Squad Games Platform Paper

Jesse B. Miller, Ezra Weller

Vision

This paper describes Squad Games, a community-owned infrastructure platform for video game mods.

The Opportunity of Video Game Mods

In the language of videogames, a "mod" refers to a modification created by fans: a new level, a new character, or even an entirely new game made by rearranging the original's pieces.

Many of today's [most popular video games](#) have benefited greatly from mods, including:

- Games that grew significantly because of their mods: Minecraft, Warcraft III, Starcraft, Roblox, Arma 3, the Elder Scrolls series, Half Life, and many more...

- Games originally invented as mods:
  - Battle royales (Fortnite)
  - MOBAs (League of Legends, Dota 2)
  - Counter Strike-style games (CSGO and CrossFire)
  - Team Fortress-style games (Overwatch)
  - Autobattlers (Teamfight Tactics)
  - Tower defense games (Flash Element TD)

Mods and mod-making, however, remain inaccessible to most players and game developers:

- Mods are difficult to create, as adding robust mod-making tools is time-consuming and expensive for game developers.
- Game developers and mod-makers lack well-aligned incentives: mod-makers typically can't draw income from or get public credit for their mods, and game developers have no sustainable way to incentivize mod-making.
- Good mods are difficult to discover: games with large modding communities lack effective curation standards for their mods.

In a world where more games are able to fully support mods:

- Mod-making, and enabling mod-making for your game, are easier. More games offer robust modding tools, more mods get made, and more value is created for players.
- Mod-makers have proper incentives to create mods, and game developers have the incentive to support them. Mod-makers are able to take public credit and draw revenue from their work, and the popularity of their mods also brings additional attention and revenue to the parent game.
- Great mods are highly discoverable, letting players efficiently find the mods they most prefer. It's easy for game developers to incorporate good discoverability into their games.

The Squad Games Vision

With a standard for accessible modding, game communities can become powerful economic engines—multi-sided marketplaces with strong network effects—bringing to videogames the kind of innovation Youtube enables for video content. This is the vision for Squad Games.

We imagine an ecosystem with millions of games and mods, evolving more quickly and effectively than any set of games developed by an isolated team could. In this ecosystem:

- Inside each game, you see a dashboard of different ways to play, with the most compelling to you shown first.
- You can create and submit new mods from within each game, creating new maps, items, and formats, or rebalancing existing ones. The range of mods is wide, from new cards in a card game to full RPGs in [RPGmaker](.).
- Games often share revenue directly with mod-creators, and vice versa; many mods share revenue with games when purchased directly.

Fundamentally, this ecosystem employs the same engine that powers the social internet today–the combination of open submission, aligned incentives, and effective curation–to the benefit of game developers, mod-makers, and players.

The Squad Games Platform

Squad Games is a platform that enables video games to support mods in three main ways:

- **Easier modding**: development tools for building games that support mods and community mod submission.
- **Compensating mod-makers**: mod-makers own the content they contribute and can earn income from their success.
- **Mod discoverability**: mods are curated using market data, so game players can effectively discover the mods they will enjoy most.

Platform Specification

The Squad Games platform includes:

- the Squad Games SDK, a development kit for building mod-enabled games, and
- squad.games, a showcase and store for games built using the SDK.

Squad Protocol

At the core of the Squad Games Platform is the Squad protocol, a simple protocol for adding licenses to user-submitted content.

Contributions

Data is submitted to the Squad protocol in the form of "contributions," which have two parts:

- *Content*, which contains the core information, or a reference to the information, of the contribution. In Squad Games, this will often be a mod.

- *License*, a set of rules (code or human language) that dictate the legal usage of the content, including financial stipulations.

Submitting Contributions, Storing Data, and Creating Markets

The Squad protocol provides a method for submitting contributions and associated market configuration that stores the contribution and makes it publically available. It also creates a market according to the market configuration that fulfills the financial portion of the license (i.e. it allows the compliant buying and selling of rights to use the contribution). The contribution and its market are linked by an identifier determined by hashing the contribution's content.

The protocol is not opinionated about the method of storage.

Squad Games SDK

The SDK is a game development toolkit that consists of APIs for:
- submitting particular contribution types to the platform's implementation of the Squad protocol,
- interacting with the markets for those contributions, and
- curating those contributions using market data.

Content Schema

(For *easier modding*)

The Squad Games SDK provides some video-game-mod-specific content schemas for contributions:

- **Game contributions**, which represent games and can be referenced by components and formats, both in revenue sharing and in identifying the games in which a given component or format is used. Game contributions optionally contain the information needed to launch a game.

- **Component contributions**, which are game-specific components like cards in a collectible card game, a hero or item in a MOBA, or a level in a platformer. Components contributions contain mod data and an array of associated games.

- **Format contributions**, which are sets of contributions that a game uses all together, like a set of balanced heroes and items for a MOBA, a set of levels and how they are connected in a platformer, or a set of legal cards for a card game. Formats have the same content pattern as components, plus they must include a list of existing contributions.

Supported Licenses

(For *compensating mod-makers*)

The SDK supports an initial set of contribution licenses and markets, with goals to provide standard copyright protections for creators, consistent income for creators when their creations are used, and revenue sharing for creators whose contributions are referenced or used by other contributions.

Specifically, the initial license set will include:
1. A standard copyright-providing license that protects against plagiarism, which may draw on typical open source licenses
2. A license that requires users to purchase contribution-usage rights for a constant price
3. Revenue sharing licenses that:
   a. require that some portion of the revenue earned by a contribution be paid to its creator
   b. require formats and components to share revenue with their associated games
   c. require formats to share revenue with their listed contributions

In supporting these licenses, the SDK will verify whenever possible that submitted contributions have valid licenses. For example, when a user submits a new format contribution, the SDK will check the licenses of the components listed in the new format and ensure the format's license complies with the components' licenses.

Automatic Markets

(For *compensating mod-makers*)

The SDK can create markets that support any valid combination of the licences listed above, but these markets also provide other benefits, such as data used to curate contributions and map stakeholders (used for governance of the Squad DAO; see below).

The first iteration of the SDK focuses on a single market archetype: linear bonding curves with constant purchase prices and revenue sharing.

"**Bonding curves**" are a type of automated market maker (AMM): a market that provides liquidity via algorithm instead of user-submitted buy and sell orders. Each bonding curve lets users buy and sell a class of token unique to that curve. Buy and sell prices are determined by the current circulating supply of the token and monotonically increasing price functions. When users buy, they receive new tokens, and their payment assets are sent to the curve. When users sell, they receive payment assets from the curve, and their tokens are destroyed.

"Linear" bonding curves have linear buy and sell functions, so the price of their tokens increase and decrease at a constant rate as the supply changes.

"Constant purchase prices" refers to receipts generated by a bonding curve's buy function that record the purchaser, buy price, and number of tokens received. These records are used as purchase receipts for fulfilling licenses.

"Revenue sharing" bonding curves contain a set of addresses with which they share revenue. Any time the curve's buy function is called, the incoming funds are split between the curve and its revenue share addresses. If revenue is being shared with a contribution, shared revenue is used to call the buy function in that contribution's market. If revenue is being shared with a non-contribution, the funds are sent directly to that address.

Beyond fulfilling the SDK's supported licenses, these bonding curves also provide inherent investment opportunities, since their token sell prices go up as their supplies increase. Accordingly, the SDK offers contribution creators a right to first purchase when submitting a contribution, giving them the best opportunity to invest in their contribution's success. The ability to invest in contributions also improves curation quality, since it incentivizes the discovery of new valuable contributions. Increasing token prices will not negatively affect game players, since they will primarily interact with the bonding curves via constant purchase prices.[1]

Example market interaction:

> *Alice submits a new format, buying 100 tokens of the format for herself at the cheapest possible price thanks to her right to first purchase. Her format contribution has a license that says addresses must have purchased at least 10 DAI worth of tokens in order to use it. The license also says this revenue must be shared with the format creator (10%) and the two component contributions referenced in the format (10% each).*
>
> *I want to play using Alice's format, so I click 'buy' to purchase it. The game charges me 10 DAI, calling the format's buy function. 1 DAI goes to Alice directly, 2 are sent to the referenced components' markets, and 7 go to buying format tokens. The number of format tokens received is recorded, along with the 10 DAI I paid, in an event on the Ethereum blockchain. The game sees this event, checks that I still hold at least that many tokens, and if I do, it allows me to play using Alice's format.*

---

[1] Securities law is not being dealt with in this paper, but it is relevant here and may affect these market designs.

> *Because of the properties of the bonding curve, each time I or another user purchases the right to play her format, the tokens Alice initially purchased go up in value. She can exchange them for funds anytime.*

Note that there are many other ways to create markets to fulfill the currently supported licenses, as well as many more potential licenses. We expect the number of licenses and markets supported by the SDK to expand over time.

Data Storage

(For *easier modding*)

The SDK will provide contribution data storage and retrieval services to applications. These services are currently hosted centrally but may transition to decentralized storage systems such as IPFS or Arweave in the future.

Curation

(For *mod discovery*)

Data generated by buys and sells in Squad markets can be used to curate contributions effectively, and the SDK will use this data to provide curated lists of contributions.

Most curation on the internet today is popularity-based. The number of upvotes, likes, shares, or purchases a piece of content receives largely determines how often platforms show it to users. Squad market data makes most such curation methods possible, and the Squad Games SDK will include a number of them.

To begin with, the SDK will order contributions by the total value locked in their bonding curves and by its acceleration. Contributions will also be sortable by the timestamp of their creation ("newness").

Later on, the SDK may also support curation methods based on the following:
- User behavior, such as a user's purchase history or the similarity of their history with others' may be added (many of the same techniques employed by mainstream recommendation systems such as Amazon's can be applied here)
- A Pagerank-style algorithm based on inter-contribution references plus market data

Game Showcase & Store: squad.games

Squad.games is a showcase and store for game contributions.

We have described the Squad Games platform as focused on video game *mods*, but astute readers will have noted that game contributions aren't always mods. Games aren't composite parts of existing applications; they *are* applications (in this case, references to applications,

specifically). We've chosen to include game contributions for a few reasons:

- It gives games access to the SDK's revenue sharing system
- In some cases, game contributions are mods – e.g. when they are acting as minigames inside other games, when they are a "remix" of another game
- Since the SDK will allow anyone to submit and access data to its implementation of the Squad protocol, the SDK's game catalog will act as a game network unbundled from any single storefront. Any game store can feature Squad Games contributions.

We believe game contributions will play an important role on the Squad Games Platform, but because they reference standalone applications, they will not often be displayed inside other games themselves. This is the purpose of squad.games: to provide the venue for showcasing game contributions.

Usage Example: Squad Chess

Squad Chess (chess.squad.games), a version of chess integrating the SDK, was built to demonstrate how game developers, mod-makers, and players may interact with Squad Games.

Game Development

Games using the SDK should be meta-data driven, meaning significant pieces of the game should be abstracted as metadata. The parts of the game defined this way can then be filled by any user-submitted mod fitting the metadata pattern.

Squad Chess uses two metadata patterns:
- chess piece: a name, an image, and a list of valid mechanics and inputs (with valid mechanics, including "move" and "capture" being hardcoded in the game)
- chess format: a list of pieces, a board shape, and a starting position

```
{
  Component: {
    name: 'Pawn',
    data: JSON.stringify({
      mechanics: {
        move: [
          { offset: [0, 1], steps: 1 }
        ],
        capture: [
          { offset: [1, 1], steps: 1 },
          { offset: [-1, 1], steps: 1 }
        ]
      },
      admechanics: {
```

```
        randomPromotion: ['default']
      },
      graphics: {
        local: {
          white: 'chesspieces/wikipedia/wP.png',
          black: 'chesspieces/wikipedia/bP.png'
        }
      }
    })
  }
}
```

Defining a chess piece in Squad Chess

Mod-Making

Taking advantage of this meta-data orientation, the Squad Chess user interface provides forms for users to build and submit their own chess pieces and formats without writing code. These forms use the Squad Games SDK to create new contributions, letting end users of Squad Chess submit monetized mods directly from the game's UI.

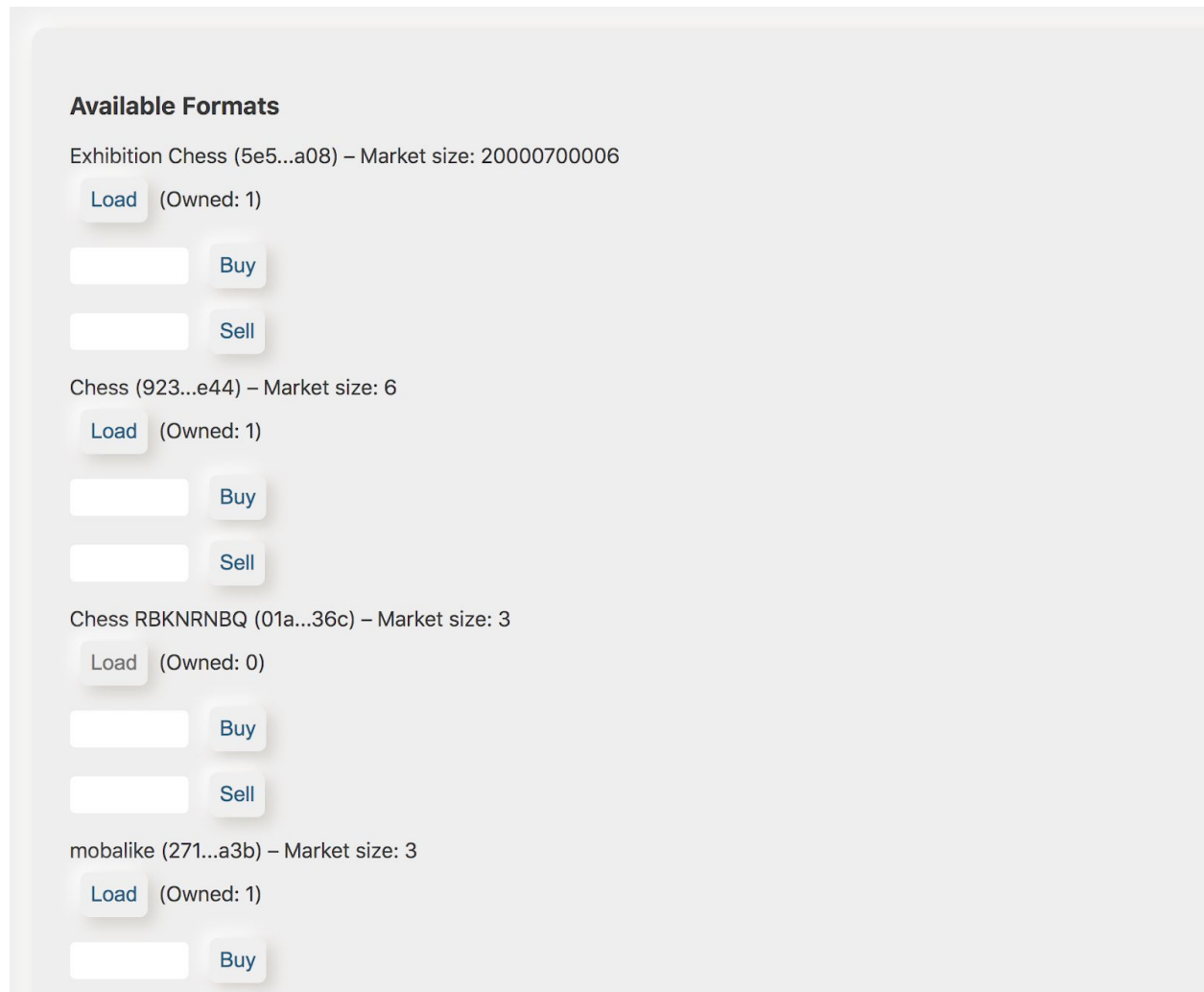*An early build of Squad Chess' format creation form*

Mod-maker Compensation

Squad Chess is free to launch but requires players to buy the rights to play a format before using that format. Players see a list of formats available to buy, and a list of playable formats they already own. Clicking "buy" for a format starts an Ethereum transaction to send funds to the format's market. Using licenses and markets, the revenue from these purchases is automatically shared with the format creators, with the creators of the pieces used in the formats, and with the Squad Chess game creators (the authors of this paper).

Mod Discovery

Using the SDK, Squad Chess displays chess formats in order by market valuation (the area under the bonding curve's sell function) and by newness in case of a tie. It uses the same curation method to display chess pieces in the format-creation form.

This curation method is simple, but it means that any new user visiting the application will be able to quickly find the formats other players are enjoying most. Similarly, users creating new formats have a convenient way to add popular pieces to their format.

Available Formats

Exhibition Chess (5e5...a08) – Market size: 20000700006

Load (Owned: 1)

[   ] Buy

[   ] Sell

Chess (923...e44) – Market size: 6

Load (Owned: 1)

[   ] Buy

[   ] Sell

Chess RBKNRNBQ (01a...36c) – Market size: 3

Load (Owned: 0)

[   ] Buy

[   ] Sell

mobalike (271...a3b) – Market size: 3

Load (Owned: 1)

[   ] Buy

The first few formats of Squad Chess' curated format list

Organizational Structure

Squad Games will undergo [progressive decentralization](#), launching with a centralized team but eventually being operated by a decentralized, autonomous organization: the Squad DAO.

We choose this approach for three reasons:

1. **Growth potential**: the opportunity for participants to earn ownership shares of the platform is a strong, novel incentive for participation. Legal decentralization also opens up new funding opportunities for the community.

2. **Sustainability**: a well-designed stakeholder model and governance system ensures that the DAO will continue to make decisions aligned with the Squad Games ecosystem, regardless of what happens to the original leadership team.
3. **Ethics**: common organizational models like corporations and nonprofits are oft-criticized for treating their workers and/or customers unfairly. Both groups are included in governance here.

Stakeholder Model

Good decentralized governance requires power to be distributed to people according to their stake in the system. Squad Games has a few major stakeholder groups:

- Users, the people purchasing rights to use contributions
- Contributors, the people creating and submitting contributions
- Investors, the people who put their money behind a belief in the Squad Games community
- Workers, the people building the Squad Games Platform

Though these groups have disparate types of stake, we attempt to reduce all their stakes to a single variable—voting power—in order to include them in the DAO.

User, contributor, and investor stake can all be reasonably modeled by tokens from bonding curve markets. Users buy tokens in order to get utility, contributors buy in order to profit from their work, and investors buy in order to get a return later. We can approximate an account's stake in the system at a specific time by summing the sell value of all the tokens that account currently owns.

Worker stake can be modeled by the monetary value of the work workers do: each time the DAO commissions a project, it may give voting power to the commissioned workers pro rata to their pay.

The stake-to-voting-power model is yet to be detailed fully.

Governed Territory

The Squad DAO will have two main areas of responsibility:

Build and maintain the software
- Canonize official versions of the Squad Games SDK and the squad.games showcase.
- Configure the SDK and squad.games:
  - Decide which licenses and markets the SDK should support
  - Set parameters, such as a potential global market tax to fund the DAO

Operate the platform

- Enforce platform terms of service (e.g. community-governed censorship of misbehaving contributions)

Path to Decentralization

Squad Games will launch under centralized control and progress toward decentralized governance over time, following this plan:

Stage 1: Product-market fit

At this stage, the Squad Games Platform will be controlled by a central team focused on building the platform and raising capital (financial and community). If the platform achieves sufficient adoption, we will move to the next stage.

Stage 2: Decentralization

The process of decentralization will begin with a testing phase to help ensure the DAO is capable of governing the system. Once confident in the DAO structure, the team will begin distributing ownership power to stakeholders and launch the fully decentralized governance system.

Extrapolation

In this paper, we outlined the missed opportunity of video game mods and how the Squad Game Platform aims to meet it by building a community-owned, cross-game modding platform. Looking further ahead, the Squad Games approach to mods might be generalized to more types of user-submitted content.

User content today is static and siloed: users give up control of their content to centralized platforms like Youtube and Twitch, and multi-author content is difficult or impossible.

With a Squad-style platform unbundling content from applications, the monopolistic network effects of platforms like Youtube might weaken. New competitors could duplicate the entire content network from day one, increasing inter-platform competition and giving creators more platforms on which to show their work. Squad might also enable much stronger collaborations between content creators: imagine libraries of music and stock footage available to video creators to seamlessly use, credit, and share revenue with.

The Squad approach to user content may also unlock new markets for user-submitted content. Squad Games is focusing on mods for video games, but with few changes, the same mechanics

can be applied to custom loadouts in any kind of software. The idea is not that strange: Twitter hashtags began as a [user-proposed "mod" for the Twitter application](#), for example. The value engine of curation-plus-open-submissions has largely been limited to single author content like videos and blogs thus far: Squad may unlock it for multi-author content like software and publications.