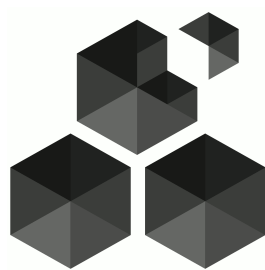


Developing Scalable Decentralized Applications for



Swarm & Ethereum

presentation by
Dániel A. Nagy at Devcon 2, Shanghai



What's wrong with Web 2.0?

- Scaling: demand and supply are disconnected
- Centralized control: excessive trust in 3rd parties

However,

- trustless computation will always involve considerable overhead.



From Web 2.0 to Web 3.0

- Common, general-purpose, distributed backend
 - Content-addressed storage (swarm/bzz)
 - Blockchain-based consensus (Ethereum/eth)
 - Asynchronous messaging (whisper/shh)
- Particular business logic pushed to clients
 - Web application written in js
 - Native mobile application

Example:

DB access by index traversal instead of queries



Swarm high-level API

- URL begins with object collection's root hash
 - `bzz://ae11358a5...b2e4228/imgs/example.png`
 - `bzz://473b50...70c004/tiles/203131100221.png`
- Any change results in new root hash
 - PUT
 - DELETE
- Root hashes can be registered on blockchain-based hierarchical name registries:
 - `bzz://joe.album/imgs/example.png`



Web 3 user experience

- Familiar: hypertext with multimedia in a browser
 - Interactive, responsive, intuitive
- Personalization and identity management
 - Selectable personae, identities
 - Part of browser, not application
- Legal and financial interactions
 - Binding agreements
 - Payment with provable receipts
 - Rate-limits, confirmations with passwords, etc.



Simple Dapp mechanics

- Current root hash registered on block chain
- Most static and dynamic data in Swarm
- Global state changes on block chain
- Local state changes stored locally
 - Optionally backed up in swarm and/or block chain
- Business logic gets executed locally
 - But verified globally by means of Ethereum



Ðapp example #1

distributed photo album

- Web-app & data hosted in swarm
 - Root hash of collections published on block chain
 - Long-term incentives make sure it is not gc'd
 - Short-term incentives drive publishing costs down
 - High performance irrespective of popularity
- No concurrent editing
 - Each collection is only edited by one contributor
 - All editing is done by the editor's computer
- No comments or ratings



Ɗapp example #1

distributed photo album

- Behind the scenes: thumbnails and blurred backgrounds are generated by client during upload.
- Is this a sustainable pattern for similar Ɗapps?
 - Panorama transformation (photo / video)
 - Video transcoding
 - Thumbnail frames for videos
 - Automatic subtitles
 - OCR for texts
 - Music fingerprinting



Delegated computations

- Incentives!
- Input: **data + specification + reward**
- Output: **result + proof**
- Specification
 - Imperative (procedural)
 - Declarative
 - Precise
 - Fuzzy
- Arbitration



Ðapp example #2

distributed shared folders

- Two end-user interfaces:
 - Web-app (swarm-hosted, blockchain-registered)
 - Locally synchronized folder
- Concurrent changes possible, but not typical
 - User alerts, manual resolution



Ðapp example #3

distributed social network

- Personalization: list of followed contributors
 - E.g. friends contributing comments, likes, etc.
 - Their number is limited
- Content is rendered by traversing this list
 - For each post, friend list is scanned for comments to this post
 - Single root hash on the block chain for each participant, changes with editing, publishing, commenting

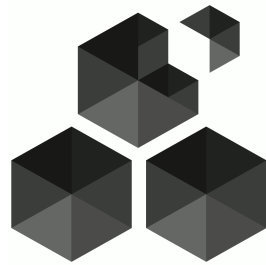


Ðapp example #4

distributed map/encyclopedia

- No “official truth”, forking is cheap
 - Alternative perspectives face no prohibitive costs
 - Continued “rebasing” keeps all versions up-to-date
 - No edit wars, no blackouts
- Groups or individuals can have own versions
 - Registered on the block chain by root hash
 - Requires some editor work, but not much
- Few versions of individual records
 - Opinions on any single topic tend to cluster around a few alternatives

Thank you!



Questions?