DevNet Project

Objective: Enable Applications to Drive Network Services

Background:

Verizon's Enterprise customers need to become increasingly agile. As their applications are rapidly changing, their network resources must shift just as quickly. DevOps teams are built to break down the human barrier between applications and operations. DevOps tools and solutions are offered to enable those teams to succeed.

Solution:

Verizon DevNet: The connection between the application DevOps tools and network services. A network that dynamically reacts to application demands instantly (without learning how networking functions).

DevNet enables application deployment to drive network services seamlessly and without human interaction or delays putting the control of application deployment back on the Developers or DevOps team.

Story:

In the drive to stay competitve, enterprises are forced to constantly enhance their applications and services at a rapid pace. They moved to a more Agile development methodology to increase speed while maintaining quality of the application.

While in the past new applications were deployed at a rate which allowed manual network configuration and re-configuration to keep up, with the new Agile methods, continuous delivery and continuous deployment, network configuration has lagged behind slowing down the agility promised by DevOps.

To fully enable this rapid enhancement to applications, network monitoring and policy requirements need to be associated directly with the applications themselves and not applied as an afterthought. This is a 'shift-left' of network configuration into the application development cycle. Once the requirements have been bound to the application, network monitoring and policy changes are no longer required to be managed manually. They can be automated in the DevOps pipeline just like a change to application architecture or code.

opsani

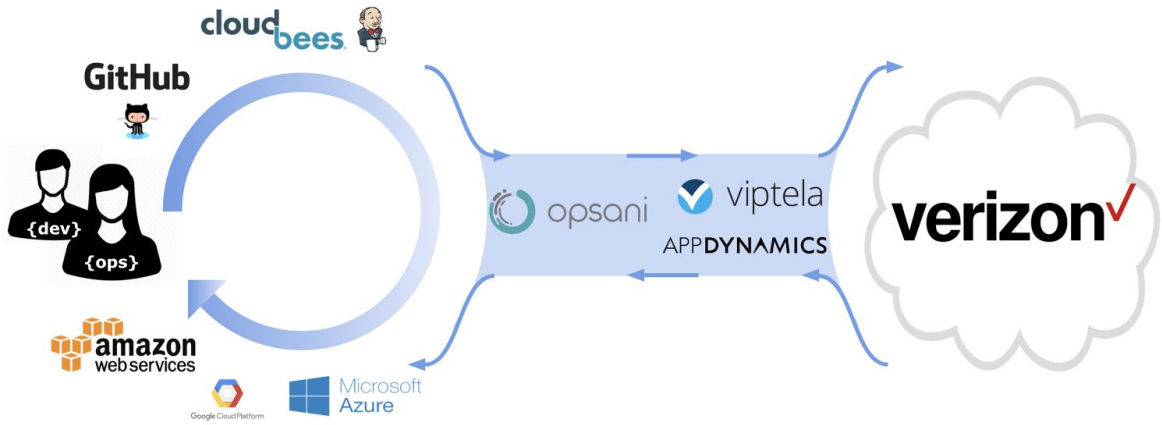
The graph above depicts the well accepted DevOps tool chain. Developers check in code and the CI/CD system builds and tests the app before deploying to infrastructure using a pipeline script. It works fine for many new stateless apps built with microservices. Unfortunately, it's problematic for many traditional enterprise applications and re-architecting them for this tool chain would be prohibitively expensive.

There are other issues with the current tool chain as well. This view of DevOps only deals with the code and deploying it to compute infrastructure. The network is completely separate. It's still configured by network admins/operators and is just a "cloud" to developers.

Opsani is expanding the ability of DevOps to work for traditional enterprise applications by providing new capabilities in the toolchain that make application architecture something developers can specify. This creates a structure for dealing with infrastructure that needs to be set up or torn down to support the app. Infrastructure like - networks.

At the same time Verizon is looking to make network services consumable in much the same way Amazon changed the way we think about the data center. By using AppDynamics and Viptela, Verizon can provide developers control of network resources.

However, the traditional DevOps tool chain doesn't have a place to incorporate network services. There's no definition of them. No orchestration of them.



With the addition of Opsani's Skopos system, Viptela and AppDynamics can extend the DevOps tool chain to embrace the network throughout the application's life-cycle. The impact will give developers the ability to directly specify network services for their apps.

To understand how this new DevOps architecture will work we can consider a few use cases ranging from the simple deployment of a new application to more advanced error resolution.

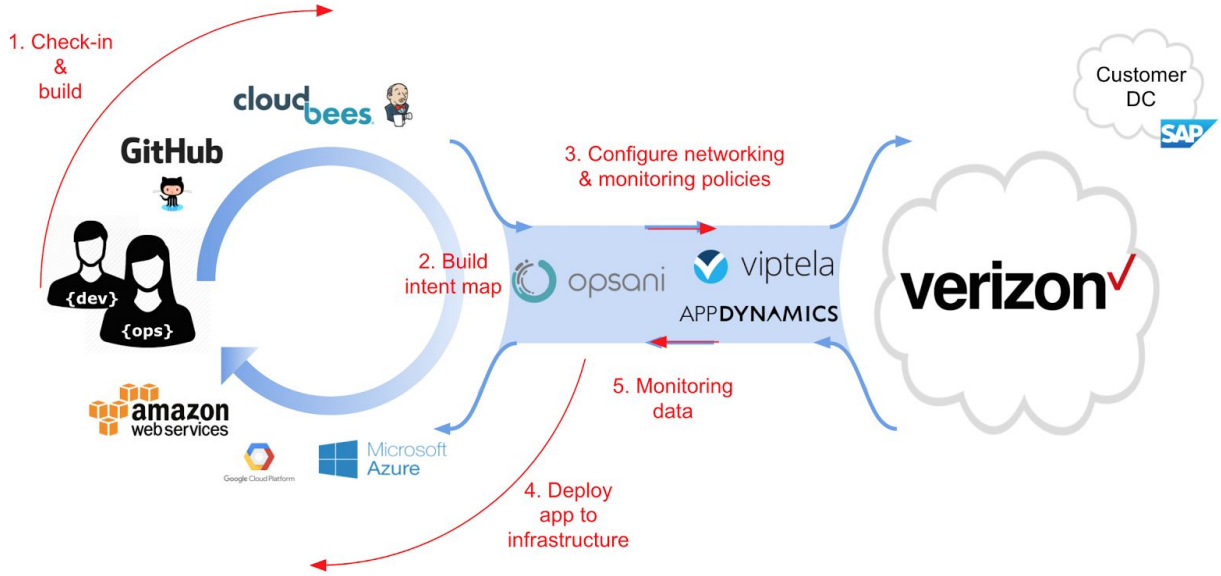
Use Cases:

1. Launch an app and its network services
2. Terminate an app and its network services
3. App changes requiring new network services
4. Adding a 2nd geography and network services
5. Geographic auto-scaling
6. Relocating an app due to DC issues

Basic App Lifecycle Use Cases

Use Case #1: Launch an App and Its Network Services

Our first use case is the deployment of a new application.



We're assuming that for security the network has been configured to either reject all unknown traffic or route it to a security system.

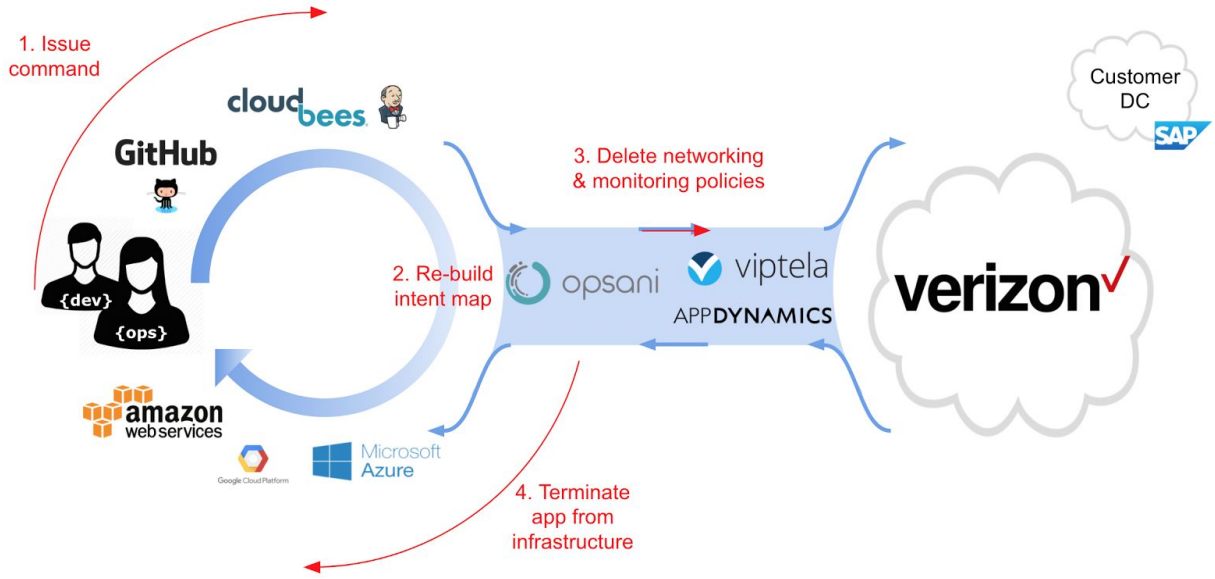
1. Our use case starts with developers checking in code for a new application. With our new DevOps architecture, the first difference is that developers have created a new file, the intent declaration for the application. This includes definitions of application architecture, dependencies, network services, performance requirements, human approvals, and more. The CI/CD system, Jenkins (from CloudBees) builds and tests the application. If everything passes the result is a set of artifacts in the repository.
2. Skopos recognizes the new artifacts, builds an intent map and compiles a plan for the deployment.
3. Skopos uses the Network API(s) to pass the network and monitoring requirements (or service level) to AppDynamics and Viptela. Viptela uses that data to configure the network.
4. Skopos deploys the application to the selected infrastructure. This includes testing, approvals, and possibly rollback if there are errors.
5. Routine monitoring data is reported (exceptions will be dealt with in a separate use case).

What's important about this use case is the network requirements are made a part of the application - checked in right along with code. It shows that developers can now consume network services.

Of note - there is nothing here that implies Skopos understands network topology. It doesn't. However, it does understand the architecture of the application and the dependencies on the network so it can pass network requirements defined by the developers at the appropriate time.

Use Case #2: Terminate an app and its network services

Our second use case is the termination of an app.

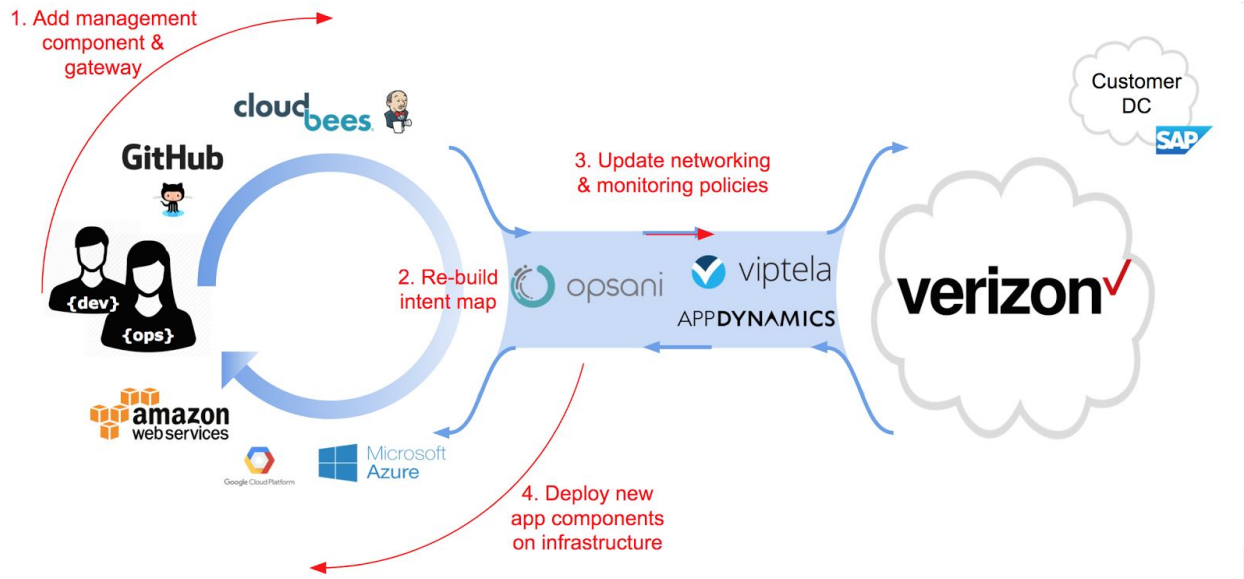


1. The process starts with a developer or operator issuing a command to Skopos to terminate the application.
2. Skopos rebuilds the intent map and compiles a plan.
3. Skopos calls the networking API(s) - AppDynamics and Viptela - to delete the requirements for the app. Ideally, this would result in the app being unable to communicate any longer.
4. Skopos stops the app and deletes it.

What's important about this use case is that when the app is terminated the network services are deleted as well. There are no orphan records left that could create security issues.

Dev and App Change Driven Use Cases

Use Case #3: App changes requiring new network services
Our next use case involves a running application being updated by developers to add new components which require new network services.

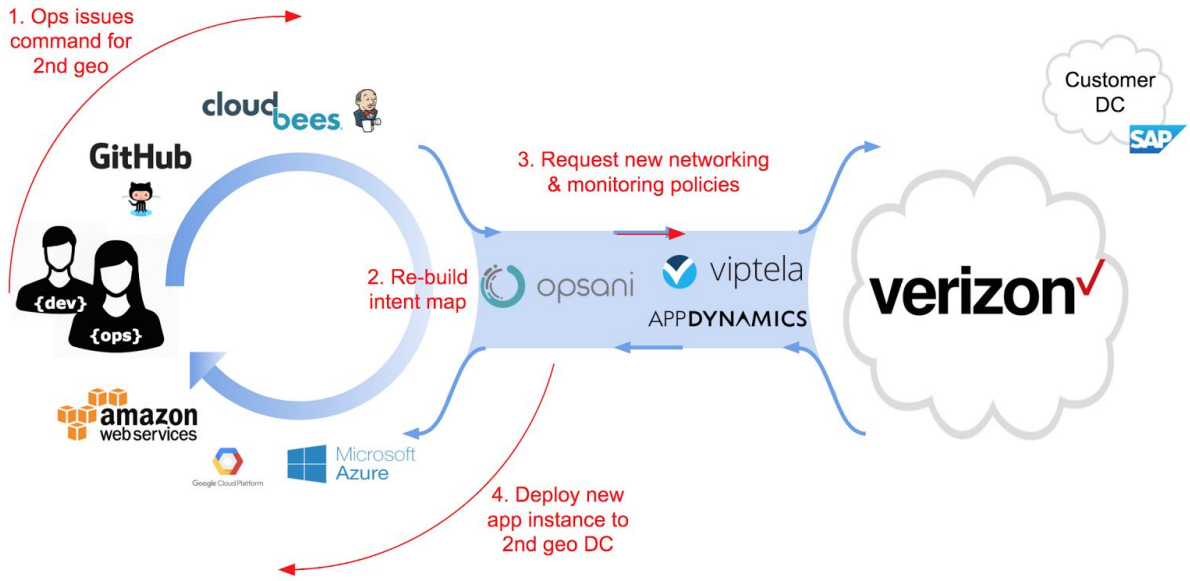


1. The process is started when developers check in code for the new component with an updated network services requirement. The CI/CD system, Jenkins (from CloudBees), builds and tests the new components and if they pass, new artifacts are placed in the repository.
2. Skopos recognizes the new artifacts and rebuilds the intent map.
3. Skopos uses the network API(s) to pass the new requirements to to AppDynamics and Viptela. Viptela uses that data to configure the network and AppDynamics uses it to configure monitoring and thresholds
4. Skopos then deploys the new component to the selected infrastructure.

What's important in this use case is the control the developers have to add network services to their apps in a programmatic way.

Use Case #4: Adding a 2nd Geography and Network Services

Our fourth use case is scaling an application geographically in response to a launch in a new region.



We're assuming an application is running properly and serving customers in another region and a new project is initiated to launch into a 2nd geography.

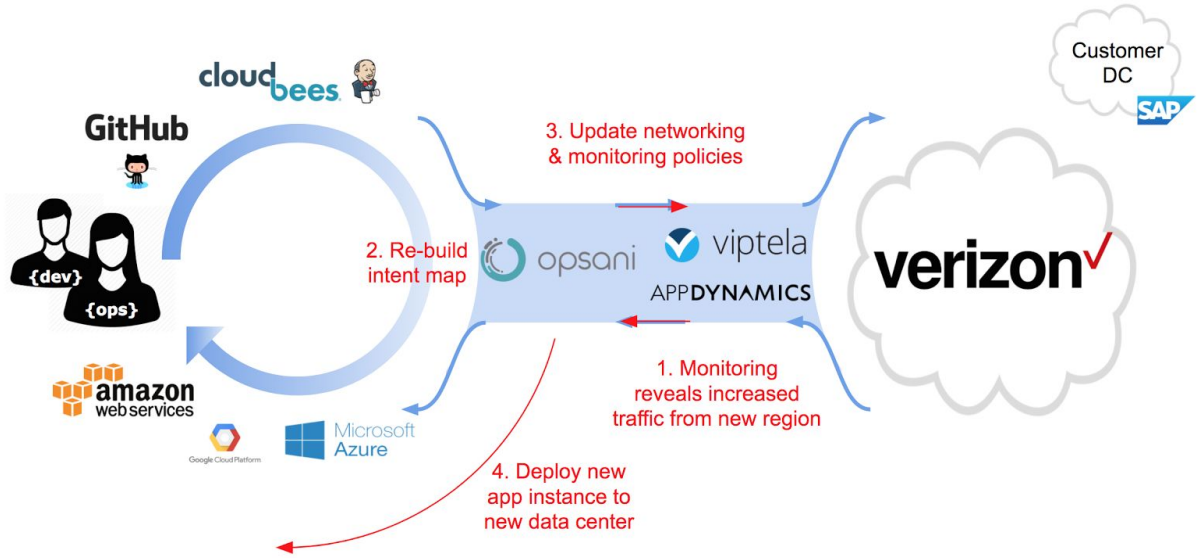
1. Operations team issues a command for the 2nd geography to launch. Since the application is already running in another environment.
2. Skopos recognizes the request, builds an intent map and compiles a plan for the deployment. (Skopos can make a quick copy using the existing approved application 'intent map' or 'model' without calling the full CI/CD pipeline. This information can be checked into GitHub as an approved/trusted app.)
3. Skopos uses the Network API(s) to pass the network and monitoring policies (or service level) to AppDynamics and Viptela. Viptela uses that data to configure the network.
4. Skopos deploys the application to the selected infrastructure. This includes testing, approvals, and possibly rollback if there are errors.
5. Traffic is now sent to the new geography automatically.

What's important here is the ability of the ops team or a project team to quickly deploy new instances of an application along with its network requirements through a single command and without having to re-write or modify the application itself.

Network Driven Use Cases

Use Case #5: Geographic auto-scaling

Our fifth use case is automatically scaling an application geographically in response to user demand.



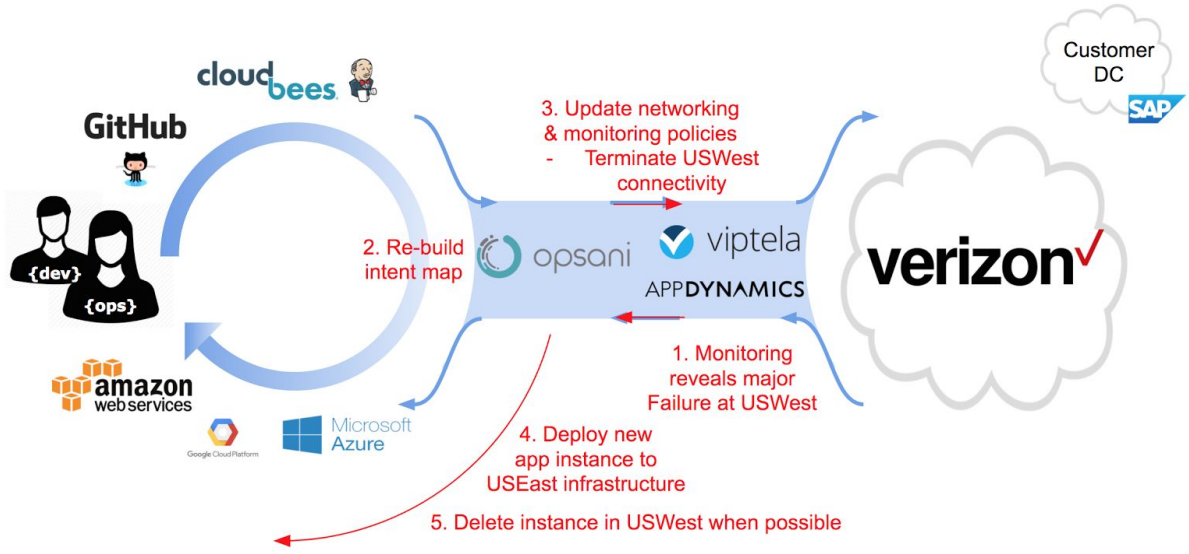
We're assuming an application is running properly and serving customers and that network monitoring (AppDynamics) has been configured to check user experience by geography.

1. Network monitoring (AppDynamics) identifies increased demand in a new geography which can't be served well from the current datacenter. It then sends the data via network APIs to Skopos.
2. Skopos rebuilds the intent map and compiles a reconciliation plan to implement a correction
3. Skopos transfers new network service requirements to Viptela via the network API(s).
4. Skopos deploys a copy of the application to a datacenter in the new geography.
5. Traffic is now sent to the new geography automatically.

What's important here is the ability of the network to provide more than transport, in this case network based monitoring configured for the need of the app, and for coordinated response to be taken based on the monitoring.

Use Case #6: Relocating an app due to DC issues

Our next use case involves relocating an app due to a datacenter issue.

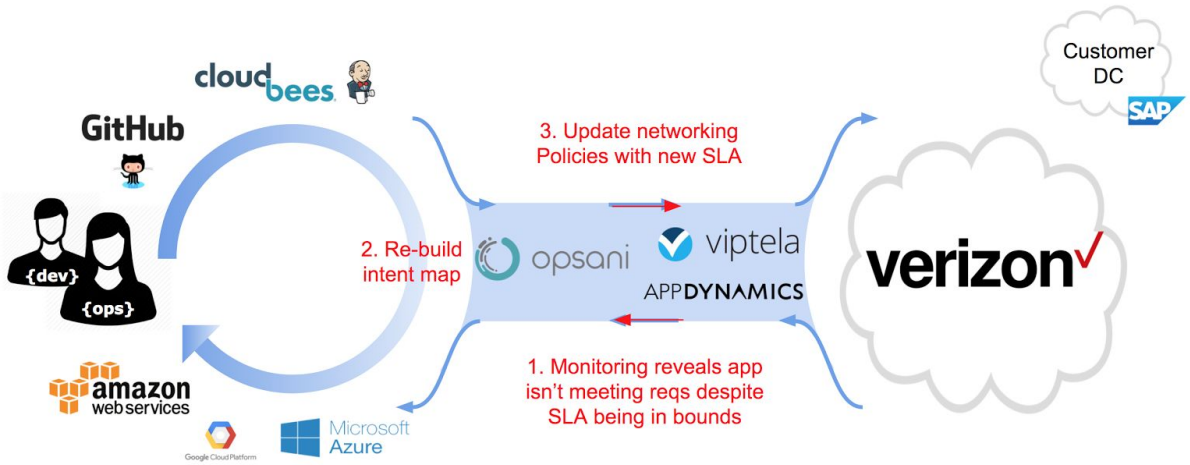


1. Network monitoring (AppDynamics) detects a serious issue with the DC and reports the data to Skopos via API(s)
2. Skopos rebuilds the intent map
3. Skopos eliminates the old network requirements and transfer new requirements via the network API(s) to Viptela.
4. Skopos deploys the new app instance to the new DC.
5. Once connection to the old DC is stable Skopos deletes the old application instance.

What's important in this use case is the ability to respond to an issue identified by network monitoring. In addition, note in step 3 by deleting network services for the old application instance we can isolate it so the orphan doesn't interfere with the operation of the new instance, which is critical for security and reliability.

Use Case #7: Updating SLA Based on Application Monitoring

Our last use case involves an application SLA issue detected and being acted upon.



1. Application monitoring detects a customer experience issue and can isolate it to WAN latency (perhaps between an app instance and a system of record). The SLA is being met, but there's still an issue perhaps because of high utilization that ends up effecting customer experience.
    a. AppDynamics sends a notification of the issue with the connection and SLA specified to Skopos.
2. Skopos rebuilds the intent map which includes a new SLA specification - perhaps lower latency - for the connection.
3. Skopos sends the updated network policy to Viptela
4. Viptela implements and enforces the new SLA

What's important in this use case is the ability to respond to an issue identified by application monitoring specific to customer experience. It's a complete connection between detection of an SLA anomaly and the action to resolve the issue.

Conclusion:

DevNet enables the Verizon network to respond to application needs by providing developers a simple, declarative way to securely control network services for their applications. Customers will vastly improve agility, resulting in new sales opportunities as Verizon becomes a force in the DevOps movement.

Tying together applications and networking services will transform networking in the same way cloud transformed the data center.