Πανεπιστήμιο Δυτικής Αττικής

Σχολή Μηχανικών

Τμήμα Μηχανικών Βιομηχανικής Σχεδίασης και Παραγωγής

**<u>Θέμα:</u> Τεχνολογίες Blockchain και αλγόριθμοι consensus**


**ΣΠΥΡΟΥ-ΜΑΝΤΑΛ ΚΟΥΝΤΡΕΤ (Α.Μ. 71446173)**

ΑΘΗΝΑ ΙΑΝΟΥΑΡΙΟΣ 2020

## ΠΕΡΙΛΗΨΗ

Η τεχνολογία Blockchain έχει διαδοθεί και έχει εξαπλωθεί αρκετά από την εισαγωγή της στην ζωή μέσω της κυκλοφορίας του πρώτου κρυπτονομίσματος που είναι το Bitcoin. Με το πέρασμα των χρόνων αυτή η τεχνολογία έχει ωριμάσει και έχει αρχίσει να βρίσκει εφαρμογές και σε άλλους τομείς της πληροφορικής και της τεχνολογίας Στην παρούσα εργασία θα αναλυθούν εκτενώς οι εναλλακτικές επιλογές για την υλοποίηση δικτύου Blockchain και θα τις συγκρίνουμε ώστε να βρεθεί η βέλτιστη επιλογή.



## ABSTRACT

Blockchain Technology became widespread after being introduced by the release of the first cryptocurrency, which is Bitcoin. By the passage of years, this technology has matured and it started to have applications on other sectors of information technology and technology in general. On this thesis we will analyze the alternative options for implementing a Blockchain network and compare them to find the best choice.

## ΠΡΟΛΟΓΟΣ

Η τεχνολογία Blockchain είναι μία σχετικά νέα και αναπτυσσόμενη τεχνολογία η οποία μπορεί να προσφέρει πολλές προοπτικές στον τρόπο με τον οποίο αντιμετωπίζουμε την καθημερινότητα μας και να αλλάξει τον τρόπο με τον οποίο δικτυώνουμε τις ψηφιακές και οικιακές μας συσκευές. Ο συνδυασμός της παραπάνω τεχνολογίας με την τεχνολογία του διαδικτύου των αντικειμένων μπορεί να προσφέρει πολλές βελτιώσεις στον τρόπο με τον οποίο χρησιμοποιούμε τις συσκευές μας, επίσης θα μπορούσε να προσφέρει πολλές ευελιξίες στον κλάδο των επιστημονικών οργάνων πεδίου-πχ μετεωρολογικοί σταθμοί, γεωδαιτικοί σταθμοί, κλπ-, προσφέροντας ευέλικτη δικτύωση και προσβασιμότητα των οργάνων από το διαδίκτυο με βέλτιστο τρόπο και μία βελτίωση στον τρόπο με τον οποίο οι επιστήμονες θα μπορούν να λαμβάνουν μετρήσεις από τα όργανα τους τα οποία βρίσκονται σε δυσπρόσιτες περιοχές ή σε περιοχές με έντονα καιρικά φαινόμενα.

Η δομή της εργασίας αποτελείται από την θεωρητική ανάλυση βασικών τμημάτων και εννοιών της τεχνολογίας Blockchain και ακολουθείται από πρακτικά παραδείγματα τα οποία αποτελούν ένα δείγμα για τις δυνατότητες που μπορούν να προσφέρουν οι τεχνολογίες και οι πλατφόρμες που αξιοποιούν την τεχνολογία Blockchain. Τέλος παρουσιάζουμε τα συμπεράσματα μας για την βέλτιστη πλατφόρμα ανάπτυξης εφαρμογών για Blockchain η οποία μπορεί να καλύψει τις ανάγκες και τα σενάρια χρήσης για εφαρμογές του διαδικτύου των αντικειμένων.

## ΕΥΧΑΡΙΣΤΙΕΣ

Αρχικά θα ήθελα να ευχαριστήσω την οικογένεια μου και τους στενούς μου φίλους, οι οποίοι με στήριξαν όλα αυτά τα χρόνια και με βοήθησαν να εκπληρώσω τους στόχους μου.

Επίσης θα ήθελα να ευχαριστήσω θερμά την κ. Ελένη-Αικατερίνη Λελίγκου και τον κ. Κόγια Δημήτριο οι οποίοι μου προσέφεραν πολύτιμη καθοδήγηση και επίσημανσεις για την ορθή πραγματοποίηση της παρούσας διπλωματικής εργασίας.

Τέλος θα ήθελα να ευχαριστήσω τον κ. Στέφανο Δέτση, ο οποίος μέσα σε αυτές τις οικονομικές συγκιρίες μου προσέφερε μία θέση εργασίας πάνω στο αντικείμενο που έχω σπουδάσει και ήθελα να ασχοληθώ, δίνοντας μου την ευκαιρία να ασχοληθώ με αυτό που σπούδασα σε πραγματικές συνθήκες.

# ΠΕΡΙΕΧΟΜΕΝΑ

# **ΚΕΦΑΛΑΙΟ 1:** ΕΙΣΑΓΩΓΗ

## 1.1 Σκοπός και στόχοι

Σκοπός αυτής της εργασίας είναι η μελέτη και η ανάλυση βασικών τμημάτων ενός συστήματος Blockchain που αποτελεί μία σχετικά καινούρια τεχνολογία στον τομέα της πιστοποίησης και ταυτοποίησης κατανεμημένων πληροφοριακών συστημάτων.

Επιμέρους στόχοι αποτελούν οι εξής:

- Έρευνα και ανάλυση των υπαρχουσών πλατφόρμων ανάπτυξης εφαρμογών Blockchain για την εύρεση της πλατφόρμας που προσφέρει πληρότητα εργαλείων για την ανάπτυξη εφαρμογών.
- Ανάλυση των πλεονεκτημάτων και μειονεκτημάτων της κάθε πλατφόρμας.
- Ανάλυση μελλοντικών προοπτικών της τεχνολογίας αυτής.
- Ανάλυση εφαρμογών της κάθε πλατφόρμας και αυτής της τεχνολογίας γενικότερα.
- Σύγκριση και ανάλυση των βασικότερων μηχανισμών όσον αφορά την ασφάλεια και την αποδοτικότητα τους.
- Υλοποίηση λογισμικού για να παρουσιαστούν οι δυνατότητες αυτής της τεχνολογίας.

## 1.2 Βασικές Έννοιες

**Blockchain:** αποτελεί μία συγκεκριμένη δομή δεδομένων η οποία παρέχει δικλείδες ασφαλείας με κρυπτογραφικούς αλγορίθμους για την ακεραιότητα των υπάρχοντων δεδομένων.

**Κρυπτονόμισμα:** αποτελεί ένα είδος ψηφιακού νομίσματος, σαν εναλλακτικός τρόπος πληρωμής αντί για τα συμβατικά νομίσματα.

**Αλγόριθμος Consensus:** αποτελεί μηχανισμό για την δημιουργία ομοφωνίας σε δίκτυο από κατανεμημένα συστήματα το οποίο δεν παρέχει κάποιο συμβατικό σύστημα ταυτοποίησης.

**Ethereum:** Το Ethereum αποτελεί πλατφόρμα ανάπτυξης εφαρμογών οι οποίες αξιοποιούν την τελευταία τεχνολογία τύπου Blockchain και παρέχει όλα τα απαραίτητα εργαλεία για προγραμματιστές που θέλουν να αναπτύξουν τις εφαρμογές τους.

**Hyperledger Fabric:** Αποτελεί πλατφόρμα ανάπτυξης εφαρμογών Blockchain.

**Smart Contract:** Μία μορφή πλήρους ψηφιακού συμβολαίου, το οποίο αποτελεί πρόγραμμα. Κάθε τμήμα του και εκτελείται ανάλογα τμήμα του με βάση τις ρήτρες του συμβολαίου που έχουν ενεργοποιηθεί.

**API:** Προγραμματιστική διεπαφή: μία «είσοδος» σε πόρους και δυνατότητες μίας πλατφόρμας για την αξιοποίηση τους από άλλες εφαρμογές.

**DDoS:** Επίθεση από ένα δίκτυο υπολογιστών σε έναν εξυπηρετητή με σκοπό την εμπλοκή της υπηρεσίας που προσφέρεται.

**Botnet:** Δίκτυο μολυσμένων υπολογιστών το οποίο χρησιμοποιείται για κυβερνοεπιθέσεις.

**Byzantine Fault Tolerant Algorithms:** Κατηγορία αλγορίθμων οι οποίοι έχουν σκοπό να εξαλείψουν την πιθανότητα καταστροφικού σφάλματος σε περίπτωση που ένας ή περισσότεροι από τους κόμβους που συμμετέχουν στο σύστημα παρουσιάζει σφάλματα.

**IOTA:** Πλατφόρμα η οποία υλοποιεί τεχνολογίες Blockchain για την διαχείριση συσκευών IoT.

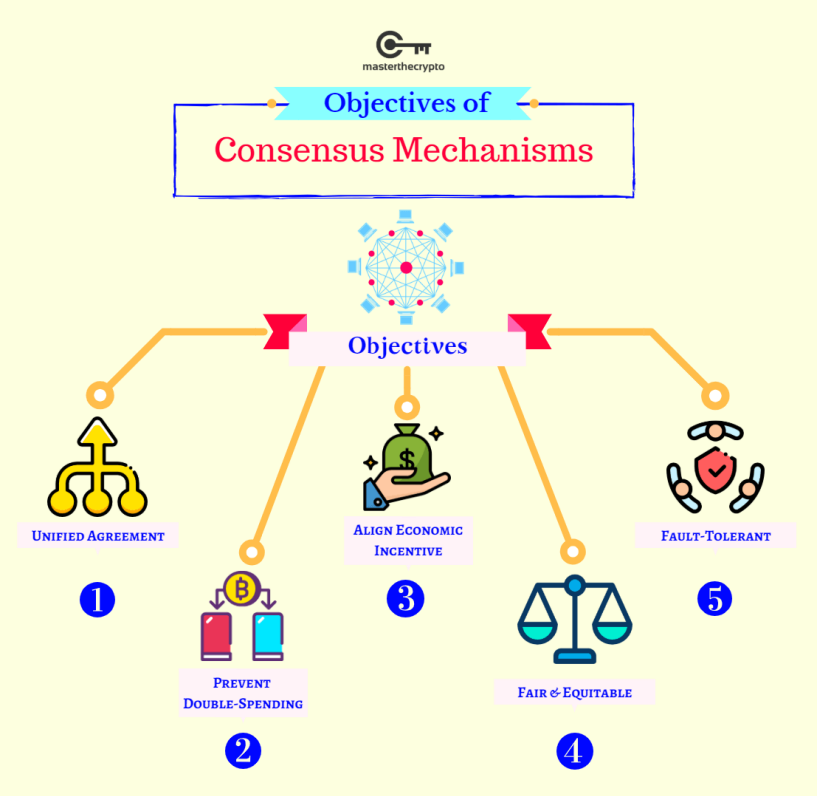
## 1.3 Σχεδιάγραμμα εργασίας

- Στο κεφάλαιο 2 θα αναλύσουμε και θα συγκρίνουμε τους βασικότερους αλγορίθμους consensus που υπάρχουν και μερικές εφαρμογές τους στην τεχνολογία Blockchain.
- Στο κεφάλαιο 3 θα ασχοληθούμε με την πλατφόρμα Ethereum. Θα την συγκρίνουμε με τον βασικό ανταγωνιστή του που είναι η πλατφόρμα hyperledger fabric και τέλος θα δούμε ένα παράδειγμα για το πώς μπορεί κάποιος να προσθέσει δικό του αλγόριθμο consensus.
- Στο κεφάλαιο 4 θα αναλύσουμε την τελευταία εξέλιξη στην τεχνολογία του Blockchain που είναι τα Smart Contracts, θα αναφέρουμε τις εφαρμογές που έχουν σε πραγματικό και ψηφιακό κόσμο και τέλος θα παρουσιάσουμε παράδειγμα για την ανάπτυξη λογισμικού ψηφιακής ψηφοφορίας.
- Στον επίλογο θα αναλύσουμε τα συμπεράσματα.

# **ΚΕΦΑΛΑΙΟ 2:** ΑΛΓΟΡΙΘΜΟΙ CONSENSUS ΚΑΙ ΟΙ ΕΦΑΡΜΟΓΕΣ ΤΟΥΣ ΣΤΟ BLOCKCHAIN

## 2.1 Τι είναι αλγόριθμος consensus;

Αλγόριθμος consensus είναι το πρωτόκολλο με το οποίο οι κόμβοι που συμμετέχουν σε ένα Blockchain ή σε οποιαδήποτε άλλη τεχνολογία DLT αποφασίζουν τις παραμέτρους με τις οποίες θα δημιουργούνται και θα επικυρώνονται τα blocks. Αυτά τα πρωτόκολλα διαμορφώνουν έναν πυρήνα ο οποίος προσφέρει ασφάλεια, εμπιστοσύνη και δικαιοσύνη σε ένα trustless δίκτυο -δηλαδή δίκτυα που η πρόσβαση σε αυτά πραγματοποιείται χωρίς κάποια διαδικασία πιστοποίησης όπως γίνεται στις «παραδοσιακές» εφαρμογές, δηλαδή login ή σε κρυπτογραφικά συστήματα τα ζεύγη δημοσίων-ιδιωτικών κλειδιών- [24]. Επειδή τα Blockchain έχουν μία φιλοσοφία δικτύωσης peer-to-peer πρέπει να υπάρχει τρόπος να διαφυλάσσεται η εύρυθμη λειτουργία και η εγκυρότητα των δεδομένων που υπάρχουν σε αυτό, όπως φαίνεται στην εικόνα 1. Για τους παραπάνω λόγους θεσπίστηκαν και υλοποιήθηκαν αλγόριθμοι consensus σαν απαραίτητο συνοδευτικό για οποιαδήποτε εφαρμογή της τεχνολογίας Blockchain. Αλγόριθμοι consensus όμως χρησιμοποιούνται και σε άλλες μορφές κατανεμημένων συστημάτων [24] και σε καίρια συστήματα των αεροσκαφών Boeing 777 [2].

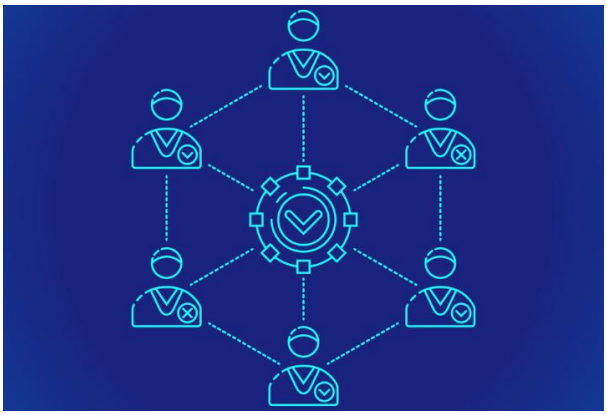



*Εικόνα 1: Γενικοί σκοποί που επιτυγχάνουν οι αλγόριθμοι consensus[14]*

## 2.2 Τύποι αλγορίθμων consensus

Σε αυτή την ενότητα θα αναφέρουμε τους τύπους αλγορίθμων consensus που βρίσκουν εφαρμογή σε τεχνολογίες blockchain και γενικότερα σε τεχνολογίες DLT. Επειδή υπάρχει μία πληθώρα αλγορίθμων consensus θα αναφέρουμε τους πιο βασικότερους τύπους και θα εξετάσουμε τα πλεονεκτήματα και τα μειονεκτήματα του κάθε τύπου. Όλοι οι τύποι των αλγορίθμων consensus, όπως φαίνεται στην εικόνα 2, έχουν μοναδικό σκοπό την εύρυθμη

λειτουργία του δικτύου Blockchain καθώς επίσης και την δίκαιη κατανομή δικαιωμάτων σε κάθε κόμβο που συμμετέχει σε ένα Blockchain και τέλος, να δημιουργήσει τις δικλίδες ασφαλείας που οι οποίες είναι απαραίτητες για να υπάρχει σωστή λειτουργία και έγκυρα δεδομένα στο δίκτυο [1].



*Εικόνα 2: Ένας αλγόριθμος ή πρωτόκολλο consensus δημιουργεί συναίνεση στο Blockchain[27]*

Σε εφαρμογές Blockchain μέχρι την σημερινή εποχή υπάρχουν και είναι υλοποιήσιμοι οι παρακάτω αλγόριθμοι consensus [7] [12] [14] [16] [29], όπως φαίνεται συνοπτικά και στην εικόνα 3:

- Proof of Work - PoW
    - Ο παραπάνω αλγόριθμος είναι ο πρώτος που υλοποιήθηκε για την παροχή ασφάλειας σε Blockchain.
    - Ο κύριος μηχανισμός του είναι η επίλυση κρυπογραφικών γρίφων [29].
    - Βασικό πλεονέκτημα του είναι ότι υπάρχει πληθώρα API και tutorial που καθιστούν αρκετά εύκολη την υλοποίηση [18].
    - Κύριο μειονέκτημα του είναι η σπατάλη επεξεργαστικών πόρων για μία διαδικασία χωρίς να προσφέρει κάποιον παραγωγικό υπολογισμό και ο λόγος που χρησιμοποιείται είναι γιατί δημιουργεί σημαντική χρονοκαθυστέρηση(delay) για να αντιληφθεί το σύστημα ύποπτες κινήσεις[ 29].
- Proof of Stake - PoS
    - Ο παραπάνω αλγόριθμος δίνει μία λύση στα προβλήματα που παρουσιάζει ο PoW[ 29].
    - Ο κύριος μηχανισμός είναι η εξέταση της παλαιότητας των νομισμάτων που κατέχει ο κάθε κόμβος και ένας τυχαίος παράγοντας [29].
    - Βασικό πλεονέκτημα του είναι η ευκολία των κόμβων να εισάγουν συναλλαγές σε συνδυασμό με κάποια μέτρα ασφαλείας [29] τα οποία θα αναλυθούν στην επόμενη ενότητα.

- Κύριο μειονέκτημα του είναι ότι αυτός ο αλγόριθμος είναι ευάλωττος σε επιθέσεις από botnets, όπου μπορούν να δημιουργήσουν «κλάδο» με παραποιημένα δεδομένα [29].

- Delegated Proof of Stake
  - Αυτός ο αλγόριθμος σχεδιάστηκε για να αντιμετωπίσει τα προβλήματα που παρουσιάζει ο PoS όσον αφορά τα botnets [29].
  - Κύριος μηχανισμός είναι η ψηφοφορία «αντιπροσώπων» οι οποίοι θα δημιουργούν και θα επικυρώνουν τις συναλλαγές [29].
  - Βασικό πλεονέκτημα του είναι ότι διατηρεί τα θετικά του PoS και εξαλείφει την αδυναμία του προηγούμενου αλγορίθμου στα botnets [29].
  - Το κύριο μειονέκτημα του είναι οι επιθέσεις DDoS στους κόμβους που έχουν επιλεγεί για να εισάγουν συναλλαγές [29].

- Proof of Weight
  - Αυτός ο αλγόριθμος αποτελεί την γενικευμένη μορφή του αλγορίθμου PoS [7].
  - Κύριος μηχανισμός είναι ένας σταθμιστικός παράγοντας ο οποίος δίνει ανάλογη βαρύτητα στην ψήφο του κάθε κόμβου [7].
  - Βασικό πλεονέκτημα του είναι ότι δίνει αρκετές επιλογές στον developer για τον παράγοντα ο οποίος θα δημιουργεί συναίνεση στο δίκτυο [7].
  - Κύριο μειονέκτημα του είναι το ίδιο που υπάρχει και στον αλγόριθμο PoS, δηλαδή είναι ευάλωττο σε κακόβουλα botnets [7].

- Proof of Authority
  - Αυτός ο αλγόριθμος consensus έχει συγκεντρωτικό χαρακτήρα[29] και συνιστάται σε private blockchains [9].
  - Ο κύριως μηχανισμός του είναι η παροχή εξουσίας από τον γεννήτριο κόμβο(ο οποίος εισάγει στο σύστημα την πρώτη συναλλαγή, γνωστή και ώς genesis block) και μόνο αυτοί οι κόμβοι έχουν δυνατότητα δημιουργίας και επικύρωσης συναλλαγών [29].
  - Βασικό πλεονέκτημα του είναι η παροχή δυνατότητα κατασκευής private blockchain [9].
  - Το κύριο μειονέκτημα του είναι ότι αν κάποιος από τους εξουσιοδοτημένους κόμβους μολυνθεί με κακόβουλο λογισμικό, τότε «σπάει» και η ασφάλεια του blockchain [29].

- Practical Byzantine Fault Tolerance
  - Είναι μία κατηγορία αλγορίθμων οι οποίοι βασίζονται στο πρόβλημα των βυζαντινών ταγματαρχών [14] [29].
  - Κύριος μηχανισμός του είναι η ψηφοφορία κόμβων οι οποίοι θα έχουν την δυνατότητα να δημιουργήσουν και να επικυρώσουν συναλλαγές, μπορεί όμως

να τους αφαιρεθεί αυτή η δυνατότητα αν γίνουν αντιληπτοί ότι δρουν κακόβουλα [14].

- Βασικό πλεονέκτημα του είναι η βέλτιστη λειτουργία του blockchain με δυνατότητες επεκτασιμότητας(scalability) [14].

- Το κύριο μειονέκτημα του είναι το ίδιο με το Delegated Proof of Stake, δηλαδή είναι ευάλωττο σε επιθέσεις οι οποίες προέρχονται από ένα εκτεταμένο δίκτυο μολυσμένων υπολογιστών(DDoS) [14].

- Directed Acyclic Graphs
  - Το παραπάνω δεν αποτελεί μόνο αλγόριθμο consensus, αποτελεί μία τεχνολογία DLT και αποτελεί εναλλακτική δομή αντί για Blockchain [12].

  - Ο κύριος μηχανισμός του είναι η άμεση επικύρωση δύο, ή και παραπάνω συναλλαγών με την είσοδο του κόμβου στο δίκτυο [12].

  - Βασικό πλεονέκτημα του είναι ο μεγάλος ρυθμός παραγωγής και επικύρωσης συναλλαγών [12].

  - Το κύριο μειονέκτημα του είναι η πολύπλοκη δομή δεδομένων που χρησιμοποιεί και επειδή ακόμα οι εφαρμογές του είναι ελάχιστες, δεν υπάρχει κάποια τυποποιημένη μορφή χρήσης του με αποτέλεσμα η υλοποίηση του να είναι δύσκολη και χρονοβόρα [12].

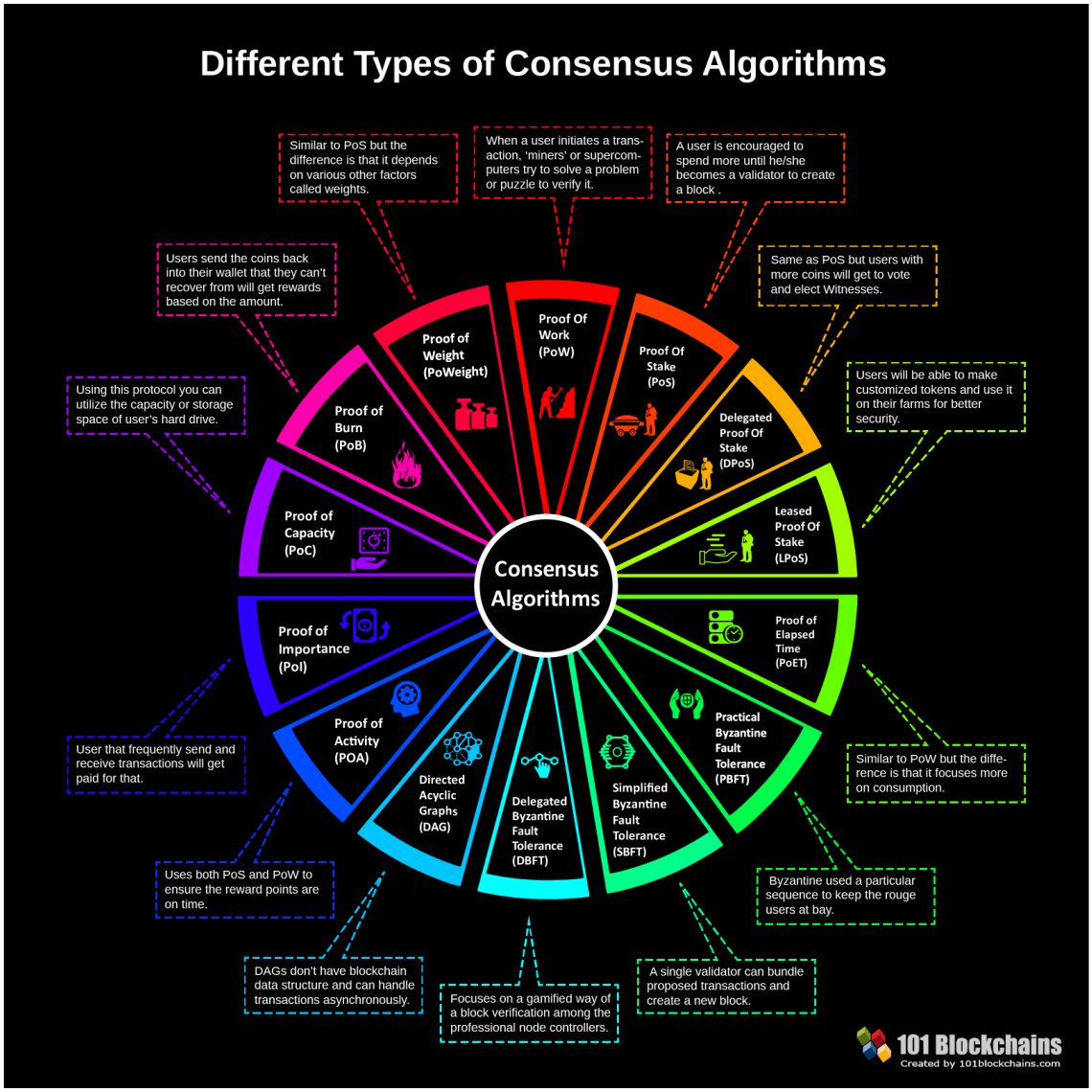
- Hashgraph
  - Αυτός ο αλγόριθμος αποτελεί μία αρκετά καλή υλοποίηση της τεχνολογίας DAG[16].

  - Ο κύριος μηχανισμός του είναι η διάδοση των έγκυρων συναλλαγών από το κάθε κόμβο στους γειτονικούς τους [16].

  - Βασικό πλεονέκτημα είναι η πολύ γρήγορη πραγματοποίηση των συναλλαγών, η ασφάλεια και η επεκτασιμότητα του [16].

  - Το κύριο μειονέκτημα του είναι ότι ο συγκεκριμένος αλγόριθμος αποτελεί πατέντα η οποία ανήκει στην Hedera και πρέπει να χρησιμοποιηθεί η πλατφόρμα Hedera Hashgraph για την αξιοποίηση του [16].

Λόγω του μεγάλου αριθμού των υπάρχοντων αλγορίθμων consensus, παραπάνω εξετάστικαν οι πιο σημαντικοί και αυτοί που χρησιμοποιούνται τακτικά για την υλοποίηση εφαρμογών Blockchain. Παρακάτω στην εικόνα 3 υπάρχουν αλγόριθμοι που δεν αναφέραμε παραπάνω οι οποίοι όμως έχουν λιγοστές εφαρμογές στην τωρινή χρονική περίοδο.




*Εικόνα 3: Τύποι αλγορίθμων consensus που υπάρχουν με συμπυκνωμένες πληροφορίες για το πώς λειτουργεί ο καθένας[1]*

## 2.3 Πως οι αλγόριθμοι consensus διασφαλίζουν την εύρυθμη λειτουργία των Blockchains

Σε αυτή την ενότητα θα αναφέρουμε τους ρόλους που παίζουν οι αλγόριθμοι consensus στην διατήρηση της ασφαλούς λειτουργίας του Blockchain και στην διασφάλιση της ακεραιότητας των δεδομένων που υπάρχουν σε αυτό. Οι αλγόριθμοι consensus αποκαλούνται και αλγόριθμοι ανοχής των βυζαντινών σφαλμάτων -Byzantine Fault Tolerant Algorithms- [18] ο οποίος είναι ένας όρος που έχει προκύψει από την τεχνολογία των κατανεμημένων συστημάτων στα οποία συμμετέχουν μη αξιόπιστοι κόμβοι, σκοπός τέτοιων αλγορίθμων δηλαδή είναι να δημιουργήσουν μία αξιοπιστία σε ένα δίκτυο με ξένους -και μη αξιόπιστους- κόμβους [14]. Στα Blockchains αυτοί οι αλγόριθμοι είναι απαραίτητοι διότι κατά βάση ένα Blockchain είναι μία κοινόχρηστη βάση δεδομένων η οποία μπορεί να αναπαραχθεί τοπικά στο κάθε κόμβο και να τροποποιηθεί με τέτοιο τρόπο που θα μπορούσε να δημιουργήσει διάφορα προβλήματα κατάχρησης -πχ το πρόβλημα της διπλής χρήσης στο bitcoin- [3] και όπως φαίνεται και στην εικόνα 4, εξυπηρετεί στην αποτροπή της κακόβουλης αξιοποίησης του

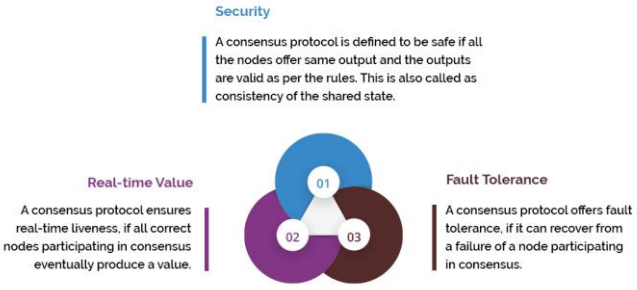
δικτύου. Για αυτό το λόγο υπάρχουν οι αλγόριθμοι consensus οι οποίοι με πλήρη αυτοματοποιημένο τρόπο αποφασίζουν:

Α) Ποιοί κόμβοι έχουν δικαίωμα να συμμετέχουν στο δίκτυο.

Β) Ποιοί κόμβοι έχουν δικαίωμα να δημιουργούν και να επικυρώνουν Blocks.

Γ) Ποιά Blocks είναι έγκυρα και άρα γίνονται δεκτά στο Blockchain και ποιά απορρίπτοναι.

Γενικότερα, αυτοί οι αλγόριθμοι αποφασίζουν πώς θα λειτουργεί το Blockchain και τι δικαιώματα θα παρέχει στο κάθε κόμβο, και με αυτό το τρόπο μπορεί να διασφαλιστεί η σωστή λειτουργία του Blockchain και η εγκυρότητα των δεδομένων που υπάρχουν σε αυτό.




*Εικόνα 4: Οι τρείς βασικοί άξονες των αλγορίθμων consensus[30]*

## 2.4 Μερικές εφαρμογές αλγορίθμων consensus σε Blockchain

Σε αυτή την ενότητα θα αναφέρουμε μερικές από τις πιο συχνές εφαρμογές των αλγορίθμων consensus πάνω στην τεχνολογία Blockchain και θα τις αναλύσουμε. Γενικότερα, ο κάθε τύπος αλγόριθμου consensus θεωρητικά μπορεί να εφαρμοστεί σε κάθε τύπο και εφαρμογή της τεχνολογίας Blockchain και γενικότερα σε DLTs. Πρακτικά όμως όπως φαίνεται και στην εικόνα 5 ο κάθε αλγόριθμος έχει περιορισμένες εφαρμογές ανάλογα το είδος της πλατφόρμας Blockchain και τι σημαίνει η κάθε συναλλαγή για αυτό. Στην τωρινή εποχή υπάρχουν εφαρμογές αυτής της τεχνολογίας οι οποίες έχουν κάποιο δικό τους «νόμισμα» αλλά οι συναλλαγές δεν αποτελούν τις «κλασσικές» χρηματικοπιστωτικές συναλλαγές, αλλά αποτελούν ανταλλαγές δεδομένων και γενικότερα διεργασίες που εκτελούν τα συστήματα όταν ένας κόμβος παρέχει το κατάλληλο ποσό για πχ την αναζήτηση συγκεκριμένων δεδομένων στο δίκτυο. Δηλαδή υπάρχουν εφαρμογές που το κρυπτονόμισμα αποτελεί μία αξία η οποία «μεταφράζεται» σε παροχή δικαιωμάτων μέσα στο σύστημα. Επίσης με την ευρεία χρήση της τεχνολογίας υπήρξαν και ανάγκες για κατασκευή ιδιωτικών ή περιορισμένης πρόσβασης Blockchains και εκεί θα δούμε ότι συγκεντρωτικοί αλγόριθμοι όπως το Proof of Authority είναι η βέλτιστη λύση. Οι βασικότερες εφαρμογές του κάθε αλγορίθμου consensus είναι οι παρακάτω:

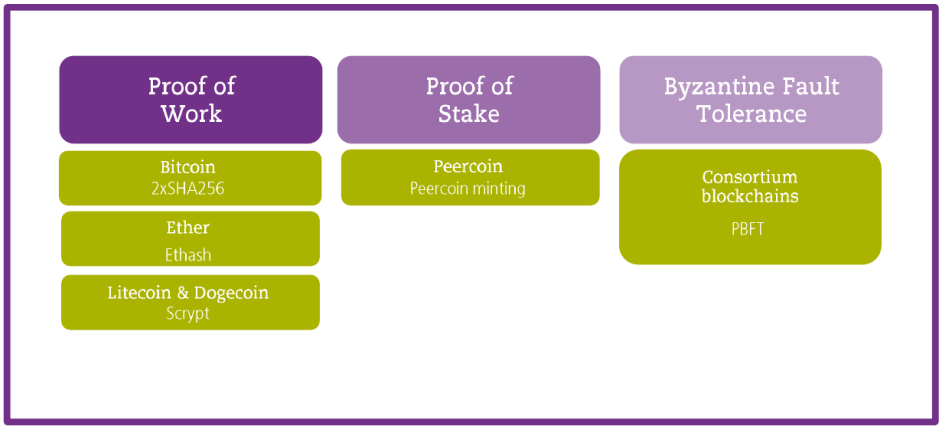
- Κρυπτονομίσματα που έχουν αξία σε fiat -κλασσικά- νομίσματα: Σε αυτή την κατηγορία εφαρμογών Blockchain συμπεριλαμβάνονται κρυπτονομίσματα όπως το Bitcoin και το Litecoin τα οποία μπορούν να χρησιμοποιηθούν για την πραγματοποίηση αγορών ή

χρηματικών συναλλαγών. Σε αυτή την κατηγορία βλέπουμε ότι οι περισσότερες εφαρμογές περιλαμβάνουν τον αλγόριθμο consensus Proof of Work, ο οποίος είναι ξεπερασμένος για τα σημερινά δεδομένα καθώς έχει αρκετά προβλήματα όπως αναλύσαμε και στην ενότητα 2.2. Πολλές εφαρμογές κρυπτονομισμάτων έχουν αρχίσει να υλοποιούν τον αλγόριθμο Proof of Stake διότι είναι η βέλτιστη επιλογή για αυτό το σενάριο χρήσης και δίνει αρκετές δυνατότητες επεκτασιμότητας [18] [29].

- Πλατφόρμες που υλοποιούν ιδιωτικής ή περιορισμένης πρόσβασης Blockchain. Είναι δίκτυα στα οποία οι κόμβοι που συμμετέχουν έχουν λάβει εξουσιοδότηση και δηλαδή το παραπάνω Blockchain περιλαμβάνει αξιόπιστους γνωστούς κόμβους. Τέτοια Blockchain χρησιμοποιούν μεγάλοι χρηματοπιστωτικοί και κυβερνητικοί οργανισμοί για να κοινοποιήσουν τα εμπιστευτικά δεδομένα τους σε ένα ασφαλές αλλά εύκολα προσβάσιμο περιβάλλον. Τέτοιες εφαρμογές ως επί το πλείστον χρησιμοποιούν το αλγόριθμο Proof of Authority ο οποίος έχει συγκεντρωτικό χαρακτήρα και παρέχει δικαιώματα στο σύστημα με βάση την αξιοπιστία της ταυτότητας του χρήστη και επίσης οι κόμβοι που δημιουργούν και επικυρώνουν Block είναι εξουσιοδοτημένοι από την αρχή και είναι η μόνη εξουσία στο Blockchain. Με τον τρόπο που λειτουργεί ο συγκεκριμένος αλγόριθμος μπορούν να αναπτυχθούν ιδιωτικά ή περιορισμένης πρόσβασης Blockchains [9].

- Πλατφόρμες διαχείρισης στόλου συσκευών IoT -Internet of Things-. Αυτός ο τύπος εφαρμογών Blockchain είναι ένα σενάριο χρήσης το οποίο έχει αρχίσει να αναπτύσσεται τα τελευταία χρόνια και υπάρχουν κάποιες υλοποιήσεις όπως το IOTA(http://www.iota.org). Σε αυτή την κατηγορία εφαρμογών ο σκοπός είναι η διαχείριση και η δυνατότητα παρακολούθησης ηλεκτρικών συσκευών οι οποίες έχουν την δυνατότητα να συνδεθούν στο διαδίκτυο και οι οποίες είναι το μέλλον των οικιακών και βιομηχανικών συσκευών και μηχανημάτων. Μία τέτοια εφαρμογή Blockchain παρέχει ευελιξία και αποδοτικότητα σε αυτό το τύπο εφαρμογών ειδικότερα σε συστήματα συλλογής δεδομένων. Σε αυτή την κατηγορία εφαρμογών η βέλτιστη τεχνολογία DLT είναι η τεχνολογία των Directed Acyclic Graphs, καθώς παρέχουν ευελιξία στο τρόπο δημιουργίας και επικύρωσης των Blocks καθώς επίσης και στην ελάχιστη επεξεργαστική ισχύ που απαιτείται από τους κόμβους που θέλουν να προσθέσουν Blocks. Αυτό το κάνει ιδανική περίπτωση για την παραπάνω κατηγορία [12] [16].

Συμπέρασμα: Όπως εξετάσαμε παραπάνω, υπάρχουν τρείς βασικές εφαρμογές της τεχνολογίας Blockchain οι οποίες έχουν τις ανάλογες απαιτήσεις από τον αλγόριθμο consensus, οπότε επιλέγεται ο κατάλληλος αλγόριθμος για την εκπλήρωση των απαιτήσεων αυτών.



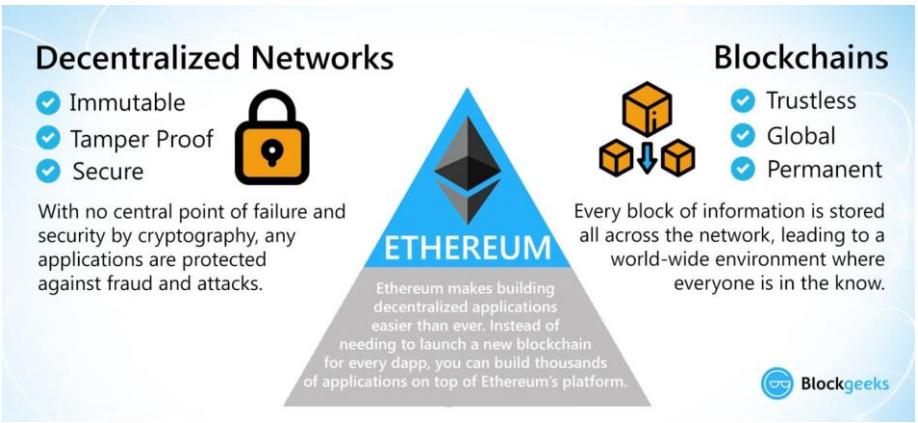

*Εικόνα 5: Μερικοί τύποι αλγορίθμων consensus και υπάρχουσες εφαρμογές τους[17]*

# **ΚΕΦΑΛΑΙΟ 3:** Η ΠΛΑΤΦΟΡΜΑ ETHEREUM

## 3.1 Τι είναι το Ethereum

Το Ethereum αποτελεί μία πλατφόρμα με εργαλεία και τεχνολογίες για την οργανωμένη και απλή υλοποίηση εφαρμογών που αξιοποιούν την τεχνολογία Blockchain και τις πιο πρόσφατες τεχνολογίες DLT. Στο Ethereum παρέχονται διάφορα εργαλεία και πακέτα ανάπτυξης -IDE- για την παροχή δυνατοτήτων σε προγραμματιστές που επιθυμούν να ενσωματώσουν την τεχνολογία των κατανεμημένων βάσεων δεδομένων τύπου Blockchain στις εφαρμογές τους. Το Ethereum παρέχει μία πληθώρα δυνατοτήτων, εργαλείων και προγραμματιστικών μοτίβων -software patterns- τα οποία σκοπό έχουν να διευκολύνουν τους προγραμματιστές στην ανάπτυξη αποδοτικών, ασφαλών και σταθερών εφαρμογών, και όπου ο πηγαίος κώδικας καθώς και η δομή του project θα προσφέρουν μία ξεκάθαρη εικόνα για το πώς λειτουργεί το code base -η ολική βάση κώδικα της εφαρμογής- και για ποιόν τομέα της εφαρμογής καθώς επίσης και τι χρήσεις αξιοποιεί το κάθε τμήμα ή αρχείο της εφαρμογής για την εκτέλεση των κατάλληλων διεργασιών. Αυτό δίνει την δυνατότητα σε μία ομάδα προγραμματιστών να δημιουργούν με σχετική ευκολία πολύπλοκες εφαρμογές οι οποίες θα αξιοποιούν τις τεχνολογίες Blockchain. Αυτή η πλατφόρμα επίσης δίνει ένα αρκετά καλό επίπεδο abstraction μέσο των υπο-προγραμμάτων του που αξιοποιούν την τεχνολογία των Smart Contracts -έξυπνα συμβόλαια- [13] όπου σε αυτή την περίπτωση επί το πλείστον αποτελούν λειτουργικά τμήματα τα οποία είναι επαναχρησιμοποιήσημα σε άλλες εφαρμογές και τα οποία εκτελούν μία συγκεκριμένη λειτουργία ή μία συγκεκριμένη επεξεργαστική εργασία, όπως επίσης και ότι μπορούν να επικοινωνήσουν μεταξύ τους -δηλαδή η έξοδος της μίας διεργασίας να τροφοδοτεί την είσοδο μίας άλλης- και με τον βασικό πυρήνα της εφαρμογής. Το Ethereum αξιοποιεί αυτή την τεχνολογία στο περιβάλλον ανάπτυξης κατανεμημένων εφαρμογών -dApps- όπως φαίνεται και στην εικόνα 6, και παρέχουν τις ίδιες δυνατότητες που παρέχουν οι βιβλιοθήκες στις «παραδοσιακές» γλώσσες προγραμματισμού, δηλαδή παρέχουν έτοιμες συναρτήσεις και κλάσεις και λειτουργίες οι οποίες μπορούν να αξιοποιηθούν στο κύριο πυρήνα της εφαρμογής [13].

Γενικότερα λοιπόν, το Ethereum αποτελεί πλατφόρμα ανάπτυξης εφαρμογών με κεντρικό σκοπό την αξιοποίηση της καινοτομικής τεχνολογίας Blockchain και την διευκόλυνση των προγραμματιστών που θέλουν να την αξιοποιήσουν σε κάποιο project.




*Εικόνα 6: Μερικά βασικά features για την πλατφόρμα Ethereum[23].*

## 3.2 Ποιά η διαφορά του από τις άλλες πλατφόρμες που αξιοποιούν την τεχνολογία Blockchain

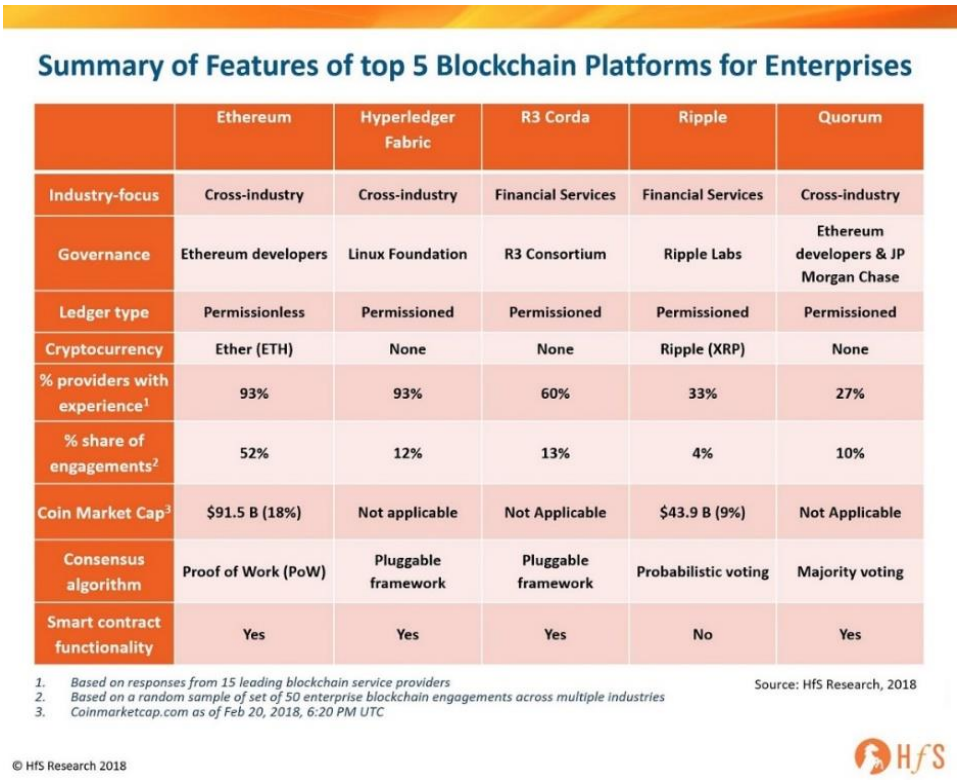
Σε αυτό το κεφάλαιο θα αναφέρουμε σε αυτή την πλατφόρμα τι είναι διαφορετικό και τι είναι μοναδικό σε αυτή την πλατφόρμα ή το αξιοποιεί στο μέγιστο των δυνατοτήτων του, το οποίο την έχει καταστήσει αρκετά δημοφιλή σε προγραμματιστές που θέλουν να αναπτύξουν εφαρμογές οι οποίες αξιοποιούν την τεχνολογία Blockchain ή άλλες τεχνολογίες που προσφέρει το Ethereum και μπορούν να προσφέρουν μία μορφή κατανεμημένου συστήματος αποθήκευσης και ανάκτησης δεδομένων. Το Ethereum όπως αναφέραμε και παραπάνω είναι μία πλατφόρμα η οποία προσφέρει μία πληθώρα εργαλείων σε προγραμματιστές τα οποία μπορούν να αξιοποιήσουν την τεχνολογία Blockchain και γενικότερα την τεχνολογία των «κατανεμημένων λογιστικών βιβλίων» - Distributed Ledger Technology. Εφόσον αυτή η πλατφόρμα προσφέρει αρκετές δυνατότητες για ανάπτυξη εφαρμογών που αξιοποιούν την τεχνολογία αυτή, προσέλκυσε αρκετούς προγραμματιστές που ήθελαν να ενσωματώσουν ένα DLT στις εφαρμογές τους, όπως επίσης ο έξυπνος τρόπος που αξιοποιεί την τελευταία τεχνολογία των smart contracts και η δυνατότητα που προσφέρει η πλατφόρμα για κατασκευή ιδιωτικού Blockchain την κατέστησε «νούμερο 1» πλατφόρμα σε δημοτικότητα για όλα αυτά τα παραπάνω που ο συνδυασμός τους προσφέρει μία μοναδική εμπειρία στους χρήστες της, όπως επίσης και η πληθώρα δυνατοτήτων και η αξιοπιστία που προσφέρει δίνουν μία πολύ θετική εντύπωση στον άνθρωπο που έχει όραμα να αξιοποιήσει την τεχνολογία για να δημιουργήσει εφαρμογές και να αξιοποιήσει όλα τα πλεονεκτήματα και οι δυνατότητες που προσφέρουν οι νέες τεχνολογίες του Blockchain και η δυνατότητα του να αποκεντρώσει τις διάφορες διεργασίες που πραγματοποιούνται σε μία κεντρική μονάδα εξυπηρέτησης -server- ή σε ένα κεντρικό «πλέγμα» από dedicated μηχανήματα εξυπηρέτησης.

Συνοπτικά το Ethereum διαφέρει από τις άλλες πλατφόρμες στους παρακάτω τομείς:

- Δεν είναι και δεν παρέχει κρυπτονόμισμα για συναλλαγές, αλλά μπορεί να υλοποιηθεί αν το επιθυμεί κάποιος προγραμματιστής. Ο βασικός σκοπός του Ethereum όμως δεν είναι αυτός [5].
- Προσφέρει εργαλεία όπως βιβλιοθήκες κώδικα υλοποιημένες σαν smart contracts και περιβάλλοντα εργασίας για την ανάπτυξη εφαρμογών που αξιοποιούν τεχνολογίες DLT [6].
- Προσφέρουν μία πληθώρα δυνατοτήτων για το ποιοι κόμβοι και οντότητες θα έχουν δυνατότητα να έχουν πρόσβαση στο Blockchain και άρα στην εφαρμογή που έχει αναπτυχθεί, καθώς επίσης και ποιος θα είναι ο μηχανισμός consensus που θα αξιοποιεί η εφαρμογή ανάλογα τις ανάγκες της [5].
- Παρέχει μία αρκετά ενδελεχή τεκμηρίωση, καθώς επίσης μαθήματα για το πώς ένας προγραμματιστής μπορεί να αξιοποιήσει τα εργαλεία της πλατφόρμας και τέλος παρέχει αρκετά παραδείγματα με έτοιμο κώδικα που μπορεί να πειράξει κάποιος για να μπορεί να καταλάβει τι λειτουργία κάνει ένα παράδειγμα και τι δυνατότητες αξιοποιεί [6].

Όπως φαίνεται και στην εικόνα 7, συμπεραίνουμε ότι η πλατφόρμα Ethereum προσφέρει μία εκτενή γκάμα δυνατοτήτων και εργαλείων τα οποία μπορούν άμεσα οι

προγραμματιστές να τα αξιοποιήσουν για την κατασκευή εφαρμογής η οποία θα μπορεί να υλοποιηθεί ακριβώς με βάση τις ανάγκες και τις απαιτήσεις της εφαρμογής και των προδιαγραφών του συστήματος, καθώς η πλατφόρμα αυτή προσφέρει αρκετές δυνατότητες που μπορούν να φέρουν το Blockchain «στα μέτρα» των δεδομένων προδιαγραφών, όπως επίσης και η φιλοσοφία της πλατφόρμας που δεν είναι να τοποθετήσει στην αγορά άλλο ένα κρυπτονόμισμα από τα εκατοντάδες που υπάρχουν, αλλά να προσφέρει ένα εργαλείο το οποίο μπορεί να αξιοποιηθεί για την κατασκευή καινοτόμων εφαρμογών τα οποία μπορούν να λειτουργούν αρκετά πιο αποδότικα αν χρησιμοποιούν την τεχνολογία Blockchain και όχι μία κλασική βάση δεδομένων τύπου SQL [5].



**Summary of Features of top 5 Blockchain Platforms for Enterprises**

| | Ethereum | Hyperledger Fabric | R3 Corda | Ripple | Quorum |
|---|---|---|---|---|---|
| Industry-focus | Cross-industry | Cross-industry | Financial Services | Financial Services | Cross-industry |
| Governance | Ethereum developers | Linux Foundation | R3 Consortium | Ripple Labs | Ethereum developers & JP Morgan Chase |
| Ledger type | Permissionless | Permissioned | Permissioned | Permissioned | Permissioned |
| Cryptocurrency | Ether (ETH) | None | None | Ripple (XRP) | None |
| % providers with experience[1] | 93% | 93% | 60% | 33% | 27% |
| % share of engagements[2] | 52% | 12% | 13% | 4% | 10% |
| Coin Market Cap[3] | $91.5 B (18%) | Not applicable | Not Applicable | $43.9 B (9%) | Not Applicable |
| Consensus algorithm | Proof of Work (PoW) | Pluggable framework | Pluggable framework | Probabilistic voting | Majority voting |
| Smart contract functionality | Yes | Yes | Yes | No | Yes |

1. *Based on responses from 15 leading blockchain service providers*
2. *Based on a random sample of set of 50 enterprise blockchain engagements across multiple industries*
3. *Coinmarketcap.com as of Feb 20, 2018, 6:20 PM UTC*

Source: HfS Research, 2018



HfS

*Εικόνα 7: Πίνακας σύγκρισης των βασικότερων πλατφόρμων ανάπτυξης εφαρμογών που αξιοποιούν την τεχνολογία Blockchain[11]*

## 3.3 Παράδειγμα: Πως μπορεί να παραμετροποιηθεί ο μηχανισμός consensus στο Ethereum

Σε αυτή την ενότητα θα παρουσιασθεί ο τρόπος και η υλοποίηση που απαιτείται από την πλατφόρμα Ethereum για να παραμετροποιηθεί η διαδικασία και ο τρόπος με τον οποίο θα λειτουργεί ο μηχανισμός consensus σε private Ethereum. Όλες οι παρακάτω λεπτομέρειες έχουν αντληθεί από το blog Talentica το οποίο ασχολείται με πρακτικά θέματα του Ethereum και η ομάδα του εργάζεται κυρίως στον τομέα της ανάπτυξης εφαρμογών για το Ethereum καθώς επίσης και με την επέκταση της λειτουργικότητας του Ethereum [25] [26]. Τα θέματα με τα οποία θα ασχοληθούμε σε αυτή την ενότητα είναι τα παρακάτω:

- Μεταφόρτωση και εγκατάσταση της γλώσσας Go! η οποία είναι απαραίτητη για την «μετάφραση» των αρχείων του Ethereum σε εκτελέσιμα αρχεία για το λειτουργικό σύστημα που θέλει ο χρήστης να το εγκαταστήσει.

- Μεταφόρτωση και εγκατάσταση εκτελέσιμων αρχείων για την χρήση της πλατφόρμας Ethereum. Γενικότερα σε αυτό το τμήμα θα εξηγήσουμε τον

τρόπο με τον οποίο κάποιος μπορεί να εγκαταστήσει το Ethereum σε ένα τοπικό υπολογιστή ή σε ένα virtual machine σε κάποιο cloud.

- Δημιουργία του αρχείου το οποίο θα παράξει το πρώτο Block, γνωστό και ως Genesis Block, το αρχείο αυτό θα είναι ένα αντικείμενο σε μορφή JSON το οποίο θα είναι σε αρχείο με την κατάληξη .json.

- Εκκίνηση του αρχικού κόμβου με τον προεπιλεγμένο μηχανισμό consensus και σύνδεση δεύτερου κόμβου με το οποίο θα εξετάσουμε κάποιες βασικές επιλογές που προσφέρει η κονσόλα Javascript του Ethereum καθώς επίσης θα εξετάσουμε και την απόδοση του mining με τις προεπιλεγμένες παραμέτρους του Ethereum.

- Δημιουργία της βασικής δομής του συστήματος και του κώδικα «σκελετού» για την προσθήκη παραμετροποιημένου μηχανισμού consensus στο Ethereum και προσθήκη δυνατοτήτων στο API για παραμετροποίηση του προγράμματος πελάτη -client- του Ethereum για την προσθήκη δυνατοτήτων στους χρήστες της εφαρμογής. Όπως επίσης και η τροποποίηση κάποιων βασικών αρχείων για την αναγνώριση του παραμετροποιημένου αλγορίθμου consensus από το Ethereum.

- Το τελικό βήμα είναι η εκκίνηση του αρχικού κόμβου με τον παραμετροποιημένο αλγόριθμο consensus ο οποίος θα έχει ρυθμιστεί στο JSON αρχείο που θα παρέχουμε στην εντολή έναρξης του, όπως και ξανά η εκκίνηση του προγράμματος πελάτη για την εξέταση των νέων δυνατοτήτων μέσο του API από την κονσόλα Javascript και η εξέταση της απόδοσης του mining με τον παραμετροποιημένο μηχανισμό consensus.

**Βήμα 1**: Λήψη και εγκατάσταση της γλώσσας Go!

Για να μπορέσουν να πραγματοποιηθούν τα επόμενα βήματα που θα αναφέρουμε θα πρέπει ο κόμβος -τοπικός ή απομακρυσμένος- να έχει εγκατεστημένη την γλώσσα Go!. Με την εκτέλεση της παρακάτω εντολής [26] μπορεί να ελέγξει κάποιος αν υπάρχει εγκατεστημένη στο σύστημα του η γλώσσα Go! [25]:

```
> go version
go version go1.9.2 darwin/amd64
```

Αν υπάρχει η γλώσσα Go! στο σύστημα τότε μπορείτε να παρακάμψετε αυτό το βήμα και να προχωρήσετε στο «Βήμα 2». Αν δεν υπάρχει τότε πρέπει να περιηγηθείτε στην παρακάτω διεύθυνση:

https://golang.org/doc/install

Θα πρέπει να ληφθεί το κατάλληλο αρχείο ανάλογα το λειτουργικό σύστημα στο οποίο θα εγκατασταθεί η πλατφόρμα καθώς επίσης θα πρέπει να ακολουθειθούν και οι αντίστοιχες οδηγίες που παρέχει η παραπάνω σελίδα για την διαδικασία εγκατάστασης για το λειτουργικό σύστημα αντίστοιχα. Μόλις ολοκληρωθεί η εγκατάσταση ίσως να χρειαστεί να γίνει επανεκκίνηση του συστήματος για να

ρυθμιστούν σωστά οι μεταβλητές περιβάλλοντος. Για να ελέγξει κάποιος αν έχουν γίνει οι κατάλληλες ρυθμίσεις θα πρέπει να λάβει το κατάλληλο αποτέλεσμα όπως παρακάτω με την εκτέλεση της αντίστοιχης εντολής [26]:

```
 > echo $GOPATH
 /Users/hemants/Projects/mist/go-workspace
```

**Βήμα 2:** Λήψη και εγκατάσταση των απαραίτητων αρχείων για την χρήση της πλατφόρμας Ethereum

Τα αρχεία της πλατφόρμας Ethereum υπάρχουν μόνο σε αποθετήριο -repository- στο Github και με την παρακάτω διαδικασία μπορεί κάποιος να εγκαταστήσει την πλατφόρμα Ethereum στο σύστημα του [26]:

```
> cd $GOPATH/src/github.com/
> mkdir ethereum
> git clone git@github.com:ethereum/go-ethereum.git
```

Εφόσον ολοκληρωθεί αυτή η ενέργεια, μπορούμε τώρα να προχωρήσουμε στην εγκατάσταση του Ethereum με χρήση της κατάλληλης εντολής συστήματος το οποίο θα κάνει compile τα απαραίτητα αρχεία για να δημιουργηθεί ένα εκτελέσιμο αρχείο που είναι η κονσόλα του Ethereum geth με την παρακάτω εντολή [26]:

```
> go install -v ./cmd/geth
```

Για να ελεγχθεί αν έχει κατασκευαστεί σωστά το βασικό εκτελέσιμο αρχείο της κονσόλας geth μπορεί να εκτελεσθεί η παρακάτω εντολή [26]:

```
> ./geth version
Geth

Version: 1.8.12-unstable

Architecture: amd64

Protocol Versions: [63 62]

Network Id: 1

Go Version: go1.9.2

Operating System: darwin

GOPATH=/Users/hemants/Projects/mist/go-workspace

GOROOT=/usr/local/Cellar/go/1.9.2/libexec
```

**Βήμα 3:** Εκκίνηση του αρχικού κόμβου με τον προεπιλεγμένο μηχανισμό consensus και εξέταση της κονσόλας Javascript

Σε αυτό το βήμα θα εξετάσουμε πως λειτουργεί ένα τοπικό ιδιωτικό δίκτυο Blockchain με την πλατφόρμα Ethereum και με τις προεπιλεγμένες παραμέτρους.

Αρχικά, για να μπορούμε να εκκινήσουμε τον βασικό κόμβο θα πρέπει να κατασκευάσουμε το JSON αρχείο από το οποίο το Ethereum θα λάβει τις ρυθμίσεις και το

περιεχομένο για την κατασκευή του Genesis Block και την εκκίνηση του αρχικού κόμβου. Παρακάτω παραθέτουμε ένα παράδειγμα για το περιεχόμενο που πρέπει να περιλαμβάνει το αρχείο ώστε να το αναγνωρίσει σωστά το instance του Ethereum [26]:

```
{
"config": {
"chainId": 15,
"homesteadBlock": 0,
"eip155Block": 0,
"eip158Block": 0
},
"difficulty": "2000000",
"gasLimit": "21000000",
"alloc": {
}
}
```

Εφόσον υπάρχει αυτό το αρχείο μπορούμε να αρχικοποιήσουμε το τοπικό Blockchain εκτελόντας την παρακάτω εντολή [26]:

```
> ./geth --datadir ~/.ethereum/myprivatenet init privategensis.json
```

Αν έχουν γίνει όλα σωστά και έχουν εκτελεσθεί σωστά τα προηγούμενα βήματα τότε θα λάβουμε το παρακάτω αποτέλεσμα [26]:

```
INFO [06-21|13:43:05.226227] Maximum peer count ETH=25 LES=0 total=25

INFO [06-21|13:43:05.240084] Allocated cache and file handles databas
e=/Users/hemants/.ethereum/myprivatenet/geth/chaindata cache=16 handl
es=16

INFO [06-21|13:43:05.244943] Writing custom genesis block

INFO [06-21|13:43:05.245018] Persisted trie from memory database node
s=0 size=0.00B time=10.217µs gcnodes=0 gcsize=0.00B gctime=0s livenod
es=1 livesize=0.00B

INFO [06-21|13:43:05.24539] Successfully wrote genesis state database
=chaindata hash=07185f…82bcc4

INFO [06-21|13:43:05.245414] Allocated cache and file handles databas
e=/Users/hemants/.ethereum/myprivatenet/geth/lightchaindata cache=16
handles=16

INFO [06-21|13:43:05.24758] Writing custom genesis block

INFO [06-21|13:43:05.247618] Persisted trie from memory database node
s=0 size=0.00B time=2.699µs gcnodes=0 gcsize=0.00B gctime=0s livenode
s=1 livesize=0.00B
```

```
INFO [06-21|13:43:05.247787] Successfully wrote genesis state databas
e=lightchaindata hash=07185f…82bcc4
```

Εφόσον έχει ολοκληρωθεί και το παραπάνω τότε θα είμαστε σε θέση να μπορούμε να εκκινήσουμε τον κόμβο μας με την παρακάτω εντολή [26]:

```
./geth -rpc -rpcapi 'web3,eth,debug,personal' -rpcport 8545 --rpccors
domain '*' --datadir ~/.ethereum/myprivatenet --networkid 15
```

Αν γίνει επιτυχώς η έναρξη του κόμβου τότε η γραμμή εντολών θα εμφανίσει το παρακάτω αποτέλεσμα [26] μόλις εκτελεσθεί η παραπάνω εντολή:

```
INFO [06-21|13:49:27.007881] Maximum peer count ETH=25 LES=0 total=25

INFO [06-21|13:49:27.01546] Starting peer-to-peer node instance=Geth/
v1.8.12-unstable/darwin-amd64/go1.9.2

INFO [06-21|13:49:27.015503] Allocated cache and file handles databas
e=/Users/hemants/.ethereum/myprivatenet/geth/chaindata cache=768 hand
les=128

INFO [06-21|13:49:27.028411] Initialised chain configuration config="
{ChainID: 15 Homestead: 0 DAO:  DAOSupport: false EIP150:  EIP155: 0
EIP158: 0 Byzantium:  Constantinople:  Engine: unknown}"

INFO [06-21|13:49:27.028465] Disk storage enabled for ethash caches d
ir=/Users/hemants/.ethereum/myprivatenet/geth/ethash count=3

INFO [06-21|13:49:27.028479] Disk storage enabled for ethash DAGs dir
=/Users/hemants/.ethash count=2

INFO [06-21|13:49:27.028513] Initialising Ethereum protocol versions=
"[63 62]" network=16

INFO [06-21|13:49:27.029799] Loaded most recent local header number=0
hash=07185f…82bcc4 td=2000000

INFO [06-21|13:49:27.029836] Loaded most recent local full block numb
er=0 hash=07185f…82bcc4 td=2000000

INFO [06-21|13:49:27.029845] Loaded most recent local fast block numb
er=0 hash=07185f…82bcc4 td=2000000

INFO [06-21|13:49:27.03009] Loaded local transaction journal transact
ions=0 dropped=0

INFO [06-21|13:49:27.030331] Regenerated local transaction journal tr
ansactions=0 accounts=0

INFO [06-21|13:49:27.030806] Starting P2P networking

INFO [06-21|13:49:29.146101] UDP listener up self=enode://1dd1494242e
e403a69fbb58a57505056f5fea5c9f4b207050ca0a1aecc36d74b4687ec1779320c62
82d044aae0078437e5aa7688c29f04e1281cc3f5a0f954e2@[::]:30303

INFO [06-21|13:49:29.146426] RLPx listener up self=enode://1dd1494242
ee403a69fbb58a57505056f5fea5c9f4b207050ca0a1aecc36d74b4687ec1779320c6
282d044aae0078437e5aa7688c29f04e1281cc3f5a0f954e2@[::]:30303

INFO [06-21|13:49:29.150067] IPC endpoint opened url=/Users/hemants/.
ethereum/myprivatenet/geth.ipc
```

```
INFO [06-21|13:49:29.150472] HTTP endpoint opened url=http://127.0.0.
1:8545cors=* vhosts=localhost
```

Εφόσον έχουμε το τοπικό Blockchain να τρέχει μπορούμε να εισέλθουμε στην κονσόλα Javascript του τοπικού Blockchain εκτελόντας την παρακάτω εντολή [26]:

```
./geth attach /Users/hemants/.ethereum/myprivatenet/geth.ipc
```

Αυτό μας δίνει πρόσβαση σε διάφορες δυνατότητες του API και δίνει δυνατότητα για web based πρόσβαση στο τοπικό Ethereum. Το αντικείμενο personal διαχειρίζεται τα διάφορα δεδομένα που έχουν να κάνουν με την ταυτότητα του κόμβου/χρήστη που έχει εισέλθει στο Blockchain. Δεδομένα όπως οι λογαριασμοί πρόσβασης, τα «πορτοφόλια» με τα οποία είναι συνδεδεμένα οι λογαριασμοί, κλπ, παρακάτω είναι όλες οι ιδιότητες και μέθοδοι του αντικειμένου personal [26]:

```
> personal

{

listAccounts: [],

listWallets: [],

deriveAccount: function(),

ecRecover: function(),

getListAccounts: function(callback),

getListWallets: function(callback),

importRawKey: function(),

lockAccount: function(),

newAccount: function github.com/ethereum/go-ethereum/console.(*bridge
).NewAccount-fm(),

openWallet: function github.com/ethereum/go-ethereum/console.(*bridge
).OpenWallet-fm(),

sendTransaction: function(),

sign: function github.com/ethereum/go-ethereum/console.(*bridge).Sign
-fm(),

signTransaction: function(),

unlockAccount: function github.com/ethereum/go-ethereum/console.(*bri
dge).UnlockAccount-fm()

}
```

Με την παρακάτω εντολή μπορούμε να δημιουργήσουμε έναν λογαριασμό χρήστη[26]:

```
> personal.newAccount("<κωδικός πρόσβασης>")
```

Εφόσον έχει δημιουργηθεί ο λογαριασμός χρήστη, τρέχοντας την παρακάτω εντολή μπορούμε να θέσουμε το σύστημα σαν miner [26]:

```
> miner.start(1)
```

Εφόσον ο miner τρέχει για κάποιο χρονικό διάστημα, μπορούμε να τρέξουμε την παρακάτω εντολή για να λάβουμε τον αριθμό των Blocks που έχουν δημιουργηθεί μέχρι την στιγμή που θα εκτελεσθεί η εντολή [26]:

```
> web3.eth.getBlockNumber(function(e,r){ console.log(r)})
2
```

Τέλος, αν θέλουμε να διακόψουμε την λειτουργία του miner, μπορούμε να εκτελέσουμε την παρακάτω εντολή [26]:

```
> miner.stop()
```

**Βήμα 4:** Δημιουργία της κατάλληλης δομής και συγγραφή του βασικού κώδικα για την εισαγωγή παραμετροποιημένου αλγόριθμου consensus

Σε αυτό το βήμα θα ασχοληθούμε με την κατασκευή του παραμετροποιημένου αλγορίθμου consensus μέσο της συγγραφής του κατάλληλου κώδικα και χρήση της κατάλληλης δομής για να φέρουμε εις πέρας το παραπάνω. Θα αρχίσουμε με την κατασκευή του αρχείου «myalgo.go» το οποίο θα περιλαμβάνει τον κύριο κώδικα που θα υλοποιεί τον παραμετροποιημένο αλγόριθμο consensus. Αυτό το αρχείο θα πρέπει να είναι στο παρακάτω φάκελο:

$GOPATH/src/github.com/ethereum/go-ethereum/consensus/myalgo

Στο παραπάνω αρχείο θα πρέπει να έχει γραφτεί ο κατάλληλος κώδικας ο οποίος αποτελεί ένα βασικό σκελετό για την υλοποίηση αλγόριθμου consensus που να ικανοποιούν το περιβάλλον εργασίας του Ethereum. Παρακάτω θα βρείτε τον κώδικα που αναφέραμε τώρα [25]:

```
myalgo.go
func (MyAlgo *MyAlgo) VerifyHeader(chain consensus.ChainReader, heade
r *types.Header, seal bool) error {

log.Info("will verfiyHeader")

return nil

}
func (MyAlgo *MyAlgo) VerifyHeaders(chain consensus.ChainReader, head
ers []*types.Header, seals []bool) (chan<- struct{}, <-chan error){

log.Info("will verfiyHeaders")

abort := make(chan struct{})

results := make(chan error, len(headers))
go func() {

for _, header := range headers {

err := MyAlgo.VerifyHeader(chain, header, false)

select {

case <-abort:

return
```

```
case results <- err:

}

}

}()

return abort, results

}
func (MyAlgo *MyAlgo) VerifyUncles(chain consensus.ChainReader, block
*types.Block) error {

log.Info("will verfiy uncles")

return nil

}
func (MyAlgo *MyAlgo) VerifySeal(chain consensus.ChainReader, header
*types.Header) error{

log.Info("will verfiy VerifySeal")

return nil

}
func (MyAlgo *MyAlgo) Prepare(chain consensus.ChainReader, header *ty
pes.Header) error{

log.Info("will prepare the block")

parent := chain.GetHeader(header.ParentHash, header.Number.Uint64()-1
)

if parent == nil {

return consensus.ErrUnknownAncestor

}
header.Difficulty = MyAlgo.CalcDifficulty(chain, header.Time.Uint64()
, parent)

return nil

}
func (MyAlgo *MyAlgo) CalcDifficulty(chain consensus.ChainReader, tim
e uint64, parent *types.Header) *big.Int {

return calcDifficultyHomestead(time, parent)

}
func (MyAlgo *MyAlgo) Finalize(chain consensus.ChainReader, header *t
ypes.Header, state *state.StateDB, txs []*types.Transaction,

uncles []*types.Header, receipts []*types.Receipt) (*types.Block, err
or){

log.Info("will Finalize the block")
```

```
header.Root = state.IntermediateRoot(chain.Config().IsEIP158(header.N
umber))

b := types.NewBlock(header, txs, uncles, receipts)

return b, nil

}
func (MyAlgo *MyAlgo) Seal(chain consensus.ChainReader, block *types.
Block, stop <-chan struct{}) (*types.Block, error){

log.Info("will Seal the block")

//time.Sleep(15 * time.Second)

header := block.Header()

header.Nonce, header.MixDigest = getRequiredHeader()

return block.WithSeal(header), nil

}
```

Με τον παραπάνω κώδικα έχουμε την βασική δομή του μηχανισμού consensus. Για να εισάγουμε νέες δυνατότητες στην κονσόλα Javascript πρέπει να δημιουργηθεί το παρακάτω αρχείο [25]:

$GOPATH/src/github.com/ethereum/go-ethereum/consensus/myalgo/api.go

Αυτό το αρχείο να περιλαμβάνει το παρακάτω κώδικα, όπως επίσης και τα αρχεία που αναφέρονται στο παρακάτω πλαίσιο να τροποποιηθούν ανάλογα [25]:

**In consensus/myalgo/api.go:**

```
type API struct {

chain consensus.ChainReader

myAlgo *MyAlgo

}

func (api *API) EchoNumber(ctx context.Context, number uint64) (uint6
4, error) {

fmt.Println("called echo number")

return number, nil
}
```

**In internal/web3ext/web3ext.go:**

```
var Modules = map[string]string{

"admin": Admin_JS,

...

"txpool": TxPool_JS,

"myalgo": MyAlgo_JS,

}
```

```
const MyAlgo_JS = `

web3._extend({

property: 'myalgo',

methods: [

new web3._extend.Method({

name: 'echoNumber',

call: 'myalgo_echoNumber',

params: 1,

inputFormatter: [null]

}),

]

})
```

Για να μπορεί να τον αναγνωρίσει η μηχανή του Ethereum θα πρέπει να τροποποιηθεί παρακάτω αρχείο:

$GOPATH/src/github.com/ethereum/go-ethereum/eth/backend.go

Συγκεκριμένα θα πρέπει να γίνει προσθήκη του παρακάτω κώδικα στην συνάρτηση “CreateConsensusEngine” [25]:

```
if chainConfig.MyAlgo != nil{

fmt.Println(“myalgo is configured as consensus engine")

return myalgo.New(chainConfig.MyAlgo, db)

}
```

Με τις παραπάνω προσθήκες/τροποποιήσεις έχουμε εισάγει στο Ethereum έναν παραμετροποιημένο αλγόριθμο consensus και είμαστε σε θέση να τον χρησιμοποιήσουμε για την εκκίνηση τοπικού Blockchain που θα τον αξιοποιεί. Θα το δούμε αυτό στο επόμενο βήμα.

**Βήμα 5:** Εκκίνηση του κόμβου με τον παραμετροποιημένο αλγόριθμο consensus και εξέταση των εργαλείων που προσφέρει η κονσόλα Javascript με αυτόν

Αυτό αποτελεί τελικό βήμα του παραδείγματος μας και θα ασχοληθεί με την εκκίνηση του τοπικού Blockchain με τον παραμετροποιημένο αλγόριθμο consensus, με την εξέταση των νέων δυνατότητων που έχουν αναπτυχθεί για την κονσόλα Javascript μέσο αυτού και τέλος με την απόδοση του mining χρησιμοποιόντας τον παραμετροποιημένο αλγόριθμο consensus. Για να ξεκινήσουμε, θα πρέπει να δημιουργήσουμε το αρχείο JSON με τις κατάλληλες παραμέτρους ώστε να αρχικοποιηθεί το Blockchain το οποίο θα κάνει χρήση του παραμετροποιημένου αλγορίθμου, παρακάτω υπάρχει ένα υποδειγματικό αρχείο JSON με τα κατάλληλα περιεχόμενα που επιτυγχάνει το σκοπό αυτό [25]:

```
{

"config": {

"chainId": 15,

"homesteadBlock": 0,

"eip155Block": 0,

"eip158Block": 0,

"myalgo" : {}

},

"difficulty": "2000000",

"gasLimit": "21000000",

"alloc": {

}

}
```

Εφόσον έχει δημιουργηθεί το παραπάνω αρχείο, ύστερα μπορούμε να εκκινήσουμε το Blockchain εκτελόντας την παρακάτω εντολή στο τερματικό [25]:

```
./geth --datadir ~/.ethereum/myprivatenet init privategensis.json
```

Με την επιτυχή εκτέλεση της παραπάνω εντολής μπορούμε να εκκινήσουμε τον κόμβο με τον ίδιο τρόπο όπως στο παράδειγμα με τον προεπιλεγμένο αλγόριθμο consensus εκτελόντας την παρακάτω εντολή [26]:

```
./geth -rpc -rpcapi 'web3,eth,debug,personal' -rpcport 8545 --rpccors
domain '*' --datadir ~/.ethereum/myprivatenet --networkid 15
```

Εφόσον έχουν γίνει όλα αυτά και ο βασικός κόμβος έχει εκκινηθεί, τότε μπορούμε όπως στο 3ο βήμα να εισέλθουμε στην κονσόλα Javascript μέσο του IPC καναλιού και να εξετάσουμε τις νέες δυνατότητες που προσφέρει ο παραμετροποιημένος αλγόριθμος consensus. Εκκινόντας ένα νέο παράθυρο τερματικού και πηγαίνοντας στο κατάλληλο φάκελο, με την εκτέλεση της παρακάτω εντολής μπορούμε να εισέλθουμε στην κονσόλα Javascript [25] [26]:

```
./geth attach /Users/hemants/.ethereum/myprivatenet/geth.ipc
```

Αρχικά μπορούμε να εξετάσουμε τι προσφέρει το API του παραμετροποιημένου αλγόριθμου consensus με την παρακάτω εντολή στην κονσόλα Javascript [25]:

```
> myalgo
{

echoNumber: function()

}
```

Η παραπάνω εντολή μας εμφάνισε ότι το API μας προσφέρει μία λειτουργία η οποία ονομάζεται echoNumber. Μπορούμε να εξετάσουμε την λειτουργία που πραγματοποιεί εκτελόντας την παρακάτω εντολή [25]:

```
> myalgo.echoNumber(5)

called echo number

5
```

Με τα παραπάνω αποτελέσματα μπορούμε να καταλάβουμε ότι η λειτουργία που εκτελεί η μέθοδος που μας προσφέρει το API είναι μία εντολή η οποία δέχεται έναν ακέραιο και η οποία τυπώνει τον ακέραιο αυτό μαζί με ένα μήνυμα στην οθόνη του χρήστη. Ύστερα από αυτό μπορούμε να πάμε να δούμε την απόδοση του miner. Αρχικά δημιουργούμε έναν νέο λογαριασμό χρήστη με την παρακάτω εντολή [26]:

```
> personal.newAccount("<κωδικός πρόσβασης>")
```

Εφόσον έχουμε δημιουργήσει το λογαριασμό χρήστη, μπορούμε να εκκινήσουμε τον miner εκτελόντας την παρακάτω εντολή:

```
> miner.start(1)
```

Ύστερα από μερικά δευτερόλεπτα μπορούμε να σταματήσουμε τον miner και να ελέγξουμε πόσα Blocks έχουν παραχθεί. Για να διακοπεί ο miner πρέπει να εκτελεσθεί η παρακάτω εντολή [26]:

```
> miner.stop()
```

Μπορούμε να ελέγξουμε πόσα Blocks έχουν παραχθεί μέχρι την στιγμή που διακόπηκε ο miner με την παρακάτω εντολή [25] [26]:

```
> web3.eth.getBlockNumber(function(e,r){ console.log(r)})
17
```

Παρατηρούμε ότι μέσα σε λίγα δευτερόλεπτα παράχθηκαν 17 Blocks [25] σε αντίθεση με την προηγούμενη περίπτωση με τις προεπιλεγμένες παραμέτρους όπου χρειάστηκε αρκετός χρόνος για να δημιουργηθούν 2 Blocks [26]. Εφόσον τελειώσαμε και με αυτό το βήμα πάμε σε μία αποτίμηση και σε παροχή συνδέσμου στο Github όπου υπάρχει το go-ethereum μαζί με τα τροποποιημένα αρχεία ολοκληρωμένα και έτοιμα για δοκιμές.

Με βάση τα παραπάνω βήματα και τις δοκιμές που έγιναν τοπικά και σε cloud μπορούμε να πούμε ότι αυτή η μέθοδος δίνει την δυνατότητα για την δημιουργία παραμετροποιημένου αλγόριθμου consensus. Όμως για να μπορεί να γίνει αυτό, πρέπει ο προγραμματιστής που θα ασχοληθεί να είναι πεπειραμένος με τα Blockchains, με την πλατφόρμα του Ethereum και με ακόμα πιο σημαντικό με την γλώσσα προγραμματισμού Go! στην οποία είναι υλοποιημένες όλες οι βασικές λειτουργίες. Παρακάτω παραθέτουμε σύνδεσμο στο Github όπου υπάρχουν τα αρχεία του Ethereum μαζί με τα αρχεία του παραμετροποιημένου αλγορίθμου consensus στο παρακάτω σύνδεσμο:

https://github.com/Talentica/EthereumCustomConsensus

Αν κάποιος θέλει να κάνει build αυτή την εκδοχή, μετά το “git clone <διεύθυνση>” θα πρέπει να μετονομάσει το φάκελο EthereumCustomConsensus σε ethereum. Με τα παραπάνω κλείνουμε το πρακτικό παράδειγμα για την εισαγωγή παραμετροποιημένου αλγορίθμου consensus σε τοπικό Ethereum.

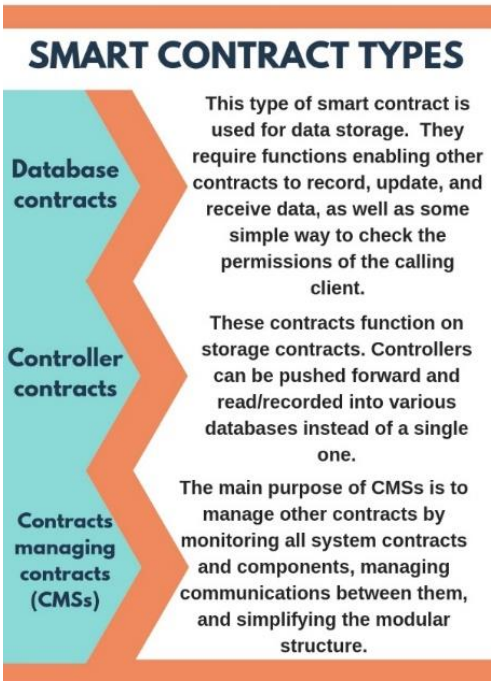
# **ΚΕΦΑΛΑΙΟ 4:** ΠΩΣ ΥΛΟΠΟΙΟΥΝΤΑΙ ΤΑ SMART CONTRACTS ΣΤΟ ETHEREUM ΚΑΙ ΠΟΙΕΣ ΟΙ ΕΦΑΡΜΟΓΕΣ ΤΟΥΣ

## 4.1 Τι είναι τα Smart Contracts

Τα Smart Contracts αποτελούν μία νέα τεχνολογία που σχετίζεται με την τεχνολογία των Blockchain και παρέχει δυνατότητες για σύναψη πλήρως ψηφιοποιημένων συμβάσεων μεταξύ δύο πλευρών, επίσης όμως αξιοποιούνται και σαν βιβλιοθήκες κώδικα ή προγραμματιστικές διεπαφές για την παροχή δυνατότητας σε προγραμματιστές να το χρησιμοποιούν στις εφαρμογές που αναπτύσουν ώστε να μπει ο προγραμματιστής στην διαδικασία να κατασκευάσει μία πολύπλοκη λειτουργικότητα η οποία υπάρχει ήδη. Τα Smart Contracts θα αποτελέσουν μία εναλλακτική στα χειρόγραφα συμβόλαια που συντάσσονται από δικηγόρους και συμβολαιογράφους στην τωρινή εποχή και λαμβάνουν υψηλά ποσά για να πραγματοποιήσουν αυτές τις εργασίες. Τα Smart Contracts τα οποία αξιοποιούνται σαν προγραμματιστικές λειτουργικές μονάδες χωρίζονται στις παρακάτω τρείς κατηγορίες [15], με βάση και τις πληροφορίες που παρουσιάζονται στην εικόνα 8:

- **Συμβάσεις δεδομένων:** Αυτός ο τύπος Smart Contract προσδιορίζει παραμέτρους που έχουν σχέση με την αποθήκευση και ανάκτηση των δεδομένων και προσδιορίζει άδειες και εξουσιοδοτήσεις για τους χρήστες όσον αφορά αυτά τα δεδομένα [15]. Επίσης αυτός ο τύπος Smart Contract απαιτεί λειτουργίες οι οποίες παρέχουν δυνατότητες για αποθήκευση και καταγραφή, ενημέρωση και ανάκτηση των δεδομένων [15].
- **Συμβάσεις ελεγκτών:** Αυτός ο τύπος Smart Contract αξιοποιεί τα Smart Contracts του προηγούμενου τύπου και η βασική τους χρήση είναι η πρόσβαση σε πολλαπλές βάσεις δεδομένων για την εκτέλεση των απαραίτητων λειτουργιών [15].

- **Συμβάσεις που διαχειρίζονται άλλες συμβάσεις:** Αυτός ο τύπος Smart Contract είναι μία διαχειριστική μονάδα η οποία έχει πρόσβαση στα υπόλοιπα Smart Contracts της εφαρμογής μέσο της παρακολούθησης του συστήματος και απλοποιεί την δομή τους για καλύτερη ενδοεπικοινωνία μεταξύ των Smart Contracts [15].




*Εικόνα 8: Τύποι Smart Contracts[15].*

Επίσης υπάρχουν Smart Contracts τα οποία όπως βλέπουμε και στην εικόνα 9 μιμούνται τα τυπικά συμβόλαια, αυτά θα τα δούμε στην επόμενη ενότητα.

Οπότε συνοπτικά, τα Smart Contracts αποτελούν τεχνολογία των τελευταίων ετών η οποία σε συνεργασία με την τεχνολογία Blockchain μπορούν να παρέχουν τυπικές συμβάσεις χωρίς συμβολαιογράφους και δικηγόρους, επίσης όμως παρέχουν την δυνατότητα για αξιοποίηση τους σαν προγραμματιστικές μονάδες και διεπαφές έτοιμου κώδικα.


*Εικόνα 9: Τομείς που μπορούν να εφαρμοσθούν τα Smart Contracts[21].*

## 4.2 Μερικές εφαρμογές των Smart Contracts

Στην παρούσα ενότητα θα αναφέρουμε τους τέσσερις βασικούς άξονες εφαρμογών των Smart Contracts που μπορούν να υλοποιηθούν. Αυτοί οι τέσσερις άξονες βασίζονται στο σκοπό για τον οποίο έχουν δημιουργηθεί τα Smart Contracts και το όραμα με το οποίο υλοποιήθηκε η όλη τεχνολογία και η αξιοποίηση της από τους χρήστες. Παρακάτω είναι αυτοί οι τέσσερις άξονες εφαρμογών, όπως βλέπουμε και στην εικόνα 10 [28]:

- **Νομικά Smart Contracts:** Αποτελούν το βασικό σκοπό για τον οποίο δημιουργήθηκαν τα Smart Contracts. Αποτελούν συμβόλαια τα οποία έχουν κάποια νομική υπόσταση και βασίζονται σε συναλλαγές δύο προσώπων -φυσικών ή νομικών- και ο σκοπός αυτής της κατηγορίας εφαρμογών Smart Contracts είναι η αντικατάσταση των παραδοσιακών έγγραφων συμβάσεων που δημιουργούνται από δικηγόρους ή συμβολαιογράφους. Συμβάσεις όπως ασφαλιστήρια συμβόλαια, συμβόλαια ενοικίασης/αγοράς κατοικίας, μνημόνια συνεργασίας μεταξύ δύο ή παραπάνω οργανισμών/επιχειρήσεων, κλπ [28].

- **Συμβόλαια αποκεντρωμένων αυτόνομων οργανισμών:** Αποτελούν συμβάσεις οι οποίες καθορίζουν τον κώδικα δεοντολογίας του οργανισμού, ομάδες συμβολαίων που καθορίζουν την περιουσία του οργανισμού -assets-, προσδιορίζουν διοικητικές και λοιπές λειτουργίες του οργανισμού και τον τρόπο που θα διεκπεραιώνονται, καθώς και τα πρόσωπα που θα καταλαμβάνουν αυτές τις διοικητικές θέσεις, δηλαδή αυτή η εφαρμογή Smart Contract αποτελεί μία πλήρως ψηφιοποιημένη μορφή του καταστατικού του οργανισμού [28].

- **Κατανεμημένες εφαρμογές:** Αυτός ο τύπος εφαρμογής Smart Contracts αποτελεί μία μορφή αντικατάστασης του παραδοσϊακού τρόπου οργάνωσης του κώδικα σε βιβλιοθήκες και APIs η οποία χρησιμοποιεί τα Smart Contracts σαν τμήματα κώδικα τα οποία το σύνολο τους εκτελούν κάποιες βασικές λειτουργίες που απαιτεί η κατανεμημένη εφαρμογή -dApp-. Με αυτό το τρόπο το Ethereum χρησιμοποιεί κυρίως την τεχνολογία των Smart Contracts [28].

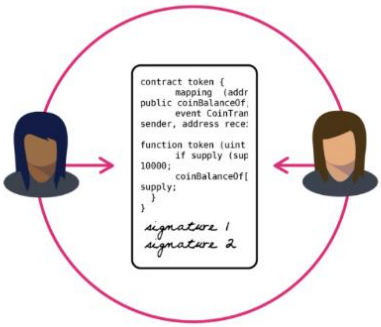
- **Συμβάσεις και επαφές μεταξύ IoT συσκευών:** Αυτή η μορφή εφαρμογής των Smart Contracts αποτελεί «σύμβαση» μεταξύ συσκευών οι οποίες μπορούν να λάβουν απομακρυσμένες εντολές χειρισμού ή εντολές για συλλογή δεδομένων από τα αισθητήρια εξαρτήματα τους. Ο σκοπός αυτών των Smart Contracts είναι ο προσδιορισμός του τρόπου με τον οποίο θα επικοινωνούν οι συσκευές αυτές, ποιές εντολές θα μπορούν να λάβουν και ποιά συστήματα θα έχουν την άδεια να λάβουν δεδομένα και να στείλουν εντολές στις συσκευές αυτές[28].

Γενικότερα, υπάρχουν αρκετές εφαρμογές των Smart Contracts οι οποίες επιχειρούν να αντικαταστήσουν παραδοσϊακά συστήματα ή μεθόδους με σκοπό να διευκολύνουν τους προγραμματιστές, να μειώσουν τα έξοδα σε ιδιώτες και επιχειρηματίες που θέλουν να συνάψουν κάποιο συμβόλαιο και γενικότερα να δώσει μία καινοτομία σε αυτό το τομέα η οποία θα τον απευλευθερώσει από ξεπερασμένες πρακτικές.


*Εικόνα 10: Βασικοί άξονες εφαρμογών των Smart Contracts[28].*

## 4.3 Πως αξιοποιεί το Ethereum τα Smart Contracts

Το Ethereum αποτελεί πλατφόρμα η οποία όπως αναφέραμε και στο προηγούμενο κεφάλαιο έχει σκοπό την δημιουργία ενός περιβάλλοντος για προγραμματιστές που θέλουν να αξιοποιήσουν την τεχνολογία Blockchain στις εφαρμογές τους. Συγκεκριμένα το Ethereum έχει την πιο ολοκληρωμένη εκδοχή μηχανισμού αξιοποίησης των Smart Contracts και όσοι προγραμματιστές θέλουν να εργαστούν με την τεχνολογία των Smart Contracts την επιλέγουν διότι είναι η μοναδική πλατφόρμα η οποία προσφέρει αυτό το επίπεδο δυνατοτήτων στην τεχνολογία των Smart Contracts. Το Ethereum αξιοποιεί όλους τους άξονες των Smart Contracts που αναφέρονται παραπάνω, αλλά η πιο συχνή και πιο διαδεδομένη χρήση τους από τους προγραμματιστής είναι ο άξονας των Smart Contracts που λειτουργούν ως προγραμματιστικές λειτουργικές μονάδες με την λογική βιβλιοθήκης ή API. Σε αυτό το τομέα τα αξιοποιεί πλήρως το Ethereum τα Smart Contracts το οποίο προσφέρει ένα παραπάνω επίπεδο αφαιρετικού σχεδιασμού και δίνει δυνατότητες στον έλεγχο των παραμέτρων εκτέλεσης επειδή τα Smart Contracts εκτελούνται ακριβώς όπως αναφέρεται στο «συμβόλαιο» [13] όπως φαίνεται και στην εικόνα 11. Με αυτή την δυνατότητα και όλες τις δυνατότητες που προσφέρονται από την κλασσική έννοια των βιβλιοθηκών κώδικα ένας προγραμματιστής μπορεί να αξιοποιήσει στο μέγιστο τις δυνατότητες που προσφέρουν οι τεχνολογίες Blockchain και η τεχνολογία των Smart Contracts και είναι ο λόγος ο οποίος επιλέχθηκε να μελετηθεί η συγκεκριμένη πλατφόρμα για την παρούσα διπλωματική εργασία.




*Εικόνα 11: Smart Contract[13]*

## 4.4 Μελλοντικές προοπτικές

Σε αυτή την ενότητα θα αναφέρουμε μερικές μελλοντικές προοπτικές που μπορούν να γίνουν πραγματικότητα αν αξιοποιηθούν τα εργαλεία που προσφέρει η πλατφόρμα Ethereum. Η πλατφόρμα Ethereum και γενικότερα η τεχνολογία Blockchain μπορεί να προσφέρει μία πληθώρα μελλοντικών εφαρμογών, όπως φαίνεται και στην εικόνα 12, οι οποίες θα μπορούν να αντικαταστήσουν κάποιες παραδοσϊακές αλλά ξεπερασμένες πρακτικές σε διάφορες βιομηχανίες και κλάδους. Τα Blockchain σε συνδυασμό με τα Smart Contracts μπορούν να δημιουργήσουν καινοτομίες στις παρακάτω βιομηχανίες[31]:

- Στον νομικό κλάδο: Μπορούν να αντικαταστήσουν τα συμβόλαια τα οποία συγγράφονται από δικηγόρους και συμβολαιογράφους, κατεβάζοντας το κόστος για την επιχείρηση ή ακόμη και τον ιδιώτη, αντικαθιστώντας τα συμβόλαια που υπογράφονται με πένα θα υπάρχει η δυνατότητα πλήρως ψηφιοποιημένων συμβολαίων τα οποία μπορούν να αξιοποιηθούν και από τις τοπικές αρχές για διάφορες διαδικασίες -π.χ. φορολογία κινητής και ακίνητης περιουσίας- [31].

- Στον κλάδο της εφοδιαστικής αλυσίδας και των logistics: Μπορούν να δώσουν δυνατότητες για παρακολούθηση ολόκληρης της εφοδιαστικής αλυσίδας δίνοντας στους εμπλεκόμενους ένα σημαντικό εργαλείο για να αξιολογείται η απόδοση της εφοδιαστικής αλυσίδας και ποιά είναι τα προβληματικά σημεία [31].

- Στον κλάδο των έξυπνων δικτυωμένων συσκευών IoT: Μπορεί να προσφέρει δυνατότητες για παρακολούθηση, εκτέλεση ενεργειών και ενδοεπικοινωνία μεταξύ των συσκευών χωρίς την επέμβαση τρίτων. Με αυτή την δυνατότητα θα μπορεί κάποιος από οπουδήποτε στο κόσμο να ελέγχει τον στόλο των συσκευών του μέσο του διαδικτύου και να αποστέλλει διάφορες εντολές σε αυτές, οι οποίες θα διαδίδονται μέσο του Blockchain στην κατάλληλη συσκευή [31].

- Στον κλάδο των χρηματικών συναλλαγών: Αυτό είναι ένα σενάριο χρήσης που υπάρχει ήδη, και υλοποιείται με κρυπτονομίσματα, όμως η εφαρμογή του είναι ελάχιστη και περιορίζεται αρκετά. Όμως αποτελεί μία εναλλακτική η οποία παρέχει αρκετές δυνατότητες και σε πελάτες και σε επιχειρηματίες στην οποία δεν εμπλέκονται τρίτοι και στην οποία οι συναλλαγές εκτελούνται άμεσα[31] και όχι μετά από τρείς μέρες -ή όσο χρειάζεται η τράπεζα του πωλητή για να επεξεργαστεί την συναλλαγή-.

Το συμπέρασμα είναι ότι τα Blockchain και τα Smart Contracts έχουν αρκετές μελλοντικές εφαρμογές και αν το Ethereum μείνει στην κορυφή των εξελίξεων του τομέα μπορεί να προσφέρει τα κατάλληλα εργαλεία στους προγραμματιστές για να αξιοποιήσουν αυτή την ευκαιρία που μπορεί να δημιουργήσει νέες και καινοτόμες πρακτικές στις βιομηχανίες και στην καθημερινότητα των ανθρώπων.


*Εικόνα 12: Μερικές από τις μελλοντικές εφαρμογές που μπορεί να προσφέρει η τεχνολογία των Blockchain σε συνδυασμό με την τεχνολογία των Smart Contracts[20].*

## 4.5 Μελέτη περίπτωσης: Ψηφιακές εκλογές με το σύστημα Remix Ballot και πώς υλοποιείται με το Ethereum IDE και την βιβλιοθήκη Solidity

Στην παρούσα ενότητα θα παρουσιάσουμε ένα παράδειγμα που παρέχει η πλατφόρμα του Ethereum η οποία είναι μία υλοποιημένη μορφή πλήρως ψηφιοποιημένης πλατφόρμας ψηφοφορίας και εκλογών στο Ethereum. Αυτό το παράδειγμα υλοποιείται με το περιβάλλον ανάπτυξης του Ethereum και με την βιβλιοθήκη Solidity. Στο παραπάνω κεφάλαιο θα εξηγήσουμε πλήρως τον κώδικα που αναπτύσσεται στην πλατφόρμα Ethereum για να υλοποιηθεί το παράδειγμα των «ψηφιακών εκλογών». Σε αυτή το το παράδειγμα περιλαμβάνονται δομές δεδομένων και λειτουργίες οι οποίες όταν συνδυαστούν παράγουν την συγκεκριμένη λειτουργικότητα [22].

```
struct vote{
address voterAddress;
bool choice;
}
```

Η παραπάνω δομή αναπαριστά την ψήφο που έχει πραγματοποιηθεί. Σε αυτή την δομή διατηρείται η -κρυπτογραφημένη- διεύθυνση του ψηφοφόρου και τι ψήφισε [22].

```
struct voter{
string voterName;
bool voted;
}
```

Αυτή η δομή αναπαριστά τον κόμβο που έχει το δικαίωμα να συμμετάσχει στην διαδικασία της ψηφοφορίας. Περιλαμβάνει το όνομα του ψηφοφόρου και αν έχει ψηφίσει ή όχι[22].

```
mapping(uint => vote) private votes;
mapping(address => voter) public voterRegister;
```

Αυτή η διαδικασία συσχετίζει τα στοιχεία του ψηφοφόρου που είναι δημόσια με την ψήφο η οποία είναι απόρριτη. Με αυτό το τρόπο υπάρχει συσχετισμός του δικαιώματος της ψήφου χωρίς να μπορεί να ταυτοποιηθεί το περιεχόμενο της ψήφου [22].

```
uint private countResult = 0;
```

Η παραπάνω μεταβλητή διατηρεί το σύνολο τον ψήφων που έχουν πραγματοποιηθεί [22].

```
uint public finalResult = 0;
uint public totalVoter = 0;
uint public totalVote = 0;
```

Οι παραπάνω μεταβλητές λαμβάνουν τιμές μόλις κλείσει η ψηφιακή κάλπη και περιλαμβάνουν δεδομένα όπως ο αριθμός τον θετικών ψήφων, ο αριθμός των ψηφοφόρων που συμμετείχαν και το σύνολο όλων των ψήφων [22].

```
address public ballotOfficialAddress;
string public ballotOfficialName;
string public proposal;
```

Οι παραπάνω μεταβλητές διατηρούν βασικές πληροφορίες για την ψηφοφορία όπως το θέμα για το οποίο γίνεται η ψηφοφορία και η κρυπογραφική διεύθυνση όπου βρίσκεται η ψηφιακή κάλπη [22].

```
enum State { Created, Voting, Ended }
State public state;
```

Αυτή είναι η μεταβλητή κατάστασης που βρίσκεται η ψηφοφορία. Μπορεί να λάβει τρείς από τις παρακάτω καταστάσεις [22]:

- Έχει δημιουργηθεί αλλά δεν διενεργείται ψηφοφορία - Created - .
- Είναι σε κατάσταση διενεργείας της ψηφοφορίας - Voting - .
- Έχει ολοκληρωθεί η ψηφοφορία - Ended -.

```
constructor(
string memory _ballotOfficialName,
string memory _proposal) public {
ballotOfficialAddress = msg.sender;
ballotOfficialName = _ballotOfficialName;
proposal = _proposal;
```

```
state = State.Created;
}
```

Αυτή η μέθοδος αρχικοποιεί την ψηφιακή κάλπη, θέτει το θέμα για την ψηφοφορία και θέτει την κατάσταση σε «δημιουργημένη κάλπη» [22].

```
function addVoter(address _voterAddress, string memory _voterName)
public
inState(State.Created)
onlyOfficial
{
voter memory v;
v.voterName = _voterName;
v.voted = false;
voterRegister[_voterAddress] = v;
totalVoter++;
emit voterAdded(_voterAddress);
}
```

Η παραπάνω μέθοδος προσθέτει και δίνει δικαίωμα σε κόμβο να συμμετέχει στην διαδικασίας της ψηφοφορίας [22].

```
modifier inState(State _state) {
require(state == _state);
_;
}
```

Η παραπάνω μέθοδος ελέγχει αν η κάλπη βρίσκεται στην σωστή κατάσταση και μπορεί να αξιοποιηθεί από το πρόγραμμα για την σωστή λειτουργία του [22].

```
modifier onlyOfficial() {
require(msg.sender ==ballotOfficialAddress);
_;
}
```

Η παραπάνω μέθοδος ελέγχει αν ο χρήστης/κόμβος είναι εξουσιοδοτημένος με κάποια διαχειριστικά δικαιώματα -chairman- και μπορεί να χρησιμοποιηθεί για τις λειτουργίες οι οποίες πρέπει να γίνονται μόνο από συγκεκριμένους εξουσιοδοτημένους κόμβους [22].

```
function startVote()
public
inState(State.Created)
onlyOfficial
{
state = State.Voting;
emit voteStarted();
}
```

Η συγκεκριμένη μέθοδος εκκινεί την εκλογική διαδικασία και μπορεί να εκτελεσθεί μόνο από κόμβο που έχει το κατάλληλο δικαίωμα -πχ chairman- [22].

```
function doVote(bool _choice)
public
inState(State.Voting)
```

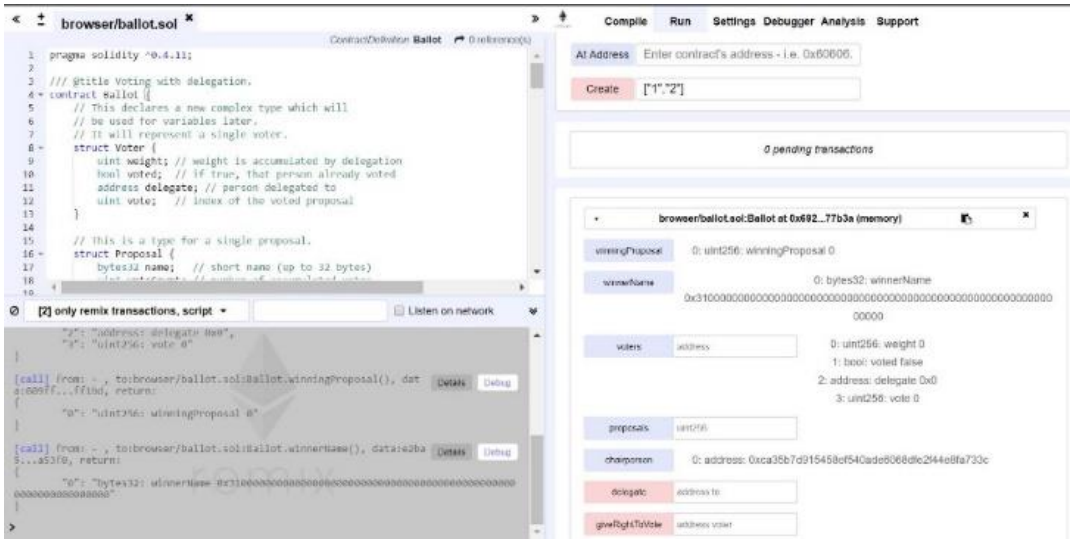
```
returns (bool voted)
{
bool found = false;

if (bytes(voterRegister[msg.sender].voterName).length != 0
&& !voterRegister[msg.sender].voted){
voterRegister[msg.sender].voted = true;
vote memory v;
v.voterAddress = msg.sender;
v.choice = _choice;
if (_choice){
countResult++; //counting on the go
}
votes[totalVote] = v;
totalVote++;
found = true;
}
emit voteDone(msg.sender);
return found;
}
```

Με την παραπάνω μέθοδο οι ψηφοφόροι μπορούν να καταθέσουν την ψήφο τους. Με την προϋπόθεση ότι έχουν λάβει το δικαίωμα από τον chairman και δεν έχουν ήδη ψηφίσει [22].

```
function endVote()
public
inState(State.Voting)
onlyOfficial
{
state = State.Ended;
finalResult = countResult; //move result from private countResult
to public finalResult
emit voteEnded(finalResult);
}
```

Αυτή αποτελεί την τελευταία μέθοδο του παραδείγματος, αυτή εκτελείται από εξουσιοδοτημένους κόμβους και σηματοδοτεί την ολοκλήρωσης της εκλογική διαδικασίας και την παρουσίαση του τελικού αποτελέσματος [22].

Το παραπάνω παράδειγμα αποτελεί ένα δείγμα για το πόσο εκτενείς είναι οι εφαρμογές που μπορούν να υλοποιηθούν στο Ethereum καθώς και σε ποιούς τομείς θα μπορούσε αυτό και γενικότερα η τεχνολογία Blockchain να δημιουργήσει βελτιωμένες πρακτικές και καινοτομίες. Το παράδειγμα υπάρχει στην σελίδα του Remix Ethereum [10] και παρέχει ένα live εργαλείο, όπως φαίνεται και στην εικόνα 13, για την εκτέλεση παραδειγμάτων αλλά και τροποποιημένου κώδικα για την πραγματοποίηση δοκιμών.




*Εικόνα 13: Παράδειγμα του Remix Ballot με το σχετικό User Interface, μέσο της πλατφόρμας Ethereum[10].*

# **ΚΕΦΑΛΑΙΟ 5:** ΕΠΙΛΟΓΟΣ

Ο σκοπός της παρούσας εργασίας ήταν η σύγκριση των δύο δημοφιλέστερων πλατφορμών ανάπτυξης εφαρμογών Blockchain-Ethereum και Hyperledger Fabric- για την εύρεση της βέλτιστης η οποία ανταποκρίνεται στις ανάγκες των προγραμματιστών. Με βάση την παραπάνω έρευνα και με βάση τις λειτουργίες και τα εργαλεία που παρέχει η πλατφόρμα Ethereum, καθώς την πληρότητα που παρέχει σε εργαλέια για την χρήση των Smart Contracts, συμπεραίνουμε ότι το Ethereum είναι το βέλτιστο εργαλείο για την ανάπτυξη εφαρμογών που υλοποιούν την τεχνολογία Blockchain. Ίσως στο μέλλον να υπάρξει κάποια νέα πλατφόρμα η οποία θα παρέχει πιο πλήρες περιβάλλον για προγραμματιστές που επιθυμούν να υλοποιήσουν εφαρμογές οι οποίες να αξιοποιούν την τεχνολογία Blockchain.

## ΒΙΒΛΙΟΓΡΑΦΙΑ


[1] H. ANAR, «Consensus Algorithms: The Root Of The Blockchain Technology,» 101 Blockchains, 25 Αυγούστου 2018. [Ηλεκτρονικό]. Available: https://101blockchains.com/consensus-algorithms-blockchain/. [Πρόσβαση 25 Οκτωβρίου 2018].

[2] W. Article, «Byzantine Fault,» Wikipedia, [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Byzantine_fault. [Πρόσβαση 26 Οκτωβρίου 2019].

[3] B. Asolo, «Double-Spending Explained,» MYCRYPTOPEDIA, 21 Δεκεμβρίου 2018. [Ηλεκτρονικό]. Available: https://www.mycryptopedia.com/double-spending-explained/. [Πρόσβαση 26 Οκτωβρίου 2019].

[4] M. T. C. F. Aziz, «GUIDE TO CONSENSUS ALGORITHMS: WHAT IS CONSENSUS MECHANISM?,» masterthecrypto, [Ηλεκτρονικό]. Available: https://masterthecrypto.com/guide-to-consensus-algorithms-what-is-consensus-mechanism/. [Πρόσβαση 26 Οκτωβρίου 2019].

[5] P. BAJPAI, «Bitcoin Vs Ethereum: What's the Difference?,» Investopedia, 03 Νοεμβρίου 2019. [Ηλεκτρονικό]. Available: https://www.investopedia.com/articles/investing/031416/bitcoin-vs-ethereum-driven-different-purposes.asp. [Πρόσβαση 04 Νοεμβρίου 2019].

[6] Blockgeeks, «Hyperledger vs Ethereum Training: Which one is better?,» BlockGeeks, Δεκέμβριος 2018. [Ηλεκτρονικό]. Available: https://blockgeeks.com/guides/hyperledger-vs-ethereum/. [Πρόσβαση 04 Νοεμβρίου 2019].

[7] J. Buntinx, «What is Proof-of-Weight,» NullTX, 11 Αυγούστου 2018. [Ηλεκτρονικό]. Available: https://nulltx.com/what-is-proof-of-weight/. [Πρόσβαση 25 Οκτωβρίου 2019].

[8] E. Ceylan, «Customizing and Using Multisig Contracts For Contract Executions,» Medium, 02 Αυγούστου 2018. [Ηλεκτρονικό]. Available: https://medium.com/coinmonks/customizing-and-using-multisig-contracts-for-other-contract-executions-9698fbb6950f. [Πρόσβαση 08 Νοεμβρίου 2019].

[9] B. Curran, «What is Proof of Authority Consensus? Staking Your Identity on The Blockchain,» Demand Solutions News, Ιούλιος 2018. [Ηλεκτρονικό]. Available: https://demandsolutionsnews.com/proof-of-authority/. [Πρόσβαση 20 Οκτωβρίου 2019].

[10] Ethereum, «Remix - Ethereum IDE,» Ethereum, [Ηλεκτρονικό]. Available: https://remix.ethereum.org/#optimize=false&evmVersion=null&version=soljson-v0.5.12+commit.7709ece9.js. [Πρόσβαση 18 Σεπτεμβρίου 2019].

[11] P. Fersht, «The top 5 enterprise blockchain platforms you need to know about,» Horses for Sources, 16 Μαρτίου 2018. [Ηλεκτρονικό]. Available: https://www.horsesforsources.com/top-5-blockchain-platforms_031618. [Πρόσβαση 05 Νοεμβρίου 2019].

[12] F. Foundation, «An Introduction to DAGs and How They Differ From Blockchains,» Medium, 20 Ιουλίου 2018. [Ηλεκτρονικό]. Available: https://medium.com/fantomfoundation/an-introduction-to-dags-and-how-they-differ-from-blockchains-a6f703462090. [Πρόσβαση 20 Οκτωβρίου 2019].

[13] A. Hertig και M. Kuznetsov, «Ethereum 101,» coindesk, 26 Σεπτεμβρίου 2019. [Ηλεκτρονικό]. Available: https://www.coindesk.com/learn/ethereum-101/what-is-ethereum. [Πρόσβαση 01 Νοεμβρίου 2019].

[14] P. Hooda, «practical Byzantine Fault Tolerance(pBFT),» Geeks for Geeks, [Ηλεκτρονικό]. Available: https://www.geeksforgeeks.org/practical-byzantine-fault-tolerancepbft/. [Πρόσβαση 19 Οκτωβρίου 2019].

[15] O. Hryniuk, «Ethereum Smart-Contract Best Practices,» 7AltCoins, [Ηλεκτρονικό]. Available: https://7altcoins.com/academy/guides/ethereum-smart-contract-best-practices/. [Πρόσβαση 06 Νοεμβρίου 2019].

[16] Y. Jia, «Demystifying Hashgraph: Benefits and Challenges,» HACKERNOON, 8 Δεκεμβρίου 2017. [Ηλεκτρονικό]. Available: https://hackernoon.com/demystifying-hashgraph-benefits-and-challenges-d605e5c0cee5. [Πρόσβαση 25 Οκτωβρίου 2019].

[17] T. Keski-Valkama και E. Patronen, «Blockchains Tech Talk,» CYBERCOM GROUP, 13 Φεβρουαρίου 2017. [Ηλεκτρονικό]. Available: https://www.cybercom.com/About-Cybercom/Blogs/the-connected-world/blockchain-tech-talk/. [Πρόσβαση 28 Οκτωβρίου 2019].

[18] A. Kore, «Implementing a simple ‘proof of work’ algorithm for the Blockchain,» Medium, 20 Ιανουαρίου 2018. [Ηλεκτρονικό]. Available: https://cryptocurrencyhub.io/implementing-a-simple-proof-of-work-algorithm-for-the-blockchain-bdcd50faac18. [Πρόσβαση 28 Οκτωβρίου 2019].

[19] N. Maria, «DAG : A Buzz or Breakthrough?,» HACKERNOON, [Ηλεκτρονικό]. Available: https://hackernoon.com/dag-a-buzz-or-breakthrough-9b433d0b5424. [Πρόσβαση 27 Οκτωβρίου 2019].

[20] S. Menking, «THE AMATEUR SOCIETY – BLOCKCHAIN APPLICATIONS – 07.03.2017,» The Hagmann Report, 04 Ιουλίου 2017. [Ηλεκτρονικό]. Available: https://www.hagmannreport.com/the-amateur-society-blockchain-applications-07-03-2017/. [Πρόσβαση 08 Νοεμβρίου 2019].

[21] PWC, «10 Challenges to the Adoption of Smart Contracts,» PWC, 08 Ιουνίου 2018. [Ηλεκτρονικό]. Available: https://blog.pwc.lu/smart-contracts-adoption-challenges/. [Πρόσβαση 07 Νοεμβρίου 2019].

[22] J. Ng, «Voting on a Blockchain: Solidity Contract Codes Explained,» Medium, 01 Μαΐου 2019. [Ηλεκτρονικό]. Available: https://medium.com/coinmonks/voting-on-a-blockchain-solidity-contract-codes-explained-c677996d94f2. [Πρόσβαση 08 Νοεμβρίου 2019].

[23] A. Rosic, «What is Ethereum? [The Most Updated Step-by-Step-Guide!],» BlockGeeks, 2016. [Ηλεκτρονικό]. Available: https://blockgeeks.com/guides/ethereum/. [Πρόσβαση 01 Νοεμβρίου 2019].

[24] M. Rouse, «Definition: consensus algorithm,» WhatIs.com, Αύγουστος 2017. [Ηλεκτρονικό]. Available: https://whatis.techtarget.com/definition/consensus-algorithm. [Πρόσβαση 26 Οκτωβρίου 2019].

[25] H. Sachdeva, «Hacking ethereum to inect our own consensus algorithm Part 2,» Talentica, 17 Αυγούστου 2018. [Ηλεκτρονικό]. Available: https://www.talentica.com/blogs/hacking-ethereum-to-inject-our-own-consensus-alogrithm-part-2/. [Πρόσβαση 05 Νοεμβρίου 2019].

[26] H. Sachdeva, «Hacking ethereum to inject our own consensus algorithm Part 1,» Talentica, 23 Ιουλίου 2018. [Ηλεκτρονικό]. Available: https://www.talentica.com/blogs/hacking-ethereum-to-inject-our-own-consensus-algorithum-part-1/. [Πρόσβαση 05 Νοεμβρίου 2019].

[27] A. Singh, «What is Blockchain Consensus Algorithm,» Official Hacker, 16 Ιουνίου 2019. [Ηλεκτρονικό]. Available: https://officialhacker.com/consensus-algorithm/. [Πρόσβαση 25 Οκτωβρίου 2019].

[28] M. Sotnichek και M. Yatsenko, «5 Security Tips for Writing Smart Contracts,» apriorit, 19 Νοεμβρίου 2018. [Ηλεκτρονικό]. Available: https://www.apriorit.com/dev-blog/581-security-tips-for-smart-contracts. [Πρόσβαση 07 Νοεμβρίου 2019].

[29] E. Tan, «Types Of Consensus Protocols used in Blockchains,» Hackernoon, 14 Οκτωβρίου 2019. [Ηλεκτρονικό]. Available: https://hackernoon.com/types-of-consensus-protocols-used-in-blockchains-6edd20951899. [Πρόσβαση 26 Οκτωβρίου 2019].

[30] B. Technologies, «What is Consensus Algorithm In Blockchain & Different Types Of Consensus Models,» Medium, 14 Μαΐου 2018. [Ηλεκτρονικό]. Available: https://medium.com/@BangBitTech/what-is-consensus-algorithm-in-blockchain-different-types-of-consensus-models-12cce443fc77. [Πρόσβαση 27 Οκτωβρίου 2019].

[31] S. Williams, «20 Real-World Uses for Blockchain Technology,» The Motley Fool, 11 Απριλίου 2018. [Ηλεκτρονικό]. Available: https://www.fool.com/investing/2018/04/11/20-real-world-uses-for-blockchain-technology.aspx. [Πρόσβαση 08 Νοεμβρίου 2019].