Πανεπιστημιο Πειραιως
Τμημα Ψηφιακων Συστηματων

Μηχανισμός βέλτιστης διαχείρισης υπηρεσιών υπολογιστικού νέφους βάσει επιχειρηματικών μοντέλων και αναγκών ελαστικότητας

Διπλωματική διατριβή για το
Π.Μ.Σ. «Διδακτική της Τεχνολογίας και Ψηφιακών Συστημάτων»
του

Νικολάου Χέλμη

Επιβλέπων: Μαρίνος Θεμιστοκλέους

Πειραιάς, Ιούλιος 2014

Ευχαριστίες

Θα ήθελα να ευχαριστήσω από τα βάθη της καρδιάς μου, τον επιβλέποντα καθηγητή μου κ. Μαρίνο Θεμιστοκλέους για το ενδιαφέρον που έδειξε, για τις πολύτιμες συμβουλές του και για την ιδιαίτερη στήριξη που μου παρείχε κατά την διάρκεια αυτής της πορείας μου.

Επίσης, θα ήθελα να ευχαριστήσω όλους τους συναδέλφους μου στην ερευνητική ομάδα με τους οποίους συνεργάστηκα άψογα και επιτυχώς όλο αυτό το διάστημα. Ιδιαίτερες ευχαριστίες ωστόσο θα ήθελα να απευθύνω στους Δημοσθένη Κυριαζή, Μιχαήλ Φιλιππάκη και στο φίλο μου Παναγιώτη Νικητόπουλο με τους οποίους μοιραστήκαμε τις πάρα πολλές ώρες της ερευνητικής εργασίας.

Τέλος, θα ήθελα να ευχαριστήσω τους γονείς μου και την αδερφή μου που πίστεψαν σε εμένα και τις επιλογές μου.

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΑ

Περίληψη

Σχολή Τεχνολογιών Πληροφορικής και Επικοινωνιών
Τμήμα Ψηφιακών Συστημάτων

Π.Μ.Σ. «Διδακτική της Τεχνολογίας και Ψηφιακών Συστημάτων»

Νικόλαος Χέλμης

Τα τελευταία χρόνια το μοντέλο της Νεφοϋπολογιστικής έχει κερδίσει το ενδιαφέρον τόσο του επιχειρηματικού όσο και του ακαδημαϊκού κόσμου. Η αυξανόμενη δημοτικότητα του μοντέλου οφείλεται στην ικανότητα του να επιτρέπει την ταχεία και αποτελεσματική πρόσβαση σε μεγάλες δεξαμενές εικονικών πόρων και υπηρεσιών που τροφοδοτούνται δυναμικά ώστε να προσαρμόζονται σε μεταβλητό φόρτο εργασίας. Οι πόροι συνήθως αξιοποιούνται από ένα pay-as-you-go μοντέλο τιμολόγησης, με το κόστος χρήσης ενός στοιχείου της υποδομής να εξαρτάται από τους πόρους που καταναλώθηκαν. Προς αυτή την κατεύθυνση, το μοντέλο της Νεφοϋπολογιστικής προσφέρεται για την αυτόματη κλιμάκωση και αποκλιμάκωση εφαρμογών, ώστε να ανταποκριθεί στις ανάγκες των χρηστών, καθιστώντας εύκολη τη γρήγορη προσαρμογή των πόρων στο φόρτο εργασίας ελαχιστοποιώντας το κόστος χρήσης επιπλέον πόρων.

Οι πάροχοι εφαρμογών αποσκοπούν στη μεγιστοποίηση της πελατειακής τους βάσης εξετάζοντας το κόστος. Προς αυτή την κατεύθυνση, απαιτούνται επιχειρηματικά μοντέλα που προτείνουν τρόπους για την προσέλκυση πελατών λαμβάνοντας υπόψη περιορισμούς κόστους. Στόχος είναι η βελτιστοποίηση της απόδοσης της "επένδυσης" σε πόρους συγκριτικά με τον προσδοκώμενο αριθμό πελατών. Παρ' όλα αυτά ο προσδοκώμενος αριθμός πελατών είναι άμεσα συνδεδεμένος με την παρεχόμενη ποιότητα υπηρεσιών (Quality of Service) και την ποιότητα της εμπειρίας των τελικών χρηστών (Quality of Experience).

Η παρούσα ερευνητική εργασία έρχεται να προτείνει ένα δυναμικό μηχανισμό ο οποίος είναι αποδοτικός για την επιχείρηση αναφορικά τόσο με τους τρέχοντες χρήστες όσο και με τους προσδοκώμενους συγκριτικά με τις ανάγκες σε πόρους. Σημείο κλειδί αποτελεί η αντιστοίχιση των δυναμικών επιχειρηματικών μοντέλων με πόρους βάσει των αναγκών.

Θεματική περιοχή: Νεφοϋπολογιστική
Λέξεις κλειδιά: Νεφοϋπολογιστική, Ελαστικότητα, Επεκτασιμότητα, Ελαστικός Μηχανισμός, Επιχειρηματικό Μοντέλο

UNIVERSITY OF PIRAEUS

Abstract

School of Information and Communication Technologies
Department of Digital Systems

Postgraduate Programme ”Technology Education & Digital Systems”

Nikolaos Chelmis

Nowadays, interest on Cloud Computing as a technical and business best practice has grown to great length. There is a huge pool of available cloud applications and services offered to end users. As application requirements, reflected to resources requirements (i.e. network, storage, computing capacity),are set by the application provider, a key issue relates to the resulting elasticity needs and their modeling. In addition to elasticity needs, application providers aim to maximize their customer base while considering the associated costs. To this end, business models are needed in order to attract customers while considering cost constraints. Their aim is to optimize the performance of the “investment” for resources compared to the expected number of customers. Nevertheless, the latter is directly linked to the provided quality of service and users’ quality of experience. To this direction, in this thesis a mechanism that dynamically maps business models with the planned and estimated resources based on varying needs is presented.

Subject: Cloud Computing
Keywords: Cloud computing, Elasticity, Scalability, Elastic Mechanic, Business Model

Ευρετήριο Περιεχομένων

Ευρετήριο Σχημάτων

Σελ.

Ευρετήριο Πινάκων

Σελ.

Ευρετήριο Ορισμών

Σελ.

ΚΕΦΑΛΑΙΟ 1

Εισαγωγή

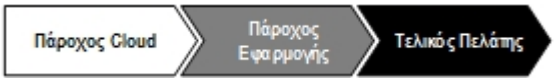
"Research is what I'm doing when I don't know what I'm doing."
˜ Wernher von Braun

Τα τελευταία χρόνια η Νεφοϋπολογιστική (cloud computing) έχει αναδειχθεί ως ένα επιτυχημένο μέσο παράδοσης υπολογιστικών λύσεων υπό τη μορφή υπηρεσίας (as a Service (aaS)). Η Νεφοϋπολογιστική αποτελεί την πλέον σύγχρονη τάση, αναφορικά με τους τρόπους που οι επιχειρηματικές υπηρεσίες πληροφορικής και τηλεπικοινωνιών διατίθενται στους καταναλωτές. Αυτό κατέστη εφικτό λόγω των πολλών τεχνολογικών μέσων που πλέον είναι διαθέσιμα καθώς και από τις συνεχόμενα εξελισσόμενα επιχειρηματικές στρατηγικές.

1.1 Ορισμός Προβλήματος

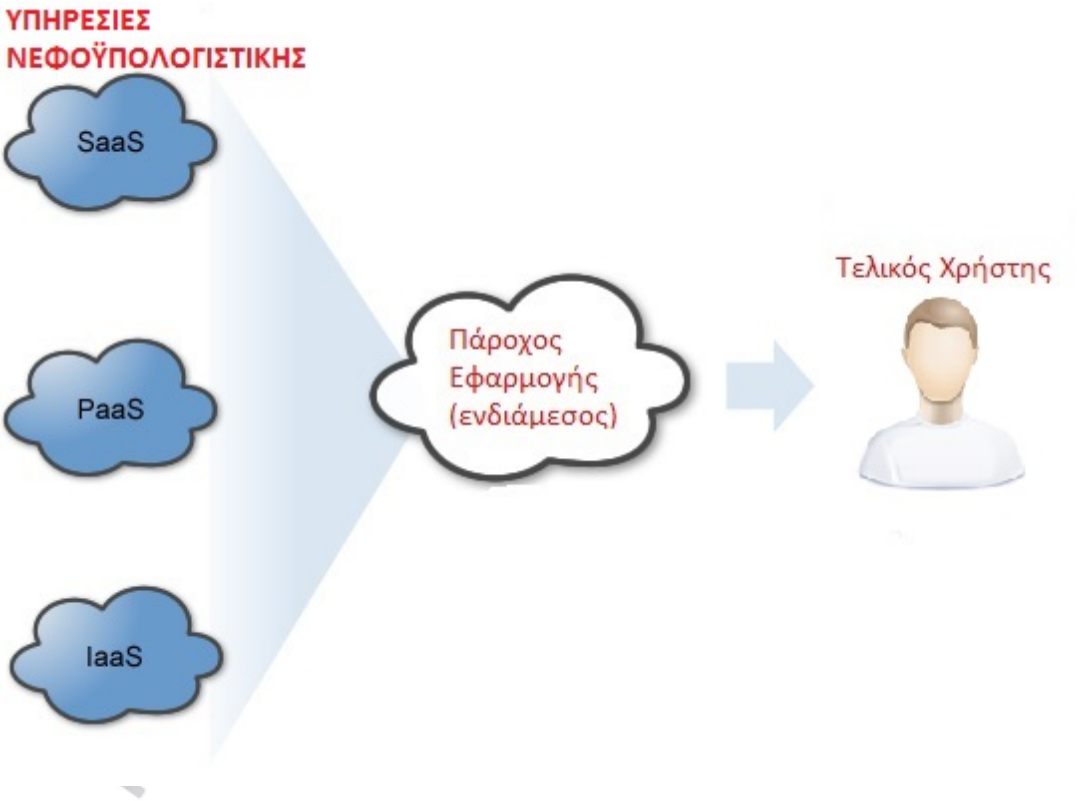
Η σύγχρονη επιχειρηματική τάση προβλέπει την παροχή υπηρεσιών από έναν συνδρομητή του υπολογιστικού νέφους στους πελάτες του όπως φαίνεται στο Σχήμα 1.1. Τα τρία βασικά μοντέλα που έχουν επικρατήσει είναι:

1. Απευθείας παροχή υπηρεσίας στον τελικό χρήστη (πελάτη). Πλέον γνωστό παράδειγμα αποτελεί η υπηρεσία Dropbox.
2. Χρήση υποδομών υπολογιστικού νέφους για τους εσωτερικούς σκοπούς του οργανισμού όπως για παράδειγμα το εσωτερικό δίκτυο μιας τράπεζας.
3. Χρήση υποδομών υπολογιστικού νέφους από μια επιχείρηση με σκοπό να διαθέσει την υπηρεσία της στους τελικούς πελάτες, όπως φαίνεται στο Σχήμα 1.2



Σχήμα 1.1: Αλυσίδα Αξίας

Η παρούσα ερευνητική εργασία εστιάζει στο τρίτο μοντέλο. Ο πάροχος της εφαρμογής που χρησιμοποιεί μια υποδομή υπολογιστικού νέφους με σκοπό να διαθέσει την υπηρεσία του, στη βιβλιογραφία αναφέρεται ως ενδιάμεσος (broker). Η υπηρεσία αυτή μπορεί να είναι οποιοδήποτε σύστημα λογισμικού που αποτελείται από ένα ή περισσότερα συστατικά. Η αρχιτεκτονική της εφαρμογής που παρέχεται θεωρείται γνωστή (ως προς τα συστατικά της, τις διεπαφές, τη λογική) και μπορεί να περιγραφεί με ακρίβεια όπως επίσης μπορούν να περιγραφούν και οι απαιτήσεις της σε πόρους υπό οποιοδήποτε φόρτο εργασίας.



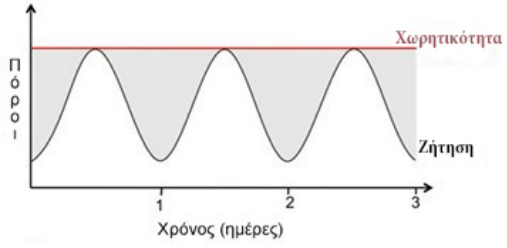
Σχήμα 1.2: Πάροχος Εφαρμογής ως Ενδιάμεσος

Καθώς οι απαιτήσεις της εφαρμογής αναφορικά με τους πόρους (δικτυακούς, αποθηκευτικούς και υπολογιστικούς) του υπολογιστικού νέφους διατυπώνονται από τον πάροχο της εφαρμογής, ένα βασικό ζήτημα αναφέρεται στις προκύπτουσες απαιτήσεις ελαστικότητας και τη μοντελοποίηση αυτών. Η ελαστικότητα και οι προαναφερθείσες απαιτήσεις εξαρτώνται από τις ακόλουθες παραμέτρους: (i) την εφαρμογή (για παράδειγμα χρήση πολλαπλών επεξεργαστών), (ii) από τη χρήση (για παράδειγμα μεταβλητός εκθετικά αυξανόμενος αριθμός τελικών χρηστών), και (iii) από τον πάροχο

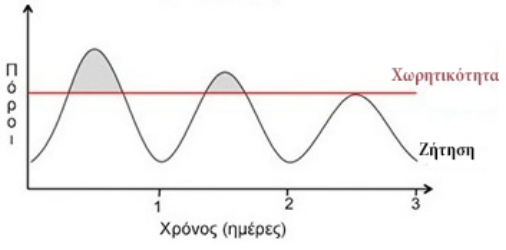
της υποδομής (για παράδειγμα διαθεσιμότητα πόρων). Με στόχο την ανάλυση των απαιτήσεων ελαστικότητας απαιτούνται μοντέλα ελαστικότητας που καλύπτουν τις παραπάνω παραμέτρους και επιτρέπουν τη χρησιμοποίηση των αποτελεσμάτων της ανάλυσης για την παροχή πόρων βάσει ζήτησης.

Πέραν των ζητημάτων ελαστικότητας, οι πάροχοι εφαρμογών στοχεύουν στη μεγιστοποίηση της πελατειακής τους βάσης εξετάζοντας παράλληλα το σχετικό κόστος. Προς αυτή την κατεύθυνση, απαιτούνται επιχειρηματικά μοντέλα που προτείνουν τρόπους για την προσέλκυση πελατών λαμβάνοντας υπόψη περιορισμούς κόστους. Στόχος είναι η βελτιστοποίηση της απόδοσης της "επένδυσης" σε πόρους συγκριτικά με τον προσδοκώμενο αριθμό πελατών. Παρ' όλα αυτά ο προσδοκώμενος αριθμός πελατών είναι άμεσα συνδεδεμένος με την παρεχόμενη ποιότητα υπηρεσιών (Quality of Service (QoS)) και την ποιότητα της εμπειρίας των τελικών χρηστών (Quality of Experience (QoE)). Για παράδειγμα αν οι προσφερόμενοι πόροι έπονται της ζήτησης, συγκεκριμένος αριθμός χρηστών θα έπρεπε να αναμένει τη διαθεσιμότητα αυτών των πόρων και ως εκ τούτου τη χρήση της υπηρεσίας.

Ακόμα κι εάν ο πάροχος της εφαρμογής μπορεί με ακρίβεια να προβλέψει το μέγιστο φόρτο εργασίας για την εφαρμογή του και έχει αποκτήσει τους απαιτούμενους πόρους με σκοπό να καλύψει το φόρτο αυτό τότε, καθ' όλη τη διάρκεια που εφαρμογή που διαθέτει λειτουργεί κάτω του μέγιστου φόρτου, μέρος των πόρων παραμένουν σε αδράνεια με αποτέλεσμα να σπαταλούνται όπως φαίνεται στο Σχήμα 1.3. Εάν ο μέγιστος φόρτος υποτιμηθεί από τον πάροχο της εφαρμογής, τότε είναι πιθανό επιπλέον χρήστες να μη χρησιμοποιήσουν την υπηρεσία. Ανενεργοί πόροι από ορθή ή μεγαλύτερη πρόβλεψη του φόρτου εργασίας είναι εύκολο να μετρηθούν οικονομικά η μη σωστή πρόβλεψη όμως είναι σαφώς δυσκολότερη. Χρήστες που δεν μπόρεσαν να χρησιμοποιήσουν την εφαρμογή πέραν του ότι δεν έφεραν εσοδα, είναι πιθανό να μην ξαναγυρίσουν ποτέ. Στο Σχήμα 1.4 παρουσιάζεται το ενδεχόμενο αυτό: Πολλοί χρήστες εγκαταλείπουν την εφαρμογή μόνιμα καθώς λαμβάνουν μιας κακής ποιότητας υπηρεσία, με αποτέλεσμα την αρνητική διαφήμιση για την υπηρεσία αυτή λόγω της κακής εμπειρίας των τελικών χρηστών και κατά συνέπεια τη διαρροή κερδών.

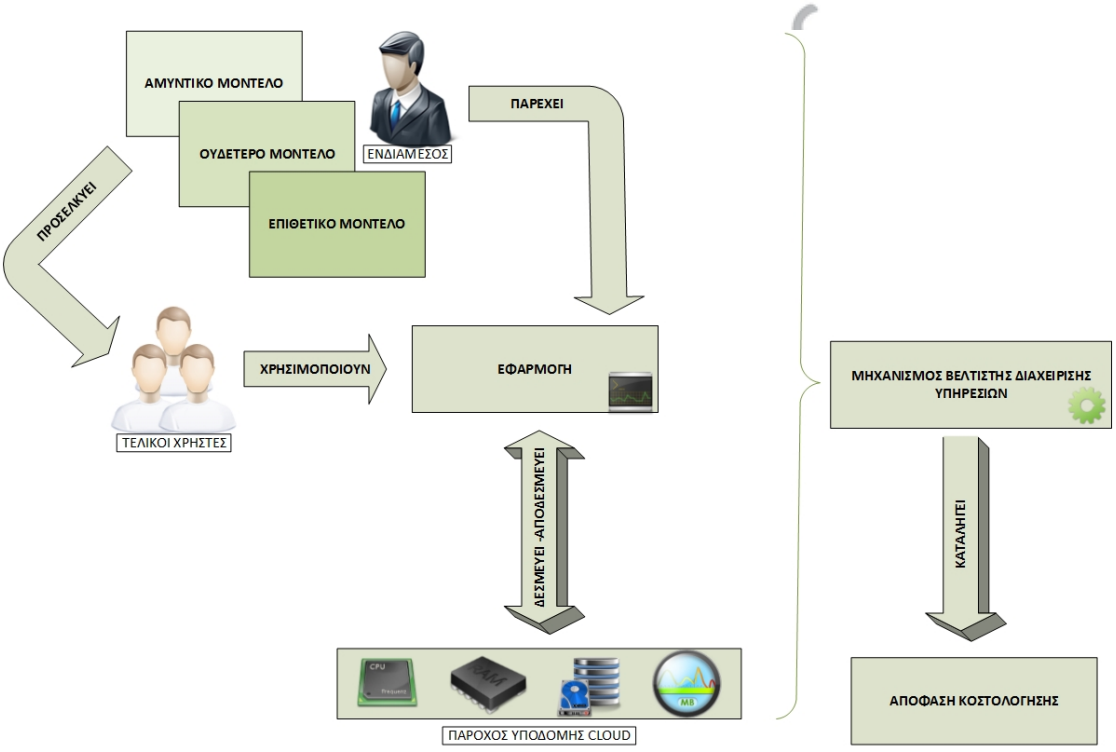


Σχήμα 1.3: Πόροι σε Αδράνεια



Σχήμα 1.4: Λάθος Πρόβλεψη

Βάσει των παραπάνω, απαιτείται ένα δυναμικό μοντέλο το οποίο είναι αποδοτικό για την επιχείρηση αναφορικά τόσο με τους τρέχοντες χρήστες όσο και με τους προσδοκώμενους συγκριτικά με τις ανάγκες ελαστικότητας. Σημαντικό σημείο αποτελεί η αντιστοίχιση δυναμικών επιχειρηματικών μοντέλων σε προβλεπόμενους πόρους βάσει αναγκών. Η αντιστοίχιση, που απεικονίζεται στο Σχήμα 1.5 πρέπει να επιτρέπει την επιλογή μοντέλων ελαστικότητας που θα ακολουθηθούν σε σχέση με το επιχειρηματικό μοντέλο (αμυντικό, επιθετικό, ουδέτερο), έχοντας ως αποτέλεσμα τη βέλτιστη διαχείριση υπηρεσιών.



Σχήμα 1.5: Εισαγωγή Μηχανισμού

1.2 Σκοπός

Σκοπός της παρούσας εργασίας είναι:

Η δημιουργία ενός μηχανισμού βέλτιστης διαχείρισης υπηρεσιών ο οποίος θα μπορεί να αντιστοιχίζει δυναμικά, τα προτεινόμενα από το πάροχο της εφαρμογής επιχειρηματικά μοντέλα με τους προβλεπόμενους πόρους.

Ο μηχανισμός υλοποιεί έναν αλγόριθμο ο οποίος αποδίδει με ακρίβεια το χρόνο απόκρισης μιας εφαρμογής βασισμένη στη κατανάλωση πόρων (στα πλαίσια της παρούσας ερευνητικής εργασίας αρχικά μοντελοποιήθηκαν Central Process Unit (CPU) και Random access memory (RAM)) που γίνεται από τον τρέχοντα αριθμό χρηστών.

Έχοντας τον μηχανισμό αυτό σαν εργαλείο, ο πάροχος της εφαρμογής θα μπορεί να κοστολογεί τους πελάτες του βάσει του επιχειρηματικού μοντέλου που ακολουθεί.

1.3 Αντικειμενικοί Στόχοι

Για την αντιμετώπιση της παραπάνω ερευνητικής πρόκλησης, της δημιουργίας ενός μηχανισμού βέλτιστης διαχείρισης υπηρεσιών, έχουν τεθεί οι εξής στόχοι:

Αντικειμενικός στόχος 1: **Κριτική ανασκόπηση βιβλιογραφίας:**
Αναφέρεται στη βιβλιογραφική μελέτη και ανάλυση, στα πλαίσια της παρούσας εργασίας και του προβλήματος, εννοιών σχετικών με τη Νεφοϋπολογιστική βάσει της βιβλιογραφίας. Επιπλέον στόχος είναι η συγκέντρωση των πλεονεκτημάτων που υπόσχεται η Νεφοϋπολογιστική και η αντιπαράθεση τους με τις ανησυχίες των χρηστών ώστε να γίνει εμφανής η ανάγκη ελαστικών μοντέλων.

Αντικειμενικός στόχος 2: **Ανάλυση θεμάτων ελαστικότητας και προσδιορισμός ανοικτών ερευνητικών περιοχών:**
Στοχεύει στην ανάλυση των θεμάτων που αφορούν την ελαστικότητα, τον ακριβή προσδιορισμό του όρου, την αποσαφήνιση του όρου από την έννοια της επεκτασιμότητας, την ανάδειξη της σημασίας της ελαστικότητας καθώς και την ανάλυση σύγχρονων τάσεων (state of the art) και παράθεση των ανοιχτών θεμάτων.

Αντικειμενικός στόχος 3: **Προσδιορισμός και ανάλυση επιχειρηματικών μοντέλων:**
Πρόταση τεχνοοικονομικών μοντέλων, βασισμένα στα σύγχρονα επιχειρηματικά μοντέλα, για κοστολόγηση με χρήση του μηχανισμού που θα παραχθεί.

Αντικειμενικός στόχος 4: **Πρόταση αρχιτεκτονικής μηχανισμού:**
Η περιγραφή και ανάλυση της αρχιτεκτονικής του μηχανισμού που προτείνει η παρούσα εργασία.

Αντικειμενικός στόχος 5: **Συλλογή δεδομένων και μοντελοποίηση:** Πραγματοποίηση πειραμάτων για συλλογή δεδομένων τα οποία εν συνεχεία θα μοντελοποιηθούν μαθηματικά ώστε να οδηγήσουν στη μαθηματική συνάρτηση.

Αντικειμενικός στόχος 6: **Αξιολόγηση:** Έλεγχος και αξιολόγηση: (i) της εγκυρότητας της μαθηματικής συνάρτησης και (ii) του μηχανισμού

1.4 Δομή Εργασίας

Η δομή της παρούσας εργασίας ακολουθεί τη μεθοδολογία όπως έχει περιγραφεί από τους Phillips and Pugh [1] και αποτελείται από τέσσερα δομικά στοιχεία:

1. **Background theory:** Εστιάζει στον ορισμό του προβλήματος (Κεφάλαιο 1) καθώς και σε βιβλιογραφικές έννοιες και αναφορές που χρησιμοποιούνται για να υποστηρίξουν το πεδίο του προβλήματος (Κεφάλαιο 2, Κεφάλαιο 3).
2. **Focal theory:** Σχετίζεται με την παραγωγή του εννοιολογικού μοντέλου καθώς και τα ερευνητικά ζητήματα (Κεφάλαιο 4, Κεφάλαιο 5).
3. **Data theory:** Αποτελείται από το σχεδιασμό της έρευνας, τη μέθοδο συλλογής των δεδομένων, την περιγραφή της διαδικασίας ανάλυσης των δεδομένων αυτών και την αξιολόγηση των αποτελεσμάτων (Κεφάλαιο 6, Κεφάλαιο 7).
4. **Novel contribution:** Γίνεται σύνοψη της έρευνας που πραγματοποιήθηκε, των αποτελεσμάτων και αναφορά σε πιθανές τροποποιήσεις για μελλοντική έρευνα (Κεφάλαιο 8).

Η δομή της παρούσας ερευνητικής εργασίας αναλύεται συνοπτικά στη συνέχεια και απεικονίζεται στο Σχήμα 1.6.

Κεφάλαιο 1: Εισαγωγή: Το κεφάλαιο αυτό αποσκοπεί στον σαφή ορισμό της ερευνητικής πρόκλησης που στοχεύει να λύσει η παρούσα εργασία. Επιπλέον ορίζεται ο σκοπός και οι αντικειμενικοί στόχοι, η επίτευξη των οποίων θα οδηγήσει στην επίλυση του προβλήματος.

Κεφάλαιο 2: Θεωρητικό Υπόβαθρο: Το κεφάλαιο αυτό χωρίζεται σε δύο ενότητες. Στην πρώτη ενότητα παρουσιάζεται η εξέλιξη της Νεφοϋπολογιστικής από τις αρχές εμφάνισης του όρου και αναλύονται τα βασικά εξελικτικά στάδια που την

έφεραν στη μορφή που έχει σήμερα. Στη δεύτερη ενότητα παρατίθενται και αναλύονται όλες οι βασικές βιβλιογραφικές έννοιες καθώς επίσης και οι βασικές τεχνολογίες που σχετίζονται με το ερευνητικό πρόβλημα.

Κεφάλαιο 3: Πλεονεκτήματα & Εμπόδια: Παρουσιάζονται τα βασικότερα πλεονεκτήματα τα οποία υπόσχεται ότι προσφέρει η Νεφοϋπολογιστική και αντιπαρατίθενται με εμπόδια που αντιμετωπίζουν οι χρήστες και τους αποτρέπουν από την υιοθέτηση της.

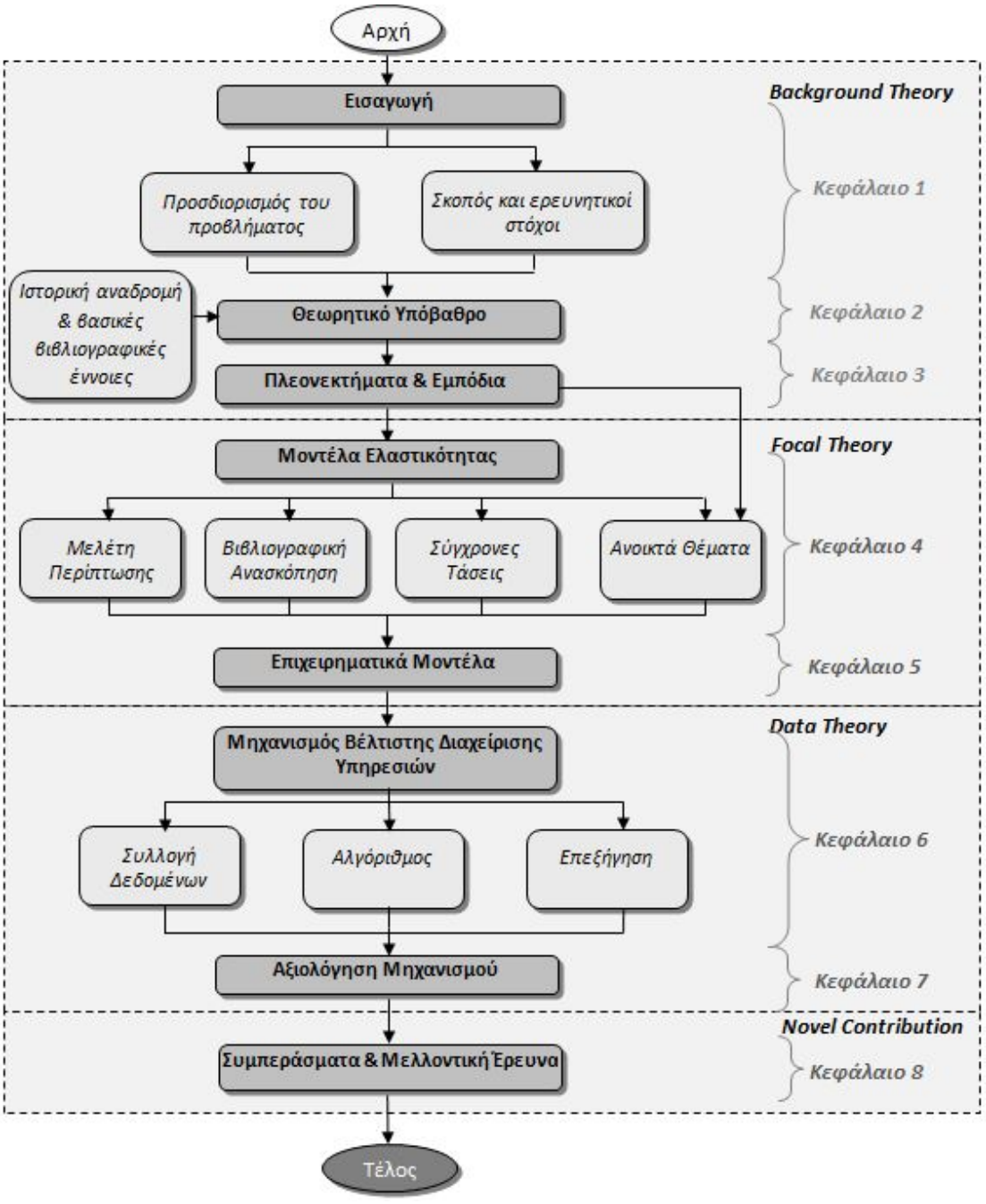
Κεφάλαιο 4: Μοντέλα Ελαστικότητας: Το κεφάλαιο αυτό αρχικά εστιάζει στην ανάδειξη της σημασίας της ελαστικότητας μέσα από ένα υποθετικό παράδειγμα καθώς και από τη γνωστότερη βιβλιογραφική μελέτη περίπτωσης, αυτής του Animoto. Εν συνεχεία βάσει της βιβλιογραφίας ορίζεται ο όρος της ελαστικότητας και αναλύονται οι βασικές πτυχές και διαστάσεις του ορισμού αυτού ενώ αποσαφηνίζεται η έννοια από τον όρο της επεκτασιμότητας. Το δεύτερο βασικό μέρος του κεφαλαίου καταγράφει τις σύγχρονες τάσεις τόσο σε εμπορικό επίπεδο όσο και σε ερευνητικά έργα. Τέλος αναλύονται τα ανοικτά θέματα που αφορούν την ελαστικότητα και σχετίζονται με την παρούσα ερευνητική εργασία.

Κεφάλαιο 5: Επιχειρηματικά Μοντέλα: Στο κεφάλαιο αυτό προτείνονται τρία βασικά τεχνοοικονομικά επιχειρηματικά μοντέλα: (i) Ουδέτερο, (ii) Αμυντικό και (iii) Επιθετικό. Τα μοντέλα αυτά θα χρησιμοποιούνται από το μηχανισμό που προτείνεται.

Κεφάλαιο 6: Μηχανισμός Βέλτιστης Διαχείρισης Υπηρεσιών: Στο κεφάλαιο αυτό αρχικά ορίζεται η αρχιτεκτονική δομή του μηχανισμού που θα προταθεί και τα βασικά του συστατικά. Στη συνέχεια παρατίθενται τα δομικά συστατικά των πειραμάτων που πραγματοποιηθήκαν, με σκοπό τη συλλογή των απαραίτητων δεδομένων, καθώς και ο τρόπος εκτέλεσης τους. Η επόμενη ενότητα παρουσιάζει τη μέθοδο με την οποία τα δεδομένα αυτά μοντελοποιήθηκαν μαθηματικά ώστε να παραχθεί η μαθηματική συνάρτηση.

Κεφάλαιο 7: Αξιολόγηση Μηχανισμού: Το κεφάλαιο αυτό παρουσιάζει τη διαδικασία αξιολόγησης του αλγορίθμου και του μηχανισμού που προτάθηκε. Τα αποτελέσματα της διαδικασίας παρατίθενται και αναλύονται.

Κεφάλαιο 8: Συμπεράσματα & Μελλοντική Έρευνα: Στο τελευταίο κεφάλαιο παρατίθενται τα συμπεράσματα από όλη τη διαδικασία εκπόνησης της παρούσας ερευνητικής εργασίας. Τέλος αναφέρονται ανοικτές ερευνητικές περιοχές και προτείνονται πιθανές τροποποιήσεις για τη παρούσα προσέγγιση.



Σχήμα 1.6: Δομή Εργασίας

ΚΕΦΑΛΑΙΟ 2

Θεωρητικό Υπόβαθρο

"The interesting thing about Cloud Computing is that we've redefined Cloud Computing to include everything that we already do.... I don't understand what we would do differently in the light of Cloud Computing other than change the wording of some of our ads."
˜Larry Ellison, 2012

2.1 Εισαγωγή

Ο όρος Νεφοϋπολογιστική (ή Υπολογιστικό Νέφος) έχει διεισδύσει στην καθημερινότητα πολλών ανθρώπων, εταιριών και οργανισμών. Για τον όρο αυτό έχουν διατυπωθεί αρκετοί ορισμοί χωρίς όμως να υπάρχει ένας κοινώς αποδεχτός καθώς χρησιμοποιείται για να περιγράψει μια ποικιλία διαφορετικών εννοιών της πληροφορικής που περιλαμβάνουν ένα μεγάλο αριθμό υπολογιστών συνδεδεμένους μεταξύ τους μέσω δικτύου [2] .

Το παρών κεφάλαιο χωρίζεται σε δύο ενότητες: (i) στην πρώτη ενότητα γίνεται μια ιστορική αναδρομή και παρουσιάζεται η εξέλιξη της Νεφοϋπολογιστικής ενώ (ii) στη δεύτερη γίνεται ανασκόπηση και ανάλυση των σχετικών βιβλιογραφικών εννοιών.

Στόχοι του κεφαλαίου αυτού είναι:

1. **Εξελικτική Πορεία:** Παρουσίαση της εξέλιξης της Νεφοϋπολογιστικής και των τεχνολογιών που την υποστηρίζουν από τις απαρχές της εμφάνισης του όρου μέχρι σήμερα.

2. **Βιβλιογραφικές Έννοιες:** Ανάλυση βασικών βιβλιογραφικών εννοιών και όρων σχετικών με το αντικείμενο της παρούσας εργασίας, που αφορούν τη Νεφοϋπολογιστική.

3. **Ορισμό:** Παράθεση και ανάλυση του επικρατέστερου βιβλιογραφικού ορισμού της Νεφοϋπολογιστικής καθώς και τη σε βάθος ανάλυση του ορισμού αυτού και των εννοιών του.

4. **Τεχνικές έννοιες:** Σε βάθος ανάλυση σχετικών με το αντικείμενο της παρούσας ερευνητικής εργασίας τεχνικών εννοιών και τεχνολογιών.

2.2 Ιστορική Αναδρομή

Αν και η πραγματική ιστορία της Νεφοϋπολογιστικής δεν είναι τόσο παλιά (οι πρώτες υπηρεσίες εμφανίστηκαν το 1999), η ιστορία του συνδέεται άμεσα με την ανάπτυξη της τεχνολογίας του Διαδικτύου, καθώς η χρήση υποδομών υπολογιστικού νέφους θεωρείται βέλτιστη πρακτική όσον αφορά τη μεγιστοποίηση του κέρδους για παρεχόμενες υπηρεσίες πληροφορικής και τηλεπικοινωνιών. Δεν είναι τυχαίο ότι στη καθημερινή χρήση ο όρος σύννεφο αποτελεί μια μεταφορά για το διαδίκτυο [3]. Η ίδια η προέλευση του όρου Νεφοϋπολογιστική είναι ασαφής.

Ο όρος σύννεφο χρησιμοποιείται συνήθως στην επιστήμη για να περιγράψει ένα μεγάλο σύνολο αντικειμένων που από μια απόσταση μοιάζει οπτικά με ένα σύννεφο και περιγράφει κάθε σύνολο πραγμάτων του οποίου τα στοιχεία δεν έχουν ελεγχτεί εις βάθος σε ένα συγκεκριμένο πλαίσιο. Στην επιστήμη των μαθηματικών ένας μεγάλος αριθμός σημείων σε ένα σύστημα συντεταγμένων θεωρείται ως νέφος. Στην επιστήμη της αστρονομίας, αστερισμοί που συνωστίζονται σε ένα σημείο στον ουρανό χαρακτηρίζονται ως νεφέλωμα (λατινική λέξη για το σύννεφο). Στην επιστήμη της φυσικής η αόριστη θέση των ηλεκτρόνιων γύρω από ένα ατομικό πυρήνα μοιάζει με σύννεφο στα μάτια ενός παρατηρητή.

Σε αναλογία με τις παραπάνω επιστήμες η λέξη σύννεφο χρησιμοποιήθηκε ως μεταφορά για το Διαδίκτυο και ένα τυποποιημένο πλέον σχήμα που μοιάζει με σύννεφο χρησιμοποιείται για να υποδηλώσει ένα δίκτυο σε τηλεφωνικά διαγράμματα και αργότερα για να απεικονίσει το ίδιο το Διαδίκτυο στα διαγράμματα δικτύων υπολογιστών. Ο όρος έγινε ιδιαίτερα δημοφιλής μετά το 2006 όπου η Amazon.com παρουσίασε το Elastic Compute Cloud [4].

Η βασική ιδέα της Νεφοϋπολογιστικής χρονολογείται από τη δεκαετία του 1950, όταν τα mainframes έγιναν διαθέσιμα σε πανεπιστήμια και επιχειρήσεις, προσβάσιμα μέσω

τερματικών υπολογιστών, γνωστά και ως στατικά τερματικά μιας και χρησιμοποιούνταν για επικοινωνίες αλλά δεν είχαν εσωτερικές δυνατότητες επεξεργασίας. Για να γίνει αποτελεσματικότερη η χρήση των mainframes, τα οποία ήταν αρκετά δαπανηρά, εξελίχθηκε μια πρακτική για να επιτρέπει σε πολλούς χρήστες να μοιράζονται τη φυσική πρόσβαση στον υπολογιστή από πολλαπλά τερματικά. Έτσι μειώθηκαν στο ελάχιστο οι χρόνοι στους οποίους τα mainframes παρέμεναν αδρανή και η επένδυση σε αυτά έγινε αποδοτικότερη.

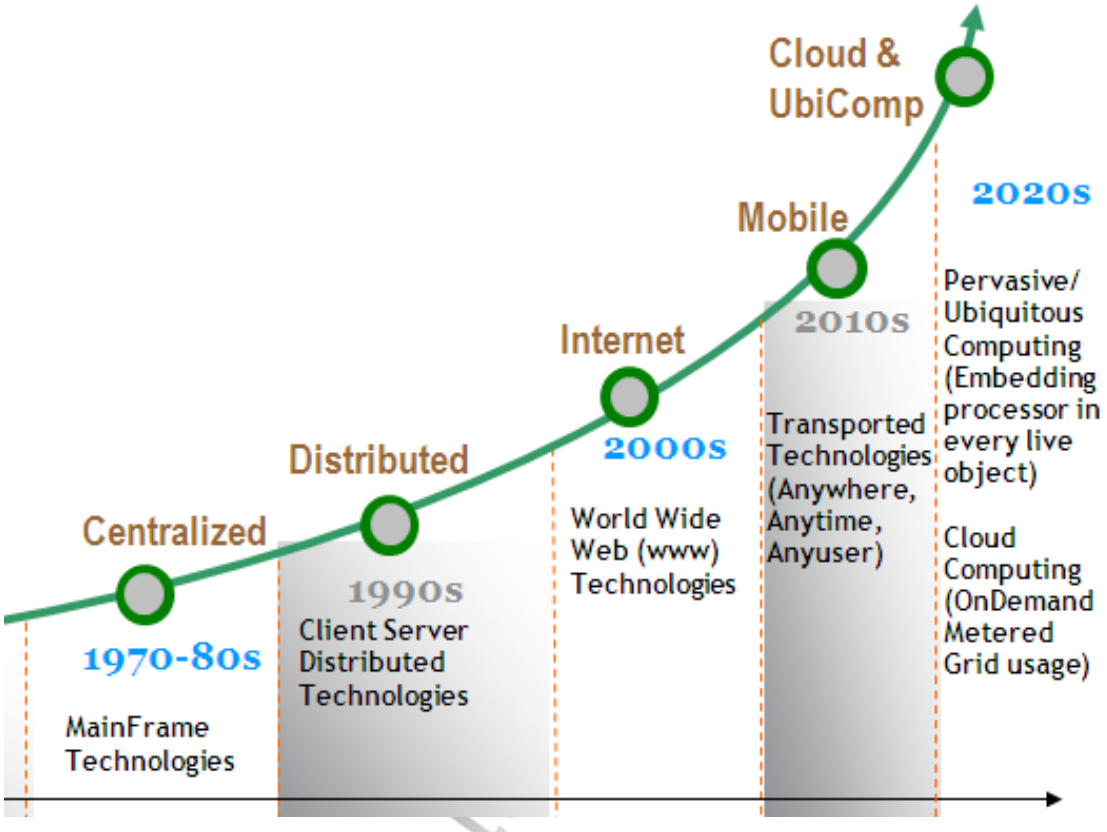
Σχεδόν όλα τα σύγχρονα χαρακτηριστικά της Νεφοϋπολογιστικής είχαν διερευνηθεί από τον Parkhill το 1966 [5]. Άλλοι μελετητές έχουν διατυπώσει τη θεωρία ότι η Νεφοϋπολογιστική έχει τις ρίζες της πίσω στο 1950 όταν ο Herb Grosch, διατύπωσε την άποψη ότι όλος ο κόσμος θα μπορούσε να λειτουργήσει με τερματικά (dumb terminals) που θα κινούνται πάνω από δεκαπέντε μεγάλα data center [6].

Στη δεκαετία του 1990, οι εταιρίες τηλεπικοινωνιών, που προηγουμένως είχαν αφιερωθεί στη παροχή point-to-point data circuits, ξεκίνησαν να παρέχουν εικονικά ιδιωτικά δίκτυα (Virtual Private Network (VPN)), με συγκρίσιμη ποιότητα υπηρεσιών αλλά αρκετά χαμηλότερο κόστος. Δρομολογώντας την κυκλοφορία με τέτοιο τρόπο ώστε να εξισορροπείται το φορτίο στους εξυπηρετητές, μπορούσαν να αξιοποιήσουν αποδοτικότερα το συνολικό εύρος ζώνης. Ξεκίνησαν να χρησιμοποιούν το σύμβολο του σύννεφου για να υποδηλώσουν τα όρια στα οποία ήταν οι ίδιοι υπεύθυνοι και που οι χρήστες ήταν υπεύθυνοι. Η Νεφοϋπολογιστική επεκτείνει το όριο αυτό καλύπτοντας τόσο τους εξυπηρετητές όσο και την υποδομή του ίδιου του δικτύου.

Η Amazon έπαιξε καθοριστικό ρόλο στην ανάπτυξη της Νεφοϋπολογιστικής με τον εκσυγχρονισμό των datacenter της, τα οποία όπως τα περισσότερα δίκτυα υπολογιστών εκείνη την εποχή, χρησιμοποιούσαν μόνο το 10% της χωρητικότητας τους ανά πάσα στιγμή, μόνο και μόνο για να αφήσει χώρο για τυχόν σημεία αιχμής. Διαπιστώνοντας ότι η νέα αρχιτεκτονική Νεφοϋπολογιστικής οδήγησε σε σημαντικές εσωτερικές βελτιώσεις της αποτελεσματικότητας όπου μικρές ομάδες (που πήραν την ονομασία “two-pizza teams”, δηλαδή ομάδες αρκετά μικρές ώστε να χορτάσουν με δύο πίτσες [7]) θα μπορούσαν να προσθέσουν νέες λειτουργίες ευκολότερα και γρηγορότερα, η Amazon ξεκίνησε μια νέα προσπάθεια ανάπτυξης προϊόντος με στόχο τη παροχή υπηρεσιών Νεφοϋπολογιστικής σε εξωτερικούς πελάτες και ίδρυσε το Amazon Web Services (AWS) [8] το 2006 [9].

Η στροφή του διαδικτύου προς όλο και περισσότερες υπηρεσίες περιγράφτηκε ως “Dynamic Web” [10]. Η διαθεσιμότητα δικτύων υψηλής χωρητικότητας, του χαμηλού κόστους υπολογιστών καθώς και συσκευών αποθήκευσης που υπάρχει σήμερα σε συνδυασμό με την υιοθέτηση της εικονοποίησης υλικού και της υπηρεσιοστρεφούς αρχιτεκτονικής έχουν οδηγήσει στη ραγδαία αύξηση χρήσης της Νεφοϋπολογιστικής.

Στο Σχήμα 2.1 [11] παρουσιάζεται συνοπτικά η εξέλιξη προς τη Νεφοϋπολογιστική:



Σχήμα 2.1: Εξέλιξη προς τη Νεφοϋπολογιστική [11]

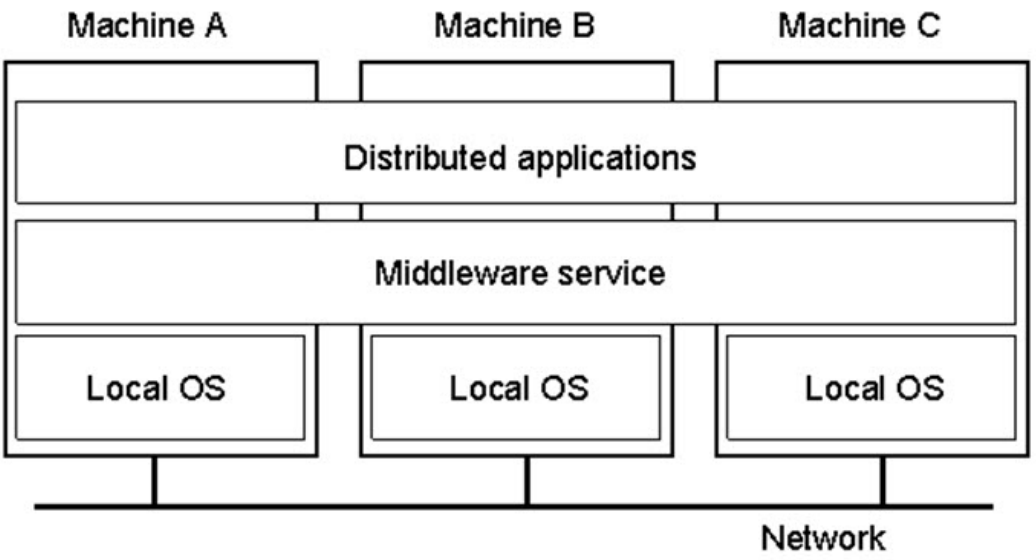
Στις επόμενες ενότητες του κεφαλαίου θα οριστούν και θα αναλυθούν συνοπτικά, βασικές έννοιες και τεχνολογίες, που βρίσκονται εντός του πλαισίου της παρούσας εργασίας, σχετικές με τη Νεφοϋπολογιστική.

2.3 Κατανεμημένα Συστήματα

Ορισμός 2.1 (Κατανεμημένα Συστήματα) :
"Ως κατανεμημένο σύστημα ορίζεται μία συλλογή από αυτόνομους υπολογιστές που συνδέονται μεταξύ τους μέσω ενός δικτύου και χρησιμοποιούν ειδικά σχεδιασμένο λογισμικό για την παροχή ενοποιημένων υπολογιστικών υπηρεσιών. Σε ένα τέτοιο σύστημα, οι διεργασίες που εκτελούνται από τους δικτυωμένους υπολογιστές επικοινωνούν μεταξύ τους και συντονίζουν τις κινήσεις τους μόνο μέσω της ανταλλαγής μηνυμάτων."[12]

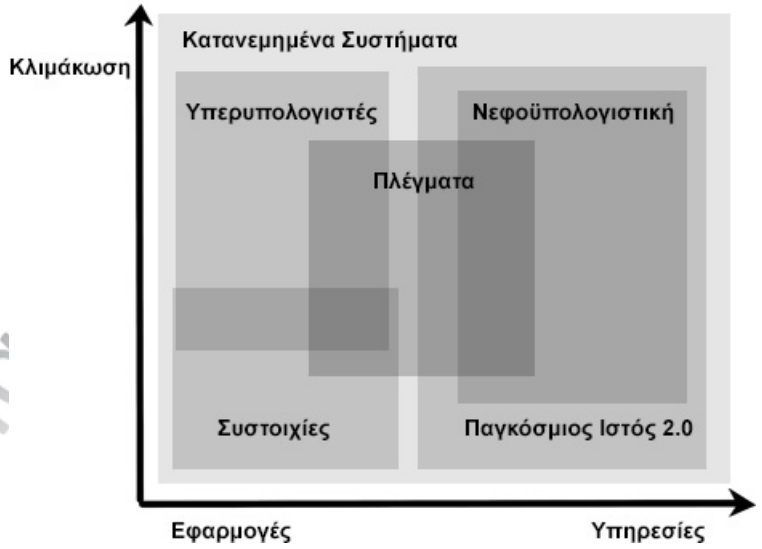
Ο παραπάνω ορισμός απεικονίζεται στο Σχήμα 2.2.

Η επικοινωνία των υπολογιστών αυτών μπορεί να είναι είτε ομογενής είτε ετερογενής και να διανέμεται σε παγκόσμιο ή τοπικό επίπεδο. Ανάλογα τα χαρακτηριστικά τους



Σχήμα 2.2: Ένα κατανεμημένο σύστημα [12]

τα κατανεμημένα συστήματα έχουν διαφορετικά υποσύνολα όπως υπερυπολογιστές, πλέγματα, συστοιχίες, web 2.0 και υπολογιστικά νέφη [13]. Τα υποσύνολα αυτά καθώς και η σχέση τους φαίνεται στο Σχήμα 2.4 [13] και αναλύονται συνοπτικά, για λόγους πληρότητας στη συνέχεια.



Σχήμα 2.3: Υποσύνολα κατανεμημένων συστημάτων [13]

2.3.1 Συστοιχίες

Χαρακτηριστικό των συστοιχιών (clusters) είναι ότι οι υπολογιστές που είναι συνδεδεμένοι μεταξύ τους διαμοιράζονται τοπικά και αποτελούνται το ίδιο φυσικό υλικό και

λειτουργικό σύστημα. Κατά συνέπεια οι σταθμοί εργασίας τύπου συστοιχιών συνδέονται μεταξύ τους και μπορούν ενδεχομένως αν χρησιμοποιηθούν ως ένας υπερυπολογιστής [14].

2.3.2 Υπερυπολογιστές

Οι υπερυπολογιστές (supercomputers) μπορούν να συγκριθούν εύκολα με τις συστοιχίες καθώς ακολουθούν την ίδια λογική με σημαντική διαφορά ότι οι πόροι είναι συγχωνευμένοι σε ένα πλαίσιο και δεν διασυνδέονται τοπικά με άλλες μηχανές [14]. Σημαντικό τους μειονέκτημα είναι το ακριβό τους κόστος καθώς οι υπερβολικά μεγάλες ενεργειακές τους απαιτήσεις.

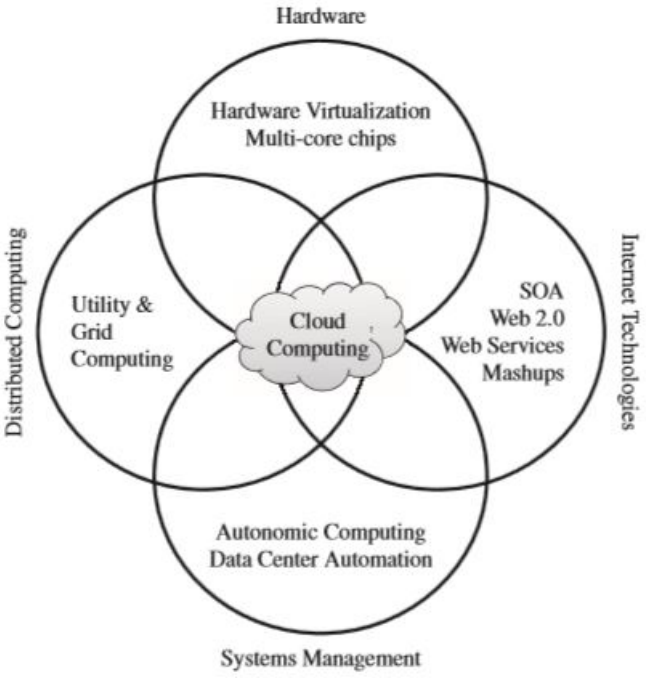
2.3.3 Πλέγματα

Ενώ οι συστοιχίες διανέμονται σε τοπικό επίπεδο και χρησιμοποιούν το ίδιο φυσικό υλικό και λειτουργικό σύστημα, τα πλέγματα (grids) περιλαμβάνουν ετερογενείς υπολογιστές συνδεμένους μεταξύ τους που διανέμονται σε παγκόσμια επίπεδο. Το λειτουργικό και το φυσικό υλικό των υπολογιστών αυτών μπορεί επίσης να διαφέρει [14].

2.4 Νεφοϋπολογιστική

Ως υπολογιστικό νέφος ή Νεφοϋπολογιστική, θεωρείται η παροχή υπηρεσιών πληροφορικής διαμέσω του διαδικτύου. Υπηρεσίες βασισμένες στη Νεφοϋπολογιστική επιτρέπουν στα άτομα και τις επιχειρήσεις να χρησιμοποιούν λογισμικό που διαχειρίζεται από τρίτους. Κοινά παραδείγματα υπηρεσιών Νεφοϋπολογιστικής είναι υπηρεσίες αποθήκευσης αρχείων, ιστοσελίδες κοινωνικής δικτύωσης και webmail. Το μοντέλο στο οποίο βασίζεται η Νεφοϋπολογιστική επιτρέπει την πρόσβαση σε πληροφορίες και υπολογιστικούς πόρους από οποιοδήποτε σημείο με την προϋπόθεση ότι υπάρχει πρόσβαση στο δίκτυο. Είναι συνώνυμο με τα κατανεμημένα συστήματα. Συχνότερα χρησιμοποιείται για να δηλώσει υπηρεσίες με βάση το δίκτυο που φαίνεται ότι παρέχονται από φυσικούς εξυπηρετητές (servers), οι οποίες όμως στην πραγματικότητα χρησιμοποιούν εικονικούς εξυπηρετητές που προσομοιώνονται από λογισμικό που τρέχει σε ένα ή περισσότερα φυσικά μηχανήματα. Καθώς οι εξυπηρετητές αυτοί είναι εικονικοί και δεν υπάρχουν φυσικά μπορούν να μετακινούνται και να κλιμακώνονται ή να αποκλιμακώνονται (scale up/down) πολύ γρήγορα χωρίς ο τελικός χρήστης να

επηρεάζεται. Στο Σχήμα 2.4 [15] παρουσιάζεται συνοπτικά η σχέση της Νεφοϋπολογιστικής με λοιπές τεχνολογίες.



Σχήμα 2.4: Σχέση Νεφοϋπολογιστικής με άλλες τεχνολογίες [15]

Ο ορισμός που δίνει το National Institute of Standards and Technology (NIST)) [16] είναι ο εξής:

Ορισμός 2.2 (Νεφοϋπολογιστική) :
"Η Νεφοϋπολογιστική είναι ένα μοντέλο που επιτρέπει ευέλικτη, κατόπιν ζήτησης δικτυακή πρόσβαση σε ένα κοινόχρηστο σύνολο παραμετροποιήσιμων υπολογιστικών πόρων (π.χ. δίκτυα, εξυπηρετητές, αποθηκευτικοί χώροι, εφαρμογές και υπηρεσίες), το οποίο μπορεί να τροφοδοτηθεί γρήγορα και να διατεθεί με ελάχιστη προσπάθεια διαχείρισης ή αλληλεπίδραση με τον πάροχο της υπηρεσίας. Αυτό το μοντέλο προωθεί την διαθεσιμότητα και αποτελείται από πέντε βασικά χαρακτηριστικά, τρία μοντέλα παροχής υπηρεσιών, και τέσσερα μοντέλα ανάπτυξης."

2.4.1 Βασικά Χαρακτηριστικά Νεφοϋπολογιστικής

Όπως αναφέρεται στον ορισμό του NIST [16] τα βασικά χαρακτηριστικά της Νεφοϋπολογιστικής είναι πέντε. Αυτά, σύμφωνα με τον ορισμό [15] είναι:

1. **On-demand self-service:** Ένας καταναλωτής, μπορεί να δεσμεύσει από μόνος του τους υπολογιστικούς πόρους που χρειάζεται, όπως χρόνο στον εξυπηρετητή

και αποθηκευτικό χώρο στο δίκτυο, ανάλογα με τις ανάγκες του αυτόματα, χωρίς να απαιτείται ανθρώπινη αλληλεπίδραση με το φορέα παροχής της εκάστοτε υπηρεσίας.

2. **Ευρεία πρόσβαση στο δίκτυο:** Οι δυνατότητες είναι διαθέσιμες μέσω του δικτύου και προσβάσιμες μέσω τυποποιημένων μηχανισμών που προωθούν την χρήση από ετερογενείς thin ή thick client πλατφόρμες (π.χ. κινητά τηλέφωνα, φορητούς υπολογιστές ή PDAs).

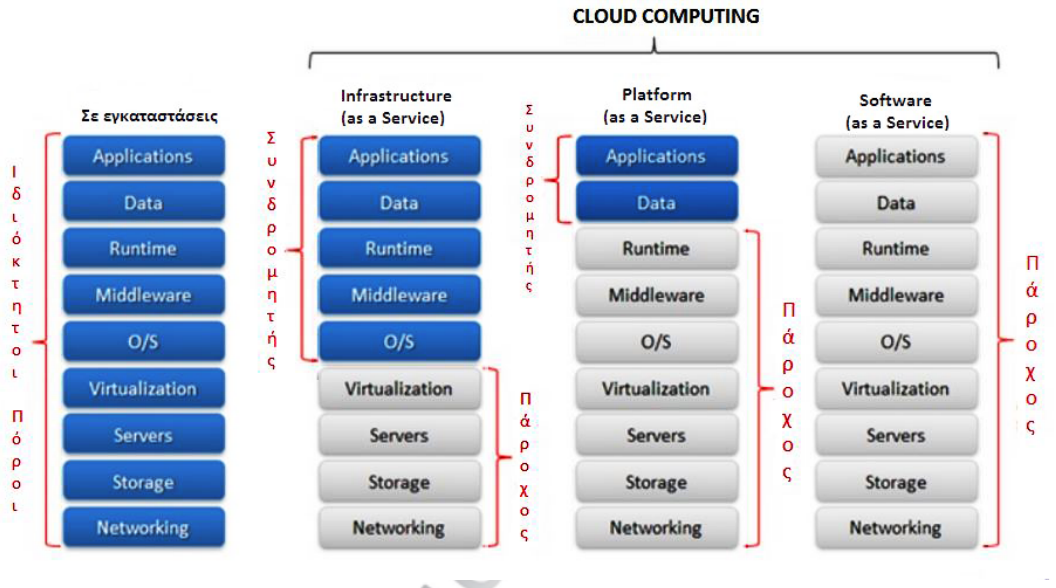
3. **Κοινή διάθεση των πόρων:** Οι υπολογιστικοί πόροι του παρόχου χρησιμοποιούνται για να εξυπηρετήσουν πολλαπλούς καταναλωτές με τη χρήση του μοντέλου πολλαπλών μισθωτών (multi-tenant), με τους διάφορους φυσικούς και εικονικούς πόρους να ανατίθενται δυναμικά και εκ νέου ανάλογα με τη ζήτηση των καταναλωτών. Υπάρχει μια αίσθηση ανεξαρτησίας από τον τόπο ανάθεσης καθώς ο πελάτης δεν έχει γενικά κανέναν έλεγχο ή γνώση σχετικά με την ακριβή τοποθεσία των παρεχόμενων πόρων, αλλά μπορεί να είναι σε θέση να προσδιορίζει την τοποθεσία σε ένα υψηλότερο επίπεδο αφαίρεσης (π.χ. χώρα, κράτος, ή datacenter).Παραδείγματα πόρων αποτελούν οι αποθηκευτικοί χώροι, η επεξεργαστική ισχύς, η μνήμη, το bandwidth του δικτύου, καθώς και οι εικονικές μηχανές.

4. **Ταχεία ελαστικότητα:** Οι πόροι μπορούν να δεσμευτούν προς χρήση γρήγορα και ελαστικά, σε ορισμένες περιπτώσεις αυτόματα, έτσι ώστε να εμφανιστούν άμεσα ως μη διαθέσιμοι (scale out) και επίσης να αποδεσμευτούν γρήγορα για να εμφανιστούν ξανά ως διαθέσιμοι (scale in). Για τον καταναλωτή, οι διαθέσιμες δυνατότητες για δέσμευση και χρήση συχνά φαίνεται να είναι απεριόριστες και μπορούν να αγοραστούν ανά πάσα στιγμή και σε οποιαδήποτε ποσότητα.

5. **Μετρήσιμα επίπεδα παροχής υπηρεσιών:** Τα συστήματα που βασίζονται στη Νεφοϋπολογιστική ελέγχουν και βελτιστοποιούν αυτόματα τη χρήση των πόρων, αξιοποιώντας μια δυνατότητα μέτρησης σε κάποιο επίπεδο αφαίρεσης που είναι κατάλληλο για το είδος της υπηρεσίας (π.χ. αποθήκευση, επεξεργασία, χρήση δικτύου, ενεργοί λογαριασμοί χρηστών). Η χρήση των πόρων μπορεί να παρακολουθείται, να ελέγχεται, και να παρουσιάζεται με τη μορφή αναφορών (reports), παρέχοντας διαφάνεια τόσο για τον πάροχο όσο και για τον καταναλωτή της χρησιμοποιούμενης υπηρεσίας.

2.4.2 Μοντέλα Παροχής Υπηρεσιών Νεφοϋπολογιστικής

Τα βασικά μοντέλα παροχής υπηρεσιών σύμφωνα με τον ορισμό του NIST [16] είναι τρία και προσφέρει το καθένα διαφορετικές δυνατότητες. Παρουσιάζονται διαγραμματικά στο Σχήμα 2.5 και αναλύονται στη συνέχεια.



Σχήμα 2.5: Διαχωρισμός ευθυνών στα μοντέλα παροχής υπηρεσιών

1. **Λογισμικό υπό μορφή Υπηρεσίας (Software as a Service (SaaS)):** Η δυνατότητα που παρέχεται στον καταναλωτή είναι να χρησιμοποιεί τις εφαρμογές του παρόχου που τρέχουν σε μια υποδομή υπολογιστικού νέφους και στοχεύει στην αντικατάσταση των εφαρμογών που τρέχουν σε ένα προσωπικό υπολογιστή [17]. Οι εφαρμογές είναι προσβάσιμες από διάφορες client συσκευές μέσω ενός thin client interface, όπως ένα πρόγραμμα περιήγησης στο διαδίκτυο (π.χ. webmail). Ο καταναλωτής δεν έχει τη διαχείριση ή τον έλεγχο της χρησιμοποιούμενης υποδομής υπολογιστικού νέφους συμπεριλαμβανομένων των δικτύων, των εξυπηρετητών, των λειτουργικών συστημάτων, των αποθηκευτικών μονάδων, ή ακόμα και μεμονωμένων δυνατοτήτων της εφαρμογής, με την πιθανή εξαίρεση κάποιων περιορισμένων, ορισμένων συγκεκριμένα για τον εκάστοτε χρήστη, ρυθμίσεων παραμετροποίησης των εφαρμογών. Πιο γνωστό παράδειγμα αποτελεί το office365.

2. **Πλατφόρμα υπό μορφή Υπηρεσίας (Platform as a Service (PaaS)):** Η δυνατότητα που παρέχεται στον καταναλωτή είναι να αναπτύσσει πάνω στην υποδομή υπολογιστικού νέφους εφαρμογές που έχει δημιουργήσει ή εφαρμογές

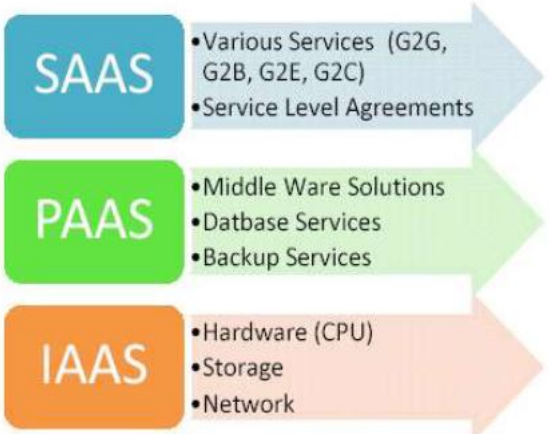
που έχει αποκτήσει, οι οποίες έχουν δημιουργηθεί με χρήση γλωσσών προγραμματισμού και εργαλείων που υποστηρίζονται από τον πάροχο [18]. Ο καταναλωτής δεν διαχειρίζεται ούτε ελέγχει τη σχετική υποδομή υπολογιστικού νέφους που συμπεριλαμβάνει τα δίκτυα, τους εξυπηρετητές, τα λειτουργικά συστήματα ή τα αποθηκευτικά μέσα, αλλά έχει τον έλεγχο των εφαρμογών που έχουν αναπτυχθεί, και ενδεχομένως, των παραμετροποιήσεων του περιβάλλοντος φιλοξενίας των εφαρμογών. Πιο γνωστά παραδείγματα τα Google AppsEngine [19] και Windows Azure Compute [20].

3. **Υποδομή υπό μορφή Υπηρεσίας (Infrastructure as a Service (IaaS)):** Η δυνατότητα που παρέχεται στον καταναλωτή είναι να μπορεί να δεσμεύσει, προς χρήση, επεξεργαστική ισχύ, αποθηκευτικά μέσα, δίκτυα, και άλλους θεμελιώδεις υπολογιστικούς πόρους, στους οποίους είναι σε θέση να αναπτύξει και να εκτελέσει λογισμικό, το οποίο μπορεί να περιλαμβάνει λειτουργικά συστήματα και εφαρμογές. Ο καταναλωτής δεν έχει τη διαχείριση ή τον έλεγχο της χρησιμοποιούμενης υποδομής υπολογιστικού νέφους, αλλά έχει τον έλεγχο των λειτουργικών συστημάτων, των αποθηκευτικών μέσων, των εφαρμογών που έχουν αναπτυχθεί και πιθανόν κάποιον περιορισμένο έλεγχο επιλεγμένου εξοπλισμού δικτύωσης [21] (π.χ. τείχη προστασίας - firewalls). Μια κατηγοριοποίηση [22] που γίνεται στο IaaS είναι:

   - **Υπολογισμός υπό μορφή Υπηρεσίας (Computation as a Service (CaaS)):** Στο μοντέλο αυτό εικονικές μηχανές ενοικιάζονται και χρεώνονται με την ώρα, βάσει των δυνατοτήτων της εικονικής μηχανής (μέγεθος μνήμης RAM και CPU, λειτουργικό σύστημα κτλ.) και
   - **Δεδομένα υπό μορφή Υπηρεσίας (Data as a Service (DaaS)):** όπου "απεριόριστος" αποθηκευτικός χώρος χρησιμοποιείται για την αποθήκευση δεδομένων των χρηστών, ανεξαρτήτως του είδους τους, με τη χρέωση να γίνεται ανά gigabyte (GB) χρήσης.

Πιο γνωστά παραδείγματα αποτελούν τα Amazon EC2 [8], Google Compute Engine [19], Windows Azure VM [20].

Η αρχιτεκτονική της Νεφοϋπολογιστικής βάσει των παραπάνω μοντέλων φαίνεται στο Σχήμα 2.6 [23]:



Σχήμα 2.6: Αρχιτεκτονική Νεφοϋπολογιστικής βάσει μοντέλων παροχής υπηρεσιών [23]

2.4.3 Μοντέλα Εφαρμογής Νεφοϋπολογιστικής

Τα μοντέλα πρακτικής εφαρμογής της Νεφοϋπολογιστικής, σύμφωνα με τον ορισμό του NIST [15], είναι τα ακόλουθα:

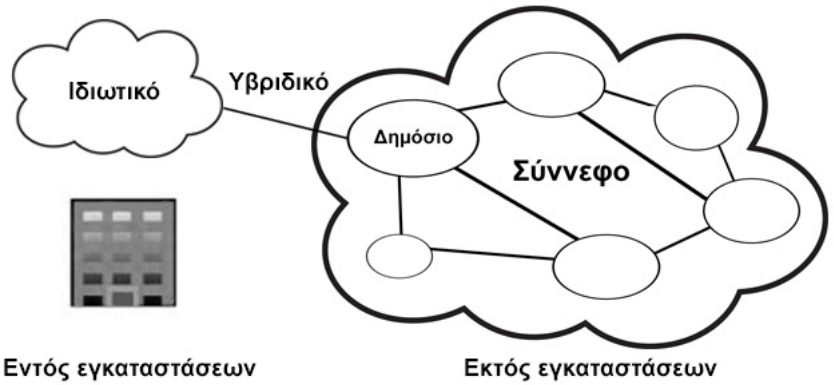
1. **Ιδιωτική Υποδομή Νεφοϋπολογιστικής (Private cloud):** Η υποδομή υπολογιστικού νέφους λειτουργεί αποκλειστικά και μόνο για έναν. Η διαχείρισή της μπορεί να γίνεται από τον ίδιο τον οργανισμό ή από τρίτους και μπορεί να βρίσκεται εντός ή εκτός των εγκαταστάσεων του οργανισμού [24].

2. **Κοινοτική Υποδομή Νεφοϋπολογστικής (Community cloud):** Η υποδομή υπολογιστικού νέφους μοιράζεται μεταξύ πολλών οργανισμών και υποστηρίζει μια συγκεκριμένη κοινότητα που έχει κοινές απαιτήσεις (π.χ. αποστολή, απαιτήσεις ασφαλείας, πολιτική και θέματα συμμόρφωσης). Η διαχείρισή της μπορεί να γίνεται από τον ίδιο τον οργανισμό ή από τρίτους και μπορεί να βρίσκεται εντός ή εκτός των εγκαταστάσεων του οργανισμού.

3. **Δημόσια Υποδομή Νεφοϋπολογιστικής (Public cloud):** Η υποδομή υπολογιστικού νέφους διατίθεται στο ευρύ κοινό ή σε μια μεγάλη ομάδα εταιρειών και ανήκει σε έναν οργανισμό που πουλά υπηρεσίες Νεφοϋπολογιστικής.

4. **Υβριδική Υποδομή Νεφοϋπολογιστικής (Hybrid cloud):** Η υποδομή υπολογιστικού νέφους είναι μια σύνθεση από δύο ή περισσότερα υπολογιστικά νέφη (ιδιωτικό, δημόσιο ή κοινοτικό) τα οποία παραμένουν μοναδικές οντότητες, αλλά συνδέονται μεταξύ τους με τυποποιημένη ή ιδιοκτήτη τεχνολογία που επιτρέπει τη φορητότητα δεδομένων και εφαρμογών (π.χ. εξισορρόπηση φόρτου εργασίας μεταξύ των υπολογιστικών νεφών).

Στο Σχήμα 2.7 [25] φαίνονται διαγραμματικά τα μοντέλα εφαρμογής της Νεφοϋπολογιστικής:



Σχήμα 2.7: Μοντέλα εφαρμογής Νεφοϋπολογιστικής [25]

2.5 Εικονοποίηση

Η εικονοποίηση αποτελεί μια από τις βασικότερες τεχνολογίες που επέτρεψαν την εξέλιξη της Νεφοϋπολογιστικής και αποτελεί δομικό στοιχείο της. Με τον όρο εικονοποίηση εννοούμε την τεχνολογία με την οποία τα φυσικά συστήματα μετατρέπονται σε ιδεατά (εικονικές μηχανές (Virtual Machine (VM))). Κάθε φυσικός πόρος (επεξεργαστική ισχύς, μνήμη, δίκτυο, αποθηκευτικά μέσα κλπ.) γίνεται ένας ενιαίος πόρος και μοιράζεται ταυτόχρονα σε πολλά εικονικά συστήματα. Για μια λύση εικονοποίησης χρειάζονται: (i) κατάλληλος εξοπλισμός (φυσικό υλικό), (ii) λογισμικό εικονοποίησης ή επόπτης (hypervisor) και (iii) λογισμικό διαχείρισης Με την εικονοποίηση το φυσικό υλικό διαχωρίζεται από το λογισμικό (λειτουργικά συστήματα και εφαρμογές). Ο επόπτης είναι ένα νέο επίπεδο (virtualization layer) μεταξύ του φυσικού υλικού και του λογισμικού, που ενοποιεί τους φυσικούς πόρους και τους διαμοιράζει στα ιδεατά συστήματα με τρόπο διάφανο. Τα ιδεατά συστήματα συνεχίζουν να νομίζουν ότι επικοινωνούν απευθείας με το φυσικό υλικό, αλλά στην πραγματικότητα επικοινωνούν με το virtualization layer. Αυτό επιτρέπει τη μετακίνηση επεξεργαστικής ισχύος, μνήμης ή και αποθηκευτικού χώρου από τη μια εικονική μηχανή σε μια άλλη εν ώρα λειτουργίας και σύμφωνα με τις ανάγκες. Το αποτέλεσμα είναι η καλύτερη εκμετάλλευση των πόρων συνολικά, μεγάλη εξοικονόμηση ενέργειας και φυσικού χώρου και ευκολότερη διαχείριση της υποδομής.

Σήμερα η εικονοποίηση έχει διεισδύσει σε πολλές επιχειρήσεις και αναμένεται ότι γρήγορα τα περισσότερα συστήματα θα γίνουν ιδεατά, καθώς οι απαιτήσεις για αποδοτική χρησιμοποίηση του φυσικού υλικού σε συνδυασμό με την μέγιστη δυνατή μείωση της ηλεκτρικής κατανάλωσης, γίνονται ολοένα και πιο επιτακτικές.

2.5.1 Εικονοποίηση Πλατφόρμας

Η δημιουργία και διαχείριση εικονικών μηχανών ονομάζεται platform virtualization ή server virtualization όπως έχει αρχίσει να ονομάζεται πρόσφατα. Η εικονοποίηση πλατφόρμας παρουσιάζει ιδιαίτερο ενδιαφέρον, όπου ένα λογισμικό ελέγχου (επόπτης) εκτελούμενο σε φυσικό υλικό προσομοιώνει ένα υπολογιστικό περιβάλλον, μία εικονική μηχανή, επάνω από το οποίο μπορεί να τρέξει κάποιο φιλοξενούμενο λογισμικό, απομονωμένο από το υπόλοιπο σύστημα. Η θεμελιώδης λογική πίσω από την εικονοποίηση πλατφόρμας είναι η αρχή πως οποιαδήποτε λειτουργία μπορεί να εκτελεστεί είτε από λογισμικό είτε από εξειδικευμένο υλικό με τις μοναδικές διαφορές να αφορούν την ευελιξία και την απόδοση. Είναι δυνατόν να προσομοιώνονται ταυτόχρονα πολλαπλές εικονικές μηχανές, εντελώς απομονωμένες μεταξύ τους, από το ίδιο λογισμικό ελέγχου. Υπάρχουν πολλά είδη εικονοποίησης πλατφόρμας. Τα σημαντικότερα είναι τα ακόλουθα:

1. **Εξομοίωση:** Η εικονική μηχανή προσομοιώνει εξολοκλήρου μία αρχιτεκτονική υλικού, πιθανώς διαφορετική από το πραγματικό υποκείμενο υλικό, επιτρέποντας έτσι να εκτελεστεί επάνω της ένα μη τροποποιημένο, φιλοξενούμενο λειτουργικό σύστημα σχεδιασμένο για τον εξομοιούμενο επεξεργαστή. Η εξομοίωση είναι διερμηνεία σε χρόνο εκτέλεσης του κώδικα του φιλοξενούμενου λειτουργικού συστήματος, με έναν κύκλο ανάγνωσης - αποκωδικοποίησης - εκτέλεσης όπου κάθε εντολή που ανήκει στο σύνολο εντολών του επεξεργαστή - πηγή μεταφράζεται σε μία εντολή του συνόλου εντολών του επεξεργαστή - στόχου. Παράλληλα η εικονική μηχανή παρέχει μία αφαίρεση της μνήμης, των συσκευών Εισόδου / Εξόδου κλπ, φροντίζοντας ώστε κάθε μεταφρασμένη εντολή που απευθύνεται σε αυτά τα υποσυστήματα να τροποποιεί μόνο τις αφαιρέσεις / λογικές αναπαραστάσεις τους, οι οποίες κατευθύνονται και υλοποιούνται από το λογισμικό ελέγχου, και όχι το πραγματικό υλικό. Προκειμένου να αυξηθούν οι επιδόσεις είναι δυνατόν να χρησιμοποιηθεί δυναμική μετάφραση αντί για απλή εξομοίωση, όπου οι μεταφρασμένες εντολές αποθηκεύονται σε κρυφή μνήμη και μπορούν να επαναχρησιμοποιηθούν αργότερα χωρίς εκ νέου μετάφραση, ή δυναμική επαναμεταγλώττιση, όπου εκτός της χρήσης κρυφής μνήμης γίνεται και βελτιστοποίηση κρίσιμων τμημάτων του κώδικα (παρόμοια με τη μεταγλώττιση JIT της Java, του .NET και άλλων παρόμοιων πλατφορμών υψηλού επιπέδου).

2. **Πλήρης:** Η εικονική μηχανή προσομοιώνει επαρκές τμήμα του πραγματικού υποκείμενου υλικού ώστε να επιτρέπει την εκτέλεση επάνω της ενός μη τροποποιημένου, φιλοξενούμενου λειτουργικού συστήματος σχεδιασμένου για τον ίδιο

τύπο επεξεργαστή με την πραγματική CPU. Στην πλήρη εικονοποίηση δεν χρειάζεται εξομοίωση του συνόλου εντολών του επεξεργαστή και μάλιστα ένα τμήμα του κώδικα του φιλοξενούμενου λειτουργικού συστήματος μπορεί να εκτελείται απευθείας από το υλικό, χωρίς μεσολάβηση του επόπτη, αρκεί να μην επηρεάζει υποσυστήματα εκτός του άμεσου ελέγχου του τελευταίου. Τα κρίσιμα σημεία του φιλοξενούμενου κώδικα ωστόσο, όπως αυτά που προσπαθούν να αποκτήσουν πρόσβαση στο υλικό (π.χ. κλήσεις συστήματος), συλλαμβάνονται από το λογισμικό ελέγχου και προσομοιώνονται, αφού τα αποτελέσματα κάθε λειτουργίας που επιτελείται σε μία εικονική μηχανή δεν επιτρέπεται να τροποποιούν την κατάσταση άλλων εικονικών μηχανών, του επόπτη ή του υλικού. Αν το πραγματικό υλικό βοηθά και επιταχύνει τη λειτουργία του λογισμικού ελέγχου τότε η πλήρης εικονοποίηση ονομάζεται εγγενής (native). Η βοήθεια αυτή αφορά κυρίως εύκολη διάκριση μεταξύ εντολών που μπορούν να εκτελεστούν απευθείας και εντολών που πρέπει να προσομοιωθούν από το λογισμικό. Όπως και στην εξομοίωση η εικονική μηχανή παρέχει στο φιλοξενούμενο λειτουργικό σύστημα μία αφαίρεση της μνήμης, των συσκευών Εισόδου / Εξόδου κλπ, ενώ η εγγενής εκτέλεση μεγάλου μέρους του κώδικα παρέχει πολύ καλύτερες επιδόσεις.

3. **Παραεικονικοποίηση:** Η εικονική μηχανή δεν προσομοιώνει επακριβώς το υλικό αλλά παρέχει στις εικονικές μηχανές ένα Application Programming Interface (API), μία προγραμματιστική διασύνδεση, ώστε να επιτρέπει την εκτέλεση επάνω της ενός τροποποιημένου, φιλοξενούμενου λειτουργικού συστήματος σχεδιασμένου για εκτέλεση από τον συγκεκριμένο επόπτη. Το προαναφερθέν API ονομάζεται διασύνδεση υπερκλήσεων και ένα λειτουργικό σύστημα πρέπει να μεταφερθεί ρητά σε έκδοση κατάλληλη για εκτέλεση από ένα σύστημα παραεικονικοποίησης, ώστε ο φιλοξενούμενος πυρήνας αντί να προσπελαύνει το υλικό άμεσα να εκτελεί υπερκλήσεις και να αναμένει απαντήσεις ή ασύγχρονες ειδοποιήσεις από τον επόπτη.

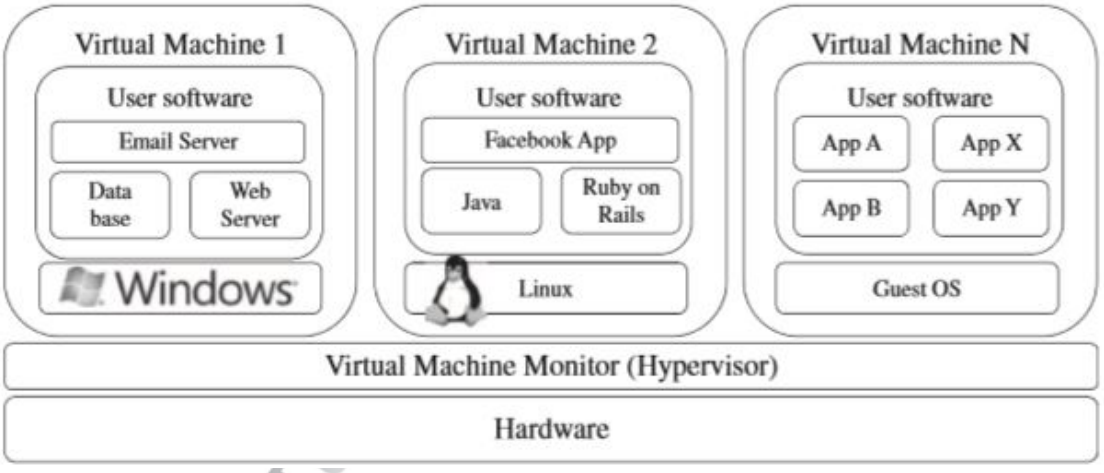
2.5.2 Εικονικές Μηχανές

Μια εικονική μηχανή είναι η υλοποίηση, μέσω λογισμικού, μιας μηχανής (ενός υπολογιστή) που εκτελεί προγράμματα και λειτουργίες όπως ένα φυσικό μηχάνημα. Οι εικονικές μηχανές χωρίζονται σε δύο βασικές κατηγορίες με βάση τη χρήση τους και το βαθμό που ανταποκρίνονται σε μια φυσική μηχανή:

1. **System Virtual Machine:** Παρέχει μια πλήρη πλατφόρμα συστήματος που υποστηρίζει την εκτέλεση ενός πλήρους λειτουργικού συστήματος [26]. Συνήθως προσομοιώνει μια υπάρχουσα αρχιτεκτονική και χρησιμοποιούνται με σκοπό:

- Είτε να παρέχουν μια πλατφόρμα για το τρέξιμο προγραμμάτων όταν το φυσικό υλικό δεν είναι διαθέσιμο για χρήση, όπως για παράδειγμα τη χρήση λογισμικού σε παρωχημένες πλατφόρμες,
- είτε για να υπάρχουν πολλαπλά instances με σκοπό την αποδοτικότερη χρήση υπολογιστικών πόρων, σε όρους κατανάλωσης ενέργειας και μείωσης του κόστους. Αυτό είναι γνωστό ως εικονοποίηση υλικου και αποτελεί το κλειδί για τη Νεφοϋπολογιστική.

2. **Process Virtual Machine** γνωστό και ως **(Language Virtual Machine)**: Είναι σχεδιασμένο ώστε να τρέχει ένα απλό πρόγραμμα, γεγονός που σημαίνει ότι υποστηρίζει μία μόνο διαδικασία. Δημιουργούνται με σκοπό να παρέχουν φορητότητα του προγράμματος και ευελιξία. Βασικό χαρακτηριστικό είναι ότι το λογισμικό που τρέχει στο εσωτερικό ενός τέτοιου εικονικού μηχανήματος περιορίζεται στους πόρους που παρέχονται από την εικονική μηχανή.



Σχήμα 2.8: Ένας εικονικός εξυπηρετητής που φιλοξενεί 3 εικονικές μηχανές [15]

2.5.3 Κύκλος Ζωής Εικονικής Μηχανής

Η ελαστικότητα έχει να κάνει με την προσαρμογή του μεγέθους ενός συστήματος αρχικοποιώντας μια εικονική μηχανή και τερματίζοντας τις υφιστάμενες όποτε χρειάζεται. Φαίνεται λοιπόν ότι και οι εικονικές μηχανές έχουν ένα κύκλο ζωής και η βελτιστοποίηση του κύκλου αυτού είναι βασικό κλειδί για επιτυχή ελαστικότητα. Οι εικονικές μηχανές περνούν από διάφορες φάσεις κατά τη διάρκεια του κύκλου ζωής τους. Από την οπτική γωνία μιας εφαρμογής οι φάσεις αυτές είναι:

1. **Προετοιμασία προτύπου (template):** Στη φάση αυτή η εικονική μηχανή και τα δεδομένα της προετοιμάζονται μέχρι το σημείο όπου μπορούν να αρχικοποιηθούν στην υποδομή υπολογιστικού νέφους. Το πρότυπο μπορεί να είναι μια

βασική εγκατάσταση ενός λειτουργικού συστήματος σε εικονικό υλικό ή περαιτέρω ειδικευμένες ενέργειες για ένα συγκεκριμένο σκοπό.

2. **Διαμόρφωση του instance:** Η φάση αυτή περιλαμβάνει βήματα όπως η επιλογή του μεγέθους της εικόνας όπως για παράδειγμα το μέγεθος της RAM και οι CPU πυρήνες που θα έχει το μηχάνημα. Η διαμόρφωση του δικτύου καθώς και άλλες διαμορφώσεις του εικονικού υλικού γίνονται σε αυτή τη φάση. Επιπλέον, ρυθμίσεις ασφαλείας όπως κλειδιά πρόσβασης (Secure Shell (SSH)) διαμορφώνονται σε αυτή τη φάση.

3. **Εκκίνηση του instance:** Με το πρότυπο να έχει επιλεχθεί και τις απαραίτητες ρυθμίσεις να έχουν γίνει η εικονική μηχανή είναι έτοιμη για εκκίνηση. Αυτή η φάση γίνεται από τον πάροχο της υπηρεσίας αλλά οι πελάτες πρέπει να είναι σε θέση να παρακολουθούν τη πρόοδο προκειμένου να βλέπουν τη πρόοδο της ανάπτυξης της. Αυτό που γίνεται στη πραγματικότητα είναι ο πάροχος να επιλέγει ένα φυσικό εξυπηρετητή στον οποίο θα διαθέσει την εικονική μηχανή και κάνει τις απαραίτητες αλλαγές στο σύστημα του ώστε να κατανείμει τμήματα της φυσικής CPU, μνήμη RAM, αποθηκευτικό χώρο και άλλους πόρους στην εικονική μηχανή. Όταν η εκκίνηση πραγματοποιηθεί ο πελάτης θα ενημερωθεί μέσω κάποιου μηχανισμού ενημέρωσης που χρησιμοποιεί ο πάροχος.

4. **Πλαισίωση του instance (instance contextualization):** Μετά την εκκίνηση η εικονική μηχανή θα πρέπει να πλαισιωθεί για το περιβάλλον εργασίας στο οποίο θα συμμετέχει. Για παράδειγμα η εικονική μηχανή θα μπορούσε να προστεθεί σε μια ομάδα εργαζομένων που προσκομίζουν στοιχεία εργασίας από μια ουρά ή να προστεθεί σε ένα σύμπλεγμα εξυπηρετητών εφαρμογών. Για την επίλυση της χρονικής καθυστέρησης όπου κάποιος διαχειριστής πρέπει να συνδεθεί στην εικονική μηχανή ώστε να τη πλαισιώσει, η εικονική μηχανή μπορεί να ρυθμιστεί ώστε να τραβήξει τα απαραίτητα δεδομένα με τη χρήση ενός script κατά την εκκίνηση ή ως ένα, ξεχωριστό από το πρότυπο, ISO. Το Open Virtualization Format (OVF) πρότυπο υποστηρίζει τη χρήση ISO εικόνων για το σκοπό αυτό [27]. Η Amazon επιπροσθέτως παρέχει μια τοπική υπηρεσία δικτύου για την αναζήτηση ειδικών για το instance μεταδεδομένων μέσω HTTP. Για τη συγκεκριμένη φάση υπάρχει πολλή ερευνητική δραστηριότητα [28–32] ώστε να προταθούν εφάπαξ και standard λύσεις.

5. **Παρακολούθηση του instance (κατάσταση λειτουργίας):** Έχοντας γίνει η πλαισίωση, το instance της εικονικής μηχανής βρίσκεται σε κατάσταση λειτουργίας. Εκτελεί τα καθήκοντα του και συγχρόνως αναφέρει την κατάσταση του όπως αυτή έχει διαμορφωθεί μέχρι τελικά να τερματιστεί.

6. **Τερματισμός:** Η φάση αυτή είναι το σημείο όπου η εικονική μηχανή πρέπει να ενημερώσει όλο το σύστημα για τον επικείμενο τερματισμό της ώστε το σύστημα ως σύνολο να αντιδράσει σε αυτήν.

Οι παραπάνω φάσεις πρέπει να είναι προσαρμόσιμες και παραμετροποιήσιμες ώστε οι πελάτες της υπηρεσίας να μπορούν να εφαρμόσουν τη δικιά τους λογική σε αυτές.

ΚΕΦΑΛΑΙΟ 3

Πλεονεκτήματα & Εμπόδια

"Ultimately, the cloud is the latest example of Schumpeterian creative destruction: creating wealth for those who exploit it; and leading to the demise of those that don't."
˜Joe Weinman

3.1 Εισαγωγή

Η Νεφοϋπολογιστική έχει συζητηθεί, έχουν γίνει αναρτήσεις σε ιστολόγια και αποτελεί πρώτο τίτλο σε διάφορα workshops, συνέδρια και περιοδικά [24, 25, 33–39] . Παρ' όλα αυτά δεν είναι ακόμα ξεκάθαρο, παρά τα πλεονεκτήματα που προσφέρει, το πότε είναι ωφέλιμο να στραφεί κάποιος σε αυτό.

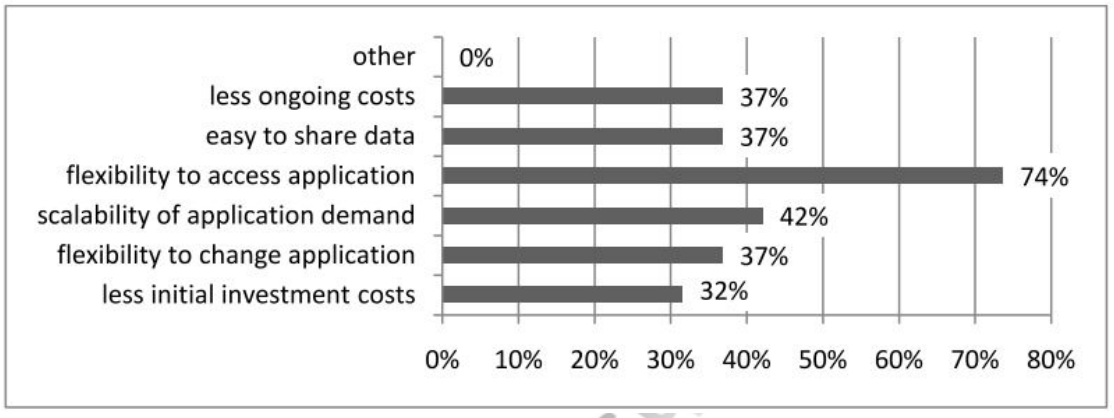
Το παρών κεφάλαιο θα εστιάσει στα πλεονεκτήματα που προσφέρει η μετάβαση στη Νεφοϋπολογιστική, οι ανησυχίες που υπάρχουν, από πλευράς χρηστών, προς τη μετάβαση αυτή καθώς και τους κινδύνους που κρύβει μια πιθανή μετάβαση σε αυτό.

Στόχοι του κεφαλαίου αυτού είναι:

1. **Καταγραφή Βασικών Πλεονεκτημάτων:** Η καταγραφή των βασικών πλεονεκτημάτων που υπόσχεται ότι προσφέρει η Νεφοϋπολογιστική καθώς και η ανάλυση τους.
2. **Καταγραφή Ανησυχιών:** Η καταγραφή των βασικών ανησυχιών των χρηστών που τους αποτρέπουν από τη υιοθέτηση λύσεων που προσφέρονται από τη Νεφοϋπολογιστική και η σε βάθος ανάλυση τους.
3. **Σύγκριση με Παρόχους:** Σύγκριση των υποσχέσεων της Νεφοϋπολογιστικής και των ανησυχιών των χρηστών με υπάρχοντες παρόχους.

3.2 Πλεονεκτήματα

Στο Σχήμα 3.1 παρουσιάζεται μια έρευνα της IDC [37] που παρουσιάζει τους λόγους για τους λόγους που θεωρούν οι πελάτες σήμερα σημαντικότερους για να περάσουν σε μια SaaS υποδομή.



Σχήμα 3.1: Λόγοι χρήσης SaaS [37]

1. **Χαμηλότερο κόστος:** Με τη Νεφοϋπολογιστική ο περιορισμός του κόστους γίνεται αρκετά ευκολότερος. Η μείωση των εξόδων γίνεται εμφανής στο γεγονός ότι δεν χρειάζεται να γίνεται χρήση ειδικευμένου ακριβού φυσικού υλικού. Η ελαστικότητα ενός περιβάλλοντος Νεφοϋπολογιστικής προσφέρει ευελιξία στο σχεδιασμό μιας υποδομής. Η Νεφοϋπολογιστική, προσφέρει ένα καλύτερο τρόπο χρήσης πόρων, όσον αφορά τις πραγματικές ανάγκες του συστήματος. Για παράδειγμα όταν κάποιος σχεδιάζει ένα διαδικτυακό εξυπηρετητή σε μια υποδομή υπολογιστικού νέφους μπορεί να νοικιάσει το φτηνότερο μηχάνημα και να προσθέτει (ή να αφαιρεί πόρους) ανάλογα τη χρήση, ενώ με το παραδοσιακό τρόπο θα έπρεπε η χωρητικότητα του να προβλεφθεί εξαρχής. Στις περισσότερες περιπτώσεις στο δεύτερο σενάριο έχουμε το φαινόμενο overprovisioning όπου αγοράζονται περισσότεροι πόροι από αυτούς που τελικά χρειάζονται.

   Οι πάροχοι υποδομών υπολογιστικού νέφους μπορούν να μειώσουν το κόστος τους καθώς έχουν συγκεντρωμένες τις υποδομές του σε μεγάλα data center σε φτηνές περιοχές. Έτσι ο χρόνος μετακινήσεως του προσωπικού μειώνεται και βελτιώνεται ο χρόνος εργασίας τους. Οι καταναλωτές υπηρεσιών που βασίζονται σε λύσεις Νεφοϋπολογιστικής μπορούν να μειώσουν δραστικά το προσωπικό τους καθώς δεν υφίσταται πλέον η ανάγκη συντήρησης του φυσικού υλικού.

2. **Χαμηλότερο κόστος εκκίνησης:** Για εταιρίες που μόλις ξεκινάνε (startups) η χρήση τεχνολογιών Νεφοϋπολογιστικής μειώνει δραστικά το κόστος εκκίνησης

[34]. Με τη χρήση υποδομών υπολογιστικού νέφους δεν υφίσταται η ανάγκη αγοράς εξοπλισμού εκ των προτέρων. Αντί να περιορίζει το κόστος μειώνει το ρίσκο. Το κόστος του φυσικού υλικού μοιράζεται σε όλη τη διάρκεια ζωής του έργου. Επιπλέον δεν υπάρχει λόγος να αγοραστεί υλικό για έργα με μικρή διάρκεια ζωής, στα οποία σε οποιαδήποτε άλλη περίπτωση το κόστος θα ήταν δύσκολο αν ελεγχτεί. Με το τέλος του έργου δεν μένει υλικό αχρησιμοποίητο.

3. **Ταχύτερη διάθεση προϊόντων στην αγορά:** Το μοντέλο της Νεφοϋπολογιστικής αναφέρει ότι τα προϊόντα που δημιουργούνται με χρήση υποδομών υπολογιστικού νέφους διατίθενται ταχύτερα στην αγορά. Οι επιχειρήσεις μπορούν να αποφύγουν το στάδιο της αρχικής προμήθειας και εγκατάστασης υποδομής επιτρέποντας έτσι της επιχειρηματικές λύσεις να προωθηθούν γρηγορότερα στην αγορά [24].

4. **Ελαστική υποδομή:** Είναι εύκολο για ένα πελάτη της Νεφοϋπολογιστικής να ανταποκριθεί σε αυξημένη ή μειωμένη ζήτηση της υπηρεσίας του, όπως για παράδειγμα η χωρητικότητα ενός δικτυακού εξυπηρετητή, καθώς υπάρχει ελαστικά διαθέσιμη υποδομή όποτε ζητηθεί [35].

5. **Pay-per-use Μοντέλο:** Οι πελάτες εκμεταλλευόμενοι το συγκεκριμένο μοντέλο διασφαλίζουν ότι χρησιμοποιούν ακριβώς όσους πόρους χρειάζονται, ενώ χάρη στην ελαστικότητα μπορούν να είναι βέβαιοι ότι οι πόροι αυτοίθα υπάρχουν διαθέσιμοι εάν χρειαστούν [36].

6. **Ευκολία διαχείρισης:** Η συντήρηση είτε της υποδομής είτε του λειτουργικού είναι απλοποιημένη με αποτέλεσμα λιγότερο φόρτο εργασίας για την ομάδα IT. Επιπλέον για τους χρήστες αυτό που απαιτείται είναι ένα πρόγραμμα περιήγησης και μια σύνδεση στο Internet [38].

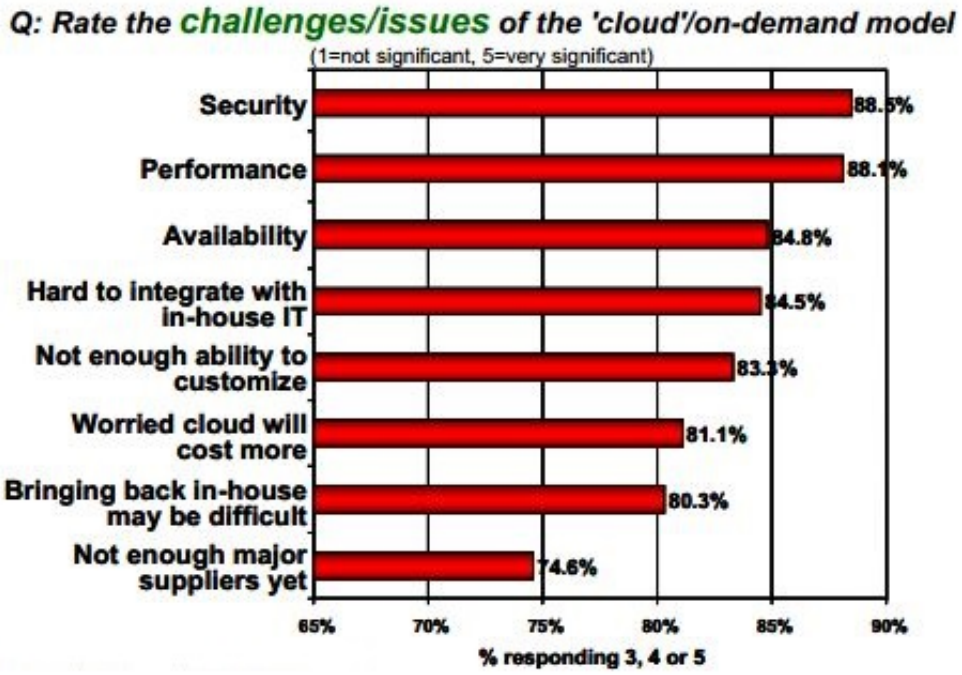
7. **Φιλικό προς το περιβάλλον:** Προϊόντα και υπηρεσίες φιλικές προς το περιβάλλον, θεωρούνται σήμερα σημαντικό βοήθημα για το μάρκετινγκ από όταν αναγνωρίστηκε το θέμα της κλιματικής αλλαγής. Οι υποστηρικτές της Νεφοϋπολογιστικής υποστηρίζουν ότι είναι μια φιλική προς το περιβάλλον λύση καθώς προσφέρει την ίδια υπολογιστική ισχύ έχοντας μικρότερο αντίκτυπο προς το περιβάλλον [37].

   Σε ένα παραδοσιακό IT τμήμα οι ρυθμίσεις σχεδιάζονται ώστε να καλύψουν και ώρες αιχμής. Για παράδειγμα σε διαδικτυακές εφαρμογές το φόρτο εργασίας μπορεί να κυμαίνεται από πολύ υψηλό σε πολύ χαμηλό καθώς εξαρτάται από τον αριθμό των χρηστών (μπορεί κανείς να σκεφτεί τους εξυπηρετητές της Γενικής Γραμματείας Πληροφοριακών Συστημάτων σε περίοδο υποβολής φορολογικών δηλώσεων όπου δέχονται τεράστιο αριθμό χτυπημάτων και σε άλλες

χρονικές περιόδους όπου δέχονται σχεδόν μηδαμινά). Προκειμένου οι εφαρμογές να δουλεύουν απρόσκοπτα σε όλες τις χρονικές περιόδους τα παραδοσιακά πλέγματα δουλεύουν με τη λογική μέγιστου φόρτου εργασίας. Αυτό που συμβαίνει στην πραγματικότητα είναι το πλέγμα να δουλεύει το πολύ στο 10% των δυνατοτήτων του το μεγαλύτερο χρονικό διάστημα. Αυτό είναι προφανές ότι συνεπάγεται με τεράστια δαπάνη πόρων και ενέργειας. Σε υποδομές υπολογιστικού νέφους με τη δυνατότητα προσθήκης και αφαίρεσης πόρων όποτε ζητηθεί το πρόβλημα αυτό παύει να υφίσταται.

3.3 Ανησυχίες Χρηστών

Μια έρευνα που πραγματοποιήθηκε από την IDC [33] και απεικονίζεται στο Σχήμα 3.2 παρουσιάζει τους βασικούς προβληματισμούς που αποτρέπουν την υιοθέτηση της Νεφοϋπολογιστικής σήμερα. Βάσει της έρευνας αυτής, παρατίθενται και αναλύονται οι βασικοί προβληματισμοί που αποτρέπουν τους χρήστες από την υιοθέτηση λύσεων υπολογιστικού νέφους.



Σχήμα 3.2: Αποτρεπτικοί παράγοντες υιοθεσίας Νεφοϋπολογιστικής [33]

1. **Ασφάλεια:** Υπάρχει μια μεγάλη φοβία σχετικά με θέματα που αφορούν την ασφάλεια σε υποδομές υπολογιστικού νέφους. Στη Νεφοϋπολογιστική οι χρήστες μοιράζονται πόρους όπως μνήμη, επεξεργαστική ισχύ και αποθηκευτικό χώρο γεγονός που έχει εισάγει νέα θέματα σχετικά με την ασφάλεια [40], για

την επίλυση των οποίων απαιτούνται νέες τεχνικές. Τα θέματα ιδιωτικότητας CPU και μνήμης προσπαθούν να επιλυθούν μέσω διαφόρων τεχνικών της εικονοποίησης (π.χ. Xen) που απομονώνουν τελείως το ένα εικονικό περιβάλλον από το άλλο [41]. Η αποθήκευση και η μεταφορά δεδομένων όμως είναι πολύ πιο ευάλωτες σε επιθέσεις.

2. **Costing Model:** Ένας πελάτης της Νεφοϋπολογιστικής πρέπει να εξετάσει τι κερδίζει και τι χάνει όσον αφορά τους τομείς της υπολογιστικής ισχύς, της επικοινωνίας και της ενσωμάτωσης. Ενώ η μεταφορά σε υποδομές υπολογιστικού νέφους μπορεί να μειώσει σημαντικά το κόστος υποδομής, αυξάνει σημαντικά το κόστος μεταφοράς των δεδομένων ενός οργανισμού από και προς την υποδομή [42].

3. **Charging Model:** Από την πλευρά ενός παρόχου η διαθεσιμότητα ελαστικών πόρων έχει κάνει την ανάλυση κόστους πολύ πιο περίπλοκη σε σχέση με τα παραδοσιακά data centers στα οποία το κόστος συνήθως υπολογίζεται βάσει κατανάλωσης στατικών πόρων. Επιπλέον μια εικονική μηχανή έχει γίνει η μονάδα ανάλυσης του κόστους κι όχι ο υποκείμενος φυσικός εξυπηρετητής. Ένα μοντέλο χρέωσης πρέπει να ενσωματώσει όλα τα παραπάνω καθώς και ό,τι σχετίζεται με μια εικονική μηχανή όπως άδειες χρήση λογισμικού, εικονική χρήση δικτύου και ούτω καθεξής. Ένα στρατηγικό και βιώσιμο μοντέλο χρέωσης είναι ζωτικής σημασίας για τη βιωσιμότητα και κερδοφορία των (SaaS ιδίως) παρόχων.

4. **Διαθεσιμότητα Υπηρεσίας:** Οι οργανισμοί ανησυχούν για το εάν η Νεφοϋπολογιστική θα προσφέρει επαρκή διαθεσιμότητα κι αυτό τους κάνει αρκετά επιφυλακτικούς. Ο πίνακας 3.1 [35] παρουσιάζει καταγεγραμμένες διακοπές λειτουργίας των Amazon Simple Storage Service, Google AppEngine και Gmail καθώς και τον λόγο της διακοπής.

Πίνακας 3.1: Διακοπές και αιτία διακοπής υπηρεσίας [35]

Υπηρεσία & λόγος διακοπής υπηρεσίας	Διάρκεια	Ημερομηνία
S3: authentication service overload leading to unavailability	2h	15/02/08
S3: Single bit error leading to gossip protocol blowup	6-8h	20/07/08
Google AppEngine: programming error	5h	17/06/08
Gmail: site unavailable due to outage in contacts system	1,5h	11/08/08

5. **Εγκλωβισμός δεδομένων (Data Lock-In):** Τα λογισμικά σήμερα λειτουργούν διαλειτουργικά και συνεργάζονται με διάφορες πλατφόρμες, αλλά τα APIs για τη Νεφοϋπολογιστική είναι ιδιόκτητα ή τουλάχιστον δεν έχουν τυποποιηθεί. Αυτό έχει ως αποτέλεσμα να μην μπορούν οι χρήστες να μεταφέρουν εύκολα τα δεδομένα τους. Πέραν της δυσκολίας εξαγωγής, ένας άλλος παράγοντας είναι ότι για υπερβολικά μεγάλο όγκο δεδομένων, με τις υπάρχουσες υποδομές δικτύου,η

μεταφορά των δεδομένων είναι υπερβολικά χρονοβόρα. Οι δύο δυσκολίες αυτές αποτρέπουν οργανισμούς από το να υιοθετήσουν μια λύση Νεφοϋπολογιστικής.

6. **Νομοθεσία:** Σε κάθε επιχείρηση, πολλές εφαρμογές χρησιμοποιούν και αποθηκεύουν προσωπικά και ευαίσθητα δεδομένα σχετικά με τους πελάτες και τους υπαλλήλους τους. Οι ιδιοκτήτες των εφαρμογών αυτών πρέπει να υπακούουν τη νομοθεσία [25] περί χρήση προσωπικών δεδομένων της χώρας στην οποία λειτουργούν. Κάποιες χώρες έχουν νομοθετικούς περιορισμούς που απαγορεύουν τη αποθήκευση τέτοιων δεδομένων σε κάποιες άλλες χώρες. Αυτό περιορίζει ένα από τα χαρακτηριστικά της Νεφοϋπολογιστικής που τη θέλει να είναι ανεξάρτητη από τοποθεσία.

Ο πίνακας 3.2 συνοψίζει τις βασικές ανησυχίες των εν δυνάμει πελατών της Νεφοϋπολογιστκής και τις υποσχέσεις διαφόρων εμπορικών IaaS υποδομών. Οι ανησυχίες χωρίζονται σε τρείς βασικές κατηγορίες: (i) χαρακτηριστικά ασφαλείας, (ii) ευκολία μετανάστευσης και (iii) αξιοπιστία.

Πίνακας 3.2: Ανησυχίες Χρηστών και Πάροχοι

	Εγγυήσεις Cloud								
	Μειώσεις Κόστους / Βελτιστοποιήσεις		Επεκτασιμότητα και Αυτοματισμοί				Επιλογές κι Ευελιξία		
	Ποικιλία Πλάνων Τιμολόγησης	*Μέσο Μηνιαίο Κόστος*	*Scale Up*	*Scale Out*	*APIs*	*Μηχανισμοί Παρακολούθησης*	*Datacenters*	*Instance Types*	*Υποστηριζόμενα Λειτουργικά Συστήματα*
Πάροχοι									
Επικρατέστεροι									
Amazon (EC2)	3	$66,43	2	2	2	2	7	14	9
Rackspace	1	$116,80	2	2	2	1	8	8	9
GoGrid	4	$116,80	2	2	2	0	3	7	5
Microsoft	3	$65,70	2	2	0	1	6	4	6
Terremark	1	$138,90	2	2	2	0	9	32	6
AT&T	1	$121,30	2	2	0	0	26	100	2
Google	1	$42,42	2	2	2	0	11	4	2
OpSource	2	$95,63	2	2	2	1	4	100	4
Softlayer	2	$182,50	2	2	2	2	6	100	7
Ανερχόμενοι									
BitRefinery	0	$71,20	2	2	0	0	4	100	4
Lunacloud	1	$45,78	2	2	2	0	5	100	8
Nephoscale	2	$91,25	0	2	2	0	1	13	4
Tier3	1	$142,35	2	2	2	0	5	32	6

3.4 Σύγκριση Επιχειρηματικών Λύσεων

Η σύγκριση που γίνεται και παρουσιάζεται στον πίνακα 3.3 επικεντρώνεται στους παρόχους IaaS των οποίων οι υπηρεσίες μπορούν να αγοραστούν απευθείας μέσω διαδικτύου χωρίς να απαιτείται επαφή με πωλητές. Η επιλογή τους έγινε βάσει των διαθέσιμων πληροφοριών που υπάρχουν για τους παρόχους αυτούς στο διαδίκτυο.

Για να πραγματοποιηθεί η σύγκριση δημιουργήθηκαν κάποιες διαστάσεις που αντικατοπτρίζουν τα σημαντικότερα πλεονεκτήματα της Νεφοϋπολογιστκής, όπως η υπόσχεση για μειωμένο κόστος, το επίπεδο υπηρεσιών που προσφέρεται στους πελάτες, η ευελιξία στη διαμόρφωση των εξυπηρετητών. Τα κριτήρια που τελικώς χρησιμοποιούνται είναι:

1. **Κόστος Τιμολόγησης:** Οι πάροχοι προσφέρουν pay-as-you-go (συνήθως με την ώρα) πλάνα τιμολόγησης, μηνιαία, εκπτώσεις "μελών" (όπου ο χρήστης λαμβάνει μια έκπτωση με αντάλλαγμα μια επιπλέον ετήσια συνδρομή), ή οποιονδήποτε συνδυασμό αυτών. Όσο περισσότερες επιλογές παρέχονται τόσο το καλύτερο, αλλά το pay-as-you-go μοντέλο είναι η πιο ενδιαφέρουσα επιλογή δεδομένου ότι επιτρέπει περισσότερη λεπτομέρεια στη χρήση του.
2. **Μέσο Μηνιαίο Κόστος:** Εκτιμώμενο κατά μέσο όρο κόστος σε αμερικάνικα δολάρια για ένα μηχάνημα 1CPU και 2GB RAM (ή τη πλησιέστερη καλύτερη επιλογή) από παρόχους που χρεώνουν βάσει τοποθεσίας και βάσει λειτουργικού συστήματος (Windows ή Linux). Όποτε ήταν διαθέσιμο χρησιμοποιήθηκε η χρέωση ανά ώρα ενώ όποτε δεν ήταν διαθέσιμο η χρέωση ανά μήνα. Τα κόστη μεταφοράς δεδομένων δεν έχουν συμπεριληφθεί καθώς είναι εκτός του σκοπού της σύγκρισης
3. **Service-Level Agreement (SLA)):** Ο χρόνος διαθεσιμότητας υπηρεσίας που προσφέρει το SLA (ανεξαρτήτως προηγούμενων επιδόσεων) που μετράται σε ποσοστιαίες μονάδες.
4. **Αριθμός Datacenters:** Ο αριθμός των Datacenter που είναι διαθέσιμα για επιλογή.
5. **Scale Up:** Εάν είναι εφικτή η κλιμάκωση πόρων όπως CPU και RAM.
6. **Scale Down:** Εάν είναι εφικτή η άμεση δημιουργία νέων instances εικονικών μηχανών.
7. **Μηχανισμοί Παρακολούθησης:** Ο αριθμός των μηχανισμών που προσφέρονται από τους παρόχους χωρίς επιπλέον κόστος.
8. **APIs:** Εάν παρέχονται ή όχι APIs για την αλληλεπίδραση των χρηστών με τις εικονικές μηχανές.
9. **Αριθμός Διαφορετικών Τύπων Instances:** Ο αριθμός των διαφορετικών διαμορφώσεων που διατίθενται.
10. **Αριθμός Λειτουργικών Συστημάτων:** Ο αριθμός των λειτουργικών συστημάτων που διατίθενται από το πάροχο.

Μέσα από τη σύγκριση αυτή γίνεται εμφανές ότι ο κάθε πάροχος διαθέτει πληθώρα τεχνοοικονομικών μοντέλων. Τα μοντέλα αυτά στοχεύουν στη κάλυψη διαφορετικών ομάδων πελατών με τη κάθε ομάδα να έχει τις δικές της ξεχωριστές απαιτήσεις.

Πίνακας 3.3: Εγγυήσεις Νεφοϋπολογιστικής και Πάροχοι

	Ανησυχίες Χρηστών				
	Χαρακτηριστικά Ασφαλείας		Ευκολία Μετανάστευσης	Αξιοπιστία	
	Πιστοποιήσεις	*Προστασία*	*Standards*	*Χρόνια Ζωής Υπηρεσίας*	*SLA*
Πάροχοι					
Επικρατέστεροι					
Amazon (EC2)	2	1	OXI	5	99,95%
Rackspace	2	0	OpenStack	5	100,00%
GoGrid	0	1	OXI	4	100,00%
Microsoft	2	0	OXI	0	99,95%
Terremark	2	1	VMware	2	100,00%
AT&T	2	0	VMware	4	0,00%
Google	2	0	OXI	0	0,00%
OpSource	2	1	VMware	5	100,00%
Softlayer	2	1	OpenStack	5	100,00%
Ανερχόμενοι					
BitRefinery	2	0	VMware	2	100,00%
Lunacloud	0	0	OXI	0	0,00%
Nephoscale	0	0	OXI	1	99,95%
Tier3	2	0	VMware	5	99,90%

3.5 Συμπέρασμα

Η Νεφοϋπολογιστική, προσφέρει πραγματικά οφέλη στους πελάτες, που αναζητούν ένα ανταγωνιστικό πλεονέκτημα στη σημερινή οικονομία. Τα περισσότερα πλεονεκτήματά της Νεφοϋπολογιστικής προέρχονται από διαφορετικές πηγές, αλλά στο τέλος όλα καταλήγουν στο φτηνότερο κόστος. Όλο και περισσότεροι πάροχοι κινούνται προς την περιοχή αυτή με αποτέλεσμα οι τιμές να μειώνονται ακόμα περισσότερο. Οι πάροχοι με τη σειρά τους προκειμένου να είναι ανταγωνιστικοί και να αυξήσουν τη πελατειακή τους βάση προσφέρουν διαφορετικά μοντέλα κοστολόγησης με διαφορετικές παροχές προς τους πελάτες. Έτσι το πιο προβεβλημένο χαρακτηριστικό της Νεφοϋπολογιστικής, ο ισχυρισμός ότι προσφέρει καλύτερες υπηρεσίες σε χαμηλότερη τιμή στο τελικό χρήστη, γίνεται πραγματικότητα όταν υπάρχει ένας δυναμικός συνδυασμός των παρεχόμενων επιχειρηματικών μοντέλων του ενδιάμεσου με τους διαθέσιμους προσφερόμενους πόρους του παρόχου της υποδομής. Το φτηνότερο, για το πελάτη κόστος, γίνεται πραγματικότητα λόγω της ελαστικότητας και των μοντέλων που παρέχει. Στο Σχήμα 3.3 [11] η IDC απεικονίζει το κόστος που μπορεί να εξοικονομηθεί με χρήση τεχνολογιών Νεφοϋπολογιστικής.

Σχήμα 3.3: Έξοδα ενός IT τμήματος [11]

Πολλοί πελάτες εξακολουθούν να μην έχουν σαφή αντίληψη για τη Νεφοϋπολογιστική και για το αν τελικά αξίζει η αξιοποίηση της, καθώς μαζί με τα πλεονεκτήματα όπως η μείωση του κόστους σε προσωπικό κι εξοπλισμό, η παροχή 24/7 υπηρεσιών υπάρχουν και οι αμφιβολίες σχετικά με την ασφάλεια, τεχνολογική ανανέωση και νομοθεσία.

ΚΕΦΑΛΑΙΟ 4

Μοντέλα Ελαστικότητας

"I try not to break the rules but merely to test their elasticity."
˜Bill Veeck

4.1 Εισαγωγή

Όπως αναφέρθηκε σε προηγούμενα κεφάλαια η Νεφοϋπολογιστική έχει τραβήξει τόσο το ενδιαφέρον του εταιρικού όσο και του ακαδημαϊκού κόσμου. Στη βιβλιογραφία πλέον η υιοθέτηση της Νεφοϋπολογιστικής από εταιρίες και ερευνητικά ιδρύματα αποτελεί κοινό τόπο. Η ελαστικότητα, είναι το βασικό χαρακτηριστικό της Νεφοϋπολογιστικής που της επιτρέπει να χρησιμοποιείται ως λύση σε διάφορα προβλήματα [43] και το χαρακτηριστικό αυτό που τη ξεχωρίζει από τα υπόλοιπα μοντέλα όπως οι συστοιχίες και τα πλέγματα. Από τη πλευρά του παρόχου η ελαστικότητα εξασφαλίζει καλύτερη χρήση υπολογιστικών πόρων και επιτρέπει σε πολλούς χρήστες να εξυπηρετούνται ταυτόχρονα. Από τη σκοπιά του πελάτη, η ελαστικότητα χρησιμοποιείται κυρίως ως μέσο αποφυγής χρήσης περισσοτέρων ή λιγότερων πόρων. Άλλες μελέτες αναφέρουν ότι η ελαστικότητα χρησιμοποιείται για τη μείωση του κόστους [44, 45] και την εξοικονόμηση ενέργειας [46]. Στο [47] αναφέρεται πως η ελαστικότητα είναι αυτή που επιτρέπει σε υπηρεσιοστρεφείς εφαρμογές να αξιοποιούν τα οφέλη της Νεφοϋπολογιστικής.

Το κεφάλαιο αυτό στοχεύει:

1. **Σημασία Ελαστικότητας:** Η ανάδειξη της σημασίας της ελαστικότητας ως βασικό χαρακτηριστικό της Νεφοϋπολογιστικής.

2. **Ορισμός:** Η παράθεση και ανάλυση του ορισμού της ελαστικότητας καθώς και των ορίων και των βασικών πτυχών του ορισμού αυτού.
3. **Συναφείς Όροι:** Η αποσαφήνιση των διαφορών της ελαστικότητας με συναφείς όρους και έννοιες όπως αυτή της Επεκτασιμότητας.
4. **Σύγχρονες τάσεις:** Η επισκόπηση των σύγχρονων τάσεων (state of the art) τόσο σε εμπορικό όσο και σε ερευνητικό επίπεδο.
5. **Ανοικτά Θέματα:** Αναφορά και ανάλυση των ανοικτών θεμάτων, σχετικών με την ελαστικότητα, που αφορούν την παρούσα ερευνητική εργασία.

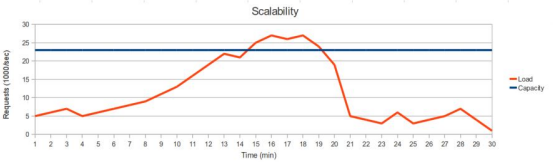
4.2 Υποθετικό Παράδειγμα Ελαστικότητας

Στον πραγματικό κόσμο τα ποσοστά χρήσης των εξυπηρετητών στα datacenter κυμαίνετε μεταξύ 5% και 20% [48]. Το ποσοστό αυτό μπορεί να ακούγεται υπερβολικά χαμηλό αλλά συνάδει με τη παρατήρηση που έχει γίνει ότι για τις περισσότερες εφαρμογές το μέγιστο φόρτο εργασίας ξεπερνάει το μέσο, σε κλίμακα 2 στα 10. Σχεδόν κανένας πάροχος εφαρμογής δεν δεσμεύει πόρους λιγότερους από αυτούς που απαιτούνται για την ικανοποίηση του αναμενόμενου μέγιστου φόρτου. Με τη πρακτική αυτή οι πόροι παραμένουν σε αδράνεια τις περισσότερες ώρες. Για να καταλάβει κανείς τα οικονομικά οφέλη που προσφέρει η ελαστικότητα μπορεί να σκεφτεί το εξής απλό παράδειγμα:

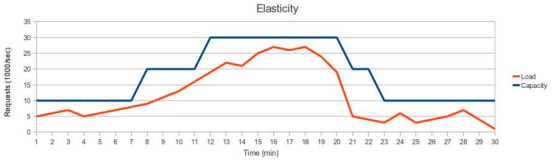
Έστω μια εφαρμογή όπου στο μέγιστο της λειτουργίας της, μεταξύ 5 και 8 το απόγευμα, απαιτεί 400 εξυπηρετητές για την λειτουργία της. Τις υπόλοιπες ώρες της ημέρας μπορεί να λειτουργήσει απρόσκοπτα με 200. Ο μέσος όρος στο παράδειγμα αυτό είναι 300 εξυπηρετητές ανά ημέρα που σημαίνει 7200 ώρες χρήσης εξυπηρετητών σε μια μέρα. Με παραδοσιακά μοντέλα ο ιδιοκτήτης της εφαρμογής θα έπρεπε να πληρώνει για 400 εξυπηρετητές όλες της ώρες της ημέρας ώστε να καλύπτει τις ώρες αιχμής της εφαρμογής, γεγονός που σημαίνει 9600 ώρες χρήσης εξυπηρετητών. Στην περίπτωση που κάποιος έχει επιλέξει μια λύση Νεφοϋπολογιστικής μπορεί εύκολα να χρησιμοποιεί 200 εξυπηρετητές και στις ώρες αιχμής 400 ώστε να καλύπτει τις ανάγκες της εφαρμογής του, ενώ με παραδοσιακά μοντέλα θα έπρεπε να πληρώνει για 400 εξυπηρετητές δηλαδή 9600 ώρες χρήσης που στην πραγματικότητα δεν αξιοποιεί.

Το παραπάνω απλοϊκό παράδειγμα υποτιμά σε μεγάλο βαθμό τα οικονομικά οφέλη που προσφέρει η ελαστικότητα, καθώς πραγματικές εφαρμογές και ιστοσελίδες, όπως για παράδειγμα το Taxisnet ή η σελίδα των Ολυμπιακών Αγώνων, έχουν μεγάλο φόρτο εργασίας σε συγκεκριμένες περιόδους μέσα στο χρόνο κι όχι σε ημερήσια βάση.

Στο Σχήμα 4.1 και Σχήμα 4.2 γίνεται μια σύγκριση μεταξύ της παραδοσιακής μεθόδου προσέγγισης χρήσης πόρων και της σύγχρονής ελαστικής προσέγγισης. Στην πρώτη περίπτωση (Σχήμα 4.1) στο σύστημα έχουν χορηγηθεί πόροι με σκοπό να ικανοποιήσουν το μέγιστο φόρτο εργασίας.



Σχήμα 4.1: Παραδοσιακή προσέγγιση



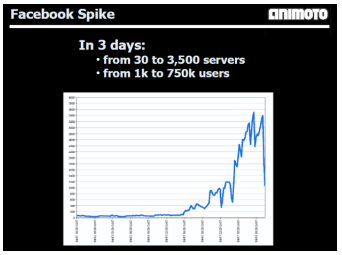
Σχήμα 4.2: Ελαστική προσέγγιση

Το περισσότερο διάστημα οι πόροι αυτοί δεν χρησιμοποιούνται ενώ τη στιγμή που υπάρχει μέγιστος φόρτος εργασίας, είναι μεγαλύτερος από αυτόν που είχε προβλεφθεί. Στη δεύτερη περίπτωση (Σχήμα 4.2) το σύστημα αντιδρά δυναμικά στην αύξηση του φόρτου εργασίας καθώς προσαρμόζεται βάσει των αναγκών, αφήνοντας μόνο ένα μικρό μέρος των δεσμευμένων πόρων σε αδράνεια.

4.3 Μελέτη Περίπτωσης: Animoto

Παραπάνω αναφέρθηκε ένα υποθετικό παράδειγμα που αποδεικνύει τα οικονομικά οφέλη της ελαστικότητας. Στην ενότητα αυτή θα παρατεθεί συνοπτικά μια πραγματική μελέτη περίπτωσης που αποδεικνύει το παραπάνω παράδειγμα και έχει χρησιμοποιηθεί ευρύτατα στη βιβλιογραφία.

Όταν η υπηρεσία Animoto, μια υπηρεσία δημιουργίας και επεξεργασίας video, έγινε διαθέσιμη μέσω Facebook έγινε άμεσα δημοφιλής με αποτέλεσμα να οδηγηθεί σε αύξηση από 30 σε 3.500 εξυπηρετητές [49] μέσα σε τρείς μόνο μέρες. Ακόμα κι αν η μέση χρησιμοποίηση του κάθε εξυπηρετητή ήταν χαμηλή, ήταν αδύνατο για τον οποιονδήποτε να προβλέψει ότι οι ανάγκες σε πόρους θα διπλασιάζονταν ξαφνικά κάθε δώδεκα ώρες για τρείς μέρες. Αφού ο μέγιστος φόρτος εργασίας υποχώρησε η κίνηση έπεσε σε επίπεδα αρκετά χαμηλότερα.



Σχήμα 4.3: Χρήση πόρων από την υπηρεσία Animoto

Σε αυτό το πραγματικό, παγκοσμίου επιπέδου παράδειγμα, βλέπει κανείς ότι η ελαστικότητα για scale-up δεν ήταν ένας τρόπος βελτιστοποίησης του κόστους αλλά απαραίτητη λειτουργική ανάγκη ενώ η ελαστικότητα για scale-down επέτρεψε η τρέχουσα κατάσταση δαπανών για υλικό να συμπίπτει με το τρέχον φόρτο εργασίας.

4.4 Επεκτασιμότητα

Ένα σύνηθες φαινόμενο είναι να συγχέεται ο όρος της επεκτασιμότητας με αυτόν της ελαστικότητας. Καθώς οι περισσότεροι βιβλιογραφικοί ορισμοί της ελαστικότητας αναφέρονται στην επεκτασιμότητα στη παρούσα ενότητα θα γίνει μια συνοπτική ανάλυση του όρου ώστε στη συνέχεια να μπορεί να γίνει ένας ξεκάθαρος διαχωρισμός του από την έννοια της ελαστικότητας.

Η επεκτασιμότητα είναι ένας όρος που μπορεί να εφαρμοστεί τόσο σε εφαρμογές όσο και σε πλατφόρμες εκτέλεσης [50].

1. **Επεκτασιμότητα εφαρμογής:** Είναι μια ιδιότητα που σημαίνει ότι η εφαρμογή θα διατηρήσει τους στόχους απόδοσης ακόμα κι εάν ο φόρτος εργασίας της αυξηθεί, έως ένα ορισμένο ανώτατο όριο. Η ύπαρξη αυτή του ανωτάτου ορίου τονίζει το γεγονός ότι η επεκτασιμότητα δεν είναι ατελείωτη και άνω του ορίου αυτού η εφαρμογή δεν θα μπορεί να διατηρήσει τους στόχους απόδοσης. Η επεκτασιμότητα της εφαρμογής περιορίζεται από τον σχεδιασμό της εφαρμογής καθώς και από τη χρήση των πόρων της πλατφόρμας εκτέλεσης που κάνει η εφαρμογή. Για παράδειγμα μια εφαρμογή που είναι ικανή να χρησιμοποιήσει δύο πυρήνες CPU δεν θα μπορέσει να κάνει scale σε μια οκταπύρινη μηχανή. Φυσικά για να μπορεί να εφαρμογή να κάνει σωστή χρήση της επεκτασιμότητας θα πρέπει η πλατφόρμα πάνω στην οποία εκτελείται να μπορεί να της παρέχει τους πόρους που απαιτεί.
2. **Επεκτασιμότητα πλατφόρμας:** Είναι η ικανότητα μιας πλατφόρμας εκτέλεσης να παρέχει όσους πρόσθετους πόρους χρειαστούν ή ζητηθούν από μια εφαρμογή. Υπάρχουν δύο διαφορετικά είδη επεκτασιμότητας πλατφόρμας κι ένα σύστημα μπορεί να είναι επεκτάσιμο και στα δύο ή σε κανένα από τα δύο:
    - **Κάθετη κλιμάκωση** (scale vertically) ή scale up: Είναι η προσθήκη περισσότερων πόρων σε ένα κόμβο της πλατφόρμας όπως πυρήνες CPU, RAM κτλ. ώστε ο κόμβος αυτός να μπορεί να χειριστεί μεγαλύτερο φόρτο εργασίας. Η μείωση πόρων αντίστοιχα ονομάζεται scale down.

- **Οριζόντια κλιμάκωση** (scale horizontally) ή scale out: Είναι η προσθήκη περισσότερων κόμβων (π.χ. εικονικές ή φυσικές μηχανές) σε ένα σύμπλεγμα ή ένα κατανεμημένο σύστημα ώστε ολόκληρο το σύστημα να μπορεί να χειριστεί μεγαλύτερο φόρτο εργασίας. Η μείωση κόμβων αντίστοιχα ονομάζεται scale in.

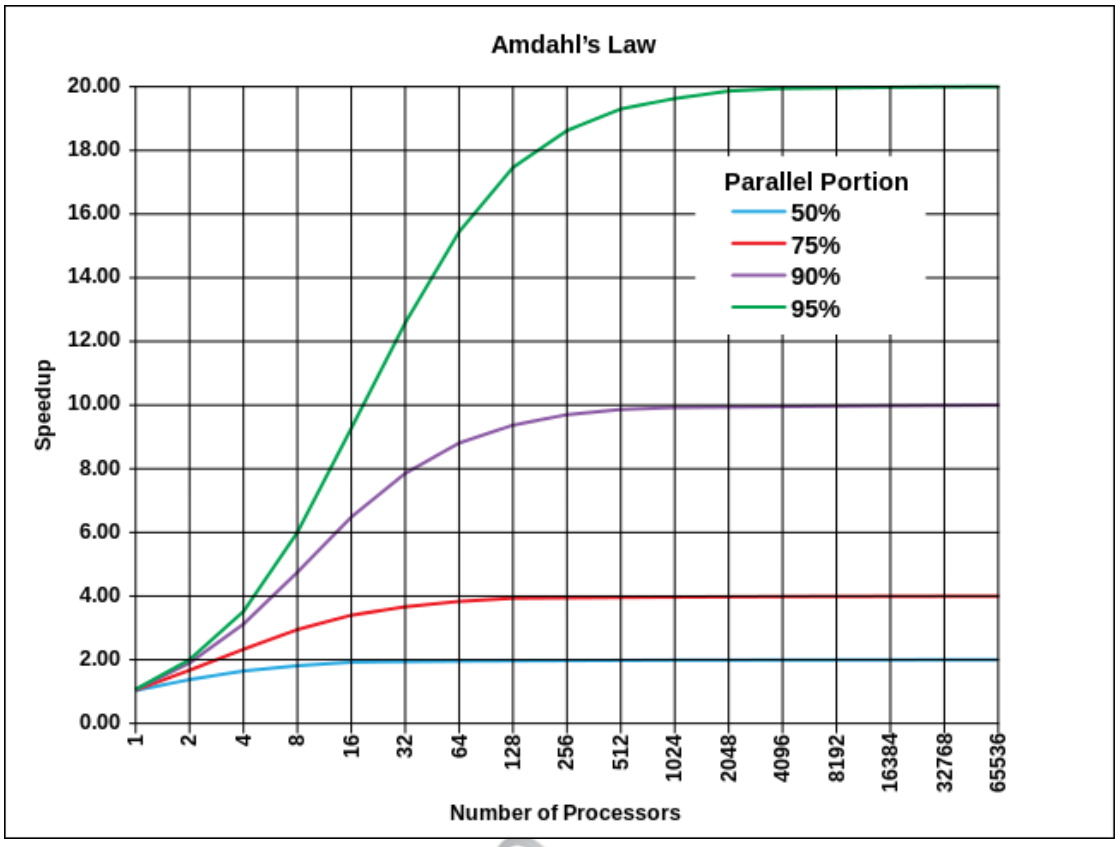
Επεκτασιμότητα είναι η ικανότητα του συστήματος να διευρυνθεί σε ένα μέγεθος το οποίο αναμένεται να φιλοξενήσει μια μελλοντική εφαρμογή ή την ικανότητα του να βελτιώσει την απόδοση του όταν προστίθενται επιπλέον πόροι. Σε ένα επεκτάσιμο περιβάλλον Νεφοϋπολογιστικής μπορούν να προστίθενται πόροι όποτε η ζήτηση αυξάνεται, προκειμένου οι εφαρμογές που εκτελούνται να λειτουργούν στο απαιτούμενο επίπεδο. Με τη σειρά της, μια εφαρμογή λέγεται επεκτάσιμη όταν η αποτελεσματικότητά της διατηρείται σε περίπτωση που το ποσό των πόρων και το μέγεθος του προβλήματος αυξάνονται αναλογικά. Η επεκτασιμότητα της εφαρμογής αντανακλά ικανότητα στην αξιοποίηση των διαθέσιμων πόρων αποτελεσματικά.

Κάθε πραγματικό σύστημα και εφαρμογή έχει τα όρια του, τα οποία καθορίζονται από το περιβάλλον του και των ενδιαφερόμενων, μέσω λειτουργικών και μη λειτουργικών απαιτήσεων. Συνήθως, κάθε εφαρμογή έχει ένα μη παραλληλισμό κομμάτι. Το μέγεθος του κομματιού αυτού καθορίζει το κατώτερο όριο, όσον αφορά το χρόνο εκτέλεσης ενός προγράμματος σύμφωνα με το νόμο του Amdahl [51]. Ο νόμος μπορεί να εκφραστεί ως μια μαθηματική συνάρτηση (4.1) η οποία δίνει τη μέγιστη επιτάχυνση (S) η οποία μπορεί να επιτευχθεί με N κόμβους να δουλεύουν παράλληλα,

$$S(N) = \frac{1}{(1-P) + \frac{P}{N}} \tag{4.1}$$

όπου P είναι το κομμάτι που μπορεί να εκτελεστεί παράλληλα και αντιστρόφως (1-P) είναι το σειριακό κομμάτι. Καθώς το N τείνει στο άπειρο το S τείνει στο 1/(1-P). Για παράδειγμα εάν ένας υπολογισμός έχει ένα σειριακό τμήμα το οποίο αποτελεί το 10% του πλήρους υπολογισμού, τότε το όφελος για την αύξηση του παραλληλισμού για το υπόλοιπο 90% του υπολογισμού θα τείνει στο μηδέν όσο αυξάνει ο αριθμός των παράλληλων κόμβων. Στο παράδειγμα αυτό το άνω όριο για το S(N) είναι 10 ανεξαρτήτως του αριθμού των παράλληλων επεξεργαστών που εισάγονται στο σύστημα. Στο Σχήμα 4.4 απεικονίζεται το παράδειγμα αυτό μαζί με άλλες τιμές του P.

Ο νόμος του Amdahl υπογραμμίζει τη σημασία της αλγοριθμικής βελτιστοποίησης ώστε να μεγιστοποιηθεί η επιτάχυνση (S) που μπορεί να επιτευχθεί με παράλληλη διαδικασία. Ωστόσο, μολονότι το όφελος της προσθήκης περισσότερων παράλληλων κόμβων τείνει στο μηδέν, η επεξεργαστική ισχύς των κόμβων αυτών δεν μειώνεται.



Σχήμα 4.4: Νόμος του Amdahl [52]

Στην πραγματικότητα, οι παράλληλοι κόμβοι παραμένουν σε αδράνεια για (1-P)% της εκτέλεσης.

Όπως φαίνεται, ο ορισμός της επεκτασιμότητας, δεν λαμβάνει υπόψη του το παράγοντα του χρόνου. Η επεκτασιμότητα ορίζει ότι όταν απαιτηθούν επιπλέον πόροι θα δοθούν και θα χρησιμοποιηθούν αλλά δεν ορίζει το πόσο γρήγορα οι πόροι αυτοί πρέπει να δοθούν.

4.5 Ελαστικότητα

Όπως ο ίδιος ο όρος της Νεφοϋπολογιστικής έχει τις ρίζες του σε άλλες επιστήμες έτσι και η ονομασία του βασικού του χαρακτηριστικού, της ελαστικότητας, προέρχεται από την επιστήμη της φυσικής. Στην φυσική η ελαστικότητα ορίζεται ως μια ιδιότητα των υλικών η οποία αντιπροσωπεύει την ικανότητα τους να επιστρέψουν στην αρχική τους μορφή - σχήμα αφότου έχουν υποστεί μια παραμόρφωση. Αποτελεί μια διαισθητική έννοια και μπορεί να περιγραφεί επακριβώς μόνο μέσω μαθηματικών τύπων. Ο όρος

εισήχθηκε στα πλαίσια της Νεφοϋπολογιστικής για λόγους μάρκετινγκ και χρησιμοποιείται ευρύτατα στις διαφημίσεις των παρόχων και αρκετές φορές ακόμα και στην ονομασία της ίδιας της υπηρεσίας [50].

Παρ' όλες τις τεράστιες προσπάθειες και επενδύσεις που έχουν γίνει και συνεχίζουν να γίνονται, ώστε να επιτραπεί στα συστήματα Νεφοϋπολογιστικής να λειτουργούν με ελαστικό τρόπο, δεν έχει ακόμα επιτευχθεί ένας κοινός ορισμός ούτε η ακριβής κατανόηση του όρου. Στο [53] παρατίθενται πέντε ορισμοί από τα που κάνουν την παραπάνω παρατήρηση εμφανή:

1. "[…] ορίζει την ελαστικότητα ως την παραμετροποίηση και την επεκτασιμότητα της λύσης […] Κυρίως αποτελεί την ικανότητα να γίνει scale up και / ή scale down ανάλογα το φόρτο εργασίας"

2. "Ταχεία ελαστικότητα: Οι πόροι μπορούν να δεσμευτούν προς χρήση γρήγορα και ελαστικά, σε ορισμένες περιπτώσεις αυτόματα, έτσι ώστε να εμφανιστούν άμεσα ως μη διαθέσιμοι (scale out) και επίσης να αποδεσμευτούν γρήγορα για να εμφανιστούν ξανά ως διαθέσιμοι (scale in). Για τον καταναλωτή, οι διαθέσιμες δυνατότητες για δέσμευση και χρήση συχνά φαίνεται να είναι απεριόριστες και μπορούν να αγοραστούν ανά πάσα στιγμή και σε οποιαδήποτε ποσότητα."

3. "Η ελαστικότητα είναι βασικά μια 'μετονομασία' της κλιμάκωσης […]" και "αφαιρεί οποιαδήποτε χειρωνακτική εργασία που απαιτείται για την αύξηση ή τη μείωση της παραγωγικής ικανότητας".

4. "Η ελαστικότητα μετρά την ικανότητα της Νεφοϋπολογιστικής να χαρτογραφεί ένα ενιαίο αίτημα του χρήστη σε διαφορετικούς πόρους"

5. "Ελαστικότητα είναι η μετρήσιμη ικανότητα διαχείρισης, μέτρησης, πρόβλεψης και ταχύτητας προσαρμογής μιας εφαρμογής βασισμένη σε πραγματικού χρόνου αιτήματα σε μια υποδομή που χρησιμοποιεί ένα συνδυασμό τοπικών και απομακρυσμένων υπολογιστικών πόρων "

Αναλύοντας τους παραπάνω ορισμούς στο [53] οι Herbst, Kounev και Reussner παρατηρούν ότι οι ορισμοί (1),(2) και (3) περιγράφουν από κοινού την ελαστικότητα ως την κλιμάκωση των πόρων ενός συστήματος για την αύξηση ή τη μείωση της παραγωγικής ικανότητας ενώ οι ορισμοί (1),(2) και (5) αναφέρουν ρητά ότι οι παρεχόμενοι πόροι σχετίζονται με το τρέχων φόρτο εργασίας. Οι ορισμοί (4) και (5) προσπαθούν να συλλάβουν την ελαστικότητα σε ένα γενικό πλαίσιο ως μια "ποσοστικοποιημένη" ικανότητα ενός συστήματος να διαχειριστεί αιτήματα χρησιμοποιώντας διάφορους

πόρους. Ο ορισμός (3) δηλώνει ότι δεν απαιτείται κανένα είδους χειρονακτική εργασία ενώ στον ορισμό του NIST (2) φαίνεται ότι οι διαδικασίες που επιτρέπουν την ελαστικότητα δεν είναι πάντα αυτοματοποιημένες. Επιπλέον στον ορισμό του NIST έχει προστεθεί το "ταχεία" ώστε να σχηματίζεται η ιδανική εικόνα της τέλειας ελαστικότητας όπου ατελείωτοι πόροι είναι διαθέσιμοι με μια κατάλληλη μέθοδο παροχής τους σε οποιαδήποτε χρονική στιγμή με τέτοιο τρόπο που ο τελικός χρήστης δεν αντιλαμβάνεται οποιαδήποτε διακύμανση των επιδόσεων.

Κατά την αξιολόγηση της ελαστικότητας τα ακόλουθα σημεία πρέπει να ελέγχονται εκ των προτέρων:

1. **Αυτόνομη Κλιμάκωση:** Τι διαδικασία προσαρμογής χρησιμοποιείται για αυτόνομη επεκτασιμότητα;
2. **Διαστάσεις Ελαστικότητας:** Ποιο είναι το σύνολο των τύπων των πόρων που κλιμακώνονται ως μέρος της διαδικασίας προσαρμογής;
3. **Μονάδες Επεκτασιμότητας πόρων:** Για κάθε τύπο πόρου, σε τι μονάδα μεταβάλλεται ο αριθμός των πόρων;
4. **Όρια Επεκτασιμότητας:** Για κάθε τύπο πόρου, ποιο είναι το μέγιστο όριο, στο ποσό των πόρων που μπορούν να διατεθούν;

4.5.1 Ορισμός Ελαστικότητας

Ο ορισμός που έχει προταθεί και ακολουθεί και η παρούσα εργασία προτάθηκε από τους οι Herbst, Kounev και Reussner [53] αναφέρει:

Ορισμός 4.1 (Ορισμός Ελαστικότητας) :
"Η ελαστικότητα είναι ο βαθμός στον οποίο ένα σύστημα είναι σε θέση να προσαρμοστεί σε αλλαγές στο φόρτο εργασίας προσφέροντας και αφαιρώντας πόρους με έναν αυτόνομο τρόπο, έτσι ώστε σε κάθε χρονική στιγμή οι διαθέσιμοι πόροι να καλύπτουν την τρέχουσα ζήτηση όσο το δυνατόν καλύτερα"

4.5.2 Διαστάσεις και Βασικές Πτυχές Ορισμού

Κάθε δεδομένη διαδικασία προσαρμογής ορίζεται στο πλαίσιο τουλάχιστον ενός ή ενδεχομένως πολλαπλών τύπων πόρων που μπορούν να γίνουν scale up ή down ως μέρος της προσαρμογής. Κάθε τύπος πόρου μπορεί να θεωρηθεί ως μια ξεχωριστή διάσταση της διαδικασίας προσαρμογής με τις δικές του ιδιότητες ελαστικότητας. Εάν

ένας τύπος πόρου αποτελεί ένα 'δοχείο' πολλαπλών τύπων πόρων, όπως για παράδειγμα μια εικονική μηχανή που του έχουν ανατεθεί πυρήνες CPU και μνήμης RAM, η ελαστικότητα μπορεί να θεωρηθεί σε πολλαπλά επίπεδα. Κανονικά, οι πόροι ενός συγκεκριμένου τύπου φυσικών πόρων μπορούν να δοθούν σε διακριτές μονάδες όπως πυρήνες CPU, εικονικές μηχανές ή φυσικούς κόμβους. Για κάθε διάσταση της διαδικασίας προσαρμογής σε σχέση με ένα συγκεκριμένο τύπο πόρου η ελαστικότητα περιλαμβάνει τις ακόλουθες βασικές πτυχές της προσαρμογής:

1. **Ταχύτητα:** Η ταχύτητα κλιμάκωσης ορίζεται ως ο χρόνος που χρειάζεται για να γίνει η μετάβαση από μια κατάσταση με έλλειψη πόρων σε μια βέλτιστη κατάσταση. Η ταχύτητα της αποκλιμάκωσης αντίστοιχα ορίζεται ως ο χρόνος που απαιτείται για να γίνει η μετάβαση από μια κατάσταση με περίσσεια πόρων σε μια βέλτιστη κατάσταση.

2. **Ακρίβεια:** Η ακρίβεια της κλιμάκωσης ορίζεται ως η απόλυτη απόκλιση του τρέχοντος ποσού των διατιθέμενων πόρων από την πραγματική ζήτηση των πόρων.

Όπως αναφέρθηκε παραπάνω, η ελαστικότητα θεωρείται πάντα σε σχέση με έναν ή περισσότερους τύπους πόρων. Έτσι, η άμεση σύγκριση μεταξύ δύο συστημάτων σε σχέση με την ελαστικότητα τους, μπορεί να γίνει μόνο εάν η κλιμάκωση γίνει με τους ίδιους τύπους πόρων, που μετρούνται σε πανομοιότυπες μονάδες.

Για να αξιολογηθεί η πραγματική παρατηρημένη ελαστικότητα σε ένα δεδομένο σενάριο, πρέπει να καθοριστεί το κριτήριο βάσει του οποίου το ποσό των πόρων που θα δοθούν θεωρείται ότι ταιριάζει με την πραγματική τρέχουσα ζήτηση που απαιτείται για να ικανοποιηθούν δεδομένες απαιτήσεις απόδοσης του συστήματος.

4.5.3 Διαφοροποιήσεις

Όπως φάνηκε από τις προηγούμενες ενότητες ο όρος της ελαστικότητας συγχέεται με τον όρο της επεκτασιμότητας. Ένας άλλος όρος που συγχέεται είναι αυτός της αποδοτικότητας. Στη παρούσα ενότητα παρουσιάζεται η διαφορά των όρων αυτών με την ελαστικότητα.

1. **Διαφοροποίηση με Επεκτασιμότητα:** Η επεκτασιμότητα αποτελεί προϋπόθεση για την ελαστικότητα, αλλά δεν λαμβάνει υπόψη χρονικές πτυχές όπως πόσο γρήγορα και πόσο συχνά πραγματοποιούνται ενέργειες επεκτασιμότητας. Η επεκτασιμότητα είναι η ικανότητα του συστήματος να διατηρήσει τον αυξανόμενο

φόρτο εργασίας κάνοντας χρήση πρόσθετων πόρων και ως εκ τούτου, σε αντίθεση με την ελαστικότητα, δεν έχει σχέση με το πόσο καλά οι πραγματικές απαιτήσεις των πόρων ταιριάζουν με τους πόρους που παρέχονται σε κάθε χρονική στιγμή.

2. **Διαφοροποίηση με Αποδοτικότητα:** Η αποδοτικότητα εκφράζει το ποσό των πόρων που καταναλώνονται για την επεξεργασία μιας συγκεκριμένης ποσότητας εργασίας. Σε αντίθεση με την ελαστικότητα, η απόδοση δεν περιορίζεται στους τύπους των πόρων που κλιμακώνονται ως μέρος των μηχανισμών προσαρμογής του συστήματος. Κανονικά, καλύτερη ελαστικότητα συνεπάγεται υψηλότερη αποδοτικότητα, όμως αυτή η σχέση δεν είναι δεδομένη καθώς η αποδοτικότητα μπορεί να επηρεάζεται και από άλλους παράγοντες ανεξάρτητους των μηχανισμών ελαστικότητας του συστήματος, όπως για παράδειγμα διάφορες υλοποιήσεις της ίδιας διαδικασίας.

4.6 Κατάταξη Ελαστικών Λύσεων

Στόχος της ενότητας αυτής είναι να μελετήσει την κατάταξη των ελαστικών λύσεων που προτάθηκε στο [54]. Η κατάταξη αυτή βασίζεται σε τέσσερα χαρακτηριστικά και απεικονίζεται στο Σχήμα 4.5:

1. **Πεδίο Εφαρμογής:** Καθορίζει το που ελέγχονται οι δράσεις της ελαστικότητας (IaaS, PaaS ή SaaS). Σε γενικές γραμμές, τα IaaS νέφη, έχουν ένα ελεγκτικό μηχανισμό ελαστικότητας ο οποίος είναι υπεύθυνος για τη μετατροπή των απαιτήσεων των χρηστών σε δράσεις που παρέχονται από τις IaaS υποδομές. Ο ελεγκτικός μηχανισμός χρησιμοποιεί δεδομένα παρακολούθησης από τις εφαρμογές και λαμβάνει την απόφαση για το εάν πρέπει ή όχι να προβεί σε κλιμάκωση πόρων [55].

   Ο έλεγχος της ελαστικότητας μπορεί να εκτελεστεί και από την ίδια την εφαρμογή ή από την πλατφόρμα εκτέλεσης της. Σε αυτή την περίπτωση, ο ελεγκτικός μηχανισμός είναι ενσωματωμένος στην εφαρμογή ή εντός του περιβάλλοντος εκτέλεσης, που αλληλεπιδρά με το νέφος με σκοπό να απελευθερώσει ή να αιτηθεί νέους πόρους. Στις περιπτώσεις αυτές, τα χαρακτηριστικά της ελαστικότητας πρέπει να υποστηρίζονται από την υποκείμενη υποδομή. Η προσέγγιση αυτή συνήθως εφαρμόζεται σε PaaS νέφη, τα οποία χρησιμοποιούν περιβάλλοντα εκτέλεσης (που ονομάζονται containers) για την αυτόματη διαχείριση των πόρων που χρησιμοποιούνται από τις εφαρμογές [55].

2. **Πολιτική:** Σχετίζεται με τις αλληλεπιδράσεις που απαιτούνται για την εκτέλεση των δράσεων της ελαστικότητας. Δύο είναι οι κύριες πολιτικές που χρησιμοποιούνται:

   - **Χειροκίνητη** (manual): Σημαίνει ότι ο χρήστης είναι υπεύθυνος για την παρακολούθηση του εικονικού του περιβάλλοντος και τις εφαρμογές του καθώς και για την εκτέλεση όλων των ελαστικών ενεργειών. Ο πάροχος πρέπει να παρέχει τουλάχιστον μια διεπαφή (συνήθως ένα API) μέσω της οποίας ο χρήστης αλληλεπιδρά με το σύστημα.
   - **Αυτόματη** (automatic): Με τη πολιτική αυτή ο έλεγχος και οι ενέργειες πραγματοποιούνται από την υποδομή νέφους ή από την εφαρμογή, σύμφωνα με τους κανόνες και τις ρυθμίσεις του χρήστη, ή που διευκρινίζονται στο SLA. Το σύστημα ελέγχου χρησιμοποιεί υπηρεσίες παρακολούθησης για τη συλλογή πληροφοριών όπως η χρήση CPU, RAM, κίνηση δικτύου κτλ., ώστε να αποφασίσει πότε και πώς να προβεί στη κλιμάκωση ή την αποκλιμάκωση πόρων. Ανάλογα με τη τεχνική που χρησιμοποιείται για την ενεργοποίηση των ελαστικών δράσεων οι αυτόματες πολιτικές μπορούν να χωριστούν σε δύο κατηγορίες:

     – **Αντίδρασης** (reactive): Βασίζονται στο μοτίβο "Κανόνας – Κατάσταση - Δράση". Ένα τέτοιο παράδειγμα του μοτίβου αυτού φαίνεται στο ψευδοκώδικα που παρατίθεται στη συνέχεια [54]. Ένας κανόνας αποτελείται από ένα σύνολο προϋποθέσεων οι οποίες όταν ικανοποιηθούν ενεργοποιούν κάποιες προκαθορισμένες δράσεις στο υποκείμενο νέφος. Κάθε προϋπόθεση ικανοποιείται από ένα γεγονός ή ένα μετρικό το οποίο συγκρίνεται με ένα όριο. Οι σχετικές με τις μετρήσεις πληροφορίες και τα γεγονότα παρέχονται από το σύστημα παρακολούθησης της υποδομής ή της εφαρμογής.

```
RULE:
if CONDITION(s) then ACTION(s)

CONDITION:
(1...\*) (if metric.value = threshold)||(if event(s) occurs)

ACTION:
(*) Cloud-enabled actions (e.g. add / remove VM)
```

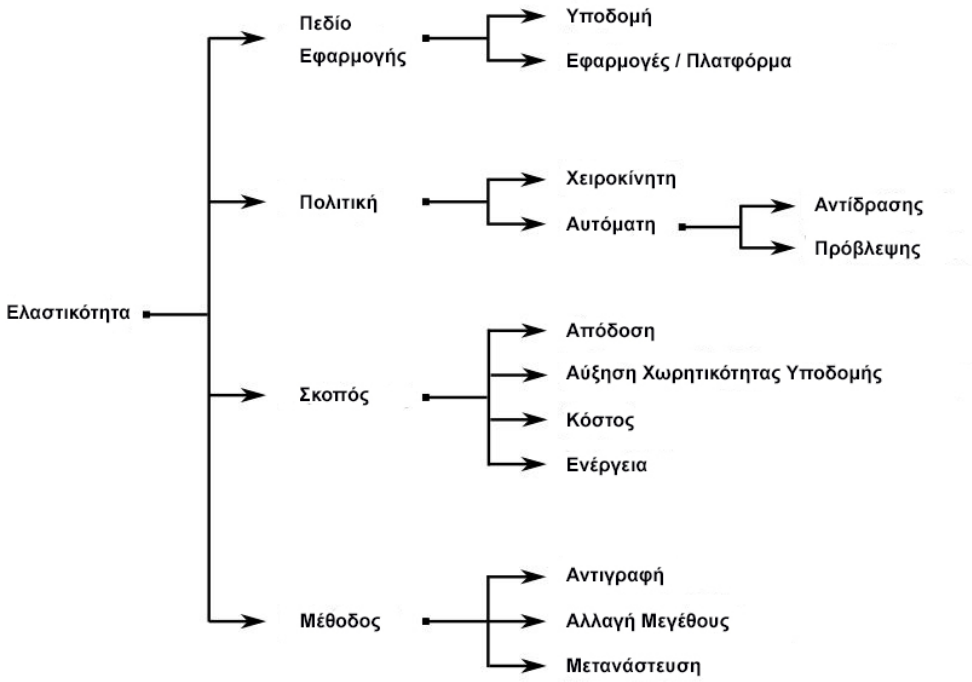
     – **Πρόβλεψης** (predictive): Η προσέγγιση αυτή χρησιμοποιεί ευρετικές και μαθηματικές / αναλυτικές τεχνικές για τη πρόβλεψη της συμπεριφοράς του συστήματος σε φόρτο εργασίας και βάσει των αποτελεσμάτων αυτών αποφασίζει πότε και πώς θα προβεί στην κλιμάκωση ή την αποκλιμάκωση πόρων.

3. **Σκοπό:** Η ελαστικότητα στη Νεφοϋπολογιστική χρησιμοποιείται για διαφόρους σκοπούς. Στα πλαίσια της κατηγοριοποίησης που προτείνεται θεωρείται ότι ο σκοπός θα υπάγεται σε μία (ή περισσότερες) από της παρακάτω κατηγορίες:

    - Καλύτερη απόδοση,
    - Αύξηση χωρητικότητας υποδομής,
    - Μείωση του κόστους,
    - Εξοικονόμηση ενέργειας

4. **Μέθοδο:** Το τελευταίο χαρακτηριστικό αναφέρεται στις μεθόδους που χρησιμοποιούνται κατά την εφαρμογή των λύσεων ελαστικότητας. Τρείς βασικές μέθοδοι υπάρχουν:

    - **Αντιγραφή** (replication): Αλλιώς ονομάζεται οριζόντια κλιμάκωση, και συνεπάγεται με την προσθήκη / αφαίρεση instances από το εικονικό περιβάλλον του χρήστη. Αποτελεί σήμερα τη πλέον διαδεδομένη μέθοδο για παροχή ελαστικότητας.
    - **Αλλαγή μεγέθους** (resizing): Αλλιώς ονομάζεται κάθετη κλιμάκωση, πόροι όπως CPU, RAM ή αποθηκευτικός χώρος μπορούν να προστεθούν / αφαιρεθούν από ένα εν λειτουργία εικονικό instance.
    - **Μετανάστευση** (migration): Είναι η μετάβαση μιας εικονικής μηχανής από ένα εξυπηρετητή σε έναν άλλο. Συνήθως χρησιμοποιείται για τη προσομοίωση τη συμπεριφορά που επιτυγχάνεται με την αλλαγή μεγέθους σε νέφη που δεν επιτρέπουν τέτοια ενέργεια.



Σχήμα 4.5: Κατάταξη ελαστικών μηχανισμών [54]

4.7 Σύγχρονες Τάσεις

Η ενότητα αυτή χωρίζεται σε δύο μέρη. Στο πρώτο μέρος γίνεται μια ανασκόπηση τριών εμπορικών πλατφορμών και περιγράφεται η προσέγγιση τους στην ελαστικότητα εφαρμογών. Στο δεύτερο μέρος αναλύονται διάφορα ερευνητικά έργα.

Στόχος της ενότητας αυτής είναι να μελετήσει τις διάφορες λύσεις με βάση τη κατάταξη που αναφέρθηκε στη προηγούμενη ενότητα.

4.7.1 Εμπορικές Πλατφόρμες

Όπως έχουμε δει στο δεύτερο κεφάλαιο υπάρχουν τρία μοντέλα παροχής υπηρεσιών (βλέπε και Σχήμα 2.5). Οι υπάρχουσες εμπορικές πλατφόρμες διαχωρίζονται βάσει των τριών αυτών μοντέλων. Έχουν επιλεγεί τρείς εμπορικές πλατφόρμες που ακολουθούν διαφορετικές προσεγγίσεις στην παροχή ελαστικών υπηρεσιών. Σε κάθε πλατφόρμα αρχικά γίνεται μια γενική επισκόπηση και στη συνέχεια αξιολογείται η ελαστική της προσέγγιση. Στο [56] πραγματοποιείται ένα συγκριτικό τεστ μεταξύ των τριών αυτών παρόχων και των δυνατοτήτων τους.

4.7.1.1 Amazon EC2

Σχήμα 4.6: Λογότυπο του Amazon EC2

Ένας πελάτης "ανεβάζει" μια εικόνα του συστήματος του και το λειτουργεί στους εξυπηρετητές της Amazon. Ο πελάτης είτε παρέχει αυτός το λειτουργικό σύστημα και τις εφαρμογές με τις οποίες θα λειτουργεί η εικονική μηχανή, είτε επιλέγει μια από τις έτοιμες εικόνες που παρέχει η Amazon. Για κάθε instance ο πελάτης επιλέγει τον τύπο (που προσδιορίζει την υπολογιστική ικανότητα σε μεγέθη CPUs, RAM,αποθηκευτικό χώρο κτλ.) και μια από τις τρείς επιλογές πληρωμής:

1. **On-Demand instances:** Είναι ευέλικτα και πληρώνονται βάσει των ωρών που λειτουργούν.
2. **Reserved instances:** Εγγυούνται υπολογιστική ικανότητα οποιαδήποτε στιγμή.
3. **Spot instances:** Επιτρέπουν στον πελάτη να ορίσει το μέγιστο ποσό που διατίθεται να πληρώνει με την ώρα. Ανάλογα με τη τρέχουσα χρήση του νέφους η τιμή ανά ώρα αλλάζει. Όποτε η τιμή είναι μικρότερη του ορίου το instance εκτελείται αλλιώς τερματίζεται.

Το EC2 παρέχει έναν auto-scale μηχανισμό για instances εφαρμογών που λειτουργεί ως εξής: Ο χρήστης ορίζει τα κατώτερα και τα ανώτερα όρια για CPU, μνήμη ή χρήση δικτύου και μια χρονική περίοδο (σε λεπτά) κατά την οποία η τιμή αξιολογείται. Εάν υπάρξει υπέρβαση ορίου ο αριθμός των instances προσαρμόζεται (ξεκινάει ένα νέο instance ή ένα υπάρχον τερματίζεται). Ο χρόνος αντίδρασης είναι το λιγότερο ένα λεπτό συν το χρόνο που απαιτείται για την εκκίνηση ενός νέου instance. Ένας ελαστικός Load Balancer παρέχεται στο TCP στρώμα ώστε να κατανέμει το φορτίο στα instances που βρίσκονται σε λειτουργία.

Το Amazon EC2 παρέχει δύο επιλογές για αποθήκευση δεδομένων:

1. **Elastic Block Store:** Είναι μια μη διαμορφωμένη block συσκευή με χωρητικότητα από 1GB ως 1TB. Μπορεί να χρησιμοποιηθεί ως μια block συσκευή για εικονικές μηχανές ή να χρησιμοποιηθεί για αποθήκευση αρχείων.
2. **Amazon Simple Storage Service (S3):** Λειτουργεί ως μια κλιμακούμενη data object αποθήκευση. Ο χρήστης μπορεί να αποθηκεύσει data objects μεγέθους

από 1byte μέχρι 5GB τα οποία χαρακτηρίζονται από ένα μοναδικό κλειδί και μπορεί να τα ανακτήσει αργότερα. Παρέχει και Representational State Transfer (REST) και Simple Object Access Protocol (SOAP) διεπαφές.

Το Amazon EC2 αποτελεί μια IaaS υποδομή κι έτσι δίνει στο πελάτη τη δυνατότητα να εκτελέσει εφαρμογές σε μια αυθαίρετη πλατφόρμα με τυχόν πιθανές ρυθμίσεις. Οι εφαρμογές μπορούν να κλιμακωθούν αυτόματα, μονάδα κλιμάκωσης είναι όλο το instance ενώ ο χρόνος αντίδρασης μετράται σε λεπτά.

4.7.1.2 Microsoft Windows Azure

Το Azure [20] είναι μια IaaS και PaaS υποδομή που προσφέρεται από τη Microsoft. Βασίζεται σε καθιερωμένα προϊόντα της Microsoft (το λειτουργικό σύστημα Windows, τη .Net πλατφόρμα και τον SQL Server) και έχει ως στόχο να επιτρέψει την εύκολη μεταφορά των υφιστάμενων εφαρμογών στο νέφος.

Το νέφος, παρέχει το λειτουργικό σύστημα, το περιβάλλον εκτέλεσης καθώς και κατανεμημένη βάση δεδομένων. Από τον πελάτη ζητείται μόνο να "ανεβάσει" και να ρυθμίσει την εφαρμογή του. Η εφαρμογή ονομάζεται υπηρεσία και αποτελείται από έναν ή περισσότερους ρόλους. Κάθε ρόλος μπορεί να εκτελεστεί σε ένα ή περισσότερα instances. Υπάρχουν δύο είδη ρόλων:

Σχήμα 4.7: Λογότυπο του Microsoft Windows Azure

1. **Web Role:** Είναι βέλτιστος για προγραμματισμό διαδικτυακών εφαρμογών και υποστηρίζεται από τον IIS [1] και το ASP.NET. [2]
2. **Worker Role:** Προορίζεται για την ανάπτυξη γενικών εφαρμογών.

Οι υπηρεσίες αναπτύσσονται και γίνονται compile με τη χρήση των Windows Azure Tools [3] . Ένας προγραμματιστής μπορεί να χρησιμοποιήσει οποιαδήποτε γλώσσα προγραμματισμού υποστηρίζεται από τη .NET πλατφόρμα.

[1] http://www.iis.net/
[2] http://www.asp.net/
[3] http://msdn.microsoft.com/en-us/library/ee405484.aspx

Είναι δυνατό να εκτελεστούν αρκετά instances ενός ρόλου. Τα instances αυτά ομαδοποιούνται αυτόματα σε upgrade domains. Είναι δυνατό να γίνουν αλλαγές στις ρυθμίσεις ενός instance και να γίνει επανεκκίνηση των instances. Αυτό μπορεί να γίνει άμεσα με τη προϋπόθεση ότι δεν υπάρχουν νέοι ρόλοι στην εφαρμογή. Ο πελάτης μπορεί να πραγματοποιήσει τις αλλαγές αυτές είτε μέσω μιας διαδικτυακής διεπαφής είτε μέσω ενός REST API. Υπάρχει ένα διαγνωστικό API μέσω του οποίου μπορεί κανείς να πάρει πληροφορίες σχετικές με το instance (αριθμός αιτήσεων, χρήση CPU, δίσκου, μνήμης, δικτύου κτλ.). Δεν υπάρχει όμως κάποιο εργαλείο για να τροποποιήσει τις ρυθμίσεις του instance αυτόματα. Ο πελάτης μπορεί να ρυθμίσει ειδοποιήσεις μέσω ηλεκτρονικού ταχυδρομείου και να πραγματοποιήσει τις αλλαγές χειροκίνητα μέσω του REST API. Το παράθυρο χρέωσης για το χρόνο εκτέλεσης ενός instance είναι η μια ώρα.

Οι υπηρεσίες αποθήκευσης του Azure μπορούν να αποθηκεύσουν δυαδικά αντικείμενα, ουρές ή πίνακες και παρέχουν πρόσβαση μέσω ενός REST API. Η Microsoft επίσης παρέχει το SQL AZURE που αποτελεί μια σχεσιακή βάση βασισμένη στον SQL που παρέχει την ίδια διεπαφή. Στο datacenter δημιουργείται μια ρεπλίκα της βάσης δεδομένων και η ανάκτηση των δεδομένων είναι εγγυημένη από το πάροχο. Το μέγεθος μια βάσης δεδομένων είναι περιορισμένο στα 10GB αλλά ένας λογαριασμός πελάτη μπορεί να περιέχει πολλές βάσεις.

Το Azure βασίζεται στη .NET πλατφόρμα κι έτσι είναι εφικτό για τον προγραμματιστή να χρησιμοποιεί όλες τις διαθέσιμες βιβλιοθήκες και τα εργαλεία ανάπτυξης. Δεν παρέχει κάποια ενσωματωμένη υποστήριξη για αυτόματη κλιμάκωση των εφαρμογών. Δεν είναι δυνατό να γίνει αλλαγή στον αριθμό των instances σε εκτέλεση άμεσα. Εάν ο αριθμός των instances αλλάξει πρέπει να γίνει επανεκκίνηση σε όλο το domain.

4.7.1.3 Google App Engine

Σχήμα 4.8: Λογότυπο του App Engine

Το Google App Engine [19] είναι μια PaaS υποδομή που προσφέρεται από τη Google. Ισχυρίζεται ότι έχει γραφτεί ώστε να είναι ανεξάρτητη προς γλώσσες προγραμματισμού πλατφόρμα αλλά προς το παρών υπάρχει υποστήριξη για java, Python, PHP και GO.

Οι εφαρμογές απομονώνονται από τη πλατφόρμα και δίνεται πρόσβαση σε αυτές μέσω του παρεχόμενου API.

Το App Engine είναι βελτιστοποιημένο για διαδικτυακές εφαρμογές (μια εφαρμογή μπορεί να κληθεί από ένα

HTTP / HTTPS αίτημα). Αφού κληθεί μια εφαρμογή έχει μέγιστο όριο 30 δευτερόλεπτα να ολοκληρώσει την εκτέλεση της και να επιστρέψει την απάντηση της.

Ο πελάτης δεν μπορεί να επιλέξει υλικό για το instance της εφαρμογής καθώς όλες οι εφαρμογές εκτελούνται σε ένα ενιαίο περιβάλλον και υπολογίζονται από τα quotas ή limits. Υπάρχουν τριών ειδών quotas/limits [4]:

1. **Free Quotas:** Είναι όρια που καθορίζονται στις δωρεάν εφαρμογές. Τα quotas αυτά μπορούν να ξεπεραστούν μόνο από εφαρμογές επί πληρωμή, μέχρι το όριο του budget της εφαρμογής ή του ορίου ασφαλείας (ανάλογα ποιο θα τεθεί πρώτο σε ισχύ).
2. **Billable Limits:** Ισχύουν για εφαρμογές επί πληρωμή και δεν μπορούν να ξεπεραστούν. Τα όρια αυτά καθορίζονται από το διαχειριστή της εφαρμογής στο τμήμα χρεώσεων της κονσόλας διαχείρισης. Τα quotas αυτά επιτρέπουν στους διαχειριστές να διαχειρίζονται το κόστος της εφαρμογής.
3. **Safety Limits:** Καθορίζονται από τη Google με σκοπό να διασφαλίσουν ότι μια μεμονωμένη εφαρμογή δεν θα προβεί σε υπερκατανάλωση πόρων σε βάρος άλλων εφαρμογών.

Για την αποθήκευση των δεδομένων το App Engine παρέχει το Data Store μια μη σχεσιακή άνευ σχήματος βάση δεδομένων.

Το App Engine προσφέρει ευκολία διαχείρισης της εφαρμογής καθώς οι λεπτομέρειες της εκτέλεσης είναι κρυφές από το χρήστη. Δεν είναι δυνατό να εφαρμοστεί μια ευρεία ποικιλία εφαρμογών . Το σύστημα των quota είναι περιοριστικό και δεν επιτρέπει υψηλή ελαστικότητα καθώς τα fixed quotas δεν μπορούν να αλλάξουν και τα quotas ανά λεπτό αποτρέπουν την εφαρμογή να αντιδράσει ευέλικτα σε σύντομες περιόδους υψηλής ζήτησης.

4.7.2 Ερευνητικά Έργα

Υπάρχει πληθώρα ερευνητικών έργων στη βιβλιογραφία που προσπαθούν να λύσουν ζητήματα σχετικά με την ελαστικότητα ή αναφέρουν ανοικτά ερευνητικά θέματα. Με βάση το [54], στη παρούσα ενότητα γίνεται μια προσπάθεια καταγραφής και κατηγοριοποίησης των έργων αυτών.

[4] https://developers.google.com/appengine/docs/quotas

4.7.2.1 Μέρος Πρώτο: IaaS Μηχανισμοί

Το έργο RESERVOIR [28] εφαρμόζει ένα μηχανισμό αντίδρασης. Η ελαστικότητα ορίζεται ρητά με την προσθήκη ελαστικών κανόνων στο Service Manifest [5]. Οι κανόνες αυτοί καθορίζουν τις προϋποθέσεις, που βασισμένες στη παρακολούθηση συγκεκριμένων γεγονότων στο επίπεδο εφαρμογής θα οδηγήσουν σε προκαθορισμένες δράσεις που διατίθενται από τη πλατφόρμα.

Οι Lime et al. [57] πρότειναν ένα μηχανισμό αντίδρασης που βασίζεται σε ένα δεδομένο μετρικό σύστημα αντί για ένα όριο όπως γίνεται στις εμπορικές πλατφόρμες, για την ενεργοποίηση των δράσεων. Σκοπός ήταν να επιλυθεί το πρόβλημα της ταλάντωσης των πόρων που προκαλείται από την ύπαρξη ενός μόνο ορίου. Το κύριο σημείο της πρότασης είναι πως το σύστημα αντιδρά μόνο όταν το μετρικό είναι εκτός του αναμενόμενου εύρους, μειώνοντας έτσι τις αχρείαστες κατανομές και ανακατανομές πόρων.

Πέραν τον προσεγγίσεων που βασίζονται σε κανόνες, μηχανισμοί πρόβλεψης έχουν χρησιμοποιηθεί για την παροχή αυτόματων ελαστικών λύσεων. Το κύριο χαρακτηριστικό τους είναι η χρήση τεχνικών για τη πρόβλεψη της συμπεριφοράς του συστήματος σε φόρτο εργασίας και η χρησιμοποίηση των προβλέψεων αυτών για τη λήψη της απόφασης του πώς θα αναβαθμιστούν οι πόροι. Παραδείγματα χρήσης τεχνικών πρόβλεψης παρουσιάζονται από τους Dawoud et al. [58], Gong et al. [59], Vasic et al. [60], Shen et al. [46] και Sharma et al. [44].

Οι Dawoud et al. [58] παρουσίασαν το Elastic VM, μια αρχιτεκτονική που αλλάζει τους πόρους της εικονικής μηχανής δυναμικά ώστε να ανταπεξέλθει με το φόρτο εργασίας και να διατηρήσει το συμφωνημένο επίπεδο υπηρεσίας. Σύμφωνα με τους συγγραφείς, το κίνητρο είναι να προσπαθήσουν να διαθέσουν τους ελάχιστους δυνατούς πόρους που απαιτούνται για το χειρισμό ενός δεδομένου φόρτου εργασίας. Για την πρόβλεψη της χρήσης CPU το Elastic VM χρησιμοποιεί ένα μηχανισμό βασισμένο σε προσαρμοστικό έλεγχο και εξετάζει την τελευταία κατάσταση χρήσης CPU καθώς και το ιστορικό χρήσης.

Το PRESS [59] αποτελεί ένα σύστημα πρόβλεψης που βασίζεται σε μοτίβα που δημιουργούνται με γρήγορους μετασχηματισμούς Φουριέ (Fast Fourier Transformation (FFT)). Το PRESS αρχικά χρησιμοποιεί το FFT για τον εντοπισμό επαναλαμβανόμενων μοτίβων, που ονομάζονται υπογραφές, και χρησιμοποιούνται για τις προβλέψεις. Εάν δεν ανακαλυφθεί καμία υπογραφή, το PRESS χρησιμοποιεί μια προσέγγιση βασισμένη σε στατιστική κατάσταση, ώστε να εντοπίσει βραχυπρόθεσμα μοτίβα ζήτησης πόρων

[5] Μια περιγραφή της υπηρεσίας που βασίζεται στο Open Virtualization Format (OVF)

και χρησιμοποιεί διακριτού χρόνου Μαρκοβιανές αλυσίδες για να προβλέψει τη ζήτηση στο εγγύς μέλλον.

Οι Vasic et al. [60], ανέπτυξαν το DejaVu, ένα πλαίσιο εργασίας για τη διαχείριση εικονικών πόρων στο νέφος. Η ιδέα πίσω από το DejaVu είναι να κατατάξει το φόρτο εργασίας σε κατηγορίες και να αποδώσει σε κάθε κατηγορία μια υπογραφή που παράγεται από τη σχέση μεταξύ του φόρτου εργασίας και των πόρων που χρησιμοποιούνται για να τον χειριστούν. Οι υπογραφές αυτές δημιουργούνται σε μια περίοδο εκμάθησης και αποθηκεύονται σε μια προσωρινή μνήμη (cache). Όταν το DejaVu ανιχνεύει αλλαγές στις συνθήκες του φόρτου εργασίας ελέγχει τη προσωρινή μνήμη και φορτώνει τις ρυθμίσεις πόρων που αντιστοιχούν στην υπογραφή.

Οι λύσεις που παρουσιάστηκαν μέχρι στιγμής δίνουν προτεραιότητα στην απόδοση του συστήματος, δίνοντας έμφαση στη κατανομή πόρων για να ανταπεξέλθουν στις δυναμικές απαιτήσεις μιας εφαρμογής. Άλλα έργα έχουν ασχοληθεί με διαφορετικά ζητήματα όπως αυτό της ενέργειας [46], το κόστος [8, 44] και ζητήματα διαχείρισης κόστους του παρόχου [61].

Το CloudScale [46] είναι ένα σύστημα που προέρχεται από το PRESS [59] το οποίο προσθέτει μηχανισμούς για μείωση της κατανάλωσης ενέργειας. Η λύση βασίζεται στην ικανότητα των σύγχρονων επεξεργαστών να λειτουργούν σε διαφορετικές τάσεις και να εναλλάσσονται μεταξύ αυτών δυναμικά. Ανάλογα με το αναμενόμενο φόρτο εργασίας του συστήματος, το CloudScale μπορεί να μειώσει ή να αυξήσει τη συχνότητα ή τη τάση της CPU.

Το ζήτημα του κόστους αντιμετωπίζεται από το Kingfisher [44]. Ο στόχος του συστήματος είναι να προσπαθήσει να ελαχιστοποιήσει το κόστος για την ανάπτυξη των εικονικών μηχανών των πελατών ενώ παράλληλα να είναι ελαστικό ως προς τις αλλαγές του φόρτου εργασίας. Το Kingfisher, λαμβάνει υπόψη το κόστος κάθε instance εικονικής μηχανής, τις πιθανότητες μετακίνησης ή αντιγραφής της εικονικής μηχανής καθώς επίσης και το χρόνο μετάβασης από τη μια διαμόρφωση στην άλλη. Τότε λύνει ένα ακέραιο γραμμικό πρόβλημα ώστε να παράξει το ελάχιστο κόστος ρύθμισης που απαιτείται για κάθε αλλαγή στο φόρτο εργασίας.

Οι Meng et al. [61] ασχολούται με το θέμα της ελαστικότητας στην διαχείριση των εσωτερικών πόρων του παρόχου. Σύμφωνα με τους συγγραφείς, τα καθήκοντα διαχείρισης (δημιουργία εικονικών μηχανών, προγραμματισμός μετανάστευσης, κτλ.) είναι υπολογιστικά δαπανηρά, συνήθως συμβαίνουν μαζεμένα και η έλλειψη πόρων για τη διαχείριση των συγκεκριμένων καθηκόντων μπορεί να επηρεάσει τους τελικούς χρήστες. Με αυτή τη παρατήρηση, προτάθηκε το TIDE, ένα αυτοκλιμακούμενο πλαίσιο

εργασίας για εικονικά κέντρα δεδομένων. Η λογική πίσω από το TIDE είναι να χειρίζεται το διαχειριστικό φόρτο εργασίας με τον ίδιο τρόπο που κάποιος θα χειριζόταν το φόρτο εργασίας μιας εφαρμογής. Όταν το TIDE συναντά εκρήξεις στη διαχείριση του φόρτου εργασίας ενεργοποιεί δυναμικά επιπλέον εικονικούς εξυπηρετητές με σκοπό να ενισχύσει τη συνολική απόδοση. Όταν ο φόρτος εργασίας της διαχείρισης μειωθεί, οι πόροι των διαχειριστικών αυτών instances μπορούν να χρησιμοποιηθούν για τις ανάγκες εφαρμογών των τελικών χρηστών.

Η ελαστικότητα έχει επίσης χρησιμοποιηθεί για την επέκταση των δυνατοτήτων κέντρων δεδομένων και ιδιωτικών νεφών. Οι Fit 'o et al.[62] παρουσίασαν το Cloud Hosting Provider (CHP) ένα ελαστικό διαδικτυακό πάροχο εξωτερικών (outsourced) πόρων με σκοπό να επωφεληθεί από υποδομές υπολογιστικών νεφών για την παροχή δυνατοτήτων υψηλής επεκτασιμότητας στις διαδικτυακές εφαρμογές που αναπτύσσονται σε αυτό. Προκειμένου να αποφευχθούν παραβιάσεις στο SLA ο χρονοπρογραμματιστής (scheduler) του CHP κατανέμει αυτόματα πόρους σε δημόσια νέφη με σκοπό να τρέξουν instances από διαδικτυακούς εξυπηρετητές σε περίπτωση που δεν μπορούν να τρέξουν σε τοπικά μηχανήματα.

Οι Marshall et al. [63] ανέπτυξαν το Elastic Site, μια πλατφόρμα για την επέκταση των πλεγμάτων με χρήση πόρων νέφους. Στο σύστημα χρησιμοποιήθηκε ένα Nimbus [6] ιδιωτικό δίκτυο και πόροι από το Amazon EC2. Όταν οι πόροι ζητούνται, ο ελεγκτής του πλέγματος μπορεί να διαθέσει αυτόματα μια μηχανή από το τοπικό πλέγμα ή μια εικονική μηχανή από το νέφος, εάν δεν υπάρχουν διαθέσιμοι φυσικοί πόροι.

Οι Zhang et al. [64] προτείνουν ένα ελαστικό μοντέλο εφαρμογής το οποίο επιτρέπει την απρόσκοπτη και διαφανή χρήση των πόρων του νέφους με σκοπό να αυξήσει την, περιορισμένη σε πόρους, ικανότητα των κινητών συσκευών. Ο στόχος του έργου είναι να σχεδιάσει μια αρχιτεκτονική και το σχετικό μεσολογισμικό ώστε να επιτρέψει σε ελαστικές εφαρμογές που αποτελούνται από πολλά δομικά στοιχεία, που ονομάζονται weblets, να λειτουργήσουν σε μια φορητή συσκευή ή στο νέφος. Η απόφαση για το που να ξεκινήσει ένα weblet λαμβάνεται όταν η εφαρμογή ξεκινά και πιθανώς τροποποιείται κατά τη διάρκεια της εκτέλεσης ενώ βασίζεται στη διαμόρφωση της εφαρμογής και / ή την κατάσταση της συσκευής, όπως ο φόρτος της CPU και τα επίπεδα της μπαταρίας.

Όπως είναι εμφανές οι λύσεις ελαστικότητας που έχουν προταθεί, χρησιμοποιούνται για την επίτευξη πολλαπλών στόχων όπως η βελτίωση των επιδόσεων, η μείωση του κόστους και η επέκταση των τοπικών πόρων.

[6] http://www.nimbusproject.org/about/

4.7.2.2 Μέρος Δεύτερο: Μηχανισμοί Εφαρμογής

Για να αξιοποιηθούν πλήρως τα οφέλη που προσφέρει η ελαστικότητα της Νεφοϋπολογιστικής, χρειάζεται παραπάνω από μια ελαστική υποδομή. Είναι απαραίτητο οι εφαρμογές να έχουν τη δυνατότητα να προσαρμόζονται δυναμικά ανάλογα με τις αλλαγές στις απαιτήσεις.

Σε γενικές γραμμές, εφαρμογές που έχουν αναπτυχθεί σε PaaS υποδομές εμπεριέχουν ρητά την έννοια της ελαστικότητας. Τα PaaS νέφη παρέχουν περιβάλλοντα εκτέλεσης (δοχεία), στα οποία οι χρήστες μπορούν να εκτελέσουν τις εφαρμογές τους χωρίς να χρειάζεται να ανησυχούν για το τι πόροι θα χρησιμοποιηθούν. Στις περιπτώσεις αυτές, το νέφος διαχειρίζεται αυτόματα τη κατανομή των πόρων ώστε οι προγραμματιστές να μη χρειάζεται να παρακολουθούν συνεχώς τη κατάσταση της υπηρεσίας ή να αλληλεπιδρούν με τη πλατφόρμα για να αιτηθούν για περισσότερους πόρους [55]. Υπάρχουν και εξαιρέσεις, όπως το Microsoft Windows Azure [20], που παρουσιάστηκε στην προηγούμενη ενότητα, στο οποίο ο χρήστης πρέπει να καθορίσει τους πόρους που χρησιμοποιούνται από τις εφαρμογές.

Ένα παράδειγμα PaaS πλατφόρμας με υποστήριξη ελαστικότητας είναι το Aneka [65]. Στο Aneka όταν μια εφαρμογή χρειάζεται περισσότερους πόρους, εκτελούνται νέα instances δοχείων, ώστε να διαχειριστούν τη ζήτηση, που χρησιμοποιούν τοπικούς ή δημόσιους πόρους του νέφους.

Μερικά επιστημονικά έργα έχουν παρουσιάσει μηχανισμούς ελαστικότητας για εφαρμογές, με κύριο στόχο να επιτρέψουν την ανάπτυξη ευέλικτων και προσαρμόσιμων εφαρμογών για περιβάλλοντα νέφους.

Ο Neamtiu [66] περιγράφει το Elastin, ένα πλαίσιο εργασίας που περιλαμβάνει ένα compiler και ένα runtime περιβάλλον, στόχος του οποίου είναι η μετατροπή ανελαστικών προγράμματων σε ελαστικές εφαρμογές. Η ιδέα πίσω από το Elastin είναι η χρήση ενός μεταγλωττιστή ο οποίος συνδυάσει διαφορετικές εκδόσεις του προγράμματος σε μια ενιαία εφαρμογή η οποία μπορεί να εναλλάσσεται μεταξύ διάφορων διαμορφώσεων κατά το χρόνο εκτέλεσης, χωρίς να τερματίζεται. Το εκτελέσιμο αρχείο αποθηκεύει πολλές διαμορφώσεις, κάθε μια για ένα δεδομένο σενάριο. Η επιλογή του ποια διαμόρφωση θα πρέπει να χρησιμοποιηθεί καθορίζεται από το χρήστη και μπορεί να αλλάξει κατά το χρόνο εκτέλεσης.

Οι Knauth & Fetzer [67] εξέτασαν τις streaming εφαρμογές εστιάζοντας στη μείωση της κατανάλωσης ενέργειας. Η προτεινόμενη λύση χρησιμοποιεί μετανάστευση και ενοποίηση εικονικών μηχανών με σκοπό την παροχή ελαστικότητας. Η βασική ιδέα είναι να ξεκινά κάθε εφαρμογή μέσα σε μια εικονική μηχανή. Όταν ο φόρτος εργασίας

είναι ο ελάχιστος, όλες οι εικονικές μηχανές ενοποιούνται σε ένα ελάχιστο σύνολο φυσικών μηχανών. Όταν ο φόρτος εργασίας αυξάνεται, οι εικονικές μηχανές μετεγκαθίστανται σε άλλους εξυπηρετητές, μέχρι κάθε φυσικός εξυπηρετητής να φιλοξενεί μια εικονική μηχανή.

Οι Rajan et al. [68] παρουσίασαν το WorkQueue, ένα πλαίσιο εργασίας για την ανάπτυξη ελαστικών master-slave εφαρμογών. Οι εφαρμογές που αναπτύσσονται με το WorkQueue επιτρέπουν τη προσθήκη σκλαβων αντιγράφων κατά τη διάρκεια εκτέλεσης. Οι σκλάβοι εφαρμόζονται ως εκτελέσιμα αρχεία που μπορούν να αρχικοποιηθούν από το χρήστη σε διαφορετικά μηχανήματα κατά τη ζήτηση. Όταν εκτελούνται οι σκλάβοι επικοινωνούν με τον master, που συντονίζει την εκτέλεση της εργασίας και την ανταλλαγή των δεδομένων.

Με την ανάπτυξη IaaS νεφών που υποστηρίζουν ελαστικότητα, είναι φυσιολογικό να προκύπτουν όλο και περισσότερες πλατφόρμες και εφαρμογές που εκμεταλλεύονται τη δυνατότητα αυτή. Η ελαστικότητα είναι ήδη σε χρήση στην ανάπτυξη επιχειρηματικών εφαρμογών, κυρίως σε PaaS και client - server εφαρμογές. Ωστόσο ακόμα υπάρχει έλλειψη σε εργαλεία και πλαίσια εργασίας που είναι ικανά να υποστηρίξουν την ανάπτυξη εφαρμογών που θα μπορούσαν να επωφεληθούν από την ελαστικότητα που παρέχεται από τα IaaS νέφη. Άλλα ανοικτά ζητήματα ελαστικότητας αναφέρονται σε επόμενη ενότητα. Στο πίνακα 4.1 συνοψίζονται όλα τα ερευνητικά έργα και οι εμπορικές πλατφόρμες που μελετήθηκαν.

4.8 Ανοικτά Θέματα Ελαστικότητας

Παρόλο που έχουν προταθεί και εφαρμοστεί πολλές προσεγγίσεις στο θέμα της ελαστικότητας υπάρχουν ακόμα ανοικτά θέματα που δεν έχουν επιλυθεί. Στη παρούσα ενότητα αναφέρονται τα τρία θέματα που είναι σχετικά με το αντικείμενο που πραγματεύεται η παρούσα εργασία ενώ στο [54] γίνεται μια προσπάθεια καταγραφής όλων των θεμάτων αυτών.

1. **Διαθεσιμότητα πόρων:** Η ελαστικότητα ενός παρόχου υποδομής υπολογιστικού νέφους περιορίζεται από την διαθεσιμότητα του σε πόρους με αποτέλεσμα οι πάροχοι να αναγκάζονται να επιβάλλουν αυστηρούς περιορισμούς (π.χ. Google App Engine [8]) στους πόρους που ένας χρήστης μπορεί να αποκτήσει σε μια συγκεκριμένη χρονική στιγμή ακυρώνοντας έτσι την υπόθεση των απεριόριστων πόρων [69].

[7] Στα ερευνητικά έργα χρησιμοποιείται η ονομασία του έργου (όπου υπάρχει) αντί , των συγγραφέων
[8] https://developers.google.com/appengine/docs/quotas

Πίνακας 4.1: Σύνοψη Κατάταξης

Σύστημα[7]	Πεδίο Εφαρμογής	Πολιτική	Μέθοδος	Σκοπός
Amazon EC2 [8]	Infrastructure	Αυτόματη-Αντίδρασης	Αντιγραφή	Απόδοση
Reservoir [28]	Infrastructure	Αυτόματη-Αντίδρασης	Αντιγραφή	Απόδοση
Lim et al. [57]	Infrastructure	Αυτόματη-Αντίδρασης	Αντιγραφή	Απόδοση
CHP [62]	Infrastructure	Αυτόματη-Αντίδρασης	Αντιγραφή	Αύξηση χωρητικότητας υποδομής
PRESS [59]	Infrastructure	Αυτόματη-Πρόβλεψης	Αλλαγή Μεγέθους	Απόδοση
Elastic VM [58]	Infrastructure	Αυτόματη-Πρόβλεψης	Αλλαγή Μεγέθους	Απόδοση
Elastic Site [63]	Infrastructure	Αυτόματη-Αντίδρασης	Αντιγραφή	Αύξηση χωρητικότητας υποδομής
TIDE [61]	Infrastructure	Αυτόματη-Αντίδρασης	Αντιγραφή	Απόδοση
Kingfisher [44]	Infrastructure	Αυτόματη-Πρόβλεψης	Αντιγραφή Μετανάστευση	Απόδοση Κόστος
CloudScale [46]	Infrastructure	Αυτόματη-Πρόβλεψης	Αλλαγή Μεγέθους	Απόδοση Ενέργεια
DejaVu [60]	Infrastructure	Αυτόματη-Πρόβλεψης	Αντιγραφή	Απόδοση
Aneka [65]	Platform	Αυτόματη-Αντίδρασης	Αντιγραφή	Αύξηση χωρητικότητας υποδομής
Azure [20]	Platform	Χειροκίνητη	Αντιγραφή	Απόδοση
Knauth & Fetzer [67]	Application	Αυτόματη-Αντίδρασης	Μετανάστευση	Απόδοση
Elastin [66]	Application	Χειροκίνητη	Αλλαγή Μεγέθους	Απόδοση
Rajan et al. [68]	Application	Χειροκίνητη	Αλλαγή Μεγέθους	Απόδοση
Zhang et al. [64]	Application	Χειροκίνητη	Αντιγραφή Μετανάστευση	Απόδοση Ενέργεια

2. **Βαθμός λεπτομέρειας (granularity) πόρων:** Στα περισσότερα IaaS νέφη οι πελάτες αποκτούν πόρους ως ένα σύνολο CPU, RAM και I/O (π.χ. οι τύποι instances του Amazon). Η ενοικίαση όμως ενός συνόλου πόρων δεν ικανοποιεί τις πραγματικές ανάγκες των πελατών. Ένα επιπρόσθετο σχετικό πρόβλημα είναι πως οι πάροχοι δεν επιτρέπουν την αλλαγή ενός instance χωρίς να γίνει επανεκκίνηση. Η δυνατότητα της δυναμικής μίξης διαφορετικών πόρων θα ήταν πολύτιμη για

την επίτευξη πραγματικής ελαστικότητας καθώς θα επέτρεπε στους χρήστες να ρυθμίσουν τους πόρους βάσει των αναγκών τους. Επιπλέον με τον τρόπο αυτό οι πάροχοι θα μπορούσαν να μεγιστοποιήσουν τα κέρδη τους χρεώνοντας δυναμικά για κάθε πόρο.

3. **Χρόνος εκκίνησης:** Το βασικό προσόν της ελαστικότητας είναι η ικανότητα που παρέχει να προσφέρει δυναμικά πόρους όταν ζητηθούν. Όμως στη δυναμική αυτή διαδικασία παρόλο που ο χρήστης έχει τη δυνατότητα να αιτηθεί για πόρους οποιαδήποτε στιγμή, μπορεί να απαιτηθεί κάποιο χρονικό διάστημα ώστε οι νεοαποκτηθέντες πόροι να είναι έτοιμη προς χρήση. Ο χρόνος αυτός έχει ονομαστεί χρόνος εκκίνησης (start-up time) ή Spin Up Time (SU) [70]. Στη θεωρία, σε ένα ελαστικό νέφος η παροχή των πόρων γίνεται στιγμιαία, στη πράξη όμως ο χρόνος εκκίνησης μπορεί να ποικίλλει (από 1 λεπτό - π.χ. Amazon EC2- έως 10 λεπτά) καθώς εξαρτάται από διάφορους παράγοντες όπως το τύπο της πλατφόρμας υπολογιστικού νέφους, το λειτουργικό σύστημα, τους πόρους που θα ζητηθούν κτλ.

ΚΕΦΑΛΑΙΟ 5

Επιχειρηματικά Μοντέλα

"Fact is, inventing an innovative business model
is often mostly a matter of serendipity."
˜Gary Hamel

5.1 Εισαγωγή

Η κινητήρια δύναμη για κάθε επιχείρηση ήταν πάντα τα εξελισσόμενα επιχειρηματικά μοντέλα που δίνουν έμφαση στη δημιουργία βιώσιμων πλεονεκτημάτων στην αγορά. Οι επιχειρήσεις αναζητούν τρόπους για να αυξήσουν τα έσοδα, να μειώσουν το κόστος, να αυξήσουν την αποδοτικότητα και να καινοτομήσουν. Τα μοντέλα αυτά όλο και περισσότερο εξαρτώνται από τις τεχνολογίες πληροφορικής. Οι τεχνολογίες αυτές, παίζουν ένα αυξανόμενο ρόλο στον καθορισμό της επιτυχίας ή της αποτυχίας των εν λόγω επιχειρήσεων. Καθώς η δικτύωση έχει αναδιαμορφώσει την αγορά, όπου η γεωγραφία και οι χρονικές ζώνες έχουν χάσει τη σημασία τους, και η online πρόσβαση σε αγορές, πελάτες, και προμηθευτές έχουν γίνει απαραίτητες, ο ρόλος των συστημάτων πληροφορικής έχει αλλάξει κι από υποστηρικτικό ρόλο έχει γίνει ένας από τους βασικούς παράγοντες επιτυχίας.

Η χρήση υπηρεσιών νέφους από τις επιχειρήσεις αποτελεί πλέον κοινό τόπο. Οι πάροχοι εφαρμογών (ενδιάμεσοι) στοχεύουν στη μεγιστοποίηση της πελατειακής τους βάσης εξετάζοντας παράλληλα το σχετικό κόστος. Προς αυτή την κατεύθυνση, απαιτούνται επιχειρηματικά δυναμικά μοντέλα τα οποία είναι αποδοτικά για την επιχείρηση αναφορικά τόσο με τους τρέχοντες χρήστες όσο και με τους προσδοκώμενους συγκριτικά με τις ανάγκες ελαστικότητας. Τα μοντέλα αυτά στόχο έχουν να προτείνουν

τρόπους για την προσέλκυση πελατών συνυπολογίζοντας το σχετικό κόστος, βελτιστοποιώντας ταυτόχρονα την απόδοση της "επένδυσης" σε πόρους συγκριτικά με τον προσδοκώμενο αριθμό πελατών.

Βάσει των παραπάνω προτείνονται τρία επιχειρηματικά μοντέλα: (i) αμυντικό, (ii) επιθετικό και (iii) ουδέτερο, τα είναι άρρηκτα συνδεμένα με τα ελαστικά μοντέλα που παρέχονται από το νέφος και στοχεύουν μέσω διαφορετικών υποσχέσεων ποιότητας υπηρεσίας και τιμολογιακών πολιτικών στη προσέλκυση πελατών βελτιστοποιώντας παράλληλα το κόστος.

Στόχοι του κεφαλαίου αυτού είναι:

1. **Πρόταση Επιχειρηματικών Μοντέλων:** Η πρόταση τριών δυναμικών επιχειρηματικών μοντέλων.
2. **Περιγραφή:** Αναλυτική περιγραφή των μοντέλων που προτάθηκαν.

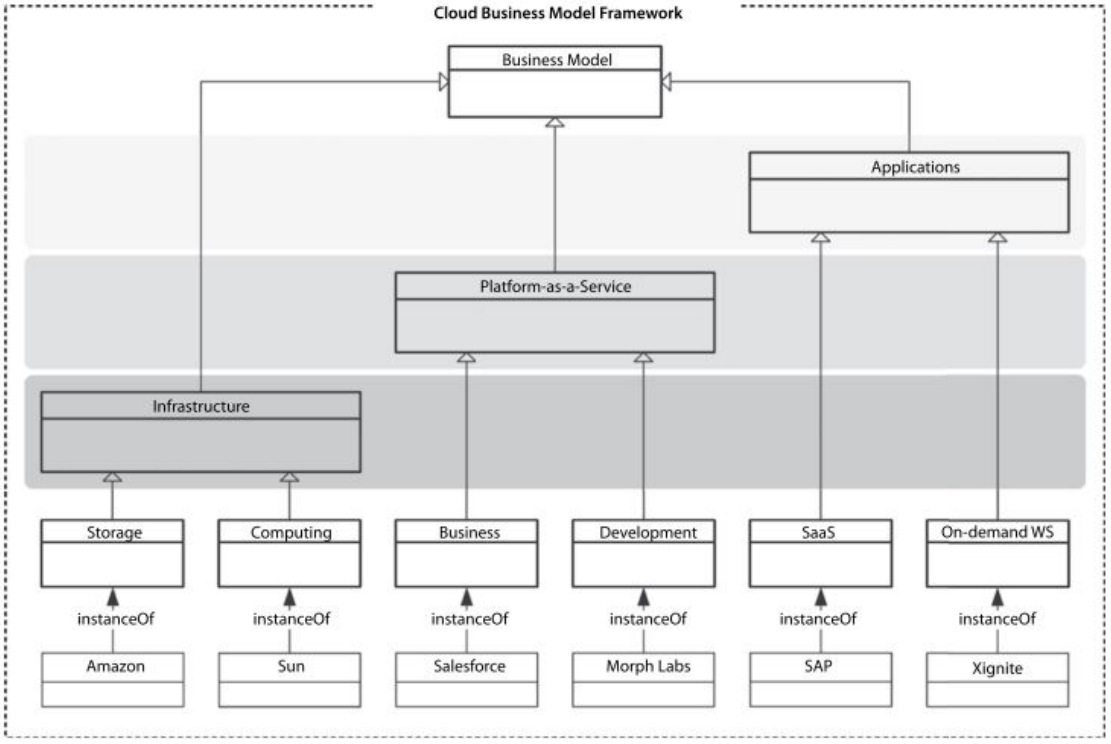
5.2 Κατηγορίες Επιχειρηματικών Μοντέλων

Ένα επιχειρηματικό μοντέλο ορίζεται [71] ως:

Ορισμός 5.1 (Επιχειρηματικό Μοντέλο) :
"Το σχέδιο που υλοποιείται από μια εταιρεία με σκοπό να δημιουργήσει έσοδα και να αποκτήσει ένα κέρδος από εργασίες. Το μοντέλο περιλαμβάνει τα συστατικά και τις λειτουργίες της επιχείρησης, καθώς και τα έσοδα που αποφέρει και τα έξοδα που περιλαμβάνει"

Κάθε πάροχος μιας εφαρμογής μπορεί να χρεώσει την υπηρεσία του βάσει διαφορετικών επιχειρηματικών μοντέλων. Το κάθε μοντέλο εγγυάται διαφορετικό ελάχιστο και μέγιστο χρόνο απόκρισης της υπηρεσίας (Response Time) και κατά συνέπεια στοχεύει σε διαφορετικού μεγέθους πελατειακή βάση, διαφορετικά έξοδα με σκοπό την εκπλήρωση των εγγυήσεων του και διαφορετικό αναμενόμενο κέρδος. Τα μοντέλα που θα χρησιμοποιήσει ο πάροχος της εφαρμογής αλληλεπιδρούν δυναμικά με τα μοντέλα που χρησιμοποιεί ο πάροχος της υποδομής, μια κατάταξη των οποίων απεικονίζεται στο Σχήμα 5.1 [72], καθώς το κόστος απόκτησης πόρων επηρεάζει άμεσα το κόστος παροχής της υπηρεσίας στους τελικούς πελάτες.

Στα πλαίσια της παρούσας ερευνητικής εργασίας, τα επιχειρηματικά μοντέλα που προτείνονται δεν λαμβάνουν υπόψη παραμέτρους όπως εργατοώρες προσωπικού, κόστη αναλώσιμων κτλ καθώς είναι εκτός πλαισίου του αντικείμενου. Οι παράμετροι που περιλαμβάνονται χωρίζονται σε τρείς βασικές κατηγορίες και παρουσιάζονται στο

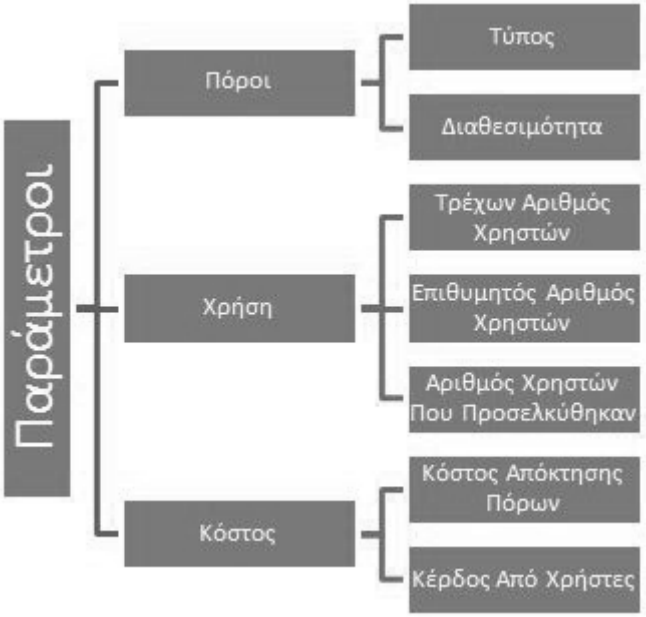


Σχήμα 5.1: Κατάταξη επιχειρηματικών Μοντέλων [72]

Σχήμα 5.2. Αναλυτικά οι κατηγορίες βάσει των οποίων ταξινομούνται οι παράμετροι είναι:

1. Πόροι από υποδομή υπολογιστικού νέφους και περιλαμβάνουν:
   - Τύπο πόρων (CPU, Memory, κτλ)
   - Διαθεσιμότητα πόρων
2. Χρήση προσφερόμενης επιχειρηματικής λύσης που χωρίζεται:
   - Αριθμό χρηστών που χρησιμοποιούν την επιχειρηματική λύση
   - Αριθμός χρηστών που στοχεύει το μοντέλο
   - Αριθμός χρηστών που τελικώς προσελκύστηκαν
3. Κόστος
   - Το κόστος για την απόκτηση των πόρων (πρώτη παράμετρος του μοντέλου)
   - Το κέρδος από του χρήστες που έκαναν χρήση της εφαρμογής (δεύτερη παράμετρος του μοντέλου)

Η σχέση των παραμέτρων αυτών είναι άμεσα και δυναμικά συνδεδεμένη. Αύξηση των χρηστών σημαίνει αύξηση των εσόδων, συνεπάγεται όμως και με αύξηση του



Σχήμα 5.2: Παράμετροι Επιχειρηματικών Μοντέλων

χρόνου απόκρισης της εφαρμογής. Προκειμένου να διατηρήσει τη δέσμευση του εκάστοτε επιχειρηματικού μοντέλου, ο πάροχος της εφαρμογής θα πρέπει να προβεί σε απόκτηση νέων πόρων. Αντίστοιχα μείωση των χρηστών σημαίνει μικρότερος χρόνος απόκρισης οπότε μειώνεται η ανάγκη χρήσης πόρων.

Βάσει των παραπάνω, τα επιχειρηματικά μοντέλα που προτείνονται, ακολουθώντας τη λογική των ελαστικών μηχανισμών αντίδρασης, χρησιμοποιούν τα στοιχεία που φαίνονται στο πίνακα 5.1.

Πίνακας 5.1: Στοιχεία Επιχειρηματικών Μοντέλων

Goal Users	Ο αριθμός της πελατειακής βάσης που στοχεύει το μοντέλο
RT	Χρόνος Απόκρισης
t^L	Ελάχιστο Response Time που μπορεί να εγγυηθεί το μοντέλο
t^U	Μέγιστο αποδεχτό Response Time που εγγυάται το μοντέλο
t^{SU}	Χρόνος Εκκίνησης πόρων
initDep	Αρχικοί πόροι που έχουν αποκτηθεί για την εκκίνηση της εφαρμογής
res	Πόροι
a, b, c	Συντελεστές βαρύτητας για την αύξηση των πόρων που ορίζονται για κάθε μοντέλο ξεχωριστά

Κάθε μοντέλο βασισμένο στις πολιτικές απόκτησης πόρων που διαθέτει (οι οποίες εκφράζονται από τις τιμές των συντελεστών βαρύτητας) αλλά και την αρχική υποδομή που έχει ενοικιάσει ο πάροχος της υπηρεσίας, μπορεί να εγγυηθεί έναν ελάχιστο χρόνο απόκρισης . Επιπλέον, μπορεί να εγγυηθεί κι ένα μέγιστο χρόνο απόκρισης. Σε περίπτωση που ο μέγιστος χρόνος ξεπεραστεί, θεωρείται πως η υπηρεσία παρέχει κακή ποιότητα υπηρεσιών. Προκειμένου να εκφραστούν οι χρόνοι αυτοί σε σχέση

με τον αριθμό των χρηστών, στη παρούσα ερευνητική εργασία ορίζονται οι μεταβλητές (t^L) και (t^U) που εκφράζουν τον ελάχιστο και μέγιστο χρόνο απόκρισης αντίστοιχα. Όπως αναφέρθηκε στην ενότητα 4.8, όπου συζητήθηκαν ανοικτά θέματα ελάστικοτητας, απαιτείται ένα χρονικό διάστημα από την αίτηση για νέους πόρους μέχρι τη στιγμή απόκτησης των πόρων το οποίο ονομάζεται spin-up time (t^{SU}). Τα μοντέλα που προτείνονται, πρέπει να λαμβάνουν το χρόνο αυτό υπ' όψιν κατά την αίτηση νέων πόρων ώστε να μη ξεπεράσουν το μέγιστο χρόνο απόκρισης κατά τη διάρκεια του spin-up time. Όπως τα επιχειρηματικά μοντέλα πρέπει φροντίζουν να μένουν μέσα στα όρια του ελαχίστου και μεγίστου χρόνου απόκρισης, έτσι πρέπει και να φροντίζουν να μην μένουν πόροι σε αδράνεια και κατά συνέπεια να μη σπαταλούνται χρήματα για πόρους που δεν επιφέρουν έσοδα. Για το σκοπό αυτό στα προτεινόμενα μοντέλα ορίζεται η αρχική υποδομή που ενοικιάζει ο πάροχος της εφαρμογής (initDep) ώστε σε περίπτωση που ο χρόνος απόκρισης πέσει κάτω του ελαχίστου, οι επιπλέον δεσμεμεύνοι πόροι να απελευθερώνονται.

5.2.1 Ουδέτερο Μοντέλο

Σκοπός του μοντέλου αυτού είναι οι χρήστες, ακόμα κι εάν ο αριθμός τους αυξηθεί ραγδαία (bursting), να μην πλησιάζουν το μέγιστο χρόνο απόκρισης. Για να επιτευχθεί αυτό, ελέγχεται συνεχώς η διαφορά του τρέχοντος χρόνου απόκρισης και του ελαχίστου χρόνου που υπόσχεται το μοντέλο. Εάν η διαφορά ξεπεράσει ένα συγκεκριμένο όριο (π.χ. τα 300 millisecond) τότε οι πόροι αυξάνονται. Η περίπτωση της ραγδαίας αύξησης των χρηστών καλύπτεται καθώς τρέχων χρόνος απόκρισης συγκρίνεται με τη διαφορά του μέγιστου χρόνου και του χρόνου εκκίνησης νέων πόρων και σε περίπτωση που είναι μεγαλύτερος της διαφοράς αυτής οι πόροι αυξάνονται. Για να αποφευχθεί το φαινόμενο αδρανών πόρων (και κατά συνέπεια της κακής χρηματικής επένδυσης) ο τρέχων χρόνος απόκρισης συγκρίνεται με τον ελάχιστο χρόνο που υπόσχεται το επιχειρηματικό μοντέλο κι εάν είναι μικρότερος τότε όλοι οι επιπλέον πόροι που έχουν αποκτηθεί σταματούν να χρησιμοποιούνται και ο πάροχος επιστρέφει στην αρχική υποδομή. Τέλος το μοντέλο επιτρέπει την αύξηση πόρων, σε περίπτωση που ο αριθμός χρηστών φτάσει ή ξεπεράσει το μέγιστο αριθμό χρηστών που αρχικά είχε προβλεφθεί.

Το παραπάνω μοντελοποιείται μαθηματικά στην (5.1) και παρουσιάζεται σε μορφή ψευδοκώδικα στον Αλγόριθμο 5.1 στη συνέχεια. Το Σχήμα 5.3 δείχνει τη ροή του μοντέλου αυτού.

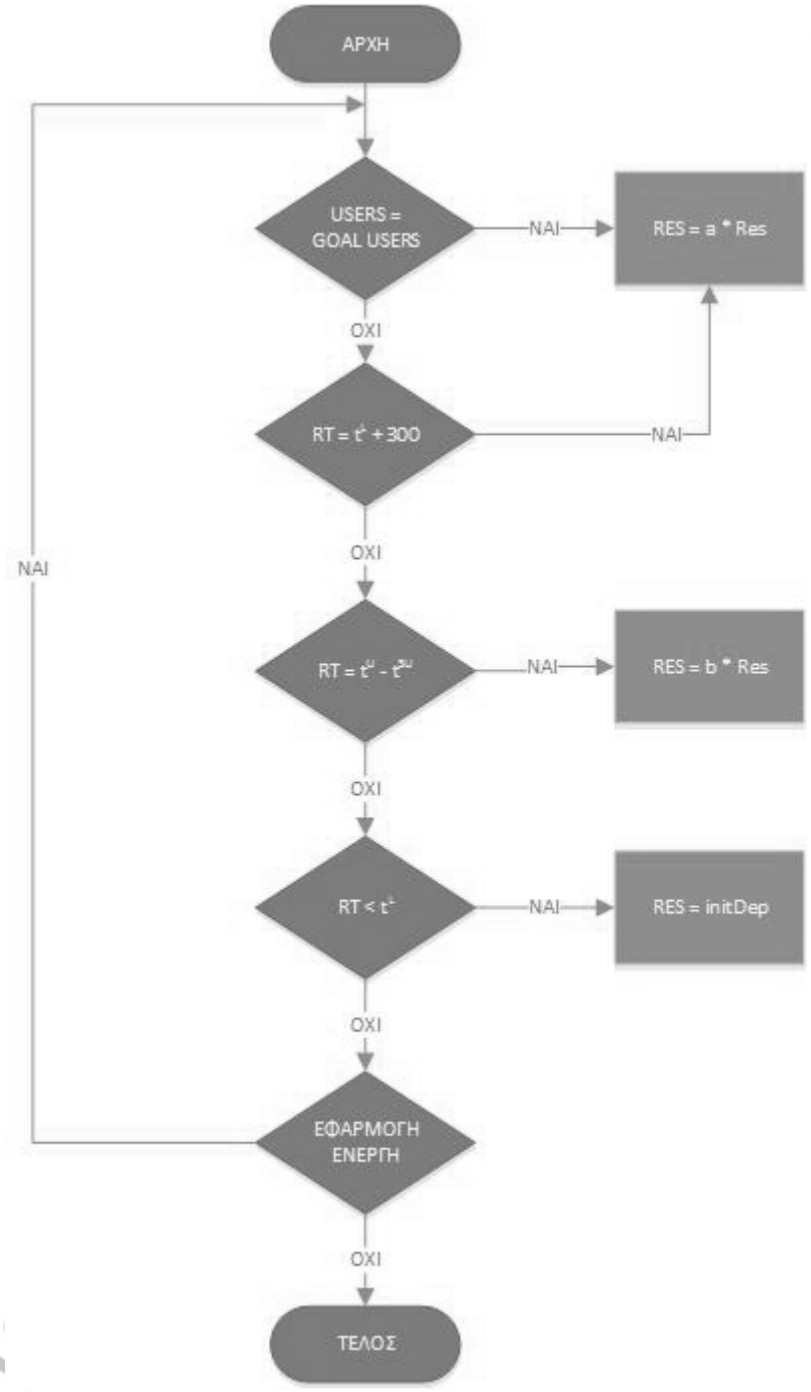
$$F(x) = \begin{cases} a * res & \text{εάν } RT > t^L + 300, \\ a * res & \text{εάν } Users = GoalUsers, \\ b * res & \text{εάν } RT >= t^U - t^S U, \\ x = initDep & \text{εάν } RT < t^L \end{cases} \tag{5.1}$$

Αλγόριθμος 5.1: Ουδέτερο Μοντέλο

Data: GoalUsers, a, b, initDep, t^U, t^L, t^{SU}

Result: Αλλαγή Πόρων

```
1 while (appIsActive = true) do
2     if (users == GoalUsers || RT > tL + 300) then
3         res = a * res ;
4     else if (RT >= tU – tSU ) then
5         res = b * res;
6     else if (RT < tL ) then
7         res = initDep;
8     end if
9 end while
```



Σχήμα 5.3: Διάγραμμα Ροής Ουδέτερου Μοντέλου

5.2.2 Αμυντικό Μοντέλο

Το μοντέλο αυτό φροντίζει οι χρήστες να μην ξεπερνούν το μέγιστο χρόνο απόκρισης αλλά επιτρέπει να τον προσεγγίζουν. Για να επιτευχθεί αυτό, ελέγχεται συνεχώς η διαφορά του τρέχοντος χρόνου απόκρισης και του ελαχίστου χρόνου που υπόσχεται το

μοντέλο. Εάν η διαφορά ξεπεράσει ένα συγκεκριμένο όριο (το οποίο είναι αρκετά μεγαλύτερο συγκριτικά με το ουδέτερο μοντέλο, π.χ. τα 500 millisecond) τότε οι πόροι αυξάνονται. Η περίπτωση της ραγδαίας αύξησης των χρηστών καλύπτεται καθώς ο τρέχων χρόνος απόκρισης συγκρίνεται με το μέγιστο χρόνο απόκρισης. Εάν οι χρόνοι αυτοί ταυτίζονται τότε οι πόροι αυξάνονται. Στο μοντέλο αυτό δεν λαμβάνεται υπόψη ο χρόνος εκκίνησης νέων πόρων. Και στο μοντέλο αυτό, για να αποφευχθεί το φαινόμενο αδρανών πόρων (και κατά συνέπεια κακής χρηματικής επένδυσης) ο τρέχων χρόνος αναμονής συγκρίνεται με τον ελάχιστο χρόνο που υπόσχεται το επιχειρηματικό μοντέλο κι εάν είναι μικρότερος τότε όλοι οι επιπλέον πόροι που έχουν αποκτηθεί σταματούν να χρησιμοποιούνται και ο πάροχος επιστρέφει στην αρχική υποδομή.

Το παραπάνω μοντελοποιείται μαθηματικά στην (5.2) και παρουσιάζεται σε μορφή ψευδοκώδικα στον Αλγόριθμο 5.2 στη συνέχεια. Το Σχήμα 5.4 δείχνει τη ροή των συνθηκών αυτών.

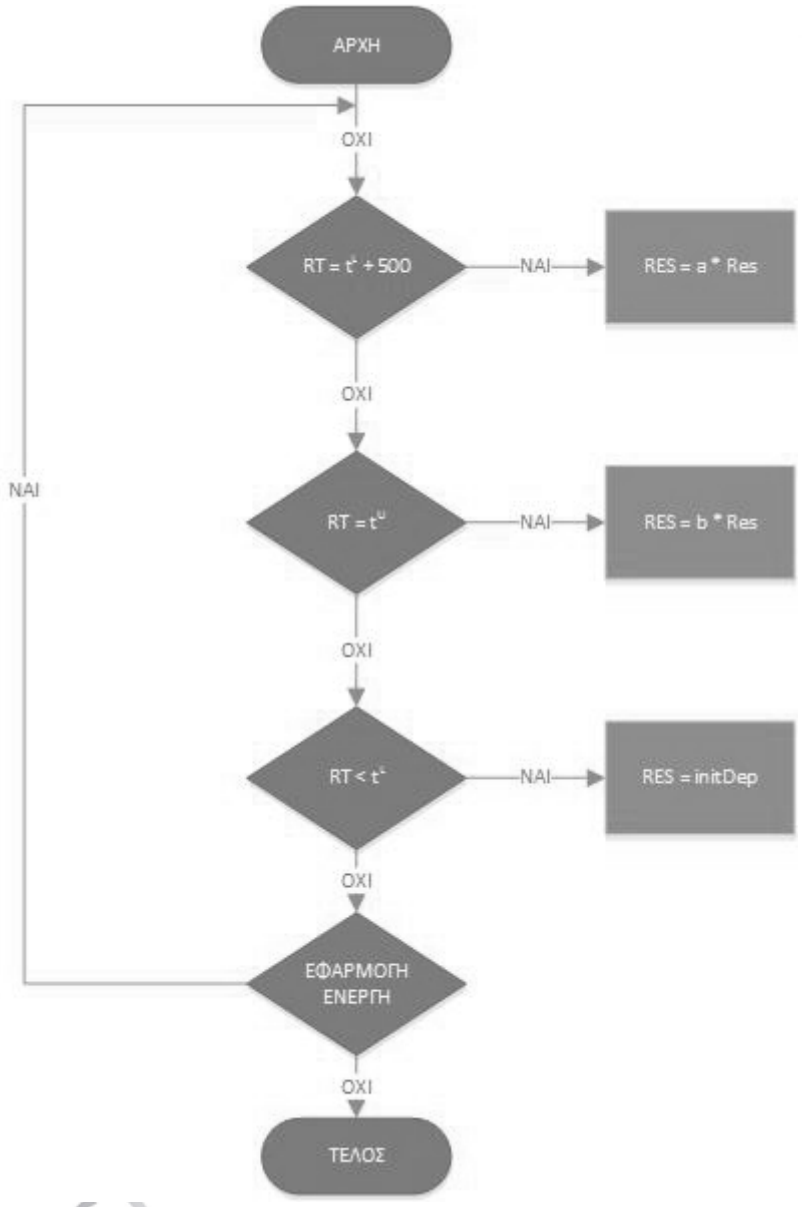
$$F(x) = \begin{cases} a * res & \text{εάν } RT > t^L + 500, \\ b * res & \text{εάν } RT = t^U, \\ x = initDep & \text{εάν } RT < t^L \end{cases} \tag{5.2}$$

Αλγόριθμος 5.2: Αμυντικό Μοντέλο

Data: a, b, initDep, t^U, t^L, t^{SU}

Result: Αλλαγή Πόρων

```
1 while (appIsActive = true) do
2     if (users == GoalUsers || RT > t^L + 500) then
3         res = a * res ;
4     else if (RT == t^U ) then
5         res = b * res;
6     else if (RT < t^L ) then
7         res = initDep;
8     end if
9 end while
```



Σχήμα 5.4: Διάγραμμα Ροής Αμυντικού Μοντέλου

5.3 Επιθετικό Μοντέλο

Το μοντέλο αυτό έχει ως στόχο οι χρήστες να βρίσκονται όσο το δυνατόν πιο κοντά στον ελάχιστο χρόνο απόκρισης γίνεται. Για να επιτευχθεί αυτό, ελέγχεται συνεχώς η διαφορά του τρέχοντος χρόνου απόκρισης και του ελαχίστου χρόνου που υπόσχεται το μοντέλο. Εάν η διαφορά ξεπεράσει ένα συγκεκριμένο όριο (το οποίο είναι αρκετά μικρότερο συγκριτικά με το ουδέτερο μοντέλο, π.χ. τα 100 millisecond) τότε οι πόροι αυξάνονται. Η περίπτωση της ραγδαίας αύξησης των χρηστών καλύπτεται καθώς ο

τρέχων χρόνος απόκρισης συγκρίνεται με τη διαφορά του μέγιστου χρόνου απόκρισης και του αθροίσματος του μέγιστου χρόνου εκκίνησης νέων πόρων και ενός χρόνου "ασφαλείας" (π.χ. 10 millisecond). Εάν οι χρόνοι αυτοί ταυτίζονται τότε οι πόροι αυξάνονται. Και στο μοντέλο αυτό, για να αποφευχθεί το φαινόμενο αδρανών πόρων (και κατά συνέπεια κακής χρηματικής επένδυσης) ο τρέχων χρόνος αναμονής συγκρίνεται με τον ελάχιστο χρόνο που υπόσχεται το επιχειρηματικό μοντέλο κι εάν είναι μικρότερος τότε όλοι οι επιπλέον πόροι που έχουν αποκτηθεί σταματούν να χρησιμοποιούνται και ο πάροχος επιστρέφει στην αρχική υποδομή. Τέλος το μοντέλο επιτρέπει την αύξηση πόρων, σε περίπτωση που ο αριθμός χρηστών φτάσει ή ξεπεράσει το μέγιστο αριθμό χρηστών που αρχικά είχε προβλεφθεί.

Το παραπάνω μοντελοποιείται μαθηματικά στην (5.3) και παρουσιάζεται σε μορφή ψευδοκώδικα στον Αλγόριθμο 5.3 στη συνέχεια. Το Σχήμα 5.5 δείχνει τη ροή των συνθηκών αυτών.

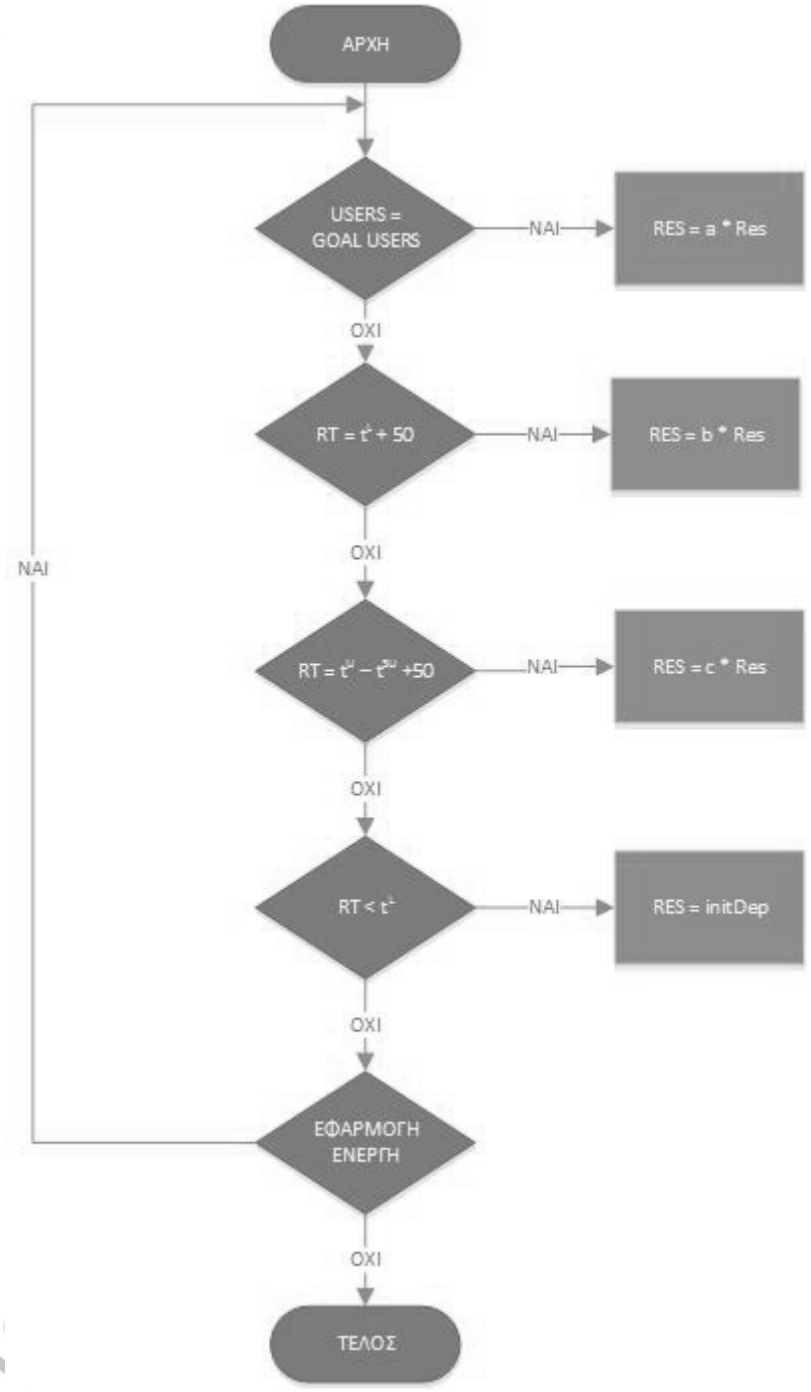
$$F(x) = \begin{cases} a * res & \text{εάν } users = GoalUsers, \\ b * res & \text{εάν } RT >= t^L + 50, \\ c * res & \text{εάν } RT >= t^U - t^SU + 50, \\ x = initDep & \text{εάν } RT < t^L \end{cases} \tag{5.3}$$

Αλγόριθμος 5.3: Επιθετικό Μοντέλο

Data: GoalUsers, a, b, c, initDep, t^U, t^L, t^{SU}

Result: Αλλαγή Πόρων

```
1  while (appIsActive = true) do
2      if (users == GoalUsers then
3          res = a * res ;
4      else if (RT >= t^L + 100 ) then
5          res = b * res;
6      else if (RT >= t^U - t^SU + 10) then
7          res = c*res;
8      else if (RT < t^L ) then
9          res = initDep;
10     end if
11 end while
```



Σχήμα 5.5: Διάγραμμα Ροής Επιθετικού Μοντέλου

ΚΕΦΑΛΑΙΟ 6

Μηχανισμός Βέλτιστης Διαχείρισης Υπηρεσιών

"The essence of mathematics is not to make simple things complicated,
but to make complicated things simple."
˜Stan Gudder

6.1 Εισαγωγή

Στα προηγούμενα κεφάλαια παρουσιάστηκαν μηχανισμοί ελαστικότητας και προτάθηκαν επιχειρηματικά μοντέλα που προτείνουν τρόπους για την προσέλκυση πελατών λαμβάνοντας υπόψη περιορισμούς κόστους. Στο παρών κεφάλαιο παρουσιάζεται ο τρόπος ανάπτυξης και ο ίδιος ο μηχανισμός βέλτιστης διαχείρισης υπηρεσιών, που προτείνεται στην ερευνητική αυτή εργασία και καλύπτει την ανάγκη αντιστοίχισης (η οποία επιτρέπει την επιλογή μοντέλων ελαστικότητας που θα ακολουθηθούν σε σχέση με το επιχειρηματικό μοντέλο) δυναμικών επιχειρηματικών μοντέλων σε προβλεπόμενους πόρους βάσει αναγκών.

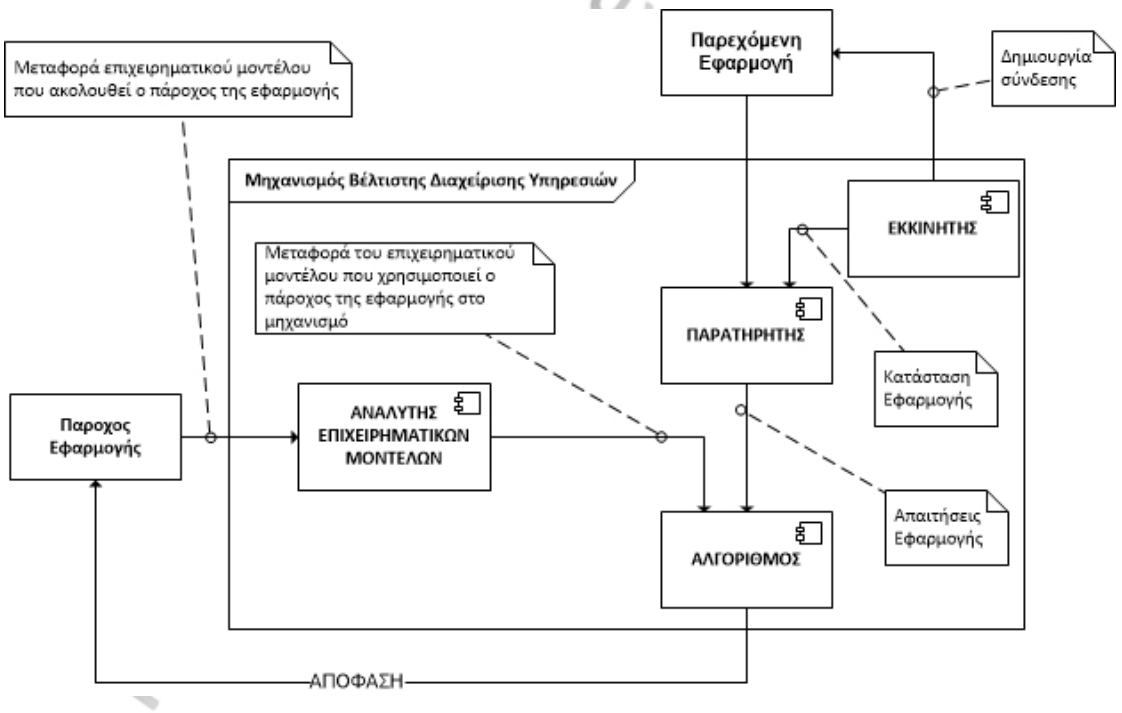
Το κεφάλαιο αυτό στοχεύει:

1. **Δομή Μηχανισμού:** Προσδιορισμός του τρόπου λειτουργίας και της δομής του μηχανιμού βέλτιστης διαχείρισης υπηρεσιών.
2. **Διεξαγωγή Πειράματος:** Περιγραφική ανασκόπηση της υποδομής του πειράματος καθώς και η αναλυτική περιγραφή των βημάτων και των ρυθμίσεων που απαιτούνται για την εκτέλεση του.
3. **Μοντελοποίηση Δεδομένων:** Η περιγραφή της μεθόδου μοντελοποίησης των δεδομένων για την παραγωγή της συνάρτησης καθώς και η ανάλυση κι επεξήγηση της.

6.2 Αρχιτεκτονική Δομή Μηχανισμού

Στο Κεφάλαιο 1 τέθηκε το ερευνητικό πρόβλημα και έγινε ξεκάθαρη η ανάγκη για την ύπαρξη ενός μηχανισμού διαχείρισης επιχειρηματικών διεργασιών ο οποίος μπορεί να αντιστοιχίζει δυναμικά επιχειρηματικά μοντέλα και απαιτήσεις σε πόρους. Στην ενότητα αυτή παρουσιάζεται και αναλύεται η αρχιτεκτονική δομή του μηχανισμού που προτείνεται στη παρούσα ερευνητική εργασία.

Πρωταρχικός στόχος είναι ο μηχανισμός να είναι ανεξάρτητος της διαδικασίας εκκίνησης της εφαρμογής έτσι ώστε να είναι δυνατόν να μπορεί να αρχικοποιηθεί και ο μηχανισμός να προστεθεί αργότερα εάν χρειαστεί. Ο επόμενος στόχος είναι ο μηχανισμός να μπορεί να χρησιμοποιεί πληθώρα επιχειρηματικών μοντέλων δυναμικά. Τέλος σημαντικό στοιχείο είναι ο μηχανισμός να παρακολουθεί συνεχώς την εφαρμογή και να μαθαίνει την κατάσταση της καθώς και τις απαιτήσεις της. Στο Σχήμα 6.1 παρουσιάζεται ο τρόπος λειτουργίας του μηχανισμού.



Σχήμα 6.1: Αρχιτεκτονική Δομή Μηχανισμού

Προκειμένου να επιτευχθούν οι παραπάνω στόχοι ο μηχανισμός που προτείνεται αποτελείται από τέσσερα, διασυνδεδεμένα μεταξύ τους, βασικά δομικά συστατικά:

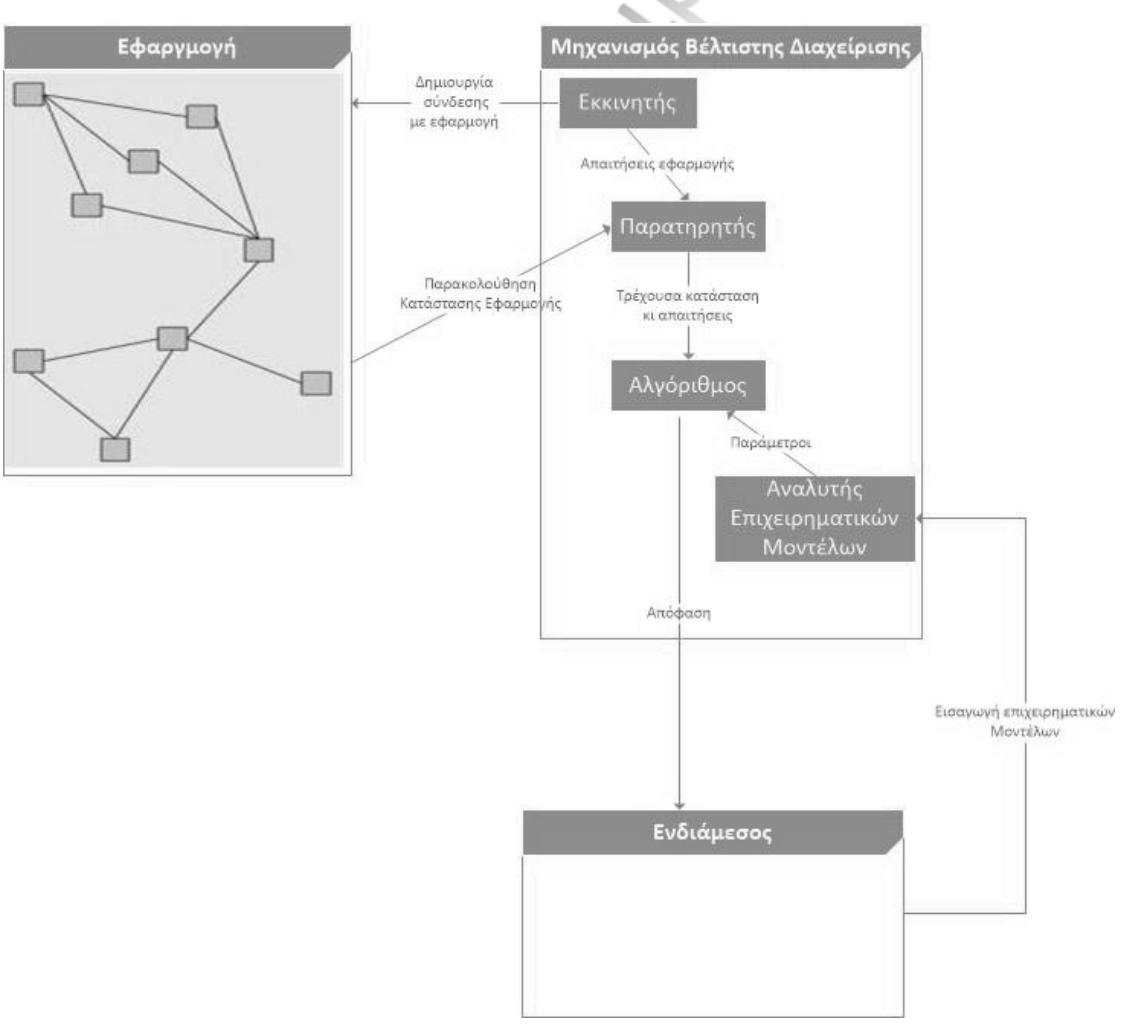
1. **Εκκινητής (starter):** Δημιουργεί τη σύνδεση με την εφαρμογή και μαθαίνει τη δομή της (συστατικά, διεπαφές, εσωτερικές επικοινωνίες κτλ). Με το πέρας της μεταφοράς των πληροφοριών αυτών στο μηχανισμό παρακολούθησης, μπαίνει σε κατάσταση αδράνειας.

2. **Αναλυτής Επιχειρηματικών Μοντέλων (Business Model Analyzer):** Αποτελεί το δομικό συστατικό που έχει την ευθύνη να γνωρίζει τα επιχειρηματικά μοντέλα τα οποία προτείνει ο πάροχος της εφαρμογής (ενδιάμεσος) και να τα μεταφέρει στον αλγόριθμο που παράγει την τελική απόφαση.

3. **Αλγόριθμος:** Παράγει την τελική απόφαση κοστολόγησης έχοντας ως παραμέτρους τόσο τις προβλεπόμενες ανάγκες ελαστικότητας όσο και το επιχειρηματικό μοντέλο που χρησιμοποιεί ο πάροχος της εφαρμογής.

4. **Παρατηρητής (monitoring):** Σκοπός του είναι η συνεχής παρακολούθηση της κατάστασης της εφαρμογής και των απαιτήσεων της σε πόρους με σκοπό τη μεταφορά των πληροφοριών αυτών στον αλγόριθμο.

Τα παραπάνω συστατικά και ο τρόπος αλληλεπίδρασης τους με την εφαρμογή καθώς και τον πάροχο της εφαρμογής παρουσιάζονται στο Σχήμα 6.2.



Σχήμα 6.2: Δομικά στοιχεία και αλληλεπίδραση μηχανισμού

6.3 Συλλογή Δεδομένων

Για τη συλλογή των απαιτούμενων δεδομένων δημιουργήθηκε μια απλή εφαρμογή για τον Apache Tomcat. Η εφαρμογή αυτή όταν λαμβάνει ένα HTTP GET αίτημα παράγει ένα εκατομμύριο ψευδοτυχαίους αριθμούς με εύρος από το μηδέν μέχρι το χίλια. Η εφαρμογή είναι προφανώς άχρηστη για οποιοδήποτε πραγματικό επιχειρηματικό σκοπό αλλά αντιπροσωπεύει άριστα τη χρήση επεξεργαστικής ισχύς σε ένα ιδιαίτερα παραλληλισμό υπολογιστικό καθήκον.

Για το πείραμα χρησιμοποιήθηκαν δυο υπολογιστές συνδεδεμένοι στο ίδιο τοπικό δίκτυο. Ο Apache Tomcat [1] , στον οποίο έχει "σηκωθεί" η εφαρμογή, τρέχει σε λειτουργικό Ubuntu Server 12.04.3. Για τη δημιουργία κίνησης χρησιμοποιήθηκε το Jmeter [2] που τρέχει σε λειτουργικό σύστημα windows 8.1 ενώ για τη καταγραφή των αποτελεσμάτων χρησιμοποιείται το PerfMon [3] Metrics Tools. Συνοπτικά η όλη υποδομή του πειράματος παρουσιάζεται στον πίνακα 6.1.

Πίνακας 6.1: Δομικά στοιχεία πειράματος

Α/Α	Δομικό Στοιχείο	Έκδοση	Σκοπός
1	Apache JMeter	2.10 r1533061	Δημιουργία εικονικών χρηστών και κίνησης
2	Apache Tomcat	7.0.47	Ο εξυπηρετητής στον οποίο τρέχει η εφαρμογή.
3	PerfMon Metrics	1.1.3	Καταγραφή αποτελεσμάτων
4	PerfMon ServerAgent [4]	2.2.1	Αποστολή αποτελεσμάτων στο PerfMon Metrics
5	Ubuntu Server	12.04.3 LTS	Το λειτουργικό του μηχανήματος που έχει στηθεί ο Tomcat και ο ServerAgent
6	Windows	8.1	Το λειτουργικό του μηχανήματος στο οποίο τρέχει το Jmeter

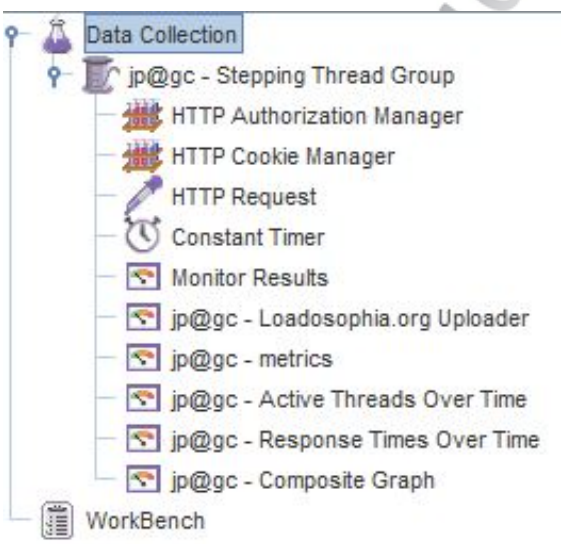
Έχοντας στήσει την υποδομή που απαιτείται, προκειμένου να γίνει η συλλογή των δεδομένων πρέπει να δημιουργηθεί κίνηση στον εξυπηρετητή. Για να επιτευχθεί ο στόχος αυτός δημιουργούνται εικονικοί χρήστες, μέσω του JMeter, οι οποίοι μέσω ενός

[1] http://tomcat.apache.org/
[2] https://jmeter.apache.org/
[3] http://jmeter-plugins.org/wiki/PerfMon/
[4] http://jmeter-plugins.org/wiki/PerfMonAgent/

HTTP GET μηνύματος αιτούνται να κάνουν χρήστη της εφαρμογής παραγωγής ψευδοτυχαίων αριθμών.

Έπειτα από δοκιμές, έγινε η διαπίστωση πώς ο μέγιστος αριθμός εικονικών χρηστών που μπορεί να δημιουργηθεί, με τη παρούσα υποδομή, είναι αυτός των διακοσίων. Αριθμός μεγαλύτερος του ορίου αυτού, δεν μπορεί να δημιουργηθεί καθώς δεν επαρκεί η υπολογιστική ισχύς του μηχανήματος που φιλοξενεί το JMeter με αποτέλεσμα δημιουργείται θόρυβος και να αλλοιώνονται τα αποτελέσματα του πειράματος.

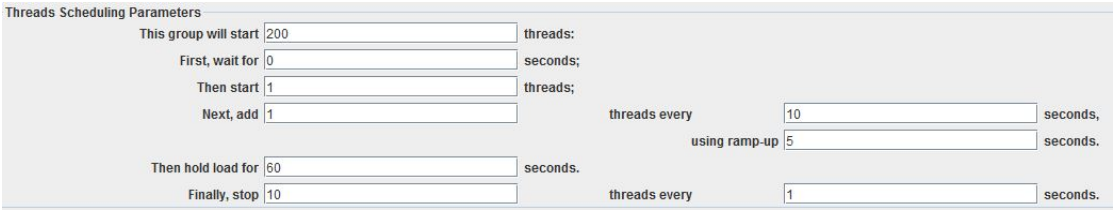
Στο Σχήμα 6.3 γίνεται μια ανασκόπηση των ρυθμίσεων και των παραμετροποιήσεων που έγιναν στο JMeter για τη διεξαγωγή του πειράματος. Στη συνέχεια αναλύονται η κάθε μια ξεχωριστά.



Σχήμα 6.3: Συνολικές ρυθμίσεις στο JMeter

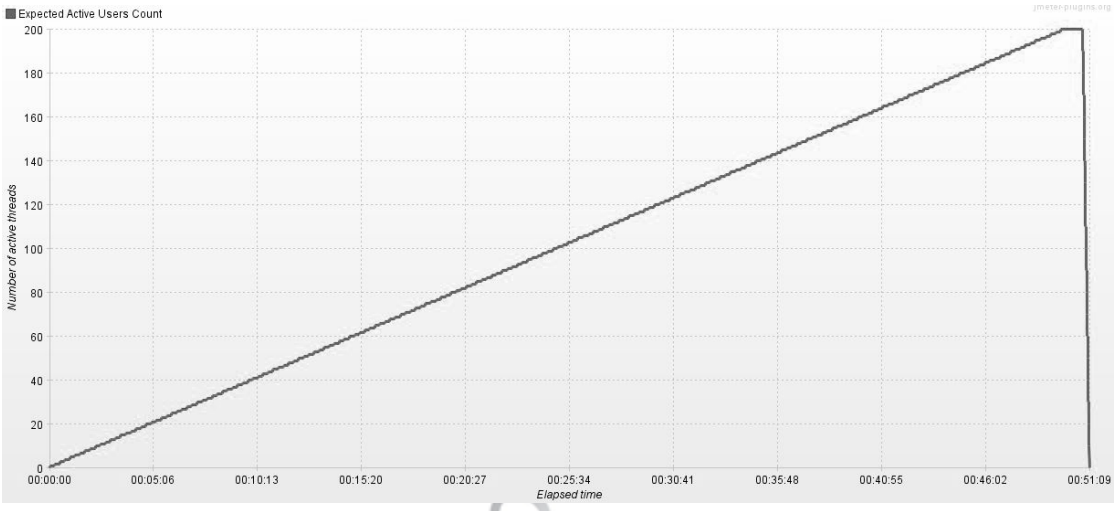
Ξεκινώντας ένα νέο τεστ στο JMeter αρχικά πρέπει να του δοθεί μια ονομασία. Για την παρούσα εργασία επιλέχθηκε το “Data Collection”. Εν συνεχεία στο πείραμα προστίθεται ένα Thread Group στο οποίο δηλώνεται ο αριθμός των εικονικών χρηστών που θα δημιουργηθούν. Για το πείραμα επιλέχθηκε το “jp@gc - Stepping Thread Group” το οποίο επιτρέπει καλύτερη διαχείριση των εικονικών χρηστών.

Για το JMeter ένα thread αντιστοιχίζεται σε ένα χρήστη. Στο Σχήμα 6.4 φαίνονται οι ρυθμίσεις που πρέπει να γίνουν για τη δημιουργία των εικονικών χρηστών.



Σχήμα 6.4: Δημιουργία εικονικών χρηστών (με βήμα 1)

Η περίοδος “ramp-up” λέει στο JMeter πόσος χρόνος θα χρειαστεί για να δημιουργήσει τη πλήρη κίνηση. Για παράδειγμα εάν επιλεχθούν 10 εικονικοί χρήστες και η περίοδος “ramp-up” είναι 100 δευτερόλεπτα, τότε το JMeter θα χρειαστεί 100 δευτερόλεπτα για τη δημιουργία και των 10 εικονικών χρηστών. Κάθε εικονικός χρήστης θα ξεκινά 10 δευτερόλεπτα (100/10) από τη χρονική στιγμή που ξεκίνησε το προηγούμενο. Στο Σχήμα 6.5 φαίνεται η αύξηση των εικονικών χρηστών με τη πάροδο του χρόνου βάσει των ρυθμίσεων που παρουσιάστηκαν στο Σχήμα 6.4.

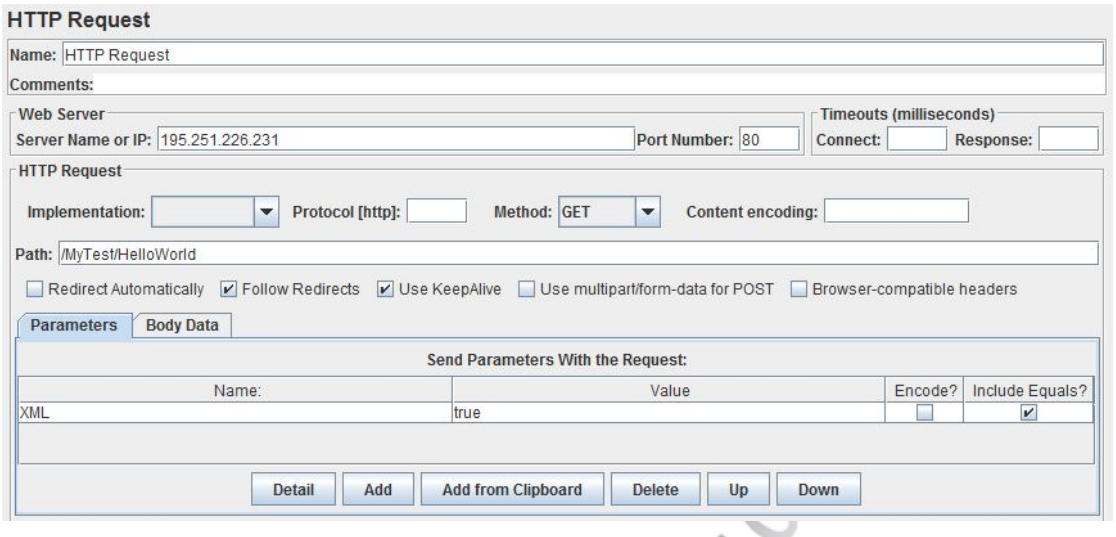


Σχήμα 6.5: Δημιουργία εικονικών χρηστών με τη πάροδο του χρόνου

Ώστε να δημιουργηθεί πραγματική κίνηση στον εξυπηρετητή κάθε εικονικός χρήστης πρέπει να κάνει χρήση της εφαρμογής που παράγει ψευδοτυχαίους αριθμούς. Για να επιτευχθεί αυτό δημιουργείται στο JMeter το HTTP αίτημα που θα κάνει ο κάθε εικονικός χρήστης. Βασικές παράμετροι που πρέπει να δηλωθούν είναι:

1. Η IP διεύθυνση του εξυπηρετητή (“Server Name or IP”) που για το πείραμα είναι η 195.251.226.231,
2. η μέθοδος (GET) και
3. η διαδρομή (path) για την εφαρμογή. Στα πλαίσια του πειράματος η διαδρομή αυτή είναι η: 195.251.226.231/MyTest/HelloWorld

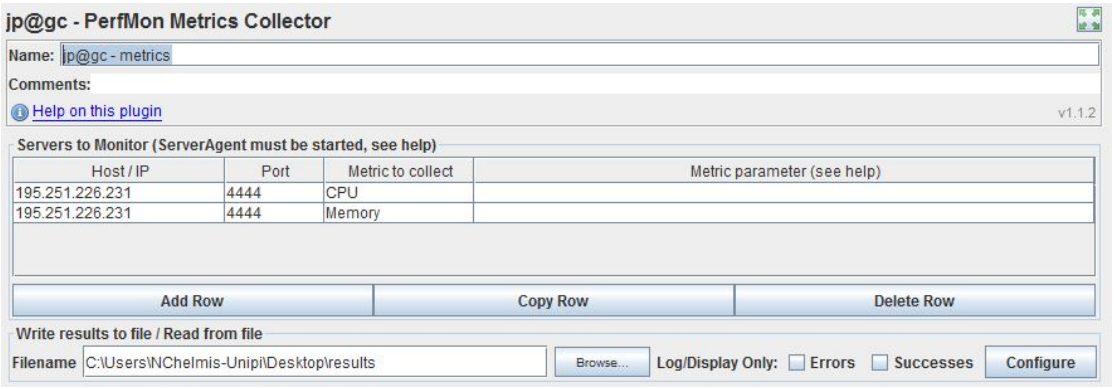
Τέλος προστίθεται κι ένα XML μήνυμα σαν παράμετρος ώστε με το πέρας του πειράματος να είναι εύκολα αναγνωρίσιμος ο αριθμός των μηνυμάτων που απέτυχαν. Στο Σχήμα 6.6 παρουσιάζονται οι ρυθμίσεις που περιγράφηκαν παραπάνω.



Σχήμα 6.6: Δημιουργία HTTP αιτήματος

Έχοντας ορίσει τον αριθμό των εικονικών χρηστών, το χρόνο δημιουργίας τους και την αίτηση HTTP που θα κάνουν, επόμενο βήμα είναι να δημιουργηθούν listeners στο JMeter ώστε να μπορεί να συλλέξει τα στατιστικά χρήστης του εξυπηρετητή. Στα πλαίσια του πειράματος δημιουργήθηκαν οι εξής listeners:

1. **jp@gc – metrics:** Ο βασικότερος listener του πειράματος καθώς κρατάει τα στατιστικά χρήσης CPU και RAM του εξυπηρετητή καθ' όλη τη διάρκεια του πειράματος. Για επιτευχθεί ο σκοπός αυτός συνεργάζεται με το PerfMon ServerAgent, που είναι ένας πράκτορας ο οποίος "σηκώνεται" στη πλευρά του εξυπηρετητή και πραγματοποιεί τις μετρήσεις που του έχουν οριστεί. Τις μετρήσεις αυτές τις στέλνει στο listener του JMeter ο οποίος τις αποθηκεύει. Οι ρυθμίσεις του listener φαίνονται στο Σχήμα 6.7. Οι παράμετροι που πρέπει να δηλωθούν είναι:
   - Η IP διεύθυνση του εξυπηρετητή,
   - η πόρτα (Port) στην οποία "ακούει" ο PerfMon ServerAgent,
   - και τα στατιστικά χρήσης που επιθυμούμε να μετρηθούν (CPU και RAM).



Σχήμα 6.7: Ρυθμίσεις listener καταγραφής στατιστικών χρήσης

2. **jp@gc - Active Threads Over Time:** Ο listener αυτός είναι επιφορτισμένος για τη καταγραφή των ενεργών εικονικών χρηστών σε κάθε χρονική στιγμή του πειράματος. Στο τέλος του πειράματος γίνεται η αντιστοίχηση των εικονικών χρηστών με τα στατιστικά χρήσης βάσει των αποτελεσμάτων του listener αυτού.

3. **jp@gc - Response Times Over Time:** Ο listener αυτός είναι επιφορτισμένος για την καταγραφή των χρόνων απόκρισης για τον κάθε εικονικό χρήστη με τη πάροδο του χρόνου. Στο τέλος του πειράματος και η μέτρηση αυτή αντιστοιχίζεται με τις προηγούμενες ώστε να δημιουργηθεί το απαραίτητο σύνολο δεδομένων για την παραγωγή της συνάρτησης.

Δείγμα των αποτελεσμάτων του πειράματος που περιγράφηκε υπάρχει στο Παράρτημα *A*.

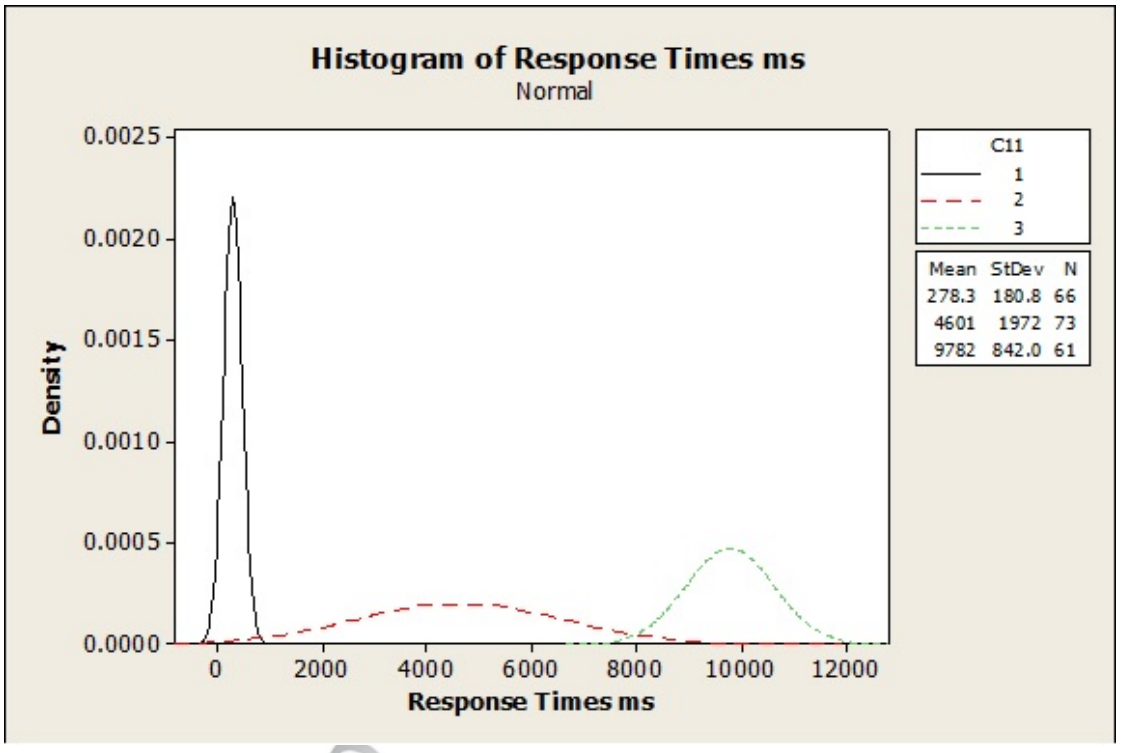
6.4 Μοντελοποίηση Δεδομένων

Μετά τη διεξαγωγή του πειράματος, βάσει της διαδικασίας που περιγράφηκε στην προηγούμενη ενότητα, με διαφορετικό ρυθμό εισαγωγής εικονικών χρηστών κάθε φορά και διαφορετικό βήμα, συγκεντρώθηκαν τα απαραίτητα για τη μοντελοποίηση δεδομένα. Από κάθε σετ δεδομένων, προτού χρησιμοποιηθεί, αφαιρέθηκαν οι “ακραίες” τιμές που δημιουργούσαν θόρυβο ώστε να μην επηρεάσουν το τελικό αποτέλεσμα.

Για τη μοντελοποίηση των δεδομένων χρησιμοποιήθηκε ανάλυση παλινδρόμησης . Αποτελεί ένα σύνολο τεχνικών που χρησιμοποιούνται για τη διερεύνηση της σχέσης

μεταξύ τουλάχιστον δύο μεταβλητών (τουλάχιστον μια ανεξάρτητη και μια εξαρτημένη μεταβλητή). Στα πλαίσια της παρούσας εργασίας εξαρτημένη μεταβλητή θεωρείται ο χρόνος απόκρισης, ενώ ανεξάρτητες μεταβλητές οι (i) αριθμός χρηστών, (ii) χρήση CPU και (iii) χρήση RAM.

Αρχικά δημιουργήθηκε ένα ιστόγραμμα, βασισμένο στους χρόνους απόκρισης, το οποίο παρουσιάζεται στο Σχήμα 6.8



Σχήμα 6.8: Ιστόγραμμα Χρόνων Απόκρισης

Από το ιστόγραμμα αυτό παρατηρείται ότι οι χρόνοι απόκρισης χωρίζονται σε τρείς ομάδες με διαφορετικό χαρακτηριστικό για κάθε ομάδα. Συγκεκριμένα, (i) η πρώτη ομάδα έχει μεγάλη κύρτωση, (ii) η δεύτερη ομάδα παρουσιάζει μεγάλη διασπορά, ενώ (iii) η τρίτη ομάδα είναι κανονική. Για το λόγο αυτό δημιουργήθηκε η ψευδομεταβλητή C11 η οποία έχει αναλάβει τη διαχείριση των ομάδων αυτών. Ειδικότερα, (i)η πρώτη ομάδα περιλαμβάνει χρόνους απόκρισης από 1 εώς 2000 χιλιοστά του δευτερολέπτου, (ii) η δεύτερη ομάδα περολαμβάνει χρόνους από 2001 Θεωρώντας τη ψευδομεταβλητή C11 ως ακόμα μια εξαρτημένη μεταβλητή η συνάρτηση που προκύπτει (6.1) και μοντελοποιεί με τη μεγαλύτερη ακρίβεια τα δεδομένα είναι:

$$RT = \frac{users}{-6878 - 296 * CPU + 2669 * RAM + 4870 * C11} \tag{6.1}$$

ΚΕΦΑΛΑΙΟ 7

Αξιολόγηση Μηχανισμού

"All models are wrong, but some are useful."
˜ George E. P. Box

7.1 Εισαγωγή

Στο προηγούμενο Κεφάλαιο περιγράφηκε η δομή του μηχανισμού που προτείνεται στην παρούσα ερευνητική εργασία καθώς και η λογική την οποία υλοποιεί για να επιτελέσει το σκοπό του. Προκειμένου να επαληθευτεί η ορθότητα τόσο της μαθηματικής συνάρτησης όσο και ο ίδιος ο μηχανισμός εκτελέστηκαν πραγματικά πειράματα επαλήθευσης.

Σκοπός του κεφαλαίο αυτού είναι:

1. **Επαλήθευση:** Δοκιμασία του προτεινόμενου μηχανισμού σε πραγματικό περιβάλλον για την επαλήθευση της εγκυρότητας του.
2. **Παρουσίαση Αποτελεσμάτων:** Η συνοπτική παρουσίαση και επεξήγηση των αποτελεσμάτων που προέκυψαν από τη διαδικασίας αξιολόγησης του μηχανισμού.

7.2 Τρόπος Αξιολόγησης

Σκοπός του πειράματος είναι να αξιολογήσει τον προτεινόμενο μηχανισμό σε πραγματικό περιβάλλον. Με αυτή τη λογική το πείραμα κατανεμηθεί σε τρείς διαφορετικές

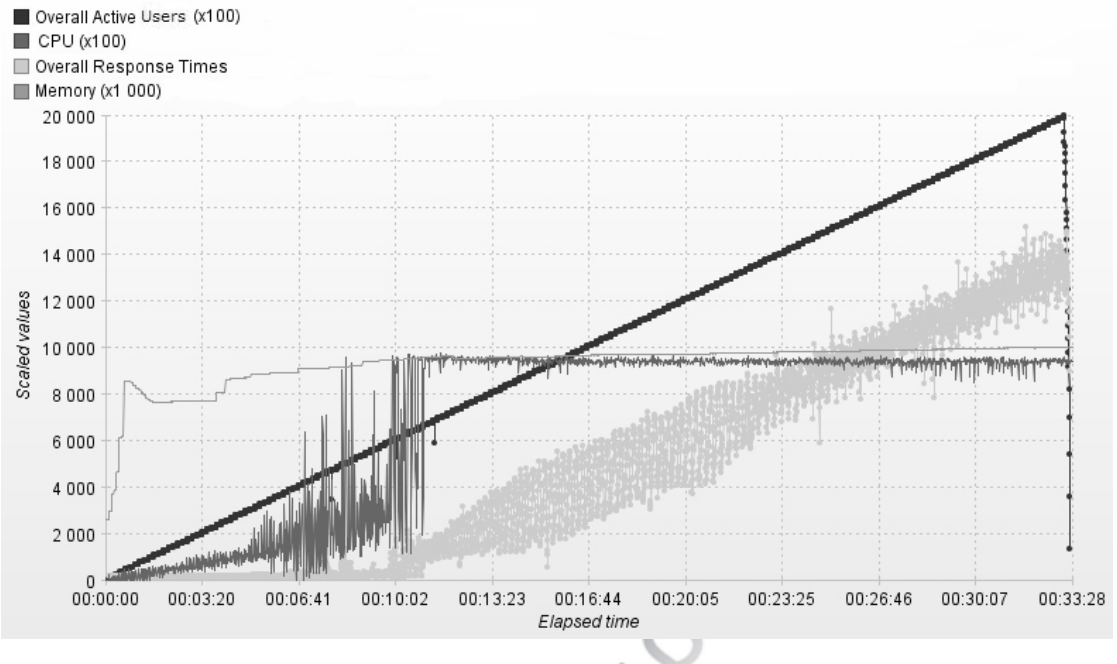
περιοχές στην Ευρώπη: (i) EPCC (Edinburgh), (ii) HLRS (Stuttgart), (iii) PSNC (Poznan). Την υποδομή παρείχε το BonFire [73].

Για τους σκοπούς του πειράματος αξιολόγησης αναπτύχθηκε μια απλή Java servlet υπηρεσιοστρεφής εφαρμογή. Όταν λάβει ένα HTTP Get αίτημα η υπηρεσία υπολογίζει ένα εκατομμύριο ψευδοτυχαίους ακέραιους αριθμούς που κυμαίνονται από το μηδέν έως το χίλια. Η εφαρμογή δεν αποθηκεύει την κατάσταση του χρήστη, καθιστώντας έτσι εύκολη τη διανομή της σε εξυπηρετητές που τρέχουν τον ίδιο κώδικα. Οι αιτήσεις φτάνουν σε ένα κόμβο εξυπηρέτησης φορτίου (Load Balancer) που αναλαμβάνει τη διανομή τους στους διάφορους κόμβους.

7.3 Αποτελέσματα

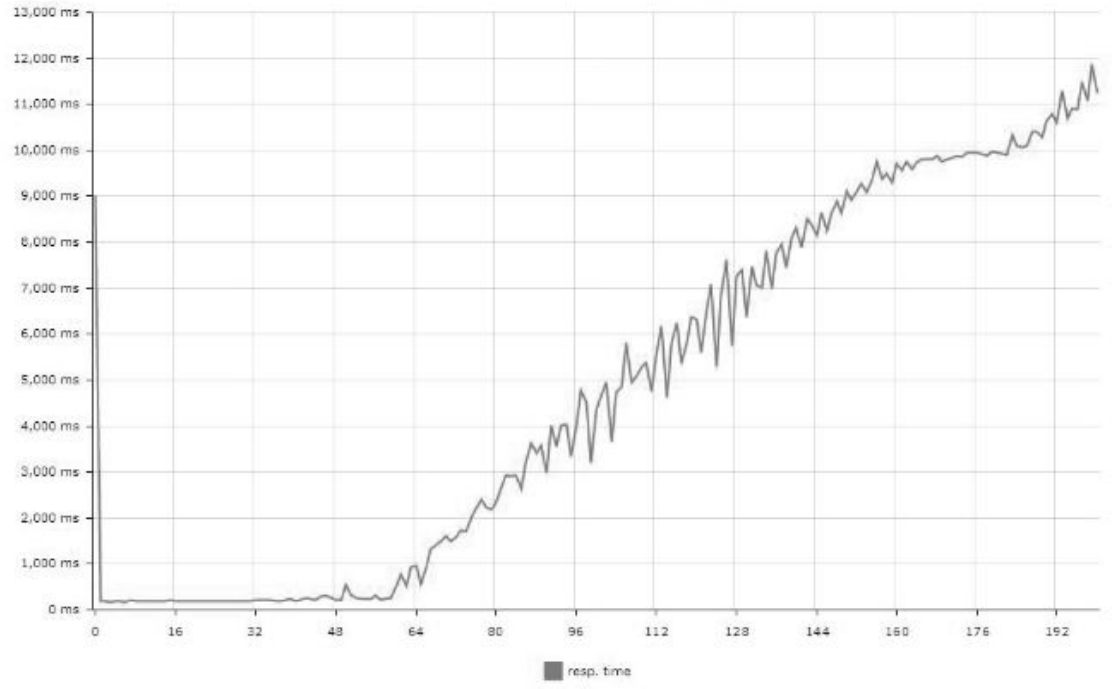
Στο πείραμα που παρουσιάζεται παρακάτω, συνολικά διακόσιοι χρήστες χρησιμοποιήθηκαν, ενώ εφαρμόζονται οι πολιτικές του ουδέτερου επιχειρηματικού μοντέλου, με ελάχιστο χρόνο απόκρισης τα 200 και μέγιστο τα 15000 χιλιοστά του δευτερολέπτου.

Τα στοιχεία που συλλέχθηκαν παρουσιάζονται στο Σχήμα 7.1. Οι πληροφορίες περιλαμβάνουν τους ενεργούς χρήστες καθ' όλη τη διάρκεια του πειράματος (Overall Active Users), τους χρόνους απόκρισης (Overall Response Times), καθώς και χρήση CPU και μνήμης (Memory). Όπως μπορεί να παρατηρηθεί η αύξηση των χρηστών, επέφερε αύξηση των πόρων με αποτέλεσμα οι χρόνοι απόκρισης να διατηρούνται σταθεροί. Η συνεχόμενη αύξηση των χρηστών οδήγησε σε αύξηση των χρόνων απόκρισης με αποτέλεσμα την αύξηση πόρων CPU. Η φύση της εφαρμογής πρέπει να ληφθεί υπ' όψιν: Απαιτεί πόρους CPU αλλά όχι μνήμης καθώς δεν χρησιμοποιεί καθόλου cashing.



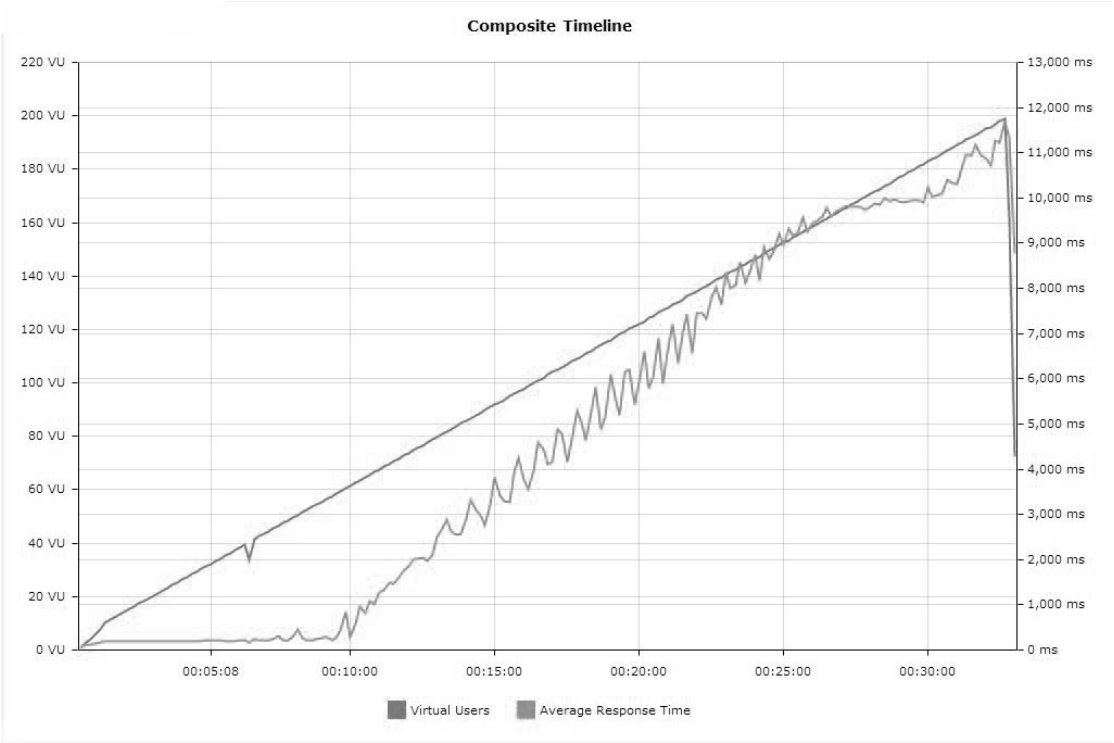
Σχήμα 7.1: Στοιχεία που συλλέχθηκαν

Όπως φαίνεται από το Σχήμα 7.1 στην αρχή του πειράματος αυξήθηκαν οι πόροι μνήμης αλλά εν συνεχεία παρέμειναν σταθεροί σε αντίθεση με τους πόρους CPU που συνέχισαν να αυξάνουν, αποδεικνύοντας πως η διαχείριση των πόρων ήταν σωστή. Η σύγκριση των χρόνων απόκρισης συγκριτικά με των αριθμό των χρηστών απεικονίζεται στο Σχήμα 7.2 και στο Σχήμα 7.3.



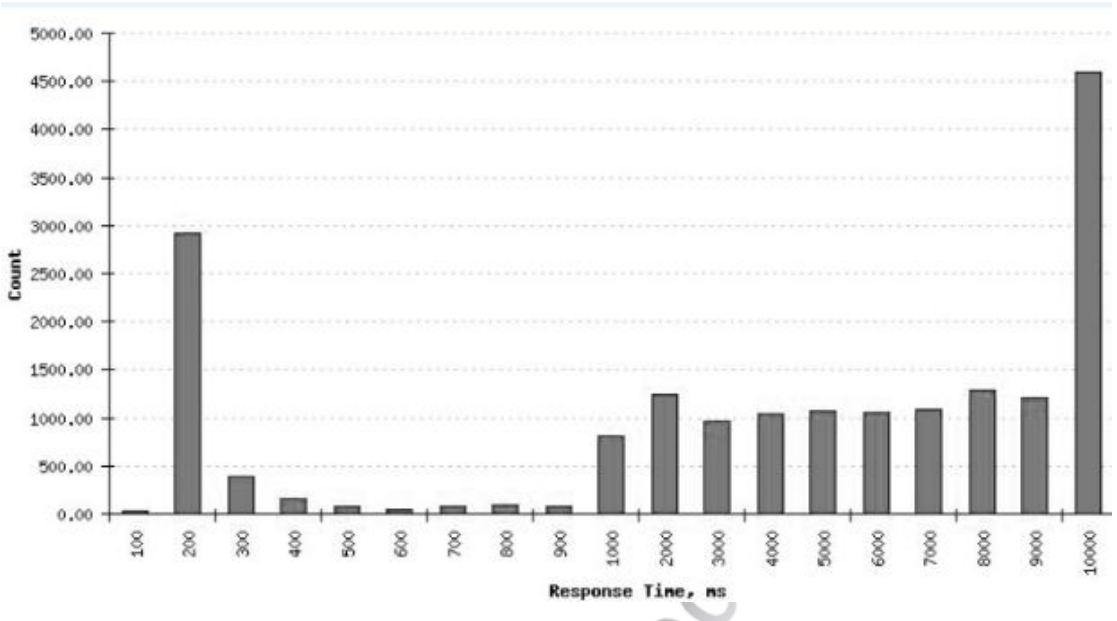
Σχήμα 7.2: Χρόνοι Απόκρισης Vs Χρήστες

Όπως φαίνεται από το Σχήμα 7.3 η άυξηση των χρηστών γίνεται γραμμικά, αντίθετα ο χρόνος απόκρισης δεν ακολουθεί τον ίδιο ρυθμό. Αρχικά διατηρείται σταθερός ενώ εν συνεχεία αρχίζει να αυξομειώνεται. Η αυξομείωση αυτή οφείλεται στη προσθήκη νέων πόρων.



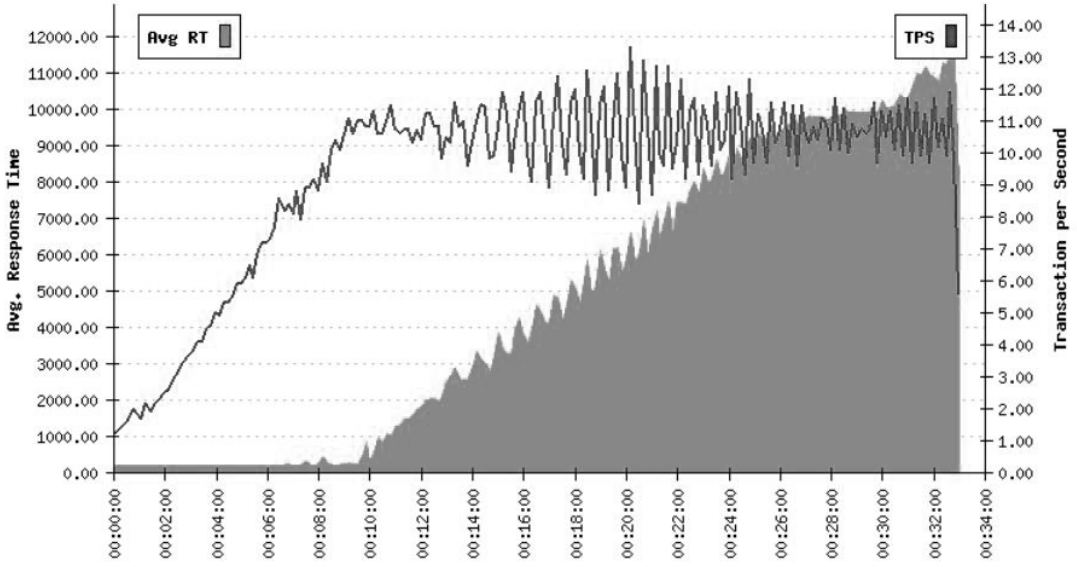
Σχήμα 7.3: Αύξηση Χρηστών & Χρόνου Απόκρισης

Η κατανομή των χρόνων απόκρισης κατά τη διάρκεια του πειράματος απεικονίζεται στο Σχήμα 7.4. Όπως μπορεί να παρατηρηθεί παρά τη γραμμική αύξηση των χρηστών ο χρόνος απόκρισης δεν ακολούθησε το ίδιο μοτίβο αλλά διατηρήθηκε σταθερός ανά περιόδους. Μια δεύτερη και σημαντικότερη παρατήρηση είναι πώς χαμηλότεροι χρόνοι απόκρισης (π.χ. 300-900 ms) εμφανίζονται αρκετά λιγότερο. Αυτό μπορεί εύκολα να εξηγηθεί καθώς όπως αναφέρθηκε χρησιμοποιούνται οι πολιτικές του ουδέτερου μοντέλου. Όσο οι χρόνοι απόκρισης παρέμεναν κοντά στον ελάχιστο υποσχόμενο χρόνο απόκρισης, δεν απαιτείτο η απόκτηση επιπλέον πόρων κατά συνέπεια η συνεχόμενη αύξηση των χρηστών επέφερε ταχύτερη αύξηση των χρόνων απόκρισης.



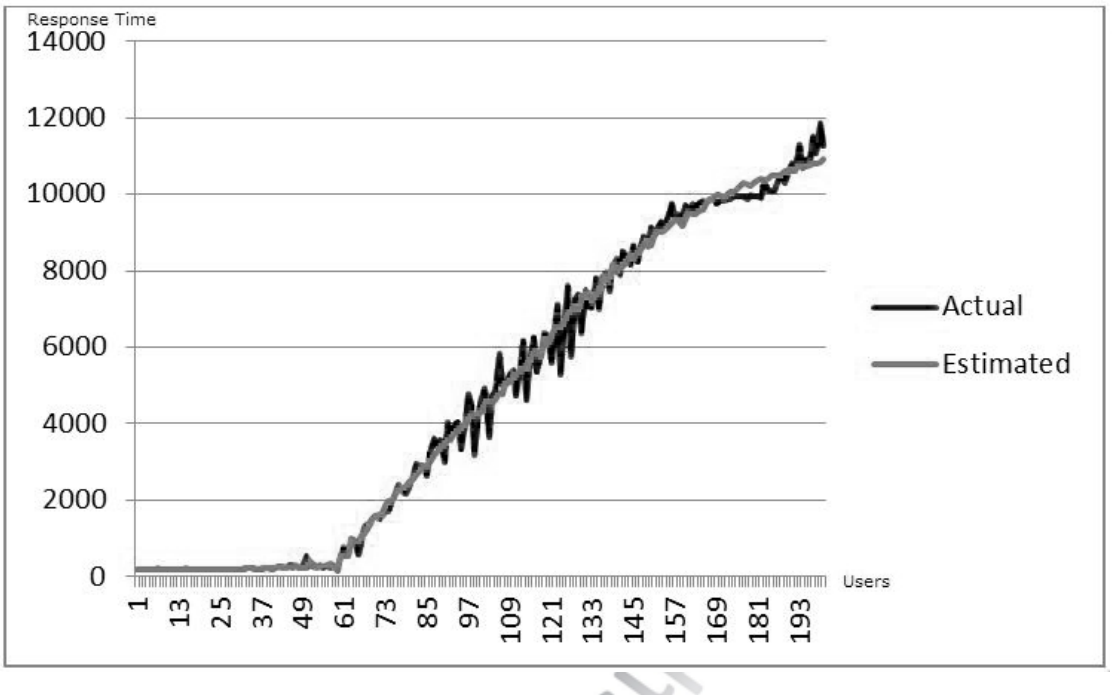
Σχήμα 7.4: Κατανομή Χρόνων Απόκρισης

Επιπλέον, συλλέχθηκαν πληροφορίες σχετικές με τις συναλλαγές (transactions) ανά δευτερόλεπτο (Transactions Per Second (TPS)) ώστε να απεικονιστεί καλύτερη η συμπεριφορά του χρόνου απόκρισης σε σχέση με την αύξηση του αριθμού των χρηστών. Οι πληροφορίες αυτές απεικονίζονται στο Σχήμα 7.5.

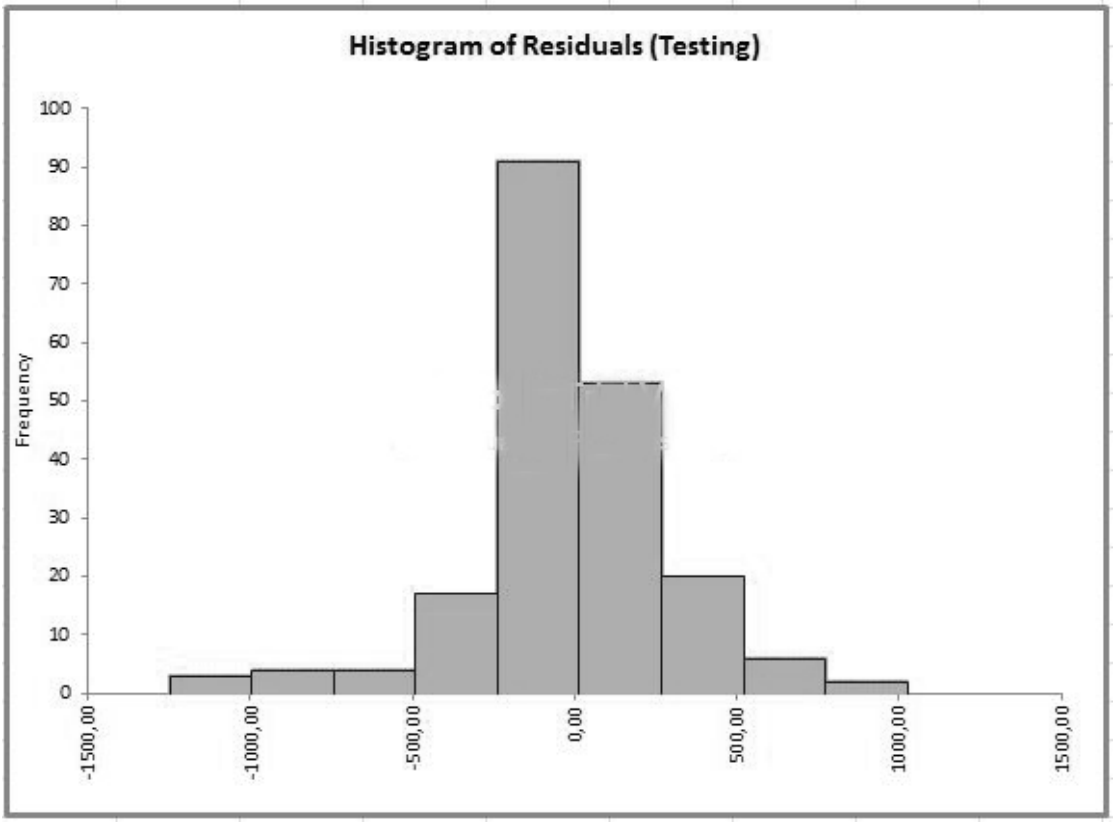


Σχήμα 7.5: Χρόνοι Απόκρισης και Ενέργειες ανά δευτερόλεπτο

Στο Σχήμα 7.6 παρουσιάζεται το βασικό κομμάτι της αξιολόγησης του μηχανισμού καθώς συγκρίνεται ο προβλεπόμενος χρόνος απόκρισης με τον πραγματικό. Όπως φαίνεται στο διάγραμμα, κάθε χρονική στιγμή, η διαφορά των πραγματικών χρόνων απόκρισης με την πρόβλεψη του μηχανισμού είναι μικρή και οι χρόνοι σχεδόν ταυτίζονται.



Σχήμα 7.6: Προβλεπόμενος Vs Πραγματικός Χρόνος Απόκρισης



Σχήμα 7.7: Ιστόγραμμα Υπολειμμάτων

Προκειμένου να απεικονιστεί εμφανέστερα το κατά πόσο συμπίπτει ο πραγματικός με τον προβλεπόμενο χρόνο απόκρισης, στο Σχήμα 7.7 παρουσιάζεται ένα ιστόγραμμα

με τα υπολείμματα (residuals) που προκύπτουν από τη μεταξύ τους σύγκριση [74].Ως υπόλειμμα μιας παρατηρούμενης τιμής, ορίζεται η διαφορά μεταξύ της παρατηρούμενης τιμής και της εκτιμώμενης. Όπως φαίνεται η μεγαλύτερη συγκέντρωση υπολειμμάτων παρουσιάζεται γύρω από το μηδέν, αποδεικνύοντας έτσι την ορθότητα των αποτελεσμάτων πρόβλεψης.Γίνεται λοιπόν εμφανές πως ο μηχανισμός προσδιορίζει των αριθμό των χρηστών συγκριτικά με τους απαιτούμενους μοντελοποιημένους πόρους με ακριβή τρόπο.

ΚΕΦΑΛΑΙΟ 8

Συμπεράσματα & Μελλοντική Έρευνα

"Science knows only one commandment—contribute to science."
˜ Galileo

8.1 Συμπεράσματα

Οι πάροχοι εφαρμογών αποσκοπούν στη μεγιστοποίηση της πελατειακής τους βάσης εξετάζοντας το κόστος. Προς αυτή την κατεύθυνση, προτάθηκαν επιχειρηματικά μοντέλα που προτείνουν τρόπους για την προσέλκυση πελατών λαμβάνοντας υπόψη περιορισμούς κόστους. Στόχος είναι η βελτιστοποίηση της απόδοσης της "επένδυσης" σε πόρους συγκριτικά με τον προσδοκώμενο αριθμό πελατών.

Ένας δυναμικός μηχανισμός ο οποίος είναι αποδοτικός για την επιχείρηση αναφορικά τόσο με τους τρέχοντες χρήστες όσο και με τους προσδοκώμενους συγκριτικά με τις ανάγκες σε πόρους, προτάθηκε. Σημείο κλειδί αποτελεί η αντιστοίχιση των δυναμικών επιχειρηματικών μοντέλων με πόρους βάσει των αναγκών.

Μέσα από την εκπόνηση της παρούσας ερευνητικής διπλωματικής εργασίας και κυρίως κατά τη διαδικασία αξιολόγησης του μηχανισμού προέκυψαν οι εξής παρατηρήσεις:

1. Ο αριθμός των πυρήνων CPU δεν αποτελεί μοναδικό δείγμα. Παρόλο που οι πάροχοι υποδομών συνήθως διπλασιάζουν τη τιμή, όταν ζητείται ένα μηχάνημα με δύο πυρήνες CPU και την οχταπλασιάζουν όταν ζητείται ένα μηχάνημα με οχτώ, η εξοικονόμηση των χρημάτων επιτυγχάνεται εάν δεν αυξηθεί η RAM. Εάν ακολουθηθεί αυτή η τακτική δεν σημαίνει ότι θα διπλασιαστεί / οχταπλασιαστεί και η απόδοση.

2. Η απόδοση ενός μηχανήματος διαφέρει από μέρα σε μέρα και από ώρα σε ώρα καθώς εξαρτάται από το συνολικό φόρτο εργασίας που έχει ο πάροχος της υποδομής.

3. Προκειμένου ένας ενδιάμεσος να έχει τη καλύτερη απόδοση στην οικονομικότερη τιμή συνίσταται να δημιουργήσει το δικό του τεστ (benchmark) που ανταποκρίνεται στις απαιτήσεις της εφαρμογής του και να το δοκιμάσει σε διαφορετικές πλατφόρμες.

8.2 Επίτευξη Στόχων

Όπως αναφέρθηκε στο Κεφάλαιο 1 για την εκτέλεση της παρούσας ερευνητικής εργασίας τέθηκαν ορισμένοι αντικειμενικοί στόχοι. Στη παρούσα ενότητα γίνεται μια ανασκόπηση των στόχων αυτών και συζητάτε το κατά πόσο ολοκληρώθηκαν.

Αντικειμενικός στόχος 1: **Κριτική ανασκόπηση βιβλιογραφίας:**
Ο στόχος αυτός επιτυγχάνεται μέσα από το δεύτερο και τρίτο κεφάλαιο της παρούσας ερευνητικής εργασίας. Οι βασικές βιβλιογραφικές έννοιες αναλύθηκαν στο Κεφάλαιο 2 ενώ παρουσιάζεται συνοπτικά η έννοια της ελαστικότητας που αποτελεί μέρος του Αντικειμενικού Στόχου 2. Στο Κεφάλαιο 3 παρουσιάζονται τόσο τα προβλήματα όσο και τα πλεονεκτήματα που προσφέρει το μοντέλο της Νεφοϋπολογιστικής καθώς και μοντέλα και λύσεις που προσφέρουν οι πάροχοι υποδομών, δείχνοντας έτσι την πραγματική ανάγκη για αντιστοίχηση μοντέλων ελαστικότητας με επιχειρηματικά μοντέλα.

Αντικειμενικός στόχος 2: **Ανάλυση θεμάτων ελαστικότητας και προσδιορισμός ανοικτών ερευνητικών περιοχών:**
Το τέταρτο κεφάλαιο είναι αφιερωμένο στην επίτευξη του στόχου αυτού. Η έννοια της ελαστικότητας αποσαφηνίζεται από αντίστοιχες έννοιες με τις οποίες συγχέεται βιβλιογραφικά, όπως αυτή της επεκτασιμότητας. Παρουσιάζεται, αφότου έχει συγκριθεί με άλλους, ο ορισμός της ελαστικότητας που η παρούσα διπλωματική εργασία θεωρεί βέλτιστο και τίθενται τα όρια και οι βασικές πτυχές του. Η ανάγκη για ελαστικότητα και γιατί αποτελεί το βασικότερο κομμάτι του μοντέλου της Νεφοϋπολογιστικής, φαίνεται

τόσο μέσα από υποθετικά όσο και πραγματικά παραδείγματα (π.χ. Animoto). Μέσα από τη βιβλιογραφία, προτάθηκε μια κατάταξη ελαστικών λύσεων, ενώ εν συνεχεία παρουσιάστηκαν οι σύγχρονες τάσεις και τα ερευνητικά έργα που ασχολούνται με ζητήματα ελαστικότητας. Τέλος, προκλήσεις και ζητήματα σχετικά με την ελαστικότητα παρουσιάζονται.

Αντικειμενικός στόχος 3: **Προσδιορισμός και ανάλυση επιχειρηματικών μοντέλων:**
Το πέμπτο κεφάλαιο, ολοκληρώνει τον στόχο αυτό καθώς ορίζει τις ανάγκες που καλούνται να καλύψουν τα επιχειρηματικά μοντέλα. Ορίζοντας όρια (μεγίστων και ελαχίστων τιμών χρόνου απόκρισης) επιτυγχάνεται η αντιστοίχηση με τα μοντέλα ελαστικότητας. Τέλος προτάθηκαν τρία τεχνοοικονομικά επιχειρηματικά μοντέλα (i)Αμυντικό, (ii)Ουδέτερο, (iii)Επιθετικό.

Αντικειμενικός στόχος 4: **Πρόταση αρχιτεκτονικής μηχανισμού:**
Στο Κεφάλαιο 6 παρουσιάστηκε αναλυτικά η λογική της αρχιτεκτονικής του μηχανισμού. Ο μηχανισμός που προτάθηκε είναι ανεξάρτητος της διαδικασίας εκκίνησης της εφαρμογής έτσι ώστε να είναι δυνατόν να μπορεί να αρχικοποιηθεί και ο μηχανισμός να προστεθεί αργότερα εάν χρειαστεί. Επιπλέον, ο μηχανισμός να μπορεί να χρησιμοποιεί πληθώρα επιχειρηματικών μοντέλων δυναμικά. Τέλος σημαντικό στοιχείο είναι ότι ο μηχανισμός να παρακολουθεί συνεχώς την εφαρμογή και μαθαίνει την κατάσταση της καθώς και τις απαιτήσεις της. Στο Σχήμα 6.1 παρουσιάστηκε ο τρόπος λειτουργίας του μηχανισμού.

Αντικειμενικός στόχος 5: **Συλλογή δεδομένων και μοντελοποίηση:**
Τα δεδομένα που οδήγησαν στη δημιουργία της μαθηματικής συνάρτησης συγκεντρώθηκαν μέσω πειραμάτων. Στο Κεφάλαιο 6 παρουσιάστηκε η λογική των πειραμάτων αυτών και αναλύθηκε ο τρόπος και η μέθοδος διεξαγωγής τους. Στο ίδιο κεφάλαιο παρουσιάστηκε και εξηγήθηκε η συνάρτηση που προέξυψε από τη μοντελοποίηση των δεδομένων αυτών καθώς και η μέθοδος παραγωγής της.

Αντικειμενικός στόχος 6: Αξιολόγηση:

Έλεγχος και αξιολόγηση: Στο Κεφάλαιο 6 παρουσιάστηκε η μέθοδος αξιολόγησης του μηχανισμού και αναλύθηκαν τα αποτελέσματα που προέκυψαν μέσω της διαδικασίας αξιολόγησης.

8.3 Μελλοντική Έρευνα

Ο μηχανισμός που παρουσιάστηκε στη παρούσα διπλωματική εργασία έρχεται ως λύση στο κομμάτι μεταξύ παρόχου εφαρμογής και τελικού πελάτη. Τα επιχειρηματικά μοντέλα που χρησιμοποιούν οι πάροχοι υποδομών δεν λήφθηκαν υπ' όψιν καθώς ήταν εκτός σκοπού. Σημαντικό σημείο αποτελεί και η επιλογή παρόχου υποδομής βάσει κριτηρίων όπως η γεωγραφική ζώνη, χρέωση υποδομής βάσει χρονικής περιόδου (π.χ. φθηνότερη χρήση της υποδομής τα μεσάνυχτα), οικονομικές ελαφρύνσεις λόγω μακροχρόνιας συνεργασίας καθώς και ο συνδυασμός αυτών. Μελλοντική έρευνα πάνω στο κομμάτι αυτό, και η πρόταση μηχανισμών που επεκτείνουν το μηχανισμό που προτάθηκε στην παρούσα εργασία μπορούν να επιτύχουν βελτιστοποίηση της χρήσης πόρων, από άκρη σε άκρη (end-to-end optimization).

ΠΑΡΑΡΤΗΜΑ A

Δείγμα Μετρήσεων Πειράματος

Πίνακας A.1: Δείγμα Μετρήσεων

# Users	CPU%	CPU (GHz)	RAM%	RAM (Gb)	Response Time (ms)
90	95,984	3,263456	9,639	1,06029	2978,022
91	95,273	3,239282	9,647	1,06117	4016,393
92	97,222	3,305548	9,653	1,06183	3559,69
93	95,728	3,254752	9,652	1,06172	4008,85
94	95,849	3,258866	9,663	1,06293	4033,708
95	95,964	3,262776	9,669	1,06359	3340,426
96	96,455	3,27947	9,671	1,06381	4017,857
97	95,984	3,263456	9,674	1,06414	4792,857
98	95,558	3,248972	9,675	1,06425	4514,019
99	95,712	3,254208	9,699	1,06689	3194,444
100	95,959	3,262606	9,711	1,06821	4344
101	95,849	3,258866	9,724	1,06964	4713,044
102	94,292	3,205928	9,729	1,07019	4946,903
103	95,255	3,23867	9,725	1,06975	3661,539
104	96,273	3,273282	9,732	1,07052	4731,092
105	96,375	3,27675	9,738	1,07118	4846,154
106	95,505	3,24717	9,742	1,07162	5822,034
107	96,683	3,287222	9,741	1,07151	4959,596
108	95,488	3,246592	9,743	1,07173	5064,815
109	95,801	3,257234	9,746	1,07206	5284,615
110	95,112	3,233808	9,747	1,07217	5388,235
111	96,343	3,275662	9,747	1,07217	4747,126
112	95,723	3,254582	9,751	1,07261	5507,143
113	96,009	3,264306	9,751	1,07261	6169,143
114	97,128	3,302352	9,748	1,07228	4613,333
115	96,014	3,264476	9,755	1,07305	5786,885
116	95,202	3,236868	9,768	1,07448	6252,174
117	95,226	3,237684	9,778	1,07558	5367,089
118	96,823	3,291982	9,777	1,07547	5813,084
119	94,943	3,228062	9,779	1,07569	6374,046
120	96,352	3,275968	9,781	1,07591	6318,681
121	95,363	3,242342	9,783	1,07613	5591,837
122	95,261	3,238874	9,783	1,07613	6521,739
123	95,744	3,255296	9,779	1,07569	7093,458
124	96,338	3,275492	9,779	1,07569	5294,118
125	95,843	3,258662	9,783	1,07613	6844,444
126	95,869	3,259546	9,779	1,07569	7615,385

Βιβλιογραφία

[1] Estelle Phillips, Derek Pugh, et al. *How to get a PhD: A handbook for students and their supervisors*. McGraw-Hill International, 2010.

[2] Mariana Carroll, Paula Kotzé, and Alta van der Merwe. Securing virtual and cloud environments. In *Cloud Computing and Services Science*, pages 73–90. Springer, 2012.

[3] Netlingo.com. cloud computing cloud definition, 2014. URL `http://www.netlingo.com/word/cloud-computing.php`.

[4] aws.amazon. Amazon ec2, 2014. URL `http://aws.amazon.com/ec2/`.

[5] Douglas F Parkhill. Challenge of the computer utility. 1966.

[6] Simson Garfinkel. The cloud imperative, 2011. URL `http://www.technologyreview.com/news/425623/the-cloud-imperative/`.

[7] Robert D. Hof. Jeff bezos' risky bet, 2006. URL `http://www.businessweek.com/stories/2006-11-12/jeff-bezos-risky-bet`.

[8] Amazon. Aws documentantion, 2006. URL `https://aws.amazon.com/documentation/?nc1=h_su_dm`.

[9] Carl Brooks. Amazon's early efforts at cloud computing? partly accidental, 2010. URL `http://itknowledgeexchange.techtarget.com/cloud-computing/amazons-early-efforts-at-cloud-computing-partly-accidental/`.

[10] Andreas Tolk. What comes after the semantic web-pads implications for the dynamic web. In *Proceedings of the 20th Workshop on Principles of Advanced and Distributed Simulation*, page 55. IEEE Computer Society, 2006.

[11] GanesanSenthilvel. Cloud computing position, 2012. URL `http://nhatnguyen.net/cloud-programming-concepts.aspx`.

[12] Andrew Tanenbaum and Maarten Van Steen. *Distributed systems*. Pearson Prentice Hall, 2007.

[13] Ian Foster, Yong Zhao, Ioan Raicu, and Shiyong Lu. Cloud computing and grid computing 360-degree compared. In *Grid Computing Environments Workshop, 2008. GCE'08*, pages 1–10. Ieee, 2008.

[14] Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, and Ivona Brandic. Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation computer systems*, 25(6):599–616, 2009.

[15] William Voorsluys, James Broberg, and Rajkumar Buyya. Introduction to cloud computing. *Cloud Computing*, pages 1–41, 2011.

[16] Peter Mell and Tim Grance. The nist definition of cloud computing. *National Institute of Standards and Technology*, 53(6):50, 2009.

[17] Chunye Gong, Jie Liu, Qiang Zhang, Haitao Chen, and Zhenghu Gong. The characteristics of cloud computing. In *Parallel Processing Workshops (ICPPW), 2010 39th International Conference on*, pages 275–279. IEEE, 2010.

[18] Shuai Zhang, Shufen Zhang, Xuebin Chen, and Xiuzhen Huo. Cloud computing research and development trend. In *Future Networks, 2010. ICFN'10. Second International Conference on*, pages 93–97. IEEE, 2010.

[19] Google App Engine. Google app engine, 2014. URL `https://developers.google.com/appengine/?csw=1`.

[20] Azure. Microsoft azure, 2014. URL `http://azure.microsoft.com/en-us/documentation/`.

[21] Pankaj Arora, RC Wadhawann, and Er SP Ahuja. Cloud computing security issues in infrastructure as a service. *International Journal of Advanced Research in Computer Science and Software Engineering*, 2(1), 2012.

[22] Hai Jin, Shadi Ibrahim, Tim Bell, Wei Gao, Dachuan Huang, and Song Wu. Cloud types and services. In *Handbook of Cloud Computing*, pages 335–355. Springer, 2010.

[23] Ashish Rastogi. A model based approach to implement cloud computing in e-governance. *International Journal of Computer Applications*, 9(7):15–18, 2010.

[24] Shyam Kumar Doddavula and Amit Wasudeo Gawande. Adopting cloud computing: enterprise private clouds. *SETLabs Briefings*, 7(7):11–18, 2009.

[25] Zaigham Mahmood and Richard Hill. *Cloud Computing for enterprise architectures*. Springer, 2011.

[26] Griffin. Virtual machines: Virtualization vs. emulation, 2006. URL `http://www.griffincaprio.com/blog/2006/08/virtual-machines-virtualization-vs-emulation.html`.

[27] S Crosby, R Doyle, M Gering, M Gionfriddo, S Hand, M Hapner, D Hiltgen, M Johanssen, J Leung, F Machida, et al. Open virtualization format specification. *Standards and Technology, no. DSP0243 in DMTF Specifications, Distributed Management Task Force*, 2009.

[28] Clovis Chapman, Wolfgang Emmerich, Fermın Galán Marquez, Stuart Clayman, and Alex Galis. Elastic service definition in computational clouds. In *Network Operations and Management Symposium Workshops (NOMS Wksps), 2010 IEEE/IFIP*, pages 327–334. IEEE, 2010.

[29] Ekasit Kijsipongse and Sornthep Vannarat. Autonomic resource provisioning in rocks clusters using eucalyptus cloud computing. In *Proceedings of the International Conference on Management of Emergent Digital EcoSystems*, pages 61–66. ACM, 2010.

[30] Johannes Kirschnick, Jose M Alcaraz Calero, Lawrence Wilcock, and Nigel Edwards. Toward an architecture for the automated provisioning of cloud services. *Communications Magazine, IEEE*, 48(12):124–131, 2010.

[31] Luis Rodero-Merino, Luis M Vaquero, Victor Gil, Fermín Galán, Javier Fontán, Rubén S Montero, and Ignacio M Llorente. From infrastructure delivery to service management in clouds. *Future Generation Computer Systems*, 26(8): 1226–1240, 2010.

[32] Huan Liu. Rapid application configuration in amazon cloud using configurable virtual appliances. In *Proceedings of the 2011 ACM Symposium on Applied Computing*, pages 147–154. ACM, 2011.

[33] Tharam Dillon, Chen Wu, and Elizabeth Chang. Cloud computing: issues and challenges. In *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*, pages 27–33. Ieee, 2010.

[34] Abzetdin Adamov and Murat Erguvan. The truth about cloud computing as new paradigm in it. In *Application of Information and Communication Technologies, 2009. AICT 2009. International Conference on*, pages 1–3. IEEE, 2009.

[35] Armando Fox, Rean Griffith, A Joseph, R Katz, A Konwinski, G Lee, D Patterson, A Rabkin, and I Stoica. Above the clouds: A berkeley view of cloud computing. *Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS*, 28:13, 2009.

[36] A Gupta. Cloud computing growing interest and related concerns. In *Computer Technology and Development (ICCTD), 2010 2nd International Conference on*, pages 462–465. IEEE, 2010.

[37] Philip Koehler, Arun Anandasivam, and MA Dan. Cloud services from a consumer perspective. 2010.

[38] Yashpalsinh Jadeja and Kirit Modi. Cloud computing-concepts, architecture and challenges. In *Computing, Electronics and Electrical Technologies (ICCEET), 2012 International Conference on*, pages 877–880. IEEE, 2012.

[39] Joel Gibson, Robin Rondeau, Darren Eveleigh, and Qing Tan. Benefits and challenges of three cloud computing service models. In *CASoN*, pages 198–205, 2012.

[40] Yanpei Chen, Vern Paxson, and Randy H Katz. What's new about cloud computing security. *University of California, Berkeley Report No. UCB/EECS-2010-5 January*, 20(2010):2010–5, 2010.

[41] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warfield. Xen and the art of virtualization. *ACM SIGOPS Operating Systems Review*, 37(5):164–177, 2003.

[42] Allan Leinwand. The hidden cost of the cloud: Bandwidth charges, 2009.

[43] Dustin Owens. Securing elasticity in the cloud. *Commun. ACM*, 53(6):46–51, 2010.

[44] Upendra Sharma, Prashant Shenoy, Sambit Sahu, and Anees Shaikh. A cost-aware elasticity provisioning system for the cloud. In *Distributed Computing Systems (ICDCS), 2011 31st International Conference on*, pages 559–570. IEEE, 2011.

[45] Dahai Xu, Ying Li, Mung Chiang, and A Calderbank. Elastic service availability: utility framework and optimal provisioning. *Selected Areas in Communications, IEEE Journal on*, 26(6):55–65, 2008.

[46] Zhiming Shen, Sethuraman Subbiah, Xiaohui Gu, and John Wilkes. Cloudscale: elastic resource scaling for multi-tenant cloud systems. In *Proceedings of the 2nd ACM Symposium on Cloud Computing*, page 5. ACM, 2011.

[47] Pavlos Kranas, Vasilios Anagnostopoulos, Andreas Menychtas, and Theodora Varvarigou. Elaas: An innovative elasticity as a service framework for dynamic management across the cloud stack layers. In *Complex, Intelligent and Software*

Intensive Systems (CISIS), 2012 Sixth International Conference on, pages 1042–1049. IEEE, 2012.

[48] Markus Klems. The cloud wars: $100+ billion at stake, 2008. URL `http://web2.sys-con.com/node/604936`.

[49] Jason. Amazon.com ceo jeff bezos on animoto, 2008. URL `http://animoto.com/blog/news/company-news/amazon-com-ceo-jeff-bezos-on-animoto/`.

[50] Michael Kuperberg, Nikolas Herbst, Joakim von Kistowski, and Ralf Reussner. *Defining and quantifying elasticity of resources in cloud computing and scalable platforms*. KIT, Fakultät für Informatik, 2011.

[51] Gene M Amdahl. Validity of the single processor approach to achieving large scale computing capabilities. In *Proceedings of the April 18-20, 1967, spring joint computer conference*, pages 483–485. ACM, 1967.

[52] Wikipedia the free encyclopedia. Amdahl's law, 2014. URL `http://en.wikipedia.org/wiki/Amdahl's_law`.

[53] Nikolas Roman Herbst, Samuel Kounev, and Ralf Reussner. Elasticity in cloud computing: What it is, and what it is not. In *ICAC*, pages 23–27, 2013.

[54] Guilherme Galante and Luis Carlos E de Bona. A survey on cloud computing elasticity. In *Proceedings of the 2012 IEEE/ACM Fifth International Conference on Utility and Cloud Computing*, pages 263–270. IEEE Computer Society, 2012.

[55] Luis M Vaquero, Luis Rodero-Merino, and Rajkumar Buyya. Dynamically scaling applications in the cloud. *ACM SIGCOMM Computer Communication Review*, 41 (1):45–52, 2011.

[56] Peter Wayner. Ultimate cloud speed tests: Amazon vs. google vs. windows azure, 2014. URL `http://www.infoworld.com/print/237169`.

[57] Harold C Lim, Shivnath Babu, Jeffrey S Chase, and Sujay S Parekh. Automated control in cloud computing: challenges and opportunities. In *Proceedings of the 1st workshop on Automated control for datacenters and clouds*, pages 13–18. ACM, 2009.

[58] Wesam Dawoud, Ibrahim Takouna, and Christoph Meinel. Elastic vm for cloud resources provisioning optimization. In *Advances in Computing and Communications*, pages 431–445. Springer, 2011.

[59] Zhenhuan Gong, Xiaohui Gu, and John Wilkes. Press: Predictive elastic resource scaling for cloud systems. In *Network and Service Management (CNSM), 2010 International Conference on*, pages 9–16. IEEE, 2010.

[60] Nedeljko Vasić, Dejan Novaković, Svetozar Miučin, Dejan Kostić, and Ricardo Bianchini. Dejavu: accelerating resource allocation in virtualized environments. *ACM SIGARCH Computer Architecture News*, 40(1):423–436, 2012.

[61] Shicong Meng, Ling Liu, and Vijayaraghavan Soundararajan. Tide: achieving self-scaling in virtualized datacenter management middleware. In *Proceedings of the 11th International Middleware Conference Industrial track*, pages 17–22. ACM, 2010.

[62] Josep Oriol Fito, Inigo Goiri, and Jordi Guitart. Sla-driven elastic cloud hosting provider. In *Parallel, Distributed and Network-Based Processing (PDP), 2010 18th Euromicro International Conference on*, pages 111–118. IEEE, 2010.

[63] Paul Marshall, Kate Keahey, and Tim Freeman. Elastic site: Using clouds to elastically extend site resources. In *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pages 43–52. IEEE Computer Society, 2010.

[64] Xinwen Zhang, Anugeetha Kunjithapatham, Sangoh Jeong, and Simon Gibbs. Towards an elastic application model for augmenting the computing capabilities of mobile devices with cloud computing. *Mobile Networks and Applications*, 16(3):270–284, 2011.

[65] Rodrigo N Calheiros, Christian Vecchiola, Dileban Karunamoorthy, and Rajkumar Buyya. The aneka platform and qos-driven resource provisioning for elastic applications on hybrid clouds. *Future Generation Computer Systems*, 28 (6):861–870, 2012.

[66] Iulian Neamtiu. Elastic executions from inelastic programs. In *Proceedings of the 6th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, pages 178–183. ACM, 2011.

[67] Thomas Knauth and Christof Fetzer. Scaling non-elastic applications using virtual machines. In *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, pages 468–475. IEEE, 2011.

[68] Dinesh Rajan, Anthony Canino, Jesus A Izaguirre, and Douglas Thain. Converting a high performance application to an elastic cloud application. In *Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on*, pages 383–390. IEEE, 2011.

[69] FBR Costa. On the amplitude of the elasticity offered by public cloud computing providers. *Federal University of Campina Grande, Campina Grande, Tech. Rep*, 2011.

[70] Paul C Brebner. Is your cloud elastic enough?: performance modelling the elasticity of infrastructure as a service (iaas) cloud applications. In *Proceedings of the 3rd ACM/SPEC International Conference on Performance Engineering*, pages 263–266. ACM, 2012.

[71] Dictionary.com. Business model, 2008. URL `http://dictionary.reference.com/browse/business+model`.

[72] Christof Weinhardt, Dipl-Inform-Wirt Arun Anandasivam, Benjamin Blau, Dipl-Inform Nikolay Borissov, Dipl-Math Thomas Meinl, Dipl-Inform-Wirt Wibke Michalk, and Jochen Stößer. Cloud computing–a classification, business models, and research directions. *Business & Information Systems Engineering*, 1 (5):391–399, 2009.

[73] Konstantinos Kavoussanakis, Alastair Hume, Josep Martrat, Carmelo Ragusa, Michael Gienger, Konrad Campowsky, Gregory Van Seghbroeck, Constantino Vázquez, Celia Velayos, Frédéric Gittler, et al. Bonfire: the clouds and services testbed. In *Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on*, volume 2, pages 321–326. IEEE, 2013.

[74] Wikipedia the free encyclopedia. Errors and residuals in statistics, 2014. URL `http://en.wikipedia.org/wiki/Errors_and_residuals_in_statistics`.

Ακρωνύμια

aaS as a Service. 1

API Application Programming Interface. 22, 30, 32, 45, 50

CaaS Computation as a Service. 18

CPU Central Process Unit. 5, 18, 22, 24, 30, 32, 38, 43, 45, 46, 48, 50, 52–54, 57, 61, 76, 78, 80, 81, 86

DaaS Data as a Service. 18

FFT Fast Fourier Transformation. 52

GB gigabyte. 18, 49, 50

HTTP Hyper Text Transfer Protocol. 24, 51, 73–76, 80

HTTPS Hyper Text Transfer Protocol Secure. 51

I/O Input / Output. 57

IaaS Infrastructure as a Service. 18, 31, 44, 49, 52, 56, 57

IDC International Data Corporation. 27, 29, 34

IP Internet Protocol. 76

ISO International Organization for Standardization. 24

NIST National Institute of Standards and Technology. 15, 17, 19, 42

OVF Open Virtualization Format. 24

PaaS Platform as a Service. 17, 44, 49, 50, 55, 56

QoE Quality of Experience. 3

QoS Quality of Service. 3

RAM Random access memory. 5, 18, 24, 32, 38, 43, 45, 46, 48, 57, 76, 78, 86

REST Representational State Transfer. 49, 50

SaaS Software as a Service. 17, 27, 30, 44

SLA Service-Level Agreement. 32, 45, 54

SOAP Simple Object Access Protocol. 49

SSH Secure Shell. 24

SU Spin Up Time. 58

TCP Transmission Control Protocol. 48

TPS Transactions Per Second. 83

VM Virtual Machine. 20

VPN Virtual Private Network. 11

XML Extensible Markup Language. 75