Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής
Πρόγραμμα Μεταπτυχιακών Σπουδών
«Προηγμένα Συστήματα Πληροφορικής»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**Προσομοίωση Υπολογιστικών Νεφών** **Simulation of Cloud Computing Infrastructures**
Ονοματεπώνυμο Φοιτητή	**Νικόλαος Κυριαζόπουλος**
Πατρώνυμο	**Λεωνίδας**
Αριθμός Μητρώου	**ΜΠΣΠ/15042**
Επιβλέπων	**Χρήστος Δουληγέρης, Καθηγητής**

Ημερομηνία Παράδοσης **Ιούνιος 2017**

Τριμελής Εξεταστική Επιτροπή

(υπογραφή)	(υπογραφή)	(υπογραφή)
Όνομα Επώνυμο Βαθμίδα	Όνομα Επώνυμο Βαθμίδα	Όνομα Επώνυμο Βαθμίδα

Ευχαριστίες

Καταρχάς, θα ήθελα να ευχαριστήσω τον καθηγητή κ. Χρήστο Δουληγέρη, για την εμπιστοσύνη που μου έδειξε στην ανάθεση της παρούσας μεταπτυχιακής διατριβής και για τον πολύτιμο χρόνο που αφιέρωσε, για τις απορίες, διορθώσεις και συναντήσεις μας.

Επίσης, ευχαριστώ τους γονείς μου, Λεωνίδα και Σοφία, για την μεγάλη ηθική, ψυχική και οικονομική υποστήριξη τους όλα αυτά τα χρόνια, που στερήθηκαν πολλά για να μου δώσουν τα απαραίτητα εφόδια ώστε να καταφέρω να πετύχω τους στόχους μου και να γίνω καλύτερος άνθρωπος. Χωρίς εκείνους δεν θα είχα φτάσει ως εδώ.

Τέλος, δεν θα μπορούσα να παραλείψω τη σύντροφο της ζωής μου, Βάσια, που με υπομονή, συμπαράσταση και αγάπη με βοηθάει όλα αυτά τα χρόνια να θέτω νέους στόχους.

ΠΕΡΙΛΗΨΗ

Προσομοίωση Υπολογιστικών Νεφών

Στην παρούσα μεταπτυχιακή διατριβή, αναλύονται τα κύρια χαρακτηριστικά, η αρχιτεκτονική και η λειτουργία των προσομοιωτών υπολογιστικών νεφών. Στα πλαίσια της διατριβής, εγκαταστάθηκαν και αναλύονται οι διαδεδομένοι προσομοιωτές CloudSim, CloudAnalyst και GreenCloud. Ο αναγνώστης μέσα από τις οδηγίες χρήσης, τις λεπτομερείς εικόνες, και τα παραδείγματα που παρουσιάζονται μπορεί εύκολα να εκτελέσει τα δικά του πειράματα και να εξαγάγει χρήσιμα συμπεράσματα για την λειτουργία των υπολογιστικών νεφών. Κλείνοντας την μεταπτυχιακή διατριβή, στο τελευταίο κεφάλαιο γίνεται σύγκριση μεταξύ των τριών προαναφερθέντων προσομοιωτών και παρουσιάζονται τα συμπεράσματα από την εμπειρία χρήσης και εγκατάστασής τους.

***Λέξεις Κλειδιά**: προσομοιωτές, υπολογιστικό νέφος, cloudsim, cloudanalyst, greencloud, εγκατάσταση, συμπεράσματα*

ABSTRACT

Cloud Computing Simulators

This master thesis analyzes the main features, the architecture and the function of computer cloud simulators. In the context of the dissertation, the widespread CloudSim, CloudAnalyst and GreenCloud simulators were installed and analyzed. The reader through the usage instructions, detailed illustrations, and examples presented can readily perform his own experiments and extract useful conclusions about how cloud computing works. The last chapter compares the three simulators mentioned above and presents the conclusions from the experience gained by their use and installation.

***Keywords**: simulators, cloud computing, cloudsim, cloudanalyst, greencloud, installation, conclusions*

Περιεχόμενα

Εικόνες

Πίνακες

Πηγαίος Κώδικας

Εισαγωγή

> "Cloud computing is the third wave of the digital revolution."
>
> Lowell McAdam, Πρόεδρος και CEO της Verizon Communications

Το Υπολογιστικό Νέφος έχει στόχο να παραδώσει αξιόπιστες, ασφαλείς, ανεκτικές στα λάθη, βιώσιμες και επεκτάσιμες υποδομές για τη φιλοξενία εφαρμογών που είναι βασισμένες στο διαδίκτυο. Η μοντελοποίηση και ο προγραμματισμός αυτών των εφαρμογών και των υπηρεσιών σε μια υποδομή Υπολογιστικού Νέφους, είναι τεράστια πρόκληση, η οποία απαιτεί διαφορετικό φόρτο και ενεργειακή απόδοση.

Όσο ο αριθμός των χρηστών στις υπηρεσίες Υπολογιστικού Νέφους αυξάνεται, τόσο οι ερευνητές και οι εταιρείες χρησιμοποιούν τους προσομοιωτές νέφους, οι οποίοι μπορούν εύκολα και γρήγορα να προσομοιώσουν την κατάσταση και την συμπεριφορά μιας υποδομής νέφους, τις πολιτικές που θα ακολουθηθούν και τον φόρτο που μπορεί να αντέξει μια τέτοια υποδομή.

Η μεταπτυχιακή διατριβή δομείται σε 4 κεφάλαια. Στο πρώτο κεφάλαιο περιγράφεται η Υπολογιστική Νέφους, γίνεται μια ιστορική αναδρομή στο παρελθόν και πώς κατέληξε να γίνει από τις πιο καινοτόμες τεχνολογίες της εποχής μας, αναλύονται τα μοντέλα υπηρεσιών Software as a Service (SaaS), Platform as a Service (PaaS), Infrastructure as a Service (IaaS), Containers as a Service (CaaS) και τα μοντέλα ανάπτυξής τους. Ακόμα αναφέρονται οι χρήσεις, τα πλεονεκτήματα και τα μειονεκτήματα αυτής της τεχνολογίας.

Στο δεύτερο κεφάλαιο παρουσιάζεται το πακέτο προσομοίωσης CloudSim. Αναλύονται όλα τα δομικά στοιχεία που το απαρτίζουν, η αρχιτεκτονική του, οι βασικές οντότητες και παρουσιάζονται τα βήματα εγκατάστασης και λειτουργίας του προσομοιωτή με τη βοήθεια του Eclipse IDE. Τέλος, ακολουθούν όλα τα παραδείγματα του προσομοιωτή μαζί με τα αποτελέσματα τους και εξάγονται συμπεράσματα.

Στο τρίτο κεφάλαιο αναλύεται το εργαλείο προσομοίωσης CloudAnalyst, το οποίο παρέχει και γραφικό περιβάλλον για την αλληλεπίδραση με τον χρήστη. Στα πλαίσια της εγκατάστασης και της χρήσης του προγράμματος, γίνεται η προσομοίωση του μέσου κοινωνικής δικτύωσης LinkedIn με βάση τα στατιστικά στοιχείων των χρηστών σε όλες τις περιοχές.

Στο τέταρτο κεφάλαιο αναλύεται το πακέτο προσομοίωσης GreenCloud. Γίνεται εκτενής περιγραφή της αρχιτεκτονικής του και στη συνέχεια παρουσιάζονται τα βήματα εγκατάστασής του. Τέλος, διεξάγεται η προσομοίωση και αναλύεται η εικόνα των αποτελεσμάτων.

Κλείνοντας, ακολουθούν τα συμπεράσματα από την χρήση των προσομοιωτών υπολογιστικών νεφών και η σύγκρισή τους.

Για την συγγραφή της διατριβής χρησιμοποιήθηκαν οι πηγές που προέρχονται από επιστημονικά άρθρα, από δημοσιεύσεις για τις τεχνολογίες των προσομοιωτών (White Papers) και από διάφορες ιστοσελίδες με οδηγίες εγκατάστασης.

Κατά την συγγραφή της μεταπτυχιακής διατριβής δεν προέκυψαν δυσκολίες, στην εγκατάσταση των προγραμμάτων και την εκτέλεση των πειραμάτων. Τα εργαλεία προσομοίωσης μπορούν να δουλέψουν σε όλα τα λειτουργικά συστήματα, με την βοήθεια εφαρμογών υπερεπόπτη (hypervisor), όπως το VMWare ή το VirtualBox.

Κεφάλαιο 1

Cloud Computing

"You don't generate your own electricity. Why generate your own computing?"

Jeff Bezos, Ιδρυτής, πρόεδρος και CEO της Amazon

1.1 Εισαγωγή

Η υπολογιστική νέφους είναι ένα μοντέλο που επιτρέπει την ευέλικτη, on-demand δικτυακή πρόσβαση σε ένα κοινόχρηστο σύνολο από υπολογιστικούς πόρους που μπορούν να παραμετροποιηθούν (π.χ. δίκτυα, εξυπηρετητές, αποθηκευτικός χώρος, εφαρμογές και υπηρεσίες), οι οποίοι μπορούν να αναπτυχθούν ή να αποδεσμευτούν πολύ γρήγορα, με ελάχιστη προσπάθεια διαχείρισης ή αλληλεπίδρασης με τον πάροχο των υπηρεσιών. Το μοντέλο του νέφους, προωθεί την διαθεσιμότητα και αποτελείται από 5 βασικά χαρακτηριστικά, τρία μοντέλα υπηρεσιών και τέσσερα μοντέλα ανάπτυξης. [1]

- Υπηρεσία on-demand: Ο καταναλωτής μπορεί μονομερώς να προμηθευτεί υπολογιστικές δυνατότητες, όπως κάποιον server και δικτυακή αποθήκευση, εάν χρειάζεται, αυτόματα και χωρίς να απαιτείται η αλληλεπίδραση του ανθρώπου με τον πάροχο υπηρεσιών νέφους.
- Ευρεία δικτυακή πρόσβαση: Οι δυνατότητες του νέφους είναι διαθέσιμες από ένα δίκτυο και προσβάσιμες από συγκεκριμένους μηχανισμούς που προωθούν την χρήση από ετερογενείς thin ή thick πλατφόρμες πελατών (π.χ. κινητά τηλέφωνα, laptops, PDAs).
- Διαμοιρασμός πόρων: Οι υπολογιστικοί πόροι του παρόχου, διαμοιράζονται για να εξυπηρετούν πολλαπλούς καταναλωτές, χρησιμοποιώντας το μοντέλο των πολλών «ενοικιαστών», με διαφορετικούς φυσικούς και εικονικούς πόρους που ανατίθενται δυναμικά και μπορούν να ανατεθούν ξανά ανάλογα με την καταναλωτική ζήτηση.
- Πολύ γρήγορη ελαστικότητα: Οι δυνατότητες του νέφους μπορεί να διατεθούν αρκετά γρήγορα και ελαστικά, για γρήγορο scale-out όταν υπάρχει αυξημένη ζήτηση πόρων και να αποδεσμευτούν τάχιστα για γρήγορο scale-in όταν η ζήτηση για τους πόρους δεν είναι τόσο υψηλή. Για τον καταναλωτή, οι δυνατότητες που είναι διαθέσιμες για να τις χρησιμοποιήσει ανάλογα με τις ανάγκες του, συχνά φαίνονται ότι είναι απεριόριστες και μπορούν να αγοραστούν ανά πάσα στιγμή σε οποιαδήποτε ποσότητα.
- Μετρούμενη υπηρεσία: Τα συστήματα νέφους διαχειρίζονται αυτόματα και αξιοποιούν με τον καλύτερο τρόπο την χρήση των πόρων. [1]

1.2 Ιστορική Αναδρομή

Η προέλευση του όρου

Η προέλευση του όρου υπολογιστική νέφους - cloud computing είναι ασαφής. Η λέξη νέφος - cloud χρησιμοποιείται ευρέως στη φυσική για να περιγράψει μια μεγάλη συσσώρευση από αντικείμενα που εμφα-

νίζονται οπτικά από απόσταση σαν ένα σύννεφο και περιγράφουν κάθε ομάδα από πράγματα, των οποίων οι λεπτομέρειες δεν έχουν εξεταστεί λεπτομερώς.[2]

Ακόμα μια εξήγηση είναι ότι τα παλιά προγράμματα που σχεδίαζαν σχηματικές αναπαραστάσεις δικτύων, κύκλωναν τα εικονίδια των servers με έναν κύκλο και μια ομάδα από servers σε ένα δικτυακό διάγραμμα είχε πολλούς επικαλυπτόμενους κύκλους, οι οποίοι έμοιαζαν με νέφος.

Σε αναλογία με την παραπάνω χρήση, η λέξη νέφος έχει χρησιμοποιηθεί σαν μεταφορά για το Διαδίκτυο και ένα προτυποποιημένο σχήμα που μοιάζει με σύννεφο, υποδήλωνε ένα δίκτυο σε σχηματικά της τηλεφωνίας. Αργότερα χρησιμοποιήθηκε για να απεικονίσει το Διαδίκτυο, σε διαγράμματα δικτύων υπολογιστών. Με αυτή την απλούστευση, το νόημα είναι ότι οι λεπτομέρειες της σύνδεσης των τελικών σημείων σε ένα δίκτυο δεν είναι σχετικές με την πρόθεση κατανόησης του διαγράμματος.

Το σύμβολο του νέφους χρησιμοποιήθηκε για να αναπαραστήσει τα δίκτυα του εξοπλισμού των υπολογιστών στο παραδοσιακό ARPANET στις αρχές του 1977 και στο CSNET το 1981, τους δύο προγόνους του Διαδικτύου.

1970s

Κατά την διάρκεια της δεκαετίας του 1960, οι αρχικές ιδέες για χρονικό διαμοιρασμό έγιναν δημοφιλείς, μέσω του Remote Job Entry (RJE). Αυτός ο όρος συνδέθηκε κυρίως με εταιρείες όπως η IBM και η DEC. Οι λύσεις για πλήρη χρονικό διαμοιρασμό έγιναν διαθέσιμες στις αρχές της δεκαετίας του 1970 σε πλατφόρμες όπως η Multics, η Cambridge CTSS, και η νεότερη UNIX ports. Όμως, το μοντέλο του data center, όπου οι χρήστες υποβάλλουν τις εργασίες τους σε παρόχους και εκείνοι με την σειρά τους τις εκτελούν σε IBM mainframes, ήταν εξαιρετικά δημοφιλές.[2]

1990s

Στη δεκαετία του 1990, οι εταιρείες τηλεπικοινωνιών που πρόσεφεραν κυρίως κυκλώματα δεδομένων point-to-point, ξεκίνησαν να παρέχουν υπηρεσίες εικονικών ιδιωτικών δικτύων Virtual Private Network (VPN) με συγκρίσιμη ποιότητα υπηρεσίας, αλλά σε χαμηλότερη τιμή. Αλλάζοντας την κίνηση των δεδομένων, για την εξισορρόπηση της χρήσης του server, μπορούσαν να χρησιμοποιήσουν το συνολικό εύρος ζώνης του δικτύου πιο αποδοτικά. Έτσι ξεκίνησαν να χρησιμοποιούν το σύμβολο του νέφους για δηλώνουν το σημείο διαχωρισμού ανάμεσα στην περιοχή ευθύνης του παρόχου και των χρηστών. Η υπολογιστική νέφους επέκτεινε αυτό το όριο για να καλύψει όλους τους servers αλλά και την δικτυακή υποδομή.[2]

2000s

Από το 2000 γεννήθηκε η υπολογιστική νέφους. Στις αρχές του 2008, το OpenNebula της Nasa, βελτιώθηκε από το χρηματοδοτούμενο πρόγραμμα της Ευρωπαϊκής επιτροπής RESERVOIR και έγινε το πρώτο λογισμικό ανοιχτού κώδικα για την ανάπτυξη ιδιωτικών και υβριδικών clouds. Την ίδια χρονιά, οι προσπάθειες συγκεντρώθηκαν στην παροχή εγγυημένης ποιότητας υπηρεσίας στις υποδομές που ήταν βασισμένες στο νέφος, στο πλαίσιο του χρηματοδοτούμενου προγράμματος της Ευρωπαϊκής επιτροπής IRMOS, έχοντας ως αποτέλεσμα ένα περιβάλλον νέφους πραγματικού χρόνου. [2]

Τον Αύγουστο του 2006 η Amazon παρουσίασε το Elastic Compute Cloud. Το Microsoft Azure ανακοινώθηκε ως Azure τον Οκτώβριο του 2008 και εκδόθηκε τον Φεβρουάριο του 2010 ως Windows Azure, πριν ονομαστεί πάλι σε Microsoft Azure στις 25 Μαρτίου 2014. Για καιρό, το Azure ήταν στην λίστα με τους 500 καλύτερους υπερυπολογιστές, μέχρι να πέσει στην κατάταξη. [2]

Τον Ιούλιο του 2010, η Rackspace Hosting και η NASA, παρουσίασαν από κοινού ένα λογισμικό νέφους ανοιχτού κώδικα, το γνωστό σε όλους OpenStack. Το πρόγραμμα OpenStack σκόπευε να βοηθήσει τους οργανισμούς που προσέφεραν υπηρεσίες υπολογιστικής νέφους και έτρεχαν σε συμβατικό hardware. Ο αρχικός κώδικας ήρθε από την πλατφόρμα Nebula της Nasa, όπως επίσης και από την πλατφόρμα cloud αρχείων της Rackspace. [2]

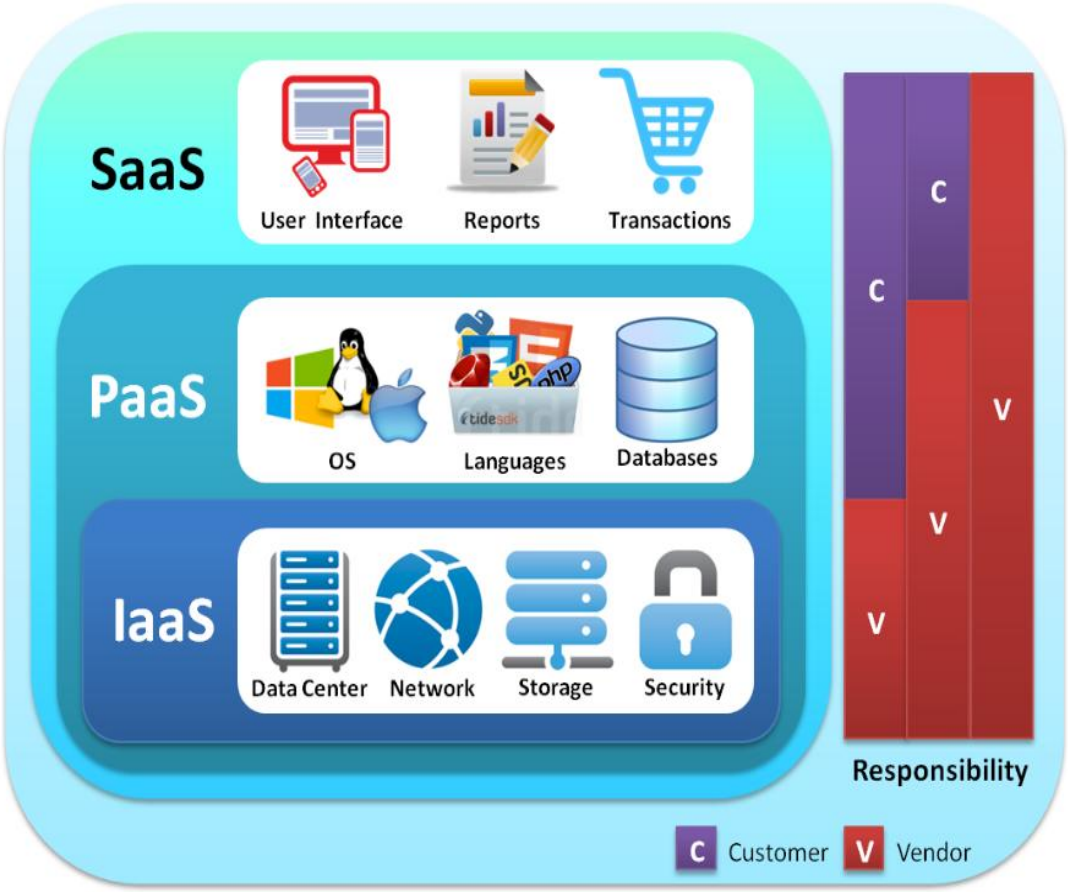
Την 1η Μαρτίου 2011, η IBM ανακοίνωσε το IBM SmartCloud framework για την υποστήριξη της στρατηγικής της που ονόμασε Smarter Planet. Ανάμεσα στα διαφορετικά στοιχεία του Smarter Computing, η υπολογιστική νέφους έπαιζε σημαντικό ρόλο. [2]

Την 7η Ιουλίου 2012, η Oracle ανακοίνωσε το Oracle CLoud. Ενώ πολλά στοιχεία του βρίσκονται ακόμα σε ανάπτυξη, αυτή η πρόταση cloud είναι η πρώτη που παρέχει πρόσβαση στους χρήστες σε μια

ενοποιημένη ομάδα IT λύσεων, συμπεριλαμβανομένων των επιπέδων Εφαρμογής (SaaS), Πλατφόρμας (PaaS) και Υποδομής (IaaS).[2]

1.3 Μοντέλα υπηρεσιών

Πρόσφατα, η υπολογιστική νέφους εμφανίζεται ως η κορυφαία τεχνολογία για την παράδοση αξιόπιστων, ασφαλών, ανεχτικών σε λάθη, βιώσιμων και επεκτάσιμων υπολογιστικών υπηρεσιών, οι οποίες παρουσιάζονται ως Software, Infrastructure ή Platform as services (SaaS, IaaS, PaaS). Επιπλέον, αυτές οι υπηρεσίες μπορεί να παρέχονται σε ιδιωτικά data centers (ιδιωτικά clouds), να είναι εμπορικά διαθέσιμες για πελάτες (δημόσια clouds) ή ακόμα να είναι δημόσια και ιδιωτικά clouds ενοποιημένα σε υβριδικά clouds. Στην πραγματικότητα υπάρχουν περισσότερα μοντέλα υπηρεσιών, από τα τρία που χρησιμοποιούνται ευρέως. Τα μοντέλα υπηρεσιών όπως η Αναλυτική Δεδομένων και το Service και HPC/Grid as a Service, αρχίζουν και γίνονται πολύ χρήσιμα. Πόσο συχνά επιλέγει κάποιος το κατάλληλο μοντέλο υπηρεσίας, εξαρτάται από τους παράγοντες όπως η διαθεσιμότητα των κατάλληλων εφαρμογών που χρειάζονται για το περιβάλλον ανάπτυξης και επαλήθευσης, η ανάγκη για αποτελεσματικό έλεγχο της υπολογιστικής υποδομής και η διαχείριση που απαιτείται για την κατανομή των δεδομένων.



Εικόνα 1: Τα μοντέλα υπηρεσιών της Υπολογιστικής Νέφους [3]

Cloud Software as a Service (SaaS). Είναι η δυνατότητα που παρέχεται στους καταναλωτές να χρησιμοποιούν τις εφαρμογές του παρόχου υπηρεσιών, που τρέχουν σε υποδομές νέφους. Οι εφαρμογές είναι προσβάσιμες από διαφορετικές συσκευές, μέσω μιας thin client διεπαφής, όπως ένας web browser (π.χ. web-based email). Ο πελάτης δεν διαχειρίζεται ούτε ελέγχει την υποδομή του νέφους, συμπεριλαμβανομένων του δικτύου, των εξυπηρετητών, των λειτουργικών συστημάτων, του χώρου αποθήκευσης ή ακόμα και τις ατομικές δυνατότητες της εφαρμογής, με δυνατότητα εξαίρεσης κάποιων περιορισμένων χαρακτηριστικών και ρυθμίσεων της εφαρμογής.

Cloud Platform as a Service (PaaS). Η δυνατότητα που παρέχει το PaaS στους πελάτες, είναι η ανάπτυξη στην υποδομή του νέφους, εφαρμογών που έχουν υλοποιηθεί ή αποκτηθεί από τους πελάτες,

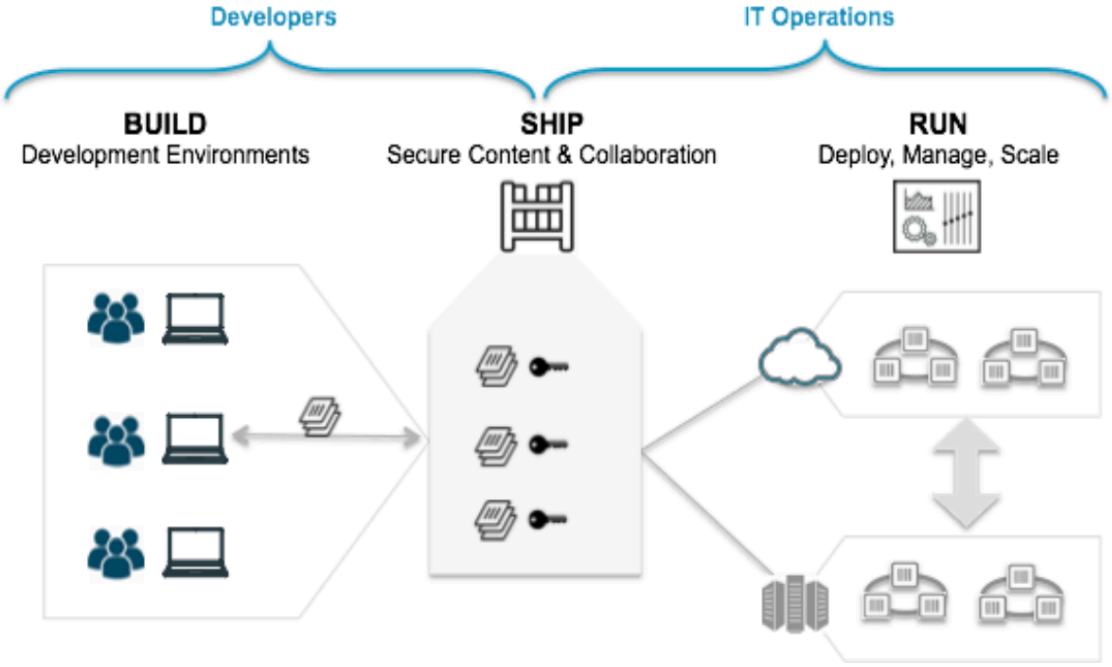
χρησιμοποιώντας γλώσσες προγραμματισμού και εργαλεία που υποστηρίζονται από τον πάροχο του νέφους. Παρομοίως και εδώ ο πελάτης δεν διαχειρίζεται ούτε ελέγχει την υποδομή του νέφους, συμπεριλαμβανομένων του δικτύου, των εξυπηρετητών, των λειτουργικών συστημάτων ή του χώρου αποθήκευσης, αλλά έχει τον έλεγχο των ανεπτυγμένων εφαρμογών και πιθανώς των ρυθμίσεων για το περιβάλλον που φιλοξενεί τις εφαρμογές.

Cloud Infrastructure as a Service (IaaS). Στο IaaS παρέχεται η δυνατότητα στους πελάτες για παροχή επεξεργασίας, αποθήκευσης, δικτύων και άλλων θεμελιωδών υπολογιστικών πόρων, όπου ο πελάτης μπορεί να αναπτύξει και να εκτελέσει αυθαίρετο λογισμικό, το οποίο μπορεί να περιλαμβάνει λειτουργικά συστήματα ή εφαρμογές. Ο πελάτης δεν διαχειρίζεται ούτε ελέγχει την υποδομή του νέφους, αλλά έχει τον έλεγχο πάνω στο λειτουργικό σύστημα, στην αποθήκευση, στις ανεπτυγμένες εφαρμογές και πιθανώς περιορισμένο έλεγχο σε επιλεγμένα στοιχεία του δικτύου (π.χ. firewalls).

Το μέλλον των μοντέλων υπηρεσιών

Containers as a Service (CaaS). Σε αυτό το μοντέλο υπηρεσιών οι προγραμματιστές και το τμήμα IT των οργανισμών δουλεύουν μαζί, για την ανάπτυξη, διανομή και εκτέλεση των εφαρμογών τους, οπουδήποτε. Το CaaS δημιουργεί, από πλευράς IT, ένα ασφαλές και διαχειρίσιμο περιβάλλον εφαρμογών που αποτελούνται από περιεχόμενο και υποδομή, στο οποίο οι προγραμματιστές είναι σε θέση να φτιάξουν και να αναπτύξουν τις εφαρμογές τους με έναν τρόπο αυτοεξυπηρέτησης. [4] Στην καρδιά του συστήματος βρίσκεται μια πλατφόρμα που οργανώνει τους containers, η οποία έχει σχεδιαστεί για να διαχειρίζεται τις λειτουργίες τους, όπως είναι η ανάπτυξη των containers και η διαχείριση του cluster. [5]

Η εικόνα 2 δείχνει το διάγραμμα ροής του μοντέλου Containers as a Service. [4]



Εικόνα 2: Τα μοντέλο υπηρεσιών CaaS [4]

Οι προγραμματιστές, όπως φαίνονται αριστερά της εικόνας, τραβούν και στέλνουν το περιεχόμενο των εφαρμογών, από μια βιβλιοθήκη με αξιόπιστες πηγές. Οι ομάδες λειτουργιών, που βρίσκονται δεξιά της εικόνας, παρακολουθούν και διαχειρίζονται τις ανεπτυγμένες εφαρμογές και την υποδομή. Οι δύο ομάδες συνεργάζονται μέσω ενός εργαλείου, το οποίο επιτρέπει των διαχωρισμό των ανησυχιών, ενώ παράλληλα ενοποιεί τις ομάδες μέσα από τον κύκλο ζωής της εφαρμογής. Το γενικό διάγραμμα ροής μπορεί να τροποποιηθεί, για μεγαλύτερο κεντρικό έλεγχο ή για την αποκέντρωση των αρχείων και της διαχείρισης σε κάθε ομάδα εφαρμογής. [4]

Οι πάροχοι του μοντέλου CaaS μπορούν να χρησιμοποιήσουν μια ποικιλία από πλατφόρμες, όπως οι Google Kubernetes, Docker Machine, Docker Swarm, Apache Mesos, το fleet από CoreOS και το nova-docker για τους χρήστες του OpenStack. [5]

1.4 Μοντέλα Ανάπτυξης

Τα μοντέλα ανάπτυξης του νέφους αντιπροσωπεύουν την ακριβή κατηγορία του περιβάλλοντος νέφους και διαχωρίζονται από την ιδιοκτησία, το μέγεθος και την πρόσβαση σε αυτό. Αναφέρονται στο σκοπό και την κύρια χρήση του νέφους. Οι περισσότεροι οργανισμοί είναι πρόθυμοι να αναπτύξουν μια υπηρεσία υπολογιστικού νέφους, επειδή μειώνει τις κεφαλαιακές δαπάνες και τα λειτουργικά έξοδα. Κάποιος που θέλει να αναπτύξει μια εφαρμογή, πρέπει να γνωρίζει τις απαιτήσεις της εφαρμογής του, έτσι ώστε να επιλέξει μεταξύ των μοντέλων ανάπτυξης που παρουσιάζονται παρακάτω. [6]

Private cloud: Είναι γνωστό και ως εσωτερικό cloud. Η πλατφόρμα για την υπολογιστική νέφους αναπτύσσεται σε ένα ασφαλές περιβάλλον βασισμένο στο νέφος, το οποίο βρίσκεται πίσω από ένα τείχος προστασίας, που είναι υπό την επίβλεψη του τμήματος IT του εκάστοτε οργανισμού. Καθώς το συγκεκριμένο μοντέλο επιτρέπει μόνο εξουσιοδοτημένους χρήστες, παρέχει στον οργανισμό μεγαλύτερο και άμεσο έλεγχο στα δεδομένα. Είναι δύσκολο να καθοριστεί τι ακριβώς συνιστά ένα private νέφους, διότι είναι χωρισμένο αναλόγως με τις υπηρεσίες και υπάρχουν αρκετές παραλλαγές. Τα εμπόδια σχετικά με την ασφάλεια μπορούν να αποφευχθούν, αλλά σε περίπτωση φυσικής καταστροφής και εσωτερικής κλοπής των δεδομένων, τότε μπορεί να είναι επιρρεπές σε ευπάθειες. Μπορεί να διαχειρίζεται από τον οργανισμό ή από κάποια τρίτη εταιρεία και μπορεί να βρίσκεται στις εγκαταστάσεις του ή σε κάποιο άλλο κτίριο. [6]

Community cloud: Η υποδομή του νέφους διαμοιράζεται ανάμεσα σε πολλούς οργανισμούς που ανήκουν σε μια συγκεκριμένη κοινότητα, όπως για παράδειγμα τράπεζες και εταιρείες ανταλλαγών. Είναι μια εγκατάσταση πολλαπλών μισθωτών (tenants), που μοιράζεται ανάμεσα σε πολλούς οργανισμούς, οι οποίοι ανήκουν σε μια ομάδα με παρόμοιες υπολογιστικές αντιλήψεις. Τα μέλη της κοινότητας μοιράζονται την ίδια ιδιωτικότητα, απόδοση και ενδιαφέρον όσον αφορά την ασφάλεια. Ο κύριος σκοπός αυτών των κοινοτήτων είναι να επιτύχουν οι επιχειρήσεις τους σχετικούς στόχους τους. Ένα community cloud μπορεί να διαχειρίζεται εσωτερικά ή από κάποιον τρίτο πάροχο. Επίσης, μπορεί να φιλοξενείται εσωτερικά ή εξωτερικά. Το κόστος μοιράζεται από συγκεκριμένους οργανισμούς, μέσα στην κοινότητα, συνεπώς με την χρήση του μειώνονται τα λειτουργικά έξοδα. Είναι κατάλληλο για οργανισμούς και επιχειρήσεις που εργάζονται σε κοινοπραξίες ή στην έρευνα και χρειάζονται μια κεντρική υπολογιστική νέφους, με την δυνατότητα για διαχείριση, χτίσιμο και ανάπτυξη παρόμοιων μελετών. [6]

Public cloud: Είναι ένας τύπος νέφους, στον οποίο οι υπηρεσίες νέφους διανέμονται μέσω ενός δικτύου, ανοιχτό στο ευρύ κοινό. Αυτό το μοντέλο είναι μια πραγματική αναπαράσταση της φιλοξενίας νέφους, κατά την οποία ο πάροχος υπηρεσιών δίνει τις υπηρεσίες και την υποδομή σε διάφορους χρήστες. Οι χρήστες δεν διακρίνονται ούτε έχουν έλεγχο στην τοποθεσία της υποδομής. Καθαρά από τεχνικής άποψης, δεν υπάρχει διαφορά ή μπορεί να υπάρχει ελάχιστη, στον δομικό σχεδιασμό ανάμεσα στα private και τα public clouds, εκτός από το επίπεδο της ασφάλειας που προσφέρεται για τις διάφορες υπηρεσίες, που δίνονται στους χρήστες των public clouds από τους παρόχους υπηρεσιών.

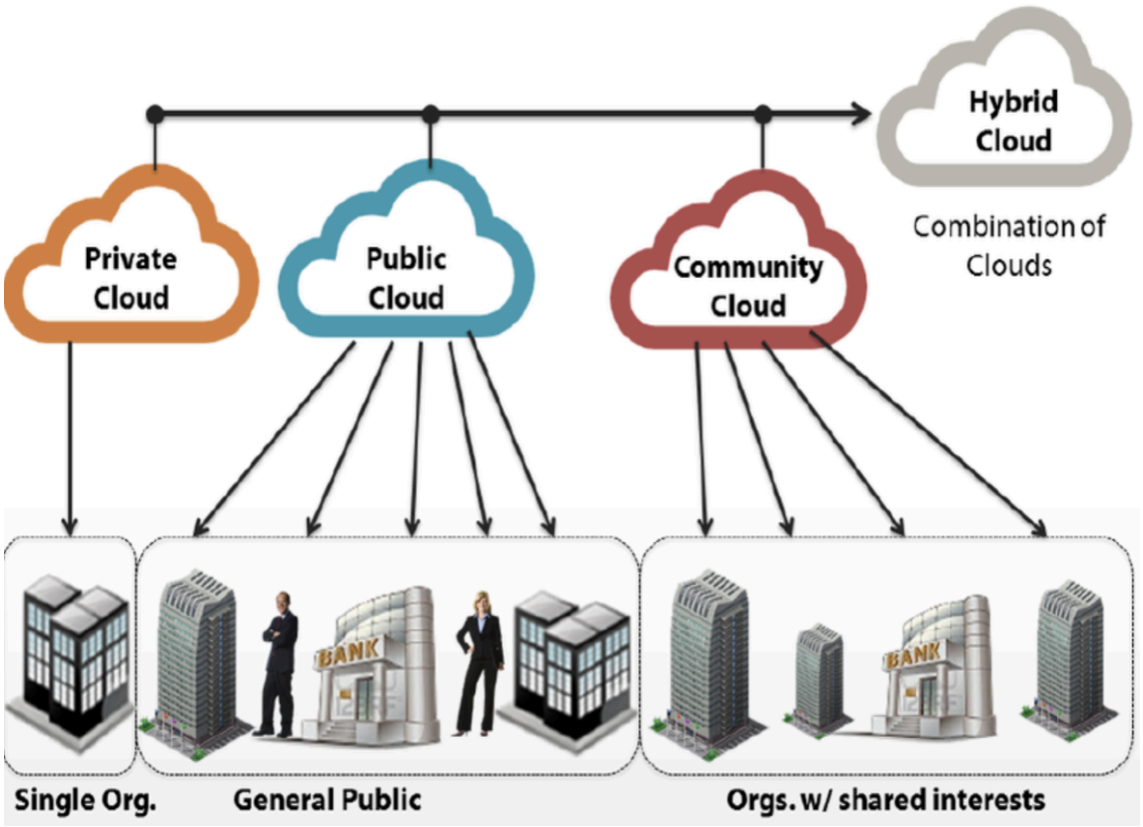
Το public cloud ταιριάζει καλύτερα σε επιχειρησιακές απαιτήσεις, οι οποίες απαιτούν διαχείριση του φόρτου, φιλοξενία εφαρμογών βασισμένη σε SaaS και διαχείριση εφαρμογών, που χρησιμοποιούν πολλοί χρήστες. Λόγω των μειωμένων κεφαλαιακών δαπανών και λειτουργικών εξόδων, αυτό το μοντέλο είναι οικονομικό. Ο πάροχος μπορεί να παρέχει την υπηρεσία δωρεάν ή με την μορφή αδειοδότησης, όπως με πληρωμή ανά χρήστη. Το κόστος μοιράζεται ανάμεσα στους χρήστες, και ως εκ τούτου συμφέρει περισσότερο τους πελάτες, αφού αυξάνουν την παραγωγικότητα και μειώνουν τα έξοδα. Ένα παράδειγμα public cloud, είναι οι υπηρεσίες που προσφέρει η εταιρεία Google. [6]

Hybrid cloud: Αυτή η υποδομή υπολογιστικού νέφους είναι σύνθεση δύο ή περισσότερων μοντέλων ανάπτυξης (private, community ή public), που παραμένουν ξεχωριστές οντότητες αλλά δεμένες μεταξύ τους. Σε μια τέτοια επιλογή νέφους, υπάρχουν πολλά πλεονεκτήματα καθώς υπάρχουν και πολλά μοντέλα ανάπτυξης. Ένα υβριδικό cloud μπορεί να ξεπεράσει την απομόνωση και τα εμπόδια από τον πάροχο, συνεπώς δεν μπορεί να κατηγοριοποιηθεί σε δημόσιο, ιδιωτικό και κοινοτικό νέφος. Επιτρέπει στον χρήστη να αυξήσει την χωρητικότητα και τις δυνατότητες κάνοντας χρήση ενός άλλου πακέτου ή υπηρεσίας νέφους. Σε ένα υβριδικό cloud, οι πόροι διαχειρίζονται και παρέχονται, είτε από εσωτερικούς είτε από εξωτερικούς παρόχους. Είναι μια προσαρμογή ανάμεσα σε δύο πλατφόρμες, στις οποίες ο φόρτος εργασίας ανταλλάσσεται μεταξύ ενός private και ενός public cloud ανάλογα με την χρήση και την ζήτηση.

Οι πόροι που δεν είναι σημαντικοί, όπως η ανάπτυξη και η δοκιμή του φόρτου εργασίας, μπορεί να φιλοξενηθούν σε δημόσια υπολογιστικά νέφη τα οποία ανήκουν σε τρίτους παρόχους. Αν ο φόρτος εργασίας είναι ζωτικής σημασίας ή ευαίσθητος τότε πρέπει να φιλοξενείται εσωτερικά. Οι επιχειρήσεις που δίνουν περισσότερη σημασία στην ασφάλεια και ζήτηση για τη μοναδική τους παρουσία, μπορούν αν αναπτύξουν υβριδικά cloud ως μια αποτελεσματική επιχειρηματική στρατηγική. Όταν αντιμετωπίζουν αιχμές στη ζήτηση, τότε οι πρόσθετοι πόροι που απαιτούνται από μια συγκεκριμένη εφαρμογή, μπορούν να αποκτηθούν

από το δημόσιο cloud. Αυτό ορίζεται ως cloud bursting και είναι διαθέσιμο σε ένα υβριδικό cloud. Οι οργανισμοί μπορούν να χρησιμοποιήσουν ένα τέτοιο μοντέλο ανάπτυξης για την επεξεργασία μεγάλων δεδομένων (big data). [6]

Η φιλοξενία σε ένα υβριδικό cloud ενεργοποιεί χαρακτηριστικά όπως επεκτασιμότητα, ελαστικότητα και ασφάλεια. Εάν κάποιος είναι έτοιμος να παραβλέψει κάποιες προκλήσεις όπως η ασυμβατότητα στις διεπαφές των εφαρμογών, τα θέματα στην δικτυακή σύνδεση και τα κεφαλαιακά έξοδα, τότε το υβριδικό cloud είναι η κατάλληλη επιλογή.



Εικόνα 3: Τα μοντέλα ανάπτυξης της Υπολογιστικής Νέφους. [7]

1.5 Χρήσεις της Υπολογιστικής Νέφους

Στις μέρες μας πολύ κόσμος κάνει χρήση της υπολογιστικής νέφους, αλλά δεν το γνωρίζει. Εάν κάνει χρήση online υπηρεσιών για να στείλει email, να μορφοποιήσει κείμενα, να δει ταινίες ή τηλεόραση, να ακούσει μουσική, να παίξει παιχνίδια ή να αποθηκεύσει φωτογραφίες και άλλα αρχεία είναι πιθανόν μια υπηρεσία που στηρίζεται στην υπολογιστική νέφους να δουλεύει στα παρασκήνια. Οι πρώτες υπηρεσίες υπολογιστικής νέφους υπάρχουν μόλις μια δεκαετία, αλλά ήδη μια πληθώρα οργανισμών, από μικρές start-ups έως παγκόσμιοι οργανισμοί, κυβερνητικές υπηρεσίες και μη κυβερνητικοί οργανισμοί, αποδέχονται την τεχνολογία για πολλούς και διάφορους λόγους. Ακολουθούν μερικά παραδείγματα, που μπορεί κάποιος να κάνει χρησιμοποιώντας μια υπηρεσία νέφους. [8]

- Δημιουργία νέων εφαρμογών και υπηρεσιών.
- Αποθήκευση, δημιουργία αντιγράφων και ανάκτηση δεδομένων.
- Φιλοξενία ιστοσελίδων και blogs.
- Μετάδοση ήχου και βίντεο.
- Διανομή λογισμικού κατά παραγγελία
- Ανάλυση δεδομένων για μοτίβα και δημιουργία προβλέψεων.

Εικόνα 4: Γνωστές εταιρείες που χρησιμοποιούν νέφος

1.6 Πλεονεκτήματα - Μειονεκτήματα

Καθώς η υπολογιστική νέφους αναπτύσσεται, κάποιος που το χρησιμοποιεί πρέπει να γνωρίζει τα πλεονεκτήματα και τα μειονεκτήματα που το συνοδεύουν. Παρακάτω παρουσιάζονται τα κύρια οφέλη για τις επιχειρήσεις και τους χρήστες. [9]

Πλεονεκτήματα

1. **Οικονομική απόδοση**
   Η υπολογιστική νέφους είναι, όσον αφορά τα οικονομικά, πιθανόν η πιο αποδοτική μεθοδολογία για χρήση, συντήρηση και αναβάθμιση. Το παραδοσιακό λογισμικό των υπολογιστών, κοστίζει στις επιχειρήσεις αρκετά χρήματα. Εάν προστεθούν και τα τέλη αδειοδότησης για τους χρήστες, μπορεί να αποδειχθούν πολύ ακριβές λύσεις. Το νέφος, από την άλλη μεριά, είναι διαθέσιμο σε πολύ χαμηλότερες τιμές και για αυτό το λόγο μπορεί να μειώσει τις δαπάνες του IT τμήματος μιας εταιρείας. Εκτός από αυτό, υπάρχουν πολλές ευκολίες πληρωμής, όπως η πληρωμή μιας φοράς, η πληρωμή καθώς συνεχίζει ο χρήστης να χρησιμοποιεί τις υπηρεσίες κ.α.
2. **Απεριόριστος αποθηκευτικός χώρος**
   Η αποθήκευση των πληροφοριών στο νέφος δίνει την δυνατότητα στους χρήστες για σχεδόν απεριόριστη χωρητικότητα αποθήκευσης.
3. **Αντιγραφή και επαναφορά**
   Από την στιγμή που όλα τα δεδομένα βρίσκονται στο νέφος, η αντιγραφή και η επαναφορά τους είναι αρκετά ευκολότερη από την ίδια διαδικασία που ακολουθείται σε μια φυσική συσκευή. Ακόμα, πολλοί πάροχοι υπηρεσιών νέφους είναι συνήθως ικανοί να διαχειριστούν την ανάκτηση των πληροφοριών εάν αυτές χαθούν. Έτσι, η διαδικασία της αντιγραφής και της επαναφοράς των δεδομένων γίνεται ευκολότερη από τις παραδοσιακές μεθόδους αποθήκευσης των δεδομένων.
4. **Αυτόματη ενοποίηση λογισμικού**

Στο νέφος, η ενοποίηση του λογισμικού είναι κάτι που συνήθως συμβαίνει αυτόματα. Αυτό σημαίνει ότι οι χρήστες δεν χρειάζεται να κάνουν επιπλέον ενέργειες για να εξατομικεύσουν και να ενοποιήσουν τις εφαρμογές τους με τις προτιμήσεις τους. Αυτό γίνεται συνήθως από μόνο του.

5. **Εύκολη πρόσβαση στην πληροφορία**
   Μόλις οι χρήστες εγγράφονται σε μια υπηρεσία του νέφους, μπορούν να έχουν πρόσβαση στις πληροφορίες από οπουδήποτε, αρκεί να έχουν σύνδεση στο διαδίκτυο. Αυτό το βολικό χαρακτηριστικό επιτρέπει στους χρήστες να ξεπεράσουν τα προβλήματα ζώνης ώρας και γεωγραφικής θέσης.
6. **Γρήγορη ανάπτυξη**
   Η υπολογιστική νέφους δίνει την δυνατότητα της γρήγορης ανάπτυξης. Καθώς ο χρήστης διαλέγει τη μέθοδο λειτουργίας, ολόκληρο το σύστημα μπορεί να είναι πλήρως λειτουργικό σε διάστημα μόλις λίγων λεπτών. Φυσικά, ο χρόνος που απαιτείται για την ανάπτυξη, εξαρτάται από το είδος της τεχνολογίας που χρειάζεται για τις ανάγκες του χρήστη ή της επιχείρησης.
7. **Εύκολη επέκταση υπηρεσιών**
   Γίνεται πιο εύκολο για τις επιχειρήσεις να επεκτείνουν την υπηρεσία τους, ανάλογα με την ζήτηση από τους πελάτες.
8. **Νέες υπηρεσίες**
   Είναι πιθανόν να δημιουργηθούν νέοι τύποι εφαρμογών και νέες υπηρεσίες που θα αλληλεπιδρούν με τους χρήστες, στο άμεσο μέλλον.

Παρόλο που οι εφαρμογές του έχουν πολλά πλεονεκτήματα, όπως αναφέρθηκαν παραπάνω, η υπολογιστική νέφους έχει και αρκετά σημαντικά μειονεκτήματα. Οι επιχειρήσεις, ειδικά οι μικρότερες, πρέπει να είναι ενήμερες για αυτά πριν χρησιμοποιήσουν μια τέτοια τεχνολογία. Τα κυριότερα μειονεκτήματα που υπάρχουν είναι τα παρακάτω. [9]

Μειονεκτήματα

1. **Τεχνικά θέματα**
   Αν και είναι αληθές ότι οι πληροφορίες και τα δεδομένα στο νέφος, μπορεί να είναι προσβάσιμα από οπουδήποτε, υπάρχουν στιγμές που το σύστημα μπορεί να έχει σοβαρές δυσλειτουργίες. Οι επιχειρήσεις πρέπει να είναι ενήμερες για το γεγονός ότι αυτή η τεχνολογία είναι επιρρεπής σε διακοπές λειτουργίας και άλλα τεχνικά θέματα. Ακόμα και οι καλύτεροι πάροχοι υπηρεσιών νέφους, αντιμετωπίζουν αυτού του είδους τα προβλήματα, παρόλο που έχουν υψηλές προδιαγραφές στη συντήρηση. Χαρακτηριστικό παράδειγμα είναι η διακοπή λειτουργίας της υπηρεσίας νέφους της Amazon στις 28 Φεβρουαρίου 2017, που αρκετές σελίδες δεν ήταν προσβάσιμες. [10]
2. **Ασφάλεια**
   Ακόμα ένα σημαντικό θέμα του νέφους, βρίσκεται στην ασφάλεια. Πριν από την χρήση αυτής της τεχνολογίας, οι χρήστες πρέπει να ξέρουν ότι παραδίδουν όλα τα ευαίσθητα δεδομένα της εταιρείας σε έναν τρίτο πάροχο υπηρεσιών υπολογιστικής νέφους. Αυτό μπορεί ενδεχομένως να αποτελέσει ρίσκο για την εταιρεία. Έτσι, οι επιχειρήσεις θα πρέπει να σιγουρευτούν ότι επιλέγουν τον πιο αξιόπιστο πάροχο υπηρεσιών, ο οποίος θα κρατήσει τα δεδομένα τους απόλυτα ασφαλή.
3. **Επιρρεπές σε επιθέσεις**
   Η αποθήκευση των δεδομένων στο νέφος, μπορεί να κάνει τις επιχειρήσεις ευάλωτες σε εξωτερικές επιθέσεις και απειλές από hackers. Επιπλέον, υπάρχει πάντα η πιθανότητα κάποιος να παραμονεύει για να δει τις ευαίσθητες πληροφορίες.
4. **Πιθανή διακοπή υπηρεσιών**
   Η υπολογιστική νέφους κάνει τις μικρές επιχειρήσεις και τους χρήστες να εξαρτώνται από την αξιοπιστία της διαδικτυακής τους σύνδεσης.
5. **Κόστος**
   Με μια πρώτη ματιά, μια εφαρμογή που φιλοξενείται σε ένα νέφος, μπορεί να είναι αρκετά φθηνότερη από την αντίστοιχη λύση που είναι εγκατεστημένη και τρέχει στους υπολογιστές. Ακόμα, οι εταιρείες πρέπει να βεβαιωθούν ότι οι εφαρμογές νέφους που χρησιμοποιούν, έχουν όλα τα χαρακτηριστικά που έχει το λογισμικό τους και εάν δεν τα έχουν να εντοπίσουν ποια από αυτά που λείπουν είναι σημαντικά για αυτούς. Επίσης, απαιτείται μια συνολική εκτίμηση και σύγκριση του κόστους και η προσεκτική επιλογή των πλάνων τιμολόγησης για την κάθε εφαρμογή.
6. **Δύσκολη μεταφορά υπηρεσιών**
   Η επιλογή ενός παρόχου υπολογιστικής νέφους συχνά σημαίνει ότι δεσμεύει την εταιρεία να χρησιμοποιεί τις δικές του κατοχυρωμένες εφαρμογές ή τύπους αρχείων. Για παράδειγμα δεν είναι εφικτό να ανοίξεις ένα αρχείο κειμένου, που έχει δημιουργηθεί από άλλη εφαρμογή, στα Google Docs.

7. **Έλλειψη υποστήριξης**
   Όταν μια εταιρεία αντιμετωπίζει κάποιο πρόβλημα, μπορεί να περιμένει και έως 48 ώρες για να λυθεί. Στα πλαίσια ενός οργανισμού αυτό αντικατοπτρίζεται σε χρόνο και χρήμα.

Κεφάλαιο 2

Προσομοιωτής CloudSim

> "I don't need a hard disk in my computer if I can get to the server faster… carrying around these non-connected computers is byzantine by comparison."
>
> Steve Jobs, Συνιδρυτής, CEO και πρόεδρος της Apple Inc.

2.1 Εισαγωγή

Το ανεπτυγμένο οικοσύστημα των αρχιτεκτονικών υπολογιστικού νέφους, μαζί με την αυξανόμενη ζήτηση για ενεργειακής απόδοσης IT τεχνολογίες, απαιτεί γρήγορες, επαναλαμβανόμενες και διαχειρίσιμες μεθοδολογίες για αξιολόγηση αλγορίθμων, υπηρεσιών και πολιτικές πριν από την ανάπτυξη cloud προϊόντων. Επειδή η χρησιμοποίηση πραγματικών tests περιορίζει τον πειραματισμό στην κλίμακα του test και κάνει την αναπαραγωγή των αποτελεσμάτων ένα εξαιρετικό δύσκολο εγχείρημα, εναλλακτικές τεχνικές για το testing και τον πειραματισμό φέρνουν στο προσκήνιο την ανάπτυξη καινούργιων τεχνολογιών νέφους. [11]

Μια κατάλληλη εναλλακτική είναι η χρήση εργαλείων προσομοίωσης, με τα οποία ο καθένας μπορεί να προσομοιώσει την υποδομή ενός υπολογιστικού νέφους, πριν από την ανάπτυξη εφαρμογών σε αυτό, όπου κάποιος μπορεί να αναπαράγει ελέγχους και δοκιμές. Συγκεκριμένα στην περίπτωση της υπολογιστικής νέφους, όπου η πρόσβαση στις υποδομές μπορεί να περιλαμβάνει πληρωμές σε πραγματικό νόμισμα, η λογική της προσομοίωσης προσφέρει σημαντικά πλεονεκτήματα, καθώς επιτρέπει στους πελάτες της υπηρεσίας νέφους, να δοκιμάσουν δωρεάν τις υπηρεσίες σε ένα επαναλαμβανόμενο και διαχειρίσιμο περιβάλλον και να διορθώσουν τα προβλήματα στην απόδοση πριν από την ανάπτυξη των υπηρεσιών σε πραγματικά Clouds. Από την μεριά του παρόχου των υπηρεσιών, τα περιβάλλοντα προσομοίωσης επιτρέπουν την αξιολόγηση των διαφόρων ειδών από πόρους, που θα μισθωθούν με διαφορετικές χρεώσεις ανάλογα με τον φόρτο και τις υπηρεσίες. Αυτές οι μελέτες μπορούν να βοηθήσουν τους παρόχους να βελτιστοποιήσουν το κόστος της πρόσβασης στους πόρους, εστιάζοντας στην αύξηση των κερδών. Κατά την απουσία μιας πλατφόρμας προσομοίωσης, οι πελάτες και οι πάροχοι του νέφους πρέπει να βασιστούν είτε σε θεωρητικές και ανακριβείς αξιολογήσεις, είτε σε προσεγγίσεις δοκιμών και βελτίωσης των λαθών που προκύπτουν, που οδηγούν σε ανεπαρκή απόδοση της υπηρεσίας και μειωμένη παραγωγή εσόδων.

Ο πρωταρχικός σκοπός του CloudSim είναι να παρέχει ένα γενικό και επεκτάσιμο πλαίσιο προσομοίωσης που ενεργοποιεί την απρόσκοπτη μοντελοποίηση, προσομοίωση και τον πειραματισμό των αναπτυσσόμενων υποδομών της Υπολογιστικής Νέφους και των υπηρεσιών των εφαρμογών. Με την χρήση του CloudSim, οι ερευνητές και οι προγραμματιστές μπορούν να εστιάσουν σε συγκεκριμένα προβλήματα σχεδίασης του συστήματος, που πρέπει να ερευνήσουν, χωρίς να τους απασχολούν οι μικρού επιπέδου λεπτομέρειες σχετικά με τις υποδομές και τις υπηρεσίες της υπολογιστικής νέφους. [11]

Το CloudSim είναι ένα πλαίσιο λογισμικού, που υποστηρίζει αρκετές βασικές λειτουργίες του νέφους, όπως ουρά για έργα/δουλειές, την επεξεργασία των γεγονότων, την δημιουργία cloud οντοτήτων, την επικοινωνία μεταξύ των οντοτήτων, την υλοποίηση των οικονομικών πολιτικών κ.α. Διανέμεται ως λογισμικό

ανοιχτού κώδικα κάτω από την άδεια Apache Version 2.0 και αναπτύσσεται από ερευνητές του Πανεπιστημίου της Μελβούρνης. [11]

2.1.1 Κύρια χαρακτηριστικά

Τα κύρια χαρακτηριστικά του εργαλείου προσομοίωσης CloudSim, φαίνονται παρακάτω. [11]

- Υποστήριξη για μοντελοποίηση και προσομοίωση μεγάλης κλίμακας data centers υπολογιστικών νεφών.
- Υποστήριξη για μοντελοποίηση και προσομοίωση εικονικών server hosts, με παραμετροποιήσιμες πολιτικές για ανάθεση των πόρων σε εικονικές μηχανές.
- Υποστήριξη για μοντελοποίηση και προσομοίωση containers εφαρμογών.
- Υποστήριξη για μοντελοποίηση και προσομοίωση ενεργειακά αποδοτικών υπολογιστικών πόρων.
- Υποστήριξη για μοντελοποίηση και προσομοίωση τοπολογιών δικτύου των data centers και εφαρμογών message-passing.
- Υποστήριξη για μοντελοποίηση και προσομοίωση για ομόσπονδα clouds.
- Υποστήριξη για δυναμική ανάθεση των στοιχείων προσομοίωσης, στάση και συνέχιση της προσομοίωσης.
- Υποστήριξη για πολιτικές που έχουν οριστεί από τους χρήστες, για κατανομή των hosts σε εικονικές μηχανές και πολιτικές για κατανομή των πόρων σε εικονικές μηχανές.

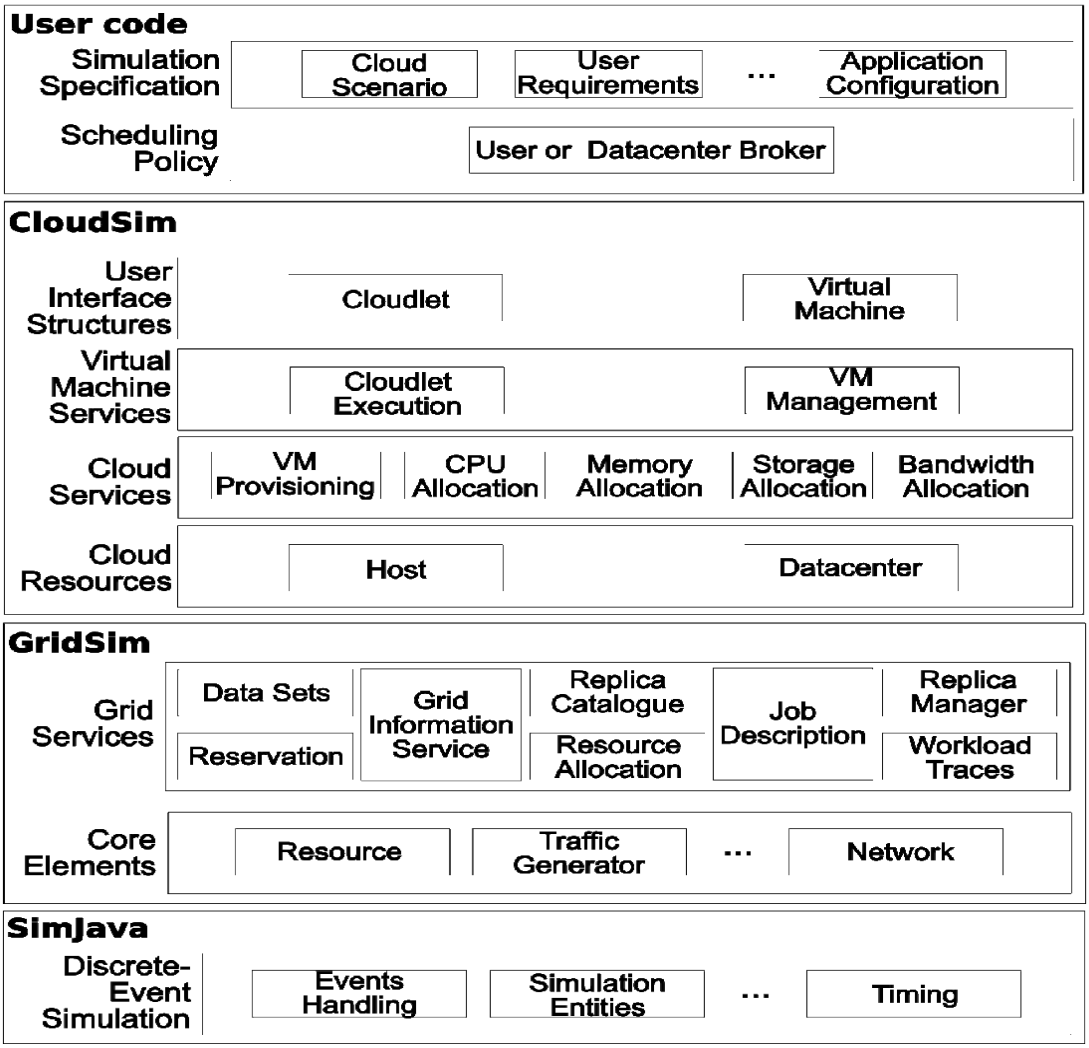
2.2 Η αρχιτεκτονική του CloudSim

Η εικόνα 5 δείχνει την ανάπτυξη σε στρώματα της εφαρμογής CloudSim και την αρχιτεκτονική της.

Στο κατώτερο επίπεδο είναι η SimJava, μια μηχανή προσομοίωσης για διακριτά γεγονότα, που υλοποιεί τις κύριες λειτουργίες που απαιτούνται για τα ανώτερα επίπεδα του πλαισίου της προσομοίωσης. Αυτές οι λειτουργίες είναι οι ουρές και η επεξεργασία των γεγονότων, η δημιουργία των στοιχείων του συστήματος (hosts υπηρεσιών, data center, broker, εικονικές μηχανές), η επικοινωνία μεταξύ των στοιχείων και η διαχείριση του ρολογιού της προσομοίωσης. Έπειτα ακολουθούν οι βιβλιοθήκες που αναπτύσσονται από το πακέτο εργαλείων GridSim που υποστηρίζει: υψηλού επιπέδου στοιχεία λογισμικού για μοντελοποίηση πολλαπλών υποδομών αρχιτεκτονικής Grid, συμπεριλαμβανομένων των δικτύων και των συσχετισμένων προφίλ κίνησης και τα Θεμελιώδη στοιχεία της αρχιτεκτονικής Grid, όπως είναι οι πόροι, οι ομάδες δεδομένων, τα αρχεία καταγραφής του φόρτου εργασίας και οι υπηρεσίες πληροφοριών.

Το CloudSim έχει αναπτυχθεί στο επόμενο επίπεδο, επεκτείνοντας προγραμματιστικά τις κύριες λειτουργίες του επιπέδου GridSim. Παρέχει καινοτόμα υποστήριξη για μοντελοποίηση και προσομοίωση των εικονικών, βασισμένων σε αρχιτεκτονική νέφους, περιβαλλόντων των data centers, όπως είναι η ξεχωριστή διεπαφή διαχείρισης για τις εικονικές μηχανές, την μνήμη, την αποθήκευση και το εύρος ζώνης. Το επίπεδο του CloudSim διαχειρίζεται την δημιουργία και την εκτέλεση των βασικών οντοτήτων (εικονικές μηχανές, hosts, data centers, εφαρμογές) κατά την διάρκεια της περιόδου προσομοίωσης. Αυτό το επίπεδο είναι ικανό, να δημιουργεί ταυτόχρονα και να διαχειρίζεται μια αρκετά μεγάλη υποδομή νέφους, ενός συστήματος που αποτελείται από χιλιάδες στοιχεία. Τα θεμελιώδη στοιχεία, όπως η παροχή hosts σε εικονικές μηχανές, βασισμένες στις απαιτήσεις των χρηστών, η διαχείριση εκτέλεσης των εφαρμογών και η δυναμική παρακολούθηση του συστήματος, διαχειρίζονται από αυτό το επίπεδο. Ένας πάροχος Cloud υπηρεσιών, ο οποίος θέλει να μελετήσει την αποτελεσματικότητα των διάφορων πολιτικών στην κατανομή των hosts, θα χρειαστεί να υλοποιήσει τις στρατηγικές του σε αυτό το επίπεδο, επεκτείνοντας προγραμματιστικά την βασική λειτουργία παροχής των εικονικών μηχανών. Υπάρχει ένας καθαρός διαχωρισμός σε αυτό το επίπεδο, στο πως ένας host κατανέμεται σε διαφορετικές, ανταγωνιστικές εικονικές μηχανές στο νέφος. Ένας host μπορεί να διαμοιραστεί ταυτόχρονα σε έναν αριθμό εικονικών μηχανών, οι οποίες εκτελούν εφαρμογές που βασίζονται στις Quality of Service (QoS) απαιτήσεις που ορίζει ο χρήστης.

Το ανώτερο επίπεδο στην στοίβα της προσομοίωσης είναι το User Code, που αναδεικνύει τις λειτουργίες σχετικά με την ρύθμιση των παραμέτρων για τους hosts (αριθμών των μηχανών, οι προδιαγραφές τους κ.α.), για τις εφαρμογές (ο αριθμός των εργασιών και οι απαιτήσεις του), για τις εικονικές μηχανές, για τον αριθμό των χρηστών και τον τύπο των εφαρμογών και τέλος για τις πολιτικές προγραμματισμού των brokers. Ένας προγραμματιστής εφαρμογών νέφους μπορεί να παράγει: ένα μείγμα από χρήστες που χρειάζονται διανομές και ρυθμίσεις παραμέτρων των εφαρμογών και σενάρια διαθεσιμότητας του νέφους και εκτέλεση ενισχυμένων ελέγχων βασισμένων σε ειδικές ρυθμίσεις, οι οποίες υποστηρίζονται ήδη από το CloudSim.



Εικόνα 5: Η αρχιτεκτονική του CloudSim [12]

Καθώς η υπολογιστική νέφους είναι μια διαρκώς αναπτυσσόμενη ερευνητική περιοχή, υπάρχει σοβαρή έλλειψη από καθορισμένη προτυποποίηση, εργαλεία και μεθόδους που να αντιμετωπίσουν αποτελεσματικά την πολυπλοκότητα της υποδομής και του επιπέδου εφαρμογής. Για αυτό το λόγο στο άμεσο μέλλον θα υπάρξει ένας αριθμός από ερευνητικές μελέτες, τόσο ακαδημαϊκά όσο και στη βιομηχανία με στόχο να οριστούν βασικοί αλγόριθμοι, πολιτικές και συγκριτικές αξιολογήσεις εφαρμογών βασισμένες στα περιεχόμενα της εκτέλεσης. Επεκτείνοντας τις βασικές λειτουργίες, που ήδη παρέχονται από το CloudSim, οι ερευνητές θα είναι σε θέση να εκτελούν ελέγχους βασισμένους σε συγκεκριμένα σενάρια και ρυθμίσεις. Επομένως θα επιτρέπεται η ανάπτυξη καλύτερων και κατάλληλων πρακτικών σε όλες τις κρίσιμες πλευρές που σχετίζονται με την υπολογιστική νέφους. [13]

2.3 Μοντελοποίηση του νέφους

Οι βασικές υπηρεσίες υποδομής υλικού που σχετίζονται με τα Clouds, μοντελοποιούνται από τον προσομοιωτή με ένα data center, εκείνο το στοιχείο που χειρίζεται τις αιτήσεις της υπηρεσίας. Αυτές οι αιτήσεις είναι στοιχεία εφαρμογών περιορισμένα μέσα στις εικονικές μηχανές, στις οποίες πρέπει να διατεθεί ένα μερίδιο επεξεργαστικής ισχύος από κάθε host του data center. Με τον όρο επεξεργασία εικονικής μηχανής εννοούμε μια ομάδα από λειτουργίες, που σχετίζονται με τον κύκλο ζωής μιας εικονικής μηχανής: παροχή ενός host σε μια εικονική μηχανή, δημιουργία, καταστροφή και μεταφορά. Ένα data center αποτελείται από μια ομάδα hosts, οι οποίοι είναι υπεύθυνοι για την διαχείριση των εικονικών μηχανών κατά την διάρκεια του κύκλου ζωής τους. Ο host είναι ένα στοιχείο το οποίο αναπαριστά ένα φυσικό υπολογιστικό κόμβο σε ένα

νέφος και του ανατίθεται μια προκαθορισμένη επεξεργαστική δυνατότητα (που εκφράζεται σε εκατομμύρια εντολές το δευτερόλεπτο - MIPS), μνήμη, χώρος αποθήκευσης και πολιτική προγραμματισμού για την ανάθεση των επεξεργαστικών πυρήνων στις εικονικές μηχανές. Ο host αναπτύσσει διεπαφές που μπορούν να υποστηρίξουν μοντελοποίηση και προσομοίωση για κόμβους μονού και διπλού πυρήνα αντίστοιχα. Η ανάθεση των εικονικών μηχανών, που είναι συγκεκριμένες για εφαρμογές, σε hosts μέσα σε ένα Cloud data center, είναι ευθύνη του στοιχείου Virtual Machine Provisioner. Αυτό το στοιχείο εκθέτει έναν αριθμό από εξατομικευμένες μεθόδους για τους ερευνητές, που έχουν ως στόχο την ανάπτυξη καινούργιων πολιτικών παροχής εικονικών μηχανών, βασισμένες στην βελτιστοποίηση. Η προεπιλεγμένη πολιτική υλοποίησης από τον VM Provisioner, είναι η ξεκάθαρη πολιτική που αναθέτει μια εικονική μηχανή σε ένα host με την λογική First-Come-First-Serve (FCFS). Οι παράμετροι του συστήματος, όπως ο αριθμός των απαιτούμενων πυρήνων επεξεργασίας, η μνήμη και ο χώρος αποθήκευσης όπως απαιτούνται από τον χρήστη του Cloud, προδιαγράφουν την λογική για αυτή την πολιτική. Άλλες πιο περίπλοκες πολιτικές μπορούν να γραφτούν από τους ερευνητές, βασισμένες στην υποδομή και στις απαιτήσεις των εφαρμογών. Για κάθε host, η κατανομή των επεξεργαστικών πυρήνων στις εικονικές μηχανές, γίνεται με βάση την κατανομή του host. Η πολιτική λαμβάνει υπόψιν της πόσοι επεξεργαστικοί πυρήνες θα ανατεθούν σε κάθε εικονική μηχανή και πόση από την χωρητικότητα των πυρήνων θα αποδοθεί αποτελεσματικά σε μια εικονική μηχανή. Έτσι είναι δυνατόν να οριστούν συγκεκριμένοι επεξεργαστικοί πυρήνες σε συγκεκριμένες εικονικές μηχανές (πολιτική διαμοιρασμού χώρου) ή να διαμοιραστεί δυναμικά η χωρητικότητα των πυρήνων ανάμεσα στις εικονικές μηχανές (πολιτική διαμοιρασμού χρόνου) και να ανατεθούν πυρήνες σε εικονικές μηχανές ανάλογα τη ζήτηση ή να οριστούν άλλες πολιτικές. [12]

2.4 Μοντελοποίηση της κατανομής των εικονικών μηχανών

Ένα από τα κύρια χαρακτηριστικά που διαφοροποιούν την υποδομή της υπολογιστικής νέφους από το Grid computing, είναι η μαζική ανάπτυξη των τεχνολογιών εικονικοποίησης και εργαλείων. Συνεπώς, σε σχέση με τα Grids, στο Cloud έχουμε ένα επιπλέον επίπεδο (εικονικοποίηση - virtualization), που συμπεριφέρεται σαν ένα περιβάλλον εκτέλεσης και φιλοξενίας για υπηρεσίες εφαρμογών βασισμένες στο Cloud. Συνεπώς, τα παραδοσιακά μοντέλα εφαρμογών που αναθέτουν ατομικά στοιχεία εφαρμογών σε υπολογιστικούς κόμβους, δεν αναπαριστούν με ακρίβεια την υπολογιστική ασάφεια που συνήθως συνδέεται με το Cloud. Για παράδειγμα, ας αναλογιστούμε έναν φυσικό host σε ένα data center που έχει ένα μοναδικό πυρήνα επεξεργασίας και υπάρχει η απαίτηση για ταυτόχρονη δημιουργία δύο εικονικών μηχανών σε αυτόν τον πυρήνα. Αν και στην πράξη υπάρχει απομόνωση ανάμεσα στις συμπεριφορές των δύο εικονικών μηχανών, το σύνολο των πόρων που είναι διαθέσιμοι σε κάθε εικονική μηχανή, περιορίζεται από την συνολική επεξεργαστική ισχύ του host. Αυτός ο κρίσιμος παράγοντας πρέπει να ληφθεί υπόψιν κατά την διάρκεια της διαδικασίας κατανομής, για την αποφυγή δημιουργίας μιας εικονικής μηχανής που απαιτεί παραπάνω επεξεργαστική ισχύ από αυτή που είναι διαθέσιμη στον host, καθώς οι πολλαπλές διεργασίες σε κάθε εικονική μηχανή, μοιράζονται τα κομμάτια χρόνου του ίδιου επεξεργαστικού πυρήνα. [13]

Για να επιτραπεί η προσομοίωση διαφορετικών πολιτικών κάτω από διαφορετικά επίπεδα απομόνωσης της απόδοσης, το CloudSim υποστηρίζει τον προγραμματισμό των εικονικών μηχανών σε δύο επίπεδα: Πρώτα στο επίπεδο του host και έπειτα στο επίπεδο της εικονικής μηχανής. Στο επίπεδο του host, είναι δυνατόν να οριστεί πόση από την συνολική επεξεργαστική ισχύ του κάθε πυρήνα θα ανατεθεί σε κάθε εικονική μηχανή. Στο επίπεδο της εικονικής μηχανής, οι εικονικές μηχανές αναθέτουν ένα συγκεκριμένο ποσοστό της διαθέσιμης επεξεργαστικής ισχύς σε ατομικές διεργασίες, οι οποίες φιλοξενούνται μέσα στην μηχανή εκτέλεσης.

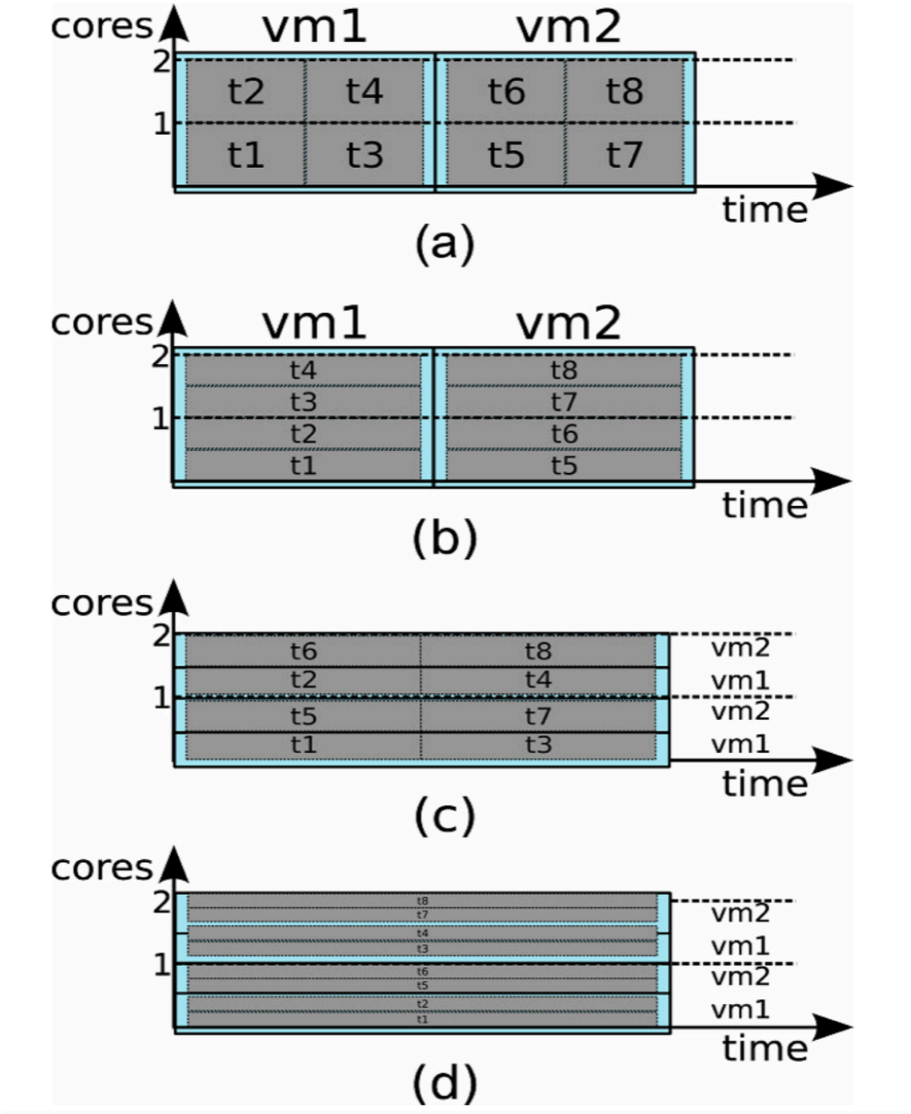
Σε κάθε επίπεδο, το CloudSim εφαρμόζει τις πολιτικές διαμοιρασμού χρόνου και χώρου για την κατανομή των πόρων. Για την ξεκάθαρη απεικόνιση της διαφοράς ανάμεσα σε αυτές τις πολιτικές και στην επίδραση που έχουν στην απόδοση της εφαρμογής, παρουσιάζεται στην εικόνα ένα απλό σενάριο προγραμματισμού. Σε αυτή την εικόνα, ένας host με δύο πυρήνες επεξεργασίας λαμβάνει ένα αίτημα, για να φιλοξενήσει δύο εικονικές μηχανές, όπου η κάθε μια απαιτεί δύο πυρήνες και τρέχει τέσσερις διεργασίες: οι t1, t2, t3 και t4 στην εικονική μηχανή VM1 και οι t5, t6, t7 και t8 στην εικονική μηχανή VM2.

Η εικόνα 6a παρουσιάζει μια πολιτική διαμοιρασμού χώρου για τις δύο εικονικές μηχανές, VM1 και VM2: κάθε εικονική μηχανή χρειάζεται δύο πυρήνες, αλλά μια εικονική μηχανή μπορεί να τρέξει σε μια χρονική στιγμή. Άρα η εικονική μηχανή VM2 μπορεί να χρησιμοποιήσει τον πυρήνα, μόνο όταν η VM1 τελειώσει την εκτέλεση των μονάδων διεργασίας. Το ίδιο συμβαίνει όταν οι διεργασίες φιλοξενούνται μέσα στην εικονική μηχανή, τότε κάθε μονάδα διεργασίας απαιτεί μόνο έναν πυρήνα, δύο από αυτές εκτελούνται ταυτόχρονα και οι άλλες δύο μένουν στην ουρά μέχρι να ολοκληρωθούν οι προηγούμενες διεργασίες.

Στην εικόνα 6b, χρησιμοποιείται η πολιτική διαμοιρασμού χώρου για την κατανομή των εικονικών μηχανών, αλλά χρησιμοποιείται και η πολιτική διαμοιρασμού χρόνου για τον καταμερισμό των ατομικών διεργασιών μέσα στις εικονικές μηχανές. Συνεπώς, κατά την διάρκεια ζωής μιας εικονικής μηχανής, όλες οι διεργασίες που της έχουν ανατεθεί αλλάζουν δυναμικά περιεχόμενο μέχρι την ολοκλήρωση τους. Αυτή η πολιτική καταμερισμού, ενεργοποιεί τις μονάδες διεργασίας ώστε να προγραμματιστούν σε πρωτύτερο χρόνο, αλλά επηρεάζει σημαντικά τον χρόνο ολοκλήρωσης των μονάδων διεργασίας που προηγούνται στην ουρά.

Στην εικόνα 6c, χρησιμοποιείται ο προγραμματισμός διαμοιρασμού χρόνου για τις εικονικές μηχανές και ο διαμοιρασμός χώρου για τις μονάδες έργου. Σε αυτή την περίπτωση, κάθε εικονική μηχανή παίρνει ένα κομμάτι χρόνου για κάθε επεξεργαστικό πυρήνα και έπειτα αυτά τα κομμάτια διανέμονται σε μονάδες έργου με την λογική διαμοιρασμού χώρου. Καθώς ο πυρήνας είναι μοιραζόμενος, το ποσό της επεξεργαστικής ισχύς που είναι διαθέσιμη για την εικονική μηχανή, είναι συγκριτικά μικρότερο από τα προαναφερόμενα σενάρια. Η ανάθεση των μονάδων έργου είναι χωρικά μοιραζόμενη, συνεπώς μόνο ένα έργο μπορεί να κατανεμηθεί σε έναν πυρήνα, ενώ τα άλλα μένουν στην ουρά για μελλοντική χρήση.

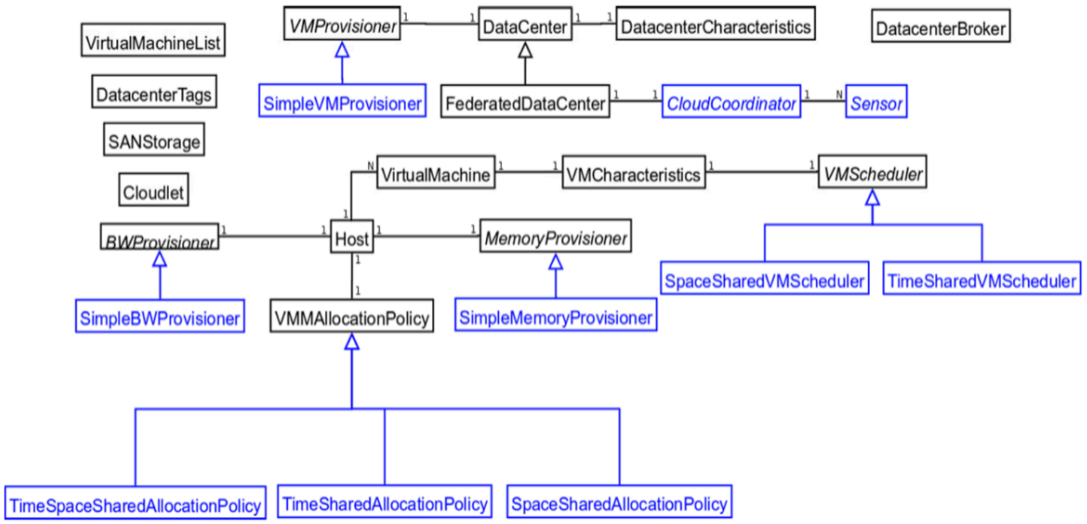
Τέλος, στην εικόνα 6d εφαρμόζεται η χρονική κατανομή για τις εικονικές μηχανές και για τις μονάδες έργου. Έτσι, η επεξεργαστική ισχύς μοιράζεται ταυτόχρονα στις εικονικές μηχανές και τα κομμάτια για κάθε εικονική μηχανή διαχωρίζονται ταυτόχρονα ανάμεσα στις μονάδες έργου που της έχουν ανατεθεί. Σε αυτή την περίπτωση, δεν υπάρχουν ουρές ούτε για τις εικονικές μηχανές ούτε για τις μονάδες έργου. [12]



Εικόνα 6: Εικονικές μηχανές: Διαμοιρασμός χρόνου και χώρου

2.5 Σχεδιασμός και ανάπτυξη του CloudSim

Το διάγραμμα σχεδιασμού κλάσεων για τον προσομοιωτή απεικονίζεται στην εικόνα 7. Σε αυτή την ενότητα, παρέχονται οι λεπτομέρειες που σχετίζονται με τις θεμελιώδεις κλάσεις του CloudSim, οι οποίες είναι τα δομικά στοιχεία του προσομοιωτή. [12]



Εικόνα 7: Διάγραμμα σχεδιασμού κλάσεων [12]

DataCenter: Αυτή η κλάση μοντελοποιεί την κύρια υποδομή επιπέδου υπηρεσιών (υλικό, λογισμικό) που παρέχεται από τους παρόχους σε ένα περιβάλλον Cloud computing. Εμπεριέχει μια ομάδα από υπολογιστικούς hosts, οι οποίοι μπορούν να είναι ομογενείς ή ετερογενείς, όσον αφορά στις ρυθμίσεις παραμέτρων των πόρων (μνήμη, πυρήνες, χωρητικότητα και αποθήκευση). Ακόμα κάθε DataCenter στοιχείο δημιουργεί ένα γενικευμένο στοιχείο παροχής πόρων, που αναπτύσσει μια σειρά από πολιτικές για την κατανομή της μνήμης, του εύρους ζώνης και των συσκευών αποθήκευσης.

DataCenterBroker: Αυτή η κλάση μοντελοποιεί τον broker, ο οποίος είναι υπεύθυνος για την μεσολάβηση ανάμεσα στους χρήστες και στον πάροχο υπηρεσιών, ανάλογα με τις απαιτήσεις QoS των χρηστών και αναπτύσσει εργασίες υπηρεσιών στα Clouds. Ο broker δρα εκ μέρους των χρηστών, αναγνωρίζει τους κατάλληλους παρόχους υπηρεσιών Cloud διαμέσου της Υπηρεσίας Πληροφοριών Cloud - Cloud Information Service (CIS) και διαπραγματεύεται με αυτούς για την κατανομή των πόρων που καλύπτουν τις απαιτήσεις των χρηστών. Οι ερευνητές και οι προγραμματιστές του συστήματος, πρέπει να επεκτείνουν αυτή την κλάση για να διεξάγουν πειράματα με τις δικές τους εξατομικευμένες εφαρμογές.

SANStorage: Αυτή η κλάση αναπαριστά ένα δίκτυο αποθήκευσης, το οποίο είναι ευρέως διαθέσιμο στα data centers που βασίζονται στο Cloud για την αποθήκευση μεγάλων κομματιών δεδομένων. Το SANStorage εφαρμόζει μια απλή διεπαφή, που μπορεί να χρησιμοποιεί για την προσομοίωση της αποθήκευσης και την ανάκτηση κάθε τύπου δεδομένων, σε κάθε χρονική στιγμή ανάλογα την διαθεσιμότητα του εύρους ζώνης του δικτύου. Η πρόσβαση στα αρχεία σε ένα SAN, την ώρα της εκτέλεσης, προκαλεί επιπλέον καθυστερήσεις στην εκτέλεση των μονάδων έργου, λόγω του χρόνου που απαιτείται για την μεταφορά των αρχείων δεδομένων, δια μέσου του εσωτερικού δικτύου του data center.

VirtualMachine: Η κλάση που μοντελοποιεί μια εικονική μηχανή - Virtual Machine (VM), της οποίας η διαχείριση κατά την διάρκεια ζωής της γίνεται από το στοιχείο Host. Όπως προαναφέρθηκε ένας host μπορεί να δημιουργήσει ταυτόχρονα πολλαπλές εικονικές μηχανές και κατανέμει τους πυρήνες με βάση τις προκαθορισμένες πολιτικές διαμοιρασμού της επεξεργασίας. Κάθε εικονική μηχανή έχει πρόσβαση σε ένα στοιχείο που αποθηκεύει τα χαρακτηριστικά που σχετίζονται με την εικονική μηχανή, όπως η μνήμη, ο επεξεργαστής, η αποθήκευση και η εσωτερική πολιτική προγραμματισμού της, η οποία επεκτείνεται από ένα αφηρημένο στοιχείο που καλείται VMScheduling.

Cloudlet: Αυτή η κλάση μοντελοποιεί τις εφαρμογές, που είναι βασισμένες στο Cloud (παράδοση περιεχομένου, κοινωνική δικτύωση, επιχειρηματική ροή εργασιών) και αναπτύσσονται συνήθως στα data

centers. Το CloudSim αναπαριστά την πολυπλοκότητα μιας εφαρμογής σε ό,τι αφορά τις υπολογιστικές της απαιτήσεις. Κάθε στοιχείο της εφαρμογής έχει ένα προκαθορισμό μήκος εντολών και ποσό μεταφοράς δεδομένων το οποίο πρέπει να ληφθεί υπόψιν για την πετυχημένη φιλοξενία της εφαρμογής.

CloudCoordinator: Αυτή η αφηρημένη κλάση, εποπτεύει τη συνολική χωρητικότητα ενός data center. Είναι υπεύθυνη όχι μόνο για την επικοινωνία με άλλες ομότιμες υπηρεσίες CloudCoordinator και Cloud Brokers, αλλά και για την παρακολούθηση της εσωτερικής κατάστασης του data center ενώ έχει ρόλο ζωτικής σημασίας στην απόφαση για την κατανομή του φόρτου - load balancing. Η παρακολούθηση συμβαίνει περιοδικά κατά την διάρκεια του χρόνου προσομοίωσης. Το συγκεκριμένο γεγονός που πυροδοτεί την μεταφορά του φόρτου, εφαρμόζεται από τους χρήστες του CloudSim, μέσω του στοιχείου Sensor. Κάθε sensor μπορεί να μοντελοποιήσει συγκεκριμένη διαδικασία πυροδότησης, η οποία μπορεί να προκαλέσει τον CloudCoordinator να αναλάβει δυναμική μείωση του φόρτου.

BWProvisioner: Είναι μια αφηρημένη κλάση που μοντελοποιεί την πολιτική παροχής του εύρους ζώνης στις εικονικές μηχανές, που είναι ανεπτυγμένες σε έναν host. Η λειτουργία αυτού του στοιχείου είναι η ανάληψη της ανάθεσης του δικτυακού εύρους ζώνης σε μια ομάδα από ανταγωνιστικές εικονικές μηχανές, ανεπτυγμένες σε όλο το data center. Οι προγραμματιστές των συστημάτων Cloud και οι ερευνητές μπορούν να επεκτείνουν αυτή την κλάση με τις δικές τους πολιτικές (προτεραιότητα, QoS) για να εκφράσουν τις ανάγκες των εφαρμογών τους.

MemoryProvisioner: Και αυτή είναι μια αφηρημένη κλάση που αναπαριστά την πολιτική παροχής για την κατανομή της μνήμης στις εικονικές μηχανές. Αυτό το στοιχείο μοντελοποιεί τις πολιτικές για την κατανομή των φυσικών χώρων μνήμης στις ανταγωνιστικές εικονικές μηχανές. Η εκτέλεση και η ανάπτυξη μιας εικονικής μηχανής σε έναν host είναι εφικτή μόνο όταν αποφασίσει το στοιχείο MemoryProvisioner, ότι ο host έχει αρκετή διαθέσιμη ελεύθερη μνήμη, που απαιτείται για την ανάπτυξη μιας εικονικής μηχανής.

VMProvisioner: Αυτή η αφηρημένη κλάση αναπαριστά την πολιτική παροχής που χρησιμοποιεί ένα στοιχείο VM Monitor για την ανάθεση εικονικών μηχανών σε Hosts. Η σημαντικότερη λειτουργία του VMProvisioner είναι η επιλογή διαθέσιμων host σε ένα data center, που πληρούν τις ανάγκες για μνήμη, αποθήκευση και διαθεσιμότητα για ανάπτυξη μιας νέας εικονικής μηχανής. Η προεπιλεγμένη ανάπτυξη SimpleVMProvisioner που παρέχεται στο πακέτο του CloudSim, αναθέτει εικονικές μηχανές στον πρώτο διαθέσιμο Host που πληρεί τις προαναφερθείσες απαιτήσεις. Οι Hosts καθορίζονται με διαδοχική σειρά. Ωστόσο, ακόμα πιο περίπλοκες πολιτικές μπορούν να εφαρμοστούν μέσα σε αυτό το στοιχείο, για την επίτευξη βελτιστοποιημένων αναθέσεων. Για παράδειγμα, η επιλογή των hosts ανάλογα με την δυνατότητα τους να πληρούν τις απαιτήσεις για QoS, όπως είναι ο χρόνος απόκρισης, ο προϋπολογισμός.

VMMAllocationPolicy: Είναι μια αφηρημένη κλάση,που εφαρμόζεται από το στοιχείο του Host και μοντελοποιεί τις πολιτικές (διαμοιρασμού χρόνου ή χώρου) που απαιτούνται για την ανάθεση επεξεργαστικής ισχύς στις εικονικές μηχανές. Οι λειτουργίες αυτής της κλάσης μπορούν εύκολα να παρακαμφθούν για να φιλοξενηθούν εφαρμογές με συγκεκριμένες πολιτικές διαμοιρασμού του επεξεργαστή. [12]

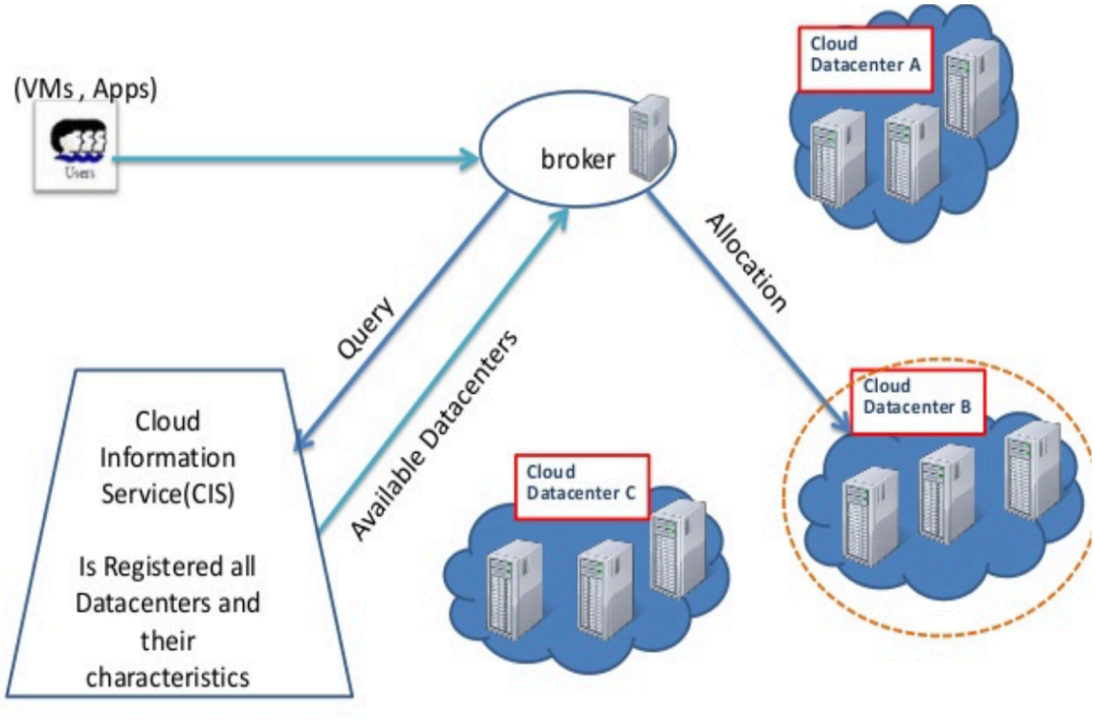
2.6 Οντότητες και threading

Καθώς το CloudSim έχει δομηθεί προγραμματιστικά, πάνω από την μηχανή προσομοίωσης γεγονότων SimJava, διατηρεί το μοντέλο νημάτων της SimJava για την δημιουργία των οντοτήτων προσομοίωσης. Ένα προγραμματιστικό στοιχείο αναφέρεται ως οντότητα, εάν επεκτείνει άμεσα το κύριο στοιχείο Sim Entity της SimJava, το οποίο εφαρμόζει την διεπαφή Runnable. Η διεπαφή αυτή φτιάχνει ένα αφηρημένο αντικείμενο που εκτελεί κώδικα όταν είναι ενεργό, ακριβώς όπως ένα νήμα.

Κάθε οντότητα είναι ικανή να στέλνει και να λαμβάνει μηνύματα, δια μέσου της διαμοιραζόμενης ουράς γεγονότων της SimJava. Η διάδοση των μηνυμάτων (αποστολή και λήψη) γίνεται δια μέσου των θυρών εισόδου-εξόδου, που η SimJava συσχετίζει με κάθε οντότητα στο σύστημα προσομοίωσης. Εφόσον τα νήματα προκαλούν φόρτο σε μνήμη και επεξεργαστή για την ανταλλαγή των περιεχομένων τους, η χρήση ενός μεγάλου αριθμού νημάτων/οντοτήτων σε ένα περιβάλλον προσομοίωσης μπορεί να δημιουργήσει συμφόρηση στην απόδοση, λόγω της μικρής επεκτασιμότητας. Για να αντιμετωπιστεί αυτή η συμπεριφορά, το CloudSim ελαχιστοποιεί των αριθμό των οντοτήτων στο σύστημα, εφαρμόζοντας μόνο τις κυριότερες (Χρήστες και data centers), ως κληρονομημένα μέλη των οντοτήτων της SimJava.

Αυτή η σχεδιαστική απόφαση είναι αρκετά σημαντική και βοηθάει το CloudSim στην προσομοίωση ενός αρκετά μεγάλου περιβάλλοντος προσομοίωσης σε έναν υπολογιστή (desktop, laptop) με μέτρια επεξεργαστική ισχύ. [12]

CloudSim Shut Down: Είναι μια οντότητα που ο κυρίως ρόλος της είναι να περιμένει τον τερματισμό



Εικόνα 8: Επικοινωνία των κλάσεων [14]

όλων των οντοτήτων, έτσι ώστε να αναγνωριστεί το τέλος της προσομοίωσης. Στον κώδικα δηλώνεται με το ID = %0.

Cloud Information Service: Είναι μια εξαιρετικά σημαντική οντότητα, που παρέχει την καταχώρηση των πόρων και καταλαβαίνει ποια υπηρεσία χρησιμοποιείται εκείνη τη στιγμή. Βασικά επεκτείνει την κλάση CloudSimCore για την εκτέλεση της προσομοίωσης στο CloudSim. Στον κώδικα δηλώνεται με το ID = %1.

Data Center: Είναι μια ακόμα σημαντική οντότητα, που βοηθάει στη δημιουργία των hosts, με ένα κατάλληλο πλήθος χαρακτηριστικών όπως RAM, CPU, Bandwidth, MIPS κ.α. Ο εκάστοτε host χωρίζεται σε εικονικές μηχανές, οι οποίες με την σειρά τους εκτελούν τις διεργασίες των χρηστών. Στον κώδικα δηλώνεται με το ID = %2.

Data Center Broker: Είναι ο πυρήνας του CloudSim. Αυτή η οντότητα βοηθάει στην εκτέλεση των διεργασιών με την κατάλληλη σειρά. Στον κώδικα δηλώνεται με το ID = %3.

Για την δημιουργία του Data Center και του Data Center Broker ορίζονται στο πρόγραμμα οι παρακάτω γραμμές του κώδικα 1:

```
1 Datacenter dcObject=new Datacenter("Dc1"); //Creation of object for Data Center with name Dc1
2 DatacenterBroker broker=new DatacenterBroker(); //Creation of broker
3 Int brokerId= broker.getId(); //It is used in order to take broker's id in CloudSim
```

Κώδικας 1: Δημιουργία data center και broker

Άλλα βασικά στοιχεία του CloudSim όπως οι εικονικές μηχανές, οι πολιτικές παροχής και οι hosts δημιουργούνται ως αυτοδύναμα αντικείμενα, τα οποία δεν καταναλώνουν πολύ μνήμη και δεν ανταγωνίζονται για επεξεργαστική ισχύ. Συνεπώς, ασχέτως από τον αριθμό των hosts σε ένα προσομοιωμένο data center, το περιβάλλον εκτέλεσης Java Virtual Machine χρειάζεται να διαχειριστεί μόνο δύο νήματα (το Data Center και το Broker).

Καθώς η επεξεργασία των μονάδων έργου γίνεται από τις εικονικές μηχανές, η πρόοδός τους πρέπει να ενημερώνεται και να παρακολουθείται σε κάθε βήμα της προσομοίωσης. Για να αντιμετωπιστεί αυτό, παράγεται ένα εσωτερικό γεγονός σχετικά με τον αναμενόμενο χρόνο ολοκλήρωσης της μονάδας έργου, το οποίο ενημερώνει την οντότητα Data center σχετικά με τα μελλοντικά γεγονότα που θα ολοκληρωθούν.

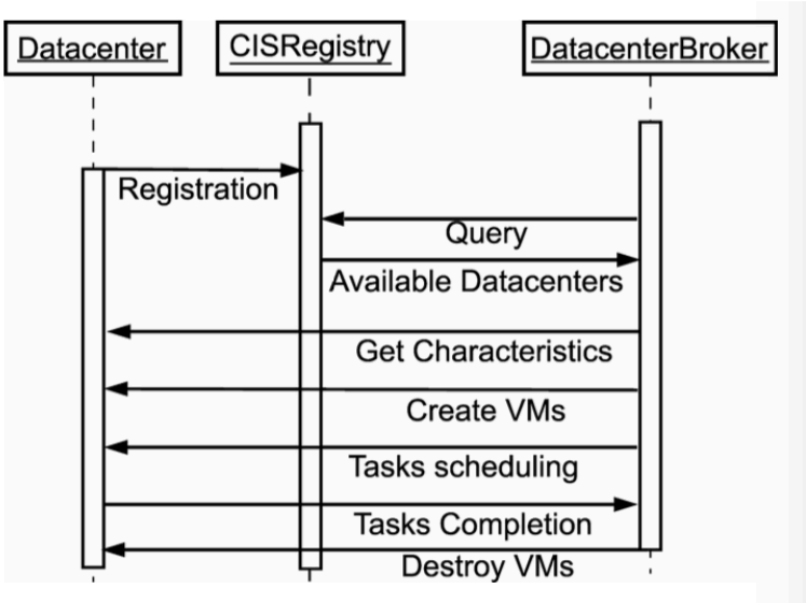
Έτσι σε κάθε βήμα της προσομοίωσης, κάθε Data Center καλεί μια μέθοδο που ονομάζεται updateVMsProcessing() για κάθε host του συστήματος, ώστε να ενημερώσει για την επεξεργασία των έργων που τρέχουν μέσα στις εικονικές μηχανές. Το όρισμα της μεθόδου αυτής είναι ο τρέχων χρόνος προσομοίωσης και ο τύπος που επιστρέφεται είναι ο επόμενος αναμενόμενος χρόνος ολοκλήρωσης ενός έργου που τρέχει σε μια από τις εικονικές μηχανές ενός συγκεκριμένου host. Ο ελάχιστος χρόνος, ανάμεσα σε όλους τους χρόνους, που επιστρέφονται κατά την ολοκλήρωση από τους hosts σημειώνεται για το επόμενο εσωτερικό γεγονός.

Στο επίπεδο του host, η κλήση της μεθόδου updateVMsProcessing(), καλεί την μέθοδο updateGridletsProcessing(), η οποία κατευθύνει κάθε εικονική μηχανή να ενημερώσει την πρόοδο κάθε μονάδας έργου (ολοκλήρωση, αναστολή, εκτέλεση) με την οντότητα Data Center. Αυτή η μέθοδος εφαρμόζει ίδια λογική με την προαναφερθείσα updateVMsProcessing() αλλά για το επίπεδο εικονικής μηχανής.

Μόλις κληθεί αυτή η μέθοδος, οι εικονικές μηχανές επιστρέφουν τον επόμενο αναμενόμενο χρόνο ολοκλήρωσης των μονάδων έργου, που διαχειρίζονται από αυτές. Ο ελάχιστος χρόνος ολοκλήρωσης ανάμεσα στις υπολογισμένες τιμές, στέλνεται στην οντότητα Data center. Αυτό έχει ως αποτέλεσμα, οι χρόνοι ολοκλήρωσης να κρατούνται σε μια ουρά από το Datacenter, μετά από κάθε βήμα επεξεργασίας γεγονότος. Εάν υπάρχουν ολοκληρωμένα έργα, τα οποία περιμένουν στην ουρά, τότε αφαιρούνται από αυτή και στέλνονται πίσω στο χρήστη.

2.7 Επικοινωνία οντοτήτων

Η εικόνα 9 παρουσιάζει τη ροή της επικοινωνίας ανάμεσα στις κύριες οντότητες του CloudSim. Στην αρχή της προσομοίωσης, κάθε οντότητα Datacenter γράφει τον εαυτό της στο αρχείο της Cloud Information Service (CIS). Η CIS παρέχει υπηρεσίες σε επίπεδο βάσης δεδομένων, που αντιστοιχούν τους χρήστες στους κατάλληλους παρόχους νέφους. Οι Brokers ενεργούν για λογαριασμό των χρηστών και συμβουλεύουν την CIS για την λίστα των Clouds που παρέχουν υποδομές υπηρεσιών και ταιριάζουν με τις απαιτήσεις εφαρμογών των χρηστών. Σε περίπτωση που υπάρχει ταίριασμα ο broker αναπτύσσει την εφαρμογή στο νέφος, που έχει προταθεί από την CIS. [12]



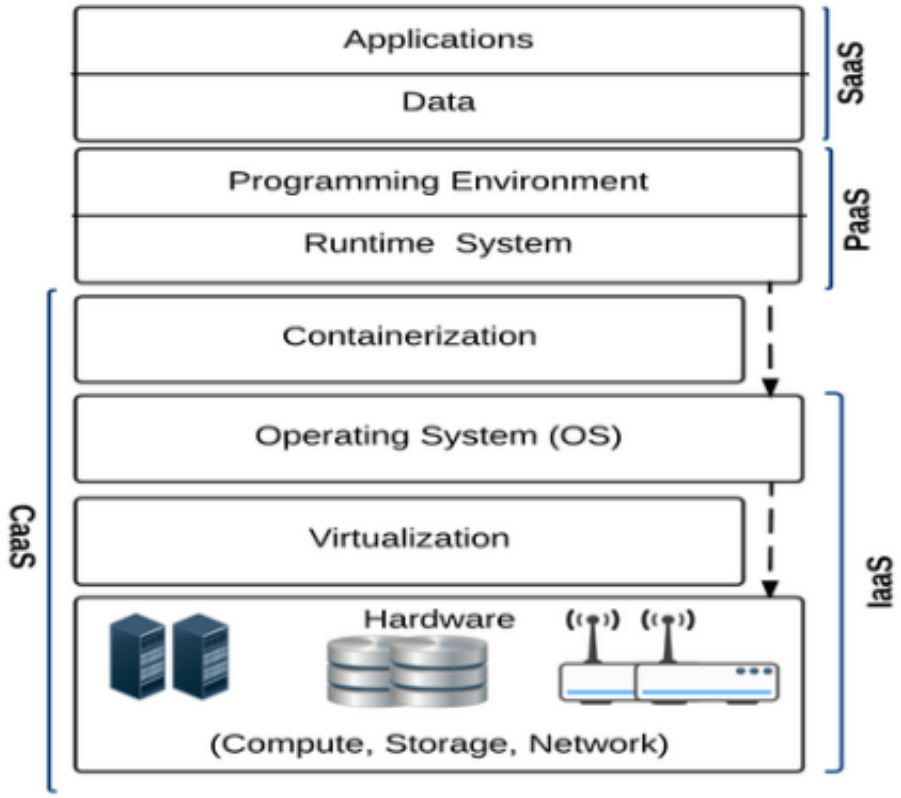
Εικόνα 9: Τα βήματα της επικοινωνίας των οντοτήτων στο CloudSim [12]

Η ροή της επικοινωνίας που περιγράφτηκε σχετίζεται με την βασική ροή σε ένα πείραμα προσομοίωσης. Κάποιες διαφοροποιήσεις σε αυτή τη ροή είναι δυνατές ανάλογα με τις πολιτικές. Για παράδειγμα, τα μηνύματα από τους Brokers στα Datacenters, μπορεί να απαιτούν επιβεβαίωση από την πλευρά του Datacenter, σχετικά με την εκτέλεση της ενέργειας. Ακόμα, ο μέγιστος αριθμός εικονικών μηχανών που μπορεί ένας χρήστης να δημιουργήσει, είναι δυνατόν να έχει διαπραγματευτεί πριν από την δημιουργία της εικονικής μηχανής.

2.8 Containers

Στην έκδοση 4.0 του CloudSim υποστηρίχθηκαν οι Containers. Επιπρόσθετα από τα παραδοσιακά μοντέλα υπηρεσιών που υπάρχουν στο νέφος, Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS) πρόσφατα ένας νέος τύπος υπηρεσίας το Containers as a Service (CaaS) έκανε την εμφάνιση του. Ένα παράδειγμα συστήματος διαχείρισης των container είναι το Docker, που επιτρέπει στους προγραμματιστές να ορίσουν containers για εφαρμογές. Οι containers μοιράζονται τον ίδιο πυρήνα με τον host, συνεπώς ορίζονται ως ελαφρά εικονικά περιβάλλοντα, σε σχέση με τις εικονικές μηχανές VMs, τα οποία παρέχουν ένα επίπεδο απομόνωσης ανάμεσα στους φόρτους εργασιών χωρίς να εμπλέκεται η εικονικοποίηση που είναι βασισμένη σε ένα υπερεπόπτη (hypervisor). Το μοντέλο CaaS βρίσκεται ανάμεσα στα μοντέλα IaaS και PaaS, όπου το IaaS προσφέρει εικονικούς υπολογιστικούς πόρους και το PaaS συγκεκριμένες υπηρεσίες για την εκτέλεση των εφαρμογών. Έτσι ενώνει αυτά τα δύο επίπεδα, παρέχοντας απομονωμένα περιβάλλοντα για τις ανεπτυγμένες εφαρμογές. [15]

Όπως φαίνεται στην εικόνα 10, οι υπηρεσίες CaaS παρέχονται συνήθως πάνω απο τις εικονικές μηχανές του επιπέδου IaaS. Οι πάροχοι CaaS, όπως η Google και η Amazon Web Services ισχυρίζονται οτι οι containers προσφέρουν κατάλληλο περιβάλλον για μερικώς έμπιστους φόρτους εργασιών, ενώ οι εικονικές μηχανές παρέχουν ακόμα ένα επίπεδο ασφάλειας για τους αναξιόπιστους φόρτους.



Εικόνα 10: Το μοντέλο υπηρεσίας Container as a Service

Οι πολιτικές διαχείρισης πόρων για την εξασφάλιση της ποιότητας υπηρεσίας QoS, η αποφυγή ενεργειακής απώλειας και ο κατακερματισμός των πόρων είναι ένα αναπόσπαστο κομμάτι των cloud συστημάτων.

Το ContainerCloudSim έχει προταθεί ως ένα περιβάλλον προσομοίωσης για την μελέτη τεχνικών διαχείρισης πόρων σε περιβάλλοντα CaaS. Έχει αναπτυχθεί ως μια επέκταση του πακέτου προσομοίωσης CloudSim και παρέχει ένα περιβάλλον για την αξιολόγηση αυτών των τεχνικών, όπως είναι ο προγραμματισμός, η τοποθέτηση και η ενοποίηση των container. Ακόμα, επιτρέπει μοναδικά στους ερευνητές να εξετάσουν τις τεχνικές διαχείρισης πόρων, για τους δύο τύπους virtualization, που περιλαμβάνουν το virtualization σε επίπεδο Λειτουργικού Συστήματος και το virtualization σε επίπεδο συστήματος/εικονικών μηχανών. Για τον τύπο των εικονικών μηχανών, οι εφαρμογές εκτελούνται μέσα στις εικονικές μηχανές και για το μοντέλο CaaS οι εφαρμογές εκτελούνται μέσα στους containers, που είναι τοποθετημένοι μέσα στις εικονικές μηχανές. Τέλος, προσφέρει ένα περιβάλλον για την αξιολόγηση διαφόρων αλγορίθμων

διαχείρισης πόρων με βάση την ενέργεια (power-aware), παρέχοντας ποικίλα μοντέλα ενέργειας σε ένα data center.

Τα κύρια χαρακτηριστικά του πακέτου επέκτασης ContainerCloudSim είναι: [15]

- Υποστήριξη για μοντελοποίηση και προσομοίωση περιβάλλοντων νέφους που είναι βασιμένα στους containers.
- Υποστήριξη για μοντελοποίηση και προσομοίωση εικονικών μηχανών που είναι βασισμένες στους containers, με προσαρμόσιμες πολιτικές για ανάθεση των πόρων των εικονικών μηχανών σε containers.
- Υποστήριξη για μοντελοποίηση και προσομοίωση ενεργειακά ενήμερων υπολογιστικών πόρων.
- Υποστήριξη για μοντελοποίηση και προσομοίωση δικτυακών τοπολογιών των data centers και εφαρμογές message passing.
- Υποστήριξη για μοντελοποίηση και προσομοίωση ομόσπονδων νεφών.
- Υποστήριξη για δυναμική εισαγωγή στοιχείων προσομοίωσης, παύση και συνέχιση της προσομοίωσης.
- Υποστήριξη πολιτικών ορισμένων απο τους χρήστες για ανάθεση εικονικών μηχανών σε containers και hosts σε εικονικές μηχανές.

2.9 Εγκατάσταση του CloudSim

Ξεκινώντας να δουλεύουμε στο CloudSim θα πρέπει να το εγκαταστήσουμε σωστά. Καθώς το toolkit του CloudSim αναπτύχθηκε με την γλώσσα προγραμματισμού Java, μπορούμε να χρησιμοποιήσουμε ένα οποιοδήποτε IDE, που υποστηρίζει την Java όπως το Eclipse, Netbeans κ.α. Ένα μεγάλο πλεονέκτημα του προσομοιωτή είναι ότι μπορεί να τρέξει το ίδιο εύκολα σε όλα τα λειτουργικά συστήματα Windows, Linux και Mac OS X, αφού τα IDE είναι διαθέσιμα για αυτά.

Για την επιτυχή εγκατάσταση του CloudSim χρησιμοποιώντας το Eclipse IDE, πρέπει να έχουμε εγκατεστημένα τα παρακάτω προαπαιτούμενα:

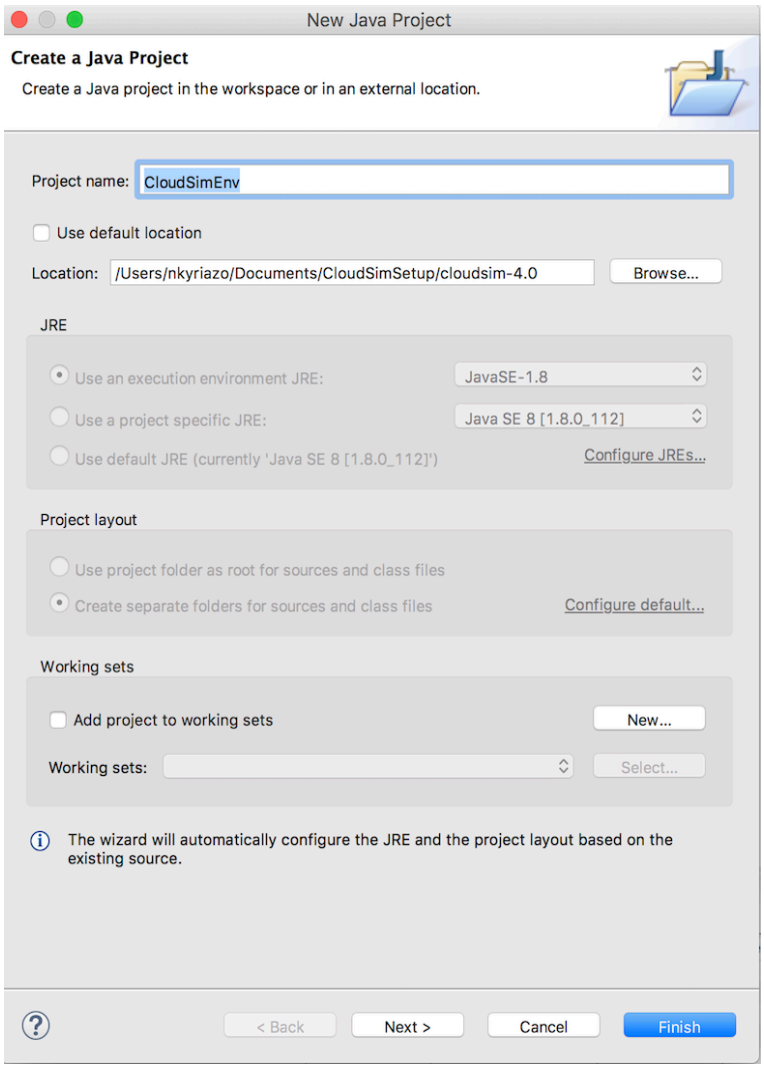
1. Το πακέτο του προσομοιωτή CloudSim, κατεβασμένο από το GitHub στη σελίδα `https://github.com/Cloudslab/cloudsim/releases`. Από εκεί επιλέγουμε την έκδοση που επιθυμούμε και κατεβάζουμε το αρχείο με κατάληξη .zip.
2. Εγκατεστημένο το Eclipse IDE for Java Developers από τη σελίδα `https://www.eclipse.org`.
3. Την τελευταία έκδοση του Java Development Kit (JDK) για τη μεταγλώττιση του κώδικα. Η τελευταία έκδοση μπορεί να βρεθεί στη σελίδα της Oracle. Για την διατριβή χρησιμοποιήθηκε η έκδοση 8+.
4. Για τη σωστή παραμετροποίηση του CloudSim, πρέπει να εγκατασταθεί και μια βιβλιοθήκη που καλείται commons-math, η οποία βρίσκεται στη σελίδα `http://commons.apache.org/proper/commons-math/download_math.cgi`. Μόλις κατέβει το .zip αρχείο πρέπει να γίνει εξαγωγή του "commons-math3-3.6.1.jar" σε κάποιον φάκελο.

Ο χρήστης ανοίγοντας το Eclipse πρέπει να δημιουργήσει ένα νέο Project όπως φαίνεται στην εικόνα 11. Σε αυτό το σημείο πρέπει να ορίσει έναν κατάλογο που θα αποθηκεύσει τα αρχεία του, οπότε στην εικόνα 11 φαίνεται ότι δεν είναι επιλεγμένο το *Use default location* και στο πεδίο Location βρίσκεται η διαδρομή για τον φάκελο του CloudSim που έχει κατέβει στον υπολογιστή. Στη συνέχεια, δεν πατάει *Finish*, αλλά το *Next* ώστε να εισαχθεί στο Project η βιβλιοθήκη common-math.

Στην επόμενη οθόνη ο χρήστης βλέπει την εικόνα 12. Εκεί πρέπει να πατήσει την επιλογή *Add External JARs* και να βρει το αρχείο jar της βιβλιοθήκης που κατέβασε. Το αρχείο αυτό θα προστεθεί στη λίστα που φαίνεται αριστερά της εικόνας 12 και τότε πατάει στο *Finish* ώστε να ολοκληρωθεί η δημιουργία του Project.

Αφού δημιουργηθεί το Project, παρουσιάζονται αριστερά οι κατάλογοι και τα αρχεία του πακέτου. Ο χρήστης μπορεί να περιηγηθεί να εκτελέσει τα παραδείγματα και να βγάλει χρήσιμα συμπεράσματα, όπως φαίνονται στο επόμενο κεφάλαιο.

Η δομή καταλόγου του εργαλείου CloudSim φαίνεται στον πίνακα 1.



Εικόνα 11: Δημιουργία νέου Project

cloudsim/	– ο αρχικός κατάλογος του πακέτου
docs/	– CloudSim API Documentation
examples/	– παραδείγματα του CloudSim
jars/	– αρχεία jar του CloudSim
sources/	– πηγαίος κώδικας του CloudSim
tests/	– CloudSim unit tests

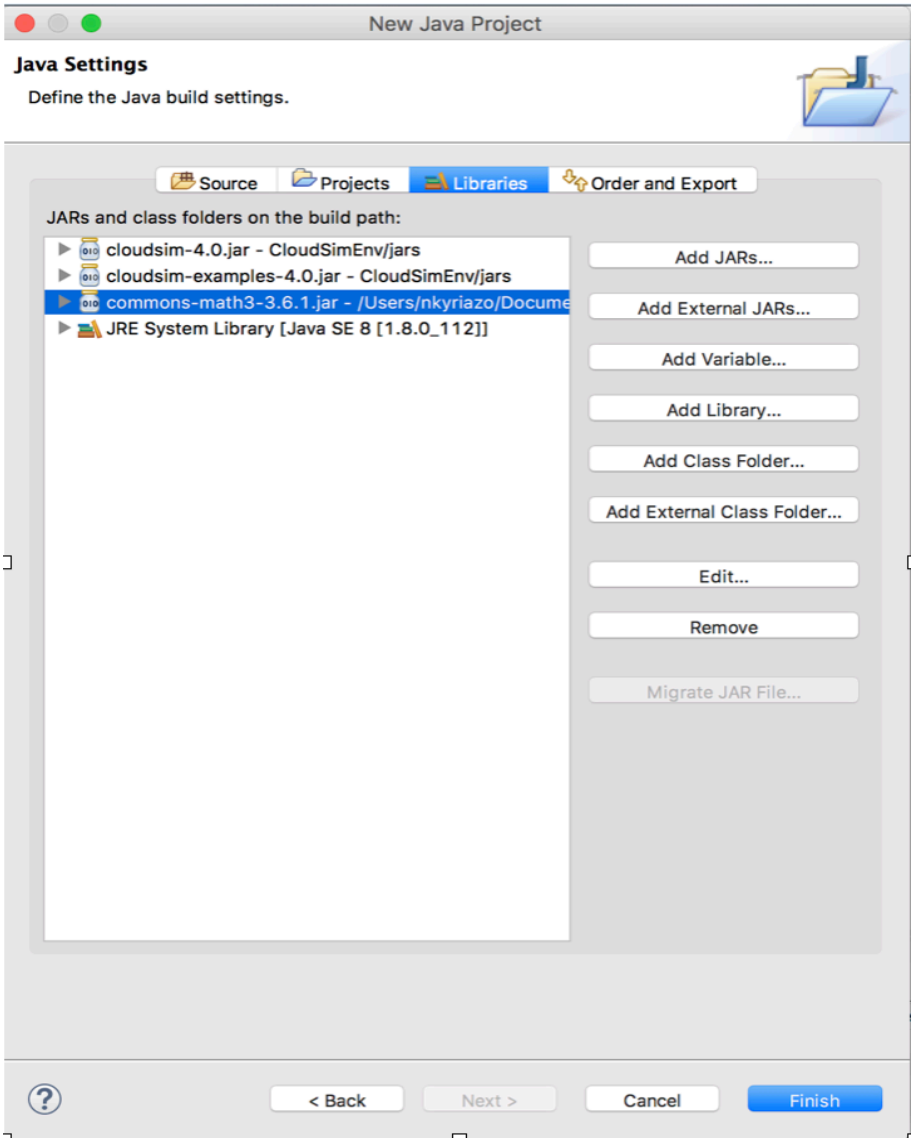
Πίνακας 1: Η δομή καταλόγου του CloudSim

2.10 Παραδείγματα προσομοίωσης

Το CloudSim είναι ένα πανίσχυρο εργαλείο, που προσφέρει πολλά παραδείγματα προσομοίωσης και συμπεριλαμβάνονται στην έκδοση κώδικα που κατεβάζει ο χρήστης από το Github.

Ο πηγαίος κώδικας των παραδειγμάτων βρίσκεται στα αρχεία του CloudSim κάτω από τον φάκελο `<ΜΟΝΟΠΑΤΙ ΓΙΑ ΤΟ ΠΑΚΕΤΟ ΤΟΥ CloudSim>/examples/org/cloudbus/cloudsim/examples/`. Εκεί βρίσκονται επίσης και οι φάκελοι network και power που έχουν τα αντίστοιχα δικτυακά παραδείγματα και παραδείγματα σχετικά με την ενέργεια.

Κάθε παράδειγμα το οποίο περιλαμβάνεται στο πακέτο "org.cloudbus.cloudsim.example" που βρίσκεται στο φάκελο παραδειγμάτων του CloudSim, ακολουθεί συγκεκριμένα βήματα για την εφαρμογή των ειδικών παραμέτρων έτσι ώστε να ξεκινήσει η προσομοίωση. Για την κατανόηση της λειτουργίας του framework προσομοίωσης του CloudSim, η γνώση αυτών των βημάτων είναι απαραίτητη. Υπάρχουν έντεκα βήματα



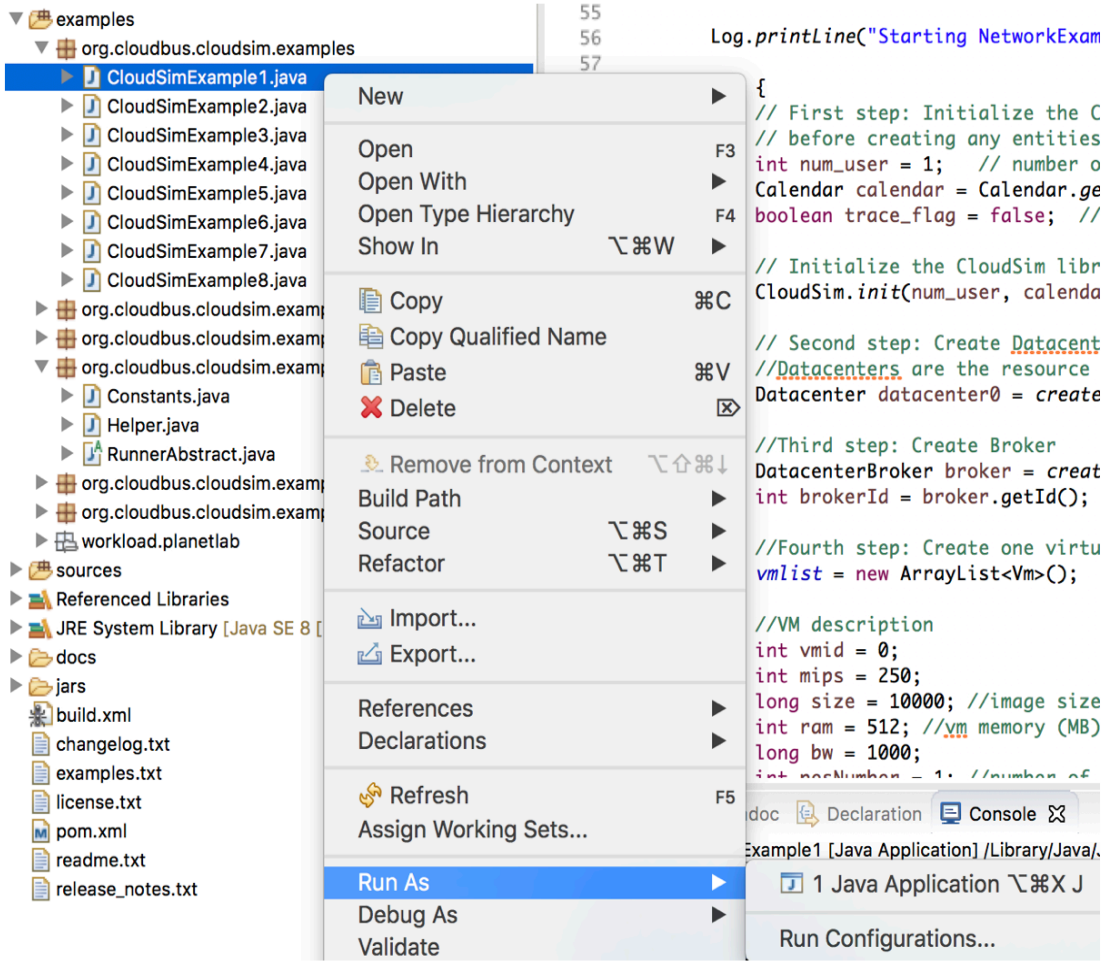
Εικόνα 12: Εγκατάσταση της βιβλιοθήκης common_math

που ακολουθούνται σε κάθε παράδειγμα με μικρές διαφοροποιήσεις μεταξύ τους. Αυτά είναι:

1. Προσδιορισμός του αριθμού των χρηστών. Αυτός ο αριθμός είναι ευθέως ανάλογος με τον αριθμό των brokers στην τρέχουσα προσομοίωση
2. Αρχικοποίηση της προσομοίωσης, που περιλαμβάνει την τρέχουσα ώρα, των αριθμό των χρηστών και το trace flag.
3. Δημιουργία του Data Center.
4. Δημιουργία του Data Center Broker.
5. Δημιουργία ενός ή περισσότερων εικονικών μηχανών.
6. Ανάθεση ενός ή περισσότερων εικονικών μηχανών στον broker.
7. Δημιουργία των cloudlets (διεργασίες) και προσδιορισμός των χαρακτηριστικών τους.
8. Ανάθεση των cloudlets στον broker.
9. Εντολή για εκκίνηση της προσομοίωσης.
10. Εάν δεν υπάρχει κάποιο γεγονός προς εκτέλεση, τότε στέλνεται η εντολή για τερματισμό της προσομοίωσης.
11. Τέλος, εκτυπώνεται στην οθόνη το τελικό αποτέλεσμα της προσομοίωσης.

Ο χρήστης μπορεί να επέμβει στον κώδικα των παραδειγμάτων και να αλλάξει τα δομικά χαρακτηριστικά του data center, των εικονικών μηχανών κ.α. Για να εκτελέσει ένα παράδειγμα, πρέπει να το επιλέξει

μέσω του Eclipse και να το τρέξει ως Java Application, όπως φαίνεται στην εικόνα 13.



Εικόνα 13: Εκτέλεση ενός παραδείγματος στο Eclipse

Στη συνέχεια, η κονσόλα του προγράμματος εμφανίζει τα αποτελέσματα της προσομοίωσης, από όπου μπορούν να εξαχθούν συμπεράσματα σχετικά με αυτό που θέλει να πετύχει ο χρήστης στο μοντέλο προσομοίωσης.

Στον πηγαίο κώδικα υλοποιούνται οι παράμετροι της εκάστοτε εικονικής μηχανής, τις οποίες μπορεί κάποιος να παραμετροποιήσει ανάλογα με τις ανάγκες του.

Στο πλαίσιο 2 που υπάρχει υλοποιημένος κώδικας σε Java, ορίζεται η κάθε παράμετρος, που παίζει καθοριστικό ρόλο στα αποτελέσματα της προσομοίωσης. Σε αυτές τις παραμέτρους μπορεί ο χρήστης να καθορίσει τις εξής τιμές: το μέγεθος της εικονικής μηχανής σε MB, το μέγεθος της RAM που είναι εγκατεστημένη στην εικονική μηχανή σε MB, τον αριθμό των εντολών που εκτελούνται ανά δευτερόλεπτο σε εκατομμύρια - MIPS, το εύρος ζώνης της εικονικής μηχανής, τον αριθμό των επεξεργαστών (cpus) και τέλος το όνομα του διαχειριστή της εικονικής μηχανής VMM.

```
// VM Parameters
int vmid = 0;
int mips = 1000;
long size = 10000; // image size (MB)
int ram = 512; // vm memory (MB)
long bw = 1000;
int pesNumber = 1; // number of cpus
String vmm = "Xen"; // VMM name
```

Κώδικας 2: Οι ρυθμίσεις των εικονικών μηχανών με κώδικα Java

2.10.1 Παράδειγμα CloudSimExample1.java

Ακολουθεί ένα απλό παράδειγμα, που δείχνει την δημιουργία ενός data center με έναν host και την εκτέλεση ενός cloudlet σε αυτό. Βλέπουμε όλα τα βήματα από την αρχικοποίηση του προγράμματος, μέχρι τον τερματισμό και το αποτέλεσμα της προσομοίωσης.

```
Starting CloudSimExample1...
Initialising...
Starting CloudSim version 3.0
Datacenter_0 is starting...
Broker is starting...
Entities started.
0.0: Broker: Cloud Resource List received with 1 resource(s)
0.0: Broker: Trying to Create VM #0 in Datacenter_0
0.1: Broker: VM #0 has been created in Datacenter #2, Host #0
0.1: Broker: Sending cloudlet 0 to VM #0
400.1: Broker: Cloudlet 0 received
400.1: Broker: All Cloudlets executed. Finishing...
400.1: Broker: Destroying VM #0
Broker is shutting down...
Simulation: No more future events
CloudInformationService: Notify all CloudSim entities for shutting down.
Datacenter_0 is shutting down...
Broker is shutting down...
Simulation completed.
Simulation completed.
```

Εικόνα 14: Εκτέλεση πρώτου παραδείγματος

Αρχικά δημιουργούνται οι οντότητες Datacenter και Broker. Η οντότητα Broker με την σειρά της ειδοποιεί το Datacenter ότι πρέπει να δημιουργήσει μια εικονική μηχανή με ID = 0. Στη συνέχεια, στέλνει μια διεργασία (cloudlet) σε αυτή την εικονική μηχανή, όπου μετά από ένα χρονικό διάστημα η διεργασία λαμβάνεται και εκτελείται. Τέλος, καταστρέφεται η εικονική μηχανή και κλείνουν όλες οι οντότητες. Η προσομοίωση έχει τελειώσει και το αποτέλεσμα της φαίνεται στην οθόνη του χρήστη. Στο αποτέλεσμα ο χρήστης μπορεί να ξεχωρίσει το Cloudlet ID, που στην συγκεκριμένη περίπτωση είναι 0, την κατάσταση της προσομοίωσης που είναι επιτυχημένη, το ID του Data center και το ID της εικονικής μηχανής. Τέλος, φαίνεται ο αρχικός, τελικός και συνολικός χρόνος εκτέλεσης της προσομοίωσης.

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID    Time    Start Time    Finish Time
    0        SUCCESS        2            0        400        0.1        400.1
CloudSimExample1 finished!
```

Εικόνα 15: Αποτελέσματα πρώτου παραδείγματος

2.10.2 Παράδειγμα CloudSimExample2.java

Στο δεύτερο παράδειγμα δημιουργείται ένα datacenter, με ένα host αλλά εκτελούνται δύο cloudlets σε αυτό. Τα cloudlets εκτελούνται σε εικονικές μηχανές με τις ίδιες απαιτήσεις MIPS και απαιτούν ίδιο χρόνο για την ολοκλήρωση της εκτέλεσης.

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID    Time    Start Time    Finish Time
    0        SUCCESS        2            0        1000        0.1        1000.1
    1        SUCCESS        2            1        1000        0.1        1000.1
CloudSimExample2 finished!
```

Εικόνα 16: Αποτελέσματα δεύτερου παραδείγματος

Αυτό που παρατηρούμε είναι ότι ο χρόνος εκτέλεσης της προσομοίωσης είναι σχεδόν ο διπλάσιος, σε σχέση με τον χρόνο εκτέλεσης του πρώτου παραδείγματος. Αυτό είναι απόλυτα λογικό αφού έχουμε ένα host, με δύο εικονικές μηχανές ανεπτυγμένες σε αυτόν και δύο διεργασίες να εκτελούνται παράλληλα σε αυτές. Οι εικονικές μηχανές έχουν ξεχωριστό ID το 0 και το 1.

2.10.3 Παράδειγμα CloudSimExample3.java

Σε αυτό το παράδειγμα, που περιλαμβάνει ένα datacenter με δύο hosts, εκτελούνται δύο cloudlets. Αυτά τρέχουν σε εικονικές μηχανές με διαφορετικές απαιτήσεις MIPS, όπως φαίνεται και στις παρακάτω γραμμές του κώδικα 3.

```
//create two VMs
Vm vm1 = new Vm(vmid, brokerId, mips, pesNumber, ram, bw, size, vmm,
                new CloudletSchedulerTimeShared());

//the second VM will have twice the priority of VM1 and so will receive twice CPU time
vmid++;
Vm vm2 = new Vm(vmid, brokerId, mips * 2, pesNumber, ram, bw, size, vmm,
                new CloudletSchedulerTimeShared());
```

Κώδικας 3: Εικονικές μηχανές με διαφορετικά MIPS

Παρατηρούμε ότι στην δεύτερη εικονική μηχανή, η παράμετρος MIPS πολλαπλασιάζεται με το δύο, οπότε έχει υψηλότερη προτεραιότητα. Τα cloudlets χρειάζονται διαφορετικό χρόνο για να ολοκληρώσουν την εκτέλεση και εξαρτώνται από την απόδοση της εικονικής μηχανής.

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID    Time    Start Time    Finish Time
    1        SUCCESS        2             1        80        0.1         80.1
    0        SUCCESS        2             0        160        0.1         160.1
CloudSimExample3 finished!
```

Εικόνα 17: Αποτελέσματα τρίτου παραδείγματος

2.10.4 Παράδειγμα CloudSimExample4.java

Ακόμα ένα σενάριο περιγράφεται στον κώδικα του τέταρτου παραδείγματος, στο οποίο δημιουργούνται δύο data centers, με ένα host το καθένα και τρέχουν δύο cloudlets σε αυτά.

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID    Time    Start Time    Finish Time
    0        SUCCESS        2             0        160        0.2         160.2
    1        SUCCESS        3             1        160        0.2         160.2
CloudSimExample4 finished!
```

Εικόνα 18: Αποτελέσματα τέταρτου παραδείγματος

Παρατηρούμε, ότι ο χρόνος εκτέλεσης είναι μικρός αφού κάθε cloudlet εκτελείται σε έναν ξεχωριστό host και σε διαφορετικό data center.

Τα data centers δηλώνονται με τις γραμμές του κώδικα 4:

2.10.5 Παράδειγμα CloudSimExample5.java

Στο πέμπτο παράδειγμα, δημιουργούνται δύο data centers με ένα host το καθένα και τρέχουν 2 cloudlets δύο διαφορετικών χρηστών.

Βλέπουμε ότι δημιουργούνται οι χρήστες με ID = 4 και ID = 5 και για κάθε χρήστη δημιουργείται ένας broker. Ο χρόνος εκτέλεσης των cloudlets είναι ο ίδιος και για τους δύο χρήστες, γιατί οι εικονικές μηχανές έχουν τα ίδια τεχνικά χαρακτηριστικά και τα cloudlets όμοιες ιδιότητες μεταξύ τους.

```java
// Second step: Create Datacenters
//Datacenters are the resource providers in CloudSim.
//We need at list one of them to run a CloudSim simulation
@SuppressWarnings("unused")
Datacenter datacenter0 = createDatacenter("Datacenter_0");
@SuppressWarnings("unused")
Datacenter datacenter1 = createDatacenter("Datacenter_1");
```

Κώδικας 4: Δήλωση των data centers στον πηγαίο κώδικα

```
=============> User 4
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID    Time    Start Time    Finish Time
    0        SUCCESS        2             0        160        0.1        160.1
=============> User 5
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID    Time    Start Time    Finish Time
    0        SUCCESS        3             0        160        0.2        160.2
CloudSimExample5 finished!
```

Εικόνα 19: Αποτελέσματα πέμπτου παραδείγματος

2.10.6 Παράδειγμα CloudSimExample6.java

Το έκτο παράδειγμα περιλαμβάνει προσομοίωση για την επεκτασιμότητα ενός υπολογιστικού νέφους. Συγκεκριμένα στον τερματισμό της προσομοίωσης, εμφανίζεται το παρακάτω αποτέλεσμα.

```
0.0: Broker: Trying to Create VM #16 in Datacenter_0
0.0: Broker: Trying to Create VM #17 in Datacenter_0
0.0: Broker: Trying to Create VM #18 in Datacenter_0
0.0: Broker: Trying to Create VM #19 in Datacenter_0
[VmScheduler.vmCreate] Allocation of VM #6 to Host #0 failed by RAM
[VmScheduler.vmCreate] Allocation of VM #6 to Host #1 failed by MIPS
[VmScheduler.vmCreate] Allocation of VM #7 to Host #0 failed by RAM
[VmScheduler.vmCreate] Allocation of VM #7 to Host #1 failed by MIPS
[VmScheduler.vmCreate] Allocation of VM #8 to Host #0 failed by RAM
[VmScheduler.vmCreate] Allocation of VM #8 to Host #1 failed by MIPS
[VmScheduler.vmCreate] Allocation of VM #9 to Host #0 failed by RAM
```

Εικόνα 20: Εκτέλεση έκτου παραδείγματος

Από αυτό συμπεραίνουμε ότι λόγω περιορισμών στην μνήμη RAM και στο MIPS, οι εικονικές μηχανές που προσπαθούν να δημιουργηθούν και έχουν παραπάνω απαιτήσεις, αποτυγχάνουν. Όμως λόγω της ικανότητας του νέφους να επεκτείνεται και να δεσμεύει πόρους ανάλογα με την ζήτηση, στο τελικό αποτέλεσμα παρατηρούμε ότι οι εικονικές μηχανές δημιουργήθηκαν επιτυχώς.

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID    Time    Start Time    Finish Time
    4        SUCCESS        2             4        3        0.2        3.2
    16        SUCCESS        2             4        3        0.2        3.2
    28        SUCCESS        2             4        3        0.2        3.2
    5        SUCCESS        2             5        3        0.2        3.2
    17        SUCCESS        2             5        3        0.2        3.2
    29        SUCCESS        2             5        3        0.2        3.2
    6        SUCCESS        3             6        3        0.2        3.2
    18        SUCCESS        3             6        3        0.2        3.2
    30        SUCCESS        3             6        3        0.2        3.2
    7        SUCCESS        3             7        3        0.2        3.2
    19        SUCCESS        3             7        3        0.2        3.2
    31        SUCCESS        3             7        3        0.2        3.2
    8        SUCCESS        3             8        3        0.2        3.2
```

Εικόνα 21: Αποτελέσματα έκτου παραδείγματος

2.10.7 Παράδειγμα CloudSimExample7.java

Ένα παράδειγμα που δείχνει πώς μπορεί να σταματήσει και να ξεκινήσει πάλι η προσομοίωση. Επίσης δημιουργείται δυναμικά μια οντότητα, συγκεκριμένα ο broker.

```
200.0: The simulation is paused for 5 sec


Adding: Broker_1
Broker_1 is starting...
```

Εικόνα 22: Αποτελέσματα έβδομου παραδείγματος

Φαίνεται ότι η προσομοίωση σταματάει προσωρινά για πέντε δευτερόλεπτα και προστίθεται ο Broker 1 δυναμικά, όπου με την σειρά του δημιουργεί τις εικονικές μηχανές και τις αναθέτει στα data centers.

2.10.8 Παράδειγμα CloudSimExample8.java

Στο τελευταίο παράδειγμα αυτής της ομάδας, προσομοιώνεται η δημιουργία οντοτήτων, στη συγκεκριμένη περίπτωση ένας broker σε πραγματικό χρόνο χρησιμοποιώντας μια καθολική οντότητα διαχείρισης, τον Global Broker.

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID        Time    Start Time    Finish Time
    0        SUCCESS        3             0          320        0.1            320.1
    5        SUCCESS        3             0          320        0.1            320.1
    1        SUCCESS        3             1          320        0.1            320.1
    6        SUCCESS        3             1          320        0.1            320.1
    2        SUCCESS        3             2          320        0.1            320.1
    7        SUCCESS        3             2          320        0.1            320.1
    4        SUCCESS        3             4          320        0.1            320.1
    9        SUCCESS        3             4          320        0.1            320.1
    3        SUCCESS        3             3          320        0.1            320.1
    8        SUCCESS        3             3          320        0.1            320.1
    101        SUCCESS        3             101            320        200.1            520.1
    106        SUCCESS        3             101            320        200.1            520.1
    103        SUCCESS        3             103            320        200.1            520.1
    108        SUCCESS        3             103            320        200.1            520.1
    100        SUCCESS        3             100            320        200.1            520.1
    105        SUCCESS        3             100            320        200.1            520.1
    102        SUCCESS        3             102            320        200.1            520.1
    107        SUCCESS        3             102            320        200.1            520.1
    104        SUCCESS        3             104            320        200.1            520.1
    109        SUCCESS        3             104            320        200.1            520.1
CloudSimExample8 finished!
```

Εικόνα 23: Αποτελέσματα όγδοου παραδείγματος

Παρατηρούμε ότι τα cloudlets με ID από 101-109, τα έχει διαχειριστεί ο Global Broker, έχουν ανατεθεί σε πέντε εικονικές μηχανές και έχουν εκτελεστεί επιτυχώς.

2.11 Εκδόσεις CloudSim

2.11.1 Έκδοση 1.0

Ήταν η πρώτη έκδοση του CloudSim και εκδόθηκε στις 7 Απριλίου 2009. Αυτή η έκδοση του CloudSim υποστήριζε την μοντελοποίηση και την προσομοίωση μεγάλης κλίμακας υποδομής υπολογιστικής νέφους, συμπεριλαμβανομένων των data centers σε ένα μοναδικό φυσικό υπολογιστικό κόμβο, μια αυτόνομη πλατφόρμα για μοντελοποίηση data center, service brokers, προγραμματισμό και πολιτικές καταμερισμού.

2.11.2 Έκδοση 2.0

Η τελευταία αναβάθμιση της έκδοσης 2.0, η 2.1.1, έγινε στις 10 Φεβρουαρίου 2010. Περιελάμβανε σημαντικές βελτιώσεις στον πυρήνα της προσομοίωσης, επιτρέποντας την βελτιωμένη επεκτασιμότητα και απόδοση των προσομοιώσεων. Τέλος, επέτρεψε την εισαγωγή και αφαίρεση των οντοτήτων κατά την διάρκεια της εκτέλεσης της προσομοίωσης. Αύξησε σημαντικά τα σενάρια, που μπορούν να χρησιμοποιηθούν στις προσομοιώσεις.

2.11.3 Έκδοση 3.0

Αυτή η έκδοση του CloudSim, έγινε διαθέσιμη στις 11 Ιανουαρίου 2011. Είναι η προτελευταία έκδοση του CloudSim και διορθώνει πολλά σφάλματα. Οι αναβαθμίσεις σε αυτή την έκδοση είναι καινούργιος προγραμματισμός εικονικών μηχανών, νέα δικτυακό μοντέλο του data center, νέα κατανομή των εικονικών μηχανών και πολιτικές επιλογής, νέα μοντέλα ενέργειας, παρακολούθηση του φόρτου εργασίας, υποστήριξη για εξωτερικούς φόρτους εργασίας και υποστήριξη για τερματισμό της προσομοίωσης από τον χρήστη. Κάποιες κλάσεις αφαιρέθηκαν από αυτή την έκδοση, όπως οι CloudCoordinator, Sensor, PowerPe και Power.PeList. Τέλος, έγιναν κάποιες αλλαγές στη Διεπαφή Προγραμματισμού Εφαρμογών - Application Programming Interface (API).

2.11.4 Έκδοση 4.0

Η τελευταία έκδοση έγινε διαθέσιμη αρκετά πρόσφατα, στις 24 Μαΐου 2016. Περιλαμβάνει αρκετές διορθώσεις σφαλμάτων στον κώδικα και προστέθηκε η υποστήριξη για εικονικοποίηση Container. [16]

Κεφάλαιο 3

Προσομοιωτής CloudAnalyst

> "Cloud is about how you do computing, not where you do computing."
>
> Paul Maritz, CEO της εταιρείας VMware

3.1 Εισαγωγή

Ο προσομοιωτής CloudAnalyst είναι ένα εργαλείο, το οποίο έχει αναπτυχθεί από ερευνητές του Πανεπιστημίου της Μελβούρνης και ο στόχος του είναι να υποστηρίξει την αξιολόγηση των εργαλείων των κοινωνικών δικτύων σύμφωνα με την γεωγραφική κατανομή των χρηστών και των data centers. Σε αυτό το εργαλείο, οι κοινότητες των χρηστών και τα data centers που υποστηρίζουν τα κοινωνικά δίκτυα χαρακτηρίζονται ανάλογα με την τοποθεσία τους. Αντίστοιχα λαμβάνονται/καταγράφονται παράμετροι, όπως είναι η εμπειρία του χρήστη κατά τη χρήση της εφαρμογής του κοινωνικού δικτύου και ο φόρτος στο data center.

Επειδή οι υποδομές νέφους είναι κατανεμημένες, οι εφαρμογές μπορεί να αναπτυχθούν σε διαφορετικές γεωγραφικές τοποθεσίες. Όμως ο τρόπος κατανομής των εφαρμογών που θα επιλεχθεί επηρεάζει την επίδοση για τους χρήστες που είναι μακρυά από το data center. Αν και οι υποδομές αυτές κάνουν ευκολότερη και φθηνότερη, την ανάπτυξη μεγάλης κλίμακας εφαρμογών, προσθέτουν παράλληλα νέα προβλήματα στους προγραμματιστές.

Καθώς οι εφαρμογές του Διαδικτύου είναι προσβάσιμες από χρήστες σε όλο τον κόσμο και επειδή η δημοτικότητα των εφαρμογών διαφέρει σε κάθε γεωγραφική περιοχή, είναι λογικό να διαφέρει και η εμπειρία χρήσης των εφαρμογών. Η ποσοτικοποίηση των επιπτώσεων του αριθμού των ταυτόχρονων χρηστών, της γεωγραφικής τοποθεσίας των σχετικών στοιχείων και του δικτύου στις εφαρμογές είναι δύσκολο να επιτευχθεί σε πραγματικές δοκιμές, λόγω της παρουσίας στοιχείων που δεν μπορούν να προβλεφθούν ή να ελεγχθούν από τους προγραμματιστές.

Ένας από τους βασικούς στόχους του CloudAnalyst είναι ο διαχωρισμός της διαδικασίας της πειραματικής προσομοίωσης από την διαδικασία προγραμματισμού, έτσι ώστε ο χρήστης που εκτελεί την μοντελοποίηση να εστιάσει στην πολυπλοκότητα της προσομοίωσης, χωρίς να σπαταλάει πολύ χρόνο στις τεχνικές λεπτομέρειες του προγραμματισμού ενός εργαλείου προσομοίωσης. Το CloudAnalyst επιτρέπει επίσης στον χρήστη της μοντελοποίησης, να εκτελεί επανειλημμένες προσομοιώσεις και να διεξάγει μια σειρά από πειράματα προσομοίωσης με μικρές μεταβολές στις παραμέτρους, με έναν πολύ εύκολο και γρήγορο τρόπο.

Τέλος, το CloudAnalyst είναι βασισμένο σε αρθρωτό σχεδιασμό και μπορεί εύκολα να επεκταθεί. Έχει αναπτυχθεί χρησιμοποιώντας τις ακόλουθες τεχνολογίες: την Java (το 100% του προσομοιωτή είναι υλοποιημένο στην πλατφόρμα της Java), το Java Swing (το γραφικό περιβάλλον χρήστη έχει δημιουργηθεί κάνοντας χρήστη των στοιχείων του Swing), το CloudSim (χρησιμοποιούνται τα χαρακτηριστικά του για μοντελοποίηση των data centers) και τέλος η SimJava όπου κάποια από τα χαρακτηριστικά αυτού του εργαλείου χρησιμοποιούνται αυτούσια στο CloudAnalyst. [17]

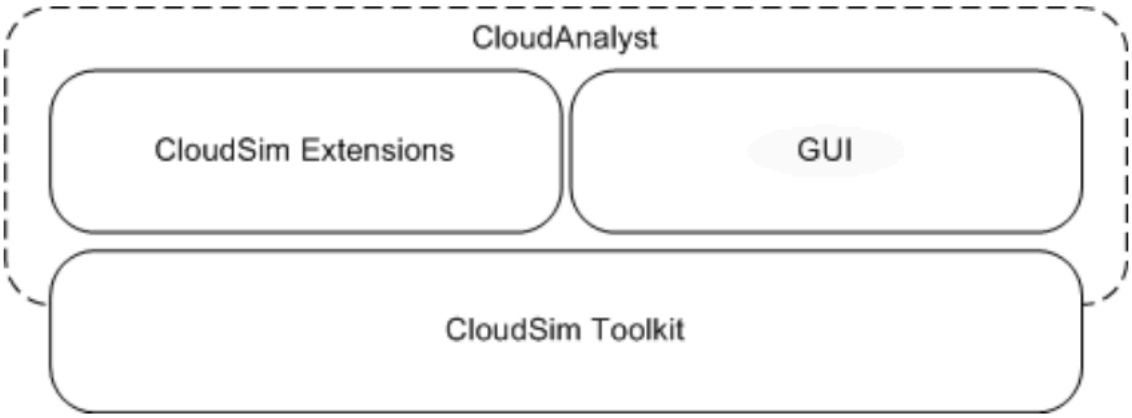
3.1.1 Κύρια χαρακτηριστικά

Το πρόγραμμα προσομοίωσης CloudAnalyst παρέχει αρκετά χαρακτηριστικά για την κατανόηση του υπολογιστικού νέφους. Μερικά είναι τα ακόλουθα: [18]

- **Ευκολία στην χρήση**. Το CloudAnalyst είναι πολύ εύκολο να εγκατασταθεί, καθώς περιλαμβάνει ένα πακέτο java και ξεκινάει απλά πατώντας πάνω στο εικονίδιο. Ένας προσομοιωτής πρέπει να παρέχει ένα γραφικό περιβάλλον, το οποίο να είναι εύκολο στη χρήση, κατανοητό και παράλληλα περιεκτικό.
- **Αποτελέσματα σε γραφικό περιβάλλον χρήστη**. Τα αποτελέσματα φαίνονται σε ένα γραφικό περιβάλλον χρήστη, το οποίο αποτελείται από πίνακες (γραμμές και στήλες), από γραφήματα και διαγράμματα που είναι αρκετά χρήσιμα και δημιουργούνται την ώρα της προσομοίωσης. Αυτή η γραφική απεικόνιση βοηθάει στην κατανόηση και την αναγνώριση των σημαντικών παραμέτρων και επίσης στη σύγκριση τους.
- **Δυνατότητα επανάληψης**. Το CloudAnalyst έχει την δυνατότητα να επαναλαμβάνει τα πειράματα, η οποία είναι μια σημαντική απαίτηση από κάθε προσομοιωτή. Με τον προσομοιωτή CloudAnalyst, εάν ένα πείραμα έχει κάποιες παραμέτρους και με την προσομοίωση παράγονται κάποια αποτελέσματα, τότε αυτά τα αποτελέσματα θα είναι τα ίδια κάθε φόρα που εκτελείται η ίδια προσομοίωση με τις ίδιες παραμέτρους στο ίδιο πείραμα. Χωρίς αυτή τη δυνατότητα, η προσομοίωση θα ήταν μια τυχαία ακολουθία από γεγονότα και όχι ένα ελεγχόμενο πείραμα.
- **Δυνατότητα αποθήκευσης των αποτελεσμάτων**. Ο προσομοιωτής έχει την δυνατότητα να αποθηκεύσει τα αποτελέσματα. Κάτι τέτοιο είναι χρήσιμο καθώς ο χρήστης μπορεί να αποθηκεύσει το πείραμα (μαζί με όλες τις παραμέτρους εισόδου και τις τιμές που πάρθηκαν κατά την διάρκεια της προσομοίωσης) σε ένα αρχείο. Στη συνέχεια, το αρχείο μπορεί να αποθηκευτεί στο σύστημα (προσωπικός υπολογιστής) ή σε κάποιο flash drive και να χρησιμοποιηθεί από άλλους υπολογιστές σε διαφορετικές τοποθεσίες.

3.2 Η αρχιτεκτονική του CloudAnalyst

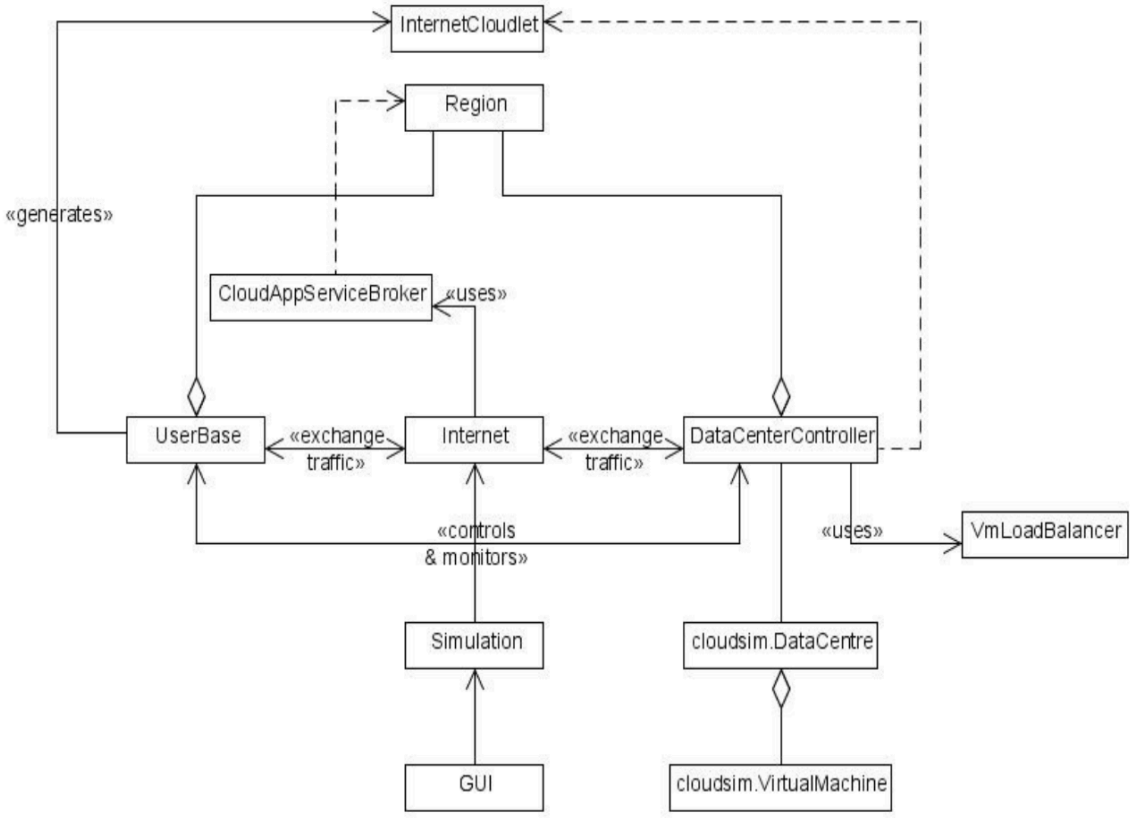
Το CloudAnalyst έχει αναπτυχθεί πάνω στη βάση του προσομοιωτή CloudSim, επεκτείνοντας την λειτουργικότητα του με τις έννοιες που μοντελοποιούν την συμπεριφορά του Internet και των εφαρμογών του Internet. Οι τρεις βασικές επεκτάσεις που έχουν παρουσιαστεί στο CloudAnalyst, σε σχέση με το CloudSim είναι το User Base, ο Data Center Controller και το Internet. Στην εικόνα 24 φαίνεται η αρχιτεκτονική του προσομοιωτή. Στη βάση παρατηρούμε ότι υπάρχει το εργαλείο προσομοίωσης CloudSim. Στο επόμενο επίπεδο ακολουθούν οι Επεκτάσεις (Extensions) και το Γραφικό περιβάλλον χρήστη - Graphical User Interface (GUI), τα οποία αποτελούν και τον προσομοιωτή CloudAnalyst.



Εικόνα 24: Η αρχιτεκτονική του CloudAnalyst [17]

Στις επόμενες υποενότητες περιγράφονται τα στοιχεία που χρησιμοποιούνται από τον προσομοιωτή και αναλύονται σε βάθος. Η εικόνα 25 δείχνει τα βασικά στοιχεία και τις κύριες οντότητες.

Ωστόσο, πριν γίνει η ανάλυση του κάθε στοιχείου καλό είναι να παρουσιαστεί η έννοια της Περιοχής (Region).



Εικόνα 25: Οι βασικές οντότητες του προσομοιωτή [19]

3.2.1 Region

Στο CloudAnalyst ο κόσμος έχει χωριστεί σε έξι περιοχές, οι οποίες καλούνται ”Regions” και ταυτίζονται με τις έξι ηπείρους του πλανήτη. Οι κύριες οντότητες όπως οι User Bases και τα Data Centers, ανήκουν σε μια από τις προαναφερθείσες περιοχές. Αυτή η γεωγραφική ομαδοποίηση χρησιμοποιείται για να διατηρηθεί ένα επίπεδο ρεαλιστικής απλότητας για την προσομοίωση μεγάλης κλίμακας που επιχειρείται στο CloudAnalyst.

3.2.2 Internet

Το Internet στο CloudAnalyst είναι μια αφηρημένη έννοια του πραγματικού Internet και έχει υλοποιημένα μόνο τα χαρακτηριστικά που είναι σημαντικά για την προσομοίωση. Μοντελοποιεί την δρομολόγηση της κίνησης σε όλο τον κόσμο κάνοντας χρήστη της κατάλληλης καθυστέρησης μετάδοσης και τις καθυστερήσεις μεταφοράς δεδομένων. Η καθυστέρηση μετάδοσης και το διαθέσιμο εύρος ζώνης μεταξύ των έξι περιοχών είναι παραμετροποιήσιμες τιμές.

3.2.3 Cloud Application Service Broker

Η δρομολόγηση της κίνησης μεταξύ των User Bases και των Data Centers είναι ελεγχόμενη από έναν Service Broker, ο οποίος αποφασίζει ποιο Data Center θα πρέπει να εξυπηρετήσει τις αιτήσεις από κάθε User Base. Σε αυτή την έκδοση του προσομοιωτή εφαρμόζονται τρεις τύποι διαφορετικών Service Brokers, όπου ο καθένας εφαρμόζει διαφορετική πολιτική δρομολόγησης.

1. Δρομολόγηση με βάση το Service Proximity. Σε αυτή την περίπτωση οι μικρές αποστάσεις είναι το συντομότερο μονοπάτι από μια User Base σε ένα Data Center και βασίζονται στην καθυστέρηση του δικτύου. Ο Service Broker θα δρομολογήσει την κίνηση των χρηστών στο κοντινότερο Data Center σε ότι αφορά την καθυστέρηση μετάδοσης.
2. Δρομολόγηση Performance Optimized. Σε αυτή την πολιτική δρομολόγησης ο Service Broker επιτηρεί ενεργά την επίδοση όλων των Data Centers και κατευθύνει την κίνηση στο data center που

υπολογίζει ότι θα δώσει τον καλύτερο χρόνο απόκρισης στον τελικό χρήστη, την ώρα που γίνεται η αίτηση.

3. Δυναμικά ρυθμισμένος δρομολογητής. Είναι μια επέκταση στη δρομολόγηση με βάση το Service Proximity, όπου η λογική δρομολόγησης είναι σχεδόν παρόμοια, αλλά ο service broker είναι επιφορτισμένος με την ευθύνη της κλιμάκωσης της ανάπτυξης της εφαρμογής με βάση τον φόρτο που αντιμετωπίζει. Αυτό επιτυγχάνεται αυξάνοντας ή μειώνοντας τον αριθμό των εικονικών μηχανών που ανατίθενται στο data center, ανάλογα με τον τωρινό χρόνο επεξεργασίας και συγκρινόμενο με τον καλύτερο χρόνο που έχει επιτευχθεί.

3.2.4 User Base

Η User Base μοντελοποιεί μια ομάδα χρηστών, που θεωρείται ως ενιαία μονάδα και η κύρια υποχρέωση της οποίας είναι να παράγει κίνηση για τις ανάγκες της προσομοίωσης. Μια μοναδική User Base μπορεί να αναπαριστά έναν αριθμό από χιλιάδες χρήστες, αλλά έχει ρυθμιστεί ως ενιαία μονάδα και η κίνηση που παράγεται σε ταυτόχρονες ”εκρήξεις” είναι αντιπροσωπευτική του μεγέθους της. Ο χρήστης της μοντελοποίησης μπορεί να επιλέξει την χρήση της User Base με έναν μόνο χρήστη, αλλά ιδανικά θα πρέπει να χρησιμοποιείται για να αναπαραστήσει ένα μεγάλο αριθμό χρηστών για την αποδοτικότητα της προσομοίωσης.

3.2.5 InternetCloudlet

Ένα InternetCloudlet αποτελεί μια ομάδα από τις αιτήσεις των χρηστών. Στο CloudAnalyst μπορεί να ρυθμιστεί ο αριθμός των αιτήσεων που ομαδοποιείται σε ένα μονό InternetCloudlet. Το InternetCloudlet μεταφέρει πληροφορίες όπως το μέγεθος των αιτημάτων της εντολής εκτέλεσης, το μέγεθος των αρχείων εισόδου και εξόδου, το id της αρχικής εφαρμογής και της εφαρμογής του στόχου που χρησιμοποιείται για τη δρομολόγηση από το Internet και τον αριθμό των αιτήσεων.

3.2.6 Data Center Controller

Ο Data Center Controller είναι η σημαντικότερη οντότητα στο CloudAnalyst. Ένας Data Center Controller αντιστοιχεί σε ένα μόνο αντικείμενο cloudsim.DataCenter και αναλαμβάνει τις δραστηριότητες διαχείρισης του data center όπως είναι η δημιουργία και η καταστροφή των εικονικών μηχανών και η δρομολόγηση των αιτήσεων που λαμβάνονται από τις User Bases, μέσω του Internet στις εικονικές μηχανές. Επίσης μπορεί να θεωρηθεί ως η πρώτη όψη που χρησιμοποιείται από το CloudAnalyst για την πρόσβαση στην καρδιά της λειτουργικότητας του εργαλείου CloudSim.

3.2.7 VmLoadBalancer

Ο Data Center Controller χρησιμοποιεί έναν VmLoadBalancer για να αποφασίσει ποια εικονική μηχανή θα πρέπει να αναλάβει το επόμενο Cloudlet ώστε να το επεξεργαστεί. Προς το παρόν, υπάρχουν τρεις VmLoadBalancers που υλοποιούν τρεις πολιτικές για εξισορρόπηση του φόρτου, οι οποίοι μπορεί να επιλεχθούν όπως απαιτείται από τον χρήστη της μοντελοποίησης.

1. **Round-robin Load Balancer**. Χρησιμοποιεί έναν απλό αλγόριθμο round-robin για την διάθεση των εικονικών μηχανών.
2. **Active Monitoring Load Balancer**. Εξισορροπεί τον φόρτο των εργασιών μεταξύ των διαθέσιμων εικονικών μηχανών με τέτοιο τρόπο ώστε να εξομαλύνει τον αριθμό των ενεργών εργασιών σε κάθε εικονική μηχανή την κάθε δεδομένη χρονική στιγμή.
3. **Throttled Load Balancer**. Σε αυτή την πολιτική εξασφαλίζει ότι μόνο ένας προκαθορισμένος αριθμός από Cloudlets θα ανατεθούν σε μια μοναδική εικονική μηχανή την κάθε χρονική στιγμή. Εάν υπάρχουν περισσότερες αιτήσεις από τον αριθμό που μπορούν να εξυπηρετήσουν οι εικονικές μηχανές στο data center, τότε κάποιες από αυτές θα πρέπει να μπουν στην ουρά, μέχρι να γίνει διαθέσιμη η επόμενη εικονική μηχανή.

3.2.8 GUI

Το Γραφικό περιβάλλον χρήστη έχει αναπτυχθεί ως μια ομάδα από εικόνες που επιτρέπουν στον χρήστη:

1. Να ορίσει τις παραμέτρους της προσομοίωσης, όπως:
   (a) τον λεπτομερή ορισμό των χαρακτηριστικών ενός Data Center, που περιλαμβάνει τις προδιαγραφές του υλικού της φάρμας των servers,
   (b) τον ορισμό των προδιαγραφών της ανάπτυξης της εφαρμογής, όπως είναι ο αριθμός των εικονικών μηχανών που πρέπει να ανατεθούν και σε ποιο data center και οι λεπτομερείς προδιαγραφές αυτών των εικονικών μηχανών,
   (c) τον ορισμό των User Bases και των χαρακτηριστικών τους, όπως είναι ο αριθμός των χρηστών, οι ώρες αιχμής και μη αιχμής της χρήσης και η συχνότητα παραγωγής της κίνησης,
   (d) τον ορισμό των ειδικών χαρακτηριστικών του Internet, που περιλαμβάνει την καθυστέρηση του δικτύου και το διαθέσιμο εύρος ζώνης,
   (e) τις παραμέτρους του προσομοιωτή που σχετίζονται με την επίδοση, όπως οι παράγοντες ομαδοποίησης των αιτήσεων των χρηστών όταν τα μηνύματα στέλνονται από τις User Bases και όταν τα μηνύματα ανατίθενται στις εικονικές μηχανές στο Data Center.
2. Να αποθηκεύσει και να φορτώσει τις ρυθμίσεις της προσομοίωσης,
3. Να εκτελέσει προσομοιώσεις έχοντας την επιλογή να τις ακυρώσει ενώ έχουν ξεκινήσει και τέλος,
4. Να δει και να αποθηκεύσει τα αποτελέσματα της προσομοίωσης με γραφικές εξόδους όπου απαιτείται.

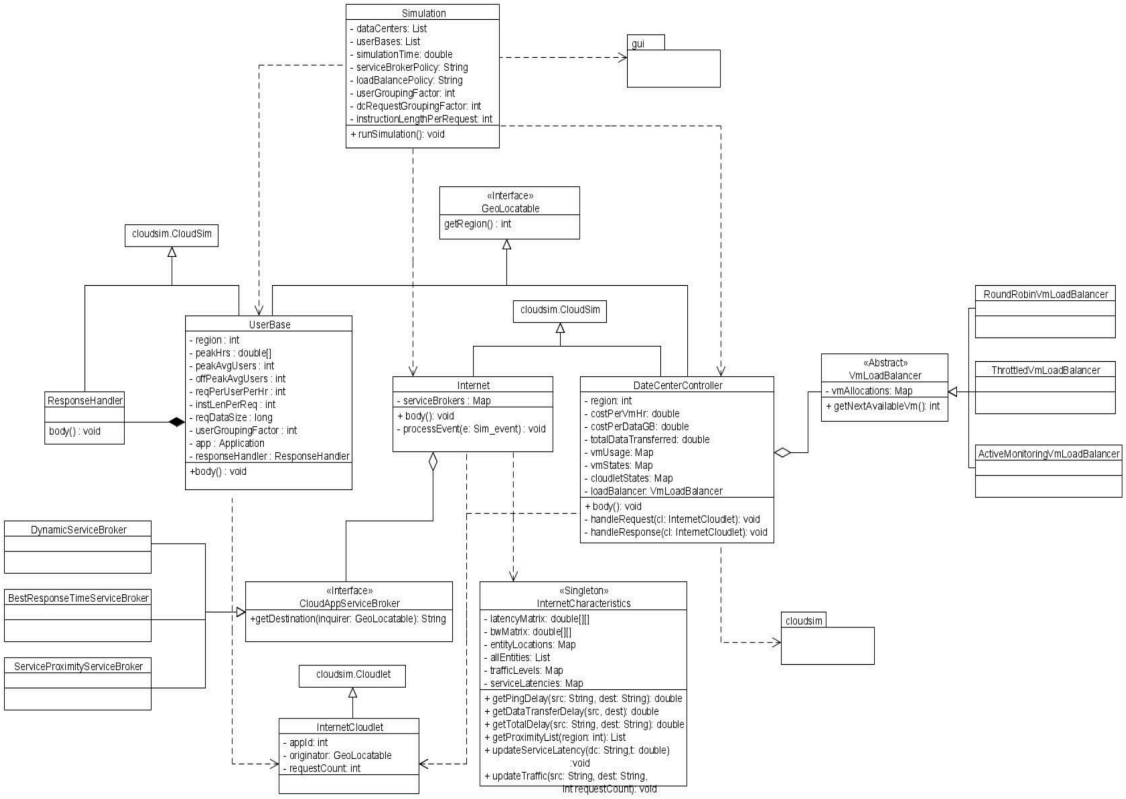
3.3 Οι κλάσεις του CloudAnalyst

Οι κλάσεις είναι υπεύθυνες για την μοντελοποίηση και την εκτέλεση των προσομοιώσεων. Το γραφικό περιβάλλον χρήστη είναι σχεδιασμένο ώστε να είναι χαλαρά συνδεδεμένο με το βασικό πλαίσιο της προσομοίωσης και έτσι φαίνεται ως ένα πακέτο στο βασικό διάγραμμα, όπως φαίνεται στην εικόνα 26.

Οι κλάσεις UserBase, Internet και DataCenterController επεκτείνουν την κλάση clousim.CloudSim, του εργαλείου CloudSim και είναι ενεργές οντότητες της προσομοίωσης, που τρέχουν σε ξεχωριστά νήματα. Η υπόλοιπη λειτουργία του Data Center (διαχείριση εικονικών μηχανών, εκτέλεση των εργασιών κ.α.) καλείται από την κλάση DataCenterController μέσω της cloudsim.DataCenter. [19]

Οι κυριότερες κλάσεις και οι ευθύνες τους είναι οι ακόλουθες:

- **GuiMain** - Είναι η βασική κλάση του GUI. Παρουσιάζει το γραφικό περιβάλλον χρήστη και ενεργεί ως ο διαχειριστής του τμήματος της εφαρμογής που φαίνεται. Διαχειρίζεται τις εναλλαγές των οθονών και των άλλων δραστηριοτήτων της διαπαφής χρήστη (UI).
- **Simulation** - Είναι υπεύθυνη για την αποθήκευση των παραμέτρων και τη δημιουργία και εκτέλεση της προσομοίωσης.
- **textbfUserBase** - Μοντελοποιεί την user base και παράγει κίνηση, εκπροσωπώντας τους χρήστες.
- **DataCenterController** - Ελέγχει τις δραστηριότητες του data center
- **Internet** - Μοντελοποιεί το διαδίκτυο και υλοποιεί την συμπεριφορά δρομολόγησης της κίνησης.
- **InternetCharacteristics** - Διατηρεί τα χαρακτηριστικά του διαδικτύου, συμπεριλαμβανομένων των καθυστερήσεων και των διαθέσιμων ευρών ζώνης μεταξύ των περιοχών, τα τρέχοντα επίπεδα της κίνησης και τα επίπεδα πληροφορίας της επίδοσης των data centers.
- **CloudAppServiceBroker** - Μοντελοποιεί τους service brokers.
- **VmLoadBalancer** - Μοντελοποιεί τη λογική εξισορρόπησης φόρτου που χρησιμοποιείται από τα data centers για την κατανομή των αιτήσεων στις εικονικές μηχανές.
- **UserBaseUIElement, DataCenterUIElement, MachineUIElement** - αυτές οι κλάσεις κρατάνε πληροφορίες για τις user bases, τα data centers και τις μηχανές για τη διεπαφή χρήστη, μέχρι να τις χρησιμοποιήσει η προσομοίωση για να δημιουργήσει τις αντίστοιχες οντότητες.



Εικόνα 26: Το διάγραμμα κλάσεων του Cloudanalyst [19]

3.4 Αλγόριθμοι

Για την εκτέλεση της προσομοίωσης χρησιμοποιούνται διάφοροι αλγόριθμοι από το CloudAnalyst. Τα data centers χρησιμοποιούν τον VMLoadBalancer και κατά αυτόν τον τρόπο μια οντότητα Data Center εξισορροπεί τον φόρτο των αιτήσεων μεταξύ όλων των διαθέσιμων εικονικών μηχανών. [18] [19]

3.4.1 Αλγόριθμος εξισορρόπησης φόρτου στις εικονικές μηχανές

Οι αλγόριθμοι που χρησιμοποιούνται αναφέρθηκαν στην υποενότητα 3.2.7. Ο πρώτος αλγόριθμος round-robin είναι αρκετά κατανοητός και δεν αναλύεται. Παρουσιάζεται όμως ο κώδικας 5, όπως υλοποιείται στην Java. Οι επόμενοι δύο αλγόριθμοι Throttled Load Balancer και Active Monitoring Load Balancer επεξηγούνται στις ακόλουθες υποενότητες. [19]

```
public class RoundRobinVmLoadBalancer extends VmLoadBalancer {
    private Map<Integer, VirtualMachineState> vmStatesList;
    private int currVm = -1;
    public RoundRobinVmLoadBalancer(Map<Integer, VirtualMachineState> vmStatesList){
        super();
        this.vmStatesList = vmStatesList;
    }
    public int getNextAvailableVm(){
        currVm++;
        if (currVm >= vmStatesList.size()){
            currVm = 0;
        }
        allocatedVm(currVm);
        return currVm;
    }}
```

Κώδικας 5: Ο αλγόριθμος Round-Robin

Throttled Load Balancer

Τα βήματα εκτέλεσης που ακολουθούνται από τον αλγόριθμο ThrottledVmLoadBalancer είναι τα παρακάτω:

1. Ο ThrottledVmLoadBalancer κρατάει έναν πίνακα περιεχομένου με τις εικονικές μηχανές και την κατάσταση τους (BUSY/AVAILABLE). Στην αρχή όλες οι εικονικές μηχανές είναι σε κατάσταση AVAILABLE.
2. Ο DataCenterController λαμβάνει μια νέα αίτηση.
3. Ο DataCenterController υποβάλει ένα ερώτημα στον ThrottledVmLoadBalancer για την επόμενη κατανομή.
4. Ο ThrottledVmLoadBalancer αναλύει τον πίνακα περιεχομένου από την αρχή μέχρι να βρει την πρώτη διαθέσιμη εικονική μηχανή ή ο πίνακας αναλύεται μέχρι το τέλος του.
   Εάν βρει μια εικονική μηχανή διαθέσιμη, τότε:
   (a) Ο ThrottledVmLoadBalancer επιστρέφει το id της εικονικής μηχανής στον DataCenterController.
   (b) Ο DataCenterController στέλνει μια αίτηση στην εικονική μηχανή που διαθέτει αυτό το id.
   (c) Ενημερώνει τον ThrottledVmLoadBalancer για την νέα ανάθεση.
   (d) Τέλος, ο ThrottledVmLoadBalancer ενημερώνει τον πίνακα που διαθέτει.

   Εάν δεν βρει διαθέσιμη εικονική μηχανή, τότε:
   (a) Ο ThrottledVmLoadBalancer επιστρέφει την τιμή -1.
   (b) Ο DataCenterController βάζει το αίτημα στην ουρά.
5. Όταν η εικονική μηχανή τελειώσει την επεξεργασία της αίτησης και ο DataCenterController λάβει ως απάντηση ένα Cloudlet, τότε ενημερώνει τον ThrottledVmLoadBalancer ότι δεν χρειάζεται την εικονική μηχανή.
6. Ο DataCenterController ελέγχει εάν υπάρχει αιτήσεις που περιμένουν στην ουρά. Εάν υπάρχουν τότε συνεχίζει από το βήμα 3.
7. Συνεχίζει από το βήμα 2

Active Monitoring Load Balancer

Αυτή η πολιτική εξισορρόπησης φόρτου προσπαθεί να διατηρήσει ίσους φόρτους εργασίας σε όλες τις διαθέσιμες εικονικές μηχανές. Ο αλγόριθμος είναι παρόμοιος με τον ThrottledVmLoadBalancer. [19]

1. Ο ActiveVmLoadBalancer διατηρεί έναν πίνακα περιεχομένου εικονικών μηχανών και τον αριθμό των αιτήσεων που έχουν κατανεμηθεί σε αυτές τις εικονικές μηχανές. Στην αρχή όλες έχουν 0 κατανομές.
2. Όταν έρχεται μια αίτηση από τον DataCenterController για την κατανομή μιας νέας εικονικής μηχανής, τότε αναλύεται ο πίνακας και αναγνωρίζεται η εικονική μηχανή με τον μικρότερο φόρτο. Εάν υπάρχουν πάνω από μια, τότε επιλέγεται η πρώτη που αναγνωρίζεται.
3. Ο ActiveVmLoadBalancer επιστρέφει το id της εικονικής μηχανής στον DataCenterController.
4. Ο DataCenterController στέλνει μια αίτηση στην εικονική μηχανή που διαθέτει αυτό το id.
5. Ενημερώνει τον ActiveVmLoadBalancer για την νέα ανάθεση.
6. Ο ActiveVmLoadBalancer ενημερώνει τον πίνακα κατανομών, αυξάνοντας τον αριθμό κατανομών για την συγκεκριμένη εικονική μηχανή.
7. Όταν η εικονική μηχανή τελειώσει την επεξεργασία της αίτησης και ο DataCenterController λάβει ως απάντηση ένα Cloudlet, τότε ενημερώνει τον ActiveVmLoadBalancer οτι δεν χρειάζεται την εικονική μηχανή.
8. Ο ActiveVmLoadBalancer ενημερώνει τον πίνακα κατανομών, μειώνοντας τον αριθμό κατανομών της εικονικής μηχανής.
9. Συνεχίζει από το βήμα 2.

```java
public class ActiveVmLoadBalancer extends VmLoadBalancer implements CloudSimEventListener {
    /** Holds the count current active allcoations on each VM */
    private Map<Integer, Integer> currentAllocationCounts;
    private Map<Integer, VirtualMachineState> vmStatesList;
    public ActiveVmLoadBalancer(DatacenterController dcb){
        dcb.addCloudSimEventListener(this);
        this.vmStatesList = dcb.getVmStatesList();
        this.currentAllocationCounts = Collections.synchronizedMap(new HashMap<Integer, Integer>());
    }

    /**
     * @return The VM id of a VM so that the number of active tasks on each VM is kept
     * evenly distributed among the VMs.
    */
    @Override
    public int getNextAvailableVm(){
        int vmId = -1;

        //Find the vm with least number of allocations

        //If all available vms are not allocated, allocated the new ones
        if (currentAllocationCounts.size() < vmStatesList.size()){
            for (int availableVmId : vmStatesList.keySet()){
                if (!currentAllocationCounts.containsKey(availableVmId)){
                    vmId = availableVmId;
                    break;
                }
            }
        } else {
            int currCount;
            int minCount = Integer.MAX_VALUE;

            for (int thisVmId : currentAllocationCounts.keySet()){
                currCount = currentAllocationCounts.get(thisVmId);
                if (currCount < minCount){
                    minCount = currCount;
                    vmId = thisVmId;
                }
            }
        }
        allocatedVm(vmId);
        return vmId;
    }
}
```

Κώδικας 6: Ο αλγόριθμος ActiveVmLoadBalancer

3.4.2 Οι αλγόριθμοι του Service Broker

Προς το παρόν υπάρχουν τρεις διαφορετικές υλοποιήσεις του Service Broker. [19]

Δρομολόγηση με βάση το Service Proximity

Είναι η απλούστερη υλοποίηση του Service Broker. [19]

1. Ο ServiceProximityServiceBroker διατηρεί έναν πίνακα περιεχομένου με όλα τα Data Centers συνοδευόμενα από την περιοχή τους.
2. Όταν το στοιχείο Internet λάβει ένα μήνυμα από μια user base, τότε υποβάλλει ένα ερώτημα στον ServiceProximityServiceBroker ζητώντας να μάθει τον παραλήπτη DataCenterController.
3. Ο ServiceProximityServiceBroker κάνει ανάκτηση της περιοχής του αποστολέα που έχει στείλει την αίτηση και ζητάει από το στοιχείο InternetCharacteristics την λίστα εγγύτητας για την συγκεκριμένη περιοχή. Η λίστα βάζει σε διάταξη τις υπόλοιπες περιοχές με τη σειρά της χαμηλότερης καθυστέρησης του δικτύου πρώτη όταν υπολογίζεται απο τη συγκεκριμένη περιοχή.
4. Ο ServiceProximityServiceBroker επιλέγει το πρώτο data center που βρίσκεται στην υψηλότερη περιοχή της λίστας εγγύτητας. Εάν περισσότερα του ενός βρίσκονται σε μια περιοχή, τότε επιλέγεται

τυχαία ένα από αυτά.

Δρομολόγηση Performance Optimized

Αυτή η πολιτική υλοποιείται από την κλάση BestResponseTimeServiceBroker, που επεκτείνει την κλάση ServiceProximityServiceBroker. [19]

1. Ο BestResponseTimeServiceBroker διατηρεί έναν πίνακα περιεχομένων με όλα τα διαθέσιμα Data Centers.
2. Όταν το στοιχείο Internet λάβει ένα μήνυμα από μια user base, τότε υποβάλλει ένα ερώτημα στον BestResponseTimeServiceBroker ζητώντας να μάθει τον παραλήπτη DataCenterController.
3. Ο BestResponseTimeServiceBroker αναγνωρίζει το κοντινότερο (από πλευράς καθυστέρησης) data center χρησιμοποιώντας τον αλγόριθμο ServiceProximityServiceBroker.
4. Τότε ο BestResponseTimeServiceBroker επαναλαμβάνει τη λίστα με όλα τα data centers και υπολογίζει τον τρέχοντα χρόνο απόκρισης σε κάθε data center, με την σειρά:
   (a) Ζητάει το στοιχείο InternetCharacteristics για τον τελευταίο καταγεγραμμένο χρόνο επεξεργασίας.
   (b) Εάν αυτός ο χρόνος έχει καταγραφεί πριν από ένα προκαθορισμένο κατώφλι, τότε ο χρόνος επεξεργασίας για το data center γίνεται 0. Αυτό σημαίνει ότι το data center είναι αδρανές για μια περίοδο μέχρι να πιάσει το κατώφλι.
   (c) Η καθυστέρηση του δικτύου από το στοιχείο InternetCharacteristics προστίθεται στην τιμή που έφτασε από τα παραπάνω βήματα.
5. Εάν ο ελάχιστος εκτιμώμενος χρόνος είναι για το κοντινότερο data center, τότε ο BestResponseTimeServiceBroker το επιλέγει. Αλλιώς, ο BestResponseTimeServiceBroker επιλέγει είτε το κοντινότερο data center ή εκείνο με τον ελάχιστο χρόνο απόκρισης με πιθανότητα 50:50.

Δυναμικός αλγόριθμος Service Broker

Ο αλγόριθμος DynamicServiceBroker έχει υλοποιηθεί από την επέκταση του αλγορίθμου ServiceProximityServiceBroker ή του καλύτερου αλγορίθμου ResponseTimeServiceBroker. [19] Τα βήματα που πραγματοποιούνται έχουν ως εξής:

1. Ο DynamicServiceBroker διατηρεί δύο λίστες: η μια έχει τους καλύτερους χρόνους απόκρισης που έχουν καταγραφεί έως εκείνη τη στιγμή για κάθε Data Center και η άλλη έχει όλα τα Data Centers.
2. Όταν το στοιχείο Internet λάβει ένα μήνυμα από μια user base, τότε υποβάλλει ένα ερώτημα στον DynamicServiceBroker ζητώντας να μάθει τον παραλήπτη DataCenterController.
3. Ο αλγόριθμος του DynamicServiceBroker κάνει χρήση των δύο αλγορίθμων, BestResponseTimeServiceBroker και ServiceProximityServiceBroker για να αναγνωρίσει τον προορισμό.
4. Τέλος, εάν ο τωρινός καταγεγραμμένος χρόνος απόκρισης είναι καλύτερος από τον αρχικό, τότε ο DynamicServiceBroker ενημερώνει τα αρχεία των καλύτερων καταγεγραμμένων χρόνων απόκρισης.

3.5 Εγκατάσταση του CloudAnalyst

Ο προσομοιωτής μπορεί να τρέξει σε όλα τα λειτουργικά συστήματα, καθώς είναι υλοποιημένος με την γλώσσα προγραμματισμού Java. Για την εγκατάστασή του απαιτείται η ύπαρξη των παρακάτω προαπαιτούμενων:

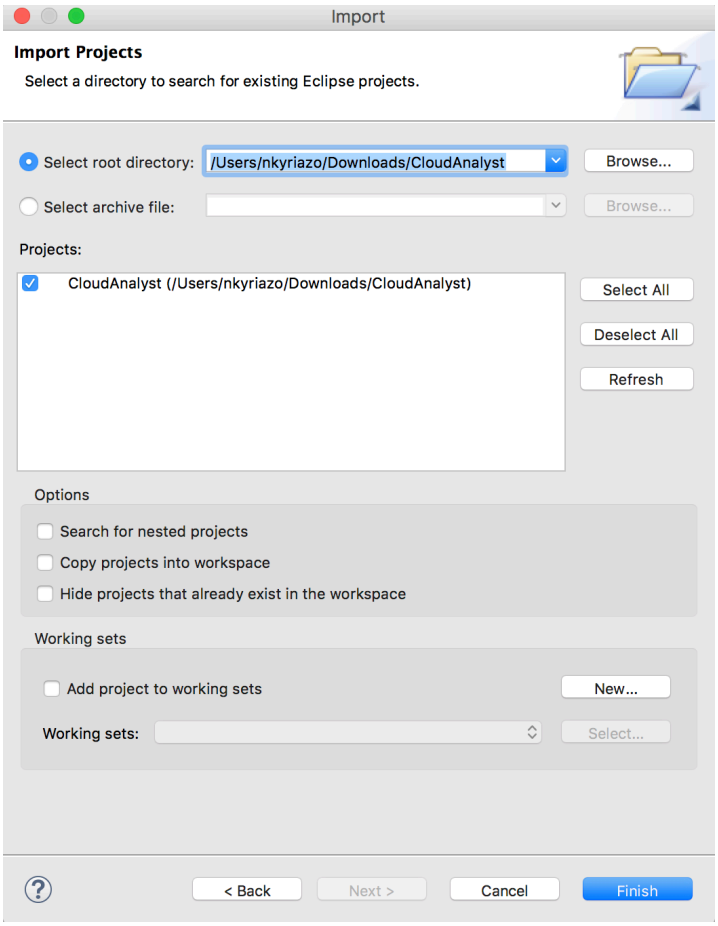
1. Εγκατεστημένο το Eclipse ή το Netbeans IDE, για την προβολή και επεξεργασία του κώδικα του προσομοιωτή. Εάν ο χρήστης θέλει απλά να κάνει προσομοιώσεις χωρίς να αλλάξει τον κώδικα, αυτό το βήμα δεν είναι απαραίτητο.
2. Το πακέτο του προσομοιωτή να βρίσκεται σε ένα τοπικό φάκελο. Το πακέτο μπορεί να το κατεβάσει ο χρήστης από τη σελίδα `http://www.cloudbus.org/cloudsim/CloudAnalyst.zip`

Μόλις ο χρήστης κατεβάσει το πακέτο του προσομοιωτή σε μορφή .zip, τότε θα πρέπει να το εξαγάγει σε κάποιο φάκελο. Η δομή καταλόγου του προσομοιωτή φαίνεται στον πίνακα 15.

cloudanalyst/
source/
test/
resources/
JRE System Library/
Referenced Libraries/
config/
jars/
javadoc/
readme.txt/
run.bat/

Πίνακας 2: Η δομή καταλόγου του CloudAnalyst

Για την εισαγωγή του project στο Eclipse, ο χρήστης πρέπει να επιλέξει μέσα από το IDE τις επιλογές *File-> Import-> GeneralExisting Projects into Workspace* και στην επόμενη οθόνη να βρει τον φάκελο του CloudAnalyst. Τότε πατώντας το *Finish*, όπως φαίνεται στην εικόνα 27 ολοκληρώνεται η εισαγωγή του project.



Εικόνα 27: Εισαγωγή του CloudAnalyst στο Eclipse

3.6 Χρήση του προσομοιωτή

Σε αυτή την ενότητα παρουσιάζονται οι ρυθμίσεις και οι οθόνες του προσομοιωτή CloudAnalyst, έτσι ώστε ο χρήστης να έχει μια πλήρη εικόνα πριν ξεκινήσει τα πειράματά του. Το γραφικό περιβάλλον είναι κατανοητό, εύκολο στη χρήση και έχει υλοποιηθεί με την Java Swing.

3.6.1 Βασικές ρυθμίσεις και οθόνες προσομοίωσης

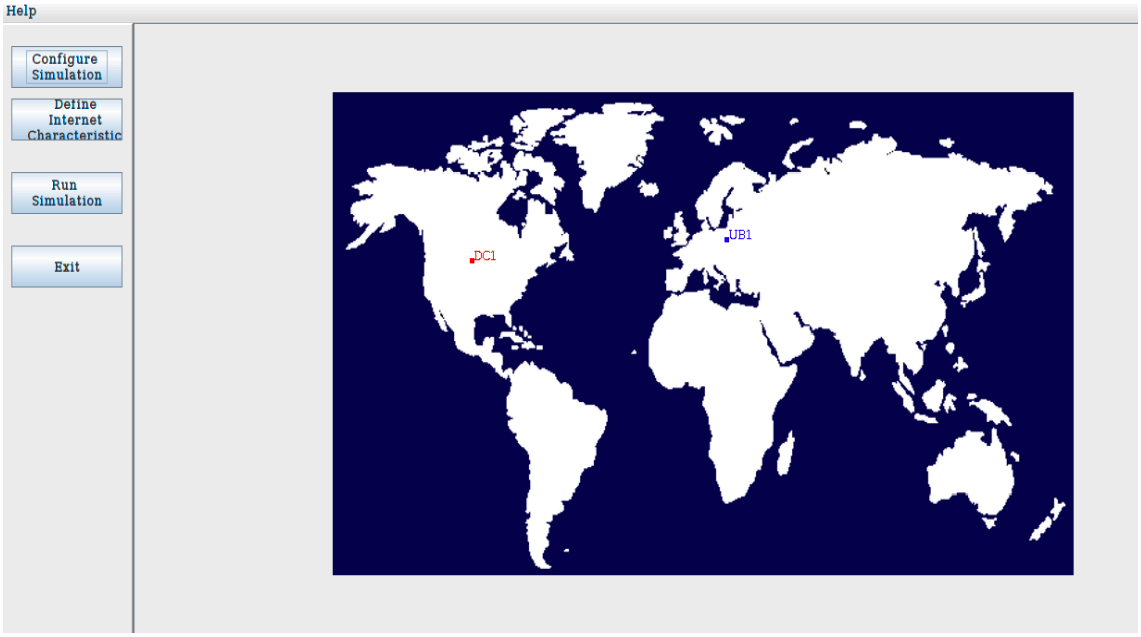
Για την εκκίνηση της προσομοίωσης, ο χρήστης πρέπει να συμπληρώσει τις ακόλουθες παραμέτρους του προγράμματος.

1. Ορισμός των User Bases. Είναι οι οντότητες που ορίζουν τους χρήστες της εφαρμογής, την γεωγραφική κατανομή τους και τις υπόλοιπες ιδιότητες όπως ο αριθμός των χρηστών, η συχνότητα χρήσης και το μοτίβο χρήσης π.χ. οι ώρες αιχμής. Αυτές οι ρυθμίσεις υπάρχουν στην κεντρική καρτέλα της οθόνης Configure Simulation.
2. Ορισμός των Data Centers. Στην καρτέλα Data Centers της οθόνης Configure Simulation ορίζονται τα Data Centers που χρειάζονται για την προσομοίωση. Ορίζεται το υλικό και τα λογιστικά στοιχεία, όπως το κόστος των εικονικών μηχανών.
3. Κατανομή των εικονικών μηχανών για την εφαρμογή σε Data Centers. Μόλις δημιουργηθούν τα Data Centers, πρέπει να κατανεμηθούν οι εικονικές μηχανές σε αυτά για την εφαρμογή προσομοίωσης χρησιμοποιώντας την κεντρική καρτέλα της οθόνης Configure Simulation. Ένα Data Center που έχει οριστεί στο βήμα 2, δεν περιλαμβάνεται στην προσομοίωση εκτός εάν κατανεμηθεί σε αυτό το βήμα. Ο χρήστης μπορεί να κατανείμει πολλούς τύπους εικονικών μηχανών στο ίδιο Data Center.
4. Έλεγχος και ρύθμιση των προχωρημένων παραμέτρων στην καρτέλα Advanced.
5. Έλεγχος και ρύθμιση του πίνακα της καθυστέρησης δικτύου και του εύρους ζώνης στην οθόνη Define Internet Characteristics.

Οι οθόνες του CloudAnalyst είναι το σημείο εισόδου των παραμέτρων για την εκτέλεση της προσομοίωσης. Ακολουθούν αναλυτικά όλες οι οθόνες και οι ρυθμίσεις τους.

3.6.2 Κύρια Οθόνη

Όταν ο χρήστης κάνει εκκίνηση του προγράμματος, πατώντας το cloudanalyst.jar στον φάκελο jars, εμφανίζεται η εικόνα 28. Στο αριστερό μέρος υπάρχουν οι επιλογές που οδηγούν στις οθόνες για την εισαγωγή των παραμέτρων και στο δεξί μέρος εμφανίζεται ο παγκόσμιος χάρτης.



Εικόνα 28: Η κύρια οθόνη του CloudAnalyst

Το CloudAnalyst χωρίζει τον κόσμο σε έξι περιοχές που συμπίπτουν με τις έξι ηπείρους. Για λόγους απλότητας, οι τοποθεσίες των στοιχείων στην προσομοίωση αναγνωρίζονται μόνο με την περιοχή τους.

Οι επιλογές του αριστερού μέρους είναι:

1. **Configure Simulation**. Είναι η επιλογή που οδηγεί στην οθόνη Configure Simulation.
2. **Define Internet Characteristics**. Οδηγεί στην οθόνη Internet Characteristics.

3. **Run Simulation**. Πατώντας αυτή την επιλογή, ο χρήστης ξεκινάει την προσομοίωση.
4. **Exit**. Έξοδος από το πρόγραμμα.

3.6.3 Οθόνη Configure Simulation

Η συγκεκριμένη οθόνη έχει τρεις καρτέλες, την Main Configuration, την Data Center Configuration και την Advanced.

Καρτέλα Main Configuration

Οι επιλογές των ρυθμίσεων που υπάρχουν στη βασική καρτέλα, όπως φαίνονται στην εικόνα 29 είναι:

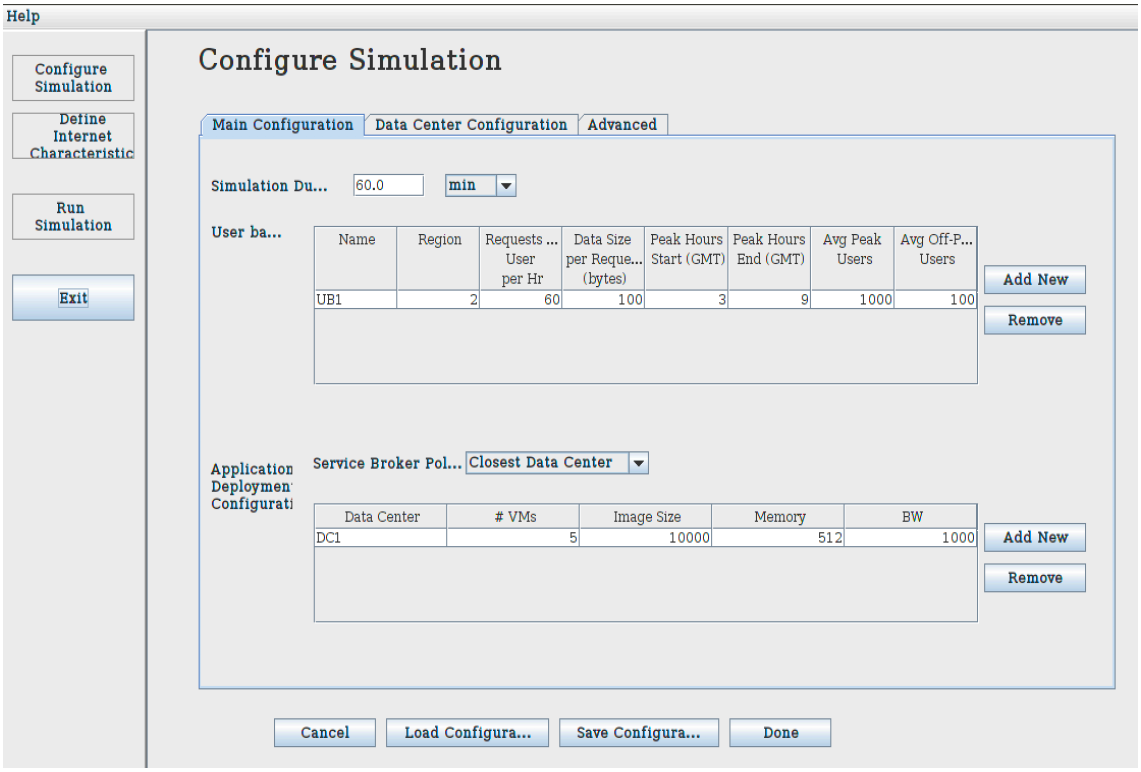
1. Επιλογή Simulation Time: Είναι η διάρκεια της προσομοίωσης, σε λεπτά, ώρες ή ημέρες.
2. Πίνακας User Bases: Είναι ο πίνακας που περιέχει όλες τις user bases για την προσομοίωση. Κάθε user base έχει τα παρακάτω πεδία:
   (a) Name
   (b) Region
   (c) Requests per user per hour
   (d) Data size per request
   (e) Peak hours
   (f) Average users during peak hours
   (g) Average users during off-peak hours
3. Πίνακας Application Deployment Configuration: Ο πίνακας αυτός περιέχει πόσες εικονικές μηχανές έχουν κατανεμηθεί για την εφαρμογή σε κάθε data center από την καρτέλα Data Centers, μαζί με τις λεπτομέρειες της εικονικής μηχανής. Τα πεδία που περιλαμβάνονται είναι:
   (a) Data Center: Ο χρήστης επιλέγει τα data centers που έχουν δημιουργηθεί στην καρτέλα Data Center.
   (b) Number of VMs: Είναι ο αριθμός των εικονικών μηχανών που θα κατανεμηθούν στην εφαρμογή από το επιλεγμένο data center.
   (c) Image Size: Είναι το μέγεθος του image μιας εικονικής μηχανής, εκφρασμένο σε bytes.
   (d) Memory: Είναι το μέγεθος της μνήμης, που είναι διαθέσιμη για μια εικονική μηχανή.
   (e) BW: Είναι το μέγεθος του εύρους ζώνης, που είναι διαθέσιμο για μια εικονική μηχανή.
4. Επιλογή Service Broker Policy: Επιτρέπει στον χρήστη να επιλέξει την πολιτική του Service Broker, ανάμεσα στα data centers, η οποία αποφασίζει ποιο data center θα λάβει την κίνηση από ποια user base. Οι διαθέσιμες πολιτικές είναι:
   (a) Closest Data Center: Είναι το data center με την ελάχιστη δικτυακή καθυστέρηση.
   (b) Optimize response time: Αυτή η πολιτική προσπαθεί να εξισορροπήσει τον φόρτο μεταξύ των data centers, όταν ένα data center υπερφορτώνεται.

Το κουμπί Save Configuration επιτρέπει στον χρήστη να αποθηκεύσει τις ρυθμίσεις σε ένα αρχείο. Τα αρχεία έχουν κατάληξη .sim. Ομοίως, το κουμπί Load Configuration μπορεί να φορτώσει ένα αποθηκευμένο αρχείο.

Καρτέλα Data Center Configuration

Η καρτέλα Data Center Configuration επιτρέπει στον χρήστη να ορίσει τις παραμέτρους του data center, όπως φαίνεται στην εικόνα 30. Με τα κουμπιά Add/Remove μπορεί να προσθέσει και να αφαιρέσει ένα data center. Τα πεδία που υπάρχουν σε αυτή την καρτέλα είναι:

1. Name
2. Region
3. Architecture
4. Operating System
5. Virtual Machine Monitor
6. Cost per VM Hour
7. Cost per 1Mb Memory Hour



Εικόνα 29: Οι ρυθμίσεις στην καρτέλα Main Configuration

8. Storage cost per Gb
9. Data Transfer cost per Gb
10. Number of servers

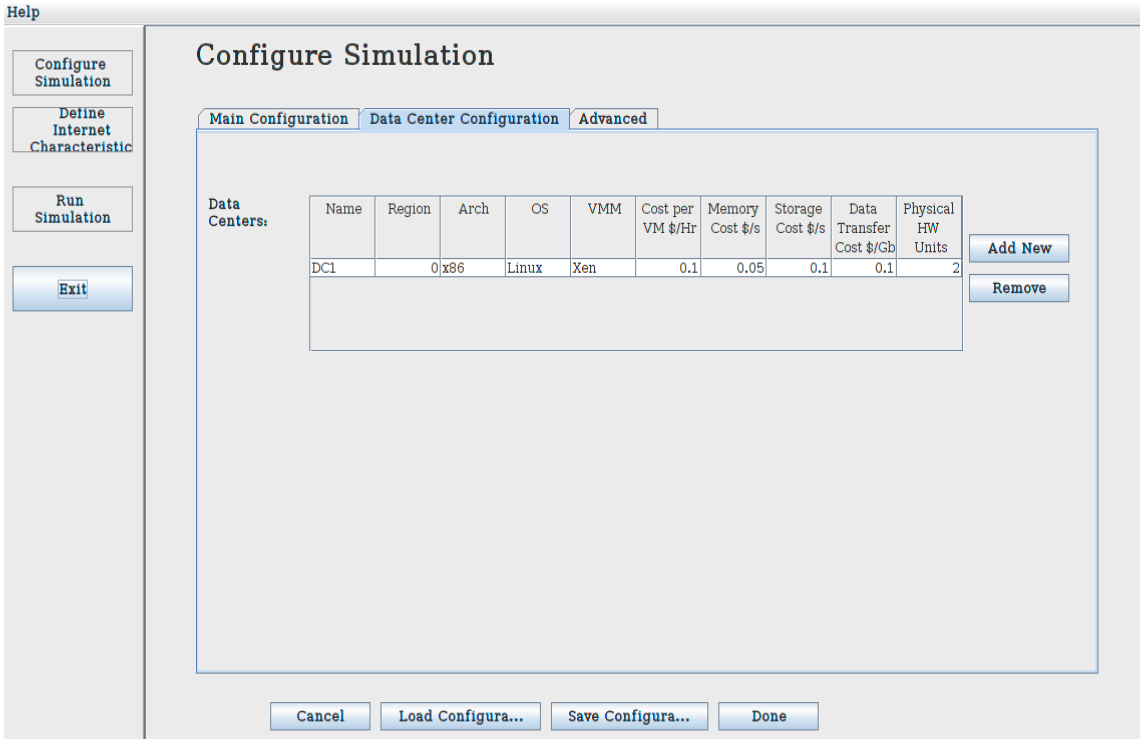
Όταν επιλεχθεί ένα data center από τον πρώτο πίνακα, τότε εμφανίζεται ένας επιπλέον πίνακας, με τις λεπτομέρειες των server στο συγκεκριμένο data center. στον νέο πίνακα υπάρχουν τα παρακάτω πεδία:

1. Machine id
2. Memory
3. Storage
4. Available network bandwidth
5. Number of processors
6. Processor speed (MIPS)
7. VM allocation policy

Καρτέλα Advanced

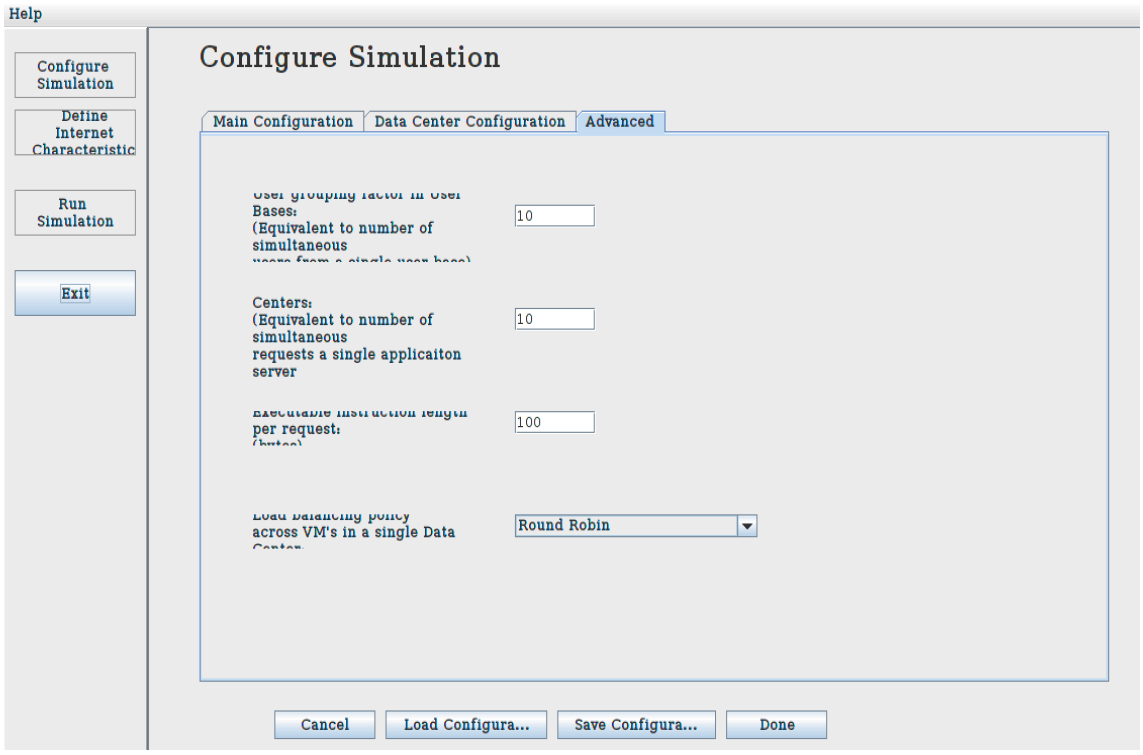
Στη συγκεκριμένη καρτέλα υπάρχουν διάφορες σημαντικές παράμετροι, που εφαρμόζονται σε όλη την προσομοίωση.

1. User Grouping Factor in User Bases: Είναι η παράμετρος, που υποδεικνύει στον προσομοιωτή πόσοι χρήστες θα αντιμετωπιστούν ως ομάδα για την παραγωγή της κίνησης. Ο αριθμός που ορίζεται σε αυτό το πεδίο, θα είναι ο αριθμός των αιτήσεων που αναπαριστώνται από ένα InternetCloudlet.
2. Request Grouping Factor in Data Centers: Είναι η παράμετρος, που υποδεικνύει στον προσομοιωτή πόσες αιτήσεις θα αντιμετωπιστούν ως μια μονάδα για επεξεργασία. Αυτές οι αιτήσεις ομαδοποιούνται και ανατίθενται σε μια εικονική μηχανή.
3. Executable instruction length in bytes: Είναι η βασική παράμετρος που επηρεάζει το μήκος εκτέλεσης μιας αίτησης.
4. Load balancing policy: Είναι η πολιτική εξισορρόπησης φόρτου, που χρησιμοποιείται από τα data centers για την κατανομή των αιτήσεων στις εικονικές μηχανές. Οι διαθέσιμες πολιτικές είναι:



Εικόνα 30: Οι ρυθμίσεις στην καρτέλα Data Center Configuration

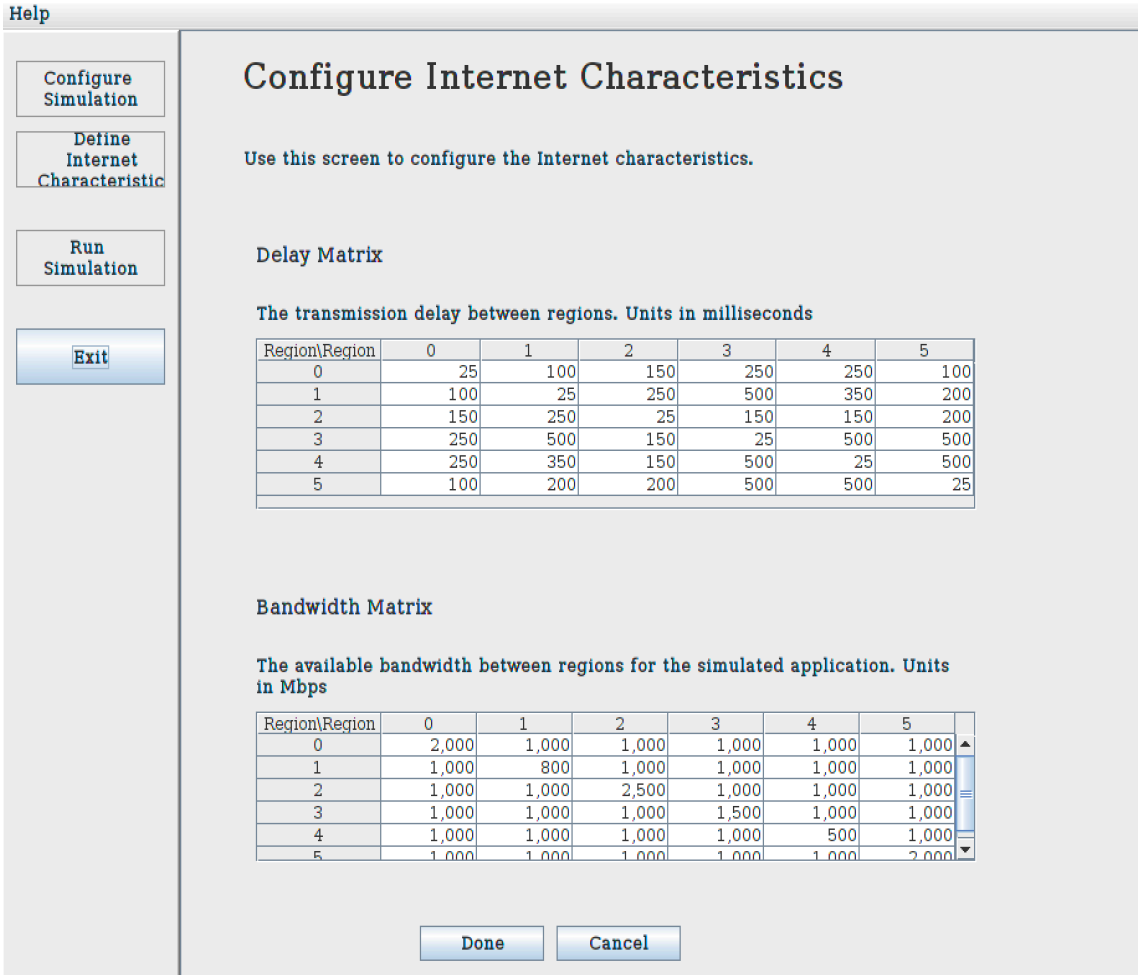
(a) Round-Robin
(b) Equally Spread Current Execution Load
(c) Throttled



Εικόνα 31: Οι ρυθμίσεις στην καρτέλα Advanced

3.6.4 Οθόνη Internet Characteristics

Η οθόνη Internet Characteristics, όπως φαίνεται στην εικόνα 32, χρησιμοποιείται για να καθοριστεί η καθυστέρηση του δικτύου και το εύρος ζώνης. Υπάρχουν δύο πίνακες για αυτές τις δύο κατηγορίες.



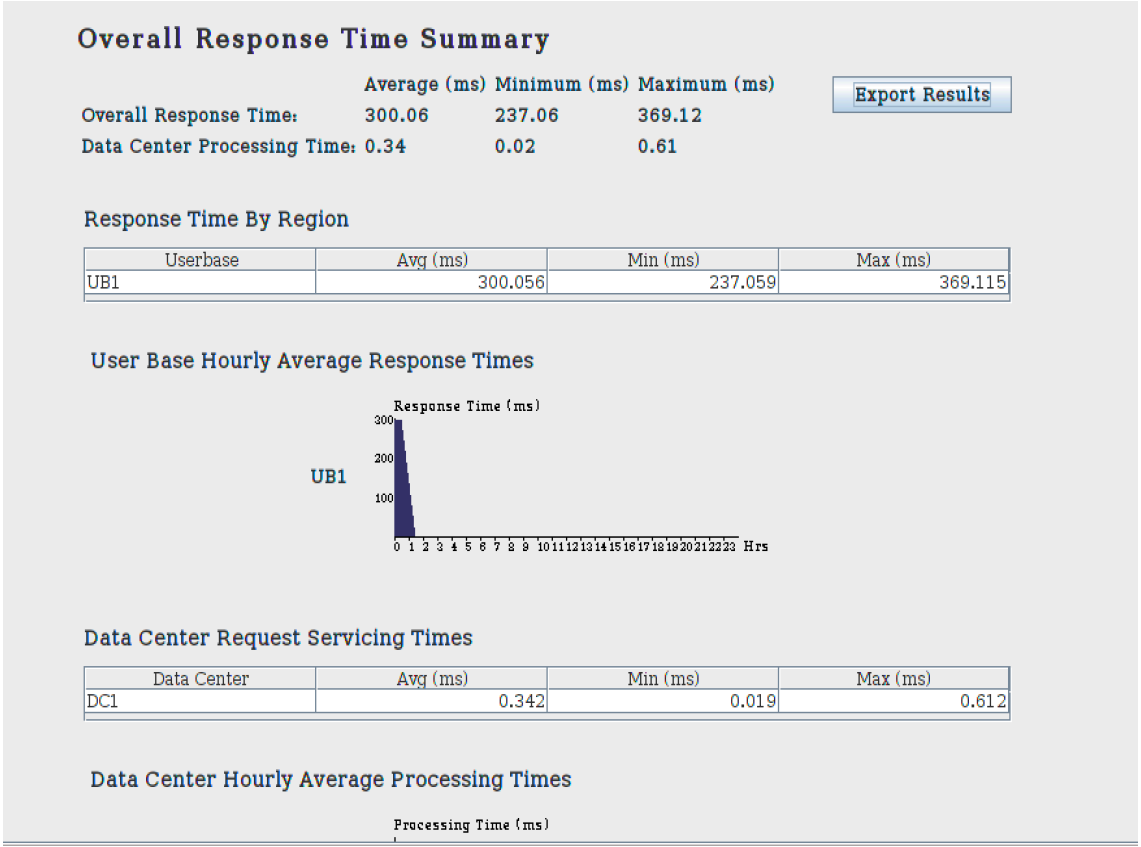
Εικόνα 32: Οι ρυθμίσεις στην οθόνη Internet Characteristics

3.7 Προσομοίωση

Το τελευταίο και σημαντικότερο κομμάτι του προσομοιωτή CloudAnalyst αφορά την προσομοίωση και την εξαγωγή των αποτελεσμάτων. Για την εκκίνηση της, ο χρήστης που βρίσκεται στην κεντρική οθόνη της εικόνας 28, θα πρέπει να πατήσει το κουμπί Run Simulation. Έπειτα μόλις τελειώσει η προσομοίωση, εμφανίζεται η καρτέλα που φαίνεται στην εικόνα 33 με πολλά ενδιαφέροντα στοιχεία και αποτελέσματα.

Τα στοιχεία που εμφανίζονται στα αποτελέσματα περιλαμβάνουν τα ακόλουθα:

1. Τον συνολικό χρόνο απόκρισης, από όλες τις user bases.
2. Τον χρόνο απόκρισης από κάθε user base, σε μορφή πίνακα.
3. Τον χρόνο απόκρισης από κάθε user base, σε γραφική μορφή χωρισμένο στις 24 ώρες της ημέρας.
4. Τον χρόνο εξυπηρέτησης από κάθε data center, σε μορφή πίνακα.
5. Τον χρόνο εξυπηρέτησης από κάθε data center, σε γραφική μορφή χωρισμένο στις 24 ώρες της ημέρας.
6. Τον φόρτο του data center (ο αριθμός των αιτήσεων που εξυπηρετήθηκαν), σε γραφική μορφή χωρισμένο στις 24 ώρες της ημέρας.
7. Τις λεπτομέρειες κόστους.



Εικόνα 33: Τα αποτελέσματα της προσομοίωσης

3.7.1 Προσομοίωση μιας μεγάλης διαδικτυακής εφαρμογής στο Cloud

Ένας τυπικός τύπος εφαρμογής ευρείας κλίμακας με πολλούς χρήστες που μπορεί να ωφεληθεί από το Cloud, είναι οι εφαρμογές μέσων κοινωνικής δικτύωσης. Σε αυτό το παράδειγμα χρησιμοποιείται η σελίδα κοινωνικής επαγγελματικής δικτύωσης LinkedIn, στην οποία ο χρήστης μπορεί να εισέλθει, να δημιουργήσει το online βιογραφικό του, να συνδεθεί με συναδέλφους και διάφορους επαγγελματίες από όλο τον κόσμο και τέλος να αναζητήσει εργασία.

Από τα στατιστικά στοιχεία της σελίδας του LinkedIn [20], αναφορικά με τους χρήστες που το χρησιμοποιούν ανά τον κόσμο, προκύπτει ο πίνακας 5 για το έτος 2016. Ο τελικός αριθμός των χρηστών, υπολογίστηκε από τον χάρτη που βρίσκεται στην ιστοσελίδα του LinkedIn.

Region	**CloudAnalyst Region id**	**Χρήστες**
North America	0	155M
South America	1	65M
Europe	2	110M
Asia	3	79M
Africa	4	5M
Oceania	5	24M

Πίνακας 3: Αριθμός χρηστών ανά περιοχή

Για τις ανάγκες της προσομοίωσης, θα χρησιμοποιηθεί περίπου το 1/10 των πραγματικών χρηστών του LinkedIn.

Ακόμα ορίζονται 6 User Bases, οι οποίες αναπαριστούν τις 6 παραπάνω περιοχές όπου βρίσκονται οι χρήστες. Ο πίνακας 4 δείχνει όλες τις παραμέτρους που ορίζονται στον προσομοιωτή.

Για ευκολία γίνονται κάποιες παραδοχές, όπως ότι οι περισσότεροι χρήστες χρησιμοποιούν την εφαρμογή τα απογεύματα μετά την δουλειά για 2 ώρες και ότι κάθε user base ανήκει σε μια ζώνη ώρας [21].

User Base	Region	Time Zone	Peak Hours (Local Time)	Peak Hours GMT	Χρηστες σε ώρες αιχμής	Χρήστες σε ώρες μη αιχμής
North America	0	GMT - 6.00	7.00–9.00 pm	13:00-15:00	775,000	77,500
South America	1	GMT - 4.00	7.00–9.00 pm	15:00-17:00	325,000	32,500
Europe	2	GMT + 1.00	7.00–9.00 pm	20:00-22:00	550,000	55,000
Asia	3	GMT + 6.00	7.00–9.00 pm	01:00-03:00	395,000	39,500
Africa	4	GMT + 2.00	7.00–9.00 pm	21:00-23:00	25,000	2,500
Oceania	5	GMT - 10.00	7.00–9.00 pm	09:00-11:00	120,000	12,000

Πίνακας 4: Οι ρυθμίσεις των User Bases

Επίσης, μόνο το 5% των συνολικών χρηστών της εφαρμογής θα είναι συνδεδεμένοι ταυτόχρονα τις ώρες αιχμής και μόνο το 1/10 αυτού του αριθμού στις ώρες μη αιχμής. Τέλος, γίνεται η υπόθεση ότι κάθε χρήστης κάνει μια αίτηση ανά 5 λεπτά.

Όσον αφορά την κοστολόγηση των υπηρεσιών για τη φιλοξενία της εφαρμογής, ας υποθέσουμε ότι χρησιμοποιείται το Amazon EC2 [22] και η τιμολόγηση είναι κοντά στις πραγματικές τιμές.

Κόστος της εικονικής μηχανής ανά ώρα	$0.10
Κόστος ανά 1Gb μεταφοράς δεδομένων	$0.10

Πίνακας 5: Κοστολόγηση υπηρεσιών

Τέλος, στον πίνακα 6 φαίνονται οι υπόλοιπες ρυθμίσεις που έχουν γίνει στο CloudAnalyst για την εκκίνηση της προσομοίωσης του παραδείγματος. Με αυτές τις παραμέτρους ολοκληρώνεται η ρύθμιση του προσομοιωτή και ο χρήστης μπορεί να τρέξει τα σενάρια που επιθυμεί.

Παράμετροι	Τιμές
VM Image Size	100000
VM Memory	10240 Mb
VM Bandwidth	10000
Data Center – Architecture	x86
Data Center – OS	Linux
Data Center – VMM	Xen
Data Center – Number of Machines	20
Data Center – Memory per Machine	2048 Mb
Data Center – Storage per Machine	100000 Mb
Data Center – Available BW per Machine	10000
Data Center – Number of processors per machine	4
Data Center – Processor speed	10000 MIPS
Data Center – VM Policy	Time Shared
User Grouping Factor	1000
Request Grouping Factor	100
Executable Instruction Length	250

Πίνακας 6: Οι τιμές των παραμέτρων

3.7.2 Σενάριο 1ο - Η εφαρμογή φιλοξενείται σε ένα data center

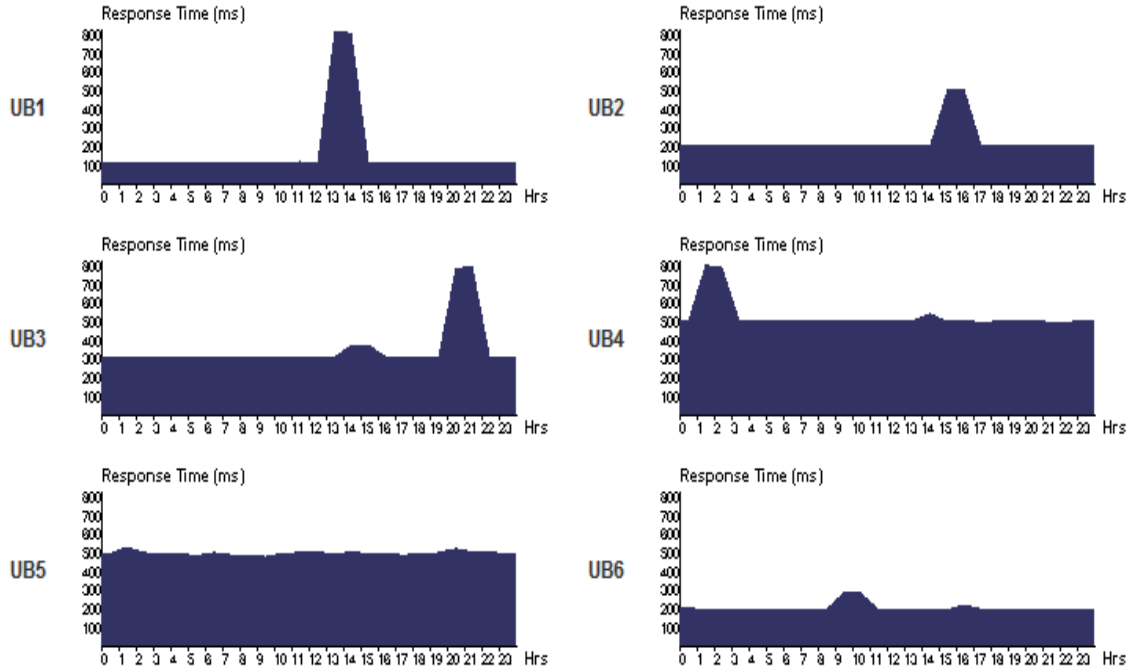
Ξεκινώντας την προσομοίωση η εφαρμογή φιλοξενείται σε ένα data center, το οποίο βρίσκεται στο Region 0 με 25 εικονικές μηχανές και δέχεται αιτήσεις από τις 6 User Bases.

Ο συνολικός χρόνος απόκρισης φαίνεται στον πίνακα 7

	Avg (ms)	Min (ms)	Max (ms)
Συνολικός χρόνος απόκρισης	481.27	42.51	1106.97
Χρόνος επεξεργασίας στο data center	233.75	0.39	891.39

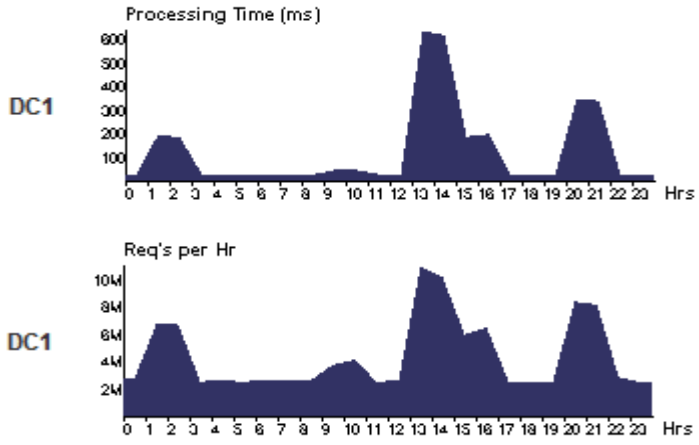
Πίνακας 7: Σενάριο 1ο - Χρόνος απόκρισης

Ο χρόνος είναι αρκετά μεγάλος διότι χρησιμοποιούνται μόνο 25 εικονικές μηχανές, οι οποίες διαχειρίζονται έναν πολύ μεγάλο αριθμό αιτήσεων. Οι χρόνοι απόκρισης που παρατηρήθηκαν στις user bases φαίνονται στην εικόνα 34.



Εικόνα 34: Σενάριο 1 - Οι χρόνοι απόκρισης στις User Bases

Οι αιχμές στους χρόνους απόκρισης φαίνονται καθαρά κατά την περίοδο των δύο ωρών αιχμής και παρατηρούμε ότι οι φόρτοι αιχμής επηρεάζουν και τις άλλες user bases. Για παράδειγμα η UB1 έχει αιχμές τις ώρες 13:00-15:00 και οι άλλες user bases έχουν μικρές αιχμές εκείνη την στιγμή. Οι επιπτώσεις όμως είναι μικρότερες, όσο ο αριθμός των αιτήσεων των user bases είναι μικρός, την δεδομένη χρονική στιγμή.



Εικόνα 35: Σενάριο 1 - Χρόνος επεξεργασίας και αιτήσεις

Όπως αναμενόταν, οι δύο γραφικές παραστάσεις αντικατοπτρίζουν η μία την άλλη. Παρατηρούμε ότι ο χρόνος επεξεργασίας αυξάνεται, όταν αυξάνονται οι αιτήσεις που λαμβάνει το data center. Ιδιαίτερα τις ώρες αιχμής 13:00-15:00, υπάρχει μια αιχμή λόγω των πολλών αιτήσεων από την user base 1.

Τέλος, το συνολικό κόστος για τη φιλοξενία της εφαρμογής, παρουσιάζεται στον πίνακα 8.

Κόστος εικονικών μηχανών ($)	60.03
Κόστος μεταφοράς δεδομένων ($)	1041.03
Συνολικό κόστος ($)	1101.05

Πίνακας 8: Σενάριο 1 - Κόστος φιλοξενίας εφαρμογής

3.7.3 Σενάριο 2ο - Η εφαρμογή φιλοξενείται σε δύο data center

Στο 2ο σενάριο, πραγματοποιούνται δύο πειράματα με δύο data centers και 50 εικονικές μηχανές το καθένα. Επίσης, στο δεύτερο πείραμα επιλέχθηκε η πολιτική Optimise Response Time για τον Service Broker.

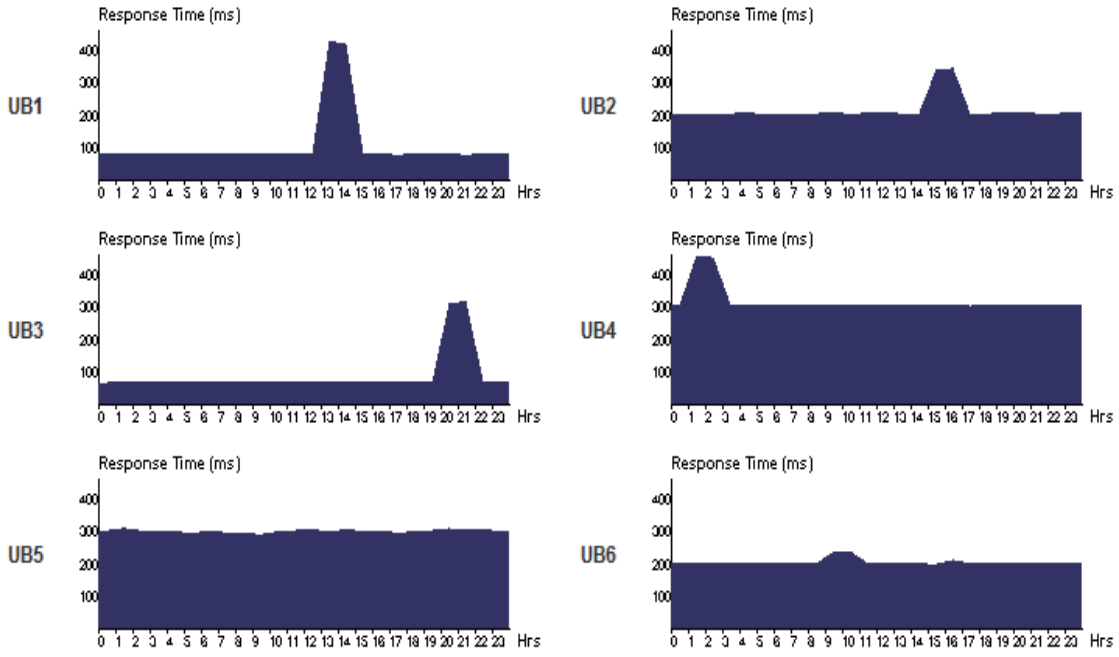
α) Δύο data center με 50 εικονικές μηχανές

Στο παρόν σενάριο σε κάθε data center ορίζονται 50 εικονικές μηχανές, με την πολιτική του Service Broker να είναι η Closest Data Center.

	Avg (ms)	**Min (ms)**	**Max (ms)**
Συνολικός χρόνος απόκρισης	256.90	41.24	588.79
Χρόνος επεξεργασίας στο data center 1	128.56	0.53	431.23
Χρόνος επεξεργασίας στο data center 2	94.52	0.37	333.59

Πίνακας 9: Σενάριο 2α - Χρόνος απόκρισης

Όπως βλέπουμε στον πίνακα 9 οι χρόνοι απόκρισης και επεξεργασίας είναι αρκετά μειωμένοι, από αυτούς του Σεναρίου 1. Αυτό γίνεται διότι διπλασιάστηκε ο αριθμός των εικονικών μηχανών και τα data center που εξυπηρετούν τις αιτήσεις. Οι χρόνοι απόκρισης που παρατηρήθηκαν στις user bases φαίνονται στην εικόνα 36.



Εικόνα 36: Σενάριο 2α - Οι χρόνοι απόκρισης στις User Bases

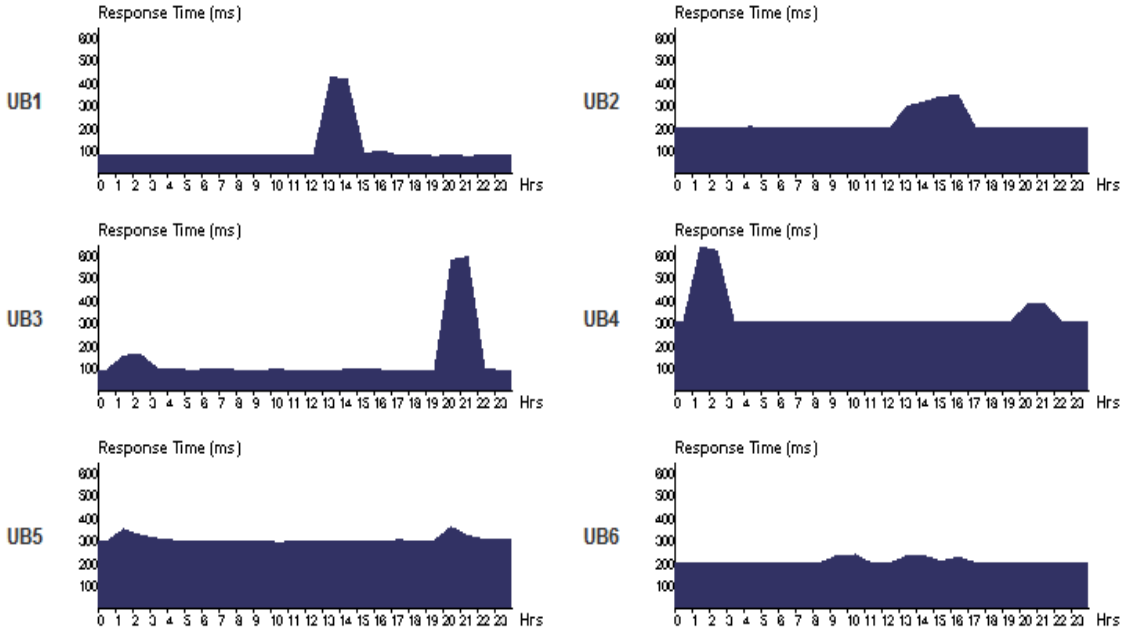
β) Δύο data center με 50 εικονικές μηχανές και μοιραζόμενο φόρτο

Σε αυτό το σενάριο αλλάζει η πολιτική του Service Broker σε Optimise Response Time.

	Avg (ms)	Min (ms)	Max (ms)
Συνολικός χρόνος απόκρισης	259.25	42.82	648.83
Χρόνος επεξεργασίας στο data center 1	128.26	0.51	431.97
Χρόνος επεξεργασίας στο data center 2	93.98	0.18	331.95

Πίνακας 10: Σενάριο 2β - Χρόνος απόκρισης

Παρατηρούμε ότι οι χρόνοι σε σχέση με το Σενάριο 2α, είναι ελαφρώς μειωμένοι, χωρίς ωστόσο να υπάρχει μεγάλη διαφορά μεταξύ τους. Οι χρόνοι απόκρισης που παρατηρήθηκαν στις user bases φαίνονται στην εικόνα 37.



Εικόνα 37: Σενάριο 2β - Οι χρόνοι απόκρισης στις User Bases

Το συνολικό κόστος για τη φιλοξενία της εφαρμογής, είναι ίδιο για τα σενάρια (2α,2β) και παρουσιάζεται στον πίνακα 11.

Κόστος εικονικών μηχανών ($)	240.10
Κόστος μεταφοράς δεδομένων ($)	1041.03
Συνολικό κόστος ($)	1281.13

Πίνακας 11: Σενάριο 2 - Κόστος φιλοξενίας εφαρμογής

3.7.4 Σενάριο 3ο - Η εφαρμογή φιλοξενείται σε τρία data center

Κλείνοντας την προσομοίωση σε αυτό το σενάριο, χρησιμοποιούνται τρία data centers με 50 εικονικές μηχανές το καθένα. Και εδώ το ένα πείραμα γίνεται με την πολιτική Optimise Response Time.

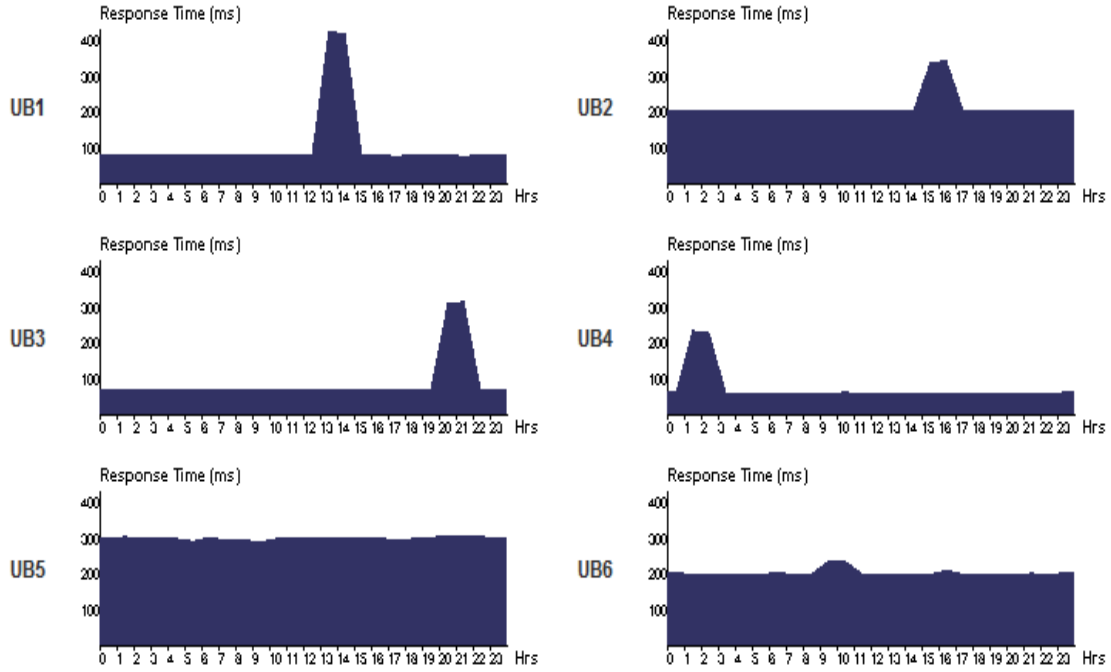
α) Τρία data center με 50 εικονικές μηχανές

Στο παρόν σενάριο ορίζονται τρία data center με 50 εικονικές μηχανές και η πολιτική του Service Broker είναι η Closest Data Center.

	Avg (ms)	Min (ms)	Max (ms)
Συνολικός χρόνος απόκρισης	214.89	41.24	540.00
Χρόνος επεξεργασίας στο data center 1	128.52	0.53	431.23
Χρόνος επεξεργασίας στο data center 2	119.84	0.80	333.59
Χρόνος επεξεργασίας στο data center 3	84.46	0.35	220.78

Πίνακας 12: Σενάριο 3α - Χρόνος απόκρισης

Ομοίως, οι χρόνοι απόκρισης και επεξεργασίας μειώθηκαν σε σχέση με αυτούς του Σεναρίου 2α, διότι υπάρχει ένα επιπλέον data center. Οι χρόνοι απόκρισης που παρατηρήθηκαν στις user bases φαίνονται στην εικόνα 38.



Εικόνα 38: Σενάριο 3α - Οι χρόνοι απόκρισης στις User Bases

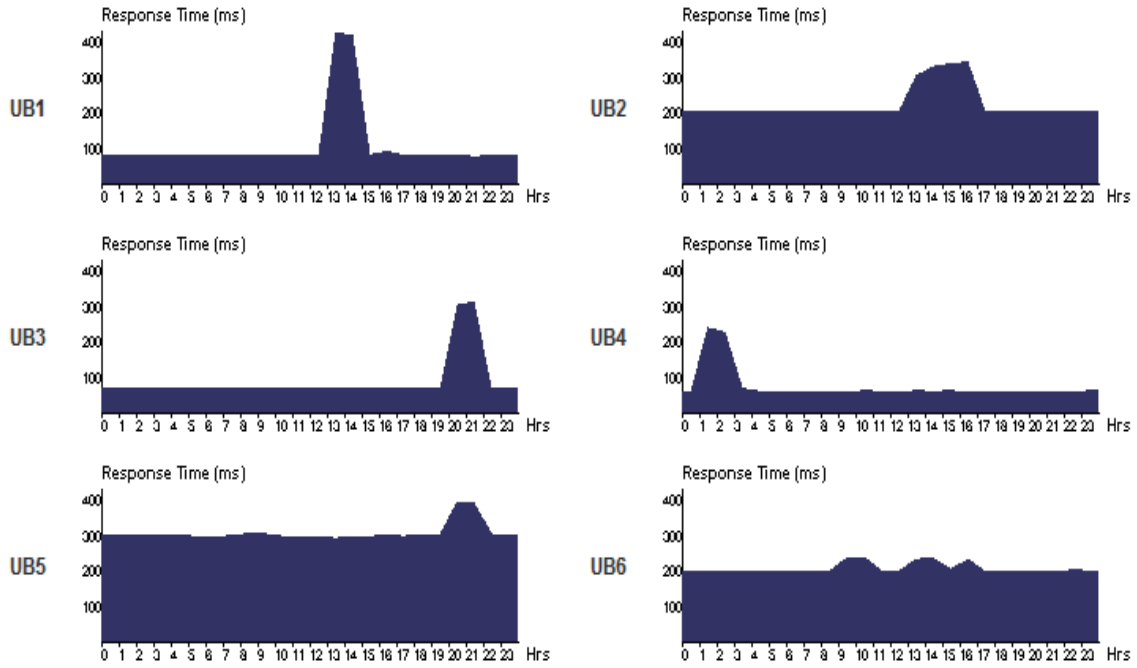
β) Τρία data center με 50 εικονικές μηχανές και μοιραζόμενο φόρτο

Σε αυτό το σενάριο αλλάζει η πολιτική του Service Broker σε Optimise Response Time.

	Avg (ms)	Min (ms)	Max (ms)
Συνολικός χρόνος απόκρισης	215.83	40.59	632.74
Χρόνος επεξεργασίας στο data center 1	128.60	0.50	432.13
Χρόνος επεξεργασίας στο data center 2	106.53	0.75	332.81
Χρόνος επεξεργασίας στο data center 3	62.36	0.34	203.03

Πίνακας 13: Σενάριο 3β - Χρόνος απόκρισης

Παρατηρούμε οτι οι χρόνοι απόκρισης μεταξύ των σεναρίων 3α και 3β έχουν μικρή διαφορά μεταξύ τους. Οι χρόνοι απόκρισης που παρατηρήθηκαν στις user bases φαίνονται στην εικόνα 39.



Εικόνα 39: Σενάριο 3β - Οι χρόνοι απόκρισης στις User Bases

Το συνολικό κόστος για τη φιλοξενία της εφαρμογής, είναι ίδιο για τα σενάρια (3α,3β) και παρουσιάζεται στον πίνακα 14.

Κόστος εικονικών μηχανών ($)	360.15
Κόστος μεταφοράς δεδομένων ($)	1041.03
Συνολικό κόστος ($)	1401.18

Πίνακας 14: Σενάριο 3 - Κόστος φιλοξενίας εφαρμογής

Κεφάλαιο 4

Προσομοιωτής GreenCloud

"Fairly cheap home computing was what changed my life."

Linus Torvalds, Δημιουργός του Linux Kernel, GIT

4.1 Εισαγωγή

Κατά τη διάρκεια των τελευταίων χρόνων, οι υπηρεσίες υπολογιστικής νέφους έχουν γίνει εξαιρετικά δημοφιλείς λόγω των εξελισσόμενων data centers και των παραδειγμάτων παράλληλου υπολογισμού. Οι πάροχοι των υπηρεσιών νέφους εκτός από το κόστος εγκατάστασης και συντήρησης του εξοπλισμού, θα πρέπει να υπολογίζουν και το ενεργειακό κόστος μιας τέτοιας εγκατάστασης. Η μείωση του ενεργειακού αποτυπώματος, απασχολεί ολοένα και περισσότερο τις εταιρείες πληροφορικής και στόχος είναι η μείωση του τα επόμενα χρόνια.

Η λειτουργία μεγάλων και γεωγραφικά διαμοιρασμένων data centers, απαιτεί σημαντική ποσότητα ενέργειας που ευθύνεται για ένα μεγάλο κομμάτι των συνολικών λειτουργικών εξόδων τους. Η εταιρεία Gartner υπολογίζει ότι η κατανάλωση ενέργειας, ευθύνεται για πάνω απο το 10% των τωρινών λειτουργικών εξόδων OPEX των data centers και αυτή η εκτίμηση μπορεί να ανέλθει στο 50% τα επόμενα χρόνια. Ωστόσο, η υπολογιζόμενη ενεργειακή κατανάλωση δεν είναι ο μόνος λόγος που αυξάνει τα λειτουργικά έξοδα. Η υψηλή κατανάλωση ρεύματος παράγει θερμότητα και απαιτείται ένα σύστημα ψύξης που θα πρέπει να εγκατασταθεί, το οποίο κοστίζει από δύο έως πέντε εκατομμύρια δολάρια τον χρόνο, για ένα κλασσικό data center.

Η αποτυχία να κρατηθούν οι θερμοκρασίες ενός data center σε λειτουργικό εύρος, μειώνει δραστικά την αξιοπιστία του υλικού και μπορεί ενδεχομένως να παραβιάσει το Συμφωνητικό παροχής υπηρεσιών - Service Level Agreement (SLA) που έχει συνάψει ο πάροχος με τους πελάτες. Μία σημαντική ποσότητα θερμότητας (πάνω από 70%) παράγεται από την υποδομή των data centers. Επομένως, η βελτιστοποιημένη εγκατάσταση της υποδομής μπορεί να παίξει σημαντικό ρόλο στη μείωση των λειτουργικών εξόδων.

Από την οπτική της ενεργειακής απόδοσης, ένα cloud computing data center μπορεί να οριστεί ως μια δεξαμενή υπολογιστικών και τηλεπικοινωνιακών πόρων, που είναι οργανωμένοι με τέτοιο τρόπο ώστε να μετατρέπουν την κοινή ενέργεια σε υπολογιστική δουλειά ή σε μεταφορά δεδομένων για να ικανοποιήσουν τις απαιτήσεις των χρηστών. Οι πρώτες λύσεις για εξοικονόμηση ενέργειας επικεντρώθηκαν στο να γίνει το υλικό των data centers ενεργειακά αποδοτικό. Οι τεχνολογίες όπως η Dynamic Voltage and Frequency Scaling (DVFS) και η Dynamic Power Management (DPM) μελετήθηκαν διεξοδικά και χρησιμοποιήθηκαν ευρέως. Επειδή, οι προαναφερθείσες τεχνολογίες στηρίζονται στις μεθοδολογίες power-down και power-off, η αποδοτικότητα τους είναι στην καλύτερη περίπτωση περιορισμένη. Στην πραγματικότητα ένας server σε κατάσταση αδράνειας μπορεί να καταναλώσει τα 2/3 του φόρτου αιχμής.

Επειδή ο φόρτος εργασίας ενός data center κυμαίνεται σε εβδομαδιαία και σε πολλές περιπτώσεις σε ημερήσια βάση, είναι κοινή πρακτική να παρέχονται υπερβολικά πολλοί υπολογιστικοί και τηλεπικοινωνιακοί πόροι για να ικανοποιήσουν τον φόρτο αιχμής. Στην πραγματικότητα, ο μέσος φόρτος ευθύνεται για το 30% των πόρων ενός data center. Αυτό επιτρέπει το υπόλοιπο 70% των πόρων να μπουν σε κατάσταση

ύπνου για την περισσότερη ώρα. Όμως για να επιτευχθεί κάτι τέτοιο απαιτεί κεντρικό συντονισμό και τεχνικές ενεργειακής επίγνωσης του φόρτου εργασίας. Οι συνηθισμένες αυτές λύσεις προσπαθούν να: συγκεντρώσουν τον φόρτο εργασίας στο ελάχιστο σύνολο από υπολογιστικούς πόρους και να μεγιστοποιήσουν τον αριθμό των πόρων που μπορούν να μπουν σε κατάσταση ύπνου.

Ένας αριθμός ερευνών έχει αποδείξει ότι πολύ συχνά μια απλή βελτιστοποίηση της αρχιτεκτονικής ενός data center και ο ενεργειακός προγραμματισμός των φόρτων εργασίας μπορεί να οδηγήσει σε σημαντική οικονομία ενέργειας.

Σε αυτό το κεφάλαιο παρουσιάζεται το περιβάλλον προσομοίωσης GreenCloud. Το GreenCloud είναι ένα προηγμένο πακέτο προσομοίωσης, για data centers υπολογιστικού νέφους υψηλής ενεργειακής απόδοσης, με επίκεντρο τις cloud επικοινωνίες. Προσφέρει ένα εξαιρετικά λεπτομερές μοντέλο της ενέργειας που καταναλώνεται από τον IT εξοπλισμό του data center, όπως οι εξυπηρετητές, τα δικτυακά switches και τα κανάλια επικοινωνίας. Μπορεί να χρησιμοποιηθεί για την ανάπτυξη καινοτόμων λύσεων στην διαχείριση, την διανομή πόρων, τον φόρτο εργασίας όπως επίσης και της βελτιστοποίησης των πρωτοκόλλων επικοινωνίας και των δικτυακών υποδομών. Είναι επέκταση του διάσημου NS2 δικτυακού προσομοιωτή. Ο NS2 χρησιμοποιεί δύο γλώσσες προγραμματισμού, την C++ και την OTCL. Οι εντολές από την TCL συνήθως περνάνε στην C++ χρησιμοποιώντας την διεπαφή TclCL. Το GreenCloud χρησιμοποιεί το 80% του κώδικα που γίνεται σε C++ (TclCL κλάσεις) και το υπόλοιπο 20% του κώδικα αναπτύσσεται χρησιμοποιώντας Tcl scripts. [23] Κυκλοφορεί κάτω από την άδεια GNU General Public License και έχει αναπτυχθεί από το Πανεπιστήμιο του Λουξεμβούργου. [24]

4.1.1 Κύρια χαρακτηριστικά

Τα κυριότερα χαρακτηριστικά του προσομοιωτή GreenCloud είναι τα ακόλουθα. [25]

- Επικεντρώνεται στο cloud networking και στην ενεργειακή γνώση.
- Προσομοίωση της CPU, της μνήμης, της αποθήκευσης και των δικτυακών πόρων.
- Παρέχει ανεξάρτητα μοντέλα ενέργειας για κάθε τύπο πόρου.
- Υποστήριξη του virtualisation και των εικονικών μηχανών.
- Γνώση της διανομής των δικτυακών πόρων.
- Πλήρης ανάπτυξη του πρωτοκόλλου TCP/IP.
- Φιλικό περιβάλλον εργασίας προς τον χρήστη ανοιχτού κώδικα.

4.2 Ενεργειακή απόδοση

Μόνο ένα μέρος της ενέργειας που καταναλώνεται από το data center, αφορά τους υπολογιστικούς servers. Ένα μεγάλο μέρος της ενέργειας χρησιμοποιείται για την διαχείριση των ενδοσυνδεδεμένων συνδέσεων και των λειτουργιών του δικτυακού εξοπλισμού. Η υπόλοιπη ενέργεια χάνεται στο σύστημα διανομής ενέργειας, διαχέεται ως θερμική ενέργεια και χρησιμοποιείται από τα συστήματα κλιματισμού. Στο GreenCloud, υπάρχουν τρία στοιχεία ενεργειακής κατανάλωσης

1. Υπολογιστική ενέργεια,
2. Ενέργεια επικοινωνιών και
3. το ενεργειακό στοιχείο που σχετίζεται με την φυσική υποδομή του data center.

Η αποδοτικότητα ενός data center μπορεί να καθοριστεί με βάση την επίδοση ανά watt, η οποία μπορεί να ποσοτικοποιηθεί με τις παρακάτω δύο μετρικές.

1. Power Usage Effectiveness (PUE)
2. Data Center Infrastructure Efficiency (DCiE)

Και οι δύο μετρικές περιγράφουν την ποσότητα της συνολικής καταναλισκόμενης ενέργειας που φτάνει στους υπολογιστικούς servers. [24]

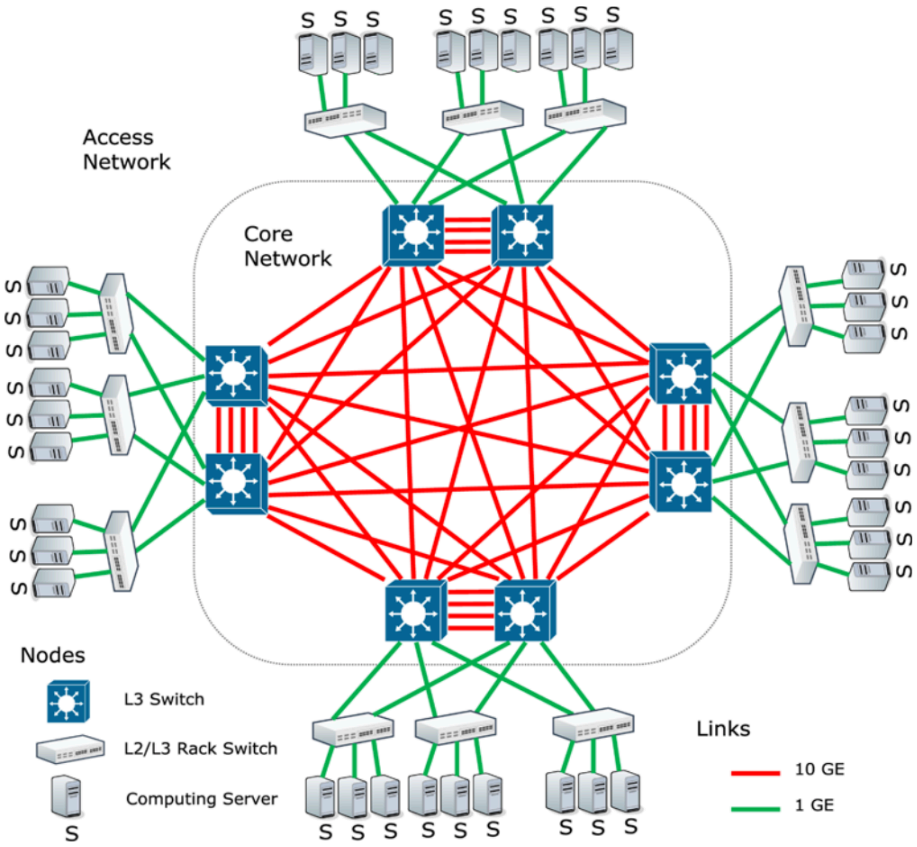
4.3 Γνωστές αρχιτεκτονικές Data Center

Τα σύγχρονα data center είναι μια δεξαμενή από πόρους (υπολογιστικούς, αποθήκευσης, δικτυακούς), οι οποίοι είναι διασυνδεδεμένοι σε ένα δίκτυο επικοινωνίας. Το Data Center Network (DCN) κατέχει ρόλο

καίριας σημασίας, καθώς διασυνδέει όλα τα δεδομένα του data center. Τα Data Center Network πρέπει να είναι επεκτάσιμα και αποτελεσματικά στην σύνδεση δεκάδων ή ακόμα και εκατοντάδων χιλιάδων servers, ώστε να καλύψουν τις απαιτήσεις του cloud computing. Στις μέρες μας, τα data centers περιορίζονται από το διασυνδεδεμένο δίκτυο.

Η αρχιτεκτονική second-tier

Η αρχιτεκτονική two-tier ενός data center φαίνεται στην εικόνα 40. Σε αυτή οι υπολογιστικοί servers (S) οργανώνονται σε racks και σχηματίζουν το δίκτυο one-tier. Στο δίκτυο two-tier, τα switches επιπέδου 3 (L3) παρέχουν πλήρως κατανεμημένη τοπολογία, χρησιμοποιώντας συνδέσεις 10 GE. Η δρομολόγηση Equal Cost Multi Path (ECMP) χρησιμοποιείται ως τεχνολογία κατανομής φόρτου, με σκοπό την βελτιστοποίηση της ροής των δεδομένων σε πολλαπλά μονοπάτια. Πετυχαίνει κατανομή φόρτου σε TCP και UDP πακέτα χρησιμοποιώντας γρήγορες τεχνικές hash αλγορίθμων, οι οποίες δεν απαιτούν επεξεργαστική ισχύ από τον επεξεργαστή των switches. Κίνηση, όπως τα πακέτα ICMP συνήθως δεν επεξεργάζεται από το ECMP και προωθείται σε ένα μονό προκαθορισμένο μονοπάτι. Αυτού του είδος η αρχιτεκτονική, ήταν καλή στα πρώιμα στάδια των data centers, τα οποία είχαν περιορισμένο αριθμό από υπολογιστικούς servers. Ανάλογα με τον τύπο των switches που χρησιμοποιούνται στο access δίκτυο, τα two-tier data centers μπορούν να υποστηρίξουν μέχρι 5.500 κόμβους. Ο αριθμός των core switches και η χωρητικότητα των συνδέσεων καθορίζει το μέγιστο εύρος ζώνης του δικτύου, που διατίθεται ανά υπολογιστικό server.

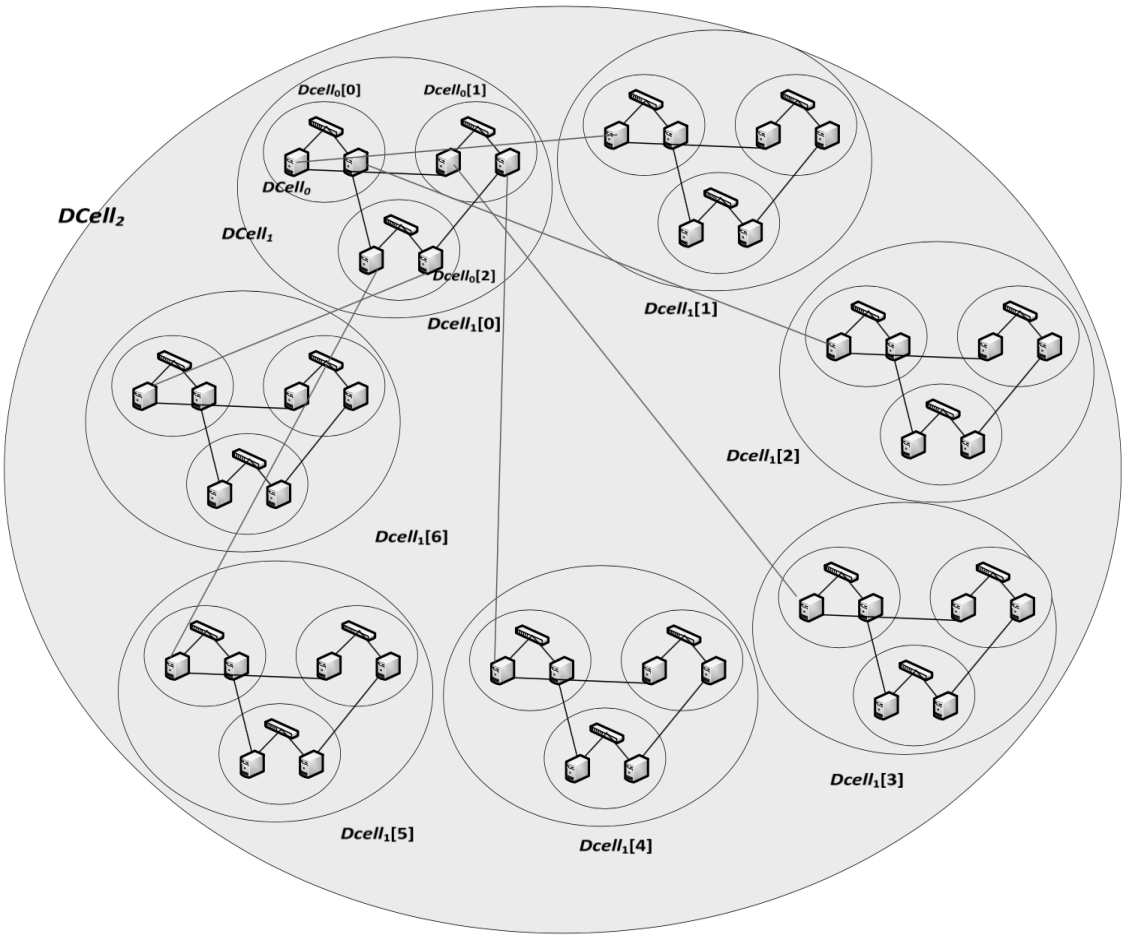


Εικόνα 40: Η αρχιτεκτονική two-tier [24]

Η αρχιτεκτονική DCell

Στην αρχιτεκτονική DCell, το πλήρες σύστημα αποτελείται από κελιά ή δεξαμενές με έναν αριθμό από n servers και ένα παρεχόμενο switch. Το κελί DCell0 στο επίπεδο 0 χρησιμοποιείται ως το δομικό στοιχείο ολόκληρου του συστήματος. Σε αυτό το επίπεδο περιλαμβάνονται οι παρεχόμενοι servers και ένα mini switch, ενώ τα υψηλότερα επίπεδα κελιών χτίζονται συνδέοντας πολλαπλά μικρότερου επιπέδου (level-1) DCells. Κάθε Dcell-1 συνδέεται με όλα τα άλλα DCell-1 στο ίδιο DCell. Το DCell παρέχει μια εξαιρετικά επεκτάσιμη αρχιτεκτονική και ένα DCell 3 επιπέδων έχοντας 6 servers στο DCell0 μπορεί να φιλοξενήσει

περίπου 3.26 εκατομμύρια servers. Η εικόνα 41 δείχνει ένα DCell επιπέδου 2, το οποίο έχει 2 servers σε κάθε DCell0. [26]



Εικόνα 41: Η αρχιτεκτονική DCell [26]

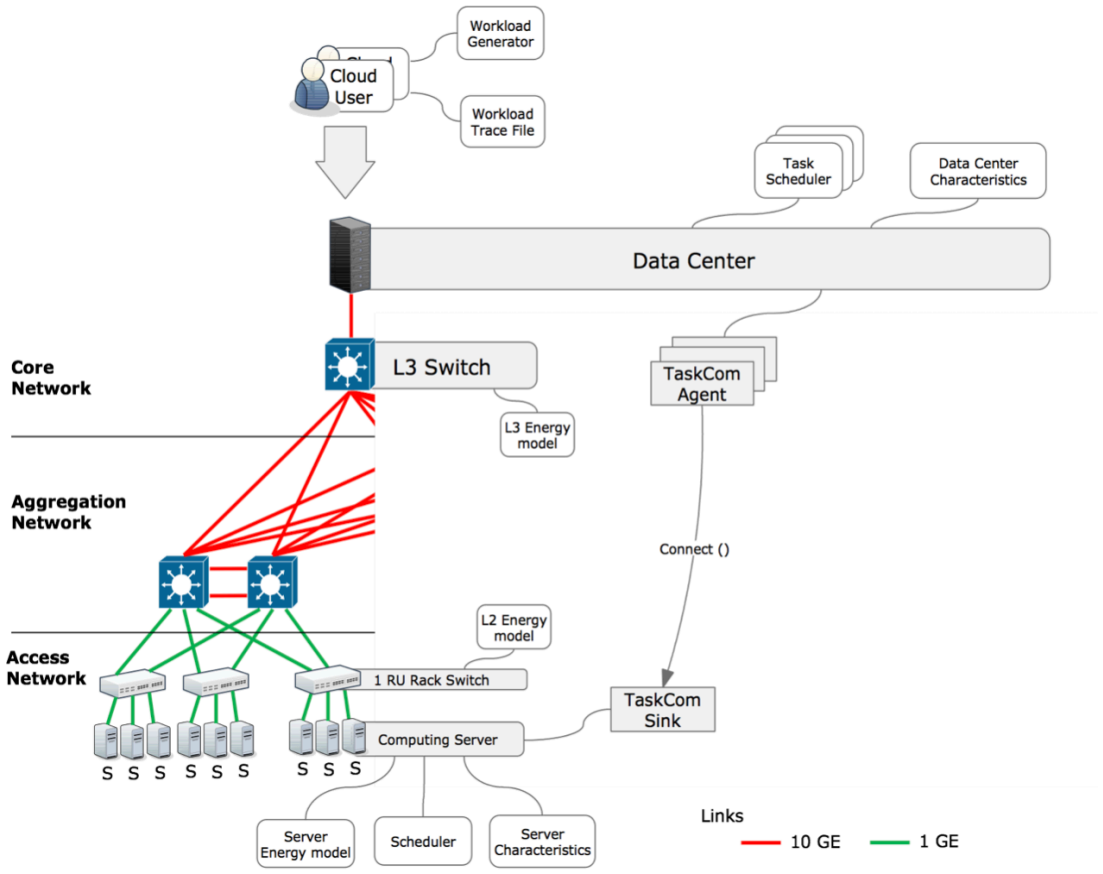
Η αρχιτεκτονική DCell χρησιμοποιεί υβριδική δρομολόγηση και πρωτόκολλο επεξεργασίας δεδομένων. Τα switches χρησιμοποιούνται για την επικοινωνία ανάμεσα στους servers στο ίδιο DCell0. Η επικοινωνία με τους servers που βρίσκονται σε διαφορετικά DCells γίνεται με τους servers να λειτουργούν ως δρομολογητές. Στην πραγματικότητα οι υπολογιστικοί servers θεωρούνται και ως δρομολογητές στο σύστημα. [26]

4.3.1 Η αρχιτεκτονική του GreenCloud

Ο προσομοιωτής GreenCloud προσφέρει μια εκτενή έρευνα της διανομής του φόρτου εργασίας. Ακόμα, δίνεται ιδιαίτερη προσοχή στις προσομοιώσεις, σε επίπεδο πακέτων, των επικοινωνιών σε μια υποδομή ενός data center. Έτσι παρέχεται ιδιαίτερα λεπτομερής έλεγχος, κάτι που δεν είναι αυτή τη στιγμή χαρακτηριστικό κανενός άλλου περιβάλλοντος προσομοίωσης cloud computing. [24]

Το GreenCloud υποστηρίζει την three-tier αρχιτεκτονική που χρησιμοποιείται ευρέως, καθώς και άλλες μοντέρνες αρχιτεκτονικές όπως είναι η DCell που περιγράφηκε παραπάνω, η BCube, η FiConn και η DPillar. Αποτελείται από το core επίπεδο που βρίσκεται στην κορυφή του δέντρου, το aggregation επίπεδο που είναι υπεύθυνο για το routing και το access επίπεδο στο οποίο βρίσκεται το σύνολο των υπολογιστικών servers, οργανωμένοι σε racks. Οι υπολογιστικοί servers είναι ενδοσυνδεδεμένοι με συνδέσεις 1 Gigabit Ethernet (GE), ενώ τα switches των επιπέδων core και aggregation είναι εξοπλισμένα με πόρτες 10 GE. Αυτό έχει ως αποτέλεσμα με τα συνηθισμένα racks, που υποστηρίζουν μέχρι 48 servers, η αναλογία στο εύρος ζώνης να είναι ίσο με 48/20 = 2,4:1 και 1.5:1 στα δίκτυα access και aggregation αντίστοιχα. Η εικόνα 42 παρουσιάζει την αρχιτεκτονική του GreenCloud που στηρίζεται στην three-tier αρχιτεκτονική του data center.

Servers(S): Είναι το βασικότερο στοιχείο ενός data center και είναι υπεύθυνοι για την εκτέλεση των εργασιών. Στο GreenCloud, τα στοιχεία των servers έχουν κόμβους μονού πυρήνα οι οποίοι έχουν ένα



Εικόνα 42: Η αρχιτεκτονική του GreenCloud [24]

προκαθορισμένο όριο επεξεργαστικής ισχύος σε Million Instructions per Second (MIPS) (εκατομμύρια εντολές το δευτερόλεπτο) ή σε Floating Point Operations per Second (FLOPS), ένα συνδυασμένο μέγεθος πόρων μνήμης και χώρου αποθήκευσης και περιλαμβάνουν διαφορετικούς μηχανισμούς προγραμματισμού των έργων, που κυμαίνονται από τον απλό αλγόριθμο round-robin έως τον εξεζητημένο αλγόριθμο Dynamic Voltage and Frequency Scaling (DVFS).

Οι servers έχουν ταξινομηθεί σε racks με ένα Top-of-Rack (ToR) switch να τους συνδέει με το access κομμάτι του δικτύου. Το μοντέλο ενέργειας που ακολουθούν οι server, εξαρτάται από την κατάσταση του εκάστοτε server και την χρήση της κεντρικής μονάδας επεξεργασίας του. Ένας server σε κατάσταση αδράνειας καταναλώνει περίπου 66% της ενέργειας συγκριτικά με την πλήρως φορτωμένη διαμόρφωση του. Αυτό οφείλεται στο γεγονός ότι οι servers πρέπει να διατηρούν την μνήμη, τους δίσκους, τους πόρους εισόδου/εξόδου και άλλα περιφερειακά σε μια κατάσταση λειτουργίας. Τότε η κατανάλωση ενέργειας αυξάνεται γραμμικά μαζί με τα επίπεδα φόρτου της CPU. Ως αποτέλεσμα το προαναφερθέν μοντέλο επιτρέπει την εφαρμογή της εξοικονόμησης ενέργειας σε έναν κεντρικό προγραμματιστή, που μπορεί να αναθέσει τον φόρτο εργασίας στον μικρότερο αριθμό πιθανών υπολογιστικών servers. [24]

Switches και συνδέσεις: Συγκροτούν το σύστημα της ενδοσύνδεσης που παραδίδει εγκαίρως τον φόρτο εργασίας στους servers για εκτέλεση. Η ενδοσύνδεση μεταξύ servers και switches απαιτεί διαφορετικές καλωδιακές λύσεις και εξαρτάται από το υποστηριζόμενο εύρος ζώνης, τα φυσικά και ποιοτικά χαρακτηριστικά της σύνδεσης. Η ποιότητα της μετάδοσης σήματος σε ένα συγκεκριμένο καλώδιο καθορίζεται με έναν συμβιβασμό ανάμεσα στο ρυθμό μετάδοσης και την απόσταση της σύνδεσης, οι οποίοι είναι παράγοντες που καθορίζουν το κόστος και την κατανάλωση ενέργειας των πομποδεκτών.

Το καλώδιο συνεστραμμένων ζευγών είναι αυτό που χρησιμοποιείται συχνά για δίκτυα Ethernet που επιτρέπουν την μετάδοση Gigabit Ethernet (GE) μέχρι 100 μέτρα με καταναλισκόμενη ενέργεια περίπου 0.4 Watt ή 10 συνδέσεις GE μέχρι 30 μέτρα με ενέργεια πομποδέκτη 6 Watt. Η καλωδίωση με συνεστραμμένα ζεύγη είναι μια φθηνή λύση. Ωστόσο, για τον οργανισμό με 10 συνδέσεις GE, είναι σύνηθες να χρησιμοποιεί πολύτροπες οπτικές ίνες, οι οποίες επιτρέπουν μετάδοση μέχρι 300 μέτρα με ενέργεια του πομποδέκτη

μόλις 1 Watt. Από την άλλη μεριά, το γεγονός ότι οι πολύτροπες οπτικές ίνες κοστίζουν περίπου 50 φορές παραπάνω το κόστος των συνεστραμμένων ζευγών, περιορίζει τη χρήση τους μόνο στο δίκτυο πυρήνα και στο συγκεντρωμένο δίκτυο, καθώς τα έξοδα για τη δικτυακή υποδομή μπορούν να φτάσουν το 10-20% του συνολικού προϋπολογισμού του data center.

Ο αριθμός των switches που θα εγκατασταθεί, εξαρτάται από την υπολοποιημένη αρχιτεκτονική του data center. Ωστόσο, καθώς οι servers οργανώνονται συνήθως σε racks, ο πιο γνωστός τύπος switch σε ένα data center είναι το Top-of-Rack (ToR) switch. Το ToR switch είναι τοποθετημένο στο ψηλότερο σημείο ενός rack (1RU) ώστε να μειωθεί η ποσότητα των καλωδίων και η θερμότητα που παράγεται. Το ToR switch μπορεί να υποστηρίξει ταχύτητες gigabit (GE) ή 10 gigabit (10 GE). Παρόλα αυτά, λαμβάνοντας υπόψιν ότι τα switches 10 GE είναι πολύ ακριβά και η χωρητικότητα του συγκεντρωμένου και του δικτύου πυρήνα περιορισμένη, τα gigabit switches βρίσκονται πιο συχνά σε racks. [24]

Φόρτοι εργασίας: Είναι τα αντικείμενα που έχουν σχεδιαστεί για καθολική μοντελοποίηση των διαφόρων υπηρεσιών των χρηστών cloud, όπως είναι η κοινωνική δικτύωση, τα άμεσα μηνύματα και η διανομή περιεχομένου.

Στο grid computing, οι φόρτοι εργασίας συχνά μοντελοποιούνται ως μια αλληλουχία από εργασίες, οι οποίες μπορεί να χωριστούν σε μια ομάδα από έργα. Αυτά τα έργα μπορεί να είναι ανεξάρτητα ή εξαρτώμενα, δηλαδή να απαιτούν ένα αποτέλεσμα από ένα άλλο έργο για να ξεκινήσουν την εκτέλεση. Επιπλέον, λόγω της φύσης των εφαρμογών του grid computing (βιολογικές, οικονομική μοντελοποίηση ή κλιματική μοντελοποίηση), ο αριθμός των εργασιών που είναι διαθέσιμες υπερισχύει του αριθμού των υπολογιστικών πόρων που είναι διαθέσιμοι. Ενώ ο κύριος στόχος είναι η ελαχιστοποίηση του χρόνου που απαιτείται για τον υπολογισμό όλων των εργασιών και μπορεί να πάρει εβδομάδες ή μήνες, οι ατομικές εργασίες δεν έχουν κάποια αυστηρή προθεσμία ολοκλήρωσης.

Στο cloud computing, οι εισερχόμενες αιτήσεις παράγονται συνήθως από εφαρμογές όπως είναι η περιήγηση στο διαδίκτυο, τα άμεσα μηνύματα ή οι εφαρμογές διανομής περιεχομένου. Οι εργασίες τείνουν να είναι πιο ανεξάρτητες, λιγότερο εντατικές υπολογιστικά, αλλά έχουν αυστηρή προθεσμία ολοκλήρωσης που περιγράφεται στο SLA. Για την κάλυψη της συντριπτικής πλειοψηφίας των εφαρμογών cloud computing, ορίζονται τρεις τύποι εργασιών:

- *Εντατικοί Υπολογιστικοί Φόρτοι Εργασίας - Intensive Workloads (CIWs)*: Σε αυτόν τον τύπο εργασίας μοντελοποιούνται Υψηλής απόδοσης υπολογιστικές εφαρμογές - High Performance Computing (HPC), που έχουν στόχο να λύσουν προχωρημένα υπολογιστικά προβλήματα. Οι εργασίες CIW φορτώνουν σημαντικά τους servers, αλλά δεν απαιτούν σχεδόν καμία μεταφορά δεδομένων στις δικτυακές ενδοσυνδέσεις του data center. Η διαδικασία του προγραμματισμού της ενεργειακής απόδοσης των CIW, πρέπει να εστιάζεται στην ενεργειακή κατανάλωση των servers, προσπαθώντας να ομαδοποιήσει τους φόρτους εργασίας στον ελάχιστο αριθμό από servers και να δρομολογήσει την κίνηση που παράγεται χρησιμοποιώντας τον ελάχιστο αριθμό διαδρομών. Δεν υπάρχει ο κίνδυνος της συμφόρησης του δικτύου λόγω των χαμηλών απαιτήσεων για μεταφορά δεδομένων και βάζοντας τα περισσότερα switches σε κατάσταση ύπνου εξασφαλίζεται χαμηλότερη ενέργεια στο δίκτυο του data center.
- *Φόρτοι Εργασίας για εντατικά δεδομένα - Data Intensive Workloads (DIWs)*: Δεν παράγουν φόρτο στους servers, αλλά απαιτούν έντονες μεταφορές δεδομένων. Τα DIW στοχεύουν στη μοντελοποίηση εφαρμογών όπως ο διαμοιρασμός αρχείων βίντεο, οπού οι απλές αιτήσεις των χρηστών μετατρέπονται σε μια διαδικασία συνεχούς ροής βίντεο. Ως αποτέλεσμα το ενδοσυνδεδεμένο δίκτυο και όχι η υπολογιστική χωρητικότητα, γίνεται το σημείο συμφόρησης του data center για τα DIW. Ιδανικά θα έπρεπε να εφαρμόζεται συνεχής ανάδραση ανάμεσα στα στοιχεία του δικτύου (switches) και στο κεντρικό προγραμματιστή του φόρτου εργασίας. Βασισμένος σε αυτή την ανάδραση, ο προγραμματιστής θα διανείμει τους φόρτους εργασίας παίρνοντας τα τρέχοντα επίπεδα της συμφόρησης στις συνδέσεις επικοινωνίας. Κατά αυτόν τον τρόπο γίνεται δυνατή η αποφυγή μεταφοράς φόρτων εργασίας πάνω από συνδέσεις με συμφόρηση ακόμα και εάν η υπολογιστική ικανότητα ενός server μπορεί να εξυπηρετήσει αυτόν τον φόρτο. Μια τέτοια πολιτική προγραμματισμού θα ισορροπήσει την κίνηση στο δίκτυο του data center και θα μειώσει τον μέσο χρόνο που απαιτείται για την παράδοση μιας εργασίας από τα switches του πυρήνα στους υπολογιστικούς servers.
- *Ισορροπημένοι Φόρτοι - Balanced Workloads (BLs)*: Στοχεύουν στη μοντελοποίηση των εφαρμογών που έχουν υπολογιστικές απαιτήσεις και απαιτήσεις μεταφοράς δεδομένων. Οι ισορροπημένοι φόρτοι φορτώνουν αναλογικά τους υπολογιστικούς servers και τις συνδέσεις επικοινωνίας. Έτσι με αυτόν τον τύπο φόρτων εργασίας, ο μέσος φόρτος στους servers είναι ίσος με τον μέσο φόρτο στο δίκτυο του data center. Μπορεί να μοντελοποιήσουν εφαρμογές όπως τα γεωγραφικά συστήματα πληροφοριών - Geographic Information System (GIS), τα οποία απαιτούν αυξημένη επεξεργασία και πολύ μεγάλες μεταφορές δεδομένων με γραφικά. Ο προγραμματισμός των ισορροπημένων φόρτων

πρέπει να γίνεται για τον φόρτο των servers και για τον φόρτο του ενδοσυνδεδεμένου δικτύου.

Η εκτέλεση του κάθε αντικείμενου φόρτου στο GreenCloud απαιτεί την πετυχημένη ολοκλήρωση των δύο βασικών στοιχείων:

(a) του υπολογισμού και
(b) του επικοινωνιακού.

Το στοιχείο υπολογισμού ορίζει το ποσό του υπολογισμού που πρέπει να εκτελεστεί πριν από μια δοσμένη προθεσμία στην κλίμακα του χρόνου. Αυτή η προθεσμία έχει στόχο να εισάγει τους περιορισμούς της ποιότητας της υπηρεσίας QoS που έχουν οριστεί στο SLA.

Το επικοινωνιακό στοιχείο του φόρτου εργασίας ορίζει την ποσότητα και το μέγεθος των μεταφορών δεδομένων, που πρέπει να εκτελεστούν πριν, κατά την διάρκεια και μετά από την εκτέλεση του φόρτου. Αποτελείται από τρία μέρη:

(a) το μέγεθος του φόρτου εργασίας
(b) το μέγεθος των εσωτερικών και
(c) το μέγεθος των εξωτερικών επικοινωνιών του data center.

Το μέγεθος του φόρτου εργασίας ορίζει των αριθμό των bytes, τα οποία αφότου έχουν χωριστεί σε IP πακέτα θα πρέπει να αποσταλούν από τα core switches στους υπολογιστικούς servers πριν ξεκινήσει η εκτέλεση ενός φόρτου εργασίας. Το μέγεθος των εξωτερικών επικοινωνιών ορίζει την ποσότητα των δεδομένων που πρέπει να μεταδοθούν έξω από το δίκτυο του data center, την στιγμή της ολοκλήρωσης μιας εργασίας και συμβαδίζει με το αποτέλεσμα της εκτέλεσης της εργασίας. Το μέγεθος των εσωτερικών επικοινωνιών του data center ορίζει την ποσότητα των δεδομένων που θα ανταλλαχθούν με έναν άλλο φόρτο εργασίας, ο οποίος μπορεί να εκτελεστεί στον ίδιο ή σε διαφορετικό server. Με αυτόν τον τρόπο μοντελοποιούνται οι αλληλεξαρτήσεις του φόρτου εργασίας. Στην πραγματικότητα, η εσωτερική επικοινωνία στο data center υπολογίζεται περίπου στο 70% των συνολικών δεδομένων που μεταδίδονται. [24]

4.4 Εγκατάσταση του GreenCloud

Ο προσομοιωτής GreenCloud διανέμεται ως ένα αρχειοθετημένο δέντρο κώδικα ή ως μια προ-ρυθμισμένη εικονική μηχανή, που δουλεύει με το VirtualBox και το VMWare Player. Η εικονική μηχανή περιλαμβάνει ένα εγκατεστημένο Eclipse IDE και έτσι είναι ο ευκολότερος τρόπος για την πρώτη γνωριμία με το GreenCloud και το ξεκίνημα των προσομοιώσεων ή της μετατροπής του πηγαίου κώδικα. [27]

Για την εγκατάσταση του προσομοιωτή με την προ-ρυθμισμένη εικονική μηχανή θα πρέπει ο χρήστης να έχει εγκαταστήσει τα παρακάτω προαπαιτούμενα:

1. Να υπάρχει εγκατεστημένο το VirtualBox ή το VMWare Player, ώστε να γίνει εισαγωγή της εικονικής μηχανής σε αυτά.
2. Να έχει κατέβάσει το αρχείο της εικονικής μηχανής από τη σελίδα `https://greencloud.gforge.uni.lu/install.html`

Εάν κάποιος χρήστης θέλει να εγκαταστήσει το πακέτο χωρίς τη βοήθεια της εικονικής μηχανής, θα πρέπει να ακολουθήσει τα αντίστοιχα βήματα:

1. Να κατεβάσει το GreenCloud από τη σελίδα `https://greencloud.gforge.uni.lu/install.html`
2. Να κάνει εξαγωγή του κατεβασμένου λογισμικού, το οποίο έρχεται ενοποιημένο με τον πηγαίο κώδικα του NS2.
3. Να περιηγηθεί στον φάκελο που έχει εξαχθεί
4. Να εκτελέσει το script ./install.sh έτσι ώστε να γίνει πλήρη εγκατάσταση (πρέπει να δουλεύει σε κάθε σύστημα βασισμένο στο Debian με πυρήνα 3.2+. π.χ. στο Ubuntu 12.x και ανώτερο)
5. Να εκτελέσει το script προσομοίωσης από το GUI (`http://localhost`) ή χρησιμοποιώντας την εντολή ./run.
6. Τέλος, η εμφάνιση του dashboard γίνεται ανοίγοντας το αρχείο show-dashboard.html.

4.5 Προσομοίωση

Το GreenCloud ξεκινάει την προεπιλεγμένη προσομοίωση με 144 servers και έναν χρήστη Cloud. Όλες οι παράμετροι μπορεί να διαφοροποιηθούν και να δοκιμαστούν, ανάλογα με τις καταχωρήσεις στα αρχεία Tcl. Η προσομοίωση ξεκινάει χρησιμοποιώντας τα αρχεία Tcl που βρίσκονται στον κατάλογο ./src/scripts. Το βασικό αρχείο main.tcl, στο οποίο περιλαμβάνεται ο κώδικας 7, αποφασίζει για την τοπολογία του data center και για τον χρόνο της προσομοίωσης. [27]

```
# ----------------------------------------------------------
# ------------- Main GreenCloud simulation script ----------
# ----------------------------------------------------------

# Type of DC architecture
set sim(dc_type) [lindex $argv 6];  # can be "three-tier", "three-tier high-speed",
                                    #"three-tier debug", "three-tier heterogenous debug"
                                    #and "three-tier heterogenous"
puts $sim(dc_type);                 # in case of heterogenous topologies make sure that VMs are not
                                    #larger than hosts. (You can also try to turn off virtualization
                                    #in setup_params.tcl: set vm(static_virtualization) 0;)

# Set the time of simulation end
set sim(post_time) 0.6
set sim(end_time) [ expr 60 + [lindex $argv 1] + $sim(post_time)];# simualtion length set to 60s
                                                                 # + deadline of tasks

# Start collecting statistics
set sim(start_time) 0.1

set sim(tot_time) [expr $sim(end_time) - $sim(start_time)]

set sim(linkload_stats) "enabled"

# Set the interval time (in seconds) to make graphs and to create flowmonitor file
set sim(interval) 0.1

# Setting up main simulation parameters
source "setup_params.tcl"

# Get new instance of simulator
set ns [new Simulator]

# Tracing general files (*.nam & *.tr)
set nf [open "$dir(traces)/main.nam" w]
# $ns namtrace-all $nf
set trace [open "$dir(traces)/main.tr" w]
# $ns trace-all $trace

# Building data center topology
source "topology.tcl"
```

Κώδικας 7: Απόφαση της τοπολογίας και του χρόνου προσομοίωσης στο αρχείο main.tcl

Επίσης, καλεί τα παρακάτω script προσομοίωσης:

- **setup_params.tcl**: περιέχει τις γενικές ρυθμίσεις των servers, των switches, των εργασιών και του ελέγχου.
- **topology.tcl**: δημιουργεί την τοπολογία δικτύου του data center. Στον κώδικα 8 δηλώνονται όλοι οι παράμετροι για κάθε τύπο data center που θέλει ο χρήστης να χρησιμοποιήσει.
- **dc.tcl**: δημιουργεί τους servers και τις εικονικές μηχανές του data center.
- **user.tcl**: καθορίζει την συμπεριφορά των χρηστών του cloud.
- **record.tcl**: δημιουργεί τις διαδικασίες για την παρουσίαση των αποτελεσμάτων της εκτέλεσης.
- **finish.tcl**: υπολογίζει και παρουσιάζει τα στατιστικά της προσομοίωσης.

Η εντολή ./run/script, χρησιμοποιείται για την εκκίνηση της προσομοίωσης και περιλαμβάνει τις παρακάτω τρεις παραμέτρους.

Φόρτος του data center: καθορίζει τον αριθμό και τις υπολογιστικές απαιτήσεις των εισερχόμενων

```
# ----------------------------------------------------
# ------------ Creating data center topology ---------
# ----------------------------------------------------

# SWITCHES

switch $sim(dc_type) {
  "three-tier high-speed" {
    set top(NCore)          2;                  # Number of L3 Switches in the CORE network
    set top(NAggr)      [expr 2*$top(NCore)];   # Number of Switches in AGGREGATION network
    set top(NAccess)        256;                # Number switches in ACCESS network per pod
    set top(NRackHosts)     3;                  # Number of Hosts on a rack
  }
  "three-tier debug" {
    set top(NCore)          1;                  # Number of L3 Switches in the CORE network
    set top(NAggr)      [expr 2*$top(NCore)];   # Number of Switches in AGGREGATION network
    set top(NAccess)        3;                  # Number switches in ACCESS network per pod
    set top(NRackHosts)     48;                 # Number of Hosts on a rack
  }
  "three-tier heterogenous debug" {
    set top(NCore)          1;                  # Number of L3 Switches in the CORE network
    set top(NAggr)      [expr 2*$top(NCore)];   # Number of Switches in AGGREGATION network
    set top(NAccess)        3;                  # Number switches in ACCESS network per pod
    set top(NRackHosts)     48;                 # Number of Hosts on a rack
  }
  # three-tier
  default {
    set top(NCore)          8;                  # Number of L3 Switches in the CORE network
    set top(NAggr)      [expr 2*$top(NCore)];   # Number of Switches in AGGREGATION network
    set top(NAccess)        64;                 # Number switches in ACCESS network per pod
    set top(NRackHosts)     3;                  # Number of Hosts on a rack
  }
}

# Number of racks
set top(NRacks) [expr $top(NAccess)*$top(NCore)]

# Number of servers
set top(NServers) [expr $top(NRacks)*$top(NRackHosts)]
```

Κώδικας 8: Δήλωση της τοπολογίας του δικτύου στο topology.tcl

εργασιών, όσον αφορά την χωρητικότητα του data center. Συνήθως ο φόρτος πρέπει να είναι μεταξύ 0 και 1. Φόρτος κοντά στο 0 αναπαριστά ένα data center σε κατάσταση αδράνειας, ενώ φόρτος ίσος ή μεγαλύτερος από 1 ένα υπερφορτωμένο data center.

Χρόνος προσομοίωσης: είναι ο μέγιστος επιτρεπόμενος χρόνος για την εκτέλεσης μιας εργασίας, ενώ οι προθεσμίες για τις εργασίες έχουν επίδραση στη συμπεριφορά του διαμοιρασμού χρόνου των εργασιών. Οι μεγαλύτερες προθεσμίες επιτρέπουν σε περισσότερες εργασίες να εκτελεστούν παράλληλα σε ένα host ή σε μια εικονική μηχανή.

Απαιτήσεις μνήμης: καθορίζει το μέγιστο μέγεθος των πόρων της προσομοιούμενης μνήμης, που μπορεί να χρησιμοποιηθεί για το multi-tasking.

4.6 Αποτελέσματα προσομοίωσης

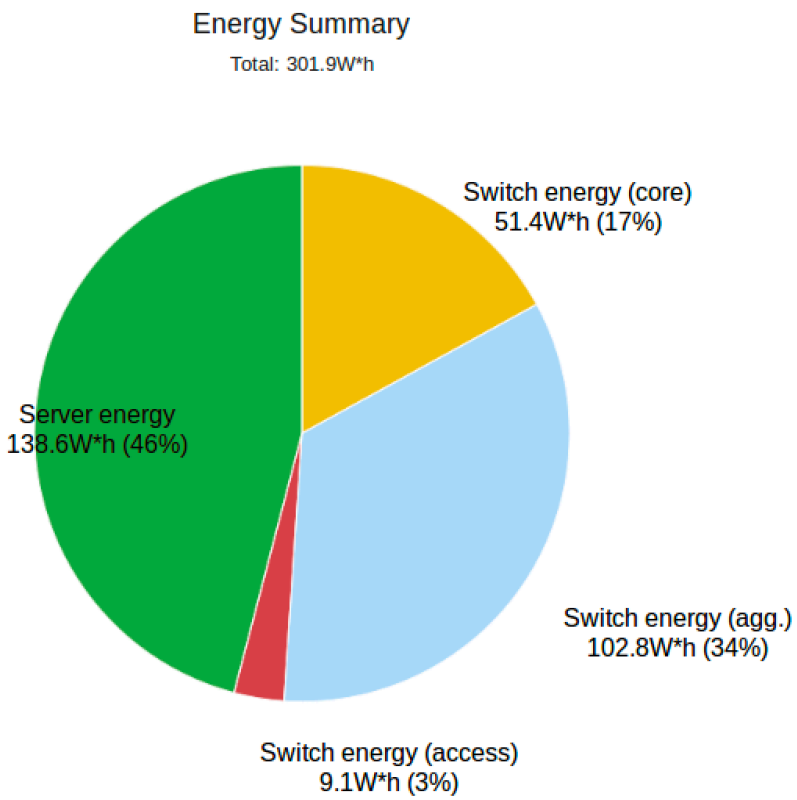
Τα αποτελέσματα είναι προσπελάσιμα ως καθαρές τιμές στα αρχεία παρακολούθησης ./traces/ ή με την μορφή dashboard και γραφικών παραστάσεων, που έχουν δημιουργηθεί από αυτές τις καθαρές τιμές. [27]

Το πρώτο τμήμα του dashboard, παρουσιάζει μια σύνοψη της προσομοίωσης και έχει ως αναγνωριστικό όνομα την ώρα και την ημερομηνία που εκτελέστηκε. Η σύνοψη περιλαμβάνει ένα κείμενο με όλα τα χαρακτηριστικά της προσομοίωσης και ένα διάγραμμα πίτας, το οποίο δείχνει την συνολική ενέργεια που καταναλώνεται από τα διαφορετικά IT συστήματα του data center.

Το τμήμα Data Center αναπαριστά την κατάσταση των στοιχείων του data center. Αποτελείται από τρεις βασικούς δείκτες: τον υπολογιστικό φόρτο, τον φόρτο της μνήμης και τον φόρτο της αποθήκευσης. Ο φόρτος παρουσιάζεται ως μια τιμή ανάμεσα στο 0 και το 1 και σχεδιάζεται ανάλογα με τον χρόνο της

Simulation duration (sec.):	65.5
Datacenter architecture:	three-tier debug
Switches (core):	1
Switches (agg.):	2
Switches (access):	3
Servers:	144
Users:	1
Power mode (servers):	DVFS DNS
Power mode (switches):	DVFS
task.mips:	300000
task.memory:	1000000
task.storage:	0
task.size:	8500
task.outputsize:	250000

Εικόνα 43: Η σύνοψη της προσομοίωσης



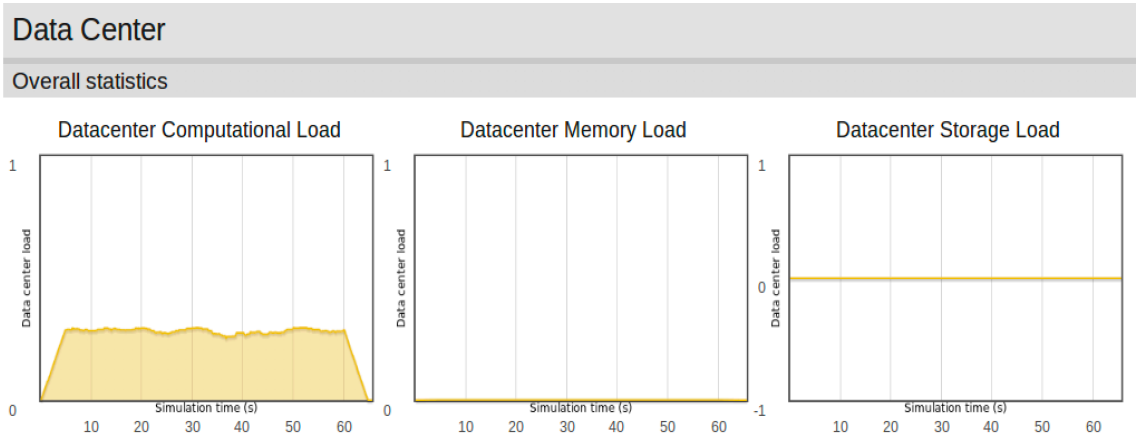
Εικόνα 44: Διάγραμμα πίτας με την συνολική ενέργεια που έχει καταναλωθεί

προσομοίωσης. Οι γραφικές παραστάσεις για τον φόρτο της μνήμης και τον φόρτο της αποθήκευσης, παρουσιάζονται με τον ίδιο τρόπο.

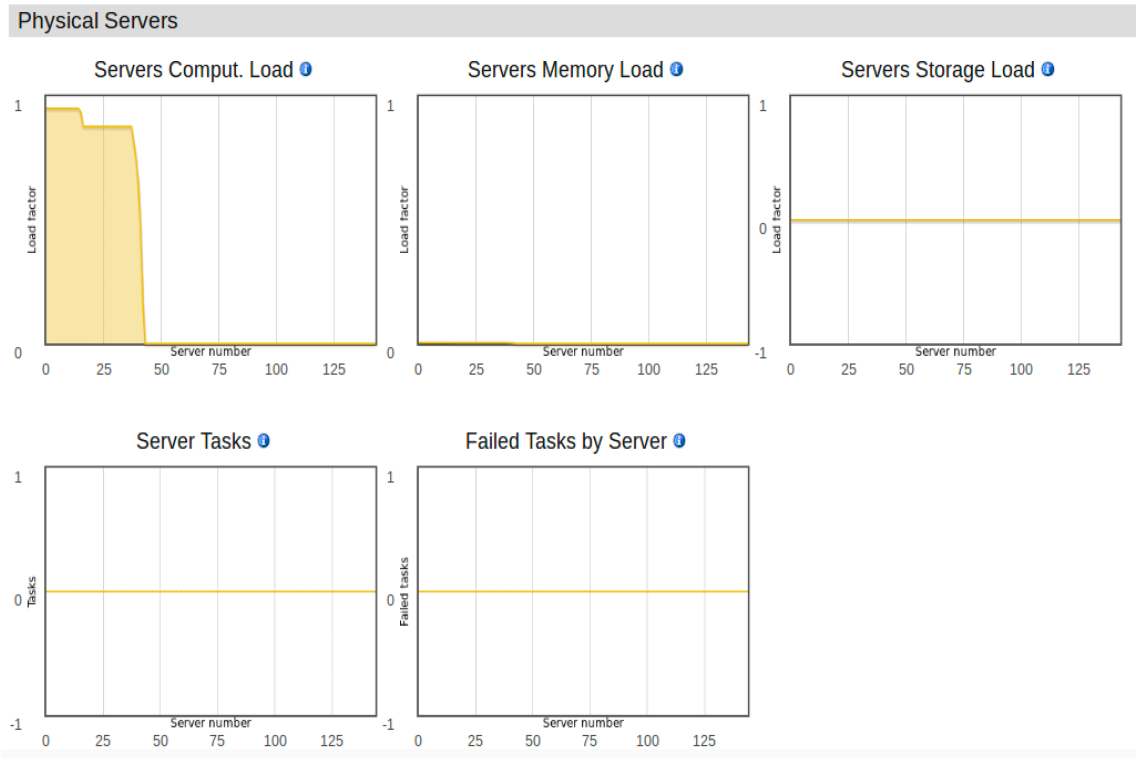
Κατά τον ίδιο τρόπο παρουσιάζονται τα τμήματα για τους Servers, όπως φαίνεται στην εικόνα 46.

και για τις εικονικές μηχανές, όπως φαίνεται στην εικόνα 47. Τέλος φαίνονται οι εργασίες που έχουν υποβληθεί και αυτές που έχουν αποτύχει για κάθε server και κάθε εικονική μηχανή.

Το τμήμα δικτύου του Data Center, που φαίνεται στην εικόνα 48 περιγράφει τον φόρτο στις συνδέσεις



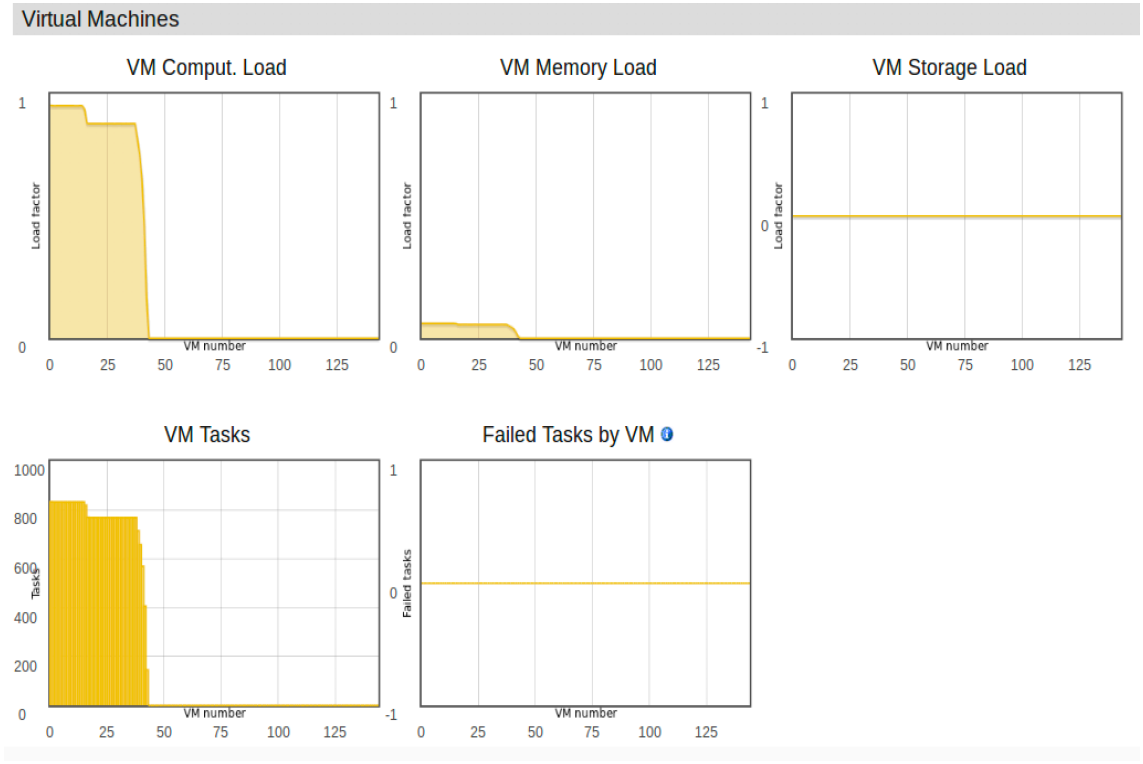
Εικόνα 45: Τα αποτελέσματα στο τμήμα Data Center



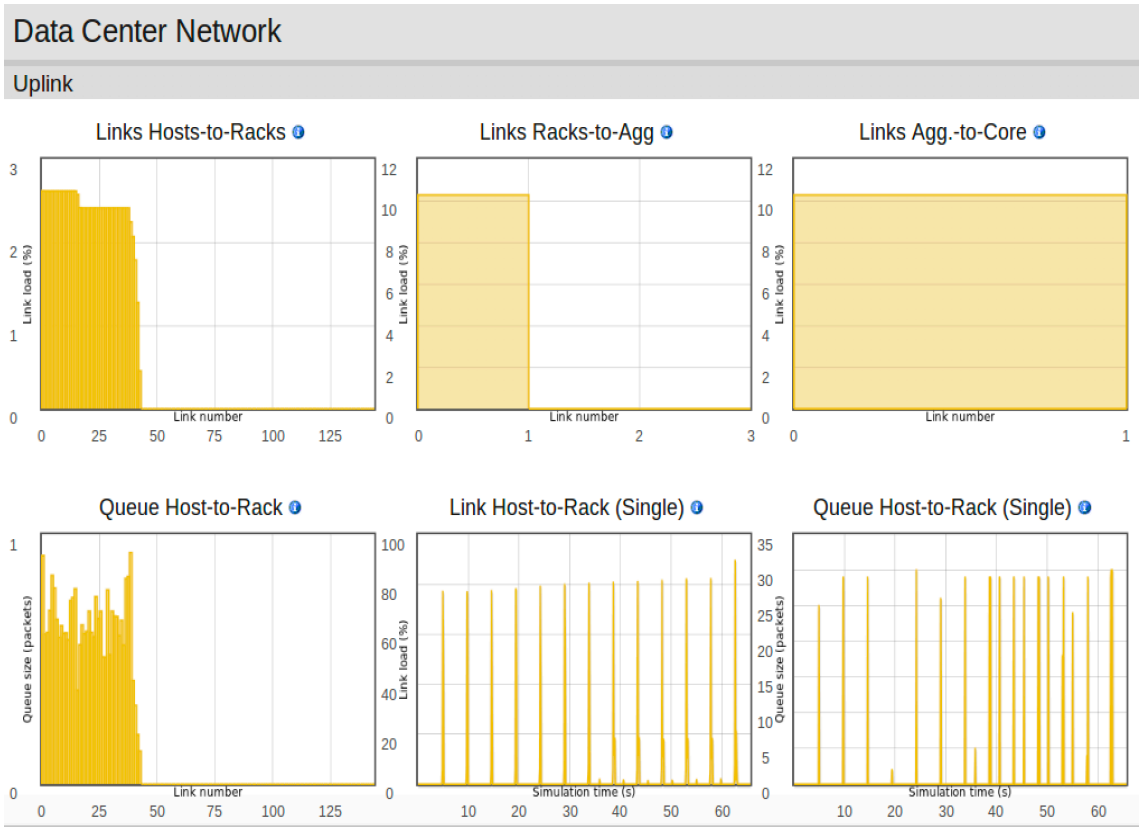
Εικόνα 46: Τα αποτελέσματα στο τμήμα των Servers

Hosts-to-Racks, Racks-to-Aggregation, Aggregation-to-Core και το μέγεθος της ουράς στις δικτυακές συνδέσεις του data center.

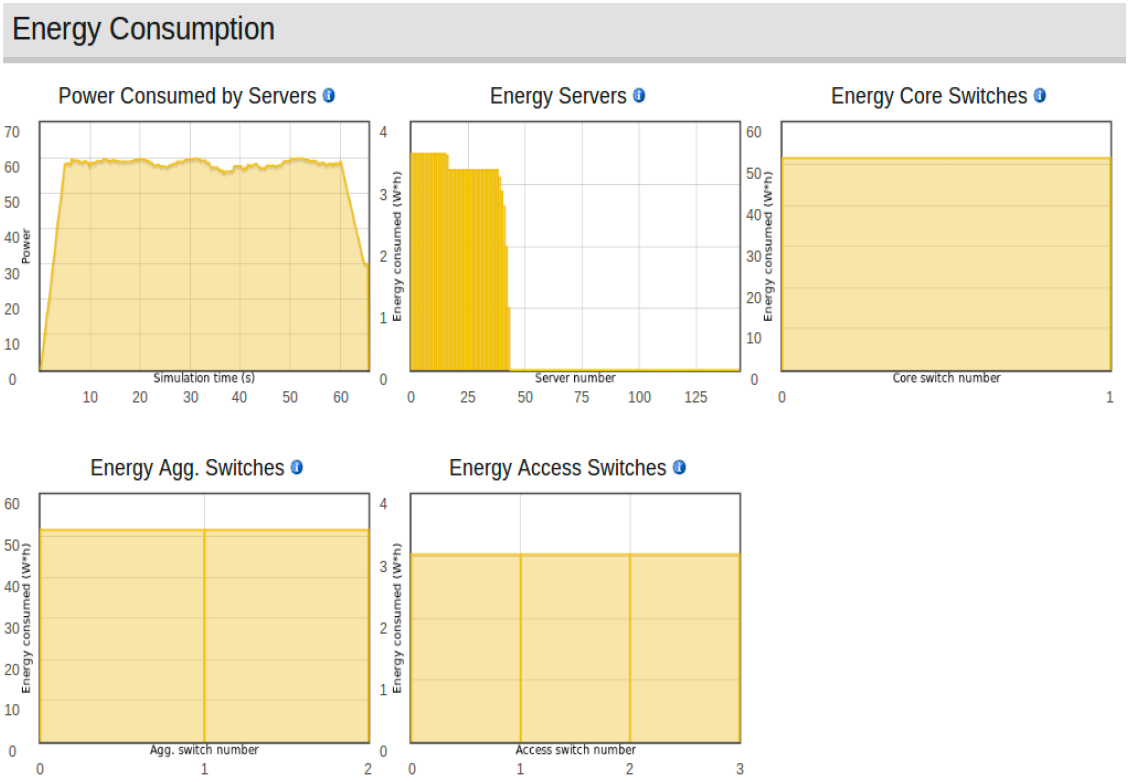
Το τελευταίο τμήμα της εικόνας 49 παρουσιάζει τα στατιστικά για την ενεργειακή κατανάλωση, ως τη συνολική κατανάλωση ανά server ή ανά core, aggregation ή access switch.



Εικόνα 47: Τα αποτελέσματα στο τμήμα των εικονικών μηχανών



Εικόνα 48: Τα διαγράμματα για τους φόρτους στις συνδέσεις



Εικόνα 49: Η ενεργειακή κατανάλωση στο Data Center

Συμπεράσματα

Κλείνοντας την μεταπτυχιακή διατριβή, γίνεται σύγκριση μεταξύ των προσομοιωτών CloudSim, CloudAnalyst και GreenCloud που αναλύθηκαν στα προηγούμενα κεφάλαια. Είναι αρκετά σημαντικό, πριν την υλοποίηση μιας cloud υπηρεσίας, ο χρήστης να μπορεί να προσομοιώσει την συμπεριφορά της υποδομής, το κόστος, τον φόρτο των συνδέσεων και άλλων σημαντικών παραμέτρων για την καλύτερη εμπειρία χρήσης. Οι πάροχοι υπηρεσιών όπως επίσης οι ερευνητές και οι χρήστες, έχουν πλέον την δυνατότητα να χρησιμοποιήσουν πολλούς και διάφορους προσομοιωτές, ανάλογα με τις απαιτήσεις και το εύρος των εφαρμογών που θέλουν να αναπτύξουν. Ακολουθούν τα χαρακτηριστικά των προσομοιωτών, όπως αυτά ερευνήθηκαν κατά την συγγραφή της παρούσας διατριβής.

Πλατφόρμα: Ένα από τα κυριότερα χαρακτηριστικά των προσομοιωτών είναι η πλατφόρμα στην οποία έχουν βασιστεί και τρέχουν. Το CloudSim χρησιμοποιεί την SimJava, το CloudAnalyst είναι βασισμένο στο CloudSim και το GreenCloud, τη γνωστή πλατφόρμα NS2.

Διαθεσιμότητα: Οι τρεις προσομοιωτές βασίζονται σε κώδικα ανοιχτού λογισμικού, όπου ο καθένας μπορεί να επέμβει και να τους τροποποιήσει ανάλογα με τις απαιτήσεις και τα πειράματα που θέλει να διεξάγει.

Γλώσσα Προγραμματισμού: Τα πακέτα προσομοίωσης CloudSim και CloudAnalyst είναι υλοποιημένα με την γλώσσα προγραμματισμού Java. Πρόκειται για την δημοφιλέστερη γλώσσα και οι προγραμματιστές, μπορούν εύκολα να καταλάβουν των κώδικα των προσομοιωτών. Αντιθέτως, το GreenCloud είναι υλοποιημένο με την C++, που είναι και αυτή μια δημοφιλής γλώσσα, αλλά και με την TCL που απαιτείται προηγούμενη εμπειρία για την κατανόησή της.

Ανάλυση Κόστους: Το CloudSim και το CloudAnalyst παρέχουν τη δυνατότητα για ανάλυση του κόστους, μιας cloud εφαρμογής σε ένα data center. Από την άλλη μεριά, το GreenCloud δεν παρέχει ανάλυση κόστους.

Γραφικό Περιβάλλον: Όσον αφορά το γραφικό περιβάλλον ο καλύτερος προσομοιωτής είναι το CloudAnalyst. Παρέχει μια λεπτομερής γραφική διεπαφή χρήστη και είναι αρκετά κατανοητό στη χρήση του. Οι προσομοιωτές CloudSim και GreenCloud έχουν περιορισμένο γραφικό περιβάλλον.

Μοντέλο επικοινωνιών: Είναι η απεικόνιση των επικοινωνιών στο εσωτερικό ενός data center για τις συνδέσεις uplink και downlink. Σε αυτό το χαρακτηριστικό υπερέχει το GreenCloud, προσφέροντας ένα πλήρες μοντέλο. Οι άλλοι δύο προσομοιωτές δεν έχουν μοντέλο επικοινωνιών.

Χρόνος προσομοίωσης: Κατά την διάρκεια της προσομοίωσης, τα εργαλεία CloudSim και CloudAnalyst τρέχουν ανά δευτερόλεπτο, ενώ το GreenCloud τρέχει ανά λεπτό.

Ενεργειακό μοντέλο: Ο προσομοιωτής που είναι σχεδιασμένος να παρέχει πληροφορίες σχετικά με την κατανάλωση ενέργειας σε ένα data center, είναι το GreenCloud. Όμως, το χαρακτηριστικό του ενεργειακού μοντέλο, δεν λείπει και από τους άλλους δύο προσομοιωτές.

Εγκατάσταση: Τέλος για την εγκατάσταση και των τριών προσομοιωτών δεν υπήρξε κάποιο πρόβλημα ή δυσκολία. Αντιθέτως, μπορούν να εγκατασταθούν σε οποιοδήποτε λειτουργικό σύστημα με την χρήση κάποιου hypervisor προγράμματος. Οι χρήστες, από αρχάριοι έως προχωρημένοι, ακολουθώντας τις οδηγίες εγκατάστασης που αναφέρονται στη διατριβή, θα ξεκινήσουν γρήγορα να φτιάχνουν τις δικές τους προσομοιώσεις.

Τα χαρακτηριστικά των προσομοιωτών φαίνονται συνοπτικά στον πίνακα 15.

Προσομοιωτές	CloudSim	CloudAnalyst	GreenCloud
Πλατφόρμα	SimJava	CloudSim	NS2
Διαθεσιμότητα	Open Source	Open Source	Open Source
Γλώσσα Προγραμματισμού	Java	Java	C++,TCL
Ανάλυση Κόστους	Ναι	Ναι	Όχι
Γραφικό περιβάλλον	Όχι	Ναι	Περιορισμένο
Μοντέλο Επικοινωνιών	Περιορισμένο	Περιορισμένο	Πλήρες
Χρόνος Προσομοίωσης	Δευτερόλεπτο	Δευτερόλεπτο	Λεπτό
Ενεργειακό Μοντέλο	Ναι	Ναι	Ναι

Πίνακας 15: Σύγκριση των προσομοιωτών

Ακρωνύμια

API Application Programming Interface. 39
ARPANET Advanced Research Projects Agency Network. 13

BLs Balanced Workloads. 67

CaaS Containers as a Service. 11, 15, 30
CIS Cloud Information Service. 26, 29
CIWs Intensive Workloads. 67
CPU Central Processor Unit. 28, 63, 66
CSNET Computer Science Network. 13

DCiE Data Center Infrastructure Efficiency. 63
DCN Data Center Network. 63, 64
DIWs Data Intensive Workloads. 67
DPM Dynamic Power Management. 62
DVFS Dynamic Voltage and Frequency Scaling. 62, 66

ECMP Equal Cost Multi Path. 64

FCFS First-Come-First-Serve. 24
FLOPS Floating Point Operations per Second. 66

GE Gigabit Ethernet. 65, 66
GIS Geographic Information System. 67
GUI Graphical User Interface. 6, 41, 43

HPC High Performance Computing. 67

IaaS Infrastructure as a Service. 11, 15, 30
ICMP Internet Control Message Protocol. 64
IDE Integrated Development Environment. 11, 31, 48, 49, 68
IP Internet Protocol. 63

JDK Java Development Kit. 31
JVM Java Virtual Machine. 28

MIPS Million Instructions per Second. 24, 28, 34–36, 66

OPEX Operating Expenditure. 62

PaaS Platform as a Service. 11, 14, 30
PUE Power Usage Effectiveness. 63

QoS Quality of Service. 22, 26, 27, 30, 68

RAM Random Access Memory. 28, 34
RJE Remote Job Entry. 13

SaaS Software as a Service. 11, 14, 30
SLA Service Level Agreement. 62, 67, 68

TCL Tool Command Language. 63, 75
TCP Transmission Control Protocol. 63, 64
ToR Top-of-Rack. 66, 67

VM Virtual Machine. 24, 26, 30
VMM Virtual Machine Manager. 34
VPN Virtual Private Network. 13

Βιβλιογραφία

[1] Peter Mell, Tim Grance, et al. “The NIST definition of cloud computing”. In: (2011).

[2] Wikipedia. *Cloud computing*. [Online; πρόσβαση Απρίλιος 2017]. 2017. URL: https://en.wikipedia.org/wiki/Cloud_computing.

[3] Jayaprakash Ramsaran. *Cloud Computing: Benefits and Challenges*. [Online; πρόσβαση Απρίλιος 2017]. 2014. URL: http://transformcustomers.com/cloud-computing-benefits-and-challenges/.

[4] Betty Junod. *Containers as a Service (CAAS) As your new platform for application development and operations*. [Online; πρόσβαση Απρίλιος 2017]. URL: https://blog.docker.com/2016/02/containers-as-a-service-caas/.

[5] Forrest Stroud. *Container-as-a-Service Platforms and Providers*. [Online; πρόσβαση Απρίλιος 2017]. URL: http://www.webopedia.com/TERM/C/caas-containers-as-a-service.html.

[6] Victor Victories and Crystal Jones. *4 Types of Cloud Computing Deployment Model You Need to Know*. [Online; πρόσβαση Απρίλιος 2017]. 2015. URL: https://www.ibm.com/developerworks/community/blogs/722f6200-f4ca-4eb3-9d64-8d2b58b2d4e8/entry/4_Types_of_Cloud_Computing_Deployment_Model_You_Need_to_Know1?lang=en.

[7] Adeeb P A. “A Seminar Report on Security in Cloud Computing”. In: (2014).

[8] Microsoft. *What is cloud computing?* [Online; πρόσβαση Απρίλιος 2017]. URL: https://azure.microsoft.com/en-gb/overview/what-is-cloud-computing/.

[9] Priya Viswanathan. *Cloud Computing and Is it Really All That Beneficial?* [Online; πρόσβαση Απρίλιος 2017]. URL: https://www.lifewire.com/cloud-computing-explained-2373125.

[10] Amazon. *Summary of the Amazon S3 Service Disruption in the Northern Virginia (US-EAST-1) Region*. [Online; πρόσβαση Απρίλιος 2017]. 2017. URL: https://aws.amazon.com/message/41926/.

[11] School of Computing and Information Systems. The University of Melbourne. *CloudSim: A Framework For Modeling And Simulation Of Cloud Computing Infrastructures And Services*. [Online; πρόσβαση Απρίλιος 2017]. URL: http://www.cloudbus.org/cloudsim/.

[12] Rajkumar Buyya, Rajiv Ranjan, and Rodrigo N Calheiros. “Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities”. In: *High Performance Computing & Simulation, 2009. HPCS’09. International Conference on*. IEEE. 2009, pp. 1–11.

[13] Rodrigo N Calheiros, Rajiv Ranjan, Anton Beloglazov, César AF De Rose, and Rajkumar Buyya. “CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms”. In: *Software: Practice and experience* 41.1 (2011), pp. 23–50.

[14] Neda Maleki. *First National Workshop of Cloud Computing*. [Online; πρόσβαση Απρίλιος 2017]. URL: https://www.slideshare.net/nedamaleki87/cloud-sim-greencloud.

[15] School of Computing and Information Systems. The University of Melbourne. *ContainerCloudSim: An Environment For Modeling And Simulation Of Containers In Cloud Data Centers*. [Online; πρόσβαση Απρίλιος 2017]. URL: http://www.cloudbus.org/cloudsim/container.html.

[16] Tarun Goyal, Ajit Singh, and Aakanksha Agrawal. “Cloudsim: simulator for cloud computing infrastructure and modeling”. In: *Procedia Engineering* 38 (2012), pp. 3566–3572.

[17] Bhathiya Wickremasinghe, Rodrigo N Calheiros, and Rajkumar Buyya. "Cloudanalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications". In: *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*. IEEE. 2010, pp. 446–452.

[18] Simar Preet Singh, Anju Sharma, and Rajesh Kumar. "Analysis of Load Balancing Algorithms using Cloud Analyst". In: *International Journal of Grid and Distributed Computing* 9.9 (2016), pp. 11–24.

[19] Bhathiya Wickremasinghe and Rajkumar Buyya. "Cloudanalyst: A cloudsim-based tool for modelling and analysis of large scale cloud computing environments". In: *MEDC project report* 22.6 (2009), pp. 433–659.

[20] LinkedIn. *About Us*. [Online; πρόσβαση Απρίλιος 2017]. URL: `https://press.linkedin.com/about-linkedin?`.

[21] timeanddate.com. *Time Zone Map*. [Online; πρόσβαση Απρίλιος 2017]. URL: `https://www.timeanddate.com/time/map/`.

[22] Amazon. *Amazon EC2*. [Online; πρόσβαση Απρίλιος 2017]. URL: `https://aws.amazon.com/ec2/`.

[23] T. S. Pradeep Kumar. *Getting Started with GreenCloud Simulator*. URL: `http://opensourceforu.com/2015/01/getting-started-greencloud-simulator/`.

[24] Dzmitry Kliazovich, Pascal Bouvry, Yury Audzevich, and Samee Ullah Khan. "GreenCloud: a packet-level simulator of energy-aware cloud computing data centers". In: *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*. IEEE. 2010, pp. 1–5.

[25] University of Luxembourg. *GreenCloud: The green cloud simulator*. URL: `https://greencloud.gforge.uni.lu/`.

[26] Kashif Bilal, Samee Ullah Khan, Joanna Kolodziej, Limin Zhang, Khizar Hayat, Sajjad Ahmad Madani, Nasro Min-Allah, Lizhe Wang, and Dan Chen. "A Comparative Study Of Data Center Network Architectures." In: *ECMS*. 2012, pp. 526–532.

[27] University of Luxembourg. *GreenCloud Simulator User Manual*. URL: `https://greencloud.gforge.uni.lu/ftp/greencloud-user-manual.pdf`.