Universidade Federal de Uberlândia
Faculdade de Engenharia Elétrica

**ALEX VAZ MENDES**

# A INTERNET DO FUTURO E AS POSSÍVEIS SOLUÇÕES E ARQUITETURAS

Uberlândia
2014

**ALEX VAZ MENDES**

# A INTERNET DO FUTURO E AS POSSÍVEIS SOLUÇÕES E ARQUITETURAS

Trabalho apresentado como requisito parcial de avaliação na disciplina Trabalho de Conclusão de Curso 2 do Curso de Engenharia Elétrica da Universidade Federal de Uberlândia.

Orientador: Luiz Cláudio Theodoro

_____

Assinatura do Orientador

Uberlândia
2014

Dedico este trabalho aos meus pais, pelo estímulo, carinho e compreensão.

## AGRADECIMENTOS

Ao Prof. Luiz Cláudio Theodoro, pelo acompanhamento de minhas atividades de pesquisa e orientação nos últimos anos e pela grande quantidade de conhecimento e experiência que me direcionou através de grupos de estudo e outras oportunidades.

À minha família, pelo suporte financeiro e emocional, além do carinho e dedicação direcionados àos meus estudos e pesquisas.

Aos colegas e amigos que auxiliaram muito no desenvolvimento da pesquisa que originou este trabalho, além de contribuírem na minha capacidade de trabalhar em equipe e construír novos conhecimentos.

## RESUMO

Este trabalho apresenta uma pesquisa direcionada para as soluções de Internet do Futuro. Primeiramente baseada na amplitude da rede e na crescente evolução das aplicações e dos negócios envolvidos. Além disso, fatores limitantes da arquitetura atual da Internet são fundamentais para as novas possibilidades de pesquisa, visto que a rede não é capaz de oferecer toda a qualidade de serviço necessária para todas as funcionalidades que pode assumir. Finalmente é realizado um estudo sobre o modelo de arquitetura definida por software, e a partir deste descrever vários projetos construídos pelo mundo, desde plataformas para testes até controladores e switches programáveis, que possam atuar baseados em um modelo virtualizado da rede e na separação dos planos de controle e encaminhamento.

## ABSTRACT

This work introduces a research aimed to solutions of Future Internet. Primarily based on the range of the network, and also, in the increasing evolution of applications and involved businesses. In addition, limiting factors of the current Internet architecture are essential for the new research possibilities, because the network is not able to offer all quality of service required for all features that can take. Finally, a study was conducted about the model of software defined architecture, and from this, describe several projects built in the world, since platforms for tests to controllers and programmable switches, that can act based on a virtualized network model and in the separation of control and forwarding planes.

## LISTA DE ILUSTRAÇÕES

# LISTA DE TABELAS

## LISTA DE ABREVIATURAS E SIGLAS

API: Application Programming Interface, 25

BGP: Border Gateway Protocol, 54

DHCP: Dynamic Host Configuration Protocol, 15

DTS: Domain Title Service, 58

DTSA: Domain Title Service Agent, 60

ETArch: Entity Title Architecture, 53

FIRE: Future Internet Research and Experimentation, 21

FP7: Seventh Framework Programme, 20

GENI: Global Environment for Network Innovations, 21

I/O: Input/Output, 41

IEEE: Institute of Electrical and Electronics Engineers, 53

IoT: Internet of Things, 18

IP: Internet Protocol, 13

M2M: Machine-to-Machine, 18

MV: Máquina Virtual, 54

NAT: Network Address Translation, 15

NGN: Next Generation Network, 50

NIC: Network Interface Controller, 55

NwGN: New Generation Network, 50

OFELIA: OpenFlow in Europe: Linking Infrastructure and Applications, 21

OSI: Open Systems Interconnection, 13

OSPF: Open Shortest Path First, 54

OVS: Open V Switch, 55

OXM: OpenFlow eXtensible Match, 32

QoS: Quality of Service, 15

SDN: Software-Defined Networking, 20

TCP: Transmission Control Protocol, 13

UDP: User Datagram Protocol, 14

VLAN: Virtual Local Area Network, 31

VoIP: Voice over Internet Protocol, 31

# SUMÁRIO

# 1 INTRODUÇÃO

## 1.1 A Internet Atual

A Internet é um tema altamente valorizado, principalmente pela sua abrangência e pela capacidade de retenção de usuários que possui. Exaltando o número de brasileiros da rede, estima-se que desde o primeiro trimestre de 2013 a quantidade de pessoas que vivem no Brasil com acesso a Internet ultrapassou 100 milhões [1].

Se em um cenário nacional os números já impressionam, considerando que este desenvolvimento da rede mundial de computadores é lento no Brasil se comparado aos países desenvolvidos, o impacto mundial desta é ainda mais impressionante.

Dados divulgados no início de 2013 mostram estimativas de 2,4 bilhões de usuários em todo o mundo, com cerca de 634 milhões de sites, sendo que destes 51 milhões foram criados no próprio ano de 2012. Outros números são impressionantes como o tráfego de 144 bilhões de e-mails diariamente, e apontam para uma utilização massiva dos recursos disponíveis na rede [2]. Com uma massa de usuários tão grande fica evidente a importância deste tipo de ferramenta no que se diz ao desenvolvimento de negócio em si.

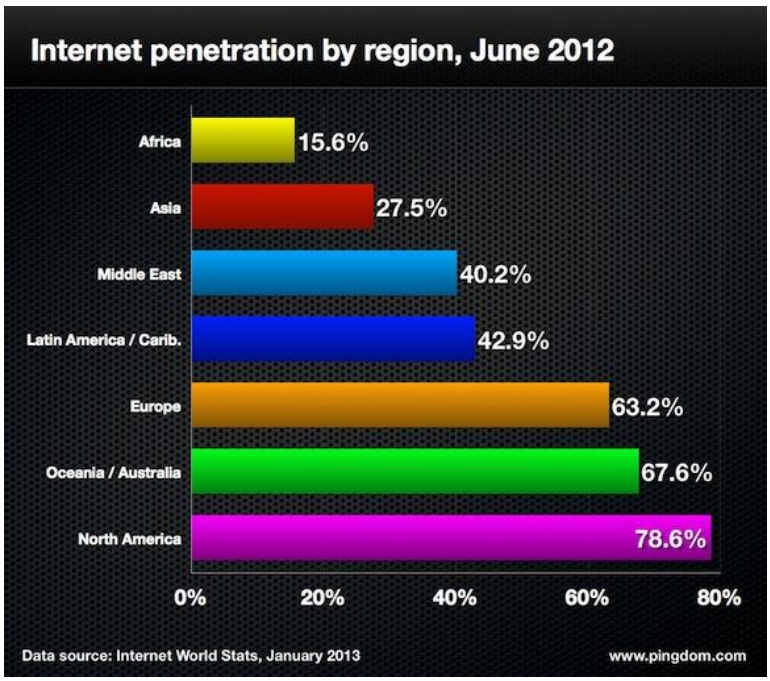


**Figura 1: Penetração da Internet por região em junho de 2012 [2]**

Com estes números em mãos e as evidências da importância da Internet, é necessário destacar um breve histórico e descrição da mesma também. A Internet nada mais é do que uma rede de

redes, formada pela conexão de várias redes locais e tem como pontos finais os hosts, que podem ser servidores ou dispositivos de acesso para os usuários (celulares, computadores, tablets, entre outros). De forma breve o histórico abrange o desenvolvimento da comutação de pacotes (1961 a 1972), as redes proprietárias (1972 a 1980), a proliferação (1980 a 1990) e a expansão (a partir da década de 90) [3]. Este breve histórico aponta uma curiosidade, na época da proliferação, quando foram criados e amplamente aceitos os protocolos da arquitetura TCP/IP, base da rede até hoje, a Internet em si não conseguia desenvolver-se até o impulso causado por esta arquitetura. O ponto para que se chama a atenção é que 30 anos depois a arquitetura ainda é a mesma de uma época onde o que era buscado era um padrão para que as redes pudessem se comunicar.

Considerado esse tempo podemos imaginar uma rede em crescimento e com números expressivos suportada por uma tecnologia criada a 30 anos que tinha entre seus princípios a simplicidade e a divisão das tarefas em camadas [4].

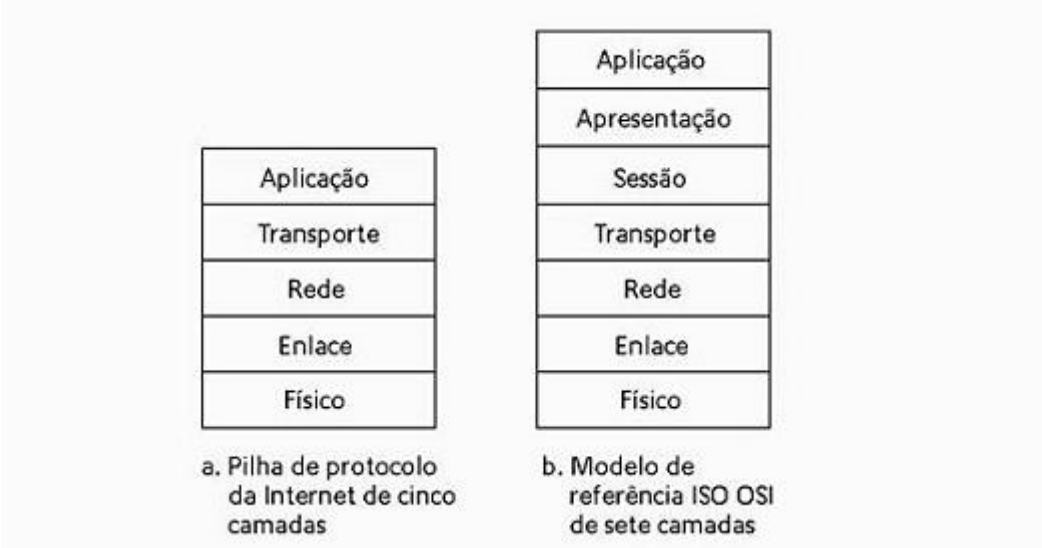


**Figura 2: As camadas dos modelos TCP/IP e OSI [3]**

O Modelo OSI foi criado na década de 1970 e formalizado pela ISO (International Organization for Standardization) uma organização internacional para normatização e apresenta uma certa conexão com o modelo da Internet (apenas existe a exclusão de duas camadas: Apresentação e Sessão) [3]. Olhando para as cinco camadas do modelo TCP/IP vemos primeiramente uma divisão de tarefas, cada camada executa uma parte do trabalho de interligar dois hosts na rede. Partindo da base, a camada física é indispensável, visto que é necessário um meio físico para o transporte dos dados sendo este cabos coaxiais, fibras ópticas, ou até o ar. Neste ponto há

grande evolução ao longo dos anos através das velocidades de transmissão cada vez maiores, possíveis através das fibras ópticas, o que tem mobilizado projetos brasileiros como o Projeto GIGA desenvolvido no CPqD (Centro de Pesquisa e Desenvolvimento) [5].

A camada de enlace, na Internet é representada pelo protocolo Ethernet. O único objetivo desta é o transporte de dados entre pontos adjacentes, o que pode ser entre switches, hosts e roteadores.

Por enquanto ignorando as camadas do meio, a última camada é a de Aplicação, e esta é a camada onde se vê o maior desenvolvimento nos últimos anos. As aplicações da rede estão cada vez mais complexas e carregam um nível de inteligência altíssimo se comparadas às que existiam poucos anos atrás. E ainda mais estas estão tomando cada vez mais o rumo do negócio, já que faz interface direta com o usuário, e o público deste meio de comunicação é enorme [6][7].

Diante destas informações voltando às camadas de rede e transporte, chegamos nas funções de conexão lógica entre hosts (rede) ou de aplicações (transporte). Estas camadas são o ponto crucial para conectar as máquinas na rede e as aplicações, que como comentado anteriormente são muito complexas. Na camada de transporte existem dois protocolos muito importantes o TCP (Transmission Control Protocol) e o UDP (User Datagram Protocol), sendo que a diferença principal entre os dois é a garantia de entrega oferecida pelo primeiro. Já na camada de rede o protocolo que realiza a função é o IP (Internet Protocol) [3]. Com o foco no IP ao observarmos seus campos vemos uma estrutura antiga e engessada, rígida e que oferece apenas a entrega dos dados de ponto a ponto.

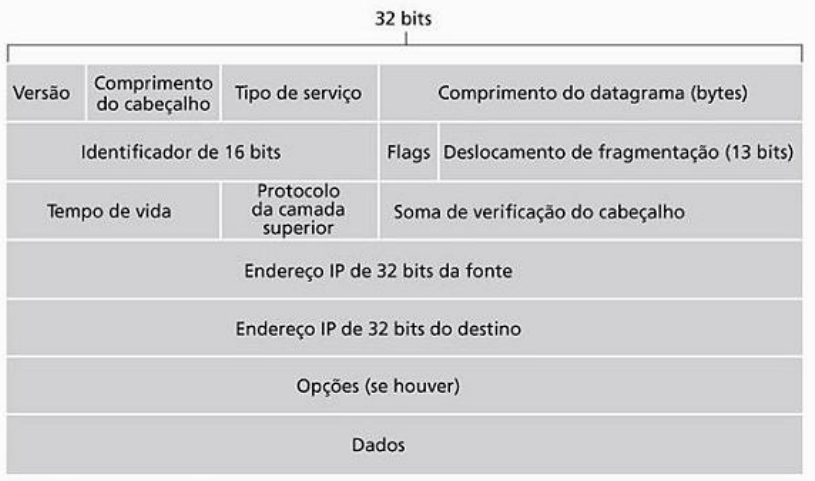


**Figura 3: Cabeçalho do IPv4 [3]**

## 1.2 Algumas limitações

O IP evoluiu ao longo dos primeiros anos até chegar ao IPv4 (quarta versão do protocolo) que é usado até os dias atuais, mas vem sendo substituído por uma versão que disponibiliza mais endereços possíveis, o IPv6 (sexta versão), já que os 32 bits separados para identificar cada host na internet, e tecnologias como DHCP (Dynamic Host Configuration Protocol) e o NAT (Network Address Translation) não foram capazes de suprir as necessidades crescentes de endereços, e assim passarão a ser 128 bits destinados a esta tarefa.

Na verdade o ponto central está no que se chama de gargalo IP, sendo que com o tempo foram surgindo novos protocolos em várias camadas para atender novas demandas, principalmente na camada de aplicação, e todos estes tiveram que se adaptar ao IP para que a ligação entre hosts pudesse acontecer [4].

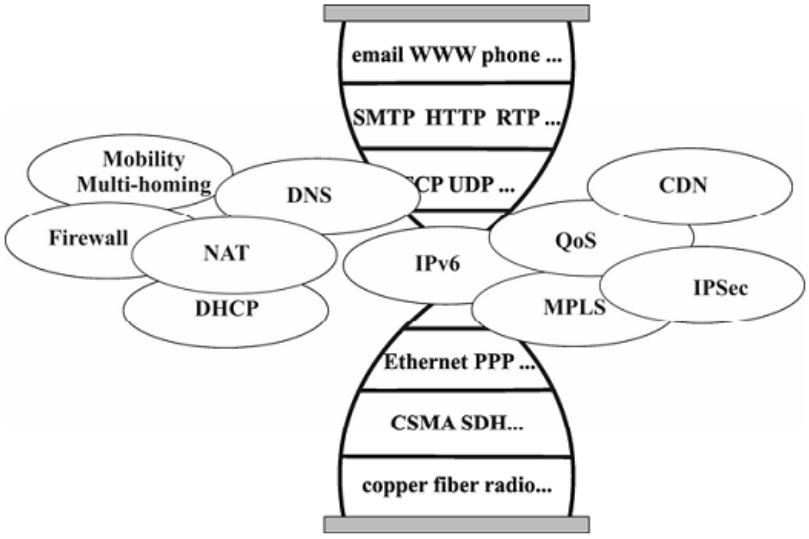


**Figura 4: O Gargalo IP [4]**

Este gargalo é assim representado pelas limitações que o protocolo IP apresenta, isso que dizer que a incapacidade desta camada de oferecer certos serviços de QoS (Quality of Service) obrigou as outras camadas, principalmente a de aplicação, a concentrarem as funcionalidades para o desenvolvimento dos serviços oferecidos.

Uma série de pesquisadores vêm concentrando seus esforços em novas tecnologias pela evidente limitação da arquitetura atual, que age como barreira em diversos pontos onde há necessidade de melhora. Estes pontos podem ser destacados como problemas com mobilidade, já que a arquitetura IP não permite mudar o padrão de conexão com a rede (*handover*) sem que

o fluxo de dados seja perdido, simplesmente porque o endereço IP muda e todo o resto da estrutura fica afetada. Outros pontos de limitação estão no plano de controle e segurança, já que não existe nada previsto na camada de rede que permita o controle dos fluxos de dados com precisão e de acordo com a aplicação, ou seja, muitas vezes aplicações que necessitam de alta largura de banda são afetadas por outras que funcionam em rajadas e não necessitam de velocidade tão grande na busca da informação, como por exemplo a concorrência do fluxo de uma vídeo conferência com uma simples navegação em páginas web. Outros problemas estão relacionados com processamento, tratamento e endereçamento dos pacotes [8].

## 1.3 Novas abordagens e requisitos para a Internet

Com todas estas considerações é possível compreender a situação atual da Internet, destacadas a dimensão da mesma além da arquitetura e problemas. Todos estes fatores são fundamentais para o impulso a pesquisas e desenvolvimentos procurando alternativas para a resolução destes problemas, ou simplesmente novos protocolos e soluções adaptadas para cada serviço novo que apareça.

Para as novas soluções existem duas alternativas, as quais dividiram grupos por todo o mundo na busca de ideias para a Internet do futuro. A primeira, que já é amplamente e claramente desenvolvida pelo mundo é o aperfeiçoamento do que já existe de estrutura hoje, isso quer dizer que, para cada novo serviço que surja, novos protocolos ou um aperfeiçoamento da aplicação, dentro da estrutura de camadas que já existe hoje, devem aparecer para suprir a necessidade. Isso cria uma estrutura cada vez mais complicada, onde as novidades tem que se adaptar ao que já é feito nas camadas da internet, lembrando sempre do gargalo anteriormente citado existente na camada de rede.

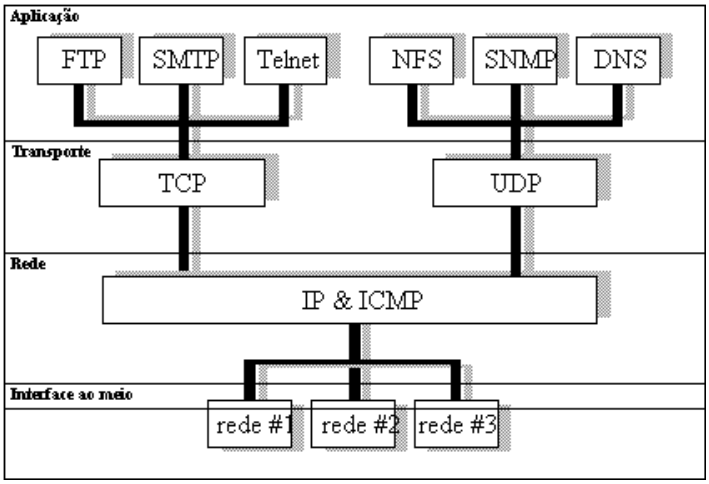


**Figura 5: Os protocolos da Internet [9]**

A figura 5 dá uma visão clara do problema desta primeira abordagem, as camadas de transporte a aplicação são todas suportadas pelos protocolos TCP, UDP e IP (o protocolo ICMP é usado para sinalização e sincronização), assim cada nova aplicação, que siga um novo paradigma, precisa se adaptar ao que estes oferecem e não têm acesso direto as camadas inferiores, ou seja, serviços de QoS como a largura de banda, não podem ser requisitados diretamente pela aplicação, por estarem em camadas inferiores (enlace e física) e por que estes protocolos anteriormente citados não são capazes de prover tal serviço. Assim o gargalo fica mais evidente, com uma infinidade de protocolos de aplicação suportados por uma tecnologia limitada e antiga. Embasado nestes problemas surge a segunda alternativa para a Internet do futuro, conhecida como *clean slate*, a qual tem ganhado bastante força nos últimos anos. Esta propõe uma visão completamente radical onde toda a estrutura atual é abandonada e uma nova arquitetura seria desenvolvida, com novos paradigmas e protocolos.

Alguns grupos radicais rejeitam aproveitar qualquer estrutura da arquitetura atual para a nova, enquanto outros vêem com bons olhos utilizar alguns pontos da estrutura que já existe para que a provável mudança não seja tão drástica, e o que já exista não seja completamente perdido.

É dentro desta perspectiva que vários projetos já foram criados pelo mundo, os quais visam ser capazes de atender necessidades fundamenteais e oferecer certos requisitos. Alguns destes requisitos são claros, os quais vão desde o desenvolvimento da própria arquitetura, até tendências já declaradas da Internet atual, sendo eles: paradigmas orientados a dados ou conteúdo, acesso ubíquo com alta mobilidade, foco em computação na núvem (Cloud Computing), segurança e por fim *testbeds* (plataformas de testes) com alta confiabilidade e disponibilidade [10]. Diante disso é que surge um triângulo que define os três pontos fundamentais para a Internet oferecer serviços mais inovadores e orientados ao negócio.

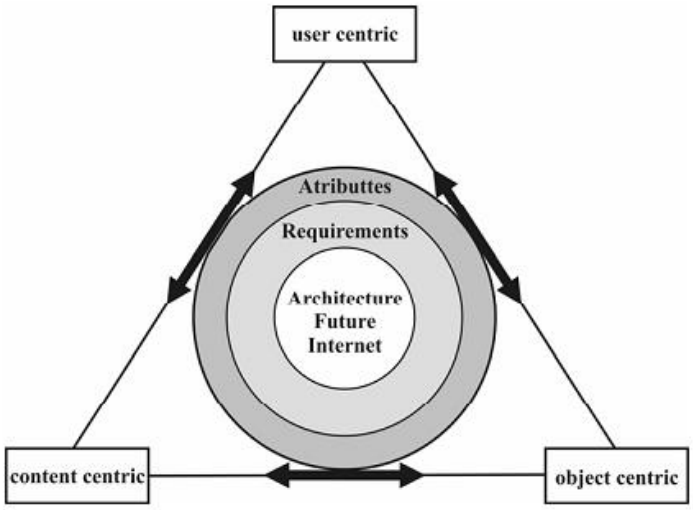


**Figura 6: Cenários para o desenvolvimento da Internet [4]**

Estes três pontos são bem claros, de forma que somados aos atributos e requisições são fundamentais em qualquer nova proposta de arquitetura para a Internet do futuro. Primeiramente o *user centric* ou centrado no usuário preve um acesso cada vez mais personalizado à rede, isso quer dizer que para cada nova pessoa que começar a usar os serviços oferecidos, estes se adaptam automaticamente ao perfil da mesma, o que fornece buscas personalizadas e direcionadas, formatação e disponibilidade de opções conforme o usuário, entre outras inúmeras vantagens que orientam o usuário ao que ele busca especificamente na rede. Isto já é bastante presente na rede, mas ainda limitado por uma série de fatores já citados e inclusive o problema da segurança, visto que informações importantes pessoais estariam circulando cada vez mais na rede.

O segundo ponto é o *object centric* ou centrado nos objetos que é atualmente relacionado com o que é chamado de Internet das coisas (IoT) ou até M2M (Machine-to-Machine). Neste ponto o importante são os objetos, e o intuito é a conexão de cada vez mais dispositivos na web, o que possibilita interação a distância e rápida com uma série de equipamentos, que podem variar desde uma geladeira até lâmpadas e outros aparelhos, o que resulta em ganhos em praticidade e comunicação. Isto já é implementado com a tecnologia atual, mas é claro que não se sabe até quanto os endereços e a forma de enderçamento IP serão satisfatórios para tal propósito.

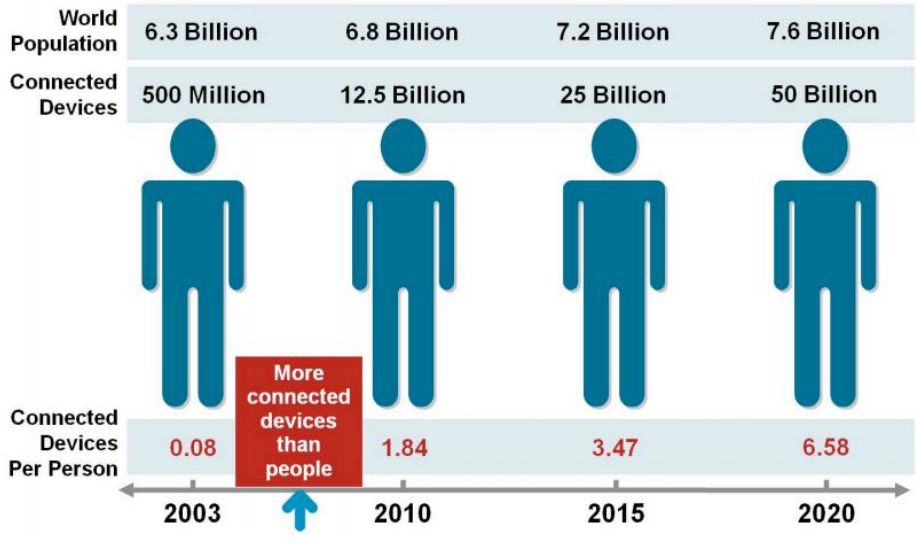


**Figura 7: Conexão de usuários e dispositivos na rede [11]**

Como indicado na figura 7, a quantidade de dispositivos na rede já ultrapassou o número de usuários e a expectativa é que o crescimento será cada vez mais expressivo, até o momento em

que tudo possa estar conectado na rede. A IoT é um passo importante para o desenvolvimento de serviços e produtos e revela-se um grande negócio para os próximos anos [11].

O último ponto é o content-centric, ou seja, centrado no conteúdo. O conteúdo talvez seja o grande produto da Internet, porque esta oferece uma grande quantidade de informação em alta velocidade, mas ainda apresenta um problema, o conteúdo está espalhado, muitas vezes replicado em vários pontos, e de acordo com o assunto o acesso é difícil. Este problema é evidenciado pela grande quantidade de investimentos em *Big Data*, isso quer dizer, serviços e hardwares capazes de lidar com alta quantidade de informação. Assim destaca-se o objetivo de se fazer a informação cada vez mais organizada e disponível para o usuário final. É neste ponto que aparece o conceito de metadata, ou metadados, que nada mais é do que a criação de dados que informem sobre outros dados, ou seja, a rede estaria totalmente ligada de acordo com o conteúdo que oferece, de forma automática e precisa.

É óbvio que uma série de fatores complica a implantação de todos estes pontos, e é por isso que esta não é uma transição rápida ou simples. Assim surgem os desafios para toda a pesquisa direcionada para as novas soluções de Internet, os quais podem ser resumidos em: capacidade, eficiência, performance, ubiquidade, generalidade, escalabilidade, entre outros, sendo que todos estes tem um apelo financeiro muito importante [4].

Dentre os aspectos que mais apontam para a Internet do Futuro, como os anteriormente citados, é impossível não citar a virtualização e pode-se dizer que este é quase um termo genérico quando se fala em tecnologia computacional. Toda a integração de pessoas, conteúdo e objetos na rede é baseada na virtualização, ou seja, na simulação de algo físico sobre algo virtual. Isto já é altamente utilizado nos *DataCenters* hoje, visto que um servidor físico pode atender a necessidade de vários clientes ou solicitantes de um serviço de hospedagem ao mesmo tempo, assim vários outros servidores são simulados dentro de um mesmo equipamento físico. É óbvio que isto reduz a capacidade e a exclusividade de quando se usa apenas uma máquina, mas amplifica a utilização e torna possível que se tratem várias requisições de diferentes sistemas em uma máquina só. E mais, cada máquina virtual pode ser vista pelo “lado de fora” como uma máquina separada e independente, apesar do compartilhamento. E mais, quando uma máquina física apresenta defeitos, as máquinas virtuais podem migrar de ambiente e serem usadas sob outra máquina [12].

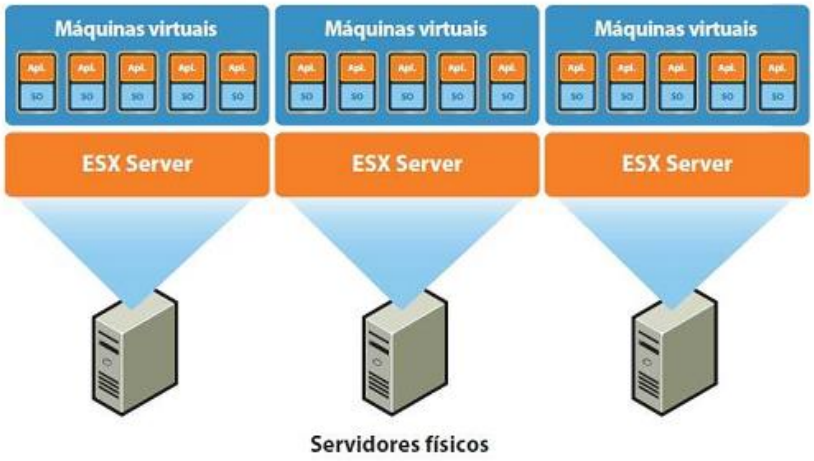


**Figura 8: Virtualização de Servidores [12]**

Mas neste caso é importante destacar que a virtualização é capaz de abstrair recursos físicos para um ambiente virtual. Isto traz a capacidade de que ocorra uma integração maior do que existe de recursos para um sistema e sobre a forma de controlá-los sobre o software. É nesse contexto que surge a SDN (Software-Defined Networking). Esta é uma proposta que será posteriormente descrita, e que movimenta uma quantidade muito grande de recursos financeiros pelo mundo.

Vários projetos se desenvolveram ao entorno desta, no Brasil e no mundo, movimentando grandes empresas e despertando interesse de várias entidades importantes.

É neste ponto em que se destaca este trabalho, desenvolvido com o intuito de analisar algumas das propostas para a Internet do Futuro, focando principalmente em algumas arquiteturas, para desenvolver um documento com informações importantes e relevantes sobre estas, observando alguns ganhos e inovações nas mesmas.

Isto é importante primeiramente para difundir tais propostas e pesquisas tão importantes no ambiente brasileiro e regional. A distribuição de recursos é tão grande que foram criados programas para a distribuição dos mesmos, como é o caso do FP7 (Seventh Framework Programme), o qual foi desenvolvido na Europa e disponibilizou em 2013 um total de 8,1 bilhões de euros para pesquisas na linha de Internet do Futuro, parte destes recursos inclusive foram direcionados ao Brasil [13].

Mais do que apenas os recursos em si, mas o grande montante financeiro disponibilizado mostra a abrangência e a importância que foi atribuída para as inovações e novas estruturas que são criadas para suprir as necessidades do futuro.

A SDN em sí já chama bastante atenção pela quantidade de grandes empresas e fornecedores de equipamentos de rede preocupados com o assunto.

Também, a abrangência da Internet e a capacidade cada vez maior de gerar negócios desta é outro fator citado anteriormente e que motiva o envolvimento direto com estas pesquisas. Grande parte das grandes marcas de hoje foram criadas para trabalhar diretamente com o usuário da Internet, e o que foi criado para comunicação entre universidades tomou a frente de empresas de telecomunicações em todo o mundo.

Um ponto final de motivação é a existência de uma ilha de testes de um projeto europeu (Projeto OFELIA) na cidade de Uberlândia, assim é importante recolher o máximo de informações sobre o que tem funcionado e o que chama a atenção da comunidade científica, tanto no Brasil (pela facilidade maior de comunicação e acesso) quanto no mundo, para que possa ser aplicado aqui, e testado, gerando representatividade para as pesquisas desenvolvidas e para os envolvidos na mesma.

Finalmente é fácil perceber a total ligação deste tema, com o contexto da universidade e com a engenharia de computação. É importante para o profissional se manter atualizado sobre o que ocorre no mundo, sobre as tecnologias que dominam o mercado ou que se desenvolvem para o futuro.

Então toda a pesquisa bibliográfica orientada serve como base para o futuro no mercado de trabalho, agregam conhecimento e um pioneirismo sobre este assunto em específico.

Alguns projetos desenvolvidos no exterior como a ideia do SDN, o protocolo OpenFlow, entre outros, serão trabalhados no documento, além de propostas nacionais como é o caso do RouteFlow, e da ilha do projeto OFELIA em Uberlândia.

Outros projetos de destaque internacional, focados em Internet do futuro, podem ser citados como fonte grande e rica de informações sobre possíveis direcionamentos para a Internet, como os projetos norte-americanos GENI, FIRE, FIA, além de todos os projetos filiados ao FP7, como o 4WARD, ANA, AUTOI, CASAGRAS, COMPAS, OneLab 2, entre outros [14], que desenvolvem partes específicas das possíveis requisições do futuro, o que quer dizer que não necessariamente um se sobrepõe ao outro, mas que informações e resultados podem ser combinados entre estes de forma a criar grupos cada vez mais fortes no desenvolvimento de soluções, sejam voltadas a QoS, ou na própria arquitetura, desenvolvimento de protocolos, e outros pontos interessantes.

# 2 DESENVOLVIMENTO

Para descrever e adentrar ao tema das arquiteturas da Internet do futuro, primeiramente deve ser descrita a ideia base de todas as pesquisas realizadas.

Sabe-se que são muitas as orientações diferentes na busca de novas soluções para a rede mundial de computadores, e em cada parte do mundo vemos ideias diferentes e focos em questões diversas.

Portanto, por isso é necessário direcionar as tecnologias que serão posteriormente citadas, e após algumas análises que envolvem desde a proximidade dos grupos de pesquisa seja ela de relacionamento ou geográfica, até a abrangência da ideia, em termos financeiros e repercussão na comunidade científica, a base para o desenvolvimento de muitas ilhas de desenvolvimento no mundo é a SDN.

Dentre os projetos financiados pelo FP7, muitos tendem a esta ideia, e a comunidade científica, principalmente europeia disponibiliza muitos documentos e sinais da expansão, inclusive comercial da mesma.

Após a descrição da SDN, parte-se para a descrição de alguns projetos baseados nesta, com o foco na arquitetura dos mesmos, para posterior análise e comparação, com destaque para aqueles desenvolvidos no Brasil.

## 2.1 SDN (Software-Defined Networking)

### 2.1.1 Conceito

O conceito de SDN surgiu na universidade de Stanford e nada mais é do que uma nova abordagem para o funcionamento da Internet. Esta nova abordagem envolve desde o controle, o desenho e a construção destas novas redes. O conceito básico deve partir do ponto em que a SDN separa o controle da rede do plano de encaminhamento, de forma a otimizar ao máximo a passagem dos pacotes na rede [15].

Quando se fala nessa separação é bom dar destaque a um problema grande das redes hoje. Os equipamentos de rede tornaram-se verdadeiras “caixas pretas”, ou seja, toda a implementação e software integrados a estas são proprietários [16]. Este é outro fator que traz grandes prejuízos, visto que este engessamento da estrutura atrapalha substancialmente o desenvolvimento de pesquisas das universidades e concentra grande parte do desenvolvimento de tecnologias novas nas mãos de empresas grandes. Temos assim um cenário que direciona ainda mais o desenvolvimento de soluções temporárias que se integram à estrutura complexa já existente.

É neste ponto que entra mais uma proposta do SDN que é o desenvolvimento aberto, e é inegável que o acesso ao código, à substância da aplicação ou software, e do hardware facilitam o compartilhamento de conhecimento e desenvolvimento mais rápido da proposta.

Voltando ao conceito inicial, a separação dos planos de encaminhamento e controle traz a necessidade da existência de um controlador, capaz de visualizar a rede como um todo.

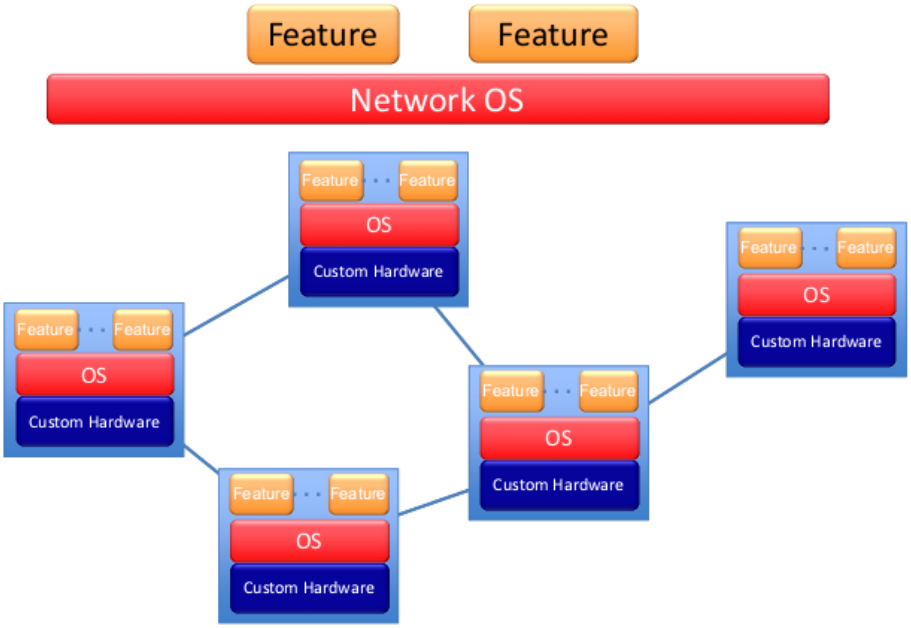


**Figura 9: Esquema básico da SDN [17]**

O Controlador representado na Figura 9 acima dos pontos interconectados faz o papel de um cérebro, ou seja, possui uma visão abstrata e centralizada de toda a estrutura suportada abaixo. Isso quer dizer que através deste é possível comunicar-se diretamente com os roteadores ou switches também representados na figura e tomar assim decisões sobre o comportamento dos mesmos, e, portanto, a forma como será direcionado todo o tráfego desta rede [15].

Isso quer dizer que esta estrutura deve ser alimentada com regras, definidas neste plano de controle, capazes de gerenciar o encaminhamento dos pacotes e as filas nas entradas dos equipamentos. E mais estas políticas podem ser adotadas inclusive conforme o conteúdo destes pacotes, não só pelo fator de urgência, mas pelas requisições que a aplicação que os solicita apresenta [18].

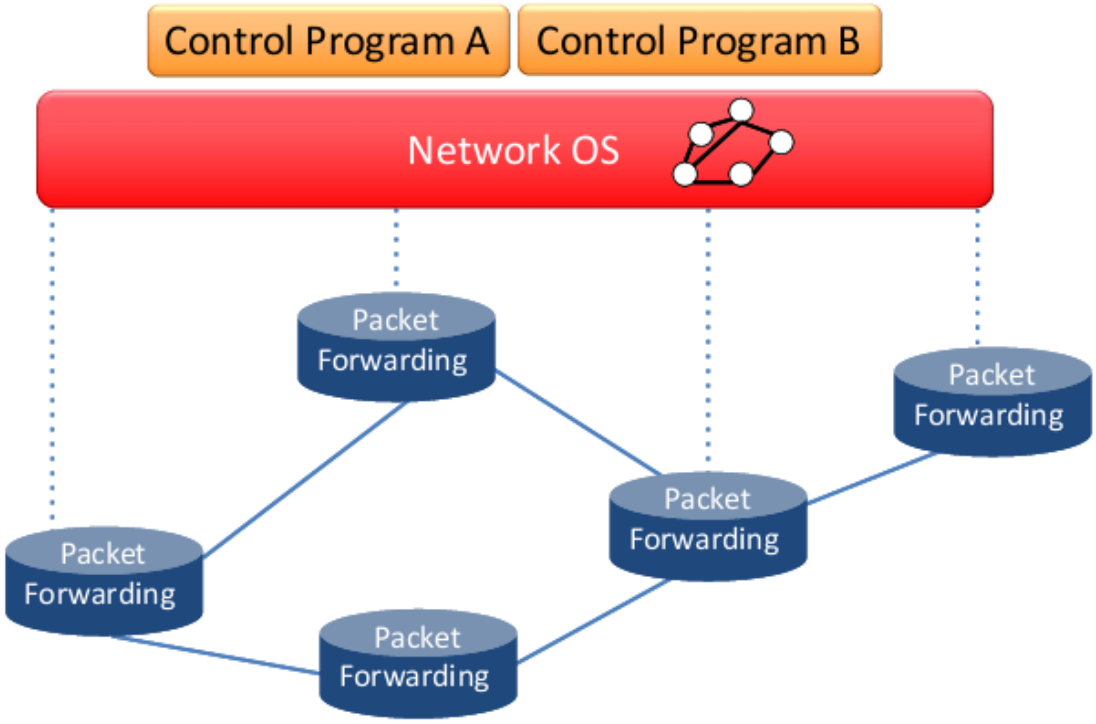


**Figura 10: Ligação entre os planos [17]**

É neste ponto, na comunicação entre os planos, é que se instaura uma grande confusão, entre o significado da SDN e do OpenFlow. O OpenFlow resumidamente é um protocolo adaptável, com campos variáveis e capaz de suportar inclusive os protocolos atuais, que comumente é usado para esta comunicação.

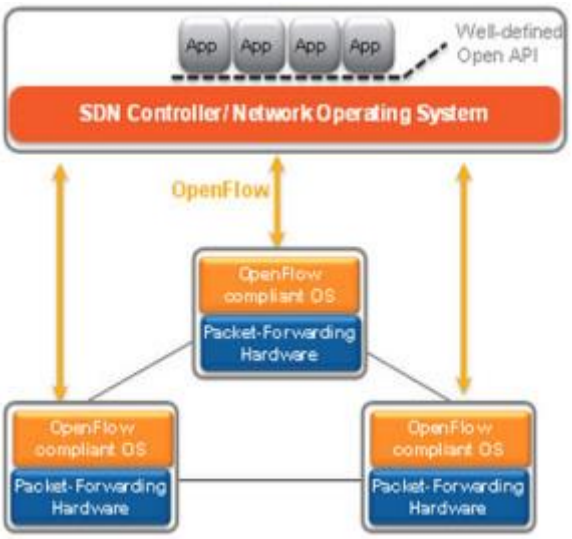


**Figura 11: O OpenFlow na interação dos planos de encaminhamento e controle [15]**

### 2.1.2 Estrutura e Arquitetura

É necessária, portanto, observando a figura 11, alguma estrutura física que exerça este papel de controle, a qual pelo princípio da ideia não pode estar presente individualmente em cada estrutura, mas em uma camada superior, para que fosse capaz de enxergar toda a estrutura abaixo. Na ideia fundamental da proposta este papel pode ser desempenhado por um servidor que suporta todas as regras de encaminhamento, além de políticas de prioridade e QoS sobre os dados e estruturas da rede.

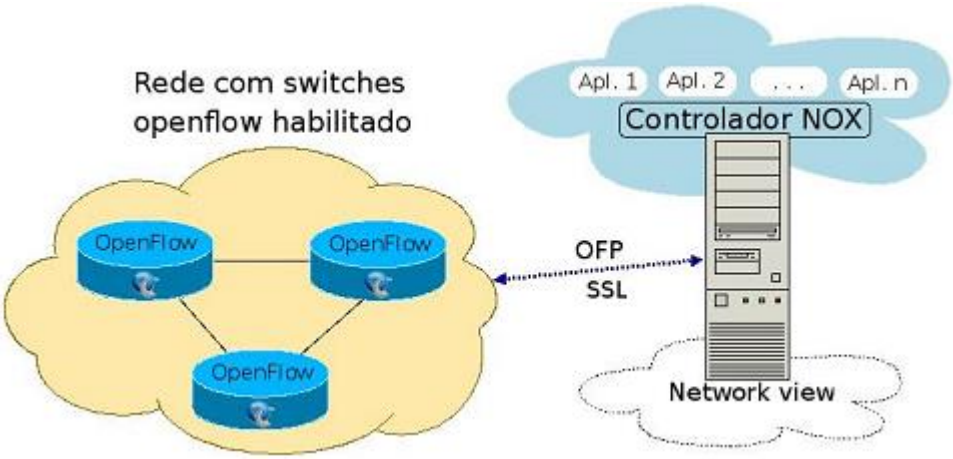


**Figura 12: Controlador sobre uma rede OpenFlow [16]**

A figura 12 mostra uma estrutura de switches com OpenFlow habilitado, mas esta estrutura é relativamente genérica e mostra uma boa visualização de como a estrutura funciona [19]. Existem abordagens para esta estrutura que colocam três camadas, onde a primeira seriam as aplicações com as regras de negócio, a segunda representaria o sistema operacional da rede, isso quer dizer que estas duas camadas são conectadas por APIs, o que lembra o funcionamento de um sistema operacional de um computador pessoal, e por fim a última camada representa toda a infraestrutura de roteamento, o que representa bem o fluxo da informação na SDN, onde a política do sistema interage com um controlador que aplica as regras desta política aos equipamentos abaixo. Para que estas três camadas funcionem bem são três pontos fundamentais [20]:

1. A separação dos dados do plano de controle do plano de encaminhamento, com o protocolo OpenFlow como interface;

2. Uma interface bem definida com o sistema operacional da rede;
3. Virtualização da rede.

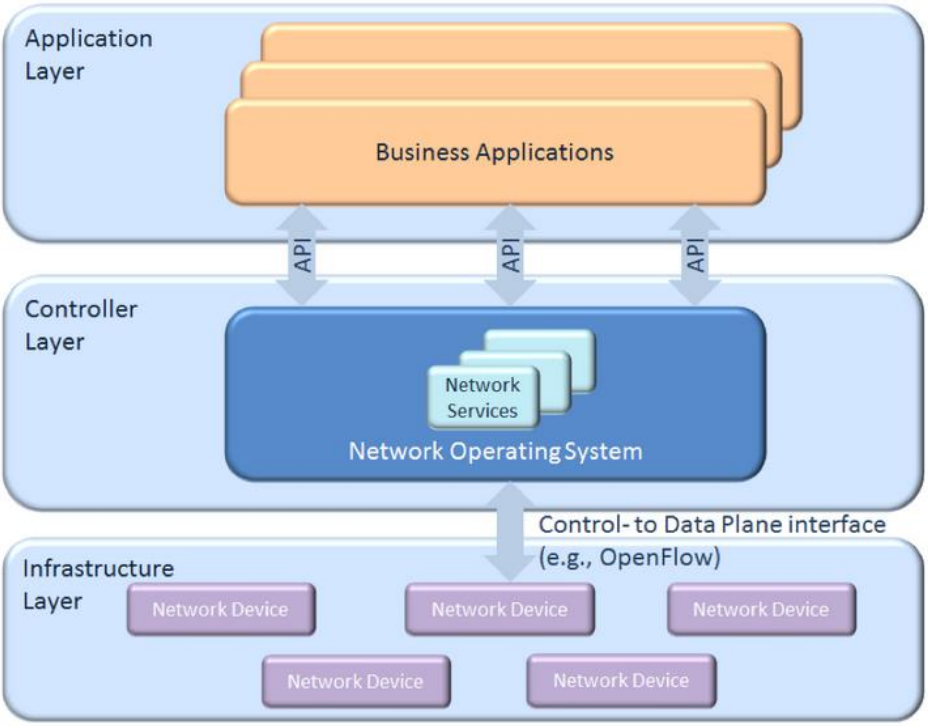


**Figura 13: As camadas da SDN de forma simplificada [20]**

A palavra chave para o funcionamento de toda esta arquitetura é a virtualização. Todos os switches, isso quer dizer, toda a estrutura de encaminhamento é representada virtualmente no controlador, isso quer dizer que ele tem uma visão completa e funcional de como interconectar clientes e servidores [21].

Isso traz também o conceito de *Flow-Tables,* que nada mais são do que tabelas que representam todo o fluxo de dados na rede, e que estão sob o "domínio" do controlador, o que dá a este uma autonomia muito grande de decisão sobre como otimizar e atender aos recursos de toda uma estrutura de rede. Isto traz também a ideia de níveis de abstração, onde primeiramente é abstraído o fluxo comum entre os tipos de redes e que provê um paradigma comum para o controle, o que dá todo o sentido para as tabelas de fluxo. O segundo nível de abstração é uma mapa comum, anteriormente citado como uma visão de toda a estrutura física de switches e roteadores que permite a criação de aplicações extensível sobre múltiplas camadas [22].

Com todos estes conceitos a estrutura fica mais completa e já apresenta detalhes que deixam a visualização da mesma com funcionalidade bem definida, como ilustra a figura 14.

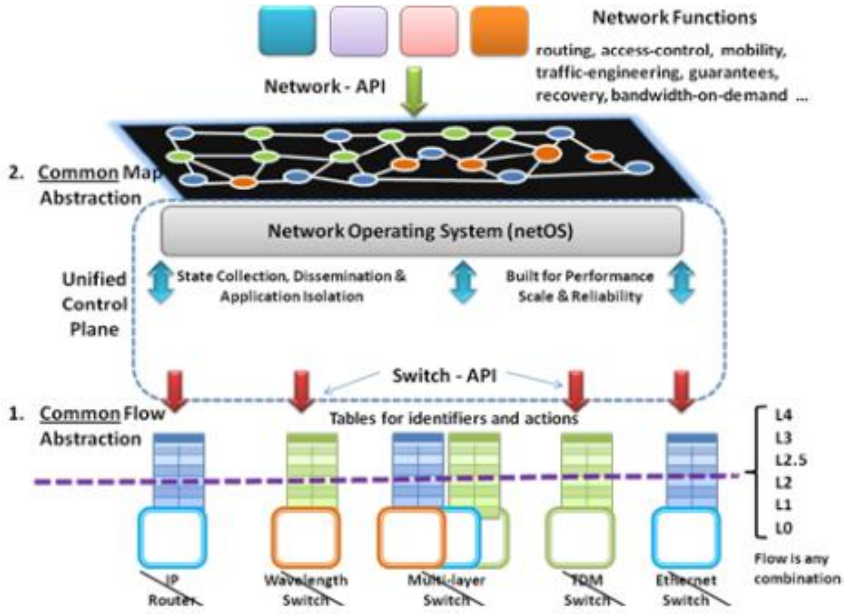


**Figura 14: Estrutura completa da SDN [22]**

Em um fluxo real em que um *host* solicita comunicação com outro, primeiramente seria necessário uma espécie de registro ou identificação de ambos junto ao controlador, depois de acordo com o tipo de fluxo, o controlador, com sua visão sobre a estrutura, age de forma direta sobre cada switch pelo caminho determinado de acordo com as regras previamente definidas para aquele tipo de comunicação (estas regras estão diretamente ligadas ao que foi definido anteriormente como camada de aplicação), assim a rota, ou circuito, entre os pontos é alocada e as regras de encaminhamento dos pacotes deste fluxo são transferidas aos switches, que de acordo com uma tabela interna é capaz de destinar corretamente os pacotes [21].

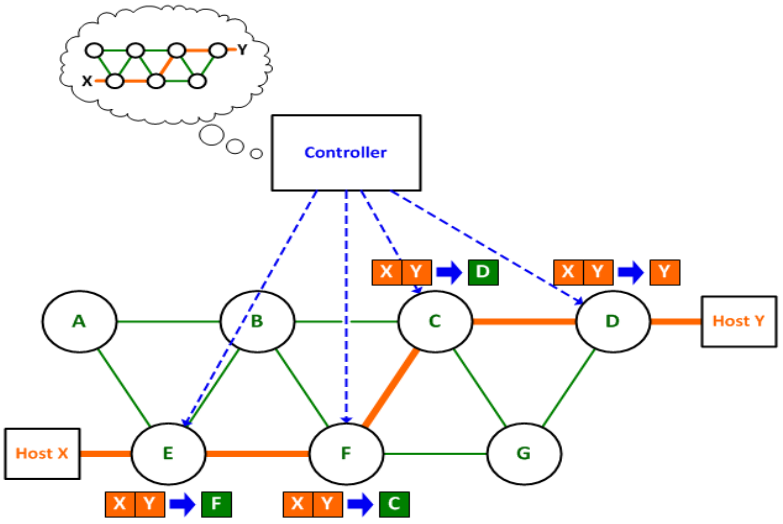


**Figura 15: Circuito ou rota reservada para um fluxo específico [21]**

### 2.1.3 Regras de encaminhamento e Ferramentas

Outro ponto importante para a estrutura é a definição das regras para o tratamento do encaminhamento e das filas de pacotes, os quais representam hoje o trabalho das camadas de enlace e rede. Estas regras podem ser dividas de várias formas, mas quatro grandes pontos podem resumi-las de forma bem objetiva [21][23].

O primeiro é a divisão em níveis de prioridade, no qual podem ser usadas informações de cabeçalhos, que sejam de acordo com endereço ou flags (bits que indicam estado de “sim” ou “não”), informações sobre a aplicação e suas requisições de QoS, entre outras [23].

O segundo ponto pode ser chamado de desempacotamento dinâmico, isto quer dizer que as regras podem ser geradas dinamicamente de acordo com os diferentes tipos de fluxos que apareçam e propriedades instantâneas do sistema, o que dá uma capacidade adaptativa de reação para todo o sistema. Este é um ponto inovador e que traz teoricamente uma capacidade muito grande de ganhos em QoS [23].

O terceiro ponto é o mais simples e diz sobre a limitação do tráfego, ou seja, apenas avitar rotas ou circuitos que já estejam com capacidade esgotada, ou que não tenham capacidade de banda e recursos suficientes para um fluxo específico [23].

O último refere-se a estatísticas que podem envolver uma série de medidas e monitoramento do sistema de controle, as quais podem trazer previsões de acordo com certos tipos de horários e aplicações de forma que o sistema não seja apenas adaptativo, mas possa prever certas situações e gerenciar fluxos conforme estas.

Finalmente um paralelo entre as redes legadas e as camadas da SDN é facilmente construído, onde as camadas da SDN podem ser colocadas em paralelo com a estrutura atual [24].

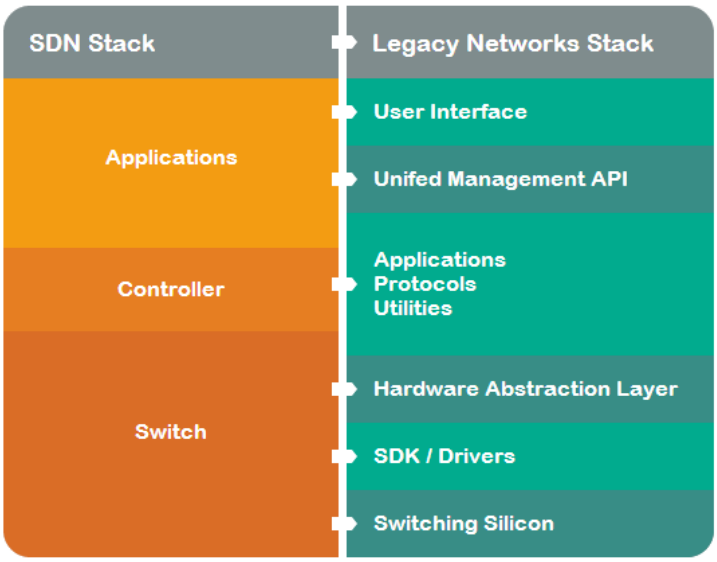


**Figura 16: Paralelo entre as camadas SDN com a pilha das redes legadas [24]**

Alguns ambientes de testes e criação de estruturas virtuais para o desenvolvimento de novas tecnologias sobre a SDN foram criados e podem ser livremente utilizados. Um dos que se destaca é o Mininet, um simulador baseado no protocolo OpenFlow que disponibiliza uma série de opções para criação de switches, conexões hosts e controladores, onde as regras podem ser dinamicamente criadas pelo sistema ou introduzidas pelo próprio usuário [25].

## 2.2 OpenFlow

O OpenFlow é considerado o primeiro protótipo da SDN, e a sua função está diretamente ligada à comunicação entre o controlador e os elementos de rede. Isso que dizer que é este o protocolo responsável por passar as informações de ajuste da rede para os elementos. É importante destacar que para que esta comunicação ocorra ele deve ser previamente adaptado em cada elemento (switch ou roteador), e grandes fabricantes já oferecem seus produtos com este tipo de tecnologia [26][27][28].

Foi proposto pela Universidade de Stanford com o intuito de atender à demanda criada pelos novos conceitos de redes que surgiram com o tempo, isto quer dizer que já estava prevista toda a estrutura com a qual este protocolo iria interagir [16].

Em suma o OpenFlow pode ser resumido como uma funcionalidade adicionada a pontos de acesso (APs) e estações, que permite que estes sejam controlados por uma API externa [29].

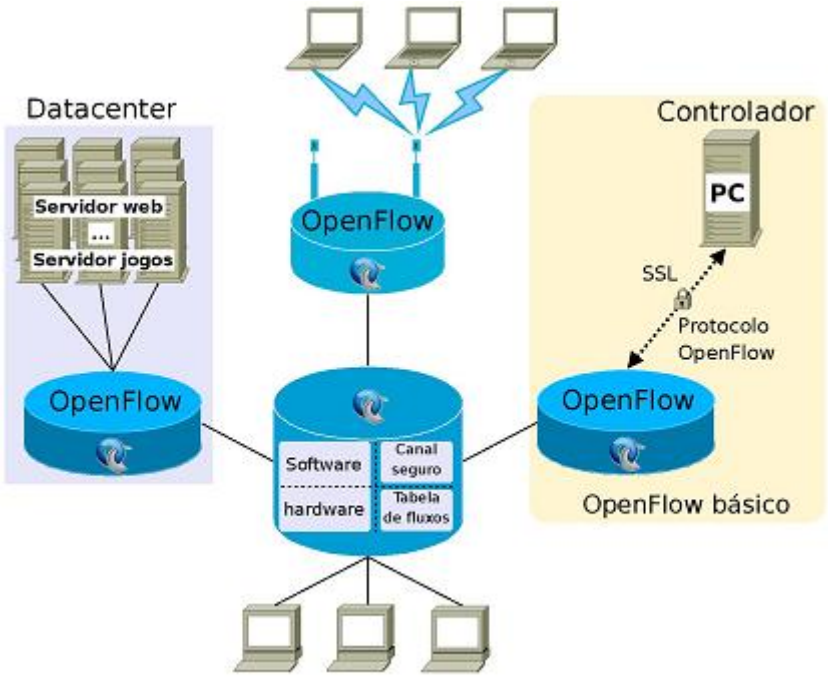


**Figura 17: O OpenFlow em uma estrutura de rede [16]**

Para seu funcionamento o OpenFlow está ligado a uma abstração imprescindível, que é o conceito de fluxo, já citado anteriormente. Os fluxos são criados por combinações de informações dos pacotes que caracterizam algum padrão de encaminhamento [16]. Isto quer dizer que os fluxos podem determinar se os pacotes serão tratados de acordo com cabeçalhos específicos dos mesmos, desde algumas características como a fonte ou destino da informação, até tipo ou padrões de QoS.

Pode-se dizer então que este protocolo realiza uma generalização do plano de dados, isto quer dizer que existe suporte para qualquer modelo de encaminhamento de dados baseado em combinação de valores de cabeçalhos [16].

| inport | Ethernet | | | VLAN | | IP | | | | TCP/UDP | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | src | dst | type | id | pri | src | dst | proto | ToS | src | dst |

**Figura 18: Trecho do cabeçalho do OpenFlow [16]**

O trecho do cabeçalho do OpenFlow mostrado acima dá uma visão de suporte aos outros protocolos, isto demonstra que informações que já são guardadas pelos mesmos podem ser usadas nesta nova tecnologia, o que ratifica a adaptabilidade desta arquitetura.

A existência destas regras definidas para o encaminhamento e os possíveis destinos de cada fluxo gera um conceito fundamental na arquitetura SDN que são as *flow-tables*. São bem similares às tabelas de encaminhamento existentes hoje e estão sob o domínio do controlador, passadas aos roteadores e switches e alteradas conforme a necessidade [26].

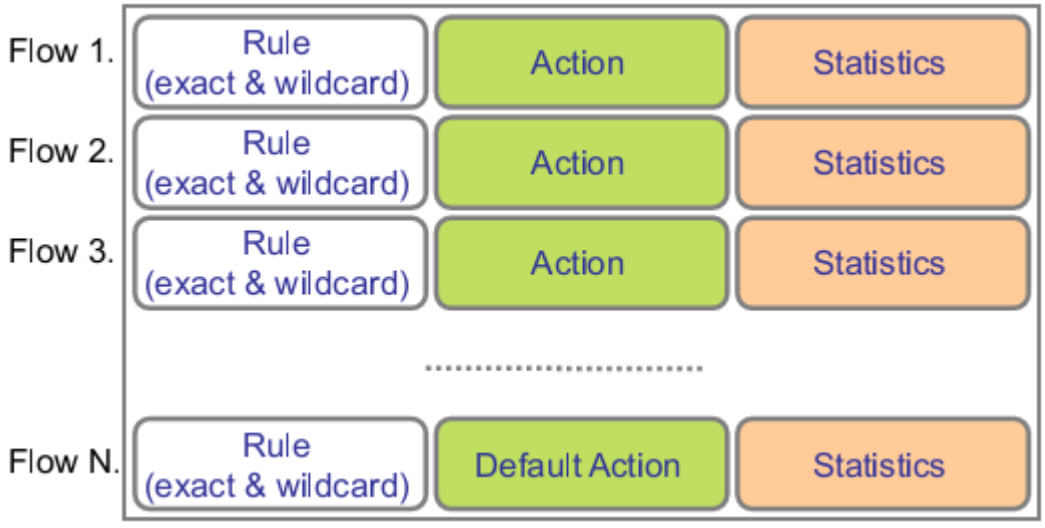


**Figura 19: Uma visão da tabela de fluxos (*flow-table*) [19]**

Com todas estas informações pode-se dizer que o OpenFlow é dividido em três partes maiores, conforme a figura 20, sendo que a primeira destas é o switch, incapaz de tomar as decisões sozinho, ele recebe do controlador as *flow tables* para realizar o encaminhamento. A segunda parte é o canal seguro entre o controlador e estes switches, e neste ponto percebemos a importância das informações trocadas entre estes componentes, que torna necessária a proteção total deste canal de comunicação. Por último o próprio protocolo que faz a função final de comunicação entre as partes.

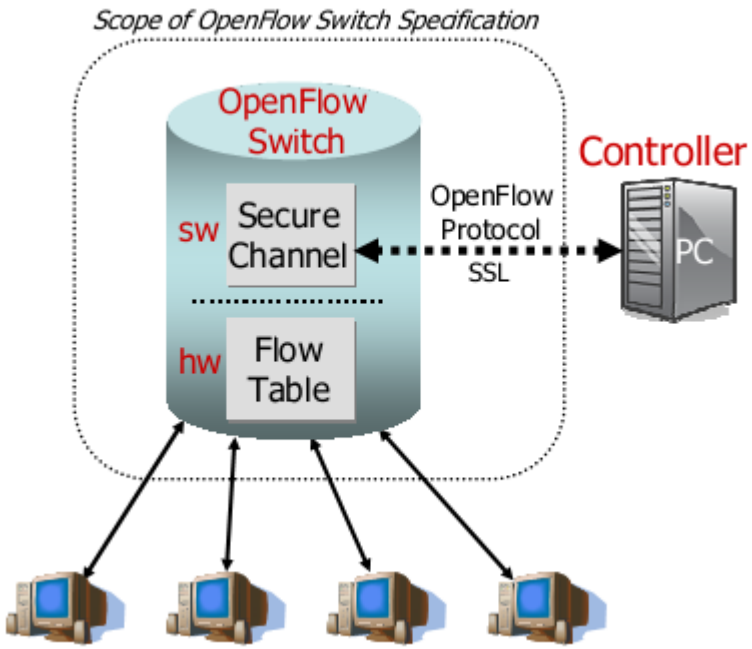


**Figura 20: As partes do OpenFlow [30]**

Com estas características uma gama de possibilidades pode ser construída com o OpenFlow, não só na questão de gerenciamento e controle de acesso, mas VLANs (Virtual Local Area Network) pela sua capacidade de destinar fluxos, criação de um serviço VoIP (Voice Over Internet Protocol) com mobilidade, graças as regras que o controlador pode definir e portanto realizar mecanismos de *handoff* (mudança de ponto de acesso) dinamicamente, dar prioridade a pacotes dentro de fluxos específicos e o mais interessante, tornar funcional uma rede que não use o protocolo IP, ou seja, podem ser construídos outros protocolos e combinações de cabeçalhos capazes de definir a localização de um dispositivo ou usuário na rede [30].

Finalmente é importante destacar que o OpenFlow possui os seus próprios campos, além de várias versões, onde novas funcionalidades foram adicionadas. A tabela 1 pode mostrar os *match fields* (campos do cabeçalho) da versão 1.0 do protocolo (primeira versão), onde a

quantidade de bits é fixa para cada entrada da *flow-table*, diferente do que já acontece nas últimas versões que possuem o que é chamado de OXM (OpenFlow eXtensible Match), onde é possível determinar o tamanho, o tipo e valor de cada *flow entry,* o que traz uma amplificação de possibilidades no tratamento dos pacotes [31].

**Tabela 1: Campos do cabeçalho da versão 1.0 do protocolo OpenFlow [31]**

| *Match field* | Descrição |
|---|---|
| Ingress Port | representação numérica da porta de entrada, começando em 1. |
| Ethernet source address | endereço ethernet de origem. |
| Ethernet destination address | endereço ethernet de destino. |
| Ethernet type | o tipo ethernet do quadro (ethernet frame type). |
| VLAN id | id VLAN |
| VLAN priority | prioridade VLAN. |
| IP source address | endereço IP de origem. |
| IP destination address | endereço IP de destino. |
| IP protocolo | são utilizados apenas os menores 8 bits do ARP. |
| IP ToS bits | especifica como um valor de 8 bits e coloca o ToS nos 6 bits maiores. |
| Transport Source Port/ ICMP Type | os menores 8 bits utilizados para o tipo ICMP. |
| Transport Destination Port/ ICMP Code | os menores 8 bits utilizados para o código ICMP |

## 2.3 FlowVisor

Além do conceito de fluxo, uma nova perspectiva pode ser apresentada para as redes do futuro, conhecida como *slicing,* que seria um fatiamento da rede. A princípio podem existir algumas confusões com relação aos fluxos, mas o *slicing* apresenta uma abrangência maior na rede. Primeiramente este conceito está diretamente ligado à virtualização. É na perspectiva do controlador, nos seus modelos e desenhos da estrutura da rede que as fatias (*slices)* fazem sentido. Diretamente, para cada regra ou política definida é possível extrair modelos virtualizados diferentes da mesma rede, ou seja, cada fatia é uma estrutura virtual diferente adaptada a requisitos específicos, baseada em uma mesma disposição física de dispositivos [32].

A figura 21 mostra claramente uma estrutura física e diferentes abstrações desta, sendo que estas são adaptadas para fluxos de dados diferentes.

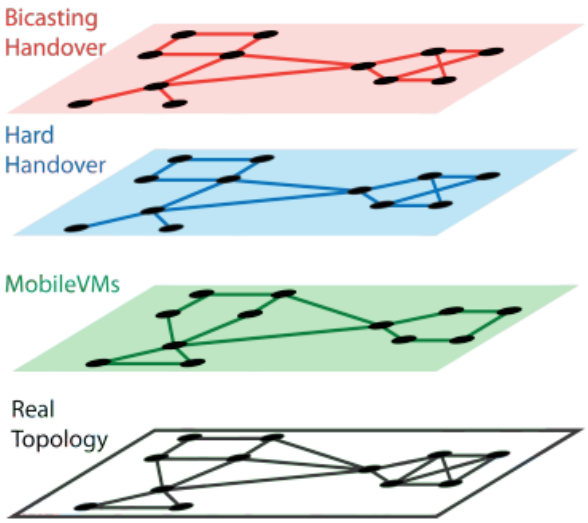


**Figura 21: Uma topologia real e diferentes abstrações virtuais [32]**

Para realizar esta função em conjunto com o OpenFlow foi desenvolvido o FlowVisor que é o "fatiador" da rede para permitir múltiplas funcionalidades compartilharem da mesma estrutura física. Isto quer dizer que uma organização, por exemplo, pode possui sua própria fatia, onde apenas caminhos desejados (por motivo de segurança ou desempenho) passam ao modelo virtualizado, assim todos os pacotes marcados como pertencentes a esta organização, trafegam em uma estrutura determinada [33].

A figura 22 faz um paralelo entre a função do FlowVisor com um sistema computacional com seus recursos de *hardware*, na qual a parte acima do FlowVisor representa o controlador (3 exemplos diferentes) similar ao sistema operacional de um computador.

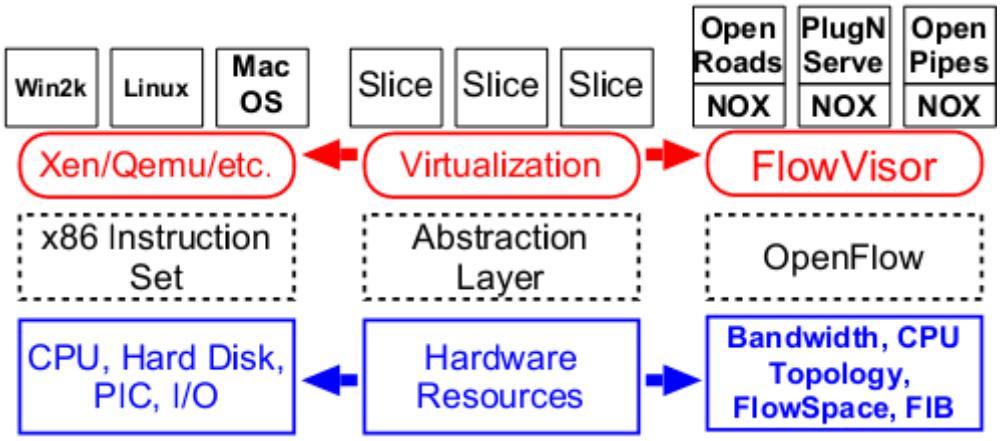


**Figura 22: A posição do FlowVisor ao lado de um sistema computacional [34]**

O FlowVisor atua em cinco dimensões determinadas para cada fatia. Isto quer dizer que estas são fatores determinantes para os recursos disponíveis em cada uma, já que o ambiente é compartilhado, e, portanto, deve haver limitações. A primeira dimensão é a largura de banda, ou seja, os enlaces comuns em determinadas fatias devem ser divididos conforme necessidade. A segunda dimensão é própria topologia, isso quer dizer que cada abstração pode selecionar os elementos e enlaces que serão virtualizados. A terceira é o trafego, que pode ser associado a uma ou mais fatias, ou completamente isolado para uma fatia específica. A quarta dimensão são os recursos computacionais dos elementos (switches e roteadores), que são limitados e devem ser considerados. Por último as próprias tabelas de repasse e encaminhamento, ou tabelas de fluxo, isto quer dizer que um mesmo switch pode dar vários destinos a um mesmo tipo de fluxo de acordo com a fatia [34].

No caso de vários controladores diferentes, podem ser criados vários módulos do FlowVisor, que interagem entre si, além de que, pode ser criada uma hierarquia destes módulos, sendo que cada um destes pode ter uma atuação limitada ou específica (sobre certos switches por exemplo) [34].

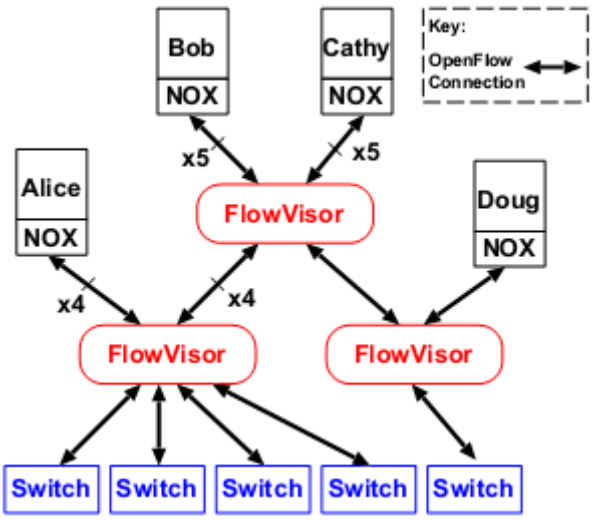


**Figura 23: Vários módulos de FlowVisor em uma mesma rede [34]**

De forma simplificada, o FlowVisor age como um proxy transparente. Mensagens OpenFlow são passadas dos dispositivos da rede para este proxy, que garante o isolamento entre as redes virtuais (fatias) encaminhando os pacotes para o controlador apropriado, sendo que o FlowVisor em si não assume nenhuma atitude na relação entre o plano de controle com o plano de encaminhamento quando não são trocadas mensagens OpenFlow. De forma mais simples, o FlowVisor atua como uma controlador virtual na perspectiva dos switches, e atua como switches

virtuais na perspectiva dos controladores. Por esse motivo ele tem uma visão de todo o estado da rede, o que permite descarte ou escrita de mensagens OpenFlow de controle, um rearranjo dos subconjuntos de switches delegados a cada fatia, além de um desacoplamento do controle, evitando interferências e conflitos entre os controladores.

É interessante destacar que é fundamental que o FlowVisor conheça as regras de cada controlador ao qual ele está conectado, para que ele seja capaz de desempenhar seus três objetivos principais: transparência, isolamento e extensibilidade na definição de cada fatia, visto que é importante a flexibilidade da ferramenta para ajustar os recursos de forma adaptável.

Serve como base para fundamentar o trabalho, devendo incluir toda a informação pertinente ao tema [34].

Pelo que já foi citado acima é importante relembrar e destacar que o FlowVisor só é capaz de atuar em switches OpenFlow (elemento “encaminhador”) pela necessidade da comunicação entre os controladores e switches através deste protocolo [35].

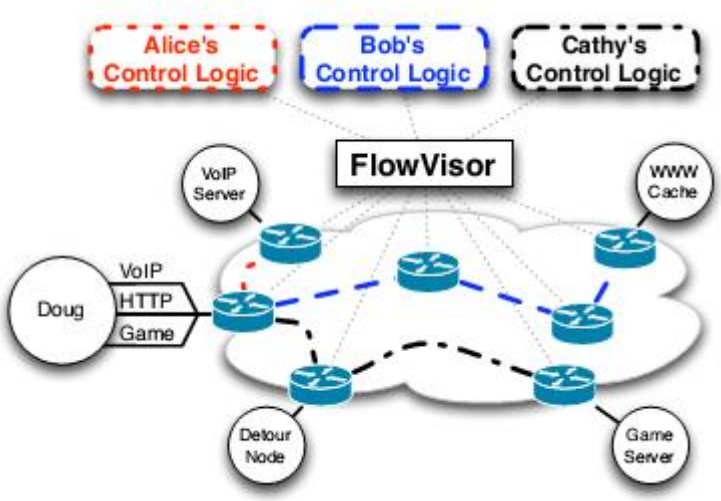


**Figura 24: O FlowVisor em uma visão geral da arquitetura [35]**

## 2.4 Controladores

### 2.4.1 Descrição Geral

A função do controlador em uma arquitetura como a proposta pela SDN é fundamental e de grande destaque, pois representa toda a “inteligência” da rede.

A responsabilidade deste componente gira em torno de manter todas as regras da rede e por fim distribuí-las para que os elementos de encaminhamento sejam capazes de tomar as decisões

baseadas em uma análise e em certos números (estatísticas da rede e requisições de QoS), diferente do que acontece na arquitetura TCP/IP [36].

Usando os conceitos já explanados, pode-se dizer, portanto, que o controlador é responsável por como a rede irá encaminhar pacotes através de entradas de fluxo válidas em uma tabela (*Flow-table*), e de todo o gerenciamento destas entradas. Especificamente nesta arquitetura é necessário o canal seguro e o protocolo OpenFlow.

Tipicamente este controlador executa em um servidor acoplado à rede, que possui diferentes configurações de acordo com alguns fatores.

Um destes é a localização, já que em uma rede podem existir vários controladores e também uma espécie de delegação de estrutura para cada um em específico [36].

Outro fator é o próprio fluxo, que é individualmente criado pelo controlador, é importante lembrar que cada fluxo corresponde a uma entrada na tabela. E mais, pode acontecer uma agregação de fluxos conforme a necessidade, em categorias, por exemplo, o que é importante para regiões onde existe alto fluxo de informação, como um *backbone* [36].

Um último fator de grande destaque é o comportamento deste controlador, isto quer dizer que as ações deste são regradas por algum tipo de padrão. Um exemplo são os controladores reativos ou proativos, isto quer dizer, podem criar fluxos de acordo com certas situações, ou simplesmente definir previamente a tabela de fluxos para cada roteador ou switch [36].

São muitas as implementações de controladores, os quais podem ser direcionados a estruturas específicas, o que traz vantagens e desvantagens para cada conjunto de características de uma rede.

**Tabela 2: Alguns controladores desenvolvidos e algumas características básicas [36]**

| Controller | Main characteristic |
|---|---|
| NOX | Open-source OpenFlow controller that provide a simplified platform for writing network control software in C++ or Python |
| Beacon | Beacon is a fast, cross-platform, modular, Java-based controller that supports both event-based and threaded operation. |
| Trema | Full-Stack OpenFlow Framework for Ruby/C |
| Maestro | A scalable control platform written in Java which supports OpenFlow switches |
| SNAC | SNAC is an OpenFlow controller, which uses a web-based policy manager to manage the network. |

### 2.4.2 Desenvolvendo um Controlador

De forma básica pode ser estruturado em passos o desenvolvimento de um controlador para uma rede específica. Os controladores nada mais são do que aplicações, similares a um Sistema Operacional em sua essência, com o dever de monitorar e atuar diretamente no modelo de encaminhamento da rede, mas que também podem ser divididas em módulos em algumas ocasiões.

Uma divisão em etapas simples pode ser colocada da seguinte forma [36]:

- Primeiramente é definido o objetivo da aplicação, que pode ter níveis de complexidade diversos, desde redirecionar pacotes ICMP (Internet Control Message Protocol) de ping até monitorar fluxos e tomar certas atitudes para modificar o fluxo.
- O desenho das decisões do controlador é um segundo passo, que representa a definição dos modos de operação do controlador, e é neste ponto que importante separar características fundamentais deste, que pode ser centralizado ou distribuído, pode ser reativo ou proativo. Cada característica direciona os fluxos de forma diferente e apresenta suas limitações.
- O terceiro ponto é a definição de um cenário. Isto é fundamental para o prosseguimento da ideia. Este cenário deve apresentar características específicas para o modelo do controlador desenvolvido, e pode avançar em complexidade conforme o projeto também avança. Todos os outros passos restantes da fase de testes podem ser acoplados a este, isto quer dizer, a ferramenta disponível para a construção deste cenário (como é o caso do Mininet citado anteriormente como software para simulação), as requisições da simulação, o teste em sí, e a definição de um diagrama de fluxos. A figura 25 mostra um cenário simples que pode ser construído usando a ferramenta Mininet, onde é possível determinar as características do controlador, criar conexões, manipular as informações de cada teste realizado e principalmente editar e criar novos paradigmas, protocolos, cabeçalhos, entre outros, que são parte fundamental no desenvolvimento de tecnologias novas. Esta fase pode sair do mundo virtual a partir para simulações reais em ambiente de homologação e testes.
- O último ponto, muitas vezes não alcançado, é o cenário real e a implementação do controlador em um ambiente real. Esta fase exige recursos como equipamentos de encaminhamento, enlaces e servidores capazes de atender a demanda da rede utilizada.

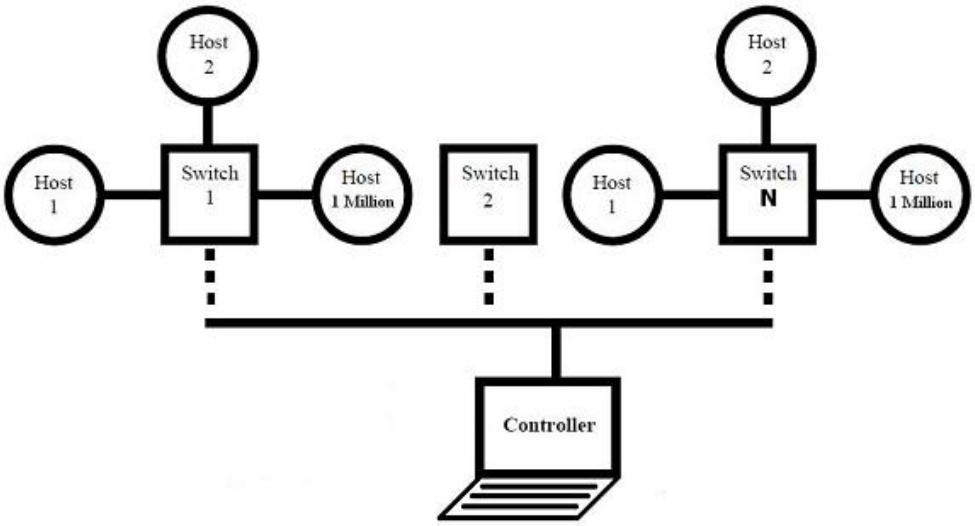


**Figura 25: Uma estrutura básica que pode ser construída no Mininet [36]**

Dentre os controladores já desenvolvidos, dois serão destacados a seguir, o NOX e o beacon, principalmente pela sua abrangência e desempenho, principalmente o NOX utilizado em uma série de projetos de arquiteturas.

### 2.4.3 NOX

O NOX é um dos controladores criados para a arquitetura SDN e vê-se um grande destaque do mesmo na comunidade científica. Ele se encaixa em um conceito definido como NOS (Network Operating System), justamente por ser uma aplicação que tem como objetivo administrar toda uma rede.

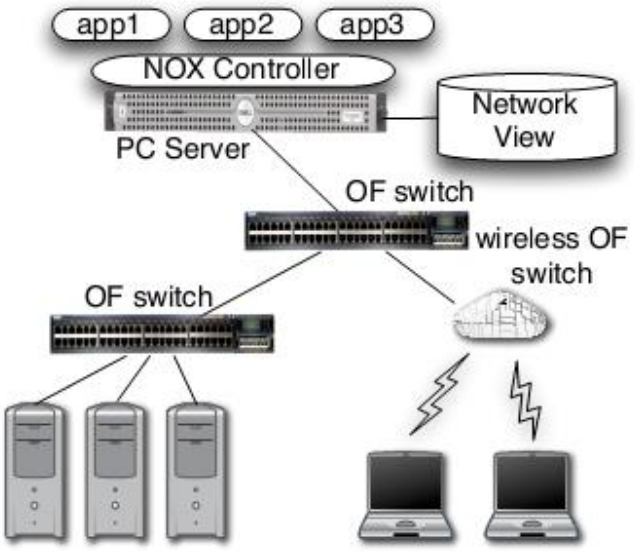


**Figura 26: Componentes de uma rede baseada em NOX [37]**

Através da figura 26 vemos o NOX em uma posição que os esquemas da SDN indicavam claramente, com a visão de toda a rede, e com a interação direta com os elementos do plano responsável pelo encaminhamento.

As funções do NOX e algumas características da estrutura podem ser descritas em grandes esferas que englobam objetivos específicos [37]:

- Componentes: A figura 26 já mostra de forma simplificada os componentes da estrutura na qual o NOX atua. A aplicação do NOX executa nos PC Servers indicados, os quais estão interligados a bancos específicos que armazenam a visualização da rede obtida através da virtualização da arquitetura. Finalmente o protocolo OpenFlow permite a troca de mensagens de controle e a própria virtualização dos elementos de rede.
- Granularidade: este é um ponto importante do controlador já que afeta diretamente a escalabilidade e a flexibilidade do sistema. Estes são fatores importantes quando é levada em consideração a dimensão que as redes de computadores podem atingir. Isso quer dizer que podem existir várias aplicações, ou subdivisões destas interagindo com partes específicas da rede. Para destacar, a visualização que o NOX tem de rede inclui a topologia, a localização dos usuários, hosts, outros elementos, e até os serviços que podem ser oferecidos (importantes para o QoS). Sobre esta é importante relembrar que a granularidade é baseada em fluxos neste caso, e, portanto, a abstração provê atuações em um certo padrão para certas características dos pacotes.
- Abstração dos switches: esta abstração é baseada no modelo no OpenFlow, isto quer dizer que os switches são virtualizados e representados por tabelas de fluxos, no modelo das entradas com a regra (combinação de cabeçalhos), as ações e os contadores e estatísticas para cada fluxo.
- Operação: o tratamento dado para cada fluxo, as ações e regras relacionadas são gravadas na *flow table,* mas eventualmente pacotes que não se encaixam em nenhum fluxo podem aparecer, sendo que estes são direcionados ao controlador, e assim é definido o modo de operação que deve construir uma observação da rede (visualização do elementos e tráfego), e determinar qual será a ação tomada para este pacote, ou se um novo fluxo deve ser criado.
- Escalabilidade: é importante definir tempos e recursos limites para o funcionamento do controlador, isto quer dizer que processos como a chegada de pacotes e definição de fluxos devem ser dinâmicos e rápidos.

- Interface de programação: todo o processo de programação está resolvido entre eventos, espaço de nomes e uma visualização da rede. Isto pode ser usado para dar funcionalidades de resolução de nomes, autenticação, entre outras para o controlador, e a forma como devem ser respondidas as sinalizações da rede. Isto é fundamental no processo de controle e na fundamentação das funcionalidades e regras presentes em uma aplicação específica do NOX. A figura 27 é apenas para ilustrar a capacidade de se programar dentro do controlador, com uma linguagem conhecida, no caso, Python.

```
# On user authentication, statically setup VLAN tagging
# rules at the user's first hop switch
def setup_user_vlan(dp, user, port, host):
    vlanid = user_to_vlan_function(user)
    # For packets from the user, add a VLAN tag
    attr_out[IN_PORT] = port
    attr_out[DL_SRC] = nox.reverse_resolve(host).mac
    action_out = [(nox.OUTPUT, (0, nox.FLOOD)),
                  (nox.ADD_VLAN, (vlanid))]
    install_datapath_flow(dp, attr_out, action_out)
    # For packets to the user with the VLAN tag, remove it
    attr_in[DL_DST] = nox.reverse_resolve(host).mac
    attr_in[DL_VLAN] = vlanid
    action_in = [(nox.OUTPUT, (0, nox.FLOOD)),
                 (nox.DEL_VLAN)]
    install_datapath_flow(dp, attr_in, action_in)
nox.register_for_user_authentication(setup_user_vlan)
```

**Figura 27: Uma aplicação NOX para regras de VLANs implementada em Python [37]**

### 2.4.4 Beacon

Assim como o NOX, o Beacon é um controlador desenvolvido para lidar com o OpenFlow. Este surge com os princípios de buscar produtividade e desempenho. Uma questão importante a se destacar é que o NOX foi o primeiro controlador que surgiu para trabalhar nesta arquitetura, e assim, serviu como base de pesquisa e estudo para montar o Beacon.

Os princípios de funcionamento deste controlador são praticamente os mesmos vistos para o NOX, apresentando módulos de topologia, gerenciamento de dispositivos e roteamento, sobre uma base similar ao *Kernel* de um sistema operacional, designada como *Core* na figura 28, e com um destaque, esta base dá condições para desenvolvimento de novos módulos, com o objetivo de estender as funcionalidades deste controlador, e esta visibilidade do sistema traz novamente a semelhança com um sistema operacional de um computador.

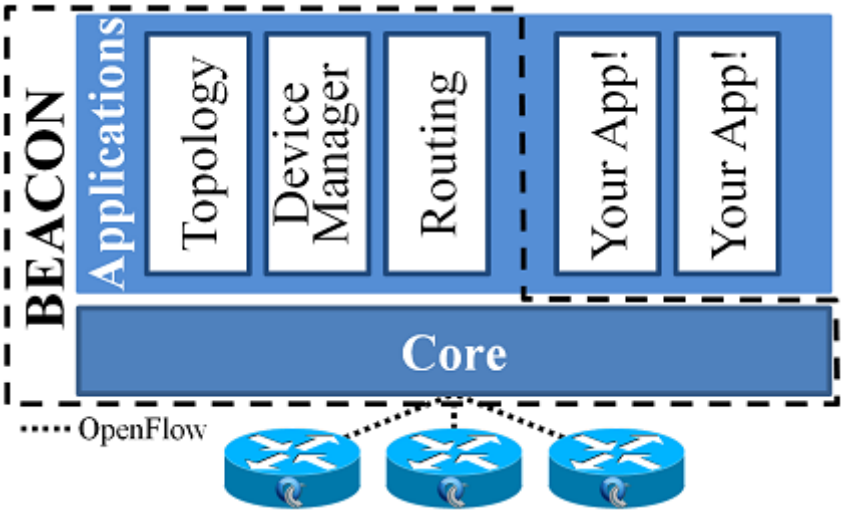


**Figura 28: Visão geral do Beacon [38]**

O funcionamento deste sistema está baseado no tratamento de eventos com a utilização de *threads* em duas versões, a primeira chamada de *Shared Queue* onde há dois conjuntos de threads, o primeiro conjunto lê e trata as mensagens OpenFlow que são colocadas em uma fila compartilhada. Cada switch está associado a uma thread, sendo que vários switches podem estar associados à mesma thread. Estas threads chamadas de I/O (*Input/Output*) são responsáveis também por escrever mensagens OpenFlow em direção aos switches. O segundo conjunto de threads pega as mensagens na fila e as processa em um *pipeline,* um tubo de aplicações, que determinam as regras e decisões para tais mensagens [38].

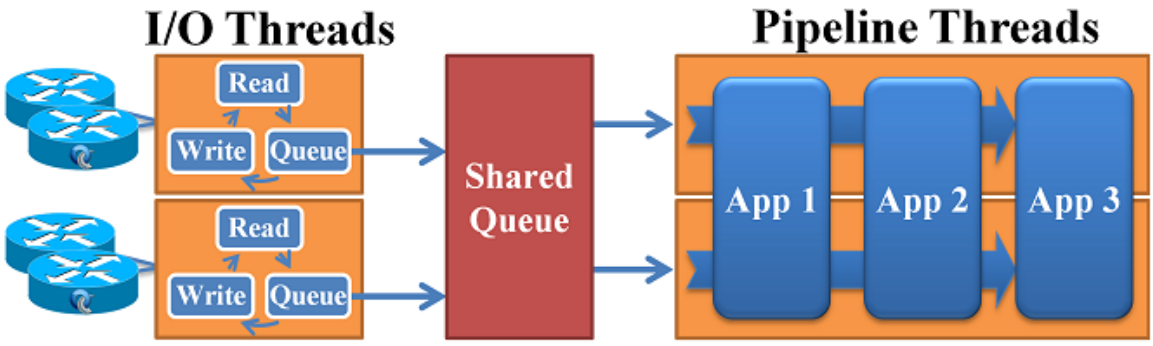


**Figura 29: O modelo da versão *Shared Queue* [38]**

O segundo modelo é o *Run-to-completion,* que é uma versão simplificada do primeiro, pois apenas possui um conjunto de threads I/O, que já direciona as mensagens para o *pipeline* de aplicações, isso quer dizer que não há uma divisão de tarefas entre as threads.

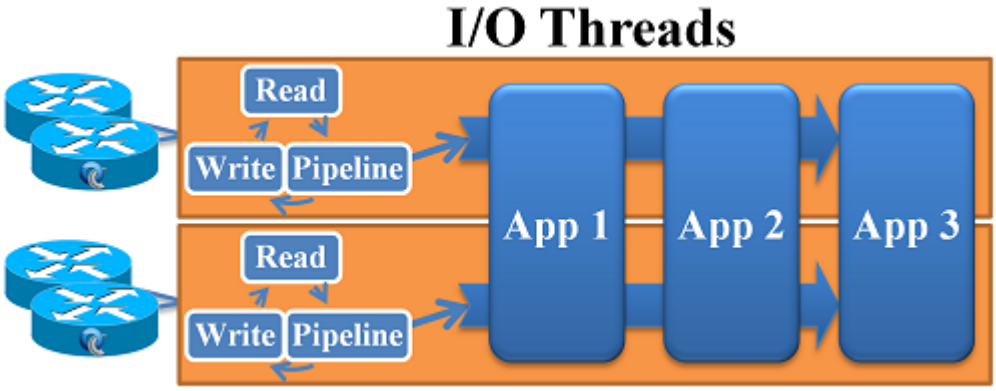


**Figura 30: O modelo da versão *Run-to-completion* [38]**

Esta é uma pequena descrição do funcionamento deste controlador desenhado para trabalhar com as linguagens C e C++, pelo desempenho que estas oferecem. Para atestar este desempenho a figura 31 mostra a comparação deste com outros controladores no que se diz a vazão com apenas uma *thread* e *multithread*, e à latência com apenas uma *thread*. Na figura o "Beacon Imm." usa um método de escrita imediato, que envia a mensagem OpenFlow imediatamente ao switch assim que é gerada, além do modelo mais simples, o *Run-to-completion*. O Beacon em sí, também usa o modelo mais simples, mas com um método de escrita que usa algumas flags para controlar as filas de saída e a escrita das mensagens. Por fim o Beacon Queue usa o mesmo método de escrita descrito anteriormente, mas com o modelo *Shared Queue* [38].

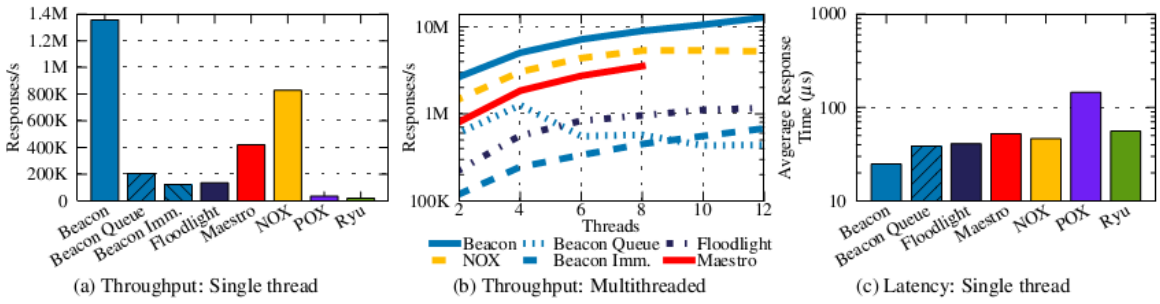


**Figura 31: Desempenho do controlador Beacon [38]**

## 2.5 *Testbeds* e alguns projetos internacionais

Para desenvolver todas as ideias anteriormente escritas foram criados projetos por todo o mundo, muitos deles em alta colaboração e interconectados entre si, para que estas propostas tomassem corpo e pudessem ser aplicadas em contextos cada vez melhores e mais desenvolvidos.

Dentre estes projetos também se destacam as *testbeds* criadas, que nada mais são do que plataformas e arquiteturas sobre as quais são realizados os testes. Geralmente depois da existência de uma estrutura adequada, são abertas as possibilidades de vários projetos pelo mundo utilizarem estas arquiteturas para desenvolverem novas tecnologias e aplicações.

Serão brevemente descritos a seguir os projetos GENI, FIRE, PlanetLab, AKARI, além do OFELIA.

### 2.5.1 GENI

O GENI (Global Environment for Network Innovations) é um projeto da NSF (National Science Foundation) e está intimamente ligado com as pesquisas desenvolvidas na Stanford University, que resultaram em grandes projetos como o próprio OpenFlow [39].

O objetivo deste projeto é criar um laboratório virtual para explorar o futuro da internet em grande escala. Isso quer dizer, disponibilizar acesso para projetos específicos e iniciar portanto um desenvolvimento em rede [40].


**Figura 32: Mapa dos usuários e gerenciadores do GENI [40]**

O GENI usa um conceito anteriormente descrito de *slices*, ou fatias, isso quer dizer que utiliza uma tecnologia de virtualização para dividir a rede em múltiplas redes virtuais para que vários pesquisadores possam compartilhar os recursos oferecidos e também para que exista uma certa independência entre eles, isso quer dizer que um trabalho não pode afetar os outros negativamente, por estar compartilhando os mesmos recursos.

Dentre a estrutura oferecida, encontram-se computadores e servidores virtuais e físicos, roteadores e switches, roteadores programáveis, acesso ao backbone nacional dos Estados Unidos, nós móveis, sensores, entre outros.

Uma parte interessante desta plataforma de testes é que foram desenvolvidas algumas ferramentas que auxiliam os pesquisadores em seus trabalhos, tanto em desenvolvimento, como em monitoramento.

A origem deste projeto vem da união entre universidades, pesquisadores e alguns parceiros especializados, ou que oferecem serviços de computação, armazenamento (*storage*) e recursos de redes. Cada um destes desenvolveu APIs próprias para acesso aos equipamentos e à estrutura. Mas em conjunto, desenvolveram uma API uniforme conhecida como *Aggregate Manager* que permite a interação com os componentes oferecidos pelo sistema [40].

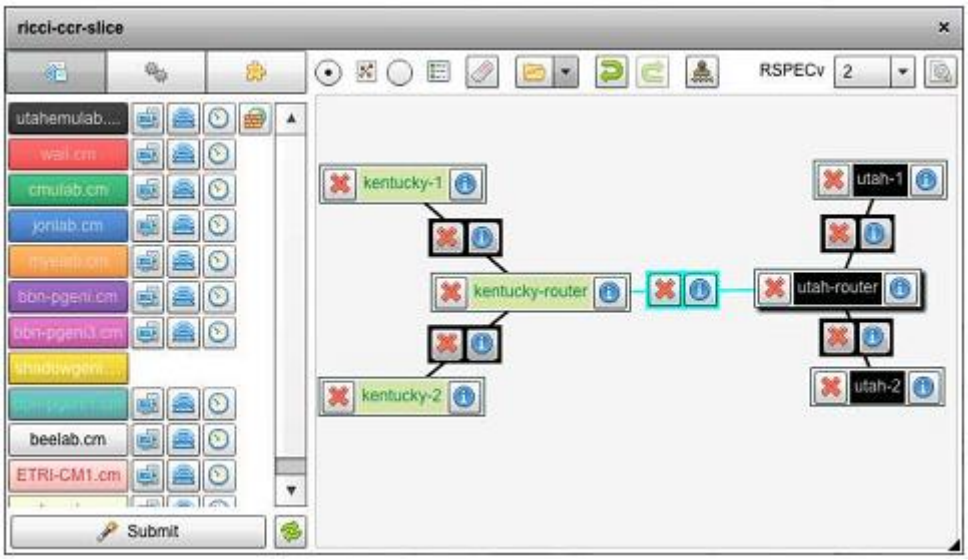


**Figura 33: Ferramenta do GENI destacando a conexão entre dois roteadores [40]**

Resumidamente, a estrutura de cada usuário desta *testbed*, é uma *slice* na qual existe uma estrutura virtual de rede (mapeada em recursos físicos), onde existem pontos de monitoramento, chamados de MPs (*Monitoring Points*), e controladores que usam as informações obtidas por estes pontos e apresentam em uma ferramenta da própria *testbed* os resultados para o que foi desenvolvido de forma gráfica. Estes resultados são armazenados, além das regras dos controladores e outras informações.

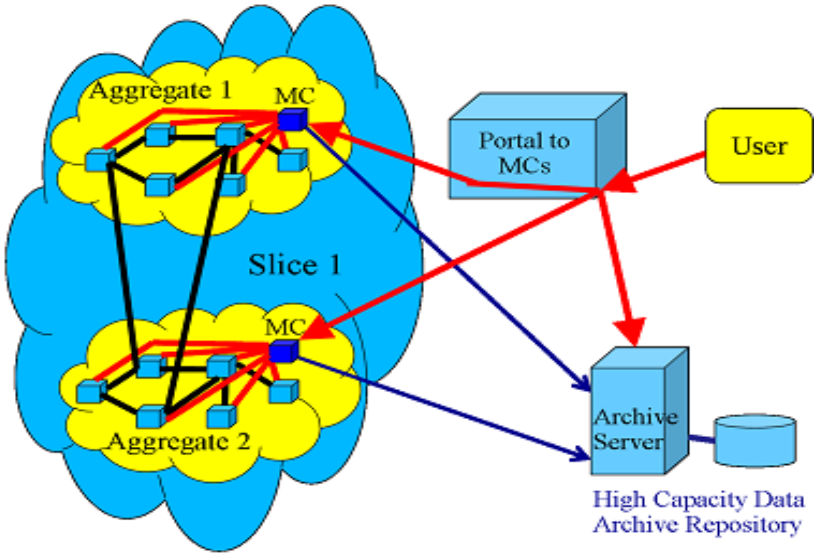


**Figura 34: Estrutura simplificada disponibilizada pelo GENI [40]**

### 2.5.2 PlanetLab

O PlanetLab é uma rede global para o desenvolvimento de novos serviços para a novas gerações de redes. O projeto é relativamente antigo e começou em 2003 e conta com apoio de diversas universidades e empresas que já desenvolveram tecnologias novas de armazenamento distribuído, mapeamento de redes, processamento de filas, entre outras, usando esta plataforma [41].

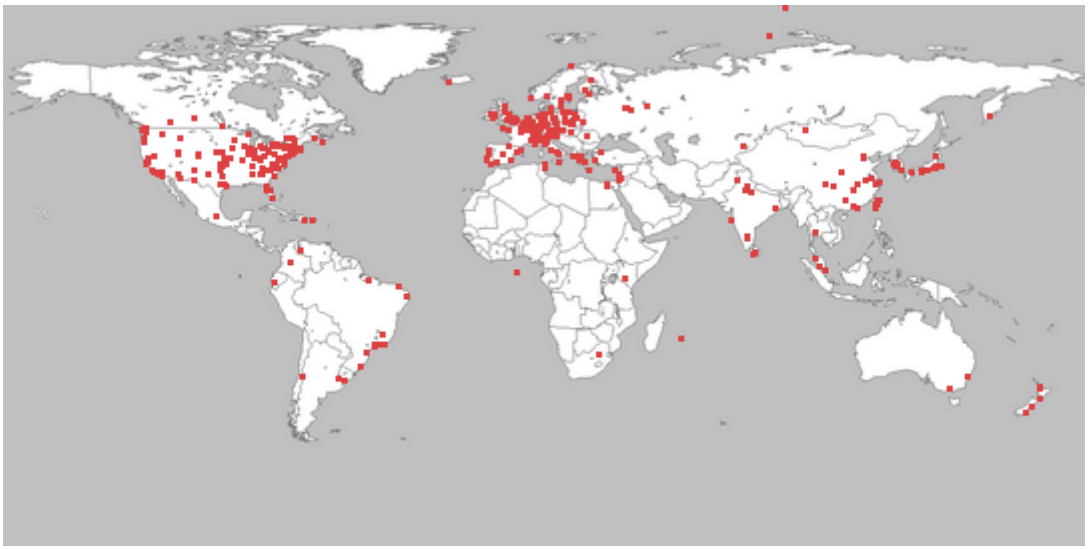

**Figura 35: Nós ligados à rede do PlanetLab [41]**

Os principais objetivos deste projeto são: prover uma plataforma com escala global para o desenvolvimento de pesquisa, prover novos serviços para rede que possam servir uma

comunidade real de usuários e por fim, catalisar a evolução da Internet em uma arquitetura orientada a serviços [42].

Assim como o GENI, este é baseado em virtualização e fatias, distribuídas aos usuários da plataforma, e apresenta um conceito de plano de controle e sistema operacional capaz de gerenciar as aplicações e separá-las do ambiente de desenvolvimento [42].

A arquitetura desta plataforma é dividida em nós, que são máquinas, ou máquinas virtuais, que possuem endereço próprio (no caso da plataforma se usa o IP), um gerenciador de nós, que executa em cada um dos nós e controla os recursos associados a cada máquina virtual, as *slices* que são um conjunto de máquinas virtuais, um serviço de criação e um de gerenciamento para as *slices*, um serviço de auditoria, que armazena os dados transmitidos de cada nó e mapeia a atividade da *slice* associada, uma autoridade de gerenciamento, que é uma aplicação que mantém os servidores e atualiza as aplicações, e por fim um *script* proprietário, mantido por cada usuários para determinar algumas preferências do mesmo, sejam aplicações ou configurações.

Existe também um objeto chamado de RSpec (*Resource Specification*) que mantém informações sobre os recursos físicos de usuário e é usado pelo gerenciador de nós, além do serviço de criação de *slices* [43].

A figura 36 mostra de forma simplificada o relacionamento entre certos componentes da plataforma com o usuário, que deve passar pelos serviços de gerenciamento, para chegar ao nó específico [43].

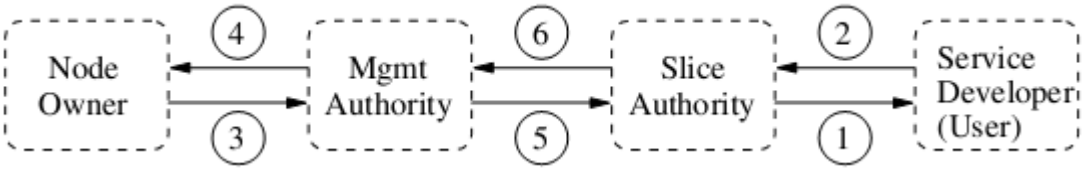


**Figura 36: Relacionamento entre componentes do PlanetLab [43]**

Outra forma de visualizar a arquitetura é com um agente (*Agent*) desenvolvido para a plataforma, que monitora os nós e faz requisições a um intermediário englobado pelos serviços da plataforma em si, este intermediário pode liberar recursos tanto ao agente como aos próprios nós, de acordo com as credenciais apresentadas pelo mesmo. Isso quer dizer que há uma gestão dos serviços oferecidos pela plataforma, que são protegidos ou liberados de acordo com o nó que solicita, além de um agente que monitora os nós e além de requisitar serviços também adiciona novos serviços conforme são disponibilizados pelos nós. Uma visão desta abstração e das interações entre os componentes está na figura 37 [44].

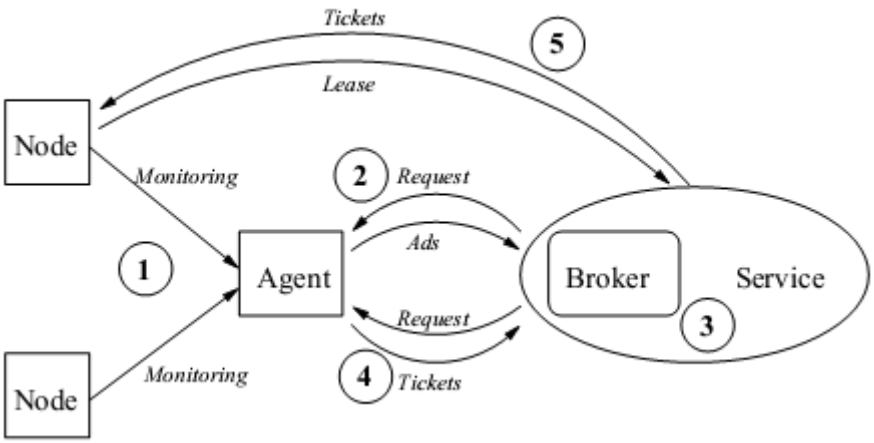


**Figura 37: Visão do funcionamento do PlanetLab [44]**

### 2.5.3 FIRE

O FIRE (*Future Internet Research and Experimentation*) é uma ferramenta direcionada a pesquisa e desenvolvimento de novas redes, arquiteturas de serviços e paradigmas de endereçamento. Esta proposta é baseada em experimentos da larga escala e em projetos individuais de *testbeds*, direcionados para validar novas tecnologias [45].

Pode-se dizer que a visão do projeto pode ser colocada em duas dimensões [46]:

- Promover a investigação de novos conceitos de Internet, voltados para uma experimentação, em uma abordagem que considere a complexidade da arquitetura da Internet.
- Estar disponível para os pesquisadores, tanto na indústria como na academia, oferecendo facilidade em experimentação e em tecnologias futuras da Internet, integrando *testbeds* relevantes na proposta de Internet do Futuro.

Diferentemente dos projetos anteriores o FIRE prevê uma inclusão de *testbeds*, isto quer dizer que não é uma estrutura específica, mas um conjunto destas associadas em uma estrutura maior. Isto quer dizer que o crescimento desta proposta está baseado no compartilhamento de informação entre os projetos participantes, experimentação em longa distância e larga escala, abordagem multidisciplinar (visto a complexidade do sistema) e na integração de propostas complementares [46].

Por ter este caráter de buscar a integração de projetos, foram definidas algumas chaves para utilizar o que é chamado de FIRE Facility, que é um provedor de serviço baseado no que é desenvolvido nos projetos de pesquisas que estão ligados nesta proposta.

Mais do que apenas o desenvolver novas tecnologias, um dos focos do projeto é a visão sobre a questão ambiental, como economia de energia e menor exploração de recursos naturais, até questões sociais como a acessibilidade e custos.

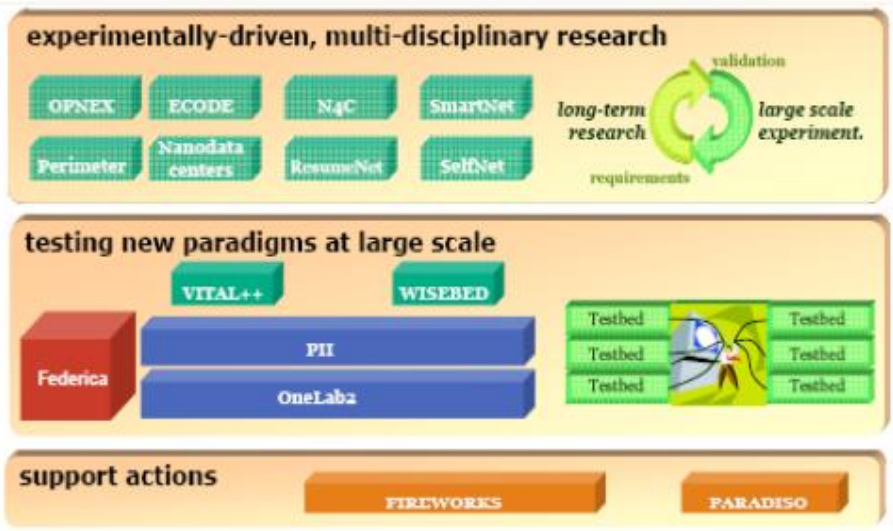


**Figura 38: Projetos ligados ao FIRE [46]**

Esta é uma iniciativa financiada pelo FP7, e que possui inclusive ligação com várias testbeds importantes como ORCA, ORBIT e até as já citadas como o PlanetLab e O GENI. Alguns dos projetos ligados ao FIRE estão divididos na figura 38, como experimentais direcionados para pesquisa, como base para testes em larga escala e também em ações de suporte [46].

São muitas as linhas de atuação dentro do FIRE, justificadas pela alta quantidade de projetos interligados. A figura 39 mostra uma divisão dos projetos em uma visão diferente, de acordo com o escopo, ou objetivo, de cada iniciativa.

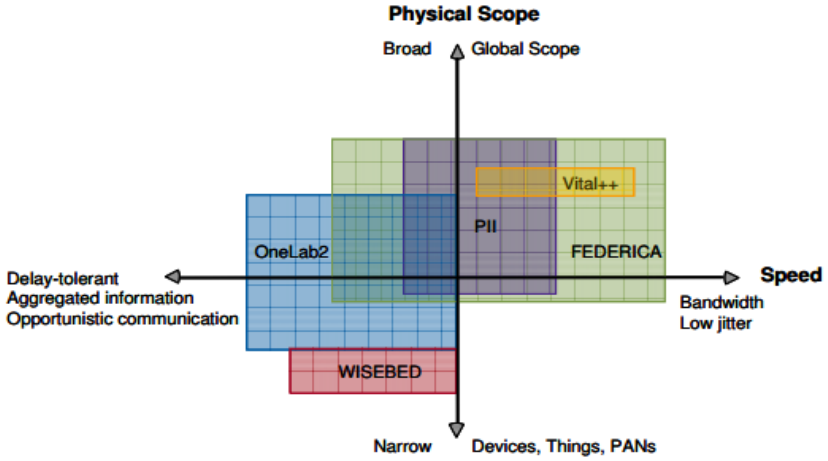


**Figura 39: Visão de alguns projetos de acordo com seu foco [46]**

### 2.5.4 AKARI

Este é um projeto japonês que coloca como seu objetivo primário o desenvolvimento de uma rede para o futuro. Isto deve acontecer através da construção de novas tecnologias para desenvolver uma nova arquitetura e criar esta rede baseada nesta arquitetura [47].

Esta nova arquitetura deve ser desenhada considerando certos fatores e princípios que englobam desde uma adaptabilidade com tecnologias atuais, até uma visão de serviços e requisições futuras. Dentre estes princípios podem ser destacados:

- Acesso óptico: considerando a utilização massiva de fibras ópticas;
- Acesso Wireless;
- Controle da camada de transporte: requisição futura para o desenvolvimento de QoS na rede;
- Roteamento com QoS;
- Segurança;
- Controle Robusto;
- Uma proposta *clean slate* para redesenhar as camadas da Internet.

Existem outros princípios, mas estes citados deixam clara a abrangência da proposta, e o relacionamento destes com a produção de novas aplicações e a manutenção da segurança da informação resultam em uma nova arquitetura. A figura 40 mostra um esquema deste desenvolvimento.

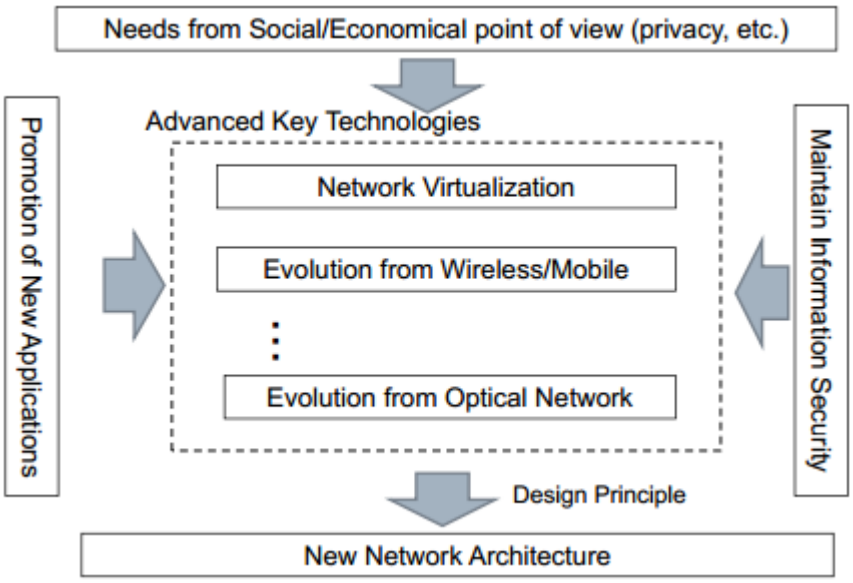


**Figura 40: Processo de construção de uma nova rede de acordo com o AKARI [48]**

Esta nova arquitetura foi denominada New Generation Network (NwGN), a qual não tem nenhuma relação com a NGN (Next Generation Network) [48].

A configuração desta rede estaria em planos que englobam a aplicação, diretamente ligada com o usuário através de uma interface, uma camada sobreposta ao encaminhamento chamada de *Overlay network,* a qual seria customizável e flexível para atender as requisições da aplicação junto as camadas inferiores, uma camada de comando responsável pelo encaminhamento e pela conexão entre o usuário e o que ele requisita, na qual não é feito o uso de IP, e por fim uma camada subposta, chamada de *Underlay network,* que representa o dispositivo final da conexão. Permeando todas as camadas existe uma camada de controle, capaz de interferir e gerenciar os serviços de cada camada [47].

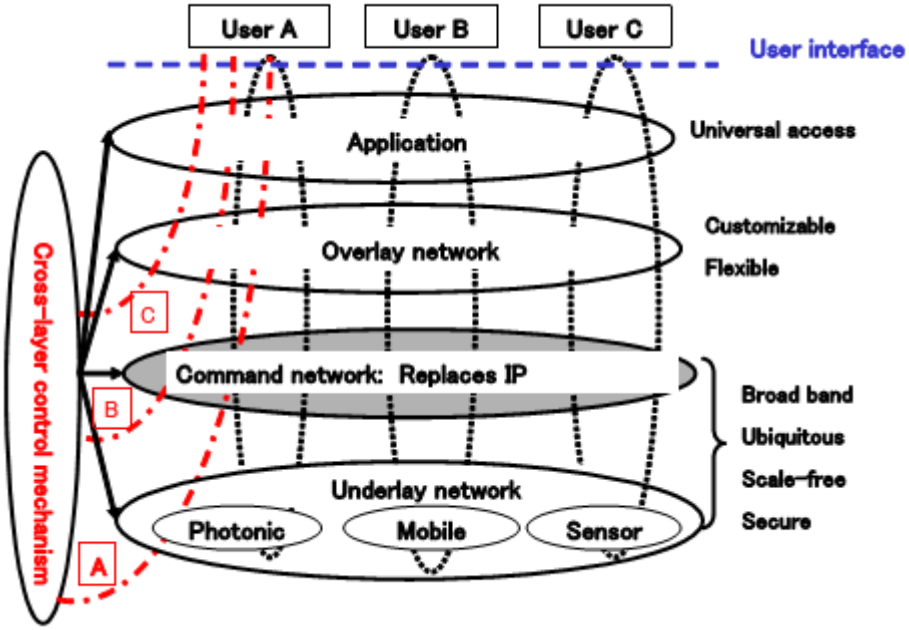


**Figura 41: Diagrama conceitual da NwGN [47]**

Esta é uma rede ainda em construção que segundo o projeto inicial estará pronta a partir do ano 2015. O objetivo é que esta abandone o uso do IP, garanta alta capacidade de transmissões, aumentar a diversidade de dispositivos conectados e de aplicações, reduzir consumo de energia em um alto fator priorizando a transmissão óptica, controle de roteamento, configuração de camadas, entre outros aspectos [49].

E mais, de acordo com a ideia do projeto é importante demonstrar experimentalmente a eficiência da rede, com resultados objetivos, e para isto cita o exemplo do *PlanetLab* como ferramenta para testes [47].

### 2.5.5 OFELIA

O OFELIA (OpenFlow in Europe: Linking Infrastructure and Applications) é mais um projeto colaborativo vinculado ao FP7. A intenção primordial deste é criar uma estrutura não só para experimentação, mas que considere a extensão e o controle da rede de forma precisa e dinâmica. É diretamente ligado ao SDN por promover a utilização do OpenFlow, além dos conceitos de virtualização e controle [50].

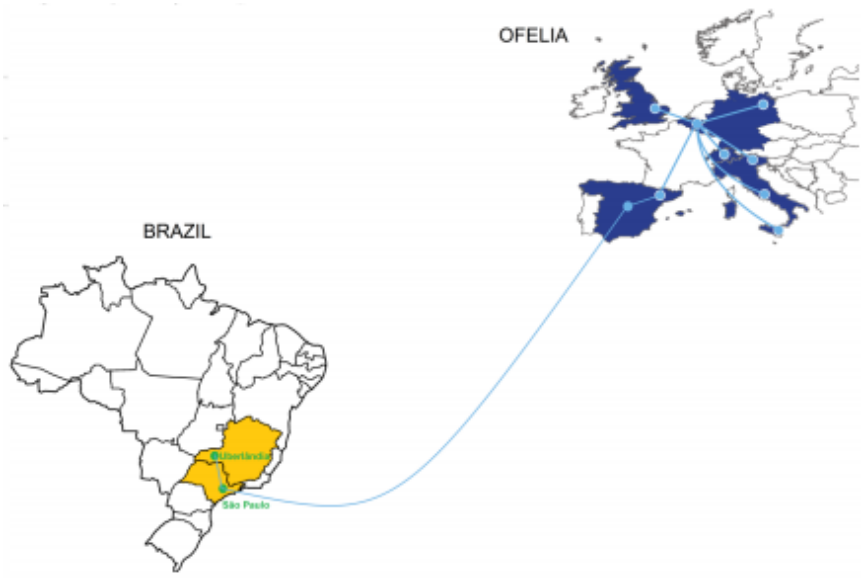


**Figura 42: Localização das ilhas do OFELIA [52]**

Esta é uma proposta que trabalha com parceiros e com o desenvolvimento de ilhas pelo mundo, cada uma com sua estrutura física, que possam estar interconectadas e que possam ser utilizadas remotamente. O OFELIA está voltado, portanto, para criar um substrato real que permita: flexibilidade no controle dos fluxos, um protocolo programável, escalável, desenvolvimento e teste de novos controladores e aplicações de controle [51].

Todo este desenvolvimento está baseado em três princípios:

- Virtualização: criação automática de *slices.*
- Extensão dos controladores para multi-domínio.
- Extensões para tecnologias wireless e ópticas.

Uma visualização básica da estrutura traz a rede como um serviço, que possui uma estrutura central responsável pelo controle, com uma separação do plano de dados do plano de controle (conforme a SDN), e com um encaminhamento ótimo dos dados, com a criação mínima de fluxos de dados e a busca do menor caminho [52].

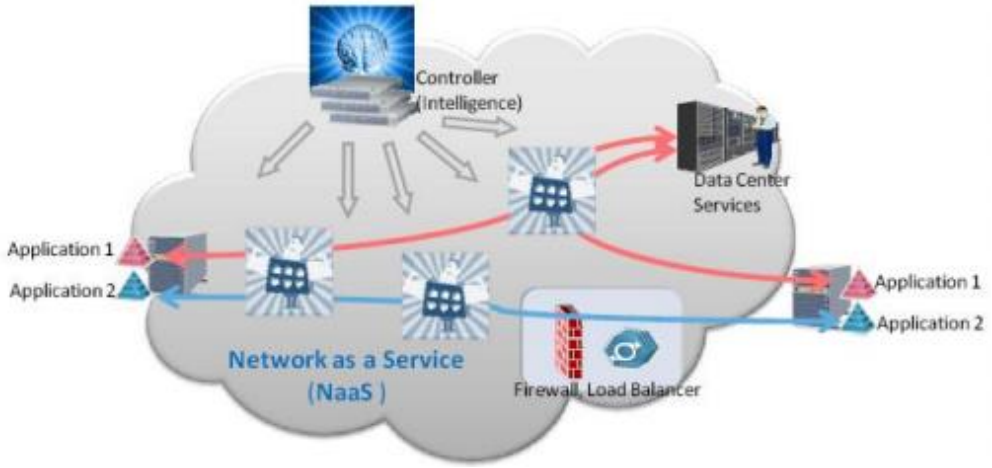


**Figura 43: Visualização básica da estrutura proposta pelo OFELIA [52]**

Especificamente a estrutura de controle é baseada em uma aplicação (uma interface gráfica para os usuários) onde estão os *plug-ins*, tanto do OpenFlow, como de virtualização, entre os outros que surgirem, sendo que esta aplicação está conectada a um servidor responsável pelos gerenciamento dos diretórios que englobam as bibliotecas destes. Cada *plug-in* está conectado a um gerenciador dos recursos deste, pelo qual passa toda a comunicação entre as estruturas ligadas em tais módulos. Na parte da virtualização existem servidores e agentes, na qual os agentes são responsáveis por controlar estes servidores de virtualização. Já na parte destinada ao OpenFlow, o gerenciador está conectado ao FlowVisor, que por sua vez recebe as informações do controlador (NOX) e passa aos switches, que também estão conectados aos servidores de virtualização. A figura 45 dá uma visão desta estrutura de controle [51].

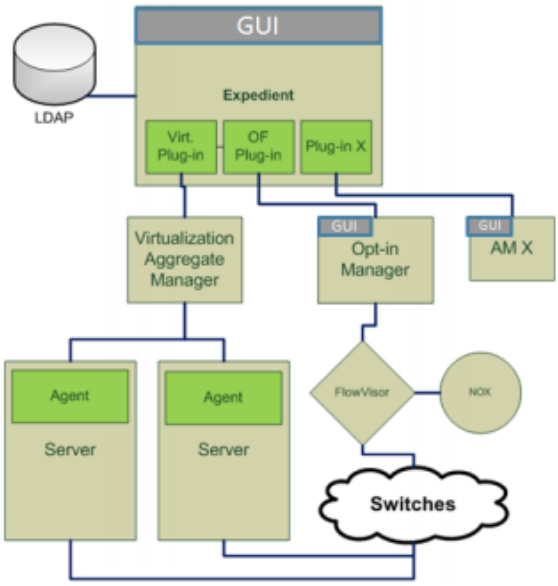


**Figura 44: visão da estrutura de controle [51]**

Apesar de oferecer a ideia básica do processo de cada ilha, cabe aos responsáveis por cada uma desenvolver a estrutura física e experimental para que estas requisições sejam atendidas. Uma demonstração disso é a estrutura mostrada na figura 45, desenvolvida até 2012 em Berlin, que apresenta três controladores, cinco switches e outros equipamentos dispostos em uma estrutura específica, com uma conexão com a rede OFELIA, para o acesso às outras ilhas.

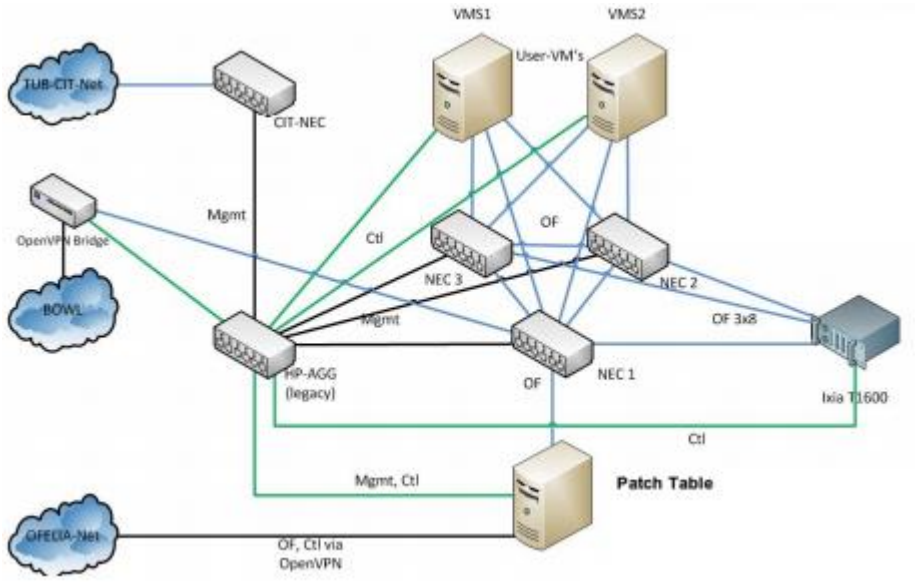


**Figura 45: Um exemplo de estrutura de uma ilha do OFELIA [51]**

Dentro do OFELIA existe um projeto com o nome de EDOBRA (Extending and Deploying OFELIA to Brazil), uma parceria com o Instituto de Telecomunicações de Aveiro em Portugal e com a Universidade Federal de Uberlândia, para construir uma ilha capaz de suportar os experimentos realizados pelos portugueses e com a estrutura contribuir com novas tecnologias ligadas a Internet do Futuro [53].

Primeiramente o desafio era iniciar uma nova arquitetura de rede chamada de ETArch (*Entity Title Architecture*) com experimentação na área de controle de mobilidade usando o protocolo IEEE 802.21, e após dar suporte a pesquisas em novas área, desde QoS até a implantação em cenários reais e maiores. Isto incrementa a política de extensão física do OFELIA, atraindo novos parceiros, desde instituições acadêmicas e centros de pesquisa até indústrias na busca por serviços inovadores e que demonstrem o ganho de qualidade se comparado ao que pode ser realizado na Internet atual [52].

O modelo da ETArch será posteriormente descrito, mas já importante destacar que os princípios básicos do OFELIA estão completamente presentes nesta arquitetura.

## 2.6 Projetos desenvolvidos no Brasil

### 2.6.1 RouteFlow

#### 2.6.1.1 Proposta e estrutura básica

O RouteFlow é uma proposta desenvolvida especificamente pelo CPqD (Centro de Pesquisa e Desenvolvimento em Telecomunicações) que visa uma oferta de serviço de roteamento IP remoto, com um desacoplamento entre os planos de controle e dados. Diferentemente de grande parte dos projetos o IP aqui não será substituído, na verdade a visão do projeto está ligada na flexibilização das redes IP, facilitando o desenvolvimento de outros protocolos e políticas de roteamento [16].

Ainda assim esta proposta usa como base o protocolo OpenFlow, visto que este é um caso de sucesso quando se fala em controle centralizado de redes. Isto quer dizer que os switches utilizados na estrutura física possuem o tratamento das mensagens OpenFlow.

O funcionamento da arquitetura se dá da seguinte forma: existem máquinas virtuais (MV) que formam a infraestrutura de rede, onde é armazenada a lógica de controle dos switches OpenFlow, cada uma executa um código de roteamento aberto. A partir do ponto em que estas máquinas são interconectadas, vê-se uma topologia de rede virtual, onde cada MV representa um switch real, e isto quer dizer que toda a estrutura física passa então a ser mapeada virtualmente.

Um servidor (ou conjunto de servidores), chamado de *RouteFlow Server* é responsável por armazenar toda esta topologia, sendo que este está ligado ao controlador da rede. Assim todas as decisões tomadas no ambiente virtual são comunicadas ao plano de encaminhamento através do controlador [16].

Isto coloca três estruturas fundamentais para o desenvolvimento da arquitetura:

- Switches OpenFlow programáveis com software embarcado;
- Pilhas de protocolo de roteamento open source;
- Servidor de prateleira com alto poder de processamento.

É neste ponto em que o RouteFlow se destaca como uma solução competitiva, pois utiliza estruturas já desenvolvidas, e que relativamente não possuem custo tão alto. E mais, protocolos de roteamento atuais como OSPF (*Open Shortest Path First*) e BGP (*Border Gateway Protocol*) ainda podem ser usados nesta estrutura, por serem mapeados para a rede virtual e posteriormente aplicados aos switches programáveis [16].

A figura 46 mostra uma visão do que foi descrito. A estrutura física de switches é mapeada pelo controlador da rede, que a repassa para o *RouteFlow Server,* o qual cria as máquinas virtuais, uma para cada switch, e por fim as conecta formando a rede virtual. Neste ponto cada MV possui seus próprios atributos, e assim o controlador repassa as regras de encaminhamento que definem toda a lógica de repasse de dados no plano virtual. Com a execução destas regras são definidas as tabelas de encaminhamento para cada MV, e assim através do controlador, estas tabelas são passadas aos switches e a lógica é transportada para o mundo físico. Esta sub-rede mapeada em um controlador pode inclusive ter sua ligação direta com a rede legada, já que isto pode ser facilmente definido nos servidores que mantém a rede virtual e os switches de borda executam normalmente o BGP [16].

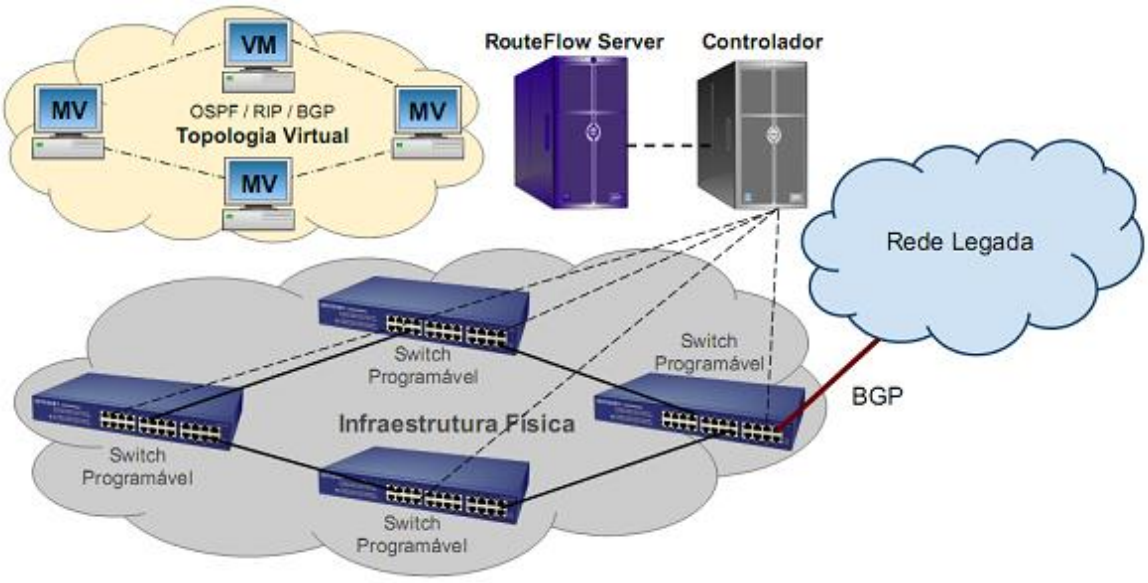


**Figura 46: Visão do RouteFlow [16]**

### 2.6.1.2 Componentes

O RouteFlow pode ser dividido em alguns componentes que podem ser desmembrados em uma estrutura que demonstra o funcionamento desta proposta.

Destaca-se primeiramente o ambiente virtual onde estão as MVs e um software chamado de OVS (Open V Switch). As MVs possuem um módulo chamado de *RouteFlow Slave* que recebe as instruções do *RouteFlow Server* através de um protocolo desenvolvido (*RouteFlow Protocol*). Estas instruções são processadas e usando uma pilha de protocolos de roteamento *open source*, no caso o Quagga, são armazenadas em tabelas de roteamento. Todas as NICs (Network Interface Controller) da cada máquina virtual são conectadas à OVS que gerencia esta

conexão e provê a rede virtual em si. É a OVS que comunica com o controlador para que as regras sejam passadas por este ao mundo físico [54].

É importante destacar que cada MV é uma máquina Linux que executa o *RouteFlow Slave* (um programa executável em C++), sendo que este programa possui as seguintes funções [16]:

- Registrar a MV no controlador como um novo recurso da topologia virtual;
- Gerenciar as interfaces de rede do sistema;
- Detectar as atualizações das tabelas;
- Converter as rotas em fluxos para que sejam instalados no plano de dados.

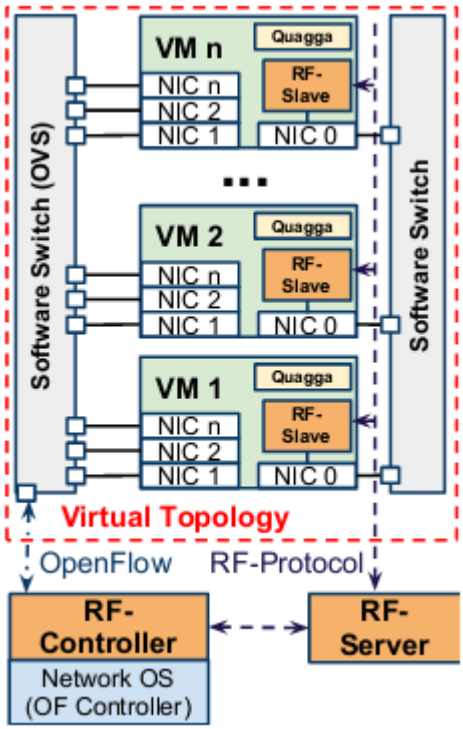


**Figura 47: Plano de controle virtual [54]**

Entre o controlador e o ambiente virtual está o RouteFlow Server, que como já foi descrito, possui o registro de todas as redes virtuais existentes e além disso faz a interface entre o controlador e as máquinas virtuais. Toda a tecnologia deste ambiente RouteFlow também foi implementada em C++, sendo que o controlador executa sobre um NOX, um modelo anteriormente descrito. O controlador possui um módulo para comunicar com as estruturas acima, que é o *RouteFlow Controller* e um módulo para passar as informações ao plano de encaminhamento, o *Network Controller,* sobre o qual são desenvolvidas as aplicações de controle, ou simplesmente as extensões de funcionalidade [54].

O *RouteFlow Controller* é o módulo central desta arquitetura, que recebe os registros das MVs, realiza o mapeamento, cria e remove os fluxos OpenFlow, ou seja, mantém toda a política de encaminhamento do sistema como um todo [16].

Finalmente abaixo do controlador, o *Network Controller* usa o protocolo OpenFlow para passar as regras para a API do switch programável, que constitui o plano de encaminhamento. Neste existe a tabela que é preenchida pelo controlador, e as portas que no plano virtual são simuladas pelas NICs, que são as estruturas responsáveis pela entrada e saída de pacotes de dados [54].

A figura 48 mostra esta interação em um esquema que deixa clara a proposta de separação de plano de controle e plano de dados, e a proposta de virtualização da rede.

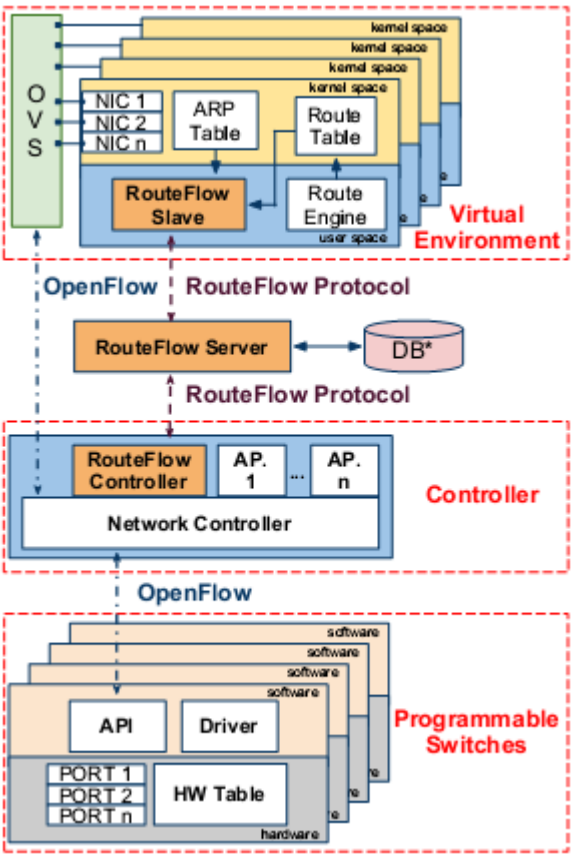


**Figura 48: Componentes do RouteFlow [54]**

### 2.6.1.3 Modos de Operação

Como existe uma virtualização dos elementos, esta separação entre planos de controle e plano de encaminhamento permite uma flexibilidade no mapeamento. Esta flexibilidade gera no RouteFlow três modos de operação diferentes, sendo eles [55]:

- Divisão Lógica (*Logical Split*): Neste modo existe um mapeamento 1:1 entre os switches reais e as máquinas virtuais. Isso quer dizer que a estrutura física será completamente

representada no ambiente virtual e cada conexão e cada switch terá seu representante no plano de controle de forme individual.

- Multiplexação: Mapeamento 1:n dos elementos físicos para os virtuais. A estrutura pode ser simplificada, alguns caminhos podem ser ignorados e alguns switches podem ser "desacoplados" da rede. Este tipo de abordagem é interessante para fatiar a rede e separar certos fluxos conforme alguns parâmetros.
- Agregação: Mapeamento m:1 ou m:n, nestes casos a função exercida por vários switches pode ser acoplada em apenas um switch. Este é um modo de simplificar a abstração da rede, ou até para separar claramente roteamento intra-domínio e fora do domínio.

A figura 49 mostra um exemplo que deixa claro os diferentes resultados de cada modo de operação sobre o mesmo substrato físico.

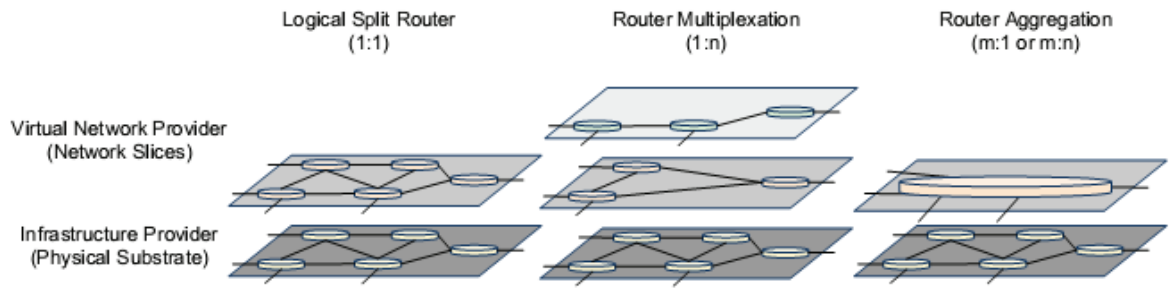


**Figura 49: Modos de operação do RouteFlow [55]**

## 2.6.2 ETArch

### 2.6.2.1 Entidades e títulos

A ETArch é uma proposta de arquitetura baseada em novo modelo de endereçamento, sem a utilização do IP, e de forma mais direta, pode-se dizer que é uma abordagem clean-slate, ou seja, será feita uma completa redefinição do que é usado hoje na Internet. Esta é uma arquitetura que pode ser descrita através de quatro pontos: workspaces, DTS (Domain Title Service), entidades e títulos [56].

Os dois últimos fatores estão ligados à nova modelagem de endereçamento, com o objetivo de desvincular com o modelo IP, atuando nos problemas [57]:

- Dificuldade de mobilidade, pelo forte acoplamento do IP de uma estação com suas vizinhanças. Sempre que se muda de localidade física, é necessário que o IP mude também, assim, por exemplo, em uma situação de *handover* entre duas redes wireless

diferentes, como o IP é alterado as conexões previamente estabelecidas são todas perdidas.

- O problema da segmentação em classes, pois em um caso onde seja ultrapassado o número de endereços IP de uma rede, é necessário alocar outro segmento, e assim, passa-se por um salto na alocação de faixas, o que afeta o crescimento planejado de uma rede.
- Quando uma estação tem duas conexões com a rede ela precisa de dois endereços, e assim uma única estação passa a se comportar como se fossem dois hosts diferentes.

A solução utilizada foi denominada Modelo de Título, no qual uma entidade é representada por um título. Uma entidade é uma representação de um "requisitante" na rede, isso quer dizer que engloba o usuário, o dispositivo e a aplicação, sendo que estes três juntos passam a ser identificados por um nome [57].

A entidade deve ser suportada por uma camada de serviços, que seja capaz de compreender e satisfazer as necessidades desta. A figura 50 mostra uma comparação entre as camadas do modelo TCP/IP e o modelo de título.

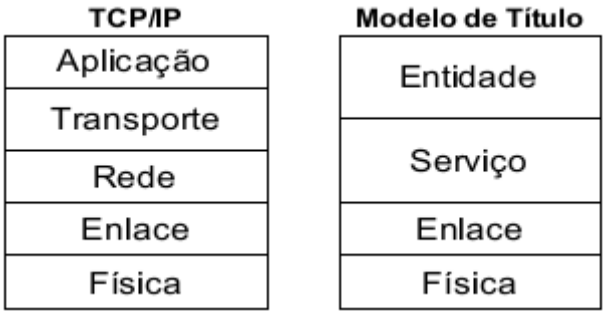


**Figura 50: Camadas TCP/IP e Modelo de Título [57]**

### 2.6.2.2 DTS

O Domain Title Service é um sistema distribuído que tem acesso a todos os elementos de rede e possui a responsabilidade de gerenciar e manter as entidades e títulos, e mais ele passa a ser responsável por oferecer os parâmetros de QoS. Em suma este é um ator do sistema que faz o papel de controlador, além de detentor dos nomes de toda a rede.

Isso traz a necessidade de um serviço de autenticação, ou seja, toda vez que uma entidade entrar na rede, ela deve usar um endereço padrão na camada de enlace para acessar o DTS e se cadastrar como válida e disponível, e isto engloba inclusive os elementos de rede, switches e roteadores, e assim é possível para este sistema obter uma descrição da rede, e mais uma vez a

virtualização do ambiente real [58]. A figura 51 mostra a topologia da rede e o posicionamento do DTS sobre esta.

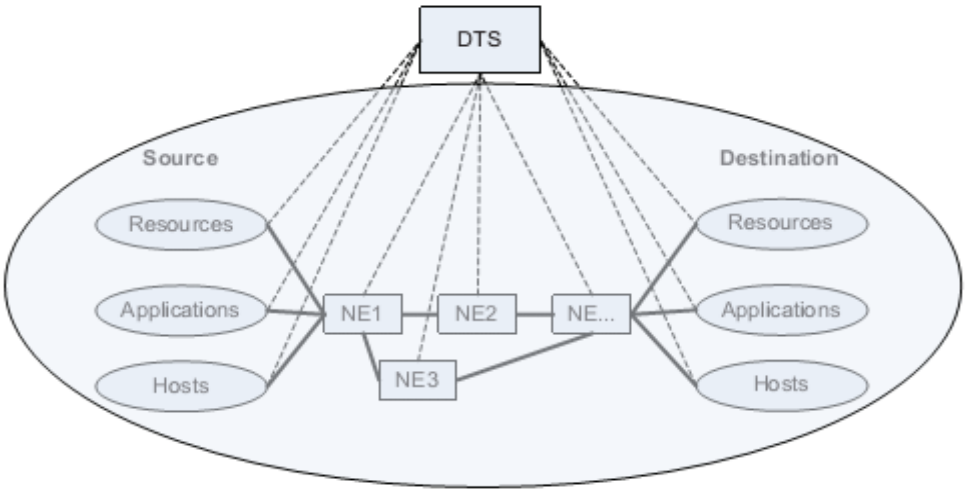


**Figura 51: Topologia DTS [58]**

Este sistema responsável pelo controle da rede é composto por uma série de agentes, ou servidores, que estão responsáveis por partes da rede, e estes agentes conectados são capazes de abranger toda a rede. O nome dado para este agente foi DTSA (Domain Title Service Agent). Sempre que uma entidade é registrada em um DTSA, ela passa para este as requisições de QoS e os protocolos que precisa usar, o DTSA determina a pilha de protocolos e concede esses recursos [58].

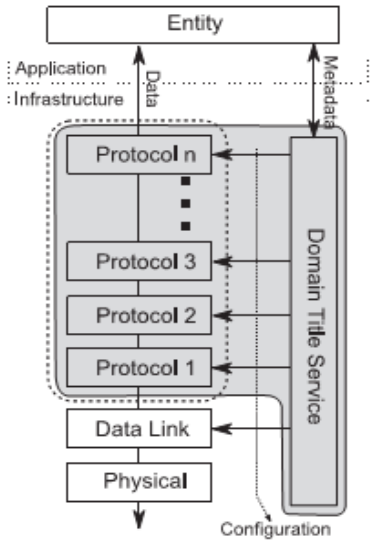


**Figura 52: DTS e a pilha de protocolos [58]**

A figura 52 mostra a ação do DTS na pilha de protocolos e o posicionamento da entidade nesta estrutura. Como os elementos de rede devem ser configuráveis, esta pilha de protocolos pode ser adaptada pelo caminho que a transmissão de pacotes vai percorrer, e este ponto é de suma importância para que a rede não fique presa a uma pilha de protocolos específicos. O DTSA pode ser implementado como um controlador OpenFlow, o que o aproxima da proposta de controlador descrita na SDN [58].

### 2.6.2.3 Workspaces

Um conceito também importantíssimo na visão da ETArch é o de *workspace*. Na verdade o este é um canal criado em uma transmissão de dados, no qual podem ser participantes várias entidades. De outra forma, é um barramento lógico que conecta várias instâncias de entidades ao mesmo tempo [56].

A inspiração desta ideia está na tecnologia *multicast*, onde os dados são enviados ao *workspace* e os participantes têm acesso ao mesmo [56].

Um exemplo pode deixar bem claro o objetivo esta ideia, sendo assim, uma entidade que busca um servidor que esta transmitindo um evento, informa ao DTSA, que é responsável por alocar o caminho entre estes, ou seja, determinar os switches que farão esta ligação, e assim um *workspace* é criado para que os dados sejam transmitidos.

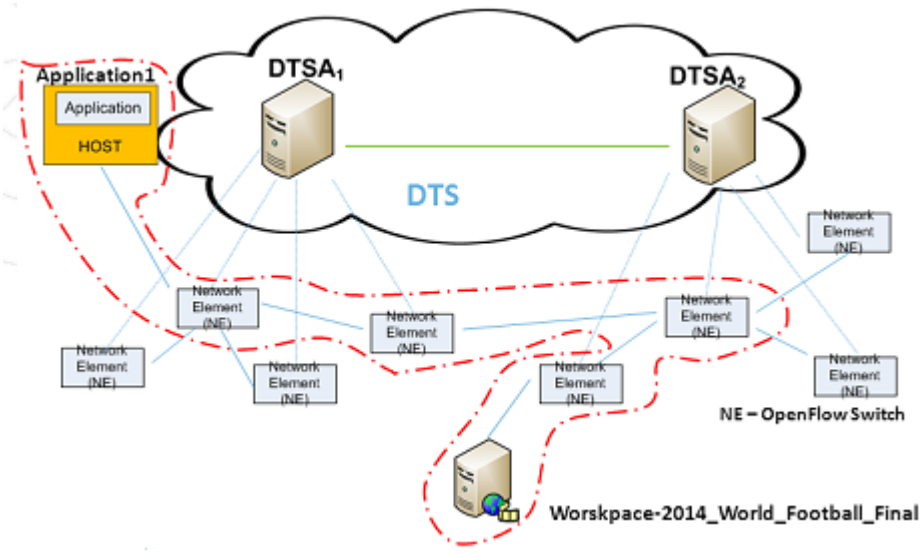


**Figura 53: Workspace com duas entidades [52]**

Se uma segunda entidade deseja buscar a mesma transmissão, ao invés de criar uma segunda conexão para o fluxo de dados como acontece no TCP, ao informar ao DTSA sobre sua

requisição, é criada uma solicitação para entrar no *workspace*. A partir deste momento é determinado o caminho para que esta entidade seja ligada ao *workpace*, e o fluxo de dados até o ponto comum (*switch*) entre os dois usuários é único, e duplicado a partir daquele ponto pelo *switch* por uma determinação do DTSA. Desta maneira evitam-se transmissões idênticas e paralelas desde a fonte dos pacotes até todos os usuários que assistem ao evento.

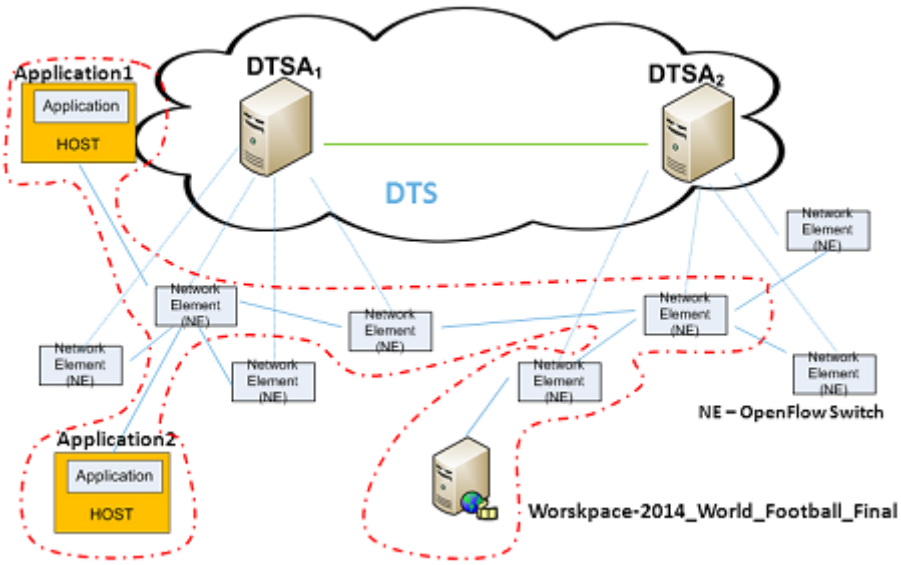


**Figura 54: O Workspace com a adesão de uma terceira entidade [52]**

Graças ao conceito de título, mesmo que uma destas entidades se mova no espaço físico, o DTSA é capaz de perceber esta movimentação e redefinir os pontos (switches) participantes deste *workspace*, o que completa o conceito de mobilidade não previsto na arquitetura TCP/IP.
Finalmente

### 2.6.2.4 Estrutura completa

A figura 55 dá uma visão da estrutura completa da ETArch, com a representação dos quatro principais protocolos para o funcionamento desta rede. Primeiramente o protocolo OpenFlow, como já descrito, é responsável pela comunicação entre o controlador (DTSA) e os elementos de rede, para que estes possam encaminhar os pacotes conforme a especificação no plano de controle.

O Protocolo do plano de dados é o mesmo da arquitetura TCP/IP, conhecido como Ethernet, responsável apenas pela conexão entre os pontos adjacentes. O *Entity Title Protocol* realiza o registro da entidade no DTSA, e ainda informa sua localização, e por isto existe a grande importância deste para a mobilidade do sistema. Finalmente o DTS *Control Protocol* faz a

comunicação entre os DTSAs, para que possam monitorar seus registros de entidades (quais são as entidades que estão na parte da rede que cada um controla) e para que *workspaces* possam ser criados em grandes distâncias, ou entre várias sub-redes [56].

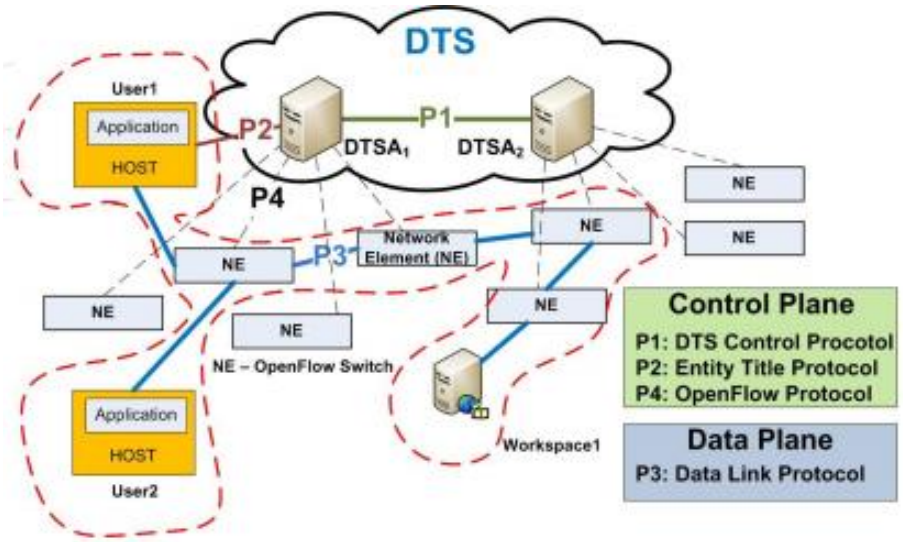


**Figura 55: Visão completa da ETArch [52]**

# 3 CONCLUSÕES

A primeira conclusão relativa às pesquisas realizadas para a construção deste documento é a necessidade, ou pelo menos a mobilização no sentido da busca por novas tecnologias relativas às arquiteturas de rede. Os protocolos das camadas da Internet são em sua grande maioria muito antigos e sem funcionalidades flexíveis e inteligentes, o que obriga o deslocamento de toda a inovação para a aplicação, e passam a atuar como fatores limitantes em relação ao desenvolvimento de novos serviços ou na qualidade destes (QoS).

A maioria dos protocolos criados nos últimos anos está na camada de aplicação, para que outras tecnologias possam “descer” ao núcleo da rede e transmitir suas informações pela Internet. Este fator é que constitui o gargalo IP, já que todos estes acabam passando pela camada de rede através desta tecnologia por ser a única opção.

A dimensão da Internet e a grande quantidade de novas ideias também trazem urgência em melhorias, pois um mercado volumoso deixa de ser explorado devido a ausência de suporte para funções como alta mobilidade.

Tecnologias divididas nos pontos de Internet dos objetos (M2M), pessoas e conteúdo são cada vez mais utilizadas, e ganhos na estrutura e definição da rede representariam ganhos nestas ideias também.

Neste contexto é que surgem vários projetos pelo mundo, cada um voltado para uma funcionalidade específica, para que apareça evolução da rede, e grande parte deles propõe o fim do uso do IP. Como acontece na SDN, uma ideia desenvolvida para fundamentar novas arquiteturas pelo mundo.

E esta ideia gerou grande movimento, tanto de pesquisadores como de recurso financeiro, e desencadeou uma série de grupos de desenvolvimento. Alguns destes grupos foram destacados, muitas vezes com focos diferentes, como desenvolver um novo modelo de endereçamento, caso da ETArch, ou fornecer recursos físicos para a conexão entre vários grupos, como o FIRE.

Em grande parte desses existem várias premissas comuns, algumas bem claras, como a virtualização dos elementos físicos em uma abstração por software, a existência de um controlador com visão sobre esta abstração e com capacidade de redefinir os padrões de endereçamento, a presença de elementos de rede configuráveis, que recebem informações dos controladores (através de tecnologias como o protocolo OpenFlow) e alteram suas tabelas ou definições de fluxos.

A presença massiva de expressões como a definição de fluxos, o fatiamento da rede (divisão em *slices*) mostram uma nova perspectiva de Internet para o futuro.

Alguns fatores ainda são limitantes, como é o caso da segurança, já que a rede passa a ser controlada por software e os usuários tem sua identificação clara definida nas estruturas de controle, além de transferirem informações em ambientes compartilhados (como os *workspaces*), casos em que invasões seriam extremamente danosas. Outro fator é a própria dimensão da rede e a estrutura já existente, toda baseada em arquitetura IP, que indicaria grandes investimentos em mudança de tecnologia para as empresas provedoras de acesso, o que pode indicar uma propagação mais lenta de uma nova arquitetura.

Fato é que grandes empresas e universidades importantes somam esforços nestas propostas e algumas destas ideias já tomam identidade comercial, por exemplo, os switches OpenFlow que já são comercializados normalmente.

Para o desenvolvimento futuro deste trabalho está a aproximação com a estrutura existente no Brasil, especificamente na Universidade Federal de Uberlândia, que já apresentou frutos no conceito de mobilidade e oferece condições para pesquisa e desenvolvimento em novas áreas, como no desenvolvimento da arquitetura em ambientes maiores (a aplicação do conceito da SDN, especificamente a ETArch, em uma rede com alta quantidade de fluxos de dados), aplicação de QoS na rede, dinamização na utilização de protocolos, entre muitas outras possibilidades.

# 4 REFERÊNCIAS


1. UOL: UNIVERSO ONLINE (São Paulo). **Número de internautas no Brasil ultrapassa 100 milhões, segundo Ibope.** 2013. Disponível em: <http://tecnologia.uol.com.br/noticias/redacao/2013/07/10/numero-de-internautas-no-brasil-ultrapassa-100-milhoes-segundo-ibope.htm>. Acesso em: 25 jan. 2014.
2. ROYAL PINGDOM. **Internet 2012 in numbers.** 2013. Disponível em: <http://royal.pingdom.com/2013/01/16/internet-2012-in-numbers/>. Acesso em: 25 jan. 2014.
3. KUROSE, James F.; ROSS, Keith W.. **Redes de Computadores e a Internet**: Uma Abordagem Top-Down. 5. ed. Boston, Ma: Pearson Education, 2010.
4. TRONCO, Tania. **New Network Architectures**: The Path to the Future Internet. Campinas: Springer-verlag Berlin Heidelberg, 2010.
5. Regina Xavier de et al. **Soluções para redes ópticas avançadas**: Projeto GIGA. Campinas: 2005.
6. TLC BRAZIL (São Caetano do Sul). **A evolução da Web na direção dos negócios**. 2011. Disponível em: <A evolução da Web na direção dos negócios>. Acesso em: 25 jan. 2014.
7. BAX, Marcelo Peixoto; LEAL, George Jamil. **Serviços Web e a evolução dos serviços em TI**: Web services and the IT services evolution. 2001. Disponível em: <http://www.dgz.org.br/abr01/Art_02.htm>. Acesso em: 25 jan. 2014.
8. EC FIARCH GROUP. **Fundamental Limitations of current Internet and the path to Future Internet**. 01 mar. 2011.
9. TURBO CISCO. **Protocolos**. 2013. Disponível em: <http://turbocisco.wordpress.com/2013/10/16/protocolos/>. Acesso em: 25 jan. 2014.
10. PAN, Jianli; PAUL, Subharthi; JAIN, Raj. **A Survey of the Research on Future Internet Architectures**. Future Internet Architectures: Design and Deployment Perspectives, Washington, Dc, p.26-36, jul. 2011.
11. EVANS, Dave. **The Internet of Things**: How the Next Evolution of the Internet Is Changing Everything. abr. 2011.
12. CACIATO, Luciano Eduardo. **Virtualização e Consolidação dos Servidores do Datacenter**. Universidade Estadual de Campinas – UNICAMP: [2009?].
13. EUROPEAN COMMISSION. **FP7**: the future of European Union research policy. 2012. Disponível em: <http://ec.europa.eu/research/fp7/index_en.cfm>. Acesso em: 25 jan. 2014.

14. EUROPEAN FUTURE INTERNET PORTAL. **FP7 Projects**. Disponível em: <http://www.future-internet.eu/activities/fp7-projects.html>. Acesso em: 25 jan. 2014.
15. SDN CENTRAL. **What's Software-Defined Networking (SDN)?** Disponível em: <http://www.sdncentral.com/what-the-definition-of-software-defined-networking-sdn/>. Acesso em: 26 jan. 2014.
16. ROTHENBERG, Christian Esteve et al. **OpenFlow e redes definidas por software**: um novo paradigma de controle e inovação em redes de pacotes. Cad. Cpqd Tecnologia, Campinas, v. 7, n. 1, p.65-76, jun. 2011.
17. MCKEOWN, Nick et al. **Software Defined Networks (SDN).** Stanford, Ca: [201?]. 51 slides, color.
18. MONSANTO, Christopher et al. **Composing Software-Defined Networks**. Princeton: [2013?].
19. MCKEOWN, Nick. **Software-defined Networking**. Infocom: 2009. 64 slides, color.
20. ONS: OPEN NET SUMMIT. **SDN:** Transforming Networking to Accelerate Business Agility. Disponível em: <http://www.opennetsummit.org/why-sdn.html>. Acesso em: 26 jan. 2014.
21. STRETCH, Jeremy. **What the Hell is SDN?** 2013. Disponível em: <http://packetlife.net/blog/2013/may/2/what-hell-sdn/>. Acesso em: 27 jan. 2014.
22. DAS, Saurav; PARULKAR, Guru; MCKEOWN, Nick. **SDN Based Unified Control Architecture**. Stanford, Ca: [201?]. 2 p.
23. FOSTER, Nate et al. **Languages for Software-Defined Networks**. Cornell University: 2012. 7 p.
24. PLVISION. **Software-Defined Networking**. Disponível em: <http://plvision.eu/expertise/networking/software-defined-networking/>. Acesso em: 27 jan. 2014.
25. MININET. **An Instant Virtual Network on your Laptop (or other PC)**. Disponível em: <http://mininet.org/>. Acesso em: 27 jan. 2014.
26. SDN CENTRAL. **What is OpenFlow?** Disponível em: <http://www.sdncentral.com/what-is-openflow/>. Acesso em: 28 jan. 2014.
27. JUNIPER NETWORKS. **OpenFlow**. Disponível em: <https://developer.juniper.net/content/learn/technologies/openflow.page>. Acesso em: 28 jan. 2014.

28. CISCO. **An Introduction to OpenFlow**. 2013. 37 slides, color. Disponível em: <http://www.cisco.com/web/solutions/trends/open_network_environment/docs/cisco_one_webcastan_introduction_to_openflowfebruary142013.pdf>. Acesso em: 28 jan. 2014.

29. YAP, Kok-klong et al. **Blueprint for Introducing Innovation into the Wireless Networks we use every day**. Stanford, Ca: 2009.

30. MCKEOWN, Nick et al. **OpenFlow**: Enabling Innovation in Campus Networks. Acm Sigcomm: Computer Communication Review, Stanford University, v. 38, n. 2, p.69-74, abr. 2008.

31. BRITO, Felippe César Fortunato de. **Avaliação do impacto no desempenho do OpenFlow eXtensible Match**. 2013. 58 f. TCC (Graduação) - Curso de Sistemas de Informação, Universidade Federal de Uberlândia, Uberlândia, 2013.

32. SHERWOOD, Rob et al. **Carving Research Slices Out of Your Production Networks with OpenFlow**. Stanford, Ca: [2009?]. 2 p.

33. SDN CENTRAL. **FlowVisor**. Disponível em: <http://www.sdncentral.com/projects/flowvisor/>. Acesso em: 29 jan. 2014.

34. SHERWOOD, Rob et al. **FlowVisor**: A Network Virtualization Layer. Los Altos, Ca: 2009. 14 p.

35. SHERWOOD, Rob et al. **Can the Production Network Be the Testbed?** Los Altos, Ca: [2009?]. 14 p.

36. MUNTANER, Guillermo Romero de Tejada. **Evaluation of OpenFlow Controllers**. out. 2012. 77 p.

37. GUDE, Natasha et al. **NOX**: Towards an Operating System for Networks. [2009?]. 6 p.

38. ERICKSON, David. **The Beacon OpenFlow Controller**. Stanford, Ca, Usa: [2013?]. 6 p.

39. GENI. **Exploring networks of the future**. Disponível em: <http://www.geni.net/>. Acesso em: 31 jan. 2014.

40. DUERIG, Jonathon et al. **Getting Started with GENI**: A User Tutorial. Acm Sigcomm Computer Communication Review, Utah, v. 42, n. 1, p.72-77, jan. 2012.

41. PLANETLAB. **An Open Plataform for developing, deploying, and accessing planetary-scale services**. Disponível em: <https://www.planet-lab.org/>. Acesso em: 31 jan. 2014.

42. PETERSON, Larry; ROSCOE, Timothy. **The Design Principles of PlanetLab**. Princeton University: [2006?]. 6 p.

43. PETERSON, Larry et al. **PlanetLab Architecture**: An Overview. Princeton University: mar. 2006. 30 p.

44. CHUN, Brent et al. **PlanetLab**: An Overlay Testbed for Broad-Coverage Services. Princeton University: jan. 2003. 9 p.

45. FIRE: FUTURE INTERNET RESEARCH EXPERIMENTATION. **What is Fire?** Disponível em: <http://www.ict-fire.eu/getting-started/what-is-fire.html>. Acesso em: 31 jan. 2014.

46. FIRE: FUTURE INTERNET RESEARCH EXPERIMENTATION. **FIRE, White Paper**, 2009. Disponível em: <http://www.ict-fireworks.eu/fileadmin/documents/FIRE_White_Paper_2009_v3.1.pdf>. Acesso em: 31 jan. 2014.

47. AKARI ARCHITECTURE DESIGN PROJECT. **New Generation Network Architecture AKARI Conceptual Design (ver1.1)**. [S.I.:s.n.], 2008. 217 p.

48. KOBOTA, Fumito. **AKARI Project**: Design for the New Generation Network. [S.I.], [200?]. 19 slides, color.

49. KOBAYASHI, Hisashi. **The New GenerationNetwork (NwGN) Project**: Its Promises and Challenges. 2011. Disponível em: <http://hp.hisashikobayashi.com/wp-content/uploads/2011/12/Speech-at-NICT-Symp-Nov-9-English-Full-Text.pdf>. Acesso em: 31 jan. 2014.

50. OFELIA. **OpenFlow in Europe**: Linking Infrastructure and Applications. Disponível em: <http://www.fp7-ofelia.eu/about-ofelia/>. Acesso em: 01 fev. 2014.

51. OFELIA. **The EU FP7 Project and The European OpenFlow Experimental Facility**. [S.I.], 2012. 12 slides, color.

52. GUIMARÃES, Carlos. **Extending and Deploying Ofelia in BRAzil (EDOBRA)**. Covilhã: 2013. 24 slides, color.

53. ATNOG. **OFELIA - EDOBRA**. 2012. Disponível em: <http://atnog.av.it.pt/content/ofelia-edobra>. Acesso em: 01 fev. 2014.

54. NASCIMENTO, Marcelo R. et al. **Virtual Routers as a Service**: The RouteFlow Approach Leveraging Software-Defined Networks. Campinas, jun. 2011. 4 p.

55. NASCIMENTO, Marcelo R. et al. **The RouteFlow approach to IP routing services on software-defined networks**. In: II WORKSHOP DE PESQUISA EXPERIMENTAL DA INTERNET DO FUTURO. Campinas. [2010?]. p. 25 - 28.

56. SILVA, Flávio de O. et al. **Enabling Network Mobility by Using IEEE 802.21 Integrated with the Entity Title Architecture**. In: IV WORKSHOP DE PESQUISA EXPERIMENTAL DA INTERNET DO FUTURO. [2013]. p. 29 - 34.

57. PEREIRA, J. H. S. et al**. Title Model for Computer Networks Optimization**. Ieee Latin America Transactions, [S.l], v. 9, n. 2, p.219-230, abr. 2011.

58. SILVA, Flávio de O. et al. **Domain Title Service for Future Internet Networks**. In: II WORKSHOP DE PESQUISA EXPERIMENTAL DA INTERNET DO FUTURO. [2011?]. p. 33 - 36.