Token Engineering 101

[PPT]

TE Framework

- Discovery Phase
  - Defining the System Goals
  - Defining the System Requirements
  - Stakeholder Definition and Analysis
  - Definition of Interactions and Value Transfers
  - Metrics Definition and Analysis
  - Causal Relationships and Systems Thinking
  - Identification of Stocks and Flows in the System
- Design Phase
  - State system representation in mathematical equations and variables

Summary

- Idea Validation
  - What problem is this protocol solving?
  - Design Goals Identification
  - Business Goals Validation
  - Technical Feasibility of the Product
- Market Analysis
  - Competitors of Product
  - Neishe strategies
  - User Base
  - Partnerships, integration opportunities
- Mechanism Design (Implementation Theory)

    - Technical Architecture and Product Workflow
        - Dominant Strategies
        - Behavior Mechanics
            - Agent Based Modeling
        - Operational Mechanics
            - DAO, Jury, Appeal Board, Dispute Resolution etc.
        - Game mechanics
            - Non financial incentives (Reputations, Weights, Voting Power, Categorization of Agents etc.)
    - Network growth / adoption strategies
        - TVL growth strategy
        - Will the utility increase with time / adoption?
        - Demand and Growth Drivers
        - Future Development opportunities
        - Holder incentives
        - Early Adopters Incentives
- Economic model
    - Financial model
        - Initial Supply
        - Max Supply
        - Different ways how Tokens enter and exit network
    - Reward Distribution
    - Token Distribution
        - Emission Schedule
        - Vesting Schedule
            - Linear, Growing, Shrinking, Dynamic
    - Fee structure
    - Staking, Lockups
    - Deflationary Model

        - Buybacks and Burns
        - Prohibited on demand token minting
        - More “Ins” than “outs”
    - Inflationary Model
        - Inflation schedule (Initial preset rate, Adjusting(time period)
        - Inflation rate
        - Approximate inflow into circulating supply
- Aspects of Game theory
    - Incentive, Disincentive and Slashing Mechanisms
    - How do incentives of actors of a system align with protocol incentives?
    - Incentivizing desired behaviors
    - Disincentivizing unwanted behaviors
    - Slashing Culprits
    - Incentives to ensure long enough token hold time
    - Slashing upon quick withdrawal
    - Alignment of user behaviors with Protocol Goals through economic incentives and disincentives
- Marketing strategies
- Compliance and regulatory Requirements / challenges

Idea Validation

Design Goals Identification

Requirement Gathering

    - Verbal idea explanation
    - Documentation (if available)

- Similar Protocols review

Questionnaire

- Main use cases and functionality of the token
- Economic goal of the token (price Appreciation, Stability, Inflation hedge etc.)
- Business goal of the project
- Vesting or pre defined token distribution (if any)
- Most similar projects and their markets
- Future utilities of token

Token Utility

- Why do you need a token?
- Who uses a token?
- Why should they use the Tokens?
- Will the utility of the token grow with time?

Economic Model Design

Mechanism Design

What is Mechanism Design?
A useful caricature is to think of mechanism design like inverse game theory. In game theory, we take the game as a given and analyze its outcomes according to players' utilities. In mechanism design we start by defining desirable outcomes and work backwards to create a game that incentivizes players towards those outcomes. Another (similarly caricatured) way of looking at it is to think of game

theory as the positive side and mechanism design as the normative side of the same coin.

Protocol Architecture / Workflow

- ABM
- How do incentives of actors of a system align with protocol incentives?
- How do different pieces of the system interact and connect with each other?
- Rules of Ecosystem
- How will these rules be implemented?

ABM

- A set of agents, their attributes and behaviors.
- A set of agent relationships and methods of interaction: An underlying topology of connectedness defines how and with whom agents interact.
- The agents' environment: Agents interact with their environment in addition to other agents.
- Agents
    - Activities / Attributes / Behaviors of agents
- Topology
    - Rules
    - Link Structure
- Environment
    - Stress test
    - Behaviors emerging from interactions of agents

Game Theory

Incentive / Disincentive Mechanisms

- Alignment of reward mechanism with enhancement of network adoption mechanism (Curve: r′=wg*wt*r, Liquidity Gauge)
- Incentivize users to participate in governance (Curve: 2.5 factor slashing)
- Profit - Sharing
- Incentives to ensure long enough token hold time
  - Amount and time weighted incentives (Curve: t/tmax factor, time weighted voting weight)
  - Slashing upon quick withdrawal
- Assign weight to factors in incentive models to prioritize network growth (Curve: Gauge weight, Gauge type weight)
- Marginal Case Analysis - Sustainability of model in controlled environment constraints

Forecasting and Predictive Analysis

- Statistical Models
- Machine Learning Models
- Data Analytics

Financial Risk Assessment / Management / Optimization

- Complex dynamic systems
- Updating parameters and monitoring

- Monitoring usage of protocol
- Characterizing Types of users
- Simulating theses cases
- Edge case scenarios simulations

Tokenomics DAO

Introductory Content

- All You Need to Know About Tokenomics
  - *Great starting point outlining the basics of tokenomics and overview of the key variables: supply, burning, monetary policy, token distribution and earnings*
- An introduction to token economics (tokenomics)
  - *Similar level of detail to link above, but with a bit more detail on token distribution*
- Tokenomics 101: The Basics of Evaluating Cryptocurrencies
  - *Breakdown of tokenomics through the lens of supply + demand as opposed to the underlying drivers (e.g. burning, monetary policy, etc.), with a high-level evaluation of Convex Finance*
- Economics Design Youtube
  - Lisa JY Tan's channel on tokenomics. She's also authored *Economics and Math of Token Engineering and DeFi : Fundamentals of Token Economics*

Sample Protocol Tokenomics Overviews

- *Each example covers initial tokenomics, governing principles and the associated trade offs made when designing incentives compared to other protocols.*
- NEAR Protocol

- Harmony One (Initial Token Economic Model, Updated Tokenomics, Economic Model)
- Florian's substack
- Ethereum Tokenomics 2021: Impact of Eth2, EIP 1559, and L2 Scaling Solutions on Demand/Supply
    - *Slightly outdated, but helpful illustration of how changes infrastructure changes impact supply/demand dynamics for ETH (or another crypto asset)*

Advanced Content

- https://ahitchhikers.substack.com/ - good deep dive substack on token engineering.
- Chapter 3: Cryptoeconomic Patterns - Value capture and distribution of protocols.

Tokenomics Evaluation Framework

Supply

Key Question: Based on supply alone, will this token hold or increase it's value? Or will that value be inflated away?

- Total Supply
    - How many tokens exist today?
    - How many will ever exist? (eg is there a supply cap)
- Issuance rate
    - Is the issuance rate fixed or variable?
    - If variable, what are the factors that determine (and can influence) issuance rate?
- Allocation/Vesting
    - How was supply initially allocated among investors, community, core team, etc? Are there any group(s) with a significant holding that could drive material selling pressure upon vesting?

- What is the vesting schedule for the largest holders?

Demand

Key Question: Why would someone hold this token?

- <u>ROI</u>
  - Excluding any price appreciation, what is the return generated by simply holding the token? (eg staking)
  - Is there an opportunity to earn additional return by yield farming?
  - Are earnings/fees generated from the protocol distributed back to token holders?
  - Does any "rebasing" take place as the protocol inflates?
    - *Rebasing works similar to a stock split, whereby holding and staking the token enables the holder to receive more, thereby offsetting any impact of inflation (eg % ownership remains constant)*
- <u>Community</u>
  - How active is their Discord/Twitter?
  - Has an ecosystem fund been announced? Grants? Hackathons?
  - How is the protocol actively working to drive additional community engagement?
  - Do one-time + ongoing initiatives drive additional token demand?
- <u>Lockups</u>
  - Is there a lockup program in place?
  - If there is a lockup program in place, what is the incremental value of rewards and what are the requirements to earn those rewards?
  - What % of total tokens outstanding are locked up?
  - How much selling pressure is generated upon lockup expiration (and when)?
  - Are there other non-monetary benefits to staking + locking up tokens? (eg increased voting power)

DAO Specific Content

- [Incentive Design & Tooling for DAOs]
    - *How to match a DAO's goals with the right incentive mechanisms to achieve them...*

Past trends in tokenomics

1. Token staking rewards i.e. yield farming
2. Pool 1 - single token staking pool
3. Pool 2 - liquidity token staking pool
4. Issues
    - Mercenary capital tends to farm and dump tokens with liquidityliquidit4y and token price spike followed by a dump and rotation into other newer projects
    - Bag holders end up blaming the project for normal market movements
    - Fine balancing act between too low APY thus unable to attract new capital and too high APY leading to being called a scam and leading to quicker dumping
    - Can lead to over paying for liquidity and capital

Different token release schedules

1. Linear

a. Easiest to implement
b. No real beneficial rewards for early or late users
c. Capital inflow will dilute rewards preventing further capital inflow

2. Growing

a. Token release increases over time ala Badger
b. Will get lots of hate from community and criticism
c. There is no real advantage to being early

d. Higher rewards later can help sustain higher APYs and thus continuous capital inflow

3. Shrinking

a. Token release shrinks over time

b. Fomo aspect incentivizing early entry into project

c. Lower outflow at the tail end of the spectrum can render the protocol, alongside normal downward price trends, unable to attract new capital

4. Dynamic

a. A dynamic token release schedule can help prevent overpaying for liquidity early on, while providing rewards for early users by reducing rewards later as theres less risk, to match APYs on other projects

b. Can be difficult to advertise and sell users on

c. Few projects implementing this

d. More work and adjustments needed

e. https://www.mechanism.capital/liquidity-targeting/

Current and new trends in tokenomics

1. Protocol owned liquidity

a. Purchase LP tokens off holders instead of renting liquidity via extended farming programs

2. Vote escrowed tokens

a. Users lock tokens for 1 week to 4 years in exchange for a time weighted token. Pioneered by curve. veToken can be used for voting, as holders are expected to vote with long term interest in mind, and can be incentivized with fee sharing or farming rewards boosts

b. Reduce circulating supply to potentially make the token deflationary and help boost price with reduced sell pressure

3. XTokens

a. Similar to veTokens but without a required lockup
b. Tokens can be staked for xTokens e.g. Trader Joe, Sushi
c. xTokens accrue rewards directly with the price of the xToken being worth >1 of the underlying token

4. Revenue or fee sharing

a. Incentivize token locking via vote-escrow schemes in exchange for protocol fee sharing
b. Offer high APY without diluting/inflating free floating token supply
c. Implemented by blizz, geist, qidao, iron.finance

5. Vested farming

a. Instead of releasing 100% of a users farmed tokens on claim, vest the tokens at a linear rate for a time period. This allows advertising higher APYs and reflects some of the tokens back to the treasury for more farming longevity or ve token lockers.
b. This has been implemented by Adamant.finance, blizz.finance, geist.finance, kyber
c. Overall, it doesn't seem like it stops farm and dumping of tokens

6. Options liquidity mining

a. Instead of, or alongside the regular tokens, farmers get a options NFT
b. Options NFT can be redeemed after a period of time,e.g. 1 month, for the token for a pre-set or floor price, allowing the user to option to buy the token at a discount from the protocol
c. If the price of the token drops below the floor price of the option, the option becomes worthless, providing a floor price for the token
d. Concurrently raises money for the protocol and treasury as the user is buying the tokens off the protocol instead of off the market
e. Pioneered by pods.finance and kp3r

Price support trends

1. Buyback and burn

a. Protocol revenue used to buyback tokens from the market and burn them
b. Theory is that reduced supply will help boost investor confidence and price
c. Buyback also provides some price support
d. Overall success of these programs seems minute

2. Buyback and make

a. Protocol fees are used to buyback tokens and redistribute
b. Can be redistributed to token holders thus boosting APY
c. Can be used to incentivize or fund protocol development
d. Provides some price support and investor confidence from the buybacks
e. Can be used to prolong project farming runway in case of capped token supply

3. Burn

a. Tokens are burnt, reducing supply to imply scarcity to boost investor confidence and potentially trigger FOMO
b. Not very effective from observations
c. Can reduce project runway as there's less tokens left to incentivize liquidity

4. Fee distribution

a. Protocol fees are distributed to token holders as mentioned before
b. Gives a direct secondary incentive for holding the protocols tokens and does not lead to users directly dumping the gained tokens thus negatively affecting token price

Miscellaneous trends

1. Reflections

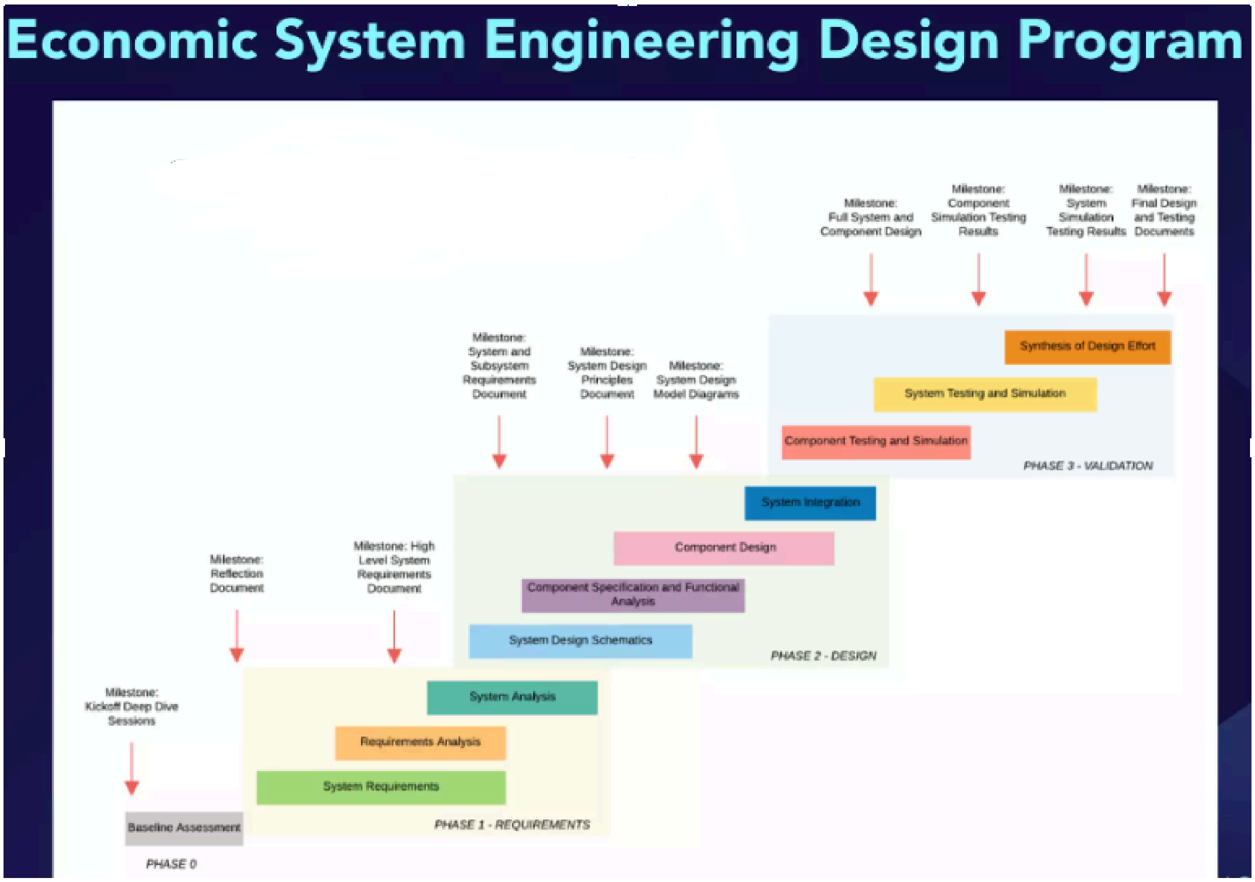
a. Wherein token transfers cause some percent of the tokens to be reflected back to the treasury /token holders / burnt

b. Mainly used by meme coin projects such as Safemoon
c. Fei used this for a while but was met with criticism
d. Transfer tax can cause transfers to fail on low slippage causing the illusions of low liquidity, which is bad
e. Can cause difficulty integrating with other protocols

Token supply trends

2. Rebasing tokens

a. Token balance changes based on rebasing to try and control token value
b. Used by protocols like Frax, Ampleforth, Badger Digg, OHM
c. https://www.mechanism.capital/algorithmic-stablecoins/
d. OHM



Domain Knowledge (Economics, Defi, Finance etc.)

Mechanism Design

https://medium.com/blockchannel/a-crash-course-in-mechanism-design-for-cryptoeconomic-applications-a9f06ab6a976

DeFi Basics

- Stablecoins

    - Algo and crypto backed
    - Asset backed
- Uniswap V3
    - Grained liquidity farming
    - Non forkable
- Sushiswap
    - Vampire Attacks
    - Well executed Uni - V3 fork
- Curve
    - Solving Slippage issue
        - Negative slippage
        - Positive Slippage
    - Stable Swap
    - CPMM
    - Liquidity Mining
        - Earn Trading fee from your liquidity provided (AMM)
        - Extra incentives for liquidity providers
        - CRV Farming
        - Curve Wars (Voting for emissions directed)
- Aave
    - Collateral backed loans
    - Collateral should be bigger than Borrowed asset
    - Borrower Pays interest
    - Use borrowed money for arbitrage opportunities
    - aTokens interest bearing derivatives
        - Interest is paid to aToken Holders
    - Flash Loan
        - Borrow without collateral
        - Used for exploits
- Maker DAO
    - Algorithmic stable Coins

        - UST
    - Crypto Collateralized Stable coins
        - DAI
            - Borrowing Stable Coins
            - 100% overcollateralized
            - Liquidations in auctions
    - Fiat Backed Stable coins
        - USDT, USDC
- Yearn Finance
    - Competitor of Curve
    - Gives Stable coin in yield farming reward instead of CRV in case of Curve
- Trending
    - Game Fi and metaverse
        - When you earn money by playing game it becomes a job not game
    - DAO 2.0
    - Insurances
    - Institutions in DeFi

Foundations of Finance

- **<u>Financial reporting:</u>**
    - Balance sheet = Assets - Liabilities
    - Cash Flow = Receipts - Payments
    - Income Statement / Profit loss Statement = Revenue - Cost

Financial statements = accounts	
Balance sheet (BS)	Assets & liabilities. Assets include **Cash**. Assets – Liabilities = **Equity**
Cash flow (CF)	Cash inflows & outflows. Net inflow + opening balance = closing balance Explains change in **Cash** balance between two balance sheet dates.
Income statement (IS, PL) **Statement of other comprehensive income** (SOCI) & **Statement of changes in equity**	Equity inflows & outflows. Profit + other gains + transactions with shareholders = total increase in equity Collectively explain change in **Equity** between two balance sheet dates.

- Inventory
- Accrued expenses / Accruals
- Interest
- Return
- Compounding interest / Growth

- **Interest only loans:**

File Home Insert Page Layout Formulas Data Review View Developer Help Acrobat

Interest + equal capital repayments loan

Annual interest and repayments

Annual interest rate	10%			
$m	Year 1	Year 2	Year 3	**Total**
Start of year	30	20	10	**30**
Interest charged	3	2	1	**6**
	33	22	11	**36**
Interest paid	(3)	(2)	(1)	**(6)**
Capital repaid	(10)	(10)	(10)	**(30)**
End of year	20	10	-	-
Total payments				
Interest	3	2	1	**6**
Capital	10	10	10	**30**
	13	12	11	**36**

- **Equal capital Repayment Loans:**

Interest + equal capital repayments loan

Annual interest and repayments

Annual interest rate	10%			
$m	Year 1	Year 2	Year 3	**Total**
Start of year	30	20	10	**30**
Interest charged	3	2	1	**6**
	33	22	11	**36**
Interest paid	(3)	(2)	(1)	**(6)**
Capital repaid	(10)	(10)	(10)	**(30)**
End of year	20	10	-	-
Total payments				
Interest	3	2	1	**6**
Capital	10	10	10	**30**
	13	12	11	**36**

-

- <u>**Linking primary Financial statements:**</u>

Balance sheet **At period-end date, say 30 September**	2020	2021(A)	Buy inventory for cash	2021(B)	Recognise accrued expense	2021(C)
Cash and cash equivalents	30	75	(20)	55		55
Inventory	0	0	20	20		20
All other net assets / (liabilities)						0
All other net assets / (liabilities)	45	45		45		45
Net assets	75	120		120		120

Cash flow **For year ended 30 September**	2021(A)	The direct method	2021(B)	The direct method	2021(C)
Receipts from customers	75		75		75
Payments to suppliers	(30)	(20)	(50)		(50)
Net receipts	45		25		25
Opening cash	30		30		30
Closing cash	75		55		55

Income statement **For year ended 30 September**	2021(A)		2021(B)		2021(C)
Revenue	75		75		75
Total net expenses	(30)		(30)		(30)
Profit / (loss) for the period	45		45		45

Cash flow **For year ended 30 September**	2021(A)	The INDIRECT method	2021(B)	The INDIRECT method	2021(C)
Profit from income statement	45		45		45
+ / - differences between profit and net cash flow					
Increase in inventory		(20)	(20)		(20)
			0		0
Net receipts = net increase /(decrease) in cash	45		25		25
Opening cash	30		30		30
Closing cash	75		55		55

Economics

- Nominal Price
  - Current dollar price
  - Price of a good in markets at a given time
- Real Price
  - (CPI_{i-1} / CPI_i) * Nominal Price
  - Constant dollar price
  - Price relative to aggregate measure of prices
  - Price adjusted for inflation
- CPI
  - Change in price of basket of goods purchased by consumers
  - Percentage changes in CPI measures the rate of inflation in economy
- PPI
  - Measures how price at wholesale market changes
  - Percentage changes in PPI measures the cost inflation
  - Predicts the future changes in CPI
- Supply and Demand Curves
  - Equilibrium
  - Changes in market Equilibriums
  - Shifting of Supply and Demand Curves based on Factors (Increase in income, Cheaper Raw material etc.)
- Price Elasticities of Supply and Demand
  - Sensitivity of a variable w.r.t another
  - Percentage change in one variable in response to 1 percent change in another variable
- Price elasticity of demand
  - Percentage change in quantity demanded in response to 1 percent change in price
  - $E_p=(\Delta Q/Q) / (\Delta P/P)$
  - $E_p>1$ Demand is price elastic
  - $E_p<1$ Demand is price inelastic

FIGURE 2.12
(a) INFINITELY ELASTIC DEMAND (b) COMPLETELY INELASTIC DEMAND

(a) For a horizontal demand curve, $\Delta Q/\Delta P$ is infinite. Because a tiny change in price leads to an enormous change in demand, the elasticity of demand is infinite. **(b)** For a vertical demand curve, $\Delta Q/\Delta P$ is zero. Because the quantity demanded is the same no matter what the price, the elasticity of demand is zero.

- Price Elasticity of Supply
  - Percentage change in quantity supplied from one percent change in price
- Indifference Curves / Maps
- Cardinal Utility Function
  - A utility function which provides numerical value of level of satisfaction of a consumer by getting a specific market basket
- Budget line

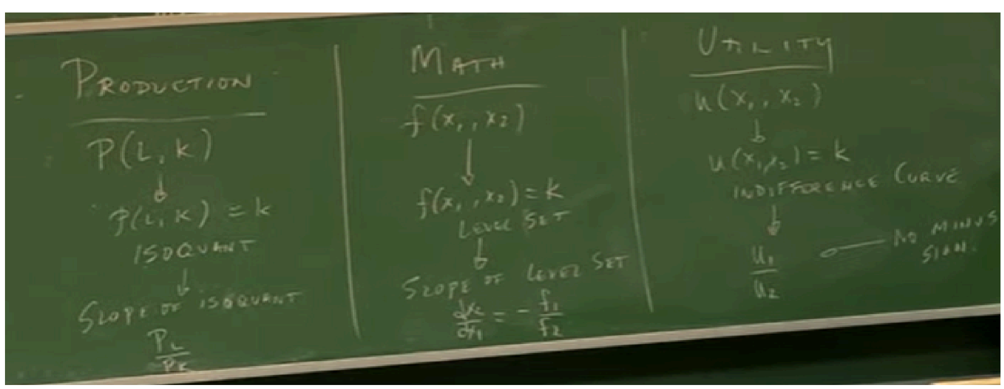
Mathematics

Resources:

Calculus

- Derivatives
  - Partial derivatives (Marginal products)
  - Slope
  - Level sets/curves of a function
  - Cobb Douglas production (utility) function
  - isoQuant
  - Level sets and gradient

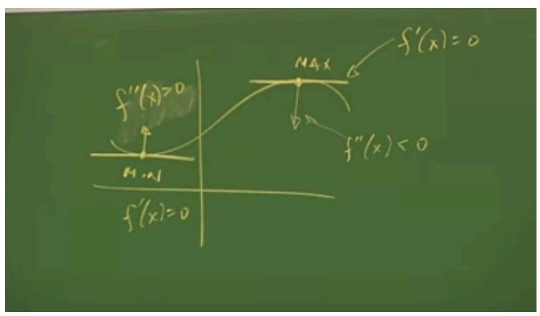
    - Mixed partials / Young's Theorem
    - Hessian and Young's Theorem
    - 1st derivative
        - Slopes of tangent
        - Vf = 0
            - Gives critical point
    - 2nd derivative
        - Concavity
        - V_2f=0
            - Gives Concave up or down / Positive, negative definite
- Differentials
    - $Dy = f' * dx$
    - $Dy = f'1 * dx + f'2 * dx$
    - Slope of level sets = $dx2/dx1 = -f1/f2$
    - Production functions (Cobb Douglas function)
        - $p(L, K) =$ 16$L^{1/4}K^{3/4}$
        - Isoquants = level sets
        - Marginal rate of technical substitution = slope of level sets
    - Utility functions
        - $$U(x1,x2) = X1 * X2$$
        - level sets = Indifferent curves
        - Marginal rate of substitution = slope of level sets



    - Positive Definite Matrices(V_2f=0) Hessian
        - Leading principal minors

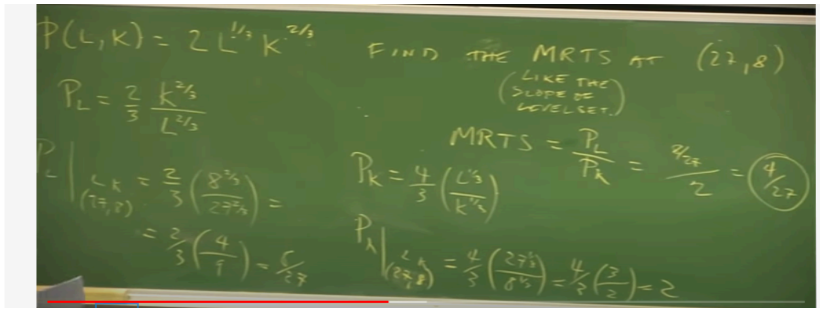
- If all determinants of leading principal minors are positive then its positive definite matrix
- Minima
- $|V_2f| > 0$, $f_{11}>0$
- Eigenvalues > 0

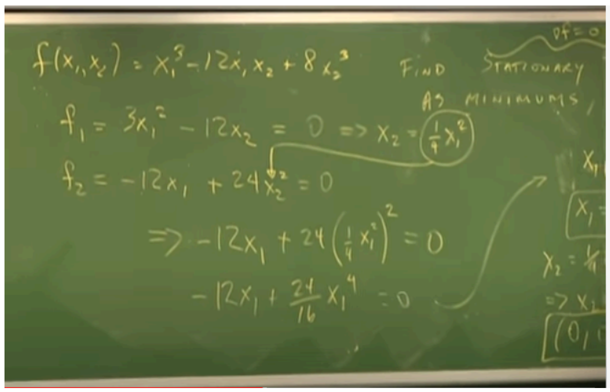
- Negative Definite Matrices ($V_2f=0$)
  - If all determinants of leading principal minors are alternate signs starting with a negative then its negative definite matrix
  - Maxima
  - $|V_2f| < 0$, $f_{11}< 0$
  - Eigenvalues < 0
- Saddle Points($V_2f=0$)
  - Neither positive nor negative definite matrix



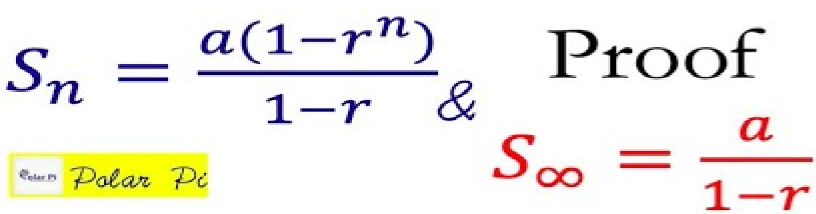
- Extreme Value theorem
  - Constraint is a point (x,y) instead of a function, which is case in lagrange multiplier

- Lagrange Multiplier
  - Vf(x,y) =
  - λVg(x,y)
  - Used To Find max/min of f(x,y) subjected to constraint g(x,y) = k
  - Budgeting for least cost
    - Cost function constrained by budget
    - VC(x,y) = λVB(x,y)
  - Cobb Douglas function
    - $p(L,K) = 2L^{1/3}K^{2/3}$
    - Constraint Budget(L,K) = 60L + 50K
- Find MRTS

- 

- Find Stationary Points


- Price Elasticity of Demand
    - R(P)=P.D(P)
    - Price elasticity= P.D'(P) / D(P)

- Chain Rule
    - Composite function
- Sets
    - Union
    - Intersection
- Geometric Series
    - PDV
    - Discount rate
    - 

Linear Regression Analysis

- **Lines of best fit / Least square estimators**

$$y = \beta_0 + \beta_1 X + \epsilon$$

  - y is the predicted value of the dependent variable (y) for any given value of the independent variable (x).
  - B_0 is the intercept, the predicted value of y when the x is 0.
  - B_1 is the regression coefficient – how much we expect y to change as x increases.
  - x is the independent variable (the variable we expect is influencing y).
  - e is the error of the estimate, or how much variation there is in our estimate of the regression coefficient.

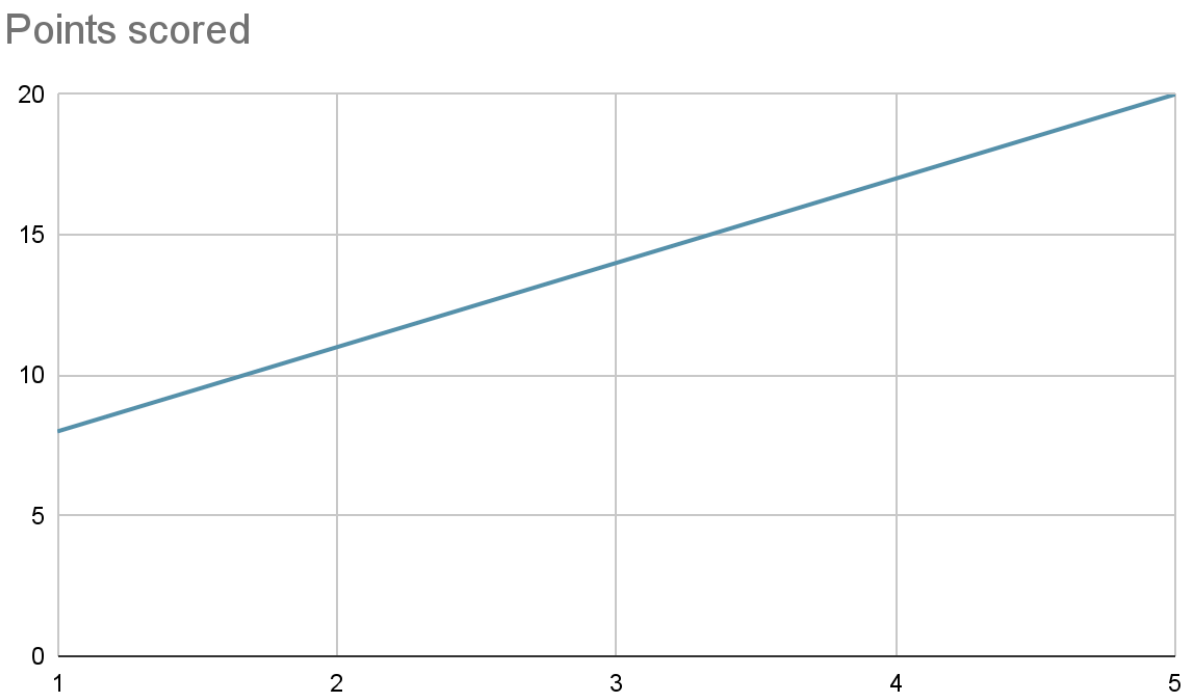
y=mx+c+u

R^2=SSE/SST

SST=$\sum_{1}^{n}$y1-ya

SSE=$\sum_{1}^{n}$y-ya

u= Residuals= y1-y



- **Least Square Estimators**

Number of man-hours and the corresponding productivity (in units) are furnished below. Fit a simple linear regression equation ˆ*Y* = *a* + *bx* applying the method of least squares.

Man-hours	3.6	4.8	7.2	6.9	10.7	6.1	7.9	9.5	5.4
Productivity (in units)	9.3	10.2	11.5	12	18.6	13.2	10.8	22.7	12.7

Solution:

The simple linear regression equation to be fitted for the given data is

ˆ*Y*ˆ = *a* + *bx*

Here, the estimates of *a* and *b* can be calculated using their least squares estimates i.e.,

$$\hat{a} = \bar{y} - \hat{b}\bar{x}$$

$$\hat{a} = \frac{1}{n}\sum_{i=1}^{n} y_i - \hat{b}\frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\hat{b} = \frac{\frac{1}{n}\sum_{i=1}^{n} x_i y_i - (\bar{x} \times \bar{y})}{\frac{1}{n}\sum_{i=1}^{n} x_i^2 - \bar{x}^2}$$

or equivalently $\hat{b} = \dfrac{n\sum_{i=1}^{n} x_i y_i - \left(\sum_{i=1}^{n} x_i \times \sum_{i=1}^{n} y_i\right)}{n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2}$

From the given data, the following calculations are made with *n*=9

Man-hours x_i	Productivity y_i	x_i^2	$x_i y_i$
3.6	9.3	12.96	33.48
4.8	10.2	23.04	48.96
7.2	11.5	51.84	82.8
6.9	12	47.61	82.8
10.7	18.6	114.49	199.02
6.1	13.2	37.21	80.52
7.9	10.8	62.41	85.32
9.5	22.7	90.25	215.65
5.4	12.7	29.16	66.42
$\sum_{i=1}^{9} x_i = 62.1$	$\sum_{i=1}^{9} y_i = 121$	$\sum_{i=1}^{9} x_i^2 = 468.97$	$\sum_{i=1}^{9} x_i y_i = 894.97$

Substituting the column totals in the respective places in the of the estimates *a*ˆ and *b*ˆ , their values can be calculated as follows:

$$\hat{b} = \frac{(9\times 894.97)-(62.1\times 121)}{(9\times 468.97)-(62.1)^2}$$

$$= \frac{8054.73-7514}{4220.73-3856.41}$$

$$= \frac{540.73}{364.32}$$

Thus, $\hat{b}$ = 1.48 .

Now $\hat{a}$ can be calculated using $\hat{b}$ as

$\hat{a}$ = 121/9 – (1.48× 62.1/9)

= 13.40 – 10.21

Hence, $\hat{a}$ = 3.19

Therefore, the required simple linear regression equation fitted to the given data is

$\hat{Y}$ = 3.19 +1.48*x*

It should be noted that the value of *Y* can be estimated using the above fitted equation for the values of *x* in its range *i.e.*, 3.6 to 10.7.

Linear Algebra

- Linear Equations Solution
    - Gauss Elimination Method
    - Gauss Jordan Method
    - LU decomposition Method
    - Jacobi iteration Method

- Gauss Seidel Method
- Substitution Method
- Row reduction - Reduced echelon form of Matrix
- Square matrix
  - x=A^-1 * b
- Augmented Coefficient matrix
- Underdetermined system
  - x=t, Elimination / substitution method
- Homogeneous systems of equations
- Matrices
  - Multiplication
  - Addition
  - Transpose
  - Inverse
  - Order
  - Determinant
  - Adjoint
- Output - Input Analysis using matrices
  - Technology matrix of Leontf
- Cramer's rule
  - Xj=|Aj| / |A|
- Eigenvalue problem
  - | A-λI | $= V$
  - Av = lambda * v
  - | A - lambda*I | = 0
  - A^n = Q * Lambda^n * Q^-1

Mathematical Modeling Basics

- Derivative
- Phase Lines and stability of equilibrium solution
- Directional Fields
- Bounded growth
- Euler’s method

Numerical Analysis

- Interpolation
    - Newton forward / Backward
    - Newton center value
    - Gauss forward / backward
    - Langrange interpolation (Unequal interval) / find polynomial
    - Inverse lagrange formula
    - Newton divided difference formula
- Numerical integration
    - Trapezoidal rule
        - Applicable For any h intervals
    - Simpson ⅓ rule
        - Applicable for Even intervals
    - Simpson ⅜ rule
        - For intervals of multiples of 3
- Solution of ODE by numerical method
    - Picard Method
    - Euler’s method and euler’s modified method (1st and 2nd order ODEs respectively)
    - Runge-kutta method (4th order ODEs)
    - Milne predictor and corrector Method

- Adam Bashford predictor and corrector method
- Curve Fitting (Straight line)
- Least square (Second degree parabola)
- Roots of an Algebraic and transcendental equation using Numerical Methods
  - Bisection Method
  - Regula Falsi Method
  - Iteration method
  - Secant Method / Chord Method
  - Newton Rophson Method

Fourier Analysis

- Fourier Series
- Laplace transform (for CT) / Z-Transform(for DT) is used for design purpose, while fourier series is used for analysis purpose
- Fourier series is an infinite representation of periodic functions in terms of sine and cosine
- https://kilthub.cmu.edu/articles/thesis/Examining_Applications_of_Fourier_Transforms_to_Financial_Data_and_Covariance_Estimation/12824255
- https://www.youtube.com/watch?v=jbn-5JeYVuY&ab_channel=MichaelPollock
- https://www.youtube.com/watch?v=VYpAodcdFfA&ab_channel=InfoQ

  - Periodic Signals
    - Continuous time fourier series
    - Discrete time fourier series
  - Non-periodic Signals
    - Continuous time fourier Transform
    - Discrete time fourier transform
- Fourier Transform
  - Converting Time-Amplitude functions to Amplitude-Frequency functions
  - Moving from time domain to frequency domain
- Laplace Transform
  - Time domain to frequency domain
  - Used to solve complex Differential equations

Statistics and Probability

- Expected Value / Mean
- Variance
- Discrete Random variable
- Probability mass function
- Probability distribution function
- Continuous Random Variable
- Probability density function
- C.D.F
- Binomial Distribution

Product Management

Digital Product Management

What is Digital Product Management?

- Features of DPM
  - Planning and organizing
  - Managing Tasks
  - Budget and control costs
- Components of DPM
  - Customer, competitor, and market research
  - Product roadmap and vision maintenance and development
  - Product and customer analysis
- What is a DPM?
  - Intersection of three components of a product
    - User experience (Desirability)
    - Technology (Feasibility)
    - Business (Viability)

The main difference between digital and non-digital product management is how available data is. Other than that, managing digital products is fundamentally the same as managing non-digital products. Just like non-digital product managers, digital product managers have to:

- Define the product
- Understand the customer's needs
- Describe and analyze the product strategy
- Explain the market strategy
- Guide the engineering of the product
- Direct marketing and sales tactics
- Compile and update requirements
- Manage the product roadmap and effectively communicate any changes
- Represent the product's point of contact

What Are a Digital Product Manager's Responsibilities and Job Description?

- The specific responsibilities of a digital product manager will differ in each company. But you can expect to find at least the following responsibilities on their digital product manager job description.
- Lead the development of a digital product (or a suite of such products).
- Analyze the market to make sure the product continues to offer advantages over its competitors.
- Understand the user persona and buyer persona and develop unique value propositions for both.
- Monitor, analyze, and act on key product analytics. Continually improve the product, increase retention rate, and boost customer lifetime value (LTV).
- Create and prioritize the strategic product roadmap.

Skills digital product managers need

- Rapid iteration and deployment
  - Digital product managers today have little choice but to embrace agile principles. These principles favor delivering working products to users frequently. This means digital products must develop the ability to rank small updates. Because the ranking ability is important to push updates out to customers on shorter timescales.
- Design thinking
  - The user experience is an essential component of a digital product. The best software or mobile app will fail if its interface and user experience is frustrating or confusing. A digital product manager must learn how to create an intuitive product experience.
- Understanding usage data
  - Digital product managers have an advantage over product managers who oversee physical products. They can track how people use their products and learn what's working and what isn't. Digital product managers must become skilled at reading and interpreting these product analytics. Because they must leverage these insights into the ability to improve their products.

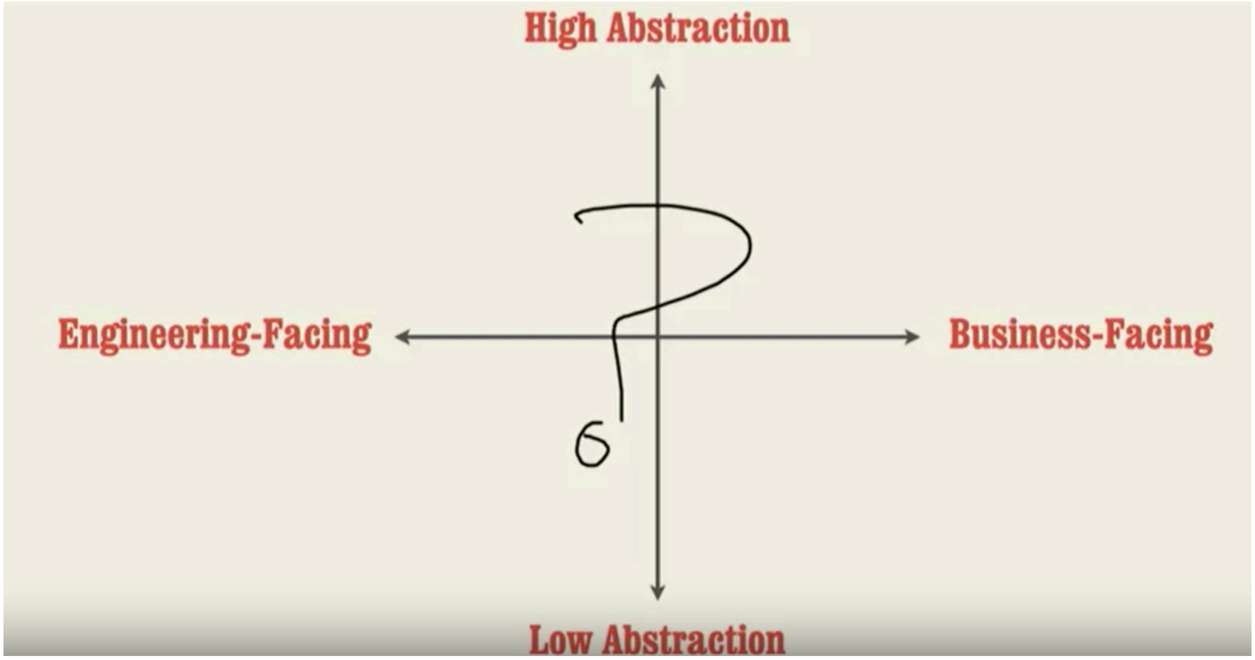
Process of Product management

- Waterfall Method
  - Discrete Units who are supposed to plan and manage with high degree of precision (Assembly Line concept
  - Applicable in organizations that are organized by departments and specialization
  - For physical and mechanical services where the world is predictable and markets are less volatile

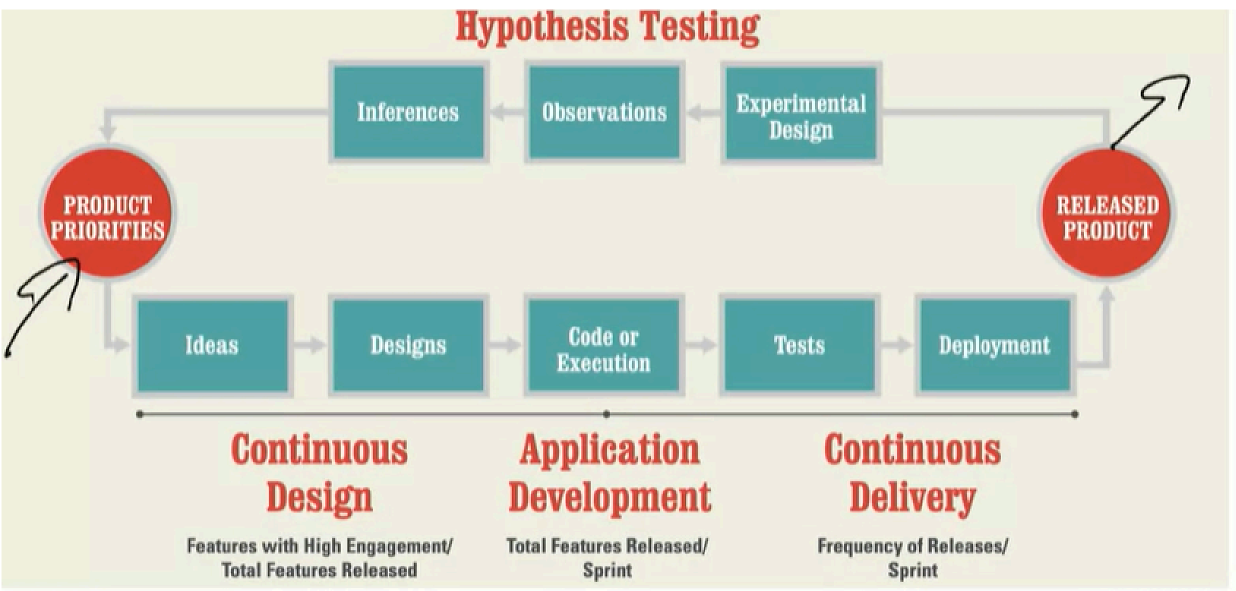
- Agile Method
    - For innovation and Tech centered projects
- Continuous Design
    - Ideas
    - Designs
- Application Development
    - Execution
    - Code
- Continuous Delivery
    - Test and Deployment
    - Cost to release a successful feature = (Cost to build a feature) / (Portion of all release content that is successful)
    - Cost to build feature = Team cost / Release content
- Hypothesis Testing
    - Product / Market Fit
    - Managing a Product Pipeline
    - Design Team
    - Development Team
        - JTBD
        - Agile User Stories
        - Prototypes
        - App Analytics
        - Usability testing
        - DevOps
    - Product Manager vs product owner
        - Focus: Market facing vs Dev / Eng facing
        - Creates: Focus, direction, roadmap vs story backlog
        - Review: Features vs Execution of user stories
        - Internally facing vs externally facing
    - Analytics and Data science Team
        - Dependent and independent variable correlation

        - Forecasting through predictive models
        - Framing dependent variable
        - How results are changing the behaviors
    - Consulting and support team
        - Physical interaction and good relationship with the support team
    - Sales and Marketing Team
        - Business model canvas
        - Getting to better ideas
    - Finance Team
        - Managerial and financial
    - Legal Team
    - Management Team
        - Understanding management's overall organizational goals
- Primary focus of a product manager
    - Identifying product market fit hypothesis
    - Testing it with interdisciplinary team
    - Facilitating Alignment with outside stakeholders
- User Journey
    - Trigger
    - Action
    - Variable Reward
    - Investment
- HOOK Methodology
    - Managing habits
    - Internal and external triggers
    - Simplest behavior of user after getting triggered
    - Variable and unpredictable reward
    - Investment of user in product which brings him back to product
- Dealing with Enterprise Customers
    - Anchoring Problems not solutions
    - Writing a full narrated story

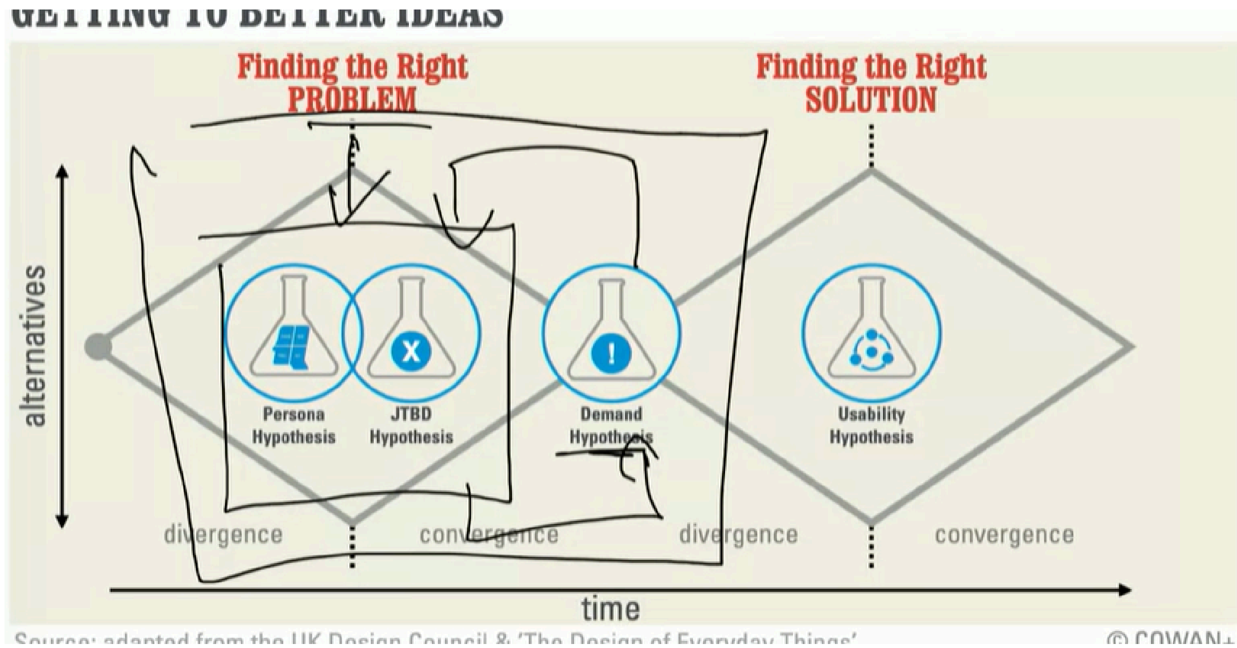
    - Work in prioritized outcome based batches
    - Test often and appropriately
    - Over invest in training
    - Over Invest in onboarding
    - Design for variation
    - If it's a 'No' say 'No'
- Product Death Cycle
    - Instead of focusing on features (which are easily quantifiable) focus on value those features/product is delivering to user
    - Focus on outcomes and think beyond features
- Storyboarding
    - AIDA Funnel
        - Awareness
        - Interest (Consideration)
        - Desire (Consideration)
        - Action (Conversion)
    - The Customer Funnel
- Managing a product with a hardware
    - Functional Decomposition
- Product Manager Spectrum



- Agile Methodology of Product Pipeline



- Find the right Problem
    - Persona hypothesis
    - JTBD hypothesis
    - Demand hypothesis
    - Identify DVs and IVs

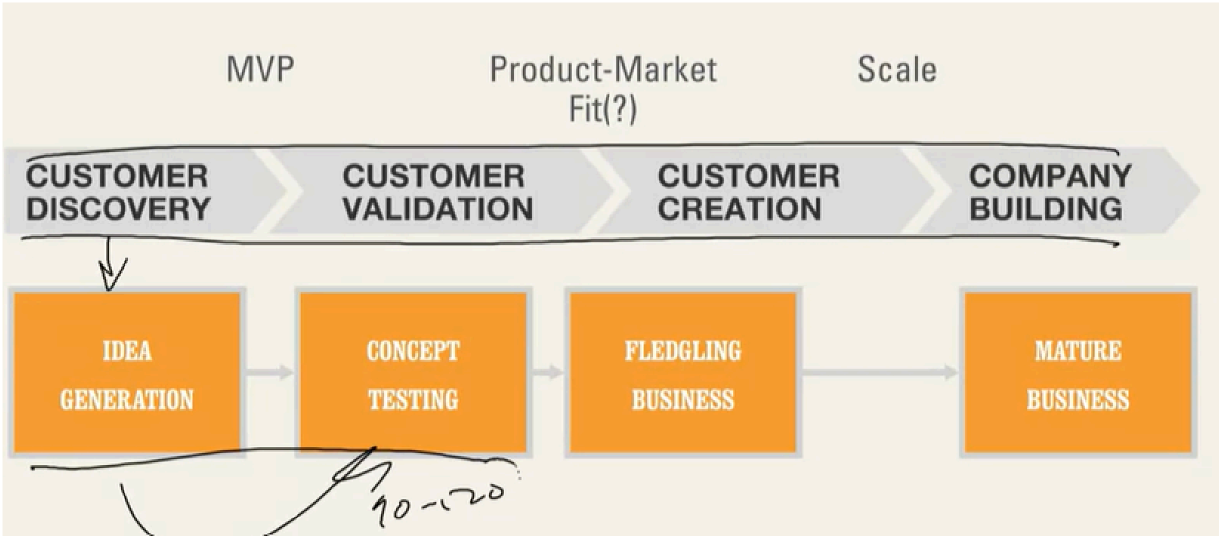


- Find the right solution
  - Usability hypothesis
  - App analytics
- Lean Startup Method
  - The lean startup methodology is a method of managing and building a business or startup by experimenting, testing, and iterating while developing products based on findings from your tests and feedback.
  - This method of business management and product development is designed to deliver products to customers at a quicker pace (often tested in the form of an MVPby focusing on product features that have been validated through explicit customer feedback at various stages in the product development cycle.
  - Entrepreneur Eric Ries first introduced this methodology in a book that he authored in 2008, which is titled "How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Business".
  - The goal of using this methodology is to get rid of wasteful practices during the initial stages of a company, which provides the company with a higher possibility of long-term success. By using the lean startup methodology, it's possible for early-stage startups to garner success without requiring high

amounts of funding, comprehensive business plans, or a product that has no flaws.
    - For the lean startup process to be successful, the startup that uses it will need to focus on obtaining customer feedback on the initial product. This feedback will help you make changes and iterations on the product that will allow it to be improved over time in accordance with the wishes of the customers. The customer feedback that you receive should also keep you from spending resources on services and features that your customers don't want. The lean startup methodology is aimed at using the lowest amount of resources possible.
- Developing new Products - Progress
    - Customer Discovery (MVP)
    - Customer Validation (MVP)
    - Customer Creation (Product Market Fit)
    - Company Building (Scale)
- Learning Vs Scaling
    - The Product Hypothesis
        - Persona
        - Problems
        - Alternatives
        - Value proposition
- Types of MVP
    - Concierge MVP
    - Sales MVP
    - Wizard of oz MVP
- Horizons of growth
    - H1
        - Strategize and optimize existing business model and tech
    - H2
        - Either new business model or tech
    - H3

- New Business model and new tech

- Corporate innovation pipeline



- Idea generation
  - Subject Interviews
  - Reverse Hackathons
  - Customer Comments
  - Consulting and support
  - Open innovation
- Concept Testing
  - Talent hunt
- Fledgling Business
- Mature Business
- Cross functional teams are more suitable for innovation as you have expertise in multiple disciplines in your team
  - Daily Standups
  - Weekly Plannings
  - Qualitative and quantitative data analytics
  - Believe in product
- Business Model Design
  - Preposition Design

- Identifying Business Model Types
    - Infrastructure Driven
    - Scope Driven
    - Product Driven
- Strategic Foresight - Context Tool
    - Channel
    - Content
    - Objective
    - Flexibility
    - Methodology
    - People
    - Cost
    - Career, Skills
- Actionable Analytics
    - Analytics Objective
    - Understand Problem
    - Diagnose Data
    - Diagnose resources and priorities
    - Prototype
    - Communicate and observe
    - Action
- Monolith Architecture Vs Microservices

Software Architecture

Technical Specification Document

What are the goals of a technical specification document?

A technical specification document, in the case of a software project also referred to as a "software design document", is aimed to provide an entry point for externals. On a high level, its goal is to understand the goals of the software and guide a potential software vendor on how the project is expected to be managed for both the customer as well as the vendor. The technical spec should help with:

Defining deliverables: What features are expected and in which way should they be delivered? The deliverables are split up into milestones, if the project is larger. These milestones usually contain an isolated aspect of the project and aim at being usable by themselves.

Environment: How does the software operate in daily business? How does it fit into the existing IT infrastructure?

Ways of communication: Who is managing this project on the customer end as well as on the end of the vendor? Key people such as project managers are usually set early on and stay on throughout the whole project.

Clarify the details of collaboration: This includes expected project updates and meetings. This ensures both sides have a clear understanding of how the project should be executed. Generally, it is wise to review the software build regularly while it's getting developed.

In the following we will go deeper into the topic by structuring these points more.

How to write a good project specification for your software project

A good software specification doesn't just describe the features in detail, it guides the reader in learning about the core ideas in all major aspects.

Introduction

As with many documents, you want to introduce the reader slowly to the story. The following two points will help your reader get started on the project:

Title page: Share the current working name of the project, if already defined. Don't forget to include you as the author with contact details in case of questions. If you've got your project team already assembled, you should mention it as well. Often software engineers know each other from meetups or other social events. This can help to make the communication more smooth.

Brief Summary: Share the high-level purpose of the project/product you are planning to build. This shouldn't be longer than 5 lines of text. Skip details about features and concentrate on the high-level picture here.

Feature overview and environment

Diving a bit deeper, the idea takes shape, the users and the environment get described further:

Core Features: What should the software "do"? Describe the functionality grouped into sets without going into the nitty gritty part yet. Share which business processes/functions should be managed/automated with the software. If your project is a website, include the expected hierarchy of pages. Table overview works well for any non-website project. Keep it brief at this point.

Users and roles: Who is using the software at the end and what are they trying to achieve? Describe the user groups with their background. The individual experience level of the users gives the software engineers insights too: A software for accountants requires a different level of detail in the user interface than a generic contact form on a website.

Environment: How should it be accessed? Is this a desktop software or a web application? What technology (e.g. operating system and browsers - including versions) should be supported? For web-applications and websites, information about the browsers are important. Version information such as "Firefox: current version minus two" are commonly understood. Project scope and costs depend on your required level of platform support. Keep it as low as possible, but as high as needed to make it work for your organisation.

Constraints: This is the moment to mention your corporate identity (CI), style guide, etc. You can append these documents in your emails or share links to relevant sources here.

Detail features

This is the moment to dive deep into the features. Similar to the previous high-level features, write a section for each core functionality. Describe the expected workflow. Mention the information entered, load/saved and displayed.

Graphics, even freehand sketches, can help with understanding of the goals. A designer can help bring sketches of user interfaces into order.

User stories are a good way to clarify workflow with more details. These are theoretical scenarios describing how a feature would be used by one of the end users. Include all steps such as input, clicks and similar actions needed to complete the user journey. Describe the expected outcome as well.

Don’t forget to include information about the required permissions: Usually not all parts of a software product are accessible by every user. Make sure to reference the user group from the previous section here. Alternatively, a permission table can make sure nothing is forgotten.
Depending on the complexity of the feature, each of the sections should take up between a half page to two pages at maximum. If you exceed two pages you should consider splitting the feature further or reducing details and adding these to a general section.

External interfaces

Software doesn’t exist in a void anymore. Often software interacts with other software. This can be through interfaces (APIs) to services or external databases. Examples could be email services or services providing real-time data such as stock prices. Describe any systems which your project needs to integrate with. Share any details such as links to documentation.

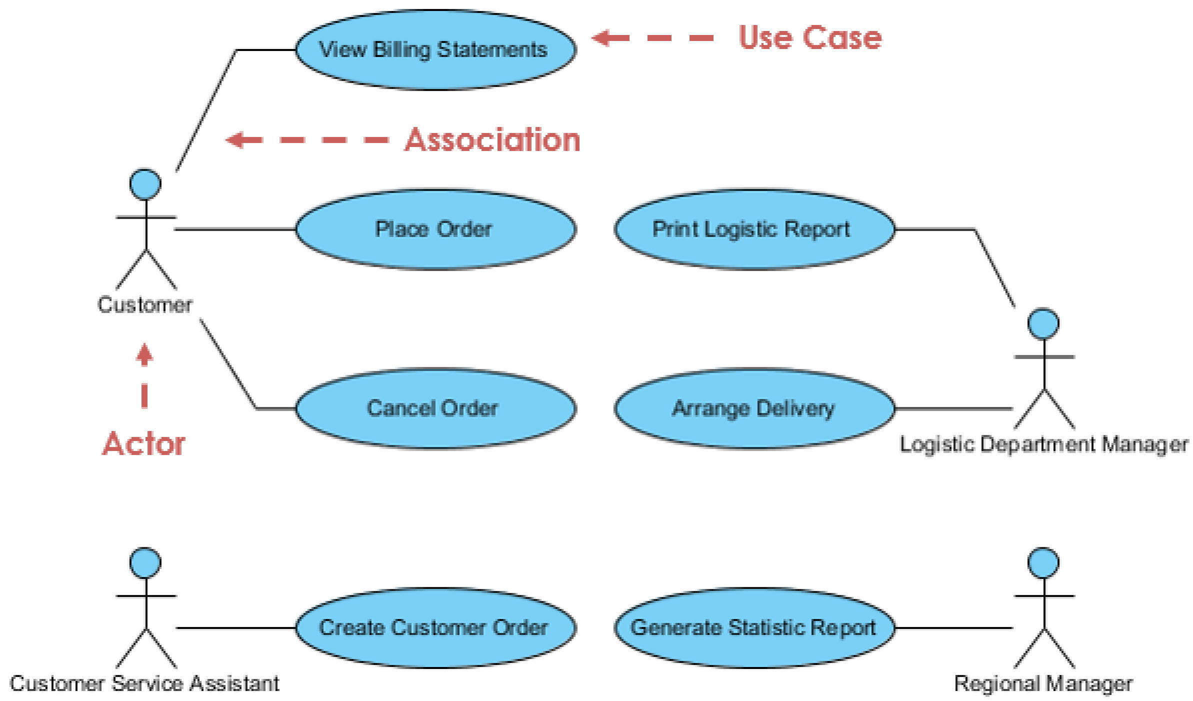
UML

What is a Use Case Diagram?

A use-case model describes a system's functional requirements in terms of use cases. It is a model of the system's intended functionality (use cases) and its environment (actors). Use cases enable you to relate what you need from a system to how the system delivers on those needs.

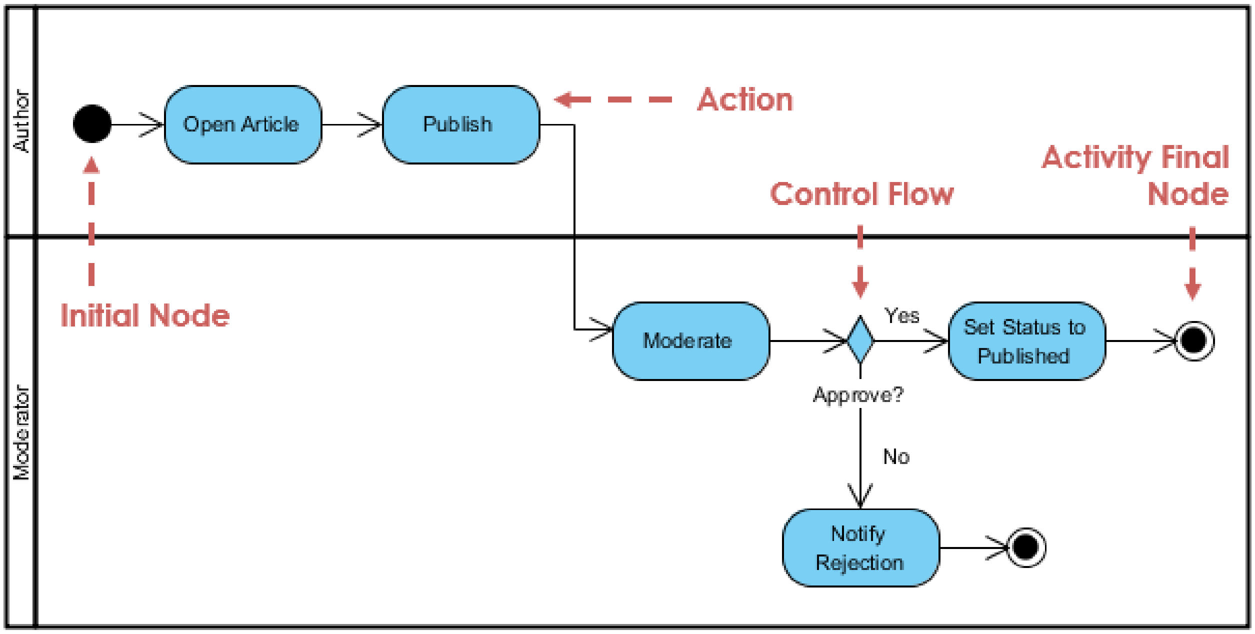
Think of a use-case model as a menu, much like the menu you'd find in a restaurant. By looking at the menu, you know what's available to you, the individual dishes as well as their prices. You also know what kind of cuisine the restaurant serves: Italian, Mexican, Chinese, and so on. By looking at the menu, you get an overall impression of the dining experience that awaits you in that restaurant. The menu, in effect, "models" the restaurant's behavior.

Because it is a very powerful planning instrument, the use-case model is generally used in all phases of the development cycle by all team members.



What is an Activity Diagram?

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. It describes the flow of control of the target system, such as the exploring complex business rules and operations, describing the use case also the business process. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e. workflows).



What is a State Machine Diagram?

A state diagram is a type of diagram used in UML to describe the behavior of systems which is based on the concept of state diagrams by David Harel. State diagrams depict the permitted states and transitions as well as the events that effect these transitions. It helps to visualize the entire lifecycle of objects and thus help to provide a better understanding of state-based systems.

Software Design and Digital Project Management

System Analysis and Design

Architectural Design

- Architectural Analysis and Design
    - Layers and partitioning

- Decisions
- Patterns
  - Multi layer
  - Repository
  - Client Server
  - Broker
  - Transaction processing
  - Pipe and filter
  - MVC

Class Design

- Use case analysis
  - Boundary Classes
  - Entity classes
  - Control classes
  - Cohesion and coupling

State machine Diagrams

- Events
- States
- Transitions
- Actions
- Composite State Machine Diagrams
- Sequential State machine Diagrams
- Objects
- Classes
- Unavailable States
- Triggers

Design Patterns

- General reusable solution to a commonly occurring problem

Software Architectural Patterns for Big Data and Blockchain

Principles of Software Design

- Assume Simplicity
- Quality work
- Incremental changes
- Rapid feedback
- Honest and open communication
- Embracing changes

Testing

- Test Doubles
  - Stub
  - Spy
  - Fake
  - Mock
- Dependency injection
- Inversion of control
  - Constructor injection
  - Setter injection
  - Interface Injection

Systems Engineering

- Requirements Definition and Management
- Systems Architecture Development
- System/Subsystem Design
- Verification/Validation
- Risk Management
- Systems Integration and Interfaces
- Life Cycle Support
- Deployment and Post Deployment
- System and Program Management

Modeling

- Functional Modeling
  - Activity Diagrams
- Behavioral Modeling
  - State Machine Diagrams
  - Use Case Diagrams
- Structural Modeling
  - Block Diagrams
  - Class Diagrams

MBSE stands for Model-Based Systems Engineering. It is an approach to systems engineering that uses models as the primary means of representing, analyzing, and designing systems. The MBSE analysis framework is a structured method for applying MBSE to the development of complex systems.

The MBSE analysis framework includes several key components:

System Requirements: The first step in the MBSE analysis framework is to define the system requirements. This involves identifying the objectives of the system, its constraints, and any performance criteria that must be met.

System Architecture: The next step is to develop a system architecture that describes the overall structure of the system and how its components interact. This includes identifying the subsystems, interfaces, and other key features of the system.

Functional Analysis: The functional analysis stage involves breaking down the system into its functional components and analyzing how these components interact to achieve the system objectives. This may involve the use of block diagrams, flowcharts, and other modeling techniques.

Behavioral Analysis: The behavioral analysis stage involves analyzing the system's dynamic behavior, including how it responds to inputs and how it operates over time. This may involve the use of simulation models, state-transition diagrams, and other techniques.

Verification and Validation: The final stage of the MBSE analysis framework is to verify and validate the system design to ensure that it meets the system requirements. This may involve the use of testing, simulation, and other methods to evaluate the system's performance.

Overall, the MBSE analysis framework provides a structured approach to the development of complex systems. By using models as the primary means of representing and analyzing the system, it enables engineers to more effectively manage the complexity of modern systems and to identify potential issues early in the design process.

cadCAD

System Modelling Techniques

- System Dynamics (Machinations)
- ABM (cadCAD)
- Networked Models
- Multiscale Modeling

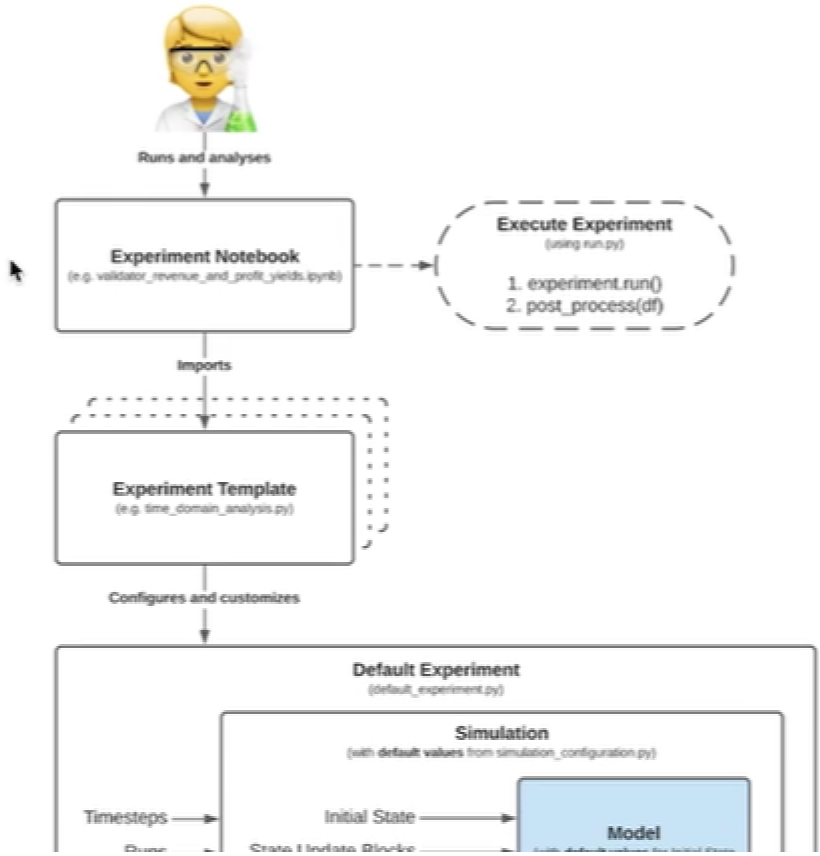
Process

1. System goals identification
2. Control parameter identification
3. Environmental parameter identification
4. Metric identification
5. Simulation
6. Optimal parameter selection

Steps

- Modeling
    - State variables
        - Active_validators
        - Ethe_price
        - Revenue_yields
        - Profit_yields
        - Validator_rewrads
        - Validator_penalities
        - Validator_slashing
        - Operational_costs
        - mev_realized

    - System Parameters
        - Used in policy functions to update the state of system
        - Validaor_environments
            - Hardware
            - Cloud, etc.
        - Time_parameters
    - Stochastic processes
        - Uses system_parametrs module to config time dependent parameters
    - Policy Functions
    - State Update Functions
        - Updates the state of system, environment and individual actor
    - Partial State Update Blocks
- Simulation
    - Configuration
    - Execution
    - Output Preparation
    - Analysis
- Experiments



- Templates module
  - Reusable simulation configurations for time domain analysis, monte carlo analysis and other type of analysis
  - Get imported by experiment notebooks
  - Time_domain analysis template
    -
- Default experiment module
  - All experiment templates are based on this module
  - Experiment template chosen will configure itself using default experiment module
- Simulation config module
  - Adaptable parameters of simulation like monte carlo runs, delta time, timesteps, etc.

Parameter Selection Under Uncertainty in Detail

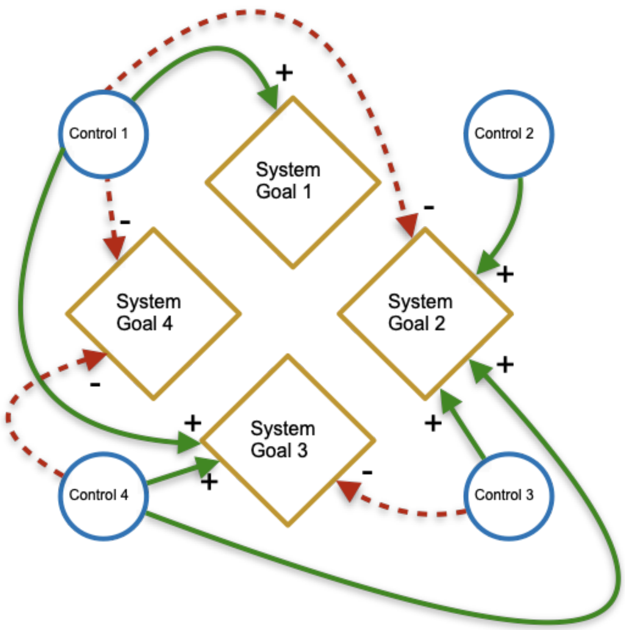
Step 1: System Goals Identification

This steps connects the qualitative definitions of targets articulated by the team to concrete and precise system goals. It helps to attach quantitative measures to the qualitative statements. It is key to ensure that simulation parameters and metrics are aligned with the overall objective.

Step 2: Control Parameters Identification

In this step, parameters are identified that are under control by the system, and impact the outcome of system goals. They can be proposed by the project team — if there is an idea what the controller is (e.g. fee rate, minting rate) — with the intention to set the ideal parameter. More generally either controllers can be unspecified or their impact might be unknown.

A key point is that trade-offs between system goals will occur and the more complex the system the more difficult it is to optimize for all goals. However there are ranges of parameters that will favor some goals simultaneously to a different extent. Therefore a parameter-goal impact assessment is advisable to allow stakeholders to make a prioritization of the system goals when impacts and consequences are sufficiently understood. This can be compared with the simulation results to arrive at a ranking of various parameter selections.



Dynamic system goals can be balanced through feedback (both positive or negative) from embedded controller mechanisms.

Step 3: Environmental Parameters Identification

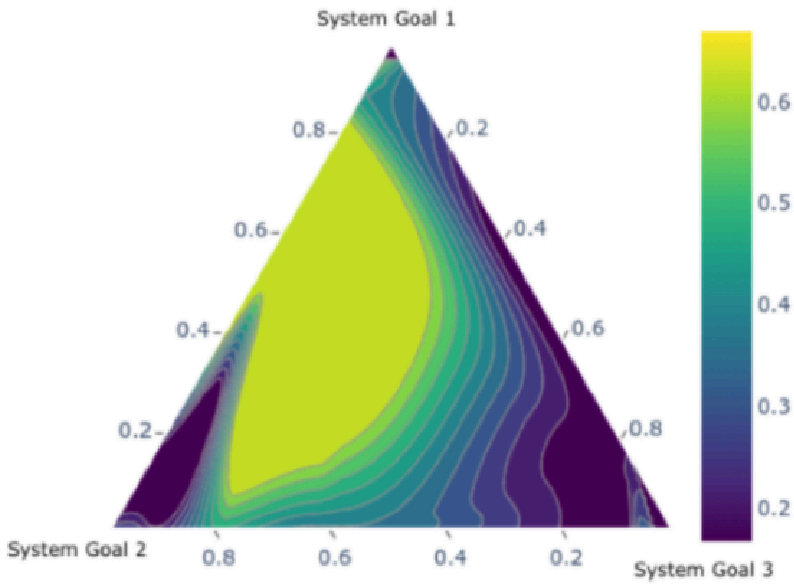
In this step all processes are recorded that cannot be controlled but are important so far as they do affect the system goals. Those systemic events and their effects can then be targeted by available control mechanisms even if they are taken as variable from the environment. For example, the price of ETH may factor into a model, but it is not under control of the model.

Step 4: Metrics Definition

In this step qualitative goals are translated into quantifiable metrics as closely as possible. System goals and metrics are not exclusive — there might be more KPIs for one system goal or conversely several system goals might share the same metrics.

Metrics can be of different types as they can measure different system characteristics e.g. stability, liquidity, responsiveness. In any case, metrics allow to not only optimize for system targets but also help to identify thresholds beyond which the system is unresponsive or unstable.

In the subsequent selection process, the control parameters are selected given the realizations of the environmental parameters such that defined metrics are fulfilled. This makes the metrics to be the criteria by which the control parameters are chosen.



Exploring tradeoffs between various system goals requires data and visualizations to make discoveries actionable for stakeholders.

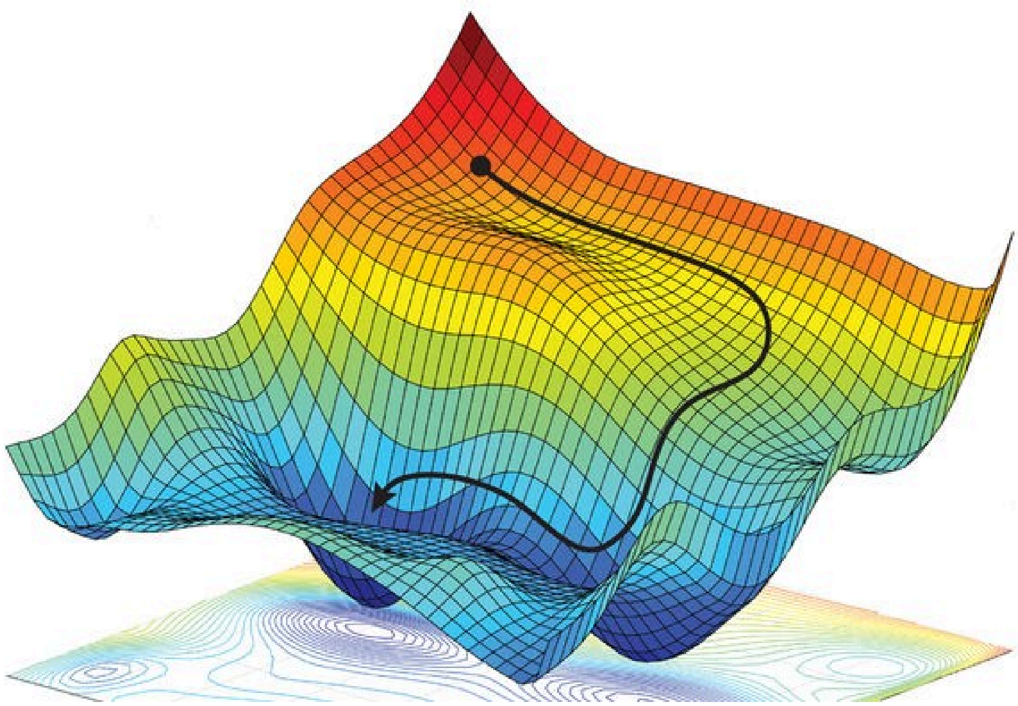
Step 5: Simulation

In this step scenarios are defined that are to be tested against the system goals. All uncontrolled environmental parameters are represented by generated processes. For controllable parameters, ranges for value sweeps are defined so that simulation experiments can be performed in multiple runs.

Afterwards, the robustness of the parameters and their impacts on the system are tested through a sensitivity analysis.

Step 6: Optimal Parameter Selection

The optimal values can be selected by assessment of the previously stated metrics. Based on basins of attraction and confidence intervals, choice recommendations can be done. A ranking allows to order optimal parameters according to the prioritized list of system goals.



Machine Learning (CS-229)

Supervised Learning

- Training set
- Learning algorithm
- Input —>hypothesis—----->output

- Linear Regression
    - (x^i,y^i) - one training sample
    - x^i is a n+1 dimensional vector with two features and x_0 as fake feature and x_1
    - y_pred=h_theta(x) = SUM_j=0 to n (Theta_j *x_j) = TEHTA^T * X
    - Cost function (MSE) = SUM i to n (y_true - y_pred)^2 = J(THETA) = ½ (SUM_i =1 to n (h_theta(x)^i - y^i)^2)) minimize this cost function to find parameter theta
    - To find the value of theta which minimizes the cost function (error) J(theta)
    - h(x) = theta_0 + theta_1 * x_1 + theta_2 * x_2
    - Theta = weights (parameters - algorithm defines parameters to predict)
    - X = inputs / features
    - h(x_j) = SUM (theta_j *x _j)
    - (x,y) = training example
    - To evaluate h at certain value of x: Fit theta to minimize cost function (MRSE) and return Theta_x
    - Cost function = J(theta) = (h(x_i) - y_i)^2 ==0 (choose values of theta which minimize difference and almost equal to zero)
    - Theta_j = theta_j - learning rate * Partial derivative of J(theta) w.r.t theta_j
    - Partial derivative of J(theta) = partial derivative of 1/2(h_theta(x) - y)^2
    - = (h(x) - y) * partial derivative(theta_0 + theta_1 *x_1 ....theta_n*x_n)
    - =(h(x) -y)*x_j
    - Theta_j = theta_j - learning rate * sum (h(x)_i -y_i) * x_j_i
    - Repeat until convergence
    - Cost function can be written in terms of Matrix = derivative J(theta) w.r.t. theta=½ derivative w.r.t THETA (X.THETA - Y)^T * (X. THETA -Y)
    - By simplifying we get THETA = (X^t X^-1) X^t * Y
    - By converting cost function in matrix form we can get matrix THETA using above formula where MATRIX operations are easier to implement
- Gradient Descent (Optimization algorithm)

  - Taking derivative of cost function for every feature theta and every value of each feature in data test and repeat this until it converges to local minima
- Stochastic gradient Descent (Optimization algorithm)
  - Instead of using every value of each feature theta just take one random value from dataset in each iteration
- Batch gradient descent (Optimization algorithm)
- Locally weighted regression
  - Fit theta to minimize modified cost function which has weighting function w_i which tells model that values of x_i which are closer to x_predicted in training data to focus more on them (x_i which are far from x_predicted multiplied by zero to nullify / decrease their impact on trained model). It tells amount of attention a value from dataset is given
  - W_i = exp(- (x_i - x_predicted)^2 / tau^2)
  - Paying attention to the points which are closer to x_predicted and ignoring others
  - Tau - Gaussian parameter which defines the width of gaussian wight function
  - Parametric learning algorithm - Linear regression (Fit fixed set of parameters to data)
  - Non parametric learning algorithm - Locally weighted regression
- Probabilistic Interpolation (Why we use least square error as cost/objective function) MLE (Maximum likelihood estimation is used to determine the cost function. MLE gives probabilities distribution (gaussian) to find cost function for linear regresission). MLE is used to prove that why a certain cost function should be selected by proving that it has max likelihood
  - y^i = Theta^T * X^i + error^i
  - Probability vs likelihood - For variable dataset with constant parameters is probability and with constant dataset variable parameters have a likelihood
  - Probability of y with x and theta is gaussian error function with theta^t*X as mean sigma^2 as variance

    - By using minimum likelihood estimation we can conclude that least squares is suitable choice to estimate error/cost function
- Logistic Regression
    - Used for classification problems
- Newton's Method (Can be used as alternative to gradient dissect as optimization algorithm to get value of theta which minimizes our cost function MSE for linear regression)
    - Find theta such that f(theta)=0
    - Derivative of function is zero at maxima and minima
    - Input theta_0 in function f and find tangent of function at point theta_0, the point where this tangent line crosses x axis is out next point theta_1
    - Theta_t+1 = theta_t -f(theta_t) / f'(theta_t)
- **Optimization algorithms:**
- These are methods used to find the optimal values of a cost function. Examples include gradient descent and Expectation-Maximization (EM).
- Learning principles:
- These are principles used to derive a cost function for a specific problem and model. Maximum Likelihood Estimation (MLE) is an example of a learning principle.
- **Cost functions:**
- Also known as loss functions or objective functions, cost functions measure the discrepancy between the predicted values and the true values. Examples include mean squared error (MSE) and cross-entropy loss.
- **Model types:**
- These are specific statistical models that describe the relationship between input features and target variables. Examples include Gaussian distribution, Gaussian Mixture Model (GMM), and linear regression, GDA, SVM, Logistic regressin.
- Now, let's see how these concepts fit together in a machine learning process:
- Choose a model type that is appropriate for the problem at hand (e.g., linear regression for a regression problem or GMM for a clustering problem).
- Use a learning principle like MLE to derive the cost function that should be minimized based on the likelihood of the observed data given the chosen model.

- Apply an optimization algorithm (e.g., gradient descent or EM) to minimize the cost function and find the optimal model parameters.
- Once the optimal model parameters are found, they can be used to make predictions for new input data points based on the specific model type. For example, in linear regression, the optimal parameters (weights and bias) can be used to make predictions by computing a linear combination of the input features.
- Loss functions play a crucial role in this process, as they guide the optimization algorithm towards finding the best model parameters that minimize the discrepancy between the true and predicted values. The choice of the loss function depends on the learning principle (e.g., MLE) and the specific model type.
- Assumption P(y^i/x^i ;theta)
- Error distribution is gaussian with theta^T * x as mean and sigma^2 as variance and y^ is predicted variable
- Least square error function which is cost function in linear regression with gaussian distribution can be estimated using MLE
- H_theta (x) = g(theta^T *x) = 1/(1+e^-theta^T*x) = sigmoid function used for logistic regression for classification problems
- P(y=1 /x ;theta) =h_tehta(x)
- P(y=0 | x; theta) = 1- h_theta(x)
- = (h(x))^y (1-(h(x))^1-y
- L(theta) = p(y|x;tehta)= Product from i=1 to n ((h(x^i))^y^i * (1-(h(x^i))^1-y^i)
- P(x | theta) = h(x) * exp (eta (theta^T (x) - A(theta)
- Use MLE to find find error function L(theta)
- l(theta) = m * log 1/sqrt(2*pie)*sigma + SUM i=1 to n -(y^i - theta_tau (x)^-1)/2 * sigma^2
- As y^pred is distributed gaussian therefore we will use mean value of y^pred = mean =theta_tau (x)^-1 and divide by variance of gaussian distribution of y^pred
- Logistic regression using log likelihood
    - l(tehta) = log L(theta)= log PRODUCT t=1 to m (1/sqrt(2*pie)*sigma * exp -(y - theta_tau (x))/2 * sigma^2 = m log 1/(sqrt 2 *pie) SUM(i=1 to m (y^i - theta^T *x)^2 / 2* sigma^2

- Probability density = P(e^i) = 1/sqrt(2*pie)*sigma exp(- e^i/2*sigma^2)
- J(tehta)= SUM i=1 to n (y^i - theta ^T *x^i)^2

- GLM
  - N is natural parameter
  - Function that connects n with mean of distribution is called canonical response function
  - Train theta to predict n which is input to the distribution and output of that distribution is variable which we want to predict
  - Linear regression gives us theta which is a line or vector
  - GLM classifies that vector in to a distribution
- Discriminative learning algorithm
  - Model classes in isolation and test data is compared with both models
  - Learn P(y |x) = h_theta(x)= 0,1
- GLA
  - Learns P(x|y) where x= feature and y=class
  - Given y=1 or y=0 predicts the feature x
  - P(y) = class prior
- Baye's rule
  - p(y=1 | x) = = P(x|Y=1) * P(y=1) / p(x)
  - p(x)= p(x |y=1) * P(y=1) + P(x|y=0) * P(y=0
  - P(x|y) and p(y) are learnt from model using these learnt terms we can calculate p(y=1 | x)
  - Use sample data to train two models in separate which learns features x if y=1 and y=0 and also learnt p(y=0,1)
  - Using baye's rule we can learn P(y=1 |x)

Unsupervised Learning

Hand Written Notes
https://drive.google.com/drive/folders/13UQy8mvkiKTH3FwCJoSKuAxEMcoxUjXH?usp=share_link

Summary CS- 229

Hypothesis function: The choice of hypothesis function will depend on the problem domain and the type of learning task. In addition to defining the functional form of the model, you may need to choose hyperparameters, such as the number of hidden layers or the regularization strength.

Loss function: The choice of loss function will also depend on the problem domain and the type of learning task. In addition to mean squared error (MSE), there are many other loss functions that may be appropriate for different types of problems, such as cross-entropy loss for classification problems. You may also need to consider how to handle class imbalance or other challenges that may affect the choice of loss function.

Parameter estimation: There are different methods for estimating the parameters of the model, including maximum likelihood estimation (MLE), maximum a posteriori (MAP) estimation, and Bayesian inference. MLE is a commonly used approach, but other methods may be more appropriate depending on the problem domain and the available data.

Optimization algorithms: Gradient descent is a commonly used optimization algorithm for minimizing the loss function, but there are many other optimization algorithms that may be more efficient or effective for different types of problems. For example, stochastic gradient descent (SGD) is a popular variant of gradient descent that can be more efficient for large datasets. Other optimization algorithms, such as Adam or Adagrad, may be more effective for optimizing deep neural networks.

Model evaluation: Once you have trained the model, you will need to evaluate its performance on a validation set or test set. In addition to standard metrics such as accuracy or mean squared error, you may need to consider other metrics that are more appropriate for specific problems, such as precision and recall for imbalanced classification problems.

Token Engineering Framework

- Incentive & token design
- Optimal pricing and resource allocation in distributed networks
- Real-world experience & business impact
- Network analytics & data-driven monitoring
- Formation, diffusion, and learning in networks
- Modeling & simulation
- Value attribution and graph-based algorithms
- Evolutionary game theory, population games, state-based potential games
- Prediction markets, automated market makers, reputation systems
- Governance process & principles

Discovery Phase

- TUC
  - Actors
  - Desired and undesired behaviors
  - Mechanism
  - Value Preposition (Creation & Capture)
- Stakeholders Map
  - Identifying participants and their roles
  - Incentives and disincentives
- Value Exchange Map
  - FLow of value within the ecosystem
  - Monetary and non monetary
- MECE (Mutually Exclusive, Cumulatively Exhaustive)
- Taxonomy of actors

Design Phase

- Design and Engineer an ecosystem which aligns to secure and incentives
- Network Analysis
    - Direct and Indirect participants
- Network Objective function definition
- Define Metrics of Network Success
- Constraints definition
- Assumptions about:
    - Participants
    - Problems (Core (Macro) and Periphery (Micro))
- Experiments:15 In this section, every assumption in the above category needs to be validated through experimentation. Experiments should be run sequentially from top to bottom and from left to right:
- Moving from top to bottom. The results from testing down through Participants Problem Incentives Solution all feed into the next sequence of experiments and mark the creation of the first interactive feedback loop in the token design. This feedback loop is critical and should be present across all stages of the token design.
- Moving from left to right. Experiments gradually become more sophisticated and specific, starting with simple surveys, then moving to experiments, modeling and simulations, and then A/B testing.
- Decision Theory: Individual's own preferences and Constraints
- Game Theory: Interactive decisions, how individual's make decisions in competitive and/or competitive environment
    - Find out contradictions in self interest of participants to the interests of the network
    - Alignment of stakeholders incentives
    - In an experiment/randomized control trial (RCT), participants are randomly assigned to treatment (where they receive the intervention) vs. control groups (where they do not receive the intervention). Randomization is crucial as it allows us to compare the effectiveness of a treatment

objectively. Using the right treatments and controls, we can get robust causal inference. In this setting, a smaller and simpler version of the a project's economy/interactions can be replicated with multiple users taking various representative roles within the network. In each treatment, we can change one single factor to determine how this would affect participants' incentives, hence their behavior and the whole ecosystem. Potential implementation fields for experimentally testing a specific behavior include MTurk and Experimental Labs . An additional option is to look at empirical data provided by existing networks, and if possible, identify and analyze natural experiments within them.

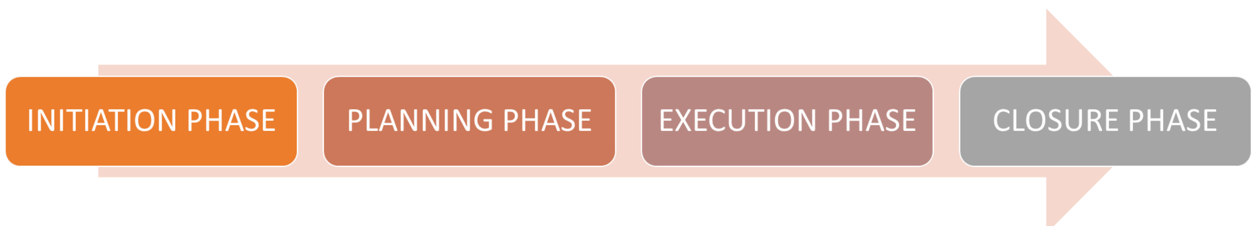
- Minimal Viable Token for our purposes is the simplest but most effective design possible to deliver upon the Objective Function, within a system's set constraints, determined during the token design process.
- Token Network Fit
- Ledger Market Fit
- 1. Create hypotheses based on the participants' incentives which were defined.
- 2. Formulate ways to test this hypothesis and decide the best avenue for testing. For example, are experiments the best option for testing? Are we setting these up quickly enough? Will we have enough statistical power to detect any effect from experimental observations? Is it better to go for interviews and surveys or run simulations?
- Deliverables of a token design process
  - Objective function of the overall network that supports the ecosystem
  - Architecture of the token
  - Circulating supply model
  - Velocity model
  - Utility model
  - Networks effects model
  - Token Distribution
    - Business
    - Technical

- Legal

Token Design - Process

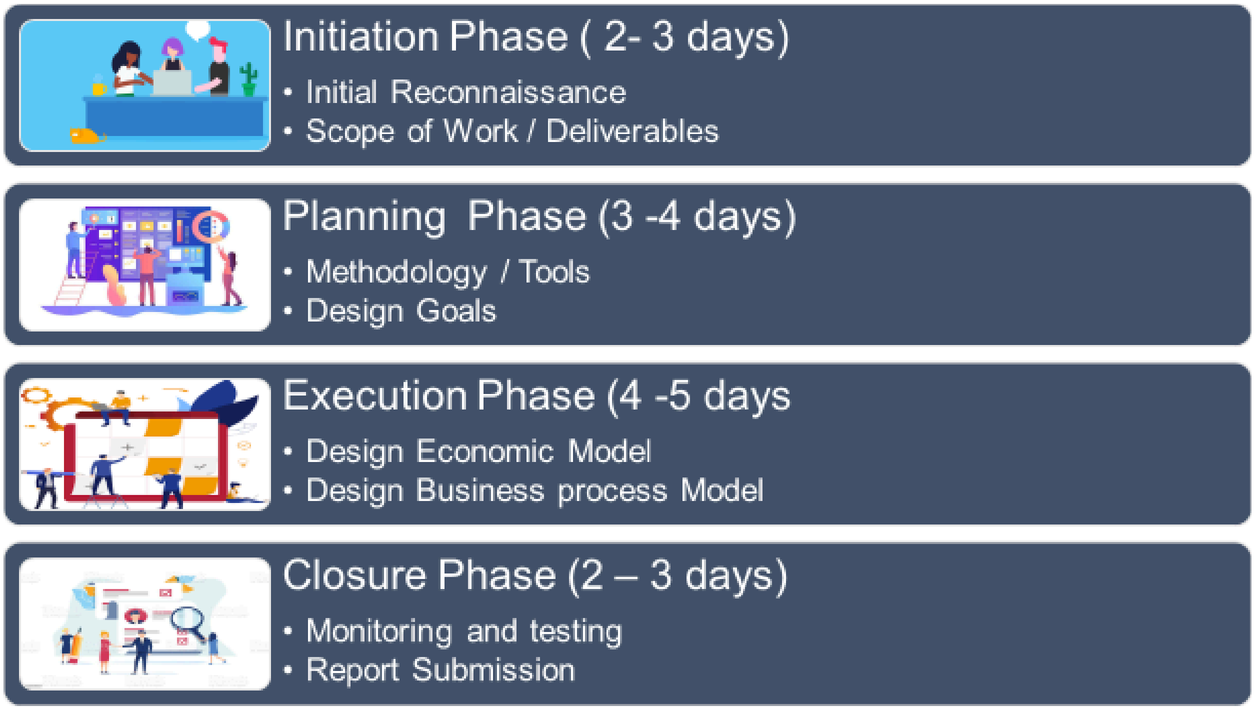
Process

Process of tokenomics design is divided in four following phases;



Timeline

Time required to complete a tokenomics project is 14 - 15 days depending upon complexity of the project. Distribution of timeline is given below;



Initiation Phase (2 - 3 days)

Initial reconnaissance

- Idea validation
- Analyze project document
- Identify and study similar project

Prepare Questionnaire

- What are agents of the system and how do they interact with each other?
- Main use cases and functionality of the token
- Economic goal of the token (price Appreciation, Stability, Inflation hedge etc.)

- Business goal of the project
- Total distribution
- Governance voting and resolution mechanisms
- Most similar projects and their markets
- Inflationary / deflationary
- Future utilities of token
- Token classification
- Overhead costs of the business

Define Scope of Work / Deliverables

- Burning and minting schedule
- Incentivization and desensitization mechanisms
- Financial Engineering / Model
- Failsafe mechanisms and triggering indicators

Initial Reconnaissance Meeting

- Submit reconnaissance report
- Share scope of work

Planning Phase (3 - 4 days)

Identify Methodology / Tools

- Empirical proof
- Agent-based modeling
- Game theory and Mathematical Modeling

- Marginal cases analysis
- Probability theory
- Financial Modeling

Identify Design Goals

- Token Utility
  - Why do you need a token?
  - Who uses a token?
  - Why should they use the Tokens?
  - Growth drivers
  - Holder incentives
  - Adoption
  - Will the utility increase with time / adoption?
    - Actions-Actors Approach

Parameters Definition / Constraints and Schedule

- Total / Circulating supply
- Unlock schedule of remaining supply
- Annual inflation / deflation
- Token Distribution
  - Different ways how Tokens enter and exit network
  - High enough circulation volume
  - More “Ins” than “outs”

Workflow Identification

- Sources of Input (Injection) & output (rewards)

- Assign weight to factors in incentive models to prioritize network growth(Curve: Gauge weight, Gauge type weight)
- Fee Distribution
  - Project Funding
  - Adding liquidity
  - Burn
  - Profit - Sharing

- Second meeting with client to update him about methodology and approach to be used

Execution Phase (4 - 5 days)

Mechanism Design

- Identification of Agents of system and their possible interactions with each other
- Set of agents, their attributes and behaviors.
- Set of agent relationships and methods of interaction: An underlying topology of connectedness defines how and with whom agents interact.
- The agents' environment: Agents interact with their environment in addition to other agents.
- Topology
  - Rules
  - Link Structure
- Environment
  - Stress test

- Behaviors emerging from interactions of agents

Economic Model

- Initial Supply
- Max Supply
- TVL growth strategy
- Deflationary Model
  - Buy Back and Burn
  - Burn on transactions
- Inflationary Model
  - Inflation schedule (Initial preset rate, Readjusting time period).
  - Inflation rate.
  - Prohibited on demand token minting.
  - Ensure optimal inflation rate.
  - Approximate inflow into circulating supply.

Mathematical Model - Business Model

- Features of token
- “Ins” and “Outs” of token from system
- Alignment of reward mechanism with enhancement of network adoption mechanism

- Incentivize users to participate in governance
- Profit - Sharing
- Burning
- Staking
- Governance
- Incentives to ensure long enough token hold time
  - Amount and time weighted incentives
  - Slashing mechanism upon quick withdrawal

Game Theory Design

- Desired behaviors of participants and their relation with protocol incentive mechanism
- Sustainability of model in controlled environment constraints
- Market sentiment analysis w.r.t incentive and disincentive mechanisms derived by network growth

Closure Phase (2 - 3 days)

Monitoring and Testing

- Agent Based Simulations testing
- Stress Test
- Financial Model dynamicity testing
- Marginal Cases Analysis

Report Submission

- Compile and submit Report
- Final Meeting with the client

1. Ecosystem Design Over The Token Design

Many industry players start with tokens. However, tokens might not be necessary and could create friction. Designing a token economy with a token-first approach can be detrimental to any decentralized application (dApp) or protocol because it removes resources from the underlying functionality, value, and utility. By prioritizing the token, developers might be more likely to neglect user experience, security, and scalability, which are critical to long-term ecosystem success.

Excessive focus on a token also creates artificial demand or speculative bubbles, ultimately resulting in market volatility. This approach can stress the internal team excessively, causing them to worry more about the token price and community sentiment than the core product itself.

Instead, the project team should adopt a product-first mentality. With a value-driven approach, the token seamlessly complements and enhances the core functionality of the dApp or protocol. The product-first mentality creates an intrinsic link between token value and platform utility, fostering organic demand and sustainable growth.

On LooksRare, for example, as much as 90% of the activity consists of wash trading, where traders engage in this practice to earn $LOOKS rewards worth more than the fees they pay. Subsequently, they dump the $LOOKS to make a profit. Does the token benefit the project or make it worse? Sometimes, tokens can create unintended consequences and harm the project.

2. Value Creation Aligned With Rewards

In tokenomics, reward alignment is the coordination of incentives for different stakeholders toward a common goal.

Aligning rewards with value creation is essential for building a thriving token economy. It fuels growth by engaging key innovators who contribute to project development. Rewarding value creation promotes adoption and fosters sustainable expansion. Honestly, this is the most essential principle since misaligning the incentives may distort the project economy and finally lead it to collapse.

We could divide tokenomics system rewards into two buckets: aligned and misaligned**.**

Aligned incentives. Well-aligned incentives create a self-sustaining ecosystem rewarding users for contributing to the network's growth and development. It can promote network security, encourage decentralized governance, and increase the value of the network's native token by reducing selling pressure on the market.

Look no further than Bitcoin for well-aligned incentives: miners contribute computational power to secure the network in exchange for new bitcoins.

Misaligned incentives. Wrong incentives create undesired outcomes undermining long-term viability and success. Such mechanisms concentrate power in the hands of a few stakeholders, reduce network security, and discourage user participation.

Airdrops are a well-known example of misaligned incentives. Users often dump the token they receive at the first chance. It's a zero-sum game in which the protocol receives no long-term value from token distribution and only wastes resources.

It is self-explanatory that you should choose aligned rewards if you want your protocol to flourish.

The key takeaway is to address and manage toxic behaviors rather than blindly hope a single good-mannered behavior will be enough to make your digital economy successful. Speaking of breaking overly optimistic lenses, it's time we talk about malicious actors in your token model.