Boson Protocol v2 Whitepaper

Web3’s Commerce Layer

Whitepaper Version 2.6: 30 September 2022

Overview

The purpose of smart contracts is to automatically execute agreements, ensuring compliance with contractual terms, whilst minimizing exceptions and displacing intermediaries[1]. Despite this promise, smart contracts are inherently disconnected from the real world. Smart contracts are unable to view the real world on their own, because they cannot trust external data sources. Instead they require data oracles in order to trust off-chain data. Similarly, smart contracts cannot reliably affect the real world, because they cannot trust that a given action will be executed off-chain. We refer to this as the physical asset oracle problem.

We present Boson Protocol v2 as a *decentralized actuator oracle*. That is, a decentralized protocol which enables the trust-minimized, automated execution of off-chain actions; which in turn can be seen as a general purpose solution to the physical asset oracle problem.

The design of Boson’s core smart contracts is unopinionated, and therefore supports the execution of any type of off-chain action. Boson is therefore extensible to a wide range of domains and can support myriad use cases. However, the initial focus for Boson Protocol v2 is decentralized commerce (dCommerce). Boson v2 represents a foundational primitive connecting Web3 with real world commerce. That is, Boson Protocol enables smart contracts to execute off-chain commercial transactions with strong and verifiable guarantees about payment for and receipt of assets.

Boson’s vision is to enable a single *digital market for physical things* (think ‘Uniswap for e-commerce’); where all the world’s products and services are listed and searchable, with commerce automated via code (think TCP/IP for commerce). Boson is built on decentralized infrastructure as a minimally extractive coordinator[2], with value mediated by the $BOSON token, ensuring that participants can share in the value they create.

In order to achieve the vision of a *single digital market for physical things*, Boson addresses three primary problems. Firstly, the *physical asset oracle problem*: if Alice tokenizes her car, and Bob buys the token; how can Bob be sure he will receive the car? Secondly, the *fair-exchange problem:* if Alice wants to remotely buy an item from Bob, how can they ensure that the exchange happens atomically (i.e either both parties receive what they are due or neither do)? Thirdly, *how to digitally represent physical items*.

[1] "Smart Contracts." https://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart.contracts.html. Accessed 16 Jul. 2022.

[2] "Protocols as Minimally Extractive Coordinators - Placeholder.vc." 6 Oct. 2019, https://www.placeholder.vc/blog/2019/10/6/protocols-as-minimally-extractive-coordinators. Accessed 16 Jul. 2022.

To address the *physical asset oracle problem,* instead of attempting to tokenize physical assets directly, Boson tokenizes commitments to trade. The protocol locks up parties' commitments to execute a commercial exchange as a type of *forward contract*, encoded within smart contracts and tokenized as a redeemable non-fungible token (rNFT).

Boson represents a novel solution to the *fair-exchange problem* and is described formally as a *decentralized optimistic fair-exchange protocol.* The protocol is *optimistic*, because it assumes that the exchange has successfully executed unless a dispute is raised within a specified period. This and other optimizations lead to *efficient* execution on decentralized infrastructure. The main payload of dispute resolution is handled by an automated Mutual Resolution process which leverages algorithmic game theory to affect local remedy between parties. Exceptionally, unresolved disputes can be escalated to an independent (and potentially fully decentralized) Dispute Resolver, who will assess the agreement, evidence and claims before deciding on the division of locked-up funds. Dispute Mutualizers assess dispute risk and offer Sellers the option to spread the cost of disputes across multiple transactions.

Redeemable NFTs address the problem of *how to digitally represent a physical asset*, by providing assurances to the bearer of an rNFT that either they will receive the item or their money back. rNFTs represent the right to redeem physical assets on-chain, thus enabling the vision of:

"a single, digital market for physical things, built on Web3 infrastructure"

Web3 technologies will revolutionize the world of commerce

By 2025, Web3 technologies will have revolutionized the world of commerce, in much the same way that Web2 transformed access to information. Physical and digital (phygital?) 'things' will be listed and traded on an open, liquid, digital market. In the early days of the internet, information was mostly siloed within proprietary online networks. However, the zero marginal cost of distribution, combined with consumer demand, led to the single, searchable, open internet of information we enjoy today. Understandably, commerce has taken longer to make the leap. With the exchange of physical assets, the need to manage counterparty risk, mediate disputes and ensure settlement requires trust. This trust is vested in either trusted intermediaries or trusted sellers. Consequently, e-commerce transactions are mostly siloed within one of many, closed, proprietary systems. The advent of Web3 technology enables the automation of settlement by smart contracts and the tokenization of physical asset commerce transactions into a universal standard such as NFTs. Just as decentralized finance's 'money Lego' applications have begun to unbundle traditional finance, an ecosystem of decentralized 'commerce Lego' protocols and applications will evolve to create an open marketplace for things, where everyone can share in the value they create.[3]

Justin Banon, Co-founder
Boson Protocol - WEF Technology Pioneer 2022
From '17 ways technology could change the world by 2027', World Economic Forum Report 2022.

[3] "17 ways technology could change the world by 2027." 10 May. 2022, https://www.weforum.org/agenda/2022/05/17-ways-technology-could-change-the-world-by-2027/. Accessed 16 Jul. 2022.

Boson Protocol v2 Overview

Boson Protocol v2 is a *decentralized optimistic fair exchange protocol,* which enables the trust-minimized, automated exchange of off-chain assets, whilst tokenizing commitments to trade as redeemable NFTs. The protocol enables the creation of a single digital market for physical assets, built on decentralized infrastructure and without the need for centralized intermediaries to enable fair exchange.

Introduction to the design

Commerce is an exchange of valuables between entities, where participants have their subjective valuation functions that need to be matched. Usually that means that a Seller sells a thing to a Buyer for remuneration. By tokenizing the promise to deliver a thing, Boson Protocol can facilitate exchanges of any item for tokens (usually fungible, a.k.a. cryptocurrencies). Supporting all types of assets in a unified way is powerful, because beyond compatibility with existing commercial concepts, it also unlocks the potential to list and trade all items in a single, digital market for physical things. As a native Web3 protocol, Boson Protocol brings autonomy to participants, nonrepudiation and superior efficiency compared to legacy because it is disintermediated at the base layer.

The use of standardized tokens when and *only* when needed, enables ecosystem interoperability. One example of such interoperability is easy access to financing. For example, while the exchange is in progress, Buyers can pick up their locked value in the form of a redeemable NFT and put that token to work in any number of DeFi protocols, secondary markets etc.

Boson Protocol is open infrastructure. While it is permissionless to use, the commitment to progressive decentralization means that the design and build of the protocol will ultimately be open too. This will lead to commerce becoming transparent, programmable, and composable with the wider Web3. As such, Boson is the foundational primitive within a nascent, but rapidly evolving dCommerce ecosystem.

Unlike Web2, the above is guaranteed by design. The commerce rails set forth by Boson Protocol's design and implemented as smart contracts are enforced and immutably the same for all.

Theoretical underpinnings

Promise Theory

To understand the power of Boson Protocol, it is worthwhile understanding the theory supporting it. Promise Theory starts with intentions: the Seller has *an intent* to sell something and the Buyer has *an intent* to purchase it. They are the elementary actors and it doesn't matter here if they are humans or machines.

In order for the two complementary intentions to match, they must be publicly announced. A stated intention is *a promise*. It is interesting to note that an actor can only make a promise about itself and only to another actor, because here we treat them as autonomous agents (e.g. "I promise to give this light saber to whomever gives me one MANA.").[4]

The value of the promise is the actor's internal function. The Seller evaluates a promise by the price they set (money) and the Buyer evaluates a promise by evaluating the worth of the promised outcome (artifact or service performed). For Boson Protocol, it only matters that the two are matched, while the potential negotiation can be done in arbitrary ways and it is out of scope of the protocol (but can be facilitated at higher layers in the Boson Stack, see for example Core SDK Components).

Similar to the boundaries of observability in blockchain, *the assessment* of whether a promise was kept is subjective, but can be done by anyone with some insight. Boson Protocol provides a contractual template between a Seller and a Buyer that specifies the acceptable assessment representations (e.g. a photo of delivery, a collection code etc.) and, as a last resort, provides a means to resolve conflicts via the dispute resolution mechanism.

Thus, in Boson Protocol the Seller and Buyer cooperate by entering into *a mutual promise*. The Seller's promise is instantiated as an *Offer*, stating the delivery of a promised item against some economic benefit, under specified conditions. When the Buyer accepts an Offer an *Exchange* is instantiated (mutual promise) and the Buyer receives a tokenized promise that can be redeemed. The token behaves like a voucher that is consumed (i.e. burned) once redeemed.

In order to maximize their cooperation, especially when they don't trust each other, these actors put their skin in the game by *committing* some funds as a strong signal that they won't break their promise. The Seller does that explicitly via a security deposit, while the Buyer agrees to be charged a penalty if not redeeming. In other words, making promises by untrusted actors works better if they make some commitment to it, else they could be empty words with no rational reason to keep them. The trustworthiness of an Offer is a function of the Seller's reputation combined with the size of the Seller deposit in relation to the price of an Offer. With this mechanism, risk is reduced for both buyer and seller irrespective of the reputation of either; a natural outcome of this across exchanges is the increase of mutually beneficial transactions.

[4] By specifying the desired end state, instead of prescribing algorithmic steps to get there in detail, the promise theory [Bergstra & Burgess, Promise Theory, Principles and Applications, 2019] argues that this is beneficial as it reduces uncertainty - the system converges towards the end state, rather than diverges in endless possibilities in making each phase transition.

Keeping a promise in Boson Protocol simply means that the Seller provided the asset offered via the exchange to the Buyer. This is called *fulfillment* as both actors have, in their opinion, kept their side of the promise. Unless a dispute is raised within the allotted time the payment and other funds are then released by the protocol.

The Buyer is afforded some time post-redemption to raise a dispute. This could be for non-delivery of the item, or if the item is not of the expected quality. Buyers and Sellers can resolve disputes mutually or ultimately escalate to an external actor called a Dispute Resolver. *Mutual resolution* aims to be maximally automated and can be modeled as a 2-player game with an equilibrium in which the players coordinate. In the event where the Buyer and Seller can not come to a mutual agreement Escalated Dispute Resolution is there as a last resort. It bears corresponding costs, is external to the protocol and can be facilitated by a centralized or decentralized actor. The cost for dispute resolution can be mutualized (across Sellers and their Offers), where any such configuration must be done when an Offer is created.

Ultimately every exchange will end up in one of these *finalized* states. The specific state into which an exchange finalizes will determine the payoffs that each actor will receive.

Phases

The above concepts can be visually represented in phases, as shown on the image below. The phases from Commit to Fulfillment/Dispute Resolution constitute what we call an Exchange. More details about the phases are presented in the General Description section below.

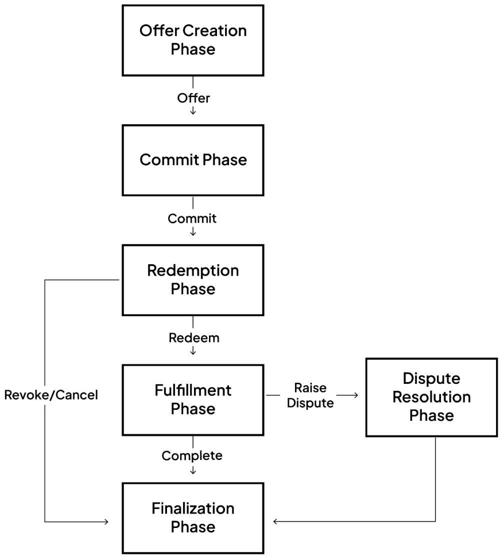

Fig. 1 - Phase diagram

Problem and Solution overview

The fair exchange problem

In online commerce, transactions are by definition ones where the buyers and sellers are not physically present, and where the resulting transactions are asynchronous. This represents a counterparty risk to both sides, referred to in the literature as the *fair exchange problem*[5], that is:

- If the Buyer pays first, they may not get the item.
- If the Seller sends the item first, they may not get paid.

Solutions to the fair exchange problem seek to achieve *atomicity*, so that: *either both parties receive what they are due or neither do*. However, real-world commerce exchanges are not so binary. Disputes may arise regarding the quality of the goods or the execution of the agreement which require more nuanced resolution.

Trust

The fair exchange problem is a problem of trust, where increased trust reduces the transaction costs of commercial exchange[6]. Traditional solutions to increase trust include: transacting with a reputable seller, transacting with a seller who is locatable within a jurisdiction with legal recourse, or transacting via a trusted intermediary. However, within the Web3 paradigm each of these solutions fails. Parties are typically pseudonymous, with limited reputation and no ability to be geographically located. Centralized intermediaries remain an option, but their use as trusted parties defeats the purpose of trustless, decentralized systems.

Centralized intermediaries

Centralized intermediaries often fulfill valuable roles such as creating markets, connecting parties and curating quality. However, market failures occur when intermediaries become monopolies and use outsized market power to enable excessive extraction and create siloed, non-interoperable markets.

Boson represents public commerce infrastructure which is designed to power applications and marketplaces which are either centralized or decentralized. This is closely aligned to the notion of building public roads to service private stores.

[5] "Fair Exchange. - ResearchGate." https://www.researchgate.net/publication/220459639_Fair_Exchange. Accessed 17 Jul. 2022.

[6] "Transaction Cost Economics | SpringerLink." https://link.springer.com/10.1007%2F978-3-642-28036-8_221. Accessed 16 Jul. 2022.

The physical asset oracle problem

A closely related problem to *fair exchange is* the problem of tokenizing physical assets on a decentralized ledger. There is a challenge in creating a digital version of a physical asset such that possession of the digital asset guarantees possession of the physical asset. Jimmy Song highlights the problem as follows: *"Whenever the digital version of the house changes ownership the physical version has to also change ownership. There's a need for the digital world to 'know about the physical world. This is known as the 'oracle problem."*[7]

Of course, there is the option to rely on trusted parties to issue tokens in lieu of physical assets, but what is the point of issuing NFTs whose ownership is immutable if the underlying physical asset relies on a trusted party to honor the claim? This just replicates the trust problem inherent in Web2 and so one may as well issue a digital voucher on a centralized system. Despite this contradiction there exist a number of physical asset tokenization systems, which are built on NFT infrastructure, yet require trust in a centralized entity for redemption.

Trust-minimized, fair-exchange assurance

The critical differentiator between Boson and trusted asset tokenization systems, is that Boson provides Buyers with strong and credible assurances that they will either receive the physical item or they receive their money back - without needing to trust a centralized entity. Instead, Boson uses game theory to incentivize good behavior and the mutual resolution of disputes. As an exception, disputes can be escalated to independent and (optionally) decentralized resolvers.

Physical assets tokenized as redeemable NFTs

Boson's solution to the physical asset oracle problem involves Buyers and Sellers making incentivized commitments to trade, which are tokenized as redeemable NFTs. Rather than tokenized physical assets per se, redeemable NFTs (rNFTs) can be thought of as tokenized forward contracts for the exchange of payment for physical things.

Trust-minimized, automated exchange of off-chain assets

Formally, Boson v2 is an *optimistic fair exchange protocol* which enables the *trust-minimized and automated* exchange of off-chain assets. A fair exchange protocol (FEP) is defined as:

> *"An electronic commerce protocol that ensures that no player gains an unfair advantage over the other player by misbehaving, misrepresenting or prematurely aborting the protocol"*[8]

[7] "The Truth about Smart Contracts - Jimmy Song - Medium." https://jimmysong.medium.com/the-truth-about-smart-contracts-ae825271811f. Accessed 16 Jul. 2022.

[8] "Fair exchange in E-commerce | Ray, Indrajit; Ray, Indrakshi ...." https://ur.booksc.eu/book/44898882/81edbc. Accessed 16 Jul. 2022.

Redeemable NFTs

Boson Protocol's NFTs (rNFTs) are redeemable for off-chain items, and represent a bridge between the isolated blockchain and physical worlds. Whoever is the owner of an rNFT can do all standard ERC-721 operations on the blockchain, but can also get the underlying off-chain item, without relying on a trusted entity.

A single, digital market for physical things

Boson tokenizes commitments to exchange real-world assets as redeemable NFTs, which can be held, transferred or traded like any other NFT. This enables commerce to shift from fractured, monopolized markets to an open and competitive: single, digital market for physical things, built on decentralized infrastructure.

Token-gated commerce

Obtaining an rNFT for certain items can be restricted to special conditions imposed on Buyers, forming so-called token-gated commerce. The conditions for a potential Buyer can be programmed to check the ownership of a specific ERC-721 token or to check the threshold balance of an ERC-721/1155/20 token at a particular contract address. Token gating enables Sellers to target customers based on the assets they hold, enabling a new form of verifiable targeting.

Digital twins and The DEX for anything

The hardest problem which Boson solves is automating the fair exchange of physical assets. Offers can be created that bundle a real-world item and its digital counterpart. The rNFT received when committing to such an Offer could be used to redeem both the physical item and its on-chain counterpart "digital twin" (i.e. an NFT). This allows sellers to provide offers that include a physical item along with an NFT that the Seller owns. Thus, the item of the exchange can be any configuration of physical, digital and experiential bundles, in what are called "phygitals". In this way Boson can be viewed as a decentralized exchange for anything.

Seller and marketplace economics

Boson Protocol rNFTs provide a verified trail of ownership and enable revenues from primary and secondary sales. Royalties are specified by a percentage of the secondary price that the Seller and/or the Agent receives. Placing royalties in-protocol is obviously beneficial to Sellers and Agents as they are thus guaranteed to collections, whilst Buyers are recommended to perform any secondary sale in-protocol for the transaction assurances that Boson Protocol provides.

Capture resistant, decentralized public infrastructure for commerce

Boson Protocol is decentralized public infrastructure for commerce which is progressively decentralizing towards community governance via the Boson DAO. As a result, Boson is aligned with the incentives of participants; this reduces platform risk for builders and is resistant to capture by minority interests. ***Boson is commerce infrastructure which anyone can use, and everyone can trust.***

Minimally extractive fees

Boson Protocol is designed so that all participants share in the value they create, with fees that are sufficient to sustain and grow the ecosystem, and with community governance to ensure that Boson remains minimally extractive, forever.

$BOSON: the native utility token of the protocol

The $BOSON token serves three primary purposes:

- **Governance -** $BOSON tokens are used by participants to govern Boson Protocol, ensuring consensus around critical decisions and the issuance of funds from the dCommerce DAO.
- **Fees** - Boson implements a minimally extractive fee that can be activated or amended by the Boson Protocol DAO through the Protocol fee switch. Fees accrue to the DAO treasury.
- **Incentivization** - $BOSON tokens are used to incentivize actions across the system. Including incentivizing supply and demand acquisition.

dCommerce Ecosystem

Although Boson Protocol stands at the core of Web3 Commerce, more components, services and applications are needed to create an end-to-end online commerce solution. The dCommerce Ecosystem serves this purpose.

dCommerce is open, transparent, decentralized, minimally extractive and resistant to capture public infrastructure for commerce. This infrastructure is a result of unbundling the services provided by eCommerce into individually atomic functional Web3 artifacts that form a composable decentralized ecosystem for commerce.

One of the benefits of Web3-native solutions is that, by virtue of composability, they behave as Legos, i.e. they can seamlessly connect with each other. Unlike Web2 platforms, where a stack is developed in a closed ecosystem, the openness of Web3 always allows for better, more effective, usable, and secure solutions.

Protocol general properties

Certain general properties of the protocol stem from the requirements which carefully follow Boson Protocol's vision:

- ***Open and fair*** - the protocol is being designed and built in a progressively decentralized fashion, which dictates that even early design decisions take decentralization into account.
- ***Permissionless*** - the protocol is public infrastructure for commerce which anyone can use.
- ***Simple*** - the protocol is simple enough for end users and understandable enough for developers.
- ***Reliable and available*** - in order to be used as ubiquitous commerce infrastructure, the protocol is built natively in the Web3 ecosystem.
- ***Efficient*** - the protocol minimizes the cost and number of on-chain transactions.
- ***Optimistic*** - the protocol makes optimistic assumptions that the happy path will be followed.
- ***Automated*** - the main dispute resolution payload is handled by the *Mutual Resolution* game theoretic mechanism, with escalation to a semi-trusted third party (sTTP) *Dispute Resolver* by exception only.

- ***Trust-minimized*** - involves, for escalated disputes only, an *Escalated Dispute Resolver (EDR) who* is a commoditized service provider with minimal market power.
- ***Cost-mutualized*** - the cost of the escalated disputes is mutualized across multiple Seller transactions to reduce the cost impact, especially on low value transactions.

Comparing Boson Protocol v1 vs v2

There are several differences between versions 1 and 2 of Boson Protocol. The salient ones are described below:

- ***Human intervention*** - v1 coordinates commerce without human intervention, but it introduces a malicious buyer attack vector. v2 seeks to resolve the main burden of disputes using algorithmic game theory, with escalation to a human dispute resolver (DR) as an exception only. Even in the limit where all disputes are mutually resolved, the role of the DR is still important. This is because the mere presence of a 'watcher' changes the game theory and blocks attack vectors which are impossible to counter without the presence of at least a semi-trusted third party[9].

- ***Optimistic vs Pessimistic Design*** - In v1 the Buyer is required to sign the exchange finalization transaction, thus degrading the UX and complicating the protocol. v2 on the other hand assumes optimistic exchange finalization: unless a signal to the contrary is received, the system assumes the happy path.

- ***Simplicity*** - The change in the human intervention approach and the optimistic design significantly simplifies the protocol design, improves efficiency and the UX. The reduced number of states in the protocol makes it more practical and commercially acceptable.

- ***Trust-minimization*** - Both versions displace market and financial intermediaries. But v2 takes the property of trust-minimization one step further and enables detailed agreements of the exchange, so that the counterparty can explicitly review and agree to the terms, and then be confident that the decentralized protocol will enforce them.

- ***Cost-optimization*** - The simplified and optimistic design of the v2 results in fewer transactions, therefore optimizing the cost of the protocol operations.

- ***Improved Deposit Model*** - When participating in an exchange, Buyers and Sellers have skin in the game. In addition to the Buyer's payment amount for the item, v1 asks for dual-sided deposits to be put in the escrow. v2 only needs a Seller's deposit and a single payment amount from the Buyer (which implicitly covers the potential penalty). This model is both cognitively simpler and it requires no extra funds to be locked away.

- ***Bundling / Twinning mechanism*** - v2 provides a novel bundling mechanism that allows selling multiple items in the same Offer, e.g. a physical item and a digital NFT-twin.

[9] "On the Impossibility of Fair Exchange without a Trusted Third Party ...." 18 Mar. 1999, https://www.cs.utexas.edu/~shmat/courses/cs395t_fall04/pagnia.pdf. Accessed 17 Jul. 2022.

General Description of Boson Protocol v2

The protocol comprises two main parts: the exchange mechanism and the dispute mechanism. The exchange mechanism governs the state transactions through the protocol where Buyer and Seller exchange the payment for the off-chain asset. If the Buyer wishes to dispute that the Seller has performed their part of the agreement, the dispute resolution mechanism is called from the exchange mechanism to let them reach a mutual agreement or escalate the dispute.

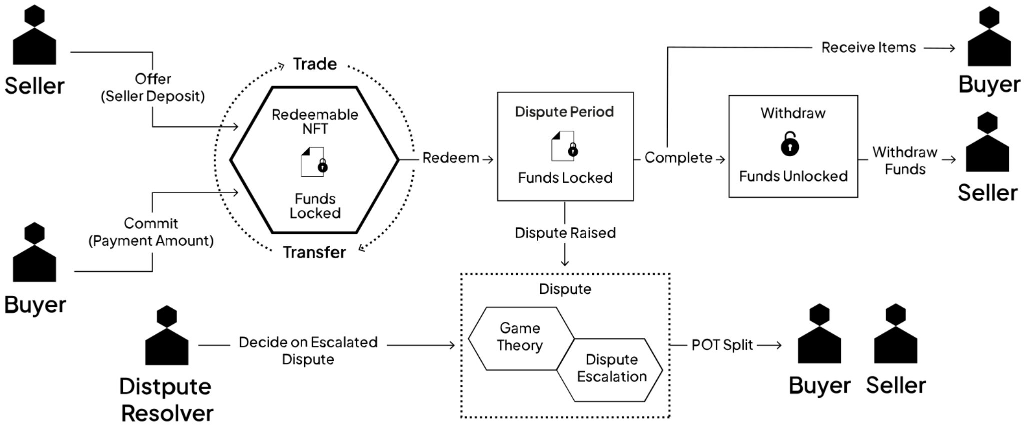

Fig. 2 - How Boson Protocol works

Phases

The exchange mechanism comprises three phases: the Offer Creation phase, the Commit phase and the Redemption phase, in addition to an exceptional Dispute Resolution phase.

Offer Creation Phase

At the Offer Creation phase, the Seller inputs details of an item which they intend to sell, by creating an Offer. The Offer is created in an immutable manner within the protocol and is set up so that the Seller can interact with their newly created Offer. The Offer sets the terms of the offer for sale made by the Seller, including including a link to some metadata containing the contractual agreement between the Buyer and Seller, and a description of the terms of the Offer including: the Buyer Payment Amount which also includes a Buyer Deposit element, the Seller Deposit, and the details of the Dispute resolver.

Commit Phase

If a Buyer agrees to the terms of the Offer, they may proceed via the Commit function which locks up the Buyer’s Payment amount into the protocol, together with the Seller Deposit. At this stage the Buyer receives an NFT which is redeemable for the off-chain asset, from here on referred to as a Redeemable NFT (rNFT) . The Buyer may then choose either to transfer or trade the rNFT, before ultimately moving to the Redemption Phase.

Redemption Phase

The Redemption Phase is the phase in which the holder of the rNFT may choose to redeem the Offer as defined in Offer Phase. The holder of the rNFT can choose to Redeem by interacting with the Redeem function in the protocol. It should be noted that Sellers will be asked to set a Redemption period when creating an Offer, which will determine the period in which Redemption can happen.

If a Buyer Cancels their commitment before Redeem, they will incur a cancellation penalty equal to the Buyer Deposit (aka Buyer Cancelation Penalty). Similarly, during this period, the Seller may Revoke the Offer, thus forfeiting the Seller deposit. In this way, the Buyer and Seller deposits set the reversibility of the respective parties' commitments to transact. As such, a committed Offer can be viewed as a type of *forward contract*[10]. That is, an agreement for the two parties to execute a commercial exchange at an agreed price at or before a specified date.

Fulfillment Phase

After Redeem is called, the protocol progresses to the Fulfillment Phase. This is the phase where the Seller needs to fulfill the promises made out in the Offer. When the Redemption Period is over, the protocol *optimistically* assumes that the Seller has fulfilled their obligations under the contractual agreement, and enables the Seller to withdraw the Payment and Seller's deposit. Alternatively, in the event that the Buyer calls the Dispute function before the end of the Redemption Period, the exchange moves into the Dispute Resolution Phase.

Dispute Resolution Phase

The Dispute Resolution Phase comprises two sub-phases, the Mutual Resolution phase and the Escalated Dispute resolution phase. If a Buyer raises a dispute, the protocol provides a path for the parties to resolve the conflict via a mutual resolution game, whereby Buyer and Seller negotiate off-chain and ultimately send an on-chain compromise proposal for the division of the escrowed funds.

If mutual resolution succeeds, the protocol automatically divides the escrow as per the mutual agreement. (The mutual resolution game is designed to handle the main payload of disputes in much the same way that chatbots handle first line call center queries).

If mutual resolution fails, the protocol escalates to an external Dispute Resolver (DR). The DR reviews the agreement, claims, evidence requirements and evidence provided. The DR then decides on how to split the funds held in the protocol for the given exchange, based on payout guidelines specified within the Offer contractual agreement.

Finalization Phase

The Finalization Phase encompases the set of states that signal that a given exchange has come to an end. There are a number of different finalization states. Each one of the end states has its own label and comes with its own set of rules governing the payout of the funds committed to the protocol. For a view of the different payoffs based on the end states, please refer to Appendix - Boson Protocol Payoff Table.

[10] "Forward Contract Definition - Investopedia." https://www.investopedia.com/terms/f/forwardcontract.asp. Accessed 29 May. 2022.

Notation

The following table defines the notations used across different phases of the protocol.

There are several actors that the core protocol recognizes, as described in the table below. The core protocol is permissionless, thus any account can take any and all roles, implying any negotiations or communications are out of scope. For example, it is in the Seller's own interest to choose a good Dispute Resolver, otherwise Buyers might detect unfair play from the public events that Boson Protocol emits. However for the core protocol it is not relevant how exactly the Seller chose a particular Dispute Resolver. Similarly, in the potential mutual resolution phase, it is not relevant to the protocol how exactly Seller and Buyer communicate, as long as the protocol observes the outcome of their agreement or disagreement.

Type	Label	Description
Actor	Bu	Buyer. Accepts the Offer and in turn receives a redeemable token which they are committed to redeem. Can raise disputes. It is also possible to commit to an Offer on behalf of someone else, in which case the Offer is accepted (and paid for) by X, but Y is recorded as the Buyer and so Y receives the rNFT and all corresponding authorizations.
	Bu'	Secondary Buyer. The bearer of an rNFT who did not originally Commit to the transaction, but has purchased the rNFT on a secondary market or otherwise obtained custody of said rNFT.
	Se	Seller. Offers things and commits to provide them in good quality. Specifies Fee Mutualizer and Dispute Resolver of an Offer.
	DR	Dispute Resolver. Resolves escalated disputes that Sellers and Buyers could not mutually resolve. Acts as a last resort only and can choose not to participate.
	FM	Mutualizer for Dispute Resolution Fee. An optional actor that mutualizes the costs of potential DR fees.
	$Agent$	An optional third party that takes a fee in successful exchanges (ending in Completed or Retracted states). E.g. a marketplace.
	DAO	Manages protocol's configuration and its treasury.
	Any	Anyone interacting with the system.

Amount	p	Offer Price or Payment Amount. The payment Buyer has to deposit at the Commit time. This includes the Buyer Cancellation Penalty.
	c	Buyer Cancellation Penalty. Paid by Buyer if they Cancel the exchange after Commit. This amount is reserved within the Offer Price.
	d_{Se}	Seller Deposit (Revocation Penalty). Paid by Seller if they Revoke the exchange after Commit.
	f_{DR}	Dispute Resolution Fee. The payment the DR gets for deciding on a Dispute.
	$d_{Escalation}$	Buyer Escalation Deposit. The deposit Buyer pays to escalate a Dispute.
	P_{Bu}	Payoff to Buyer. The funds Buyer receives at the end of Exchange or Dispute.
	P_{Se}	Payoff to Seller. The funds Seller receives at the end of Exchange or Dispute.
	POT	Available escrowed funds to split between Buyer and Seller. $POT = p + d_{Se}$ and in case of escalation: $POT = p + d_{Se} + d_{Escalation}$
	P_{Bu}'	Payoff to Buyer that represents the percentage number [0,100] of the POT that Buyer receives at the end of Dispute.
	f_{Ag}	Optional fee an Agent charges for successful exchanges.
	f_{Proto}	Fee that Boson Protocol charges for successful exchanges.
Exchange Time Periods	T_{Offer}	Offer Validity Period. Period of time when Buyer can Commit to Offer.
	$T_{Inactive}$	(Optional) Inactive Period. Period of time post Buyer Committed where Buyer can't Redeem.
	T_{Redeem}	Redemption Period. Period of time when Buyer can Redeem an rNFT.
	$T_{Dispute}$	Dispute Period. Period of time when Buyer is waiting for the item to be received and during which a Dispute can be raised.
Mutual Resolution Game	G_{MR}	Mutual Resolution Game. A game that Buyers and Sellers play to mutually resolve a Dispute.
	T_{MR}	Mutual Resolution Period. Period of time when Buyer and Seller can mutually agree on the split of the locked funds.
	$T_{Proposal}$	Time period that can be used in G_{MR} to make split proposals.
	$T_{Escalation}$	Escalation Response Period. Period of time Dispute Resolver has to decide on Dispute.
Misc.	n	Number of Redeemable NFTs that can be issued from Offer.

	$E\ (d)$	Predicate function that returns 1 if a Dispute was escalated and returns 0 otherwise.
	MAX_{fee}	Maximum total fee percentage that can be taken from the payment amount (f_{Ag}, f_{Proto} etc.). 100% by default.

Table 1 - Notations

The Exchange Mechanism

In the following sections, we provide a general description of Boson Protocol v2. We detail the protocol as a state machine, with the relevant actors making actions that enable the state transitions. Given a smart contract implementation of the protocol, the aforementioned actions are transactions that perform state transitions on the state machine which BPv2 implements.

The following diagram represents the state machine of the protocol with the transitions between the states of different phases.

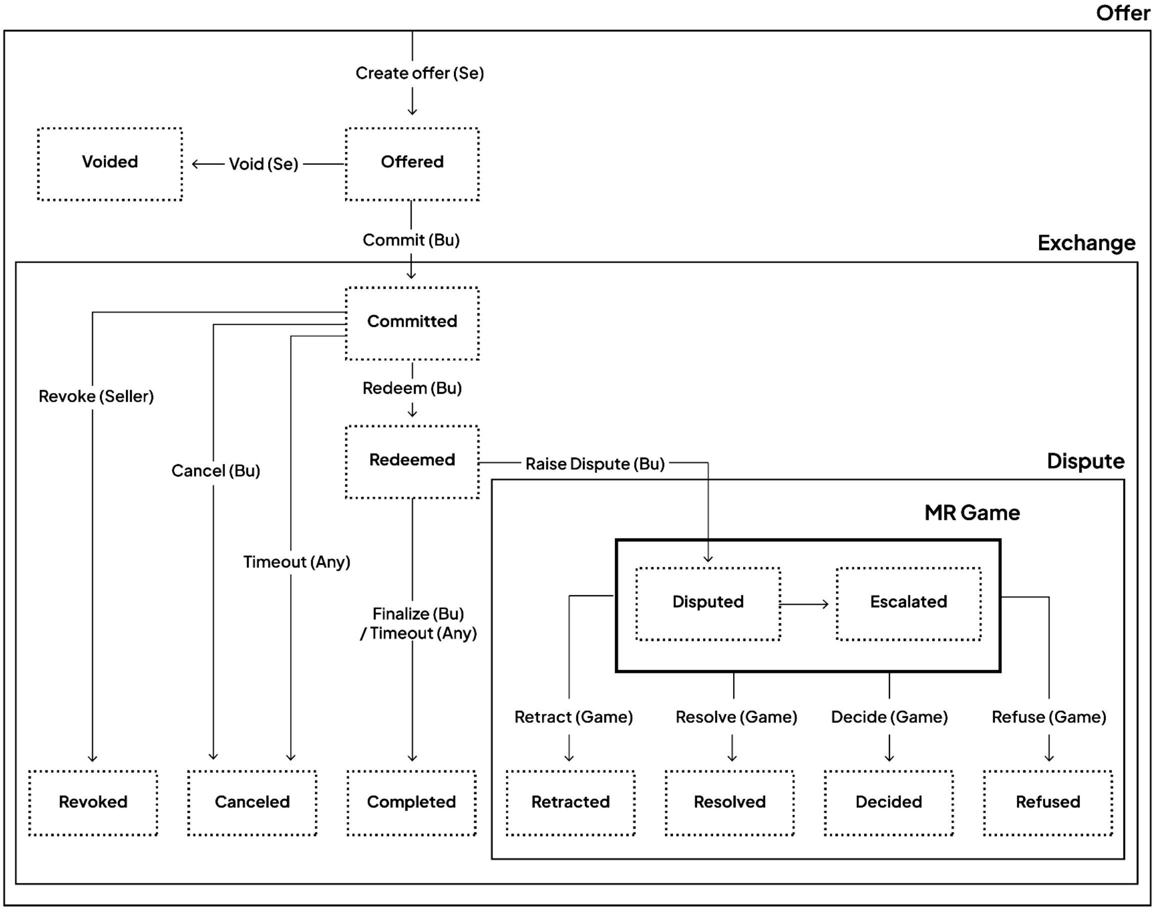

Fig. 3 - The Boson state machine

Offer Creation

The exchange mechanism starts with the Offer Creation phase, when Seller (Se) creates an Offer for an item to be exchanged. The Offer is considered to be an agreement between Buyer (Bu) and

Seller (Se). The protocol aims to ensure that an exchange executes in accordance with the intent of the agreement between parties.

- Se identifies a Dispute Resolver (DR) they want to use for the exchange.
- Se creates an Offer for an item to be sold, specifying the parameters of the exchange and some metadata detailing the nature of the offer, including the terms of sale as well as the offer description. The parameters are:
  - p: price of the Offer, denominated in the exchange token, where $p \geq 0$
  - c: Buyer Cancellation Penalty, denominated in the exchange token, where $c \geq 0$, $c \leq p$
  - d_{Se}: Seller Deposit, denominated in the exchange token, where $d_{Se} \geq 0$
  - n : number of rNFTs that can be minted from Offer, where $n > 0$ practically unlimited
  - time periods: T_{Offer}, $T_{Inactive}$, T_{Redeem} , $T_{Dispute}$ that specify temporal bounds
  - G_{MR}: details of the Mutual Resolution Game with time periods T_{MR}, $T_{Proposal}$ as well as other game parameters
  - assigning the DR: Dispute Resolver
- Se performs action "Create Offer" on Boson Protocol smart contracts, which starts the T_{Offer} time period.
- Se should deposit m to the Seller Pool, where $m \geq d_{Se}$, denoting the amount of tokens deposited by Se. The Seller Pool is a smart contract that covers Seller Deposits (d_{Se}) and needs to have enough funds available when a Bu commits.

Se performs action "Create Offer", by defining all necessary parameters for the Offer and depositing at least d_{Se} into the protocol[11]*.* *Protocol proceeds to the "Offered" state.*

Se can Void an Offer at any time. The Void Offer action does not incur any penalties for the Se. This action ends Bu's ability to Commit to that Offer. However, this does not affect previously committed or redeemed rNFTs, which continue their lifecycle through the core exchange mechanism.

In protocol state "Offered":
Se performs action " Void". Se can withdraw the funds that they deposited. *Protocol proceeds to the "Voided" state.*

[11] As a practical optimization for capital efficiency, there is no requirement for Sellers to deposit their funds for Offers that are yet to be committed by any Buyer. This is implemented by having a pool of Seller's funds from which the required deposit can be drawn from into the protocol when a Buyer Commits to one of the Seller's Offers.

Commit

Once the Offer has been created, it is available for any suitable Bu to commit to by making the payment amount p. The protocol acts as an escrow that holds these funds in order to provide its practical atomicity guarantees ("the exchange goes through or your money back").

- Bu reviews the Offer parameters and any additional information related to the agreement proposed by the Se. Then Bu performs the "Commit" action by making payment p to the protocol that starts the sequential time periods $T_{Inactive}$ + T_{Redeem}, where $T_{Inactive} \geq 0$, $T_{Redeem} > 0$.
- A new Redeemable NFT is minted and transferred to the $Bu's$ account, the exchange begins.

In protocol state "Offered":
Bu performs action "Commit" by sending funds p into the protocol for the Offer. Bu is given an rNFT, a forward contract that can be used to Redeem a Seller's Offer. *Protocol proceeds to the "Committed" state.*

Redeemable NFT (aka Voucher)

The voucher is a token of commitment of the Buyer to a specific Offer, encapsulated as a Redeemable NFT (rNFT). The holder of the token is entitled to receive the item, or in case of refund, the funds locked in escrow. For all practical purposes in the protocol, the holder of the rNFT is treated as *the* Buyer.

Secondary sale royalties are guaranteed to the original Buyer by means of standard token transfer functionalities.

Redeem

The Redeem action is available to Bu at the Committed state of the protocol. Bu signals their readiness to receive the item by turning in their rNFT. This action burns the rNFT and starts the time period when Se has to fulfill their commitment.

- During the time periods $T_{Inactive}$ + T_{Redeem}, Bu may transfer or trade the rNFT outside of the protocol.
- Bu can Redeem the voucher at the Committed state of the protocol during the time period T_{Redeem} . This ends the voucher's transferability and the protocol records that it is being consumed for redemption. At this point the time $T_{Dispute}$ period begins.

In protocol state "Committed":
Bu performs action "Redeem" by turning in the rNFT to signal intention to redeem. *Protocol proceeds to the "Redeemed" state.*

Seller Revokes

In the Committed state, Se can Revoke the exchange during $T_{Inactive}$ + T_{Redeem} at the cost of their deposit d_{Se} defined in the Offer. It results in the following payoffs: $P_{Bu} = p + d_{Se}$; $P_{Se} = 0$. This action is possible during time period $T_{Inactive}$ + T_{Redeem} .

In protocol state "Committed":
Se performs action " Revoke". Bu will receive the entire escrowed amount: $p + d_{Se}$ and Se will get back nothing. *Protocol proceeds to the "Revoked" state.*

Buyer Cancels

Similarly, Bu can Cancel the exchange during $T_{Inactive}$ + T_{Redeem} at the cost of c defined in the Offer. It results in the following payoffs: $P_{Bu} = p - c$; $P_{Se} = d_{Se} + c$.

If Bu does not redeem the rNFT before T_{Redeem} expires, they are assumed to have cancelled the exchange. It results in the same payoffs: $P_{Bu} = p - c$; $P_{Se} = d_{Se} + c$.

In protocol state "Committed":
Bu performs action " Cancel". Bu will get back: $p - c$, and Se will get back $d_{Se} + c$. *Protocol proceeds to the "Cancelled" state.*

Fulfillment

After the Bu redeems their rNFT, the Se has to fulfill their end of the exchange by delivering the promised item. The Buyer, on the other hand, can raise a dispute if they are not happy with the exchange. The protocol can be used to support various modes for order fulfillment, such as click & collect, and online shipping.

- $T_{Dispute}$ starts after a successful redemption transaction. If no dispute is raised by Bu before the $T_{Dispute}$ is over, the exchange is completed successfully. This optimistic protocol design assumes the Bu is happy unless they raise a dispute for the exchange.
- Alternatively, the Bu can explicitly Finalize the exchange during $T_{Dispute}$ to expedite the transition.

In protocol state "Redeemed":
If Bu performs the action "Finalize" or takes no action (i.e. timing out)[12]. Se will get the entire escrowed amount $d_{Se} + p$. *Protocol can then proceed to the "Completed" state.*

The Dispute Mechanism

After redeeming the rNFT, if the Bu asserts that the Se has failed to meet their obligations within the agreement, then the Bu can use the Dispute action to push the protocol into the Disputed state. In this state, there are three paths forward: Mutual Resolution, Escalated Dispute Resolution, and Retract (i.e., Bu retracts their dispute and finalizes the purchase as per the original offer).

In protocol state "Redeemed":
Bu performs action "Dispute". *Protocol proceeds to the "Disputed" state.*

Mutual Resolution

If the Bu and Se mutually agree to a split of the escrowed funds, then they are able to inform the protocol of their mutual decision to resolve the dispute by applying that split. For this, the protocol requires a message signed by both parties to authenticate the mutual decision.

- Bu raises a Dispute during $T_{Dispute}$ that enters the Mutual Resolution path and starts execution of the game (G_{MR}).
- Bu and Se negotiate, i.e. they play the game (G_{MR}) that has been designed based on Game Theoretic principles), and is defined by the Se in the Offer.
- G_{MR} specifies time periods T_{MR} and $T_{Proposal}$ and other parameters to incentivise Bu and Se to achieve the mutual agreement on how to split the $POT = p + d_{Se}$.
- If Bu and Se reach the mutual agreement, they perform the action Resolve to finalize the Dispute and split the POT, where $P_{Bu} \leq POT$ and $P_{Se} = POT - P_{Bu} = p + d_{Se} - P_{Bu}$
- In the trivial game G_{MR}, Bu and Se use the time period T_{MR} to mutually agree on the split. Se can extend T_{MR}, if they need more time to reach an agreement.
- It should be noted that regardless of the dispute resolution state, Bu and Se can always mutually agree on a split and Resolve the Dispute.

[12] If no action is performed for T_{Redem} time, the protocol times out, defaulting to the "Completed" state.

In protocol state “Disputed”:

Either Bu or Se perform the action “Resolve”, with a message signed by the other party that contains a split of the escrow to be refunded to Bu & Se. The message must provide $P_{Bu} \leq POT$ which is the amount awarded to Bu. Implicitly, Se gets back the amount $P_{Se} = POT - P_{Bu}$

Protocol proceeds to the “Resolved” state.

Escalated Dispute Resolution

The Bu is able to seek escalated dispute resolution from the Dispute Resolver (DR).[13] The DR will analyze the case, offline, and provide a decision for the split of escrowed funds that is to be refunded to each party. The DR checks the contractual agreement along with the evidence from the Bu and Se to determine an appropriate split.

- In the trivial form of the mutual resolution game G_{MR}, Bu can escalate a Dispute to a DR during the time period T_{MR}. It ends the previous time period and starts the time period $T_{Escalation}$, during which the DR has to Decide on the Dispute (provide a split of the POT).

In protocol state “Disputed”:

Bu performs action “Escalate” to signal that DR is required to make a resolution as to the split of the escrowed funds that is to be refunded to each party. Bu is required to pay $d_{Escalation}$ in order to perform this action.

Protocol proceeds to the “Escalated” state.

- DR can use the $T_{Escalation}$ to collect evidence from Bu and Se and provide a decision on the Dispute, where
  - $P_{Bu} \leq POT$
  - $P_{Se} = POT - P_{Bu} = p + d_{Se} + d_{Escalation} - P_{Bu}$
- The split provided by the DR may be used as an input for a game G_{MR} as the state of truth. G_{MR} then can use it to output the final payoffs of the Dispute based on the Game Theory.

[13] However, it is recommended that the parties attempt Mutual Resolution first. In order to prevent frivolous escalations, the Buyer must deposit an additional amount $d_{Escalation}$ into the escrow.

In protocol state "Escalated":
DR performs action " Decide", with a split of the escrowed funds that is to be refunded to each party. DR must provide $P_{Bu} \leq POT$, which is the amount awarded to Bu. Se gets back the amount $P_{Se} = POT - P_{Bu}$. *Protocol proceeds to the "Decided" state.*

- Bu has an option to give up, i.e. Retract, on the Dispute at any time after it was raised, but before it was finalized. In this case Bu agrees to receive no funds back:
  - $P_{Bu} = 0$
  - if Dispute was not escalated: $P_{Se} = POT = p + d_{Se}$
  - If Dispute was escalated: $P_{Se} = POT = p + d_{Se} + d_{Escalation}$

In protocol state "Disputed" or "Escalated":
Bu performs action " Retract" to signal the intention to give up on the Dispute. Bu gets no refund back $P_{Bu} = 0$. Se gets all funds in the escrow $P_{Se} = POT$ *Protocol proceeds to the "Retracted" state.*

- If DR fails to provide the split for the Dispute during $T_{Escalation}$ or explicitly refuses to decide on the Dispute, the protocol ends up in the Refused state, assuming this behavior will affect $DR's$ reputation in the future. The funds committed are reverted to the original parties as per the original Offer:
  - $P_{Se} = d_{Se}$
  - $P_{Bu} = p + d_{Escalation}$

In protocol state "Escalated":
DR performs action " Refuse" or fails to decide on Dispute during $T_{Escalation}$. The funds are reverted to the original parties: $P_{Se} = d_{Se}$ and $P_{Bu} = p + d_{Escalation}$ *Protocol proceeds to the "Refused" state.*

Description of, and rationale for, the mutual resolution mechanism

The protocol's primary objective is to ensure efficient, fair exchange and so aims to minimize the incidence of Escalated Resolution (ER). In general, making sure any disagreement process is handled within the Mutual Resolution Mechanism rather than as part of ER ensures that the protocol executes in an automated and reliable manner. We conjecture that the more mutual resolution occurs, the more trust users will end up having in the claims made by the protocol.

While the repayment of funds committed on the happy path should be $p + d_{Se}$ to seller and 0 to the buyer, in case of dispute, they may be split in different ways, depending on the outcome of the resolution process. The latter holds even when the players reach an agreement and do not call the dispute resolution procedure. The contract should know this split in order to make transfers to players, so the agreement should specify the split.

In case of not reaching agreement in the game G_{MR}, we record at most two additional numbers, one proposal from each player or just one proposal from the seller. Recording these numbers must also be an inexpensive on-chain operation. Both players are punished in proportion to their deviation from the truth as output by the procedure. This incentivizes both players to lower their proposed shares and reach mutual agreement. Such punishment also protects the system from malicious attackers who might call the costly dispute resolution procedure each time just to damage the system.

Appropriate deposits should be made in order to make the threat of dispute resolution procedure credible, even though it will not happen on an equilibrium path for most of the types of players. We may want to make suggestions on deposit levels in the future.

Properties of proposed MR games

1. ***Fairness*** - the mechanism treats both players equally. Therefore, assuming that the DR is fair, then protocol fairness is readily implied.
2. ***Cost and time effectiveness*** - the MR game lowers the cost of financing DR, as it is called less and lowers the time to reach resolution for the same reason.
3. ***UX impact*** - the MR game adds two additional transactions.

Contractual Agreement between Buyers and Sellers

The infrastructure for exchanges of off-chain assets for on-chain tokens is based on Boson Protocol's smart contracts and a human readable buyer-seller contractual agreement. The combination of the on-chain and human readable contract represents an important innovation that aims to revolutionize disputes and legal recourse in commerce.

Boson Protocol is designed to ensure a fair exchange for both parties involved in the exchange, regardless of the jurisdictions in which counterparties are based or whether legal recourse is indeed possible due to pseudonymity of parties. The protocol links a human-readable contractual agreement to the smart contracts, providing a clear description of the transaction process and defining certain rights and obligations for both the seller and the buyer. This allows for a fair and transparent dispute resolution process, with clear documentation and proof of evidence.

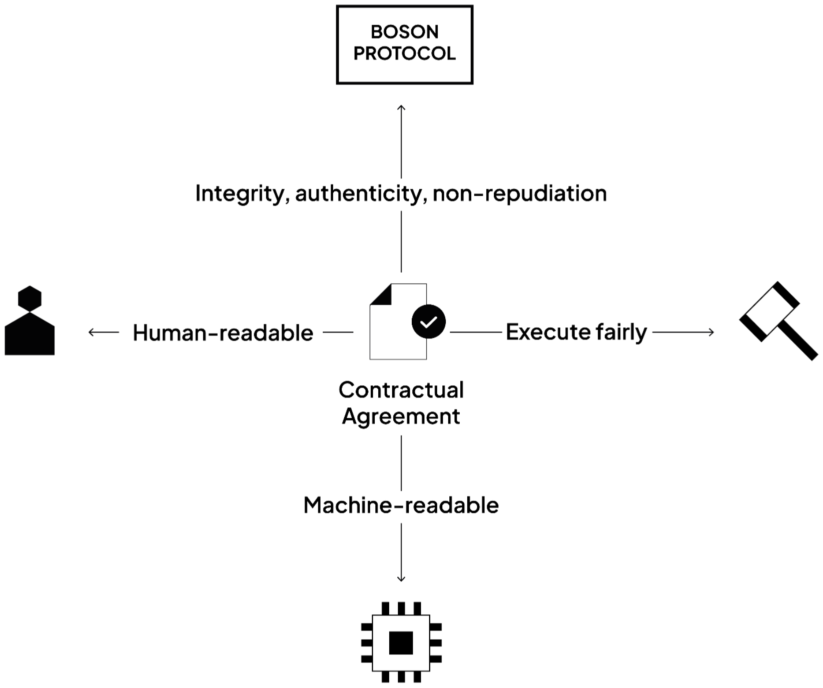

Fig. 4 - Readability and execution of the Contractual Agreement

The challenge is creating an irrevocable connection between a smart contract and a human-readable contract. The solution involves creating a template for the human-readable contract and then populating it with data from the smart contract and from an immutable metadata file linked to from the smart contract (stored e.g. on IPFS). This combination of public and immutable data allows external parties to easily verify the terms of any Offer created in the protocol.

There are three key components, which the contractual agreement is based upon.

A blockchain is used to:

- record the signatures of the Buyer and Seller interacting with the Boson Protocol smart contracts;
- immutably store a unique identifier (hash) of the human-readable contractual agreement;
- keep track of the relevant parameters that are required for the smart contract execution and that will feed into the contractual agreement.

An immutable decentralized off-chain storage is used to:

- keep track of other parameters that are not necessary for the execution of the smart contract but that are required to complete the contractual agreement.

The Boson Protocol dApp is used to:

- provide a user-friendly interface for the parties to interact with the system and finalize their agreement about the transaction.

Technology overview

Architecture overview

Boson Protocol has been designed with composability in mind. Beyond the mandatory on-chain modules that cover the main logic, there are some that are used optionally or can be customized. The multi-layered protocol as a whole acts as a Web3 building block on its own and could well be taken as a black box that facilitates off-chain exchanges, though if one wants, the black box could be inspected in detail as the protocol is entirely open.

The key part of the protocol is the core exchange mechanism subsystem that handles the exchange of the on-chain value for the off-chain value between two parties. The protocol provides other features through a set of functional and optional protocol modules.

The Boson Protocol stack

The below diagram represents the modular and composable nature of Boson Protocol v2's tooling and applications. These components are a set of reusable libraries and visual elements/components that provide developers and integrators with the tools to create high quality and easily maintainable applications with Boson Protocol.

Describing the stack bottom-up, it is worth observing that Boson Protocol aims to be a composable block in the wider Web3 space. It has many parts customizable, but for some essential modules in the Core Protocol Layer which are enforced and configurable only by the governing DAO.

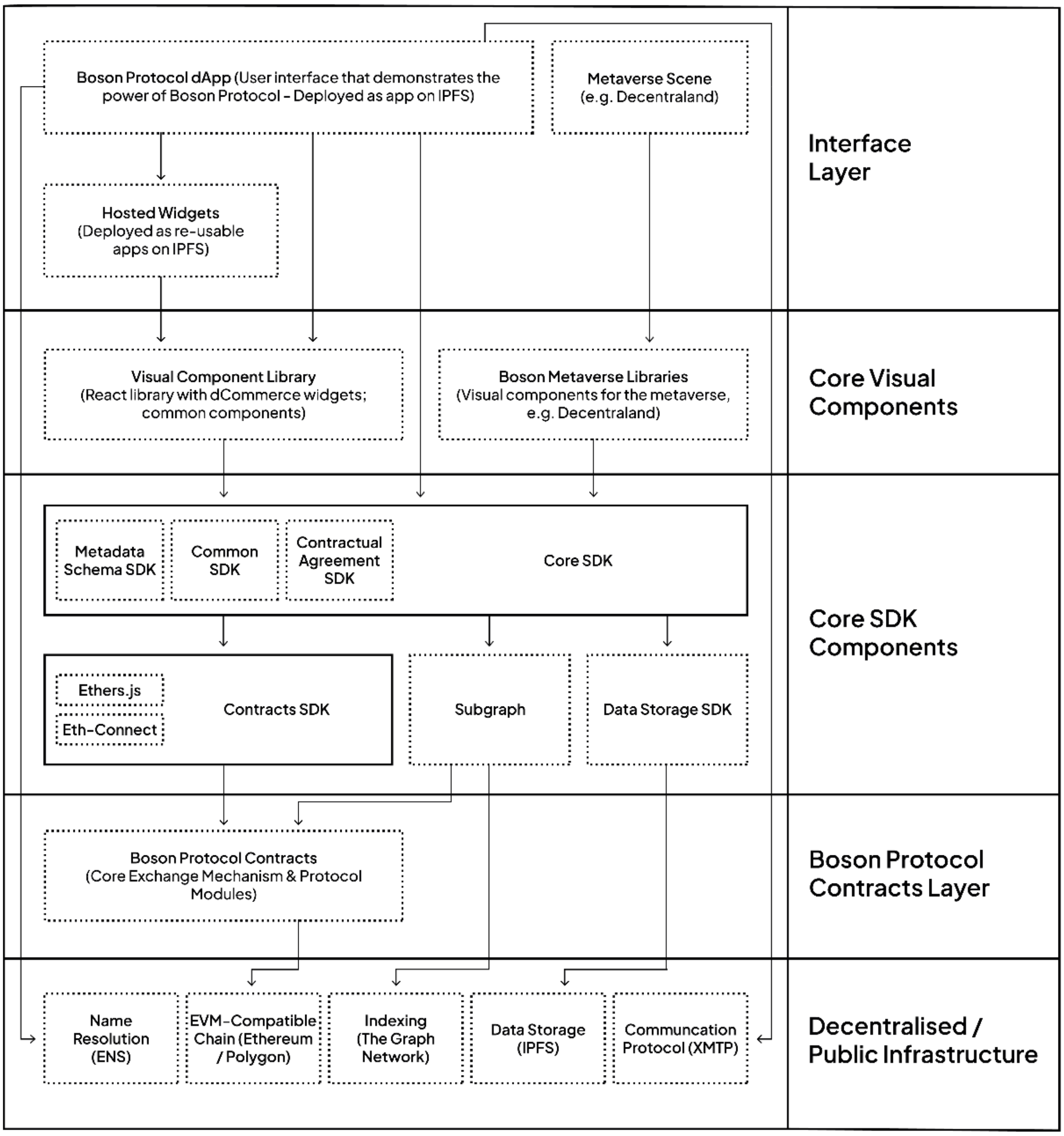
Boson Protocol dApp (User interface that demonstrates the power of Boson Protocol - Deployed as app on IPFS)
Metaverse Scene (e.g. Decentraland)
Hosted Widgets (Deployed as re-usable apps on IPFS)
Interface Layer
Visual Component Library (React library with dCommerce widgets; common components)
Boson Metaverse Libraries (Visual components for the metaverse, e.g. Decentraland)
Core Visual Components
Metadata Schema SDK
Common SDK
Contractual Agreement SDK
Core SDK
Ethers.js
Eth-Connect
Contracts SDK
Subgraph
Data Storage SDK
Core SDK Components
Boson Protocol Contracts (Core Exchange Mechanism & Protocol Modules)
Boson Protocol Contracts Layer
Name Resolution (ENS)
EVM-Compatible Chain (Ethereum / Polygon)
Indexing (The Graph Network)
Data Storage (IPFS)
Communcation Protocol (XMTP)
Decentralised / Public Infrastructure

Fig. 5 - The technology stack

The Decentralized/Public Infrastructure layer

The core protocol can run on any EVM-compatible blockchain e.g. Ethereum, Polygon, Gnosis Chain etc. The protocol is designed so that it can be used with meta-transactions, enabling users to sign transactions in a standardized way, which can then be relayed to the target chain by another party.

For data indexing, The Graph network has been leveraged. This provides a rich interface for accessing the state of the protocol along with the data stored off-chain. The primary motivation for using The Graph is that it provides a means for indexing data stored on-chain alongside data stored in an immutable decentralized file-system, such as IPFS. The Graph's ability to combine data stored on chain with data stored on IPFS is the main reason for using The Graph and IPFS combined.

As a data storage layer the IPFS network can be used for storing images and large bodies of text in a decentralized and immutable manner, requiring only a minimum storage capacity to be kept on-chain.

Another piece of decentralized infrastructure leveraged is XMTP. XMTP protocol enables secure exchange of messages between Ethereum addresses, this is leveraged to enable Buyer and Seller communication as well as to enable dispute resolution in a private and decentralized manner.

Boson Protocol Contracts Layer

The smart contract layer is the substrate of Boson Protocol. It consists of four main modules (offer, exchange, rNFT, dispute resolution) and several additional ones.

As a state machine, the protocol can be represented with the state of its data structures (about the protocol itself and the exchanges recorded within) and the transitions of these structures, initiated by various actors, as shown in the below diagram.

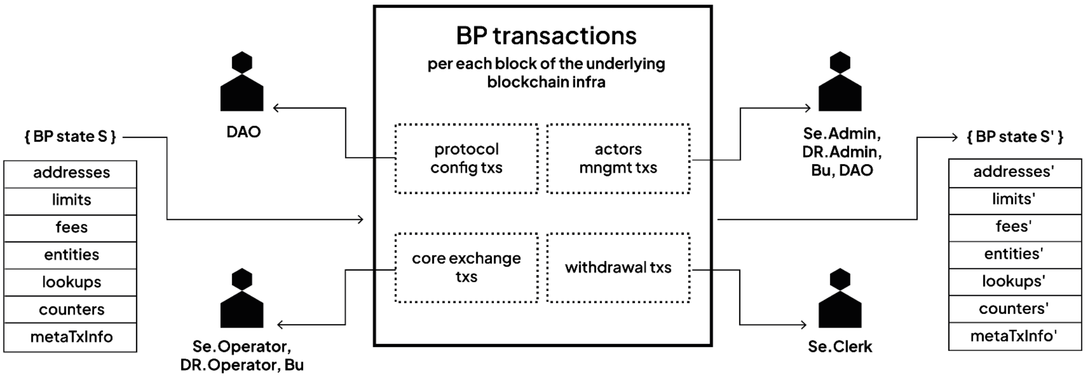

Fig. 6 - *Transitions and states*

It is implemented as a set of smart contracts, written in Solidity and based on the EIP-2535 diamond pattern[14] which enables a single point-of-entry to the protocol's multitude of functionalities, as well as upgradeability and other benefits.

Core exchange modules cover the main process, starting with the Offer module for creating offers that can be simple, such as one-item offers, or more complex, involving groups of things, physical-digital twinning, conditional offers etc. The exchange module handles the interactions of Sellers and Buyers related to their commitments and fulfillment of promises. The NFT module manages the vouchers as redeemable NFTs that exist intermittently to provide Buyers with features to transfer and trade the right to claim the offered thing. The dispute resolution module provides a path for Buyers and Sellers to either match their proposals or escalate it to the ultimate resolver.

Non-core exchange modules provide supporting functionalities to the protocol or provide additional features, of which some are optional and offer more complex use-cases.

[14] https://eips.ethereum.org/EIPS/eip-2535

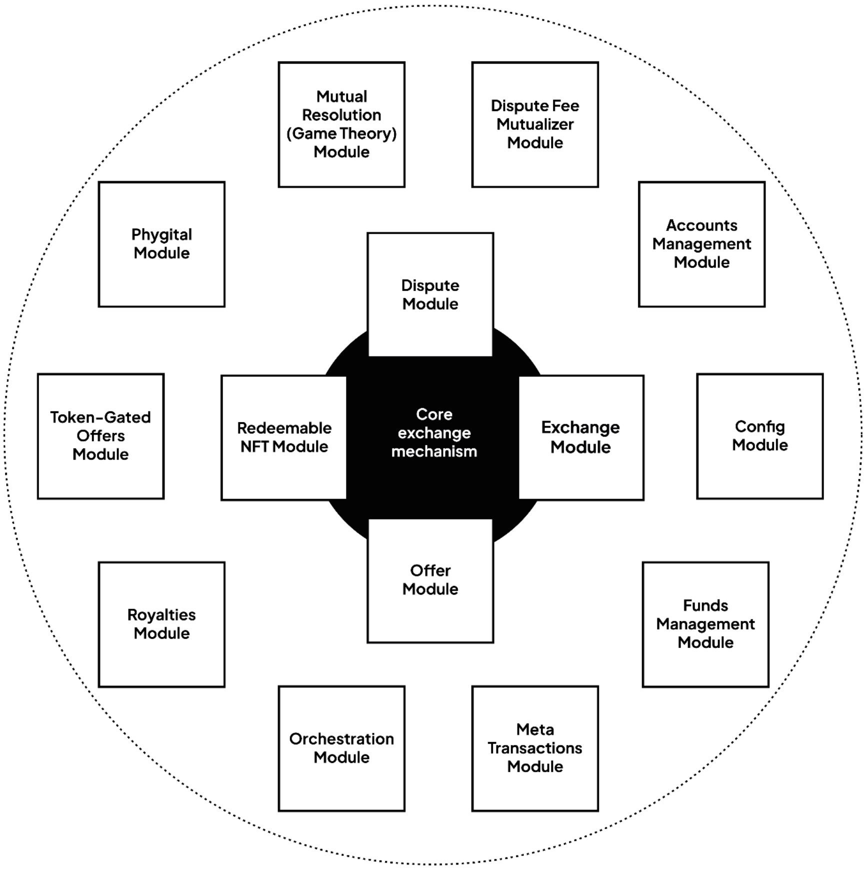

Fig. 7 - *Layer schematic of Boson Protocol's modules*

Phygital module makes it possible to link a physical thing to its digital counterpart, thus making a "phygital" twin, and furthermore to bundle multiple items or twins within a single Offer. The phygital module allows the creation of an Offer for an off-chain asset along with a number of ERC-721, ERC-1155 and ERC-20 tokens. The organization of smart contracts makes these complex scenarios still easy to use.

Token-gated Offers module facilitates conditional offers that are available only to holders of specified token(s). This token gating provides restrictions (such as exclusivity) on the Buyers that are allowed to participate in such offers. The conditions can be set to fungible tokens (balance condition) and non-fungible tokens (ownership condition).

Accounts module provides the basic structure and functionality for managing account profiles of actors in Boson Protocol. Its purpose is to separate responsibilities within an organization. In particular, those are the operation, administration, and funds withdrawal activities.

Mutual resolution module covers the on-chain part of the mutual resolution game that Sellers and Buyers are encouraged to partake in for negotiating disputes on their own. As such, it plays an important role in Boson Protocol. Its implementations rely heavily on game theory and it is the Seller that chooses the best suited template for a particular use case.

Dispute resolution mutualization module mainly provides support for handling the mutualizing fees for dispute resolution.

Funds management module manages the custody of funds and withdrawals of available balances. There are several accounts/buckets involved, though not all reside in-protocol, as shown in the figure below. Per each exchange, the protocol encumbers all corresponding funds into the escrow when the Buyer commits to an offer. In case all the required funds can't be pulled in, the transaction reverts. After the exchange process completes (i.e. reaches its final state), the funds are split according to the payoff algorithm and made available for parties to withdraw their balance.

There are several types of funds involved, concerning different participants/accounts:

- Seller deposit,
- Seller secondary sale royalties,
- Buyer payment,
- Buyer cancellation penalty,
- Buyer escalation deposit,
- Dispute Resolver fee,
- Dispute Resolution Mutualizer fee,
- Boson Protocol fee,
- Agent commission,
- Agent secondary sale royalties.

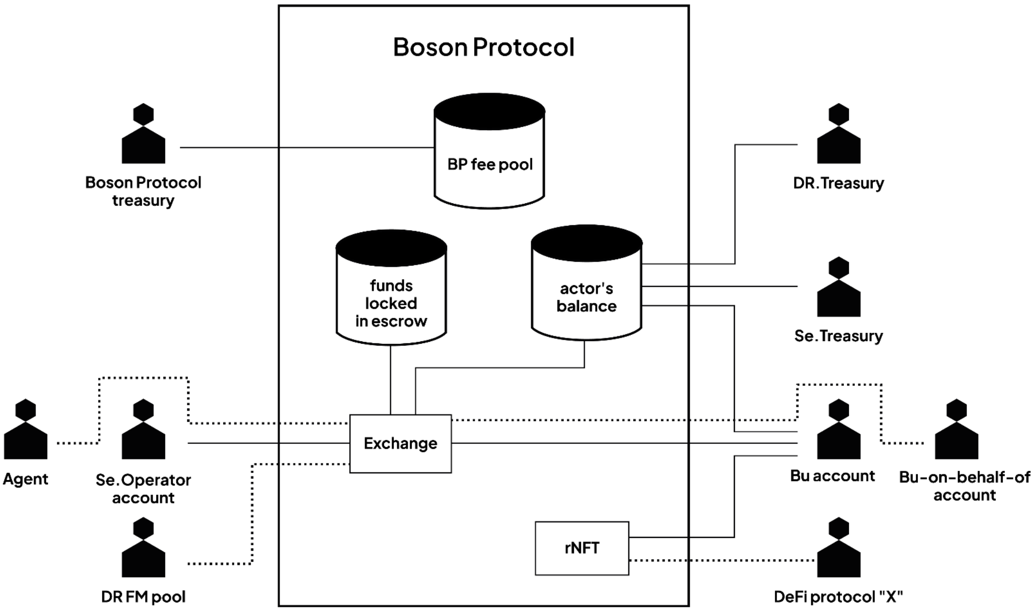

Fig. 8 - *Actors and funds within the protocol*

Core SDK Components

These consist of four main components: the Core SDK, the Contracts SDK, the Subgraph and the Data Storage SDK. The Core SDK provides a single abstraction for interacting with any of the other components; developers can use this as a single dependency to perform read/write operations to the protocol itself, query the Subgraph for rich data, as well as leveraging the Data Storage SDK to store/read data in IPFS. The latter is implemented in an abstract manner so as to allow developers to use the data storage solution of their choosing.

Various other packages and SDKs form parts of these core components; a metadata schema package exists to enable the definition of new metadata formats for offers on Boson Protocol while other helper packages provide common type definitions, a standardised way of rendering a contractual agreement between exchange parties, and more.

Core Visual Components

Visual components consist of libraries providing common UI elements for interactions with the protocol, starting with a React UI kit, and of a set of widgets which provide access to the core functionality of the protocol and can be reused in applications with minimum development cost.

Boson's metaverse libraries enable interactions with the protocol from the metaverse. Decentraland is the first integration to be provided. Support for other various metaverses will be extended in the future by adding metaverse-specific libraries.

Interface Layer

This layer represents various interfaces or front ends that can be built on top of the core protocol using the Core Components. There are numerous possibilities, for example a web marketplace or a metaverse storefront application. The composability of the Core Components allows for applications to be built specifically for the web, mobile web, metaverse, desktop, etc.

The interface layer provides access to a set of hosted front end widgets, which are deployed to IPFS, and allows integrators with minimum technical skills to embed the Boson Protocol flows into their own applications.

Fees and Governance

Protocol Fee

Boson Protocol is designed so that all participants share in the value they create, with fees that are sufficient to sustain and grow the project and its ecosystem. Its governance aims to ensure that Boson remains minimally extractive, forever. Boson Protocol v2 implements a minimally extractive Protocol fee that is configurable by the DAO.

The Protocol fee (PF) is levied on the item Price (p) and is charged for successful exchanges only. PF is between [0,100] and represents the percentage number of the Item Price that is taken as protocol fee.

The fee charged is calculated as $PF = Item\ Price\ *\ PF\ /\ 100$

The Protocol fee is set to 0 if Offers are priced in $BOSON.

Protocol fee revenue will be collected by the DAO treasury, where, ultimately, $BOSON holders will vote on how funds are allocated. This is essential to sustain and grow Boson Protocol itself, and the ecosystem of applications and tooling leveraging Boson.

Incentivization

New networks face the bootstrapping problem, where overall utility is low for users if the overall number of users is low. To overcome this bootstrapping problem, Boson Protocol leverages the $BOSON token.

The Boson Protocol DAO will distribute $BOSON tokens to early users to incentivize early adoption of the protocol. This incentivization will continue until Boson Protocol has achieved strong network effects.

Governance strategy and roadmap

Boson Protocol follows the principle of progressive decentralization. As permissionless public infrastructure for commerce, the question of how Boson should be governed is critical.

We continue to follow our original governance aims:

- fair and equitable distribution of ownership, value and control,
- capture-resistant from centralized, extractive entities,
- regulatory compliance with legitimate authorities,
- community ownership and operation.

	Start-up	Scale-up	Decentralized
Product & Technology	Centralized design & build	More decentralized design & build	Decentralized design & build
Economic sustainability	Zero fees. Early adopter incentives	Signaling vote for fees	Decentralized fee setting
Ecosystem funding	Snapshot proposals with team filtering	2-stage signaling funding	DAO-controlled funding
Treasury management	Foundation	Signaling vote & foundation	Decentralized
Community participation	Involve community in decisions	Split into core units & outsource to ecosystem	Exit to the community
Token distribution	Early buyers and participants	Airdrops to early adopters & contributors	Widely distributed across the community

Boson's decentralization is rapidly evolving from start-up to scale-up and then to a fully decentralized organization. This decentralized organization, the Boson DAO, is a community-governed DAO for shaping the growth of the dCommerce ecosystem.

Boson Protocol is progressively decentralizing the main management and business operations to the ecosystem, particularly to different groups that will take over the development of specific parts of the protocol. Core units will oversee some of the related projects, such as applications built, as well as other operations supportive to Boson Protocol, like marketing and communication activities.

The target end state for the Boson DAO is for tokens to be widely distributed across the community of protocol users, ensuring alignment of incentives and robust governance. The funding in the Boson DAO will be done through grants voted by the DAO and will control the funding of the groups, units, and core communities working to support and grow Boson Protocol. It is important for the protocol to be self-sustaining and finance its own operations, therefore the community will be free to activate, set and distribute the fees via DAO voting. At that time the core team will step back and the community, through the Boson Protocol DAO, will ultimately take responsibility for the design and the build of the protocol. We will make sure that adequate protection is set up against mass token burning and other governance attacks.

Appendix 1 - Boson Protocol Payoff Table

Boson Protocol v2 Payoff Table						
State	**Seller Payoff**		**Buyer Payoff**		**Protocol Payoff**	**Condition for successful execution**
	Protocol Seller Payoff	**Total Seller Payoff**	**Protocol Buyer Payoff**	**Total Buyer Payoff**		
Voided					0	
Revoked	0	- d_{Se}	$d_{Se} + p$	d_{Se}	0	
Canceled	$d_{Se} + c$	c	$p - c$	$- c$	0	$p \geq c$
Completed	$d_{Se} + p - f_{Proto} - f_{Ag}$	$p - f_{Proto} - f_{Ag}$	0	$- p$	f_{Proto}	$f_{Ag} + f_{Proto} \leq p \times MAX_{fee}$
Retracted	$d_{Se} + p - f_{Proto} - f_{Ag} + + E(d) \times d_{Escalation}$	$p - f_{Proto} - f_{Ag} + + E(d) \times d_{Escalation}$	0	$- p - E(d) \times d_{Escalation}$	f_{Proto}	$f_{Ag} + f_{Proto} \leq p \times MAX_{fee}$
Resolved	$(d_{Se} + p + E(d) \times \times d_{Escalation}) \times (100 - P_{Bu}')$	$(d_{Se} + p + E(d) \times d_{Escalation}) \times \times (100 - P_{Bu}') - d_{Se}$	$(d_{Se} + p + E(d) \times \times d_{Escalation}) \times P_{Bu}'$	$(d_{Se} + p + E(d) \times d_{Escalation}) \times \times P_{Bu}' - p - E(d) \times d_{Escalation}$	0	$P_{Bu}' \leq 100$
Decided	$(d_{Se} + p + d_{Escalation}) \times \times (100 - P_{Bu}')$	$(d_{Se} + p + d_{Escalation}) \times \times (100 - P_{Bu}') - d_{Se}$	$(d_{Se} + p + d_{Escalation}) \times P_{Bu}'$	$(d_{Se} + p + d_{Escalation}) \times \times P_{Bu}' - p - d_{Escalation}$	0	$P_{Bu}' \leq 100$
Refused	d_{Se}	0	$p + d_{Escalation}$	0	0	

Appendix 2 - Mutual Resolution Game Theory

This chapter describes Boson Protocol on a high level, its functioning, three proposals for the mutual resolution mechanism and the glossary.

The Contract

The buyer, denoted by B, and the seller denoted by S, engage in the trade. Constant variables of the contract agreed upon by both parties are T_r and T_c. These variables denote different time intervals during the protocol execution. Depending on the final state, the contract defines payments/transfers to both players denoted by a pair of numbers (P_B, P_S), where P_B is a transfer to buyer and P_S is a transfer to the seller.

1. The seller creates an offer of an item for price p and deposit d_S, for the seller. A cancellation fee is included in the price p, and is denoted as c.
2. The buyer signs an offer: she puts up a price p in the contract.
3. There is a time interval T_r, during which a buyer can cancel the trade or a seller can revoke the trade:

1. In case of a revoke or a cancel, the deposit of the player who takes action goes to the other player.
2. Buyer gets his/her price from the contract back minus the cancellation fee if applied. That is, if it is a buyer, final payments are $(P_B, P_S) = (p - c,\ d_S + c)$, state **CANCELLED** is realized. If it is a seller, final payments are $(P_B, P_S) = (p + d_S, 0)$, state **REVOKED** is realized.
3. After T_r time, if revoke or cancel function is not called, both the buyer and the seller are fully committed.

From this point on, during T_C, buyer can complain. If they do not complain we assume the happy path is realized, and we are in **COMPLETED** state. Payments to players are $(0, p + d_S)$.

1. The buyer and the seller both act honestly.
The seller sends the good.
The buyer signs a redemption.
The delivered product is of high quality.
The buyer gets their deposit back.
The seller gets their deposit back.

In this case, **COMPLETED** state is realized.

2. Or during this time T_c, buyer complains, and we are in a new state of contract, called **Mutual Resolution**, denoted by MR. In the following we propose 3 different variants for

the MR game. The MR game results in 3 possible final states: **RETRACTED**, **RESOLVED** and **DECIDED**.

MR game proposals

Two time intervals in this state of the mechanism are T_m and T_p. The first one denotes the time interval where players can agree on a split. The second denotes the time interval where player(s) can propose in case they disagree during the previous time interval.

1. There is a pot $P := p + d_S + d_E$ to split between two players. d_E denotes the escalation deposit paid by a buyer. It prevents buyers from complaining for no reason, as they might lose it later in the protocol. If this deposit is not present, the worst case for the buyer is to pay a price for the item, and therefore, incentive to complain every time. Note that the buyer may get reimbursed this deposit completely, if the dispute resolution procedure decides so or in the state where buyer gets all pot back.
2. The buyer and the seller need to achieve an agreement, in a certain amount of time T_m.

There are two possible outcomes:

1. They achieve an agreement, and a message signed by both is sent to a contract. The amount is split according to an agreement, whether or not it is fair. State is denoted by **RESOLVED**.
2. They do not achieve an agreement, which is captured by the fact that no agreement message is received by a contract.

In the latter case, we assume that there is a procedure that outputs the right split of the pot — the shares players should get.

From this point on there are two different black-box proposals for **Mutual Resolution**, which specify how the players communicate information to the protocol and specifies the transfers/payments to players in such a way that the escalated dispute resolution procedure use is discouraged:

1. Both the buyer and the seller make their own proposals/bids on how to split a pot. The bids are sealed/private. In this case, we may or may not introduce an averaging rule if the bids are close enough.
2. In the second approach, the seller publicly puts down the proposal for the split which the buyer accepts or rejects.

That is, we implement 3 different punishment rules.

1. Bids are blind and there is no averaging, final state is denoted by **ER1**.
2. Bids are blind and there is averaging, final state is denoted by **ER2**.

In both cases, the time allocated to make proposals is T_p. If only a seller makes a proposal, payments are made according to the seller's proposal, the final state is denoted as **RESOLVED**. If only a buyer makes a proposal, then buyer gets all.

If none of the players make a proposal, final payments are made according to the happy path, final state is therefore **RETRACTED**. If the sum of the proposals is not larger than the whole pot $(p + d_S + d_E)$, then the procedure is not called and parties are paid proportionally to their proposals, making in total the pot size. Final state here is denoted as **RESOLVED**.

In **ER1**, procedure is called when both players make a proposal and the sum is larger than the pot.

The only difference between **ER1** and **ER2** is that the latter allows the procedure not to be called if the sum of proposals is not much higher than the size of the whole pot P. In particular, if the sum of proposals: $B_p + S_p \leq 1.1 \cdot P$, then the payoffs are $(P_B, P_S) = (\frac{P}{B_p+S_p} B_p, \frac{P}{B_p+S_p} S_p)$.

In both **ER1** and **ER2**, if the sum of proposals is less than the pot, then resolution is achieved and both players get scaled up payments. In particular, if $B_p + S_p < P$, then $(P_B, P_S) = (\frac{P}{B_p+S_p} B_p, \frac{P}{B_p+S_p} S_p)$. That is, the payoffs still sum up to P.

Bids are open. The seller makes a proposal. The seller has time T_p to make a proposal. If he fails to make it, all the money goes to the buyer and the final state is denoted by **RESOLVED**.

If the seller makes proposal, the buyer has time T_b to accept or reject it. If the buyer does not respond, the payments are made according to the happy path, and the final state is denoted by **RETRACTED**. Note that in this case the buyer escalation deposit d_E goes to the seller. If the buyer does not accept, final payments are made according to the punishment rule, corresponding to a final state **ER3**. If the buyer accepts, payments are made according to agreement, in this case the seller's proposal, and the final state is **RESOLVED**.

The buyer and the seller are allowed to resolve by sending a message signed by both of them at any time of the contract deployment.

Payment formulas in Punishment Schemes

The final outcome for the players in states **ER1** and **ER2** are calculated by the following formula: $max(0, x - c\, max(0, x' - x))$.

1. x is a true share of player $X \in \{S, B\}$, determined by the procedure.
2. x' is the reported share by player X.
3. c is the parameter chosen by the system. For now proposed parameter is 1, but it may increase/decrease depending on experimentation.
4. We take inner $max(0, x' - x)$ to make sure that if a player underestimates his/her own share, we are not rewarding him/her.
5. The amount, $max(0, x' - x)$, goes to a pool that is used to pay the cost of the procedure or is "burnt".

6. We take the outer max function to make sure that all transactions are happening within bounds of available amount, deposited in the contract.

Blind bids can be executed very easily. The buyer and the seller send a decrypted number, e.g., by sending $g^m \, mod \, p$, where p is some large prime number, m is their bid/message and in this situation, the key, and g is some generator. After both bids are received, parties need to send their keys (bids). If neither sends it in a certain time interval, then the protocol defaults to the happy path. If only one of them sends it, the protocol will make payments according to the proposal of the sender. If both of them send their keys, then protocol proceeds as described above.

The final outcome for the players in the state **ER3** is calculated by the following formula:

1. If the seller proposal is correct or lower than the correct amount (determined by the dispute resolver), the buyer is punished by subtracting 10% from his payment, while the seller gets his true share.
2. If the seller overestimates his share by any amount, then the seller is punished by subtracting 10% from his own share, while the buyer gets his own share fully.

Rationale for mutual resolution mechanism

The protocol's main goal has to be to minimize the usage of Escalated Resolution (ER). Minimizing the incidences where ER needs to be utilized, helps us decrease both protocol and users' psychological costs. In general, making sure any disagreement process is handled in the mechanism rather than as part of ER ensures that the protocol serves the intended purpose of ensuring transactions happen in an automated and reliable manner. We conjecture that it will also increase the trust of the users into the system.

While deposit repayments on the happy path should be $p + d_S$ to seller and 0 to the buyer, in case of dispute, they may split it differently, depending on the state. The latter holds even when the players reach an agreement and do not call the dispute resolution procedure. The contract should know this split to make transfers to players, so the agreement should specify the split.

In case of not reaching agreement, we record at most two additional numbers, one proposal from each player or just one proposal from the seller. Recording these numbers must also be cheap on-chain operations. Both players are punished proportionally to their deviation from the truth that the procedure outputs. This creates pressure on both players to lower their proposed shares and reach an agreement. Such punishment also protects the system from malicious attackers who might call the costly dispute resolution procedure each time just for the damage of the system.

Another way to protect from such attacks is to make dispute resolution costly for the users, but we rejected this approach for several reasons, for example, this cost might be irrelevant compared to the total deposited amount or it can be too high in some cases. For paying such costs there are few proposals in place:

1. protocol creates a pool that pays ER costs.
2. traders pay insurance fee.

Appropriate deposits have to be made in order to make the threat of dispute resolution procedure credible, even though it will not happen on an equilibrium path for most of the types of players. We may want to make suggestions on deposit levels in the future.

Comparison between proposed MR games

1. **ER1** and **ER2** record 2 proposals as transactions, while **ER3** records only 1 proposal from the seller as a transaction, and 1 transaction for the buyer to accept or reject it.
2. **ER1** and **ER2** allow both players to make proposals. In a way it is a fairer procedure, compared to **ER3** which allows only the seller to make a proposal.
3. Because of the previous point, we expect agreement to reach more often in **ER3** than in **ER2**. Note than **ER1** reaches agreement (weakly) less times than **ER2**.

Properties of proposed MR games

4. Fairness: mechanism treats both players equally. Assuming DR is fair, protocol fairness is readily implied.
5. Cost- and time -effectiveness: lowers the cost of financing DR, as it is called less; lowers the time for the same reason.
6. UX impact: 2 additional transactions.

Experiments

First Experiment

Short report on the experiment results conducted on 23rd July, 2021:

We had 36 registered participants, that is, 18 pairs of players. In the end, 9 pairs were formed, since many registered participants did not show up. We had pre-arranged the pairs and their numbers. This will not be the case in a real experiment with real money. We will arrange numbers/pairs as people will arrive and we expect everyone to arrive.

Out of 9 pairs, 8 pairs reached an agreement. Only 1 pair did not reach an agreement, but they said the reason was not enough time to calculate the agreement.
There were 2 basic ways of reaching the agreement:

1. split the excess amount equally: e.g., if numbers were 50 and 70, the excess is 70 + 50 – 100 = 20, and they would agree on 50 – 20/2 = 40 and 70 – 20/2 = 60.
2. split the excess amount proportionally: e.g. if numbers were 50 and 70, the excess is 20, and they would agree on 50 – 5/12 * 20 = 41.66 and 70 – 7/12 * 20 = 58.33.

In only one case it happened that the lower number holder lowered his/her number more than the higher number holder, but we did not have a chance to enquire what exactly happened. It can be that the high number holder cheated on his/her number and reported an even higher number and then applied one of the two ways described above. In the interviews that we conducted nobody overshot/cheated about his/her number. This may and will change with the real experiment, but we do not expect it to happen often.

One misunderstanding reported after the interview: 2 people thought they would be "punished" even in case of agreement.

Second Experiment

Update on the experiment with real incentives: We conducted another pilot experiment on 12 people (master students with math/CS background) with real incentives. 6 teams, each of our 3 experiment setups played twice. In all games agreement were achieved, with fair division in 5 games, and one unfair division. In this game, real numbers were 35 and 65, while they received 45 and 75. We added expected value 10 to all numbers, but the instructions still said uniform random number in $[0, 20]$ interval was added. Second player asked the first player what their received number was. The first player said the correct number, and the second player said he matched with the first player's number so that the sum of both numbers was almost 140, in fact he said his number was 93, instead of 75, arguing that both of them got almost maximally deviated numbers. Then he offered to agree both of them to decrease, him 20 and the other 18, leaving them with 27 and 73. He used the so called second mover advantage.

We got 3 feedbacks from the pilot and participants:

1. higher punishment means more chances to agree but more extortion as well.
2. sequence of moves and types of players matter. While this type of player observed above is not surprising ex-post, it was quite unexpected ex-ante. In reality, there will be such players, and we can not avoid them. But if the first player in the above mentioned game was better, he would for example ask for the proof of receipt of his numbe.
3. while this experiment is a good approximation, it does not reflect reality completely. E.g., in the experiment, players do not lose anything. The worst thing that can happen to them is not to win. When it comes to losing their own money, behavior will change (maybe to better). However, we can not experiment on that, at least in the framework of academia.

Theoretical Set-Up for the Analysis

The revoke and cancel functions of the mechanism serves two purposes. First, it allows both the buyer and the seller to stop the procedure and leave the contract if something changes in their corresponding agendas. Second, if one player calls the function, the deposit of this player goes to the other player, which pays-off the opportunity costs of waiting incurred by the other player. This function happens before the final commitment is made and it is incentive compatible. Therefore, for us the game starts with the final commitment.
There are three essential reasons why the agreement may not be achieved.

1. Overestimation of one's share. This behavior is well documented, both in the empirical and experimental literature. One of the main reasons is lack of information.
2. Selfishness, or in other words, maximizing own payoff.
3. Maliciousness, that is, trying to minimize the payoff of the other player. This aspect deals with potential attackers, for example, a competitor seller setting up a fake buyer and trying to steal as much as possible from the pot, with the final goal to discourage the seller from trading.

The first two aspects can be measured with the same number, denoted by g_X for player $X \in \{S, B\}$. We will need a different number for the third quantity, denoted by m_X, for player $X \in \{S, B\}$. The final payoff of the player X is then calculated as $u_X = g_X t_X - m_X t_Y$, where t_X is the payoff made by the mechanism to player X, and t_Y is the payoff made to the other player $Y \in \{S, B\} \backslash X$. Our goal is to come up with a mechanism that ends up with an agreement for large sets of parameters (g_S, m_S, g_B, m_B).

The strategy set of players' is $[0, 1]$, that is, they can choose any number between 0 and the maximum, the amount they get. Since the bargaining game is hidden and we see only the outcome how the players agree on, this is a reasonable assumption. We consider the solution concept of the pure strategy Nash equilibrium.

1. We say that a set of parameters (g_S, m_S, g_B, m_B) is good if in any Nash equilibrium of the simultaneous game, agreement is achieved.
2. The set of parameters (g_S, m_S, g_B, m_B) are implementable, if there is a Nash equilibrium solution in which the agreement is reached.
3. The set of parameters is non-implementable if there is no Nash equilibrium solution in which the agreement is achieved.

Punishing scheme of the mechanism defines the set of Nash equilibria solutions, and therefore, level of implementability of parameter sets defined above. Exactly which Nash equilibrium solution will be implemented is determined by Nash Bargaining solution concept. Our punishment scheme defines the set of disagreement points, instead of one disagreement point in a classic Nash Bargaining Problem.

For the simplicity of the analysis, we will assume that parameters g_X and m_X are distributed independently at random by distributed functions G and M. In reality it might be that there are huge correlations between the two parameters. In that case we will have to assume a joint distribution GM. The goal of the experiment is to get estimates of the distributions. Later, once the mechanism is deployed, we will refine the estimates, to maybe fine tune the parameters of the punishment function. Initially we may assume that they are uniform random.

Escalated dispute resolution procedure

In this section, we propose one particular scheme for dispute resolution procedure. Assume that there is a pool of experts or judges. For each dispute resolution, we choose $2n + 1$ members of the pool of judges uniformly at random. They have to make a decision who between the buyer and the seller gets how much of the whole pot P. Therefore, each member has a strategy set equal to the closed interval $[0, P]$. Every member of the committee, generic member denoted by i, sends a message a_i, denoting how much the buyer gets. We sort these numbers in the increasing order $a_1 \leq a_2 \leq ... \leq a_n \leq a_{n+1} \leq ... \leq a_{2n+1}$. The outcome is the median of this set of numbers, a_{n+1}. Therefore, the seller gets $P - a_{n+1}$. Suppose members of the committee have single-peaked preferences, that is, they have an ideal point of the split on the line and further is the split from that point, less utility they derive. Then we get the following proposition:

Proposition: The procedure described above is dominant strategy incentive compatible (DSIC).

Proof: Consider any member i of the committee, and suppose the others report numbers $a_1 \leq a_2 \leq ... \leq a_n \leq a_{n+1} \leq ... \leq a_{2n}$. Suppose the saddle point of i is s. There are 3 cases:

1. $s \leq a_n$. In this case, announcing his right preference s implements a_n. Announcing anything lower than a_n also implements a_n. However, if i announces something bigger than a_n, it results into implementing a result strictly bigger than a_n, which is worse for i. Therefore, it is the best strategy to announce s.
2. $s > a_{n+1}$. This case is similar to the previous case. Announcing his right preference s implements a_{n+1}. Announcing anything higher than a_{n+1} also implements a_{n+1}. However, if i announces something smaller than a_{n+1}, it results into implementing a result strictly smaller than a_{n+1}, which is worse for i. Therefore, it is the best strategy to announce s.
3. $s \in [a_n, a_{n+1}]$. In this case announcing s implements s as a result, which is a saddle point for i.

We showed that in all three possible cases, announcing true preference is optimal, regardless of others' preferences or their announcements, which implies that this mechanism is dominant strategy incentive compatible.

We take small values of n, for example 1 or 2. That is, committee size is 3 or 5.