentralization.** But this answers only part of our problem. We need decentralization because only thus can we insure that the knowledge of the particular circumstances of time and place will be promptly used. But the "man on the spot" cannot decide solely on the basis of his limited but intimate knowledge of the facts of his immediate surroundings. There still remains the problem of communicating to him such further information as he needs to fit his decisions into the whole pattern of changes of the larger economic system.

> The price system is just one of those formations which man has learned to use (though he is still very far from having learned to make the best use of it) after he had stumbled upon it without understanding it.[41]

Hayek had a deep understanding of the relationship between information and value, and saw that Humanity was just beginning our exploration into the full potential of the efficient and open distribution of knowledge. We think that he would approve of what Delphi is trying to achieve today.

We aim to minimize informational friction. We believe that the most profound and empowering gift that we can provide society is the free and open flow of information. Furthermore, we believe that functional distributed oracles, the applications they enable, and high-quality interface toolsets to interact with such systems have the potential to herald an information renaissance, and we intend to spearhead that movement.

References

[1] Nakamoto, S. *Bitcoin: A Peer-to-Peer Electronic Cash System.*
`https://bitcoin.org/bitcoin.pdf`

[2] Buterin, V. *Ethereum White Paper: A Next Generation Smart Contract & Decentralized Application Platform.*
`http://www.the-blockchain.com/docs/Ethereum_white_paper-a_next_generation_smart_contract_and_decentralized_application_platform-vitalik-buterin.pdf`

[3] Krug, J. *Augur Master Plan.*
`https://medium.com/@AugurProject/augur-master-plan-42dda65a3e3d`. Jun 9, 2017.

[4] Blockgeeks. *Smart Contracts: The Blockchain Technology That Will Replace Lawyers.*
`https://blockgeeks.com/guides/smart-contracts/`

[5] Consensys. *A Visit to the Oracle.*
`https://media.consensys.net/a-visit-to-the-oracle-de9097d38b2f`

[6] Szabo, N. *Smart Contracts Glossary.*
`http://www.alamut.com/subj/economics/nick\_szabo/smartC\_gloss.html` Alamut.com, 1995.

[7] Bartoletti, M., Pompianu, L. *An empirical analysis of smart contracts: platforms, applications, and design patterns.*
`https://arxiv.org/pdf/1703.06322.pdf`

[8] Mellin, W. *BIZMAC UNIVAC, GARBAGE IN-GARBAGE OUT.* Times Daily, Hammond, Indiana, Nov 10, 1957.

[9] Sztorc, P. *The Case Against Augur.*
`http://bitcoinhivemind.com/blog/case-against-augur/#the-gift-that-keeps-on-giving` Dec 16, 2015.

[10] Krug, J., Peterson, J. *Building a better lie detector.*
`http://augur.strikingly.com/blog/building-a-better-lie-detector` Jun 26, 2015.

[11] Fischer, M., Lynch, N., Paterson, M. *Impossibility of Distributed Consensus with One Faulty Process.*
`https://groups.csail.mit.edu/tds/papers/Lynch/jacm85.pdf` 1985.

[12] Lamport, L. *Paxos Made Simple.*
`http://lamport.azurewebsites.net/pubs/paxos-simple.pdf` Nov 1, 2001.

[13] Ongaro, D., Ousterhout, J. *In Search of an Understandable Consensus Algorithm.*
`https://web.stanford.edu/~ouster/cgi-bin/papers/raft-atc14` 1985.

[14] Peterson, J., Krug, J. *Augur: a Decentralized, Open-Source Platform for Prediction Markets.*
`https://arxiv.org/pdf/1501.01042.pdf`

[15] Sakich, T. *Augur & The Crowdsale Model for Open Source Blockchain/Decentralized Projects.*
`https://medium.com/@TonySwish/augur-the-crowdsale-model-for-open-source-blockchain-decentralized-projects-b2eefc4cb7cc`. Oct 13, 2015.

[16] Harkin, C. *Token Sale Exceeds $1.7M, but Augur's 'Reputation' Isn't All About the Money.*
`https://bitcoinmagazine.com/articles/token-sale-exceeds-1-7m-augurs-reputation-isnt-money-1440027734/`. Bitcoin Magazine, Aug 19, 2015.

[17] Crypto Mining Blog. *Augur's Reputation Crowdsale Finishes With Over 5 Million USD.* http://cryptomining-blog.com/6095-augurs-reputation-crowdsale-finishes-with-over-5-million-usd/

[18] Peterson, J., Krug, J. *Augur Design Diagram.* https://www.websequencediagrams.com/files/render?link=kUm7MBHLoO87M3m2dXzE

[19] Bertani, T. *Use case-driven approach.* https://blog.oraclize.it/use-case-driven-approach-a54b1fcbd2d2 Dec 7, 2015.

[20] TLSNotary. *TLSNotary: A new kind of auditing - cryptographic proof of online accounts.* https://tlsnotary.org/

[21] Oraclize. *Oraclize Documentation.* https://docs.oraclize.it/

[22] Rouviere, S. *Why & How Decentralized Prediction Markets Will Change Just About Everything.* https://medium.com/@ConsenSys/why-how-decentralized-prediction-markets-will-change-just-about-everything-15ff02c98f7c

[23] Schlack, J. *The power of prediction markets.* https://www.quirks.com/articles/the-power-of-prediction-markets May 2013.

[24] Tomaino, N. *Improving the Flow of Information in the World.* https://thecontrol.co/improving-the-flow-of-information-in-the-world-87396ca2d776. Apr 10, 2017.

[25] Wolfers, J., Zitzewitz, E. *Prediction Markets.* http://www.nber.org/papers/w10504.pdf

[26] Sztorc, P. *Market Empiricism.* http://bitcoinhivemind.com/papers/1_Purpose.pdf

[27] Sztorc, P. *Capitalizing on Market Manipulation with "Augmentation".* http://bitcoinhivemind.com/papers/5_PM_Manipulation.pdf

[28] Ozimek, A. *The Regulation and Value of Prediction Markets.* https://www.mercatus.org/system/files/Ozimek_PredictionMarkets_v1.pdf. Mar 2014.

[29] Sztorc, P. *Extra-Predictive Applications of Prediction Markets.* http://bitcoinhivemind.com/papers/3_PM_Applications.pdf

[30] Hanson, R. *Shall We Vote on Values, But Bet on Beliefs?.* http://mason.gmu.edu/~rhanson/futarchy2013.pdf. Mar 2014.

[31] Liston, M. *How Prediction Markets Can Save Democracy From Itself.* https://blog.gnosis.pm/how-prediction-markets-can-save-democracy-from-itself-a813a87cf9bd. Mar 8, 2017.

[32] Bellare, M., Neven, G. *Identity-Based Multi-signatures from RSA.* https://doi.org/10.1007/11967668_10. 2006.

[33] Khovratovich, D., Vladimirov, M. *Secure Token Development and Deployment.* https://edcon.io/ppt/two/Dmitry%20Khovratovich_Secure%20Token%20Development%20and%20Deployment_EDCON.pdf

[34] Green, G. *The MiniMe Token: Open Sourced by Giveth.* https://medium.com/giveth/the-minime-token-open-sourced-by-giveth-2710c0210787. Nov 16, 2016.

[35] Dexaran. *ERC223 token standard.*
https://github.com/Dexaran/ERC223-token-standard

[36] Williams, E. *Ten rules for web startups.*
http://kevin.lexblog.com/2005/11/29/ten-rules-for-web-startups-evan-williams/. November 29, 2005.

[37] Law, E., Roto, V., Hassenzahl, M., Vermeeren, A., Kort, J. *Understanding, Scoping and Defining User Experience: A Survey Approach.*
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.150.180&rep=rep1&type=pdf. Apr 7, 2009.

[38] LiftIgniter Staff. *The Business Value Of Good UX: Why User Experience Is Everything.*
https://www.liftigniter.com/the-business-value-of-good-ux-why-user-experience-is-everything-2/. Jan 12, 2016.

[39] Gube, J.. *What Is User Experience Design? Overview, Tools And Resources.*
https://www.smashingmagazine.com/2010/10/what-is-user-experience-design-overview-tools-and-resources/. Oct 5th, 2010.

[40] Desmet, P., Overbeeke, K., Tax, S. *Designing products with added emotional value: development and application of an approach for research through design.*
http://www.tandfonline.com/doi/abs/10.2752/146069201789378496. The Design Journal, 4(1), 32–47. 2001.

[41] Hayek, F. *The Use of Knowledge in Society.*
http://www.econlib.org/library/Essays/hykKnw1.html American Economic Review. XXXV, No. 4. pp. 519-30, Sep 1945.