The roles of FORTH in blockchain technologies and beyond

We investigate the roles of FORTH in Blockchain related technologies and novel decentralised schemes beyond Blockchain, with the potential of reinventing the Internet, but without the huge footprint required by Blockchain.

Liang Ng, November 2025
omnixtar.github.io/svfig
https://www.linkedin.com/in/liang-ng

- Bitcoin uses FORTH like syntax to verify the hash of public key.

1. Extend the use of hash of public key as user identifier in decentralised network and explore various properties of hash codes,
2. Omnihash, literally meaning a hash code for representing all types of digital assets, and DJSON decentralized JSON, a special type of JSON string containing at least one Omnihash
3. Omnihash as the basis of hash filesystem, which becomes the foundation of a database system without database server.
4. 3X architecture, namely database-less, domain-less and kernel-less
5. Phoscript and Omnihash as used in Web and mobile applications,
   Phoscript metaprogramming shell: ported to any host programming language, including front end as well as back end environments
   Phase II of 3X architecture: Sandwich model of code substitution
   substitute user space services such as Web server and I2P routers with FORTH or Phoscript exclusive code base.

A Bitcoin address is only a hash, so the sender can't provide a full public key in scriptPubKey. When redeeming coins that have been sent to a Bitcoin address, the recipient provides both the signature and the public key. The script verifies that the provided public key does hash to the hash in scriptPubKey, and then it also checks the signature against the public key.

Checking process:

Stack	Script	Description
Empty.	<sig> <pubKey> OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	scriptSig and scriptPubKey are combined.
<sig> <pubKey>	OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	Constants are added to the stack.
<sig> <pubKey> <pubKey>	OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	Top stack item is duplicated.
<sig> <pubKey> <pubHashA>	<pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	Top stack item is hashed.
<sig> <pubKey> <pubHashA> <pubKeyHash>	OP_EQUALVERIFY OP_CHECKSIG	Constant added.
<sig> <pubKey>	OP_CHECKSIG	Equality is checked between the top two stack items.
true	Empty.	Signature is checked for top two stack items.

- Bitcoin uses FORTH like syntax to verify the hash of public key.

1. Extend hash of public key as user identifier in decentralised network and explore various properties of hash codes, (important for subsequent steps)
2. Omnihash, literally meaning a hash code for representing all types of digital assets, and DJSON decentralized JSON, a special type of JSON string containing at least one Omnihash
3. Omnihash as the basis of hash filesystem, which becomes the foundation of a database system without database server.
4. 3X architecture, namely database-less, domain-less and kernel-less
5. Phoscript and Omnihash as used in Web and mobile applications,
   Phoscript metaprogramming shell: ported to any host programming language, including front end as well as back end environments
   Phase II of 3X architecture: Sandwich model of code substitution
   substitute user space services such as Web server and I2P routers with FORTH or Phoscript exclusive code base.

DJSON Decentralised JSON is a JSON object or its encoded string where at least one of the fields is an Omnihash, representing the owner of this JSON object.

- `["2025-10-24T14:25:28.207+0000","like","CXAGcRKevA==","CXAGcRKevA==","HymWBzfj9A==",{"repo":"https://github.com/omnixtar/omnixtar.github.io/","contract":"https://omnixtar.github.io/contract/","ghh":"https://github.com/omnixtar/omnixtar.github.io/commit/19bb258190d57d6246840bf8ccc8957ae880e341","datetime":"2025-10-24T04:41:21.000Z"}]`
- Omnihash: DgV6_qnujw==

Try this yourself:

1. Press F12 to bring up browser console.
2. Run the following code:

```
omnistart()
j0=["2025-10-24T14:25:28.207+0000","like","CXAGcRKevA==","CXAGcRKevA==","HymWBzfj9A==",{"repo":"http
s.push(JSON.stringify(j0))
f('h53: b64: path:')
s[s.length-1]
```

- Bitcoin uses FORTH like syntax to verify the hash of public key.

1. Extend the use of hash of public key as user identifier in decentralised network and explore various properties of hash codes,
2. Omnihash, literally meaning a hash code for representing all types of digital assets, and DJSON decentralized JSON, a special type of JSON string containing at least one Omnihash
3. Omnihash as the basis of hash filesystem, which becomes the foundation of a database system without database server.
4. 3X architecture, namely database-less, domain-less and kernel-less
5. Phoscript and Omnihash as used in Web and mobile applications,
   Phoscript metaprogramming shell: ported to any host programming language, including front end as well as back end environments
   Phase II of 3X architecture: Sandwich model of code substitution
   substitute user space services such as Web server and I2P routers with FORTH or Phoscript exclusive code base.

```
hongwu@hongwu-Latitude-5480:/var/www/dmeta/oxm/auth$ cat Graph/hg/FtyMvMgeoA\=\= ;echo
{"a":"1","t":"2025-10-16 09:04:43.683200","n":"Adam in MY wish to send MYR 100k to Donald in TH.","s
n":"1","job":"payment","n2":"variables","MYR":"currency","amount":"100k","sender":"Adam","recipient"
:"Donald"}
```

Hash of JSON is FtyMvMgeoA==

```
hongwu@hongwu-Latitude-5480:/var/www/html/oxw/auth/Graph/dmeta$ \
> cat H-xchGCVBg\=\=/BtiFODs5CQ\=\= ;echo
["2025-10-18T00:10:43.684+0000","reply_to","H-xchGCVBg==","G5bClrzsVg==",
"Ge6NiA5cLw==","Graph\/hg\/FtyMvMgeoA=="]
```

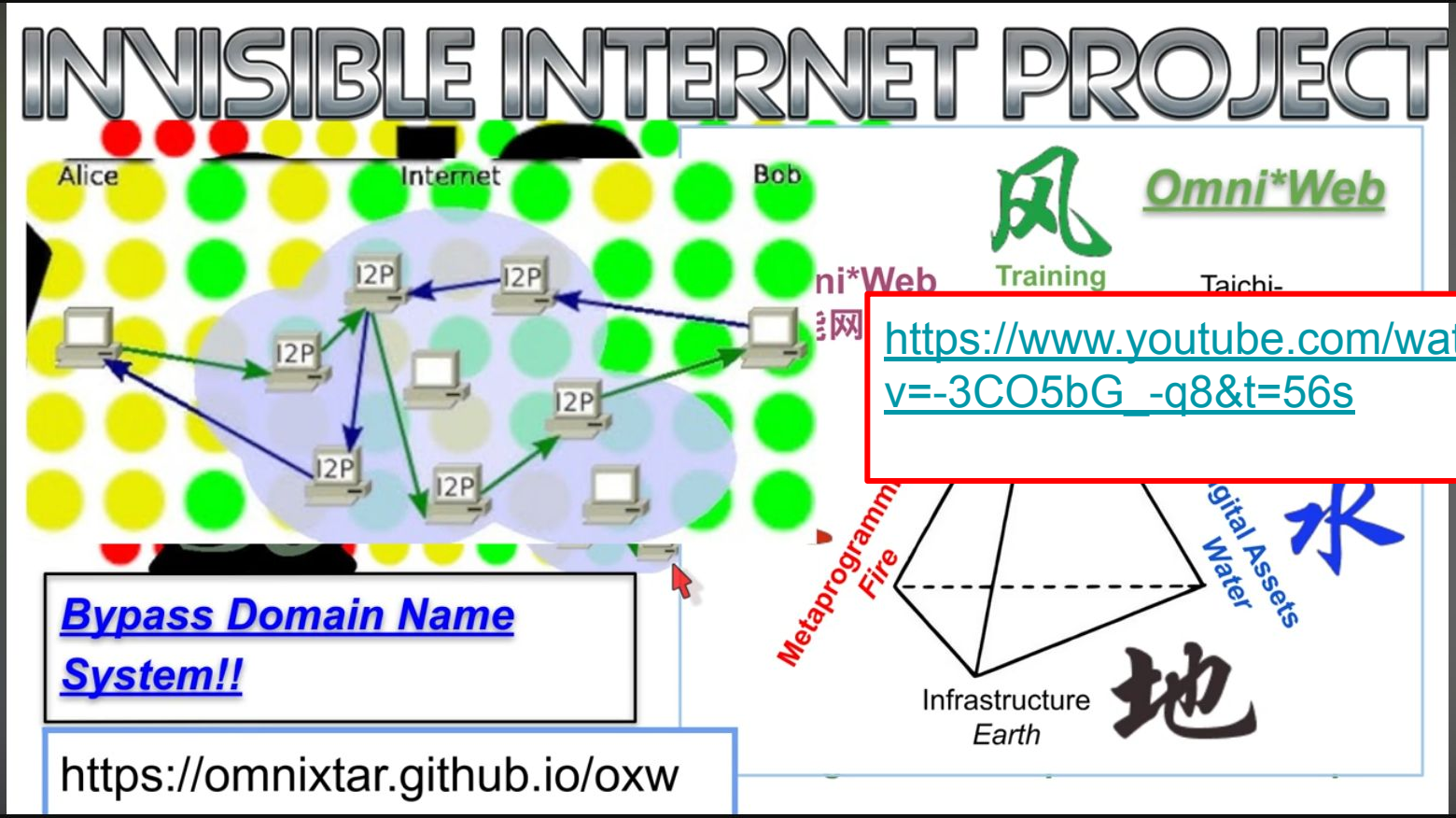
User Adam H-xchGCVBg== sends FtyMvMgeoA== to User Donald G5bClrzsVg==

```
hongwu@hongwu-Latitude-5480:/var/www/html/oxw/auth/Graph/dmeta$ \
> cat H-xchGCVBg\=\=/CxenfQAHxw== ;echo
["2025-10-18T00:58:01.196+0000","reply_to","G5bClrzsVg==","H-xchGCVBg==",
"BtiFODs5CQ==","ACCEPT chris agent apk:"]
```

User Donald replied ACCEPT and appointed User Chris as his agent.

- Bitcoin uses FORTH like syntax to verify the hash of public key.

1. Extend the use of hash of public key as user identifier in decentralised network and explore various properties of hash codes,
2. Omnihash, literally meaning a hash code for representing all types of digital assets, and DJSON decentralized JSON, a special type of JSON string containing at least one Omnihash
3. Omnihash as the basis of hash filesystem, which becomes the foundation of a database system without database server.
4. 3X architecture, namely database-less, domain-less and kernel-less
5. Phoscript and Omnihash as used in Web and mobile applications,
   Phoscript metaprogramming shell: ported to any host programming language, including front end as well as back end environments
   Phase II of 3X architecture: Sandwich model of code substitution
   substitute user space services such as Web server and I2P routers with FORTH or Phoscript exclusive code base.

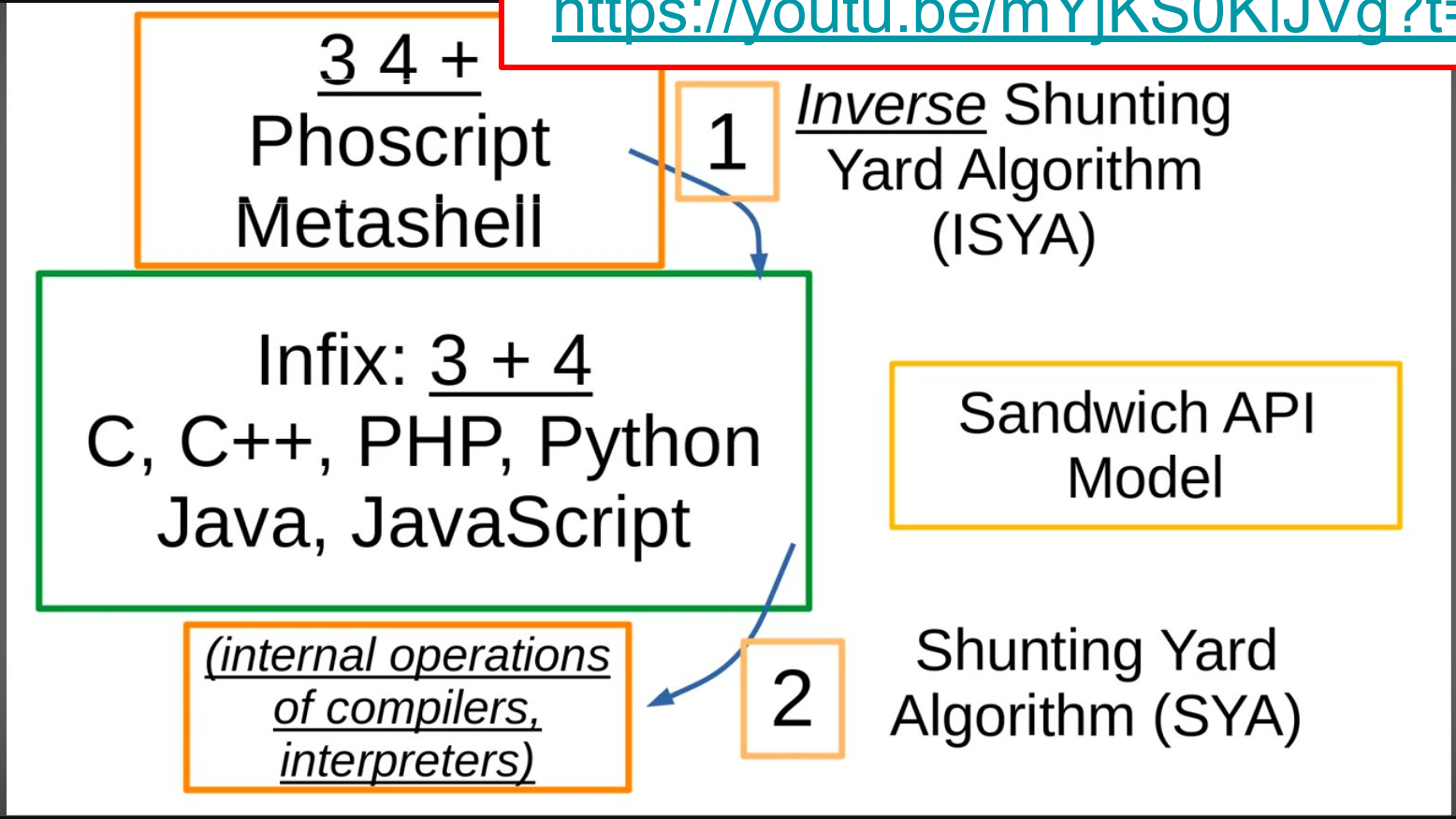


Omni*Web: Set Up YrOwn GLOBALLY ACCESSIBLE Web Server w/o Domain Name System in 4 Simple Steps (I2P)

- Bitcoin uses FORTH like syntax to verify the hash of public key.

1. Extend the use of hash of public key as user identifier in decentralised network and explore various properties of hash codes,
2. Omnihash, literally meaning a hash code for representing all types of digital assets, and DJSON decentralized JSON, a special type of JSON string containing at least one Omnihash
3. Omnihash as the basis of hash filesystem, which becomes the foundation of a database system without database server.
4. 3X architecture, namely database-less, domain-less and kernel-less
5. Phoscript and Omnihash as used in Web and mobile applications,
   Phoscript metaprogramming shell: ported to any host programming language, including front end as well as back end environments
   Phase II of 3X architecture: Sandwich model of code substitution
   substitute user space services such as Web server and I2P routers with FORTH or Phoscript exclusive code base.

https://youtu.be/mYjKS0KiJVg?t=455



Bidirectional Shunting Yard Algorithm (BISYA) and Sandwich API Model: Unifying Programming Languages

3X architecture: database-less, domain-less and kernel-less

6. Phase III of 3X architecture: applied to the Linux kernel
   - FORTH Phoscript exclusive code base,
   - the ultimate post-blockchain, light-weight, card size computers plus kilobyte virtual machines
   - turn almost every electronic device into a FORTH computer node
   - connected using decentralised hash codes (hash filesystem)
   - to realise FORTH version of Sun Microsystem vision of "the network is the computer": Decentralised AI, nodes owned by users, NOT MMAGA!!
7. MMAGA (Microsoft,Meta,Amazon,Google,Apple): USD 1.8Trillion (2024 revenues)
   - Omni*Web 3X Architecture: 0.1% of MMAGA revenues by 2030?
   - 0.1% of Bitcoin values for 0.1% of world population by 2035?
   - Metanarchy (Decentralised Autonomous Organisations) – better government Decentralised Global Governance based on Transactions in Metaverse (Outside United States of America & China)?

Phoscript-Linux/C-FORTH Sandwich Model

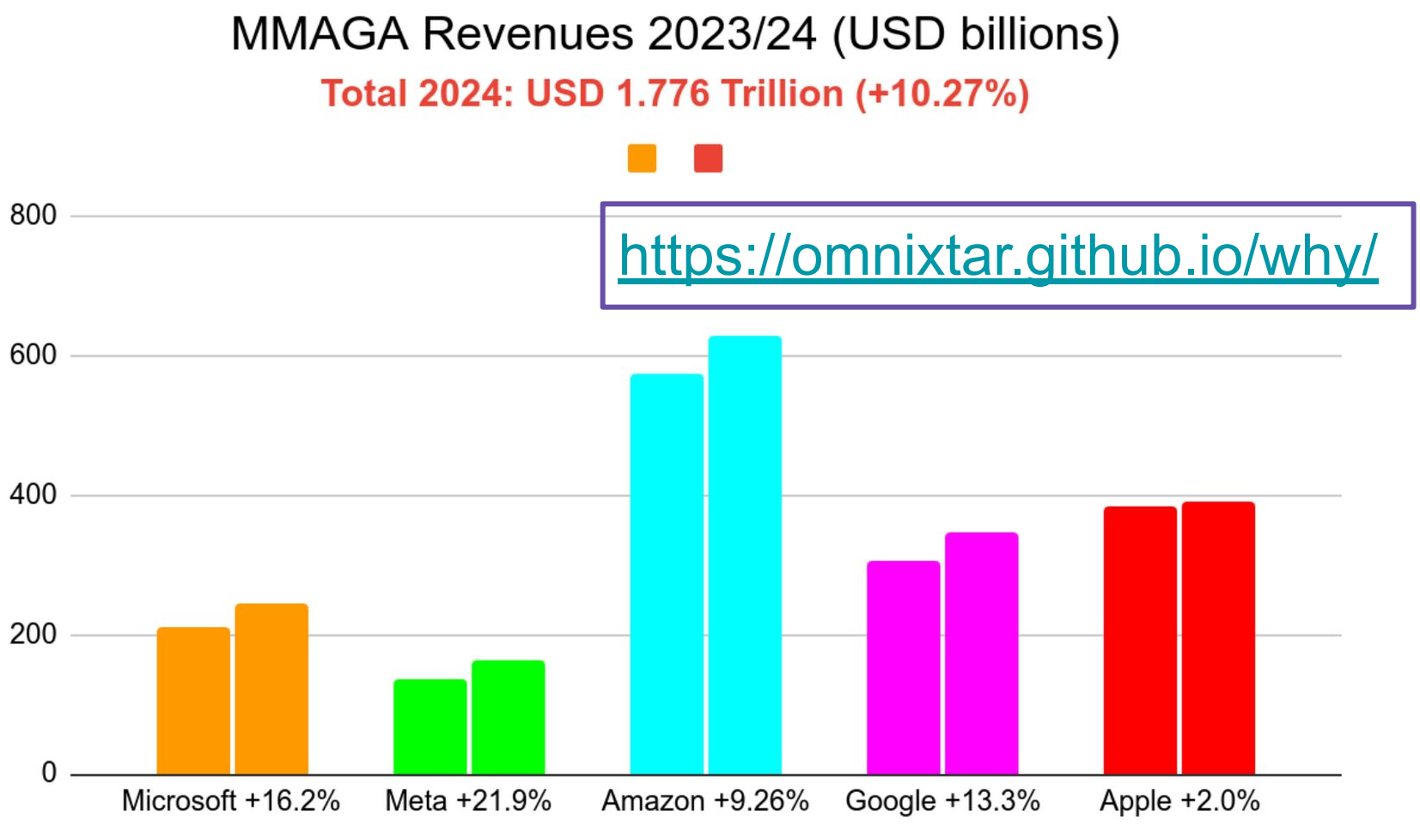
1. Minimal VM: Linux Java I2P Apache php
2. Sandwich Model (user space programs):
   a. Top: P2C (Phos to C) P2J (Phos to Java) P2PHP
      [start: 0, end: replace host language functions]
   b. Middle: C-lib, J-lib (Java), PHP-lib in .o (object);
      [start: 100%, end: replaced by Top & Bottom]
   c. Bottom: need equivalent F-lib (FORTH)
      [start: 0, end: replace Middle layer host language
3. Replace Kernel? (Need experience from Phase 2)
   a. Use P2C to replace C code with Phoscript code.
   b. Write FORTH code to replace low level C code.

Phase II of 3X architecture will begin with the current state of Phoscript metaprogramming shell,
- end goals of eliminating user space services, by substituting them with FORTH based libraries,
- such as web server and I2P Invisible Internet Project or equivalent services, that is
- to achieve the domain-less goal.

3X architecture: database-less, domain-less and kernel-less

6. Phase III of 3X architecture: applied to the Linux kernel
   - FORTH Phoscript exclusive code base,
   - the ultimate post-blockchain, light-weight, card size computers plus kilobyte virtual machines
   - turn almost every electronic device into a FORTH computer node
   - connected using decentralised hash codes (hash filesystem)
   - to realise FORTH version of Sun Microsystem vision of “the network is the computer”: Decentralised AI, nodes owned by users, NOT MMAGA!!

7. MMAGA (Microsoft,Meta,Amazon,Google,Apple): USD 1.8Trillion (2024 revenues)
   - Omni*Web 3X Architecture: 0.1% of MMAGA revenues by 2030?
   - 0.1% of Bitcoin values for 0.1% of world population by 2035?
   - Metanarchy (Decentralised Autonomous Organisations) – better government Decentralised Global Governance based on Transactions in Metaverse (Outside United States of America & China)?


MMAGA Revenues 2023/24 (USD billions)
Total 2024: USD 1.776 Trillion (+10.27%)
https://omnixtar.github.io/why/
800
600
400
200
0
Microsoft +16.2%
Meta +21.9%
Amazon +9.26%
Google +13.3%
Apple +2.0%

(In millions, except per share amounts) (Unaudited)	Three Months Ended March 31, 2025	2024	Nine Months Ended March 31, 2025	2024
Revenue:				
Product	**$ 15,319**	$ 17,080	**$ 46,810**	$ 51,556
Service and other	**54,747**	44,778	**158,473**	128,839
Total revenue	**70,066**	61,858	**205,283**	180,395
Cost of revenue:				
Product	**3,037**	4,339	**10,187**	13,834
Service and other	**18,882**	14,166	**53,630**	40,596
Total cost of revenue	**21,919**	18,505	**63,817**	54,430
Gross margin	**48,147**	43,353	**141,466**	125,965
General and administrative	**1,737**	1,912	**5,233**	5,363
Operating income	**32,000**	27,581	**94,205**	81,508
Other expense, net	**(623)**	(854)	**(3,194)**	(971)
Income before income taxes	**31,377**	26,727	**91,011**	80,537
Provision for income taxes	**5,553**	4,788	**16,412**	14,437
Net income	**$ 25,824**	$ 21,939	**$ 74,599**	$ 66,100

https://www.microsoft.com/en-us/investor/earnings/fy-2025-q3/income-statements

Information contained in these documents is current as of the earnings date, and not restated for new accounting standards

- Earnings Call Slides >
- Earnings Call Transcript >
- Financial Statements >
- Outlook >
- Press Release >
- 10Q >
- FY25Q3 Product Release List >

ASSET PACKAGE

- Metrics >
- Performance >
- Press Release & Webcast >
- Financial Statements >
- Segment Results >
- Customer & Partner Highlights >

META PLATFORMS, INC.

CONDENSED CONSOLIDATED STATEMENTS OF INCOME

https://investor.atmeta.com/investor-news/press-release-details/2025/Meta-Reports-Third-Quarter-2025-Results/default.aspx

	Three Months Ended September 30,		Nine Months Ended September 30,	
	2025	2024	2025	2024
Revenue	$ 51,242	$ 40,589	$ 141,073	$ 116,116
Costs and expenses:				
Cost of revenue	9,206	7,375	25,269	21,322
Research and development	15,144	11,177	40,237	31,693
Marketing and sales	2,845	2,822	8,581	8,107
General and administrative	3,512	1,865	8,455	8,978
Total costs and expenses	30,707	23,239	82,542	70,100
Income from operations	20,535	17,350	58,531	46,016
Interest and other income, net	1,128	472	2,047	1,095
Income before provision for income taxes	21,663	17,822	60,578	47,111
Provision for income taxes*	18,954	2,134	22,888	5,589
Net income	$ 2,709	$ 15,688	$ 37,690	$ 41,522
Earnings per share:				
Basic	$ 1.08	$ 6.20	$ 14.96	$ 16.37
Diluted	$ 1.05	$ 6.03	$ 14.62	$ 15.88
Weighted-average shares used to compute earnings per share:				
Basic	2,517	2,529	2,520	2,536
Diluted	2,572	2,600	2,578	2,615

https://ir.aboutamazon.com/news-release/news-release-details/2025/Amazon-com-Announces-Third-Quarter-Results/

	Three Months Ended September 30,		Nine Months Ended September 30,	
	2024	2025	2024	2025
Net product sales	$ 67,601	$ 74,058	$ 190,085	$ 206,274
Net service sales	91,276	106,111	260,082	297,264
Total net sales	158,877	180,169	450,167	503,538
Operating expenses:				
Cost of sales	80,977	88,670	227,395	246,455
Fulfillment	24,660	27,679	70,543	78,248
Technology and infrastructure	22,245	28,962	64,973	79,122
Sales and marketing	10,609	11,686	30,783	32,865
General and administrative	2,713	2,875	8,496	8,468
Other operating expense (income), net	262	2,875	587	3,382
Total operating expenses	141,466	162,747	402,777	448,540
Operating income	17,411	17,422	47,390	54,998
Interest income	1,256	1,100	3,429	3,251
Interest expense	(603)	(538)	(1,836)	(1,595)
Other income (expense), net	(27)	10,186	(2,718)	14,052
Total non-operating income (expense)	626	10,748	(1,125)	15,708
Income before income taxes	18,037	28,170	46,265	70,706
Provision for income taxes	(2,706)	(6,910)	(6,940)	(14,141)
Equity-method investment activity, net of tax	(3)	(73)	(81)	(87)
Net income	$ 15,328	$ 21,187	$ 39,244	$ 56,478
Basic earnings per share	$ 1.46	$ 1.98	$ 3.76	$ 5.31
Diluted earnings per share	$ 1.43	$ 1.95	$ 3.67	$ 5.22
Weighted-average shares used in computation of earnings per share:				
Basic	10,501	10,674	10,447	10,638
Diluted	10,735	10,845	10,705	10,815

Alphabet Inc.
CONSOLIDATED STATEMENTS OF INCOME
(In millions, except per share amounts, unaudited)

	Quarter Ended September 30,		Year To Date September 30,	
	2024	2025	2024	2025
Revenues	$ 88,268	$ 102,346	$ 253,549	$ 289,007
Costs and expenses:				
Cost of revenues	36,474	41,369	105,693	116,768
Research and development	12,447	15,151	36,210	42,515
Sales and marketing	7,227	7,205	20,445	20,478
General and administrative	3,599	7,393	9,783	16,141
Total costs and expenses	59,747	71,118	172,131	195,902
Income from operations	28,521	31,228	81,418	93,105
Other income (expense), net	3,185	12,759	6,154	26,604
Income before income taxes	31,706	43,987	87,572	119,709
Provision for income taxes	5,405	9,008	13,990	21,994
Net income	$ 26,301	$ 34,979	$ 73,582	$ 97,715
Basic net income per share	$ 2.14	$ 2.89	$ 5.96	$ 8.06

https://s206.q4cdn.com/479360582/files/doc_financials/2025/q3/2025q3-alphabet-earnings-release.pdf

calculation 12,419 12,203 12,480 12,230

Apple Inc.

CONDENSED CONSOLIDATED STATEMENTS OF OPERATIONS (Unaudited)
(In millions, except number of shares, which are reflected in thousands, and per-share amounts)

	Three Months Ended		Nine Months Ended	
	June 28, 2025	June 29, 2024	June 28, 2025	June 29, 2024
Net sales:				
Products	$ 66,613	$ 61,564	$ 233,287	$ 224,908
Services	27,423	24,213	80,408	71,197
Total net sales [1]	94,036	85,777	313,695	296,105
Cost of sales:				
Products	43,620	39,803	147,097	140,667
Services	6,698	6,296	19,738	18,634
Total cost of sales	50,318	46,099	166,835	159,301
Gross margin	43,718	39,678	146,860	136,804
Operating income	28,202	25,352	100,623	93,625
Other income/(expense), net	(171)	142	(698)	250
Income before provision for income taxes	28,031	25,494	99,925	93,875
Provision for income taxes	4,597	4,046	15,381	14,875
Net income	$ 23,434	$ 21,448	$ 84,544	$ 79,000

https://www.apple.com/newsroom/2025/07/apple-reports-third-quarter-results/#:~:text=CUPERTINO%2C%20CALIFORNIA%20Apple%20today%20announced,12%20percent%20year%20over%20year.

Bitcoin, FORTH & Hash of Public Key

(B1A) This slide shows a quick demo of how Bitcoin uses FORTH like syntax to verify the hash of public key.

As you can see in the table, it starts with an empty stack shown at the top of the left column, and in the middle column is the list of tokens to be evaluated.

So the tokens in the script column will be removed one by from the front, and the results are shown in the stack column.

I am sure all FORTH programmers are familiar with this. I believe this is a good introduction to all programmers who are unfamiliar with FORTH.

BITCOIN

Bitcoin: Technical Concepts

Pay-to-PubKey-Hash (**Pay-to-Public-Key-Hash**, **P2PKH**) is the basic form of making a transaction and is the most common form of transaction on the Bitcoin network. Transactions that pay to a Bitcoin address contain P2PKH scripts that are resolved by sending the public key and a digital signature created by the corresponding private key.

The ScriptPubKey and ScriptSig for a transaction is shown below:

Table of Contents

https://bitcoinwiki.org/wiki/pay-to-pubkey-hash

Pay-to-PubKey-Hash Review

Two types of payment are referred as P2PK (pay to public key) and P2PKH (pay to public key hash).

Satoshi later decided to use P2PKH instead of P2PK for two reasons:

Elliptic Curve Cryptography (the cryptography used by your public key and private key) is vulnerable to a modified Shor's algorithm for solving the discrete logarithm problem on elliptic curves. In plain English, it means that in the future a quantum computer might be able to retrieve a private key from a public key. By publishing the public key only when the coins are spent (and assuming that addresses are not reused), such attack is rendered ineffective.

With the hash being smaller (20 bytes) it is easier to print and easier to embed into small storage mediums like

A Bitcoin address is only a hash, so the sender can't provide a full public key in scriptPubKey. When redeeming coins that have been sent to a Bitcoin address, the recipient provides both the signature and the public key. The script verifies that the provided public key does hash to the hash in scriptPubKey, and then it also checks the signature against the public key.

Checking process:

Stack	Script	Description
Empty.	<sig> <pubKey> OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	scriptSig and scriptPubKey are combined.
<sig> <pubKey>	OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	Constants are added to the stack.
<sig> <pubKey> <pubKey>	OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	Top stack item is duplicated.
<sig> <pubKey> <pubHashA>	<pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	Top stack item is hashed.
<sig> <pubKey> <pubHashA> <pubKeyHash>	OP_EQUALVERIFY OP_CHECKSIG	Constant added.
<sig> <pubKey>	OP_CHECKSIG	Equality is checked between the top two stack items.
true	Empty.	Signature is checked for top two stack items.

(B2A) Since 2009, Bitcoin has grown and moved into many different directions.

What we are about to show in this presentation are something very lightweight, as we shall demonstrate, which you may wish to help us verify that they are indeed novel as peer reviewers, and we are seeking collaborators, as we have not attempted to publish them as academic publications, due to a combination of factors including lack of resources and uncertain directions.

Crypto-Metaprogramming (CMP)

Bitcoin – makes FORTH *the most popular programming language* by number of mining nodes(??), and variants of blockchains.
- and *the most VALUABLE programming language*.

But tied to miners and mining nodes – how to liberate the wealth and knowledge?

- 3X Architecture: No DNS, No Database, No Kernel.

Knowledge of Metaprogramming: revolutionise STEM (Science, Technology, Engineering, Mathematics) education, foundation of Metanarchy?

(C1A) As we have mentioned in the introduction, Our first innovation is that We extend the use of hash of public key as user identifier in decentralised network and explore various properties of hash codes, arriving at Omnihash, literally meaning a hash code for representing all types of digital assets, and DJSON decentralized JSON, a special type of JSON string containing at least one Omnihash, being our second innovation.

JSON strings exist in array and key-value forms, but they make no difference as far as DJSON is concerned, as the one hash string in DJSON should either be the hash of public key representing a user identifier, or the hash code of another DJSON, which eventually must contain one user identifier hash code.

As such, by virtue of association, if a user identifier exists alongside another hash code which points to a piece of digital asset, say a file containing source code of a program, then we may assume said user is claiming ownership over said digital asset.

In addition, the user may add other hash codes for different purposes, such as a detailed contract containing the terms and conditions for using the said digital assets.

And the most important point is that the claim of ownership is encoded in the DJSON itself, not depending on external systems, which has been the Achilles heel of other existing digital asset management systems.

Omni*Web

https://omnixtar.github.io/contract/

Omni*Contract: Ownership & Rights of Use of Digital Asset (Source Code)

Like

- On the Separation of Disclosure and Royalties of the Source Code July 21, 2024

1. You, a human agent of a company or government agency, may read the source code without making payments to the author or authors, but if you execute this program on behalf of your company or agency for commercial purposes, we reserve the rights to claim royalties from you or your company or agency.

2. Your copy of source code shall be attached with at least one Omni* Hash Contract bearing the Omnihash of a Omni* Agent and your own Omnihash, to authorise you the permissions to use or modify said source code, otherwise you shall pay maximum penalties allowed by a legal court of your jurisdiction, for the damages you have incurred for deploying the source code pertaining to clause (1).

Omnihash = Hash for any type of digital asset (photos, documents, videos, animations, simulations, PROGRAM SOURCE CODE etc.) = Hash of DJSON

DJSON Decentralised JSON = JSON string (object) containing at least one Omnihash (recursive definition)

Omni*Web: * = anything in Linux, cannot be copyrighted in English speaking countries? (NVIDIA, OpenAI, etc, all has “Omni” products.)

Crypto-Metaprogramming. Links.
https://omnixtar.github.io/djson/

Omni*Web: Crypto-Metaprogramming (CMP) as alternative to Model-View-Controller; towards Metanarchy
https://www.youtube.com/watch?v=P_M3PVn9J7I

DJSON Decentralised JSON is a JSON object or its encoded string where at least one of the fields is an Omnihash, representing the owner of this JSON object.

- `["2025-10-24T14:25:28.207+0000","like","CXAGcRKevA==","CXAGcRKevA==","HymWBzfj9A==",{"repo":"https://github.com/omnixtar/omnixtar.github.io/","contract":"https://omnixtar.github.io/contract/","ghh":"https://github.com/omnixtar/omnixtar.github.io/commit/19bb258190d57d6246840bf8ccc8957ae880e341","datetime":"2025-10-24T04:41:21.000Z"}]`
- Omnihash: DgV6_qnujw==

Try this yourself:

1. Press F12 to bring up browser console.
2. Run the following code:

```
omnistart()
j0=["2025-10-24T14:25:28.207+0000","like","CXAGcRKevA==","CXAGcRKevA==","HymWBzfj9A==",{"repo":"http
s.push(JSON.stringify(j0))
f('h53: b64: path:')
s[s.length-1]
```

Separation of Disclosure & Royalties

(C4A) Based on the dual chained property of hash codes, we arrived at one very important innovation on free software licenses, known as “separation of disclosure and royalties”, that is the author of the source code or the programmer, shall give permissions to third parties to view and analyse the source code in development and test environments, but shall reserve the rights to claim royalties if the source code and its binaries, if applicable, is used in production or commercial environments. We call this “separation of disclosure and royalties” as the conventional free software licenses mix them up. One might wonder, given hundreds of millions of very smart free software programmers out there, why is the idea of “separation of disclosure and royalties” almost unheard of, or are being restricted in isolated and small communities?

Omni*Web

https://omnixtar.github.io/contract/

Omni*Contract: Ownership & Rights of Use of Digital Asset (Source Code)

Like

- On the Separation of Disclosure and Royalties of the Source Code July 21, 2024

1. You, a human agent of a company or government agency, may read the source code without making payments to the author or authors, but if you execute this program on behalf of your company or agency for commercial purposes, we reserve the rights to claim royalties from you or your company or agency.

2. Your copy of source code shall be attached with at least one Omni* Hash Contract bearing the Omnihash of a Omni* Agent and your own Omnihash, to authorise you the permissions to use or modify said source code, otherwise you shall pay maximum penalties allowed by a legal court of your jurisdiction, for the damages you have incurred for deploying the source code pertaining to clause (1).

DJSON Decentralised JSON is a JSON object or its encoded string where at least one of the fields is an Omnihash, representing the owner of this JSON object.

- `["2025-10-24T14:25:28.207+0000","like","CXAGcRKevA==","CXAGcRKevA==","HymWBzfj9A==",{"repo":"https://github.com/omnixtar/omnixtar.github.io/","contract":"https://omnixtar.github.io/contract/","ghh":"https://github.com/omnixtar/omnixtar.github.io/commit/19bb258190d57d6246840bf8ccc8957ae880e341","datetime":"2025-10-24T04:41:21.000Z"}]`
- Omnihash: DgV6_qnujw==

Try this yourself:

1. Press F12 to bring up browser console.
2. Run the following code:

```
omnistart()
j0=["2025-10-24T14:25:28.207+0000","like","CXAGcRKevA==","CXAGcRKevA==","HymWBzfj9A==",{"repo":"http
s.push(JSON.stringify(j0))
f('h53: b64: path:')
s[s.length-1]
```

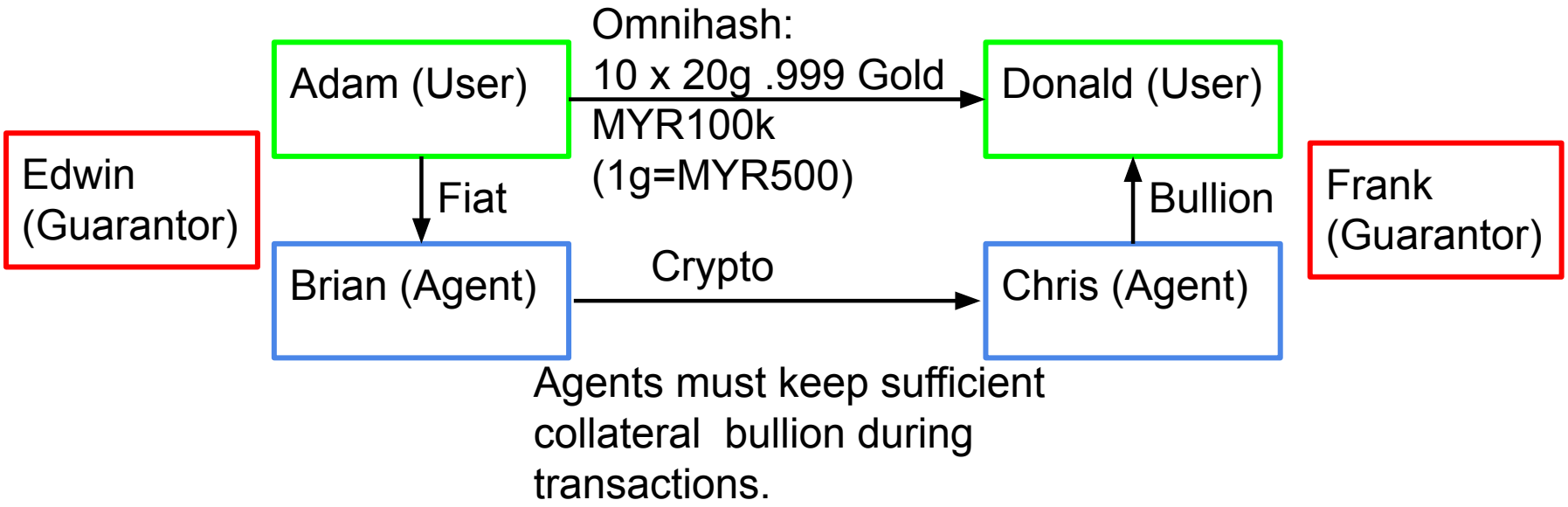
Hash Filesystem

(C5A) The fourth magical property of hash string is that it can be readily used as the filename of the hashed contents, with all the necessary security features required by a database system without a database server, so we call it “hash filesystem”. This is the most important property that gives rise to the 3X architecture, namely database-less.

- Due to lack of time, we will just summarise the benefits of “hash filesystem” by pointing out the fact that a filename that is a hash code may point to a JSON string containing the hash of public key that is a user identifier, therefore eliminating the centralised user authentication scheme which requires the user to manually enter his or her password.
- THIS MAGICAL PROPERTY eliminates the need for blockchains altogether!!
- ***TRILLION DOLLAR PROPERTY,*** quite literally!!

HBC: Omnihash Bullion Coins

1. HBC: Omnihash Bullion Coins (Physical Gold/Silver),
2. Trispecies Monetary System: Bullion, Fiat, Crytocurrencies
3. Liberalism (Metanarchy) vs. “Omnipotent Government”



Hashcode

```
hongwu@hongwu-Latitude-5480:/var/www/dmeta/oxm/auth$ cat Graph/hg/FtyMvMgeoA\=\= ;echo
{"a":"1","t":"2025-10-16 09:04:43.683200","n":"Adam in MY wish to send MYR 100k to Donald in TH.","s
n":"1","job":"payment","n2":"variables","MYR":"currency","amount":"100k","sender":"Adam","recipient"
:"Donald"}
```

Hash of JSON is FtyMvMgeoA==

```
hongwu@hongwu-Latitude-5480:/var/www/html/oxw/auth/Graph/dmeta$ \
> cat H-xchGCVBg\=\=/BtiFODs5CQ\=\= ;echo
["2025-10-18T00:10:43.684+0000","reply_to","H-xchGCVBg==","G5bClrzsVg==",
"Ge6NiA5cLw==","Graph\/hg\/FtyMvMgeoA=="]
```

User Adam H-xchGCVBg== sends FtyMvMgeoA== to User Donald G5bClrzsVg==

```
hongwu@hongwu-Latitude-5480:/var/www/html/oxw/auth/Graph/dmeta$ \
> cat H-xchGCVBg\=\=/CxenfQAHxw== ;echo
["2025-10-18T00:58:01.196+0000","reply_to","G5bClrzsVg==","H-xchGCVBg==",
"BtiFODs5CQ==","ACCEPT chris agent apk:"]
```

User Donald replied ACCEPT and appointed User Chris as his agent.

a. Adam in MY (Malaysia) wish to send MYR 100k to Donald in TH (Thailand).
b. Adam sends message to Brian (Agent in MY).
c. Brian shows gold 20g worth MYR 10k via live camera feed and weighing machine, generates sensor hash code.
d. Brian sends message to Christ (Agent in TH)
e. Adam sends Brian 10 batches of MYR 10k, in MYR, by cash or local bank transfer.
f. Brian sends Chris 10 batches of MYR 10k, in cryptocurrencies USDT etc.
g. Chris sends Donald 10 batches of THB 77.5k, by cash or local bank transfer.
h. Guarantors Edwin and Frank (*or a “chain” of Guarantors*) may provide their hashcodes to all the parties above, to be included in DJSON for verifying each steps.
i. All parties concerned “may” choose to disclose the transactions to local authorities.

Hash filesystem: new way of managing Model-View-Controller architecture

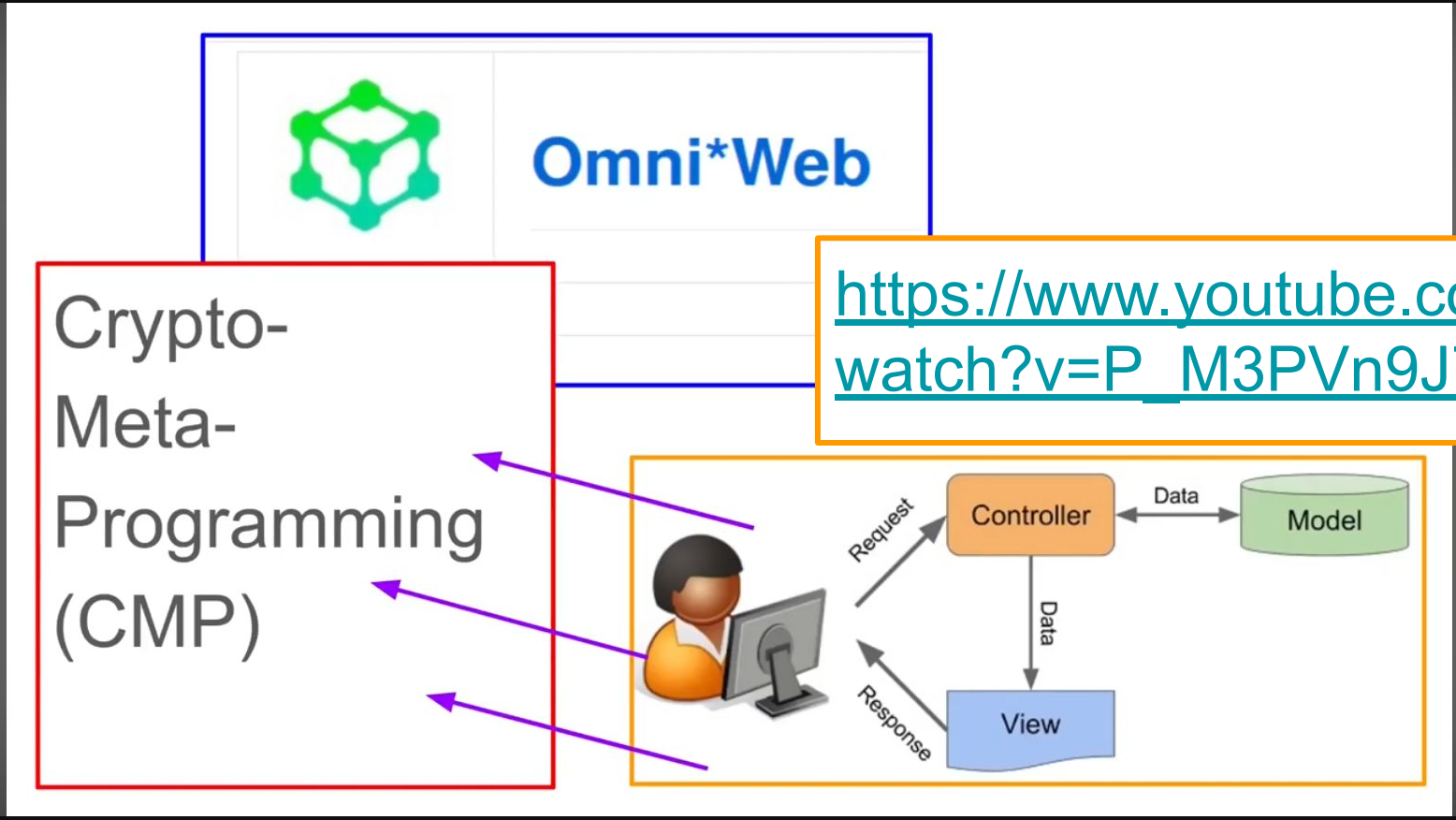
- security features of hash codes
- substitutes the functions managed by the conventional monolithic Model-View-Controller application programming interface,
- manages messages sent to and fro amongst the various components of model, view and controller.

Front end components in web applications and mobile applications

- send messages to back end components
- according to strict rules specified in the model-view-controller application programming interface (API),
- complete with essential network security features.

Crypto-MVC (Model-View-Controller)

- Enhancing MVC with Hash codes & related cryptographic algorithms



Omni*Web: Crypto-Metaprogramming (CMP) as alternative to Model-View-Controller; towards Metanarchy

MI4 Metaprogramming in Forth
56 subscribers

Hash filesystem substitutes such network security features

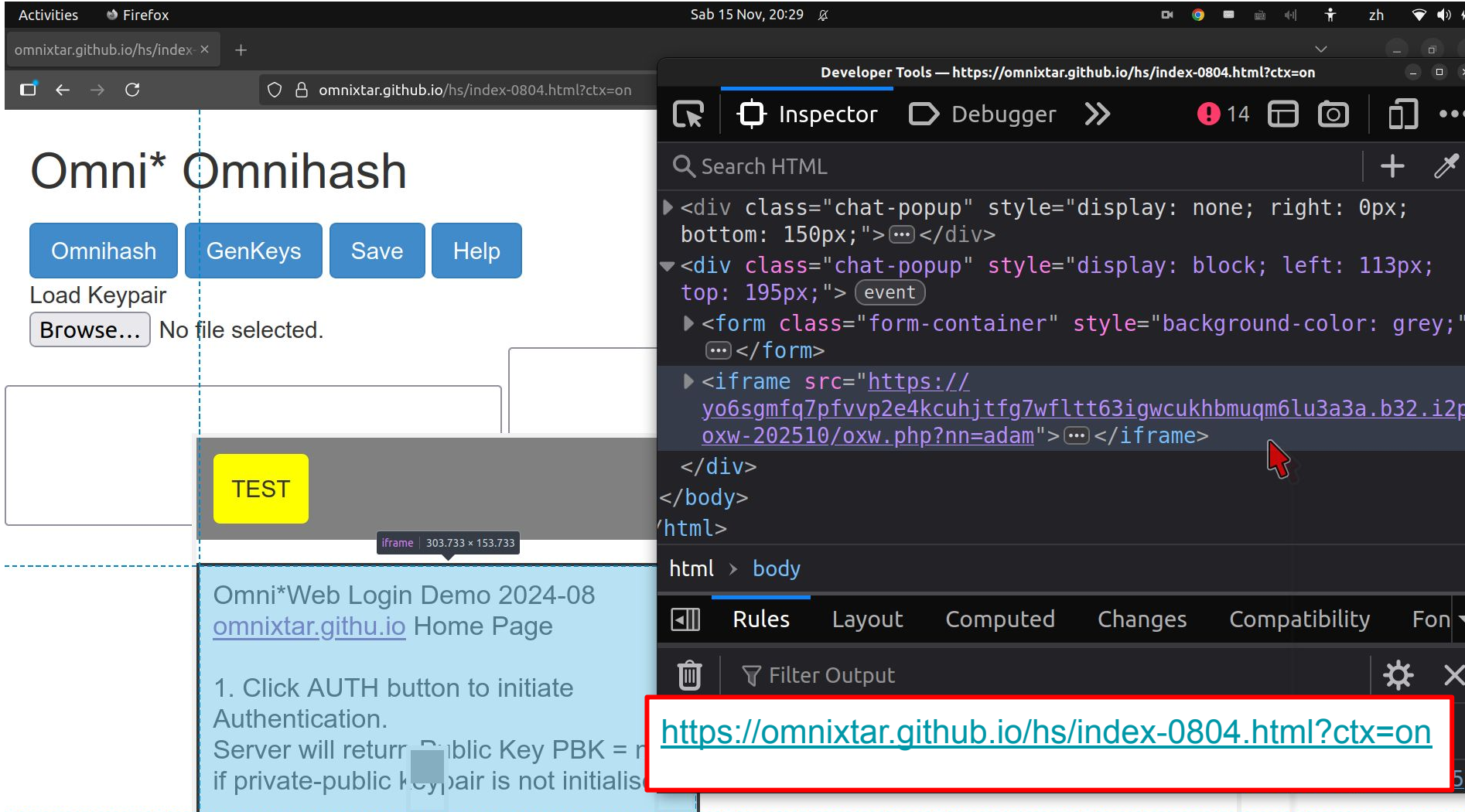
- required in model-view-controller architecture,
- simplifying the code structures,

Together with metaprogramming facilities

- provided by Phoscript metaprogramming shells,
- enabling one unified syntax derived from the FORTH programming language,
- to be hosted in the front end as well as back end environments,
- where the host programming language in the front end and back end environments may be different.

Application: SVFIG Membership Management, or for any company or organisation!

- Start page at omnixtar.github.io/* (1 account for unlimited number of pages)
- Use iframe to “tunnel” to I2P Invisible Internet Project or NGROK server (backend). Use Omnihash to manage security – novel, unprecedented, unlimit number of users, completely unlimited!


Activities
Firefox
Sab 15 Nov, 20:29
zh
omnixtar.github.io/hs/index-
omnixtar.github.io/hs/index-0804.html?ctx=on
Developer Tools — https://omnixtar.github.io/hs/index-0804.html?ctx=on
Inspector
Debugger
14
Search HTML

 event
<form class="form-container" style="background-color: grey;"
</form>
<iframe src="https://yo6sgmfq7pfvvp2e4kcuhjtfg7wfltt63igwcukhbmuqm6lu3a3a.b32.i2p/oxw-202510/oxw.php?nn=adam"> </iframe>

</body>
</html>
html › body
Rules
Layout
Computed
Changes
Compatibility
Fon
Filter Output
Omni* Omnihash
Omnihash
GenKeys
Save
Help
Load Keypair
Browse…
No file selected.
TEST
iframe | 303.733 × 153.733
Omni*Web Login Demo 2024-08
omnixtar.githu.io Home Page
1. Click AUTH button to initiate Authentication.
Server will return Public Key PBK =
if private-public keypair is not initialis
https://omnixtar.github.io/hs/index-0804.html?ctx=on

(E1) The hash filesystem described above enables a new way of managing Model-View-Controller architecture, as the security features of hash codes substitutes the functions managed by the conventional monolithic Model-View-Controller application programming interface, which usually manages messages sent to and fro amongst the various components of model, view and controller. In plain English, in conventional practice, front end components in web applications and mobile applications send messages to back end components according to strict rules specified in the model-view-controller application programming interface, complete with essential network security features. Hash filesystem substitutes such network security features required in model-view-controller architecture, thus simplifying the code structures, together with metaprogramming facilities provided by Phoscript metaprogramming shells, enabling one unified syntax derived from the FORTH programming language, to be hosted in the front end as well as back end environments, where the host programming language in the front end and back end environments may be different.

Phase I of 3X architecture

Phoscript metaprogramming shell: FORTH like syntax

- hosted on a host programming language, such as JavaScript, PHP, Python, Java, C++ and so on,
- stack machine loop which is only around 20 lines of JavaScript of equivalent
- implemented in web browsers, mobile environments etc. as a shell function.
- Therefore it still requires conventional operating system support with web servers and related infrastructure.
- Phase II & III: remove conventional operating system infrastructure.
- very lightweight: PHP, oldest and most comprehensive implementation –
- around 7000 lines (including comments and blank lines)
- around 200 kilobytes unzipped, around 38 kilobytes in tar gzipped format.
- Many of the functions can be removed in optimised implementation as the figures reported here include plenty of test and unused functions.

Phase I of 3X architecture

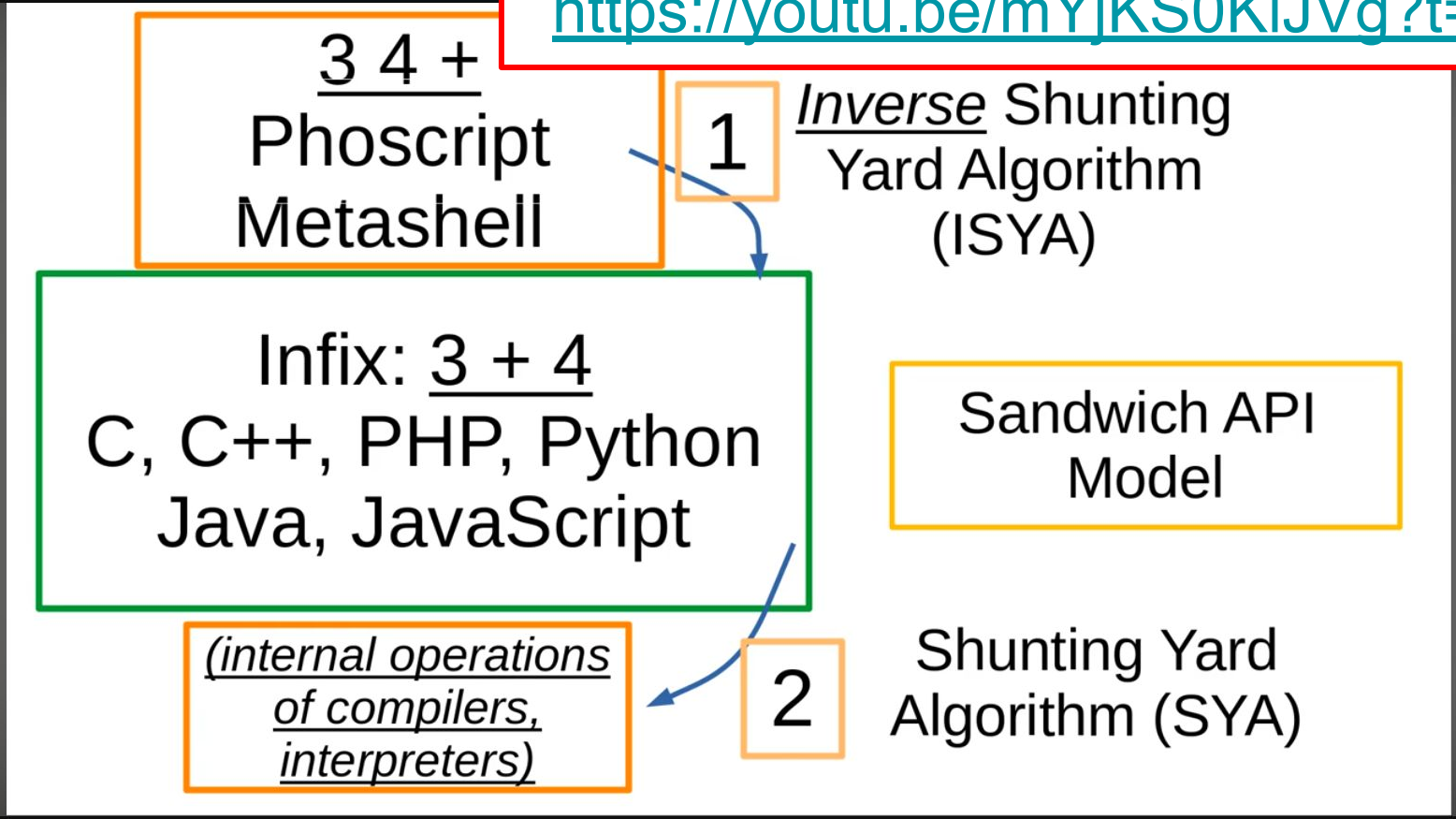
Phoscript metaprogramming shell: FORTH like syntax

- in development since 2017 and the hash filesystem was introduced around 2020.
- Only until recently in 2025 we realised that the hash filesystem is essentially a database system without a database server, and hence can be used by all FORTH implementations,

3X architecture: database-less, domain-less and kernel-less.

- the ultimate post-blockchain, light-weight, card size computers plus kilobyte virtual machines that can
- turn almost every electronic device into a FORTH computer node, and
- connected using decentralised hash codes, to realise
- FORTH version of Sun Microsystem vision of “the network is the computer”.

https://youtu.be/mYjKS0KiJVg?t=455



Bidirectional Shunting Yard Algorithm (BISYA) and Sandwich API Model: Unifying Programming Languages

https://github.com/omnixtar/omnixtar.github.io/blob/main/js/omni.js

Code Issues 3 Pull requests Actions Projects Wiki Security 1 ...

main omnixtar.github.io / js / omni.js Go to file t ...

omnixtar f_path js/omni.js 0a83e06 · 3 weeks ago

279 lines (250 loc) · 9.17 KB

Code Blame Raw

```
1  function include(url) {
2    var s = document.createElement("script");
3    s.setAttribute("type", "text/javascript");
4    s.setAttribute("src", url);
5    document.body.appendChild(s);
6  }
7
8  // 2025-10-24
```

Code Blame Raw

```
71
72  var phosinit=function(){
73  var Phos=function(){
74      // var S=[] // this is local, not accessible outside
75      var $ = this // macro; 20250804 use var to localise, $ is used by jquery
76      this.S=[]
77      var S=this.S // still need var S for local code access
78      S[0]={}
79      var S0=S[0]
80      var $CDW = {}
81      S[0].$CDW = $CDW;
82      S0.skip = 0;
83      S0.CDW = [];
84      S0.dlb = {};
85      var FGLA = function($WA) {
86      // arguments[0].split(' ').map(e=>{
87      var c_cdw=false; var i=0, ic, W=$WA;
88      $WA.map(e=>{ // WORD ARRAY
89          console.log(i, e);
90          var $v=e, $vk=i; $l=$v.length;
```

Code Blame Raw

```
72  var phosinit=function(){
85      var FGLA = function($WA) {
88      $WA.map(e=>{ // WORD ARRAY
89            console.log(i, e);
90            var $v=e, $vk=i; $l=$v.length;
91            if (!c_cdw && $v==':') { // COLON DEFINITION WORD
92               c_cdw=true;
93               console.log('  CDW start ', W[i+1])
94               ic = i+2; // start index of CDW
95               CDN=W[i+1]
96               $CDW[CDN]=[]
97            }
98            if (c_cdw) {
99              if ($v==';') {
100               console.log('  end CDW', $v);
101               c_cdw=false;
102             }
103             console.log('  in  CDW', $v);
104             if (i>=ic) $CDW[CDN].push($v)
```

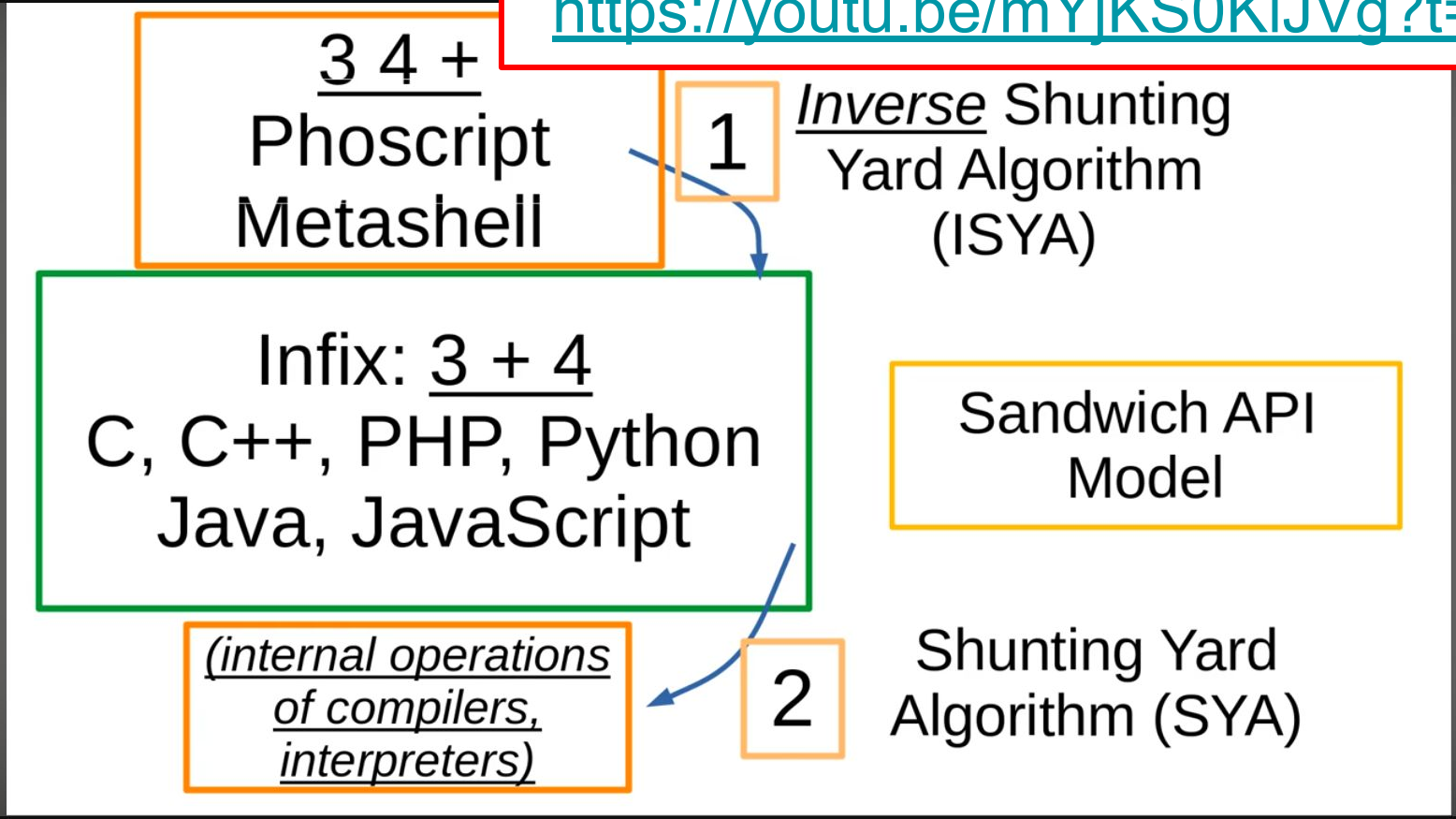
```
72    var phosinit=function(){
85        var FGLA = function($WA) {
112               else if (in_array($v, array_keys($CDW))) {
113                   var $WA = $CDW[$v];
114                   if (end($WA) == ';') array_pop($WA); // remove ; in definition before execution
115                   S0.CDW.push([ $v, $vk, {} ]); // console.log(1176, 'before FGLA', JSON.stringify(S0.CDW
116                   FGLA($WA); S0.CDW.pop(); // console.log(1183, 'after FGLA', JSON.stringify(S0.CDW));
117                   S0.cda = end(S0.CDW);
118               }
119               else if ($v[$l - 1] == ':') { // colon suffix word after symbol else : will fail
120                 var $fn = $v.substr(0, $l-1);
121                 if (typeof eval("f_"+$fn)!=="undefined") // console.log('is func', $v, typeof eval("f_"+$
122                   eval("f_"+$fn+"()")
123               } else s.push(e)
124             }
125             i++
126           }
127           )
128       }
```

Phoscript-Linux/C-FORTH Sandwich Model

1. Minimal VM: Linux Java I2P Apache php
2. Sandwich Model (user space programs):
   a. Top: P2C (Phos to C) P2J (Phos to Java) P2PHP
      [start: 0, end: replace host language functions]
   b. Middle: C-lib, J-lib (Java), PHP-lib in .o (object);
      [start: 100%, end: replaced by Top & Bottom]
   c. Bottom: need equivalent F-lib (FORTH)
      [start: 0, end: replace Middle layer host language
3. Replace Kernel? (Need experience from Phase 2)
   a. Use P2C to replace C code with Phoscript code.
   b. Write FORTH code to replace low level C code.

Phase II of 3X architecture will begin with the current state of Phoscript metaprogramming shell,
- end goals of eliminating user space services, by substituting them with FORTH based libraries,
- such as web server and I2P Invisible Internet Project or equivalent services, that is
- to achieve the domain-less goal.

https://youtu.be/mYjKS0KiJVg?t=455



Bidirectional Shunting Yard Algorithm (BISYA) and Sandwich API Model: Unifying Programming Languages

```
f('DSC dlb:') // FORTH (1968) words Phoscript 符式 黃大一 in chris dlb, create DSC dlb
f('textarea ce: DSC app:')
f('dsc_cmd cmd DSC 1 dom:')
f('dsc_sign 123 DSC 5 dom:')

f('textarea ce: DSC app:')
f('dsc_ecr Encrypted_Message DSC 6 dom:')
f('textarea ce: DSC app:')
f('dsc_aor Any_other_remarks DSC 7 dom:')
f('textarea ce: DSC app:')
f('dsc_misc Misc DSC 8 dom:')

// var S0=S[0]
var S0=window.M.S[0];
S0.dlb.DSC[2].style.right='0px'
S0.dlb.DSC[2].style.bottom='150px'

// chris browser 2021-2-20 2255
f(': do_sign awa: im_pbk ss: mk_sign ;')
f(': im_pbk . dup: js: jd: msg ix: awa: impbk: ss: ;')
f(': mk_sign awa: sign_kp: rsa_sign ;')
f(': ecr_sig 5 pick: s2ab: 1 pick: 1 pick: catab: 4 pick: 1 pick: awa: ecr_t: ab_scj ;') // : ab_
scj ab2s: btoa: scj: ;') // sig done
f(': rsa_sign saveks: privateKey ix: dsc_sign geid: value: awa: sign: do_scj ;')
```

Crypto-Metaprogramming (CMP)

Bitcoin – makes FORTH *the most popular programming language* by number of mining nodes(??), and variants of blockchains.
- and *the most VALUABLE programming language*.

But tied to miners and mining nodes – how to liberate the wealth and knowledge?

- 3X Architecture: No DNS, No Database, No Kernel.

Knowledge of Metaprogramming: revolutionise STEM (Science, Technology, Engineering, Mathematics) education, foundation of Metanarchy?

- Bitcoin uses FORTH like syntax to verify the hash of public key.

1. Extend hash of public key as user identifier in decentralised network and explore various properties of hash codes, (important for subsequent steps)
2. Omnihash, literally meaning a hash code for representing all types of digital assets, and DJSON decentralized JSON, a special type of JSON string containing at least one Omnihash
3. Omnihash as the basis of hash filesystem, which becomes the foundation of a database system without database server.
4. 3X architecture, namely database-less, domain-less and kernel-less
5. Phoscript and Omnihash as used in Web and mobile applications,
   Phoscript metaprogramming shell: ported to any host programming language, including front end as well as back end environments
   Phase II of 3X architecture: Sandwich model of code substitution
   substitute user space services such as Web server and I2P routers with FORTH or Phoscript exclusive code base.

3X architecture: database-less, domain-less and kernel-less

6. Phase III of 3X architecture: applied to the Linux kernel
   - FORTH Phoscript exclusive code base,
   - the ultimate post-blockchain, light-weight, card size computers plus kilobyte virtual machines
   - turn almost every electronic device into a FORTH computer node
   - connected using decentralised hash codes (hash filesystem)
   - to realise FORTH version of Sun Microsystem vision of "the network is the computer": Decentralised AI, nodes owned by users, NOT MMAGA!!

7. MMAGA (Microsoft,Meta,Amazon,Google,Apple): USD 1.8Trillion (2024 revenues)
   - Omni*Web 3X Architecture: 0.1% of MMAGA revenues by 2030?
   - 0.1% of Bitcoin values for 0.1% of world population by 2035?
   - Metanarchy (Decentralised Autonomous Organisations) – better government Decentralised Global Governance based on Transactions in Metaverse (Outside United States of America & China)?