On Medium-of-Exchange Token Valuations

One kind of token model that has become popular among many recent token sale projects is the "network medium of exchange token". The general pitch for this kind of token goes as follows. We, the developers, build a network, and this network allows you to do new cool stuff. This network is a sharing-economy-style system: it consists purely of a set of sellers, that provide resources within some protocol, and buyers that purchase the services, where both buyers and sellers come from the community. But the purchase and sale of things within this network must

be done with the new token that we're selling, and this is why the token will have value.

If it were the developers themselves that were acting as the seller, then this would be a very reasonable and normal arrangement, very similar in nature to a Kickstarter-style product sale. The token actually would, in a meaningful economic sense, be backed by the services that are provided by the developers.

We can see this in more detail by describing what is going on in a simple economic model. Suppose that (N)

people value a product that a developer wants to release at (\$x)

, and believe the developer will give them the product. The developer does a sale, and raises (N)

units for (\$w < x)

each, thus raising a total revenue of (\$Nw)

. The developer builds the product, and gives it to each of the buyers. At the end of the day, the buyers are happy, and the developer is happy. Nobody feels like they made an avoidable mistake in participating, and everyone's expectations have been met. This kind of economic model is clearly stable.

Now, let's look at the story with a "medium of exchange" token. (N)

people value a product that will exist in a decentralized network at (\$x)

; the product will be sold at a price of (\$w < x)

. They each buy (\$w)

of tokens in the sale. The developer builds the network. Some sellers come in, and offer the product inside the network for (\$w)

. The buyers use their tokens to purchase this product, spending (\$w)

of tokens and getting (\$x)

of value. The sellers spend (\$v < w)

of resources and effort producing this product, and they now have (\$w)

worth of tokens.

Notice that here, the cycle is not complete, and in fact it never will be; there needs to be an ongoing stream of buyers and sellers for the token to continue having its value. The stream does not strictly speaking have to be

endless; if in every round there is a chance of at least (\frac{v}{w})

that there will be a next round, then the model still works, as even though someone

will eventually be cheated, the risk of any individual participant becoming that person is lower than the benefit that they get from participating. It's also totally possible that the token would depreciate in each round, with its value multiplying by some factor (f)

where (\frac{v}{w} < f < 1)

, until it eventually reaches a price of zero, and it would still be on net in everyone's interest to participate. Hence, the model is theoretically feasible, but you can see how this model is more complex and more tenuous than the simple "developers as seller" model.

Traditional macroeconomics has a simple equation to try to value a medium of exchange:

(MV = PT)

Here:

- (M)

is the total money supply; that is, the total number of coins

- (V)

is the "velocity of money"; that is, the number of times that an average coin changes hands every day

- (P)

is the "price level". This is the price of goods and services in terms of

the token; so it is actually the inverse

of the currency's price

- (T)

is the transaction volume: the economic value of transactions per day

The proof for this is a trivial equality: if there are (N)

coins, and each changes hands (M)

times per day, then this is (M \cdot N)

coins' worth of economic value transacted per day. If this represents (\$T)

worth of economic value, then the price of each coin is (\frac{T}{M \cdot N})

, so the "price level" is the inverse of this, (\frac{M \cdot N}{T})

.

For easier analysis, we can recast two variables:

- We refer to (\frac{1}{V})

with (H)

, the time that a user holds a coin before using it to make a transaction

- We refer to (\frac{1}{P})

with (C)

, the price of the currency (think (C = cost)

)

Now, we have:

(\frac{M}{H} = \frac{T}{C})

(MC = TH)

The left term is quite simply the market cap. The right term is the economic value transacted per day, multiplied by the amount of time that a user holds a coin before using it to transact.

This is a steady-state model, assuming that the same quantity of users will also be there. In reality, however, the quantity of users may change, and so the price may change. The time that users hold a coin may change, and this may cause the price to change as well.

Let us now look once again at the economic effect on the users. What do users lose

by using an application with a built-in appcoin rather than plain old ether (or bitcoin, or USD)? The simplest way to express this is as follows: the "implicit cost" imposed by such a system on users the cost to the user of holding those coins for that period of time, instead of holding that value in the currency that they would otherwise have preferred to hold

.

There are many factors involved in this cost: cognitive costs, exchange costs and spreads, transaction fees, and many smaller items. One particular significant factor of this implicit cost is expected return. If a user expects the appcoin to only grow in value by 1% per year, while their other available alternatives grow 3% per year, and they hold $20 of the currency

for five days, then that is an expected loss of roughly (\$20 \cdot 2% \cdot 5 / 365 = \$0.0054)

.

One immediate conclusion from this particular insight is that appcoins are very much a multi-equilibrium game

. If the appcoin grows at 2% per year, then the fee drops to $0.0027, and this essentially makes the "de-facto fee" of the application (or at least a large component of it) 2x cheaper, attracting more users and growing its value more. If the appcoin starts falling at 10% per year, however, then the "de-facto fee" grows to $0.035, driving many users away and accelerating its growth.

This leads to increased opportunities for market manipulation, as a manipulator would not just be wasting their money fighting against a single equilibrium, but may in fact successfully nudge a given currency from one equilibrium into another, and profit from successfully "predicting" (ie. causing) this shift. It also means there is a large amount of path dependency, and established brands matter a lot; witness the epic battles over which fork of the bitcoin blockchain can be called Bitcoin for one particular high-profile example.

Another, and perhaps even more important, conclusion is that the market cap of an appcoin depends crucially on the holding time (H)

. If someone creates a very efficient exchange, which allows users to purchase an appcoin in real time and then immediately use it in the application, then allowing sellers to immediately cash out, then the market cap would drop precipitously. If a currency is stable or prospects are looking optimistic, then this may not matter because users actually see no disadvantage from holding the token instead of holding something else (ie. zero "de-facto fee"), but if prospects start to turn sour then such a well-functioning exchange can acelerate its demise.

You might think that exchanges are inherently inefficient, requiring users to create an account, login, deposit coins, wait for 36 confirmations, trade and logout, but in fact hyper-efficient exchanges are around the corner. Here is a thread discussing designs for fully autonomous synchronous on-chain transactions, which can convert token A into token B, and possibly even then use token B to do something, within a single transaction

. Many other platforms are being developed as well.

What this all serves to show is that relying purely on the medium-of-exchange argument to support a token value, while attractive because of its seeming ability to print money out of thin air, is ultimately quite brittle. Protocol tokens using this model may well be sustained for some time due to irrationality and temporary equilibria where the implicit cost of holding the token is zero, but it is a kind of model which always has an unavoidable risk of collapsing at any time.

So what is the alternative? One simple alternative is the etherdelta approach, where an application simply collects fees in the interface. One common criticism is: but can't someone fork the interface to take out the fees? A counter-retort is: someone can also fork the interface to replace your protocol token with ETH, BTC, DOGE or whatever else users would prefer to use. One can make a more sophisticated argument that this is hard because the "pirate" version would have to compete with the "official" version for network effect, but one can just as easily create an official fee-paying client that refuses to interact with non-fee-paying clients as well; this kind of network effect-based enforcement is similar to how value-added-taxes are typically enforced in Europe and other places. Official-client buyers would not interact with non-official-client sellers, and official-client sellers would not interact with non-official-client buyers, so a large group of users would need to switch to the "pirate" client at the same time to successfully dodge fees. This is not perfectly robust, but it is certainly as good as the approach of creating a new protocol token.

If developers want to front-load revenue to fund initial development, then they can sell a token, with the property that all fees paid are used to buy back some of the token and burn it; this would make the token backed by the future expected value of upcoming fees spent inside the system. One can transform this design into a more direct utility token by requiring users to use the utility token to pay fees, and having the interface use an exchange to automatically purchase tokens if the user does not have tokens already.

The important thing is that for the token to have a stable value, it is highly beneficial for the token supply to have sinks

- places where tokens actually disappear and so the total token quantity decreases over time. This way, there is a more transparent and explicit fee paid by users, instead of the highly variable and difficult to calculate "de-facto fee", and there is also a more transparent and explicit way to figure out what the value of protocol tokens should be.