# ProtoMediaShares Proposal

## Objective

* Prove DAC’s (Decentralized Autonomous Corporations) can be implemented successfully
* Decentralize distribution of all forms of media including but not limited to news, movies, music and books
* Create a viable way for people to make money from non-tangible goods such as text, video and audio via mass distribution whilst making the cost free for the end user
* Remove all censorship of media
* Allow for people to submit content anonymously
* Allow for people to view content anonymously
* Encourage the highest quality of content possible through market mechanisms

## Proposed Model

* A user has a form of media they wish to distribute to a wider audience
* The user purchases a “MediaShare” specific to the media they wish to release
* The user then creates a “link”[[1]](#footnote-1) between the MediaShare andthe desired file
* Now the user is ready to add their MediaShare to the market
* The file can now be viewed and distributed by anyone with there being no copyright on the content
* Each time the file is distributed through the network the creator of the content will receive a percentage of the transmission fees
* If an end user likes the content they accessed, they can choose to buy into the creators’ shares, which will entitle them to a percentage of the fees generated from the distribution on the network.
* Alternatively if an end user feels the content was poor they can go short and make money on the falling value of the MediaShare.
* The MediaShare can then be held onto for the Dividend Payment or sold in the market for profit
* Ultimately producers of high content will see high initial costs for adding their content as speculation surrounding their abilities will drive the cost up but they will be redeemed through more vested interests in their work and thus a higher return on the transactions fees
* An end user doesn’t need to buy into the shares at all and can simply enjoy the content for free

## Identified Problems

Problem 1: Possibility for people to plagarise content and submit as own

Solution: Have a system in place which automatically flags content thought to be copied from existing content, then the content is flagged as such it is sent to random anonymous third parties (possibly 7) who check the flagged content against the match in the system. If >50% believe it to be a copy it is added to the system, otherwise it is rejected. Coupled with this, individuals who check content for plagiarism will receive a payment based on their quality rating which is worked out at each time by matching their answer to the other 6 users. Ultimately people trying to exploit the system will fail and give up.

Problem 2: Linking various Media to the MediaShares

Solution:

Problem 3: Mechanism for contributors to purchase shares and distribute them effectively

(Possible)Solution: Possibly have the process go, the creator buys in for any bit BitUSD amount they wish and receives ¼ of MediaShares produced for the article. The remaining ¾ are available for the market to purchase. In our example lets say the author feels the price of a share in the article is BitUSD5. We will in this example say all new media receive 1000 shares so the author in this instance would get 250 shares, with 750 available to the market. The total value would be BitUSD5000 but this would be in the same way as an IPO would occur so the value would be only based on what the author feels they are worth. The article is then released into the wild, when a person reads the article they will have an option to buy if they agree with the price per share thereby giving them the amount of shares they purchased / 1000 percent of royalties generated from transmission fees. Ultimately, if people agree that the price set by the creator is acceptable all the 750 shares will be purchased and bids will appear if it’s a good investment and the price of the Media Share will rise

1. Weather this is via “linking” it to where the file is stored or by embedding the file to the MediaShare is yet to be determined [↑](#footnote-ref-1)