

# ARQUITECTURA DE JUNIPER APSTRA

Resuelva problemas complejos del centro de datos gracias a la automatización diseñada especialmente de todo el ciclo de vida de la red

# ÍNDICE

Introducción.....	3
El desafío principal: composición .....	3
El conocimiento es poder: cómo lidiar con el cambio de manera confiable .....	5
Orquestación con estado .....	5
Extensibilidad: adaptabilidad a los futuros cambios tecnológicos de las redes del centro de datos .....	12
Escalabilidad: crecimiento sin obstáculos .....	14
Visión general de la arquitectura de Apstra .....	15
Beneficios de la arquitectura de Apstra .....	17
Conclusión .....	18
Acerca de Juniper Networks.....	18

# RESUMEN EJECUTIVO

Construir, implementar, operar, administrar y solucionar problemas de una red de centro de datos puede ser difícil, costoso y requerir muchos recursos. Según un **informe de Gartner** de 2019, la simplificación de la infraestructura de la TI es la prioridad estratégica número 1 de las empresas en la actualidad. Juniper® Apstra se diseñó especialmente para automatizar el ciclo de vida completo de la red del centro de datos.

En este documento, se describen algunos de los problemas más complejos que enfrentan los equipos de redes del centro de datos en la actualidad y se explica cómo la arquitectura de Apstra aborda cada uno de ellos.

---

## Introducción

Los equipos de arquitectura y operaciones de redes del centro de datos de la actualidad enfrentan cuatro desafíos principales:

- **Composición.** Reunir un sistema coherente y que funcione sin problemas con piezas de infraestructura de diferentes proveedores, cada una con diferentes capacidades, puede ser extremadamente difícil. Juniper® Apstra crea un conjunto integral coherente, un modelo, que consta de toda la información necesaria para implementar el sistema según la intención explícita. El modelo pasa a la infraestructura física y permite que un operador preste servicios confiables y fáciles de consumir.
- **Cambio confiable.** El cambio es un factor constante en el centro de datos. Puede provenir de un operador que intenta agregar nuevos servicios o expandir la capacidad, o de la infraestructura en forma de condiciones de falla. De cualquier manera, los operadores deben encontrar mejores formas de lidiar con el cambio. De acuerdo con diversos estudios realizados en los últimos años<sup>1</sup>, entre el 65 y el 70 % de las interrupciones de la red se deben a errores de configuración humana durante la ejecución de un cambio planificado.
- **Extensibilidad.** Para fomentar la innovación y adaptarse a los inevitables y rápidos cambios en la tecnología, los centros de datos deben poder agregar nuevas capacidades y funcionalidades de manera sencilla y sin problemas. A medida que la tecnología evoluciona, los proveedores innovan y crean nuevas funciones que los operadores aprovechan para mantener su competitividad. Esta evolución provoca un cambio en el tiempo de diseño, lo que crea la necesidad de nuevos contratos de comportamiento, o diseños de referencia, para regir los servicios que se desean implementar. En otras palabras, su composición debe ser extensible para permitir una mayor innovación o abordar nuevos requisitos.
- **Escalabilidad.** Para abordar todo esto a gran escala, las redes del centro de datos deben poder crecer y adaptarse a la innovación y la creciente complejidad.

El objetivo principal de la arquitectura de Apstra es abordar estos problemas directamente.

## El desafío principal: composición

En la actualidad, el principal desafío al que se enfrentan los operadores que tienen en ejecución infraestructura de informática/red/almacenamiento es la composición. La composición es la capacidad de crear un conjunto integral y coherente más grande que la suma de sus componentes funcionales y mantener esa coherencia con el paso del tiempo en presencia de cambios inevitables, a la vez que se prestan servicios de red a los consumidores. En esencia, Apstra está diseñada para estar a la altura del desafío de la composición.

Los centros de datos modernos son computadoras escalables horizontalmente. Necesitan un sistema operativo que brinde una funcionalidad semejante a la que brinda un sistema operativo host en una sola máquina en la actualidad: gestión de recursos y aislamiento de procesos.

La virtualización informática ya logra esto en la informática mediante un solo servidor. Pero, para el procesamiento de centro de datos con escalabilidad horizontal, un operador de centro de datos primero debe componerlo; solo entonces puede proporcionar particiones de recursos.

<sup>1</sup> Entre los ejemplos se incluye: <https://www.computerweekly.com/news/2240179651/Human-error-most-likely-cause-of-datacentre-downtime-finds-study>; <https://www.networkworld.com/article/3142838/top-reasons-for-network-downtime.html>; <https://www.ponemon.org/library/national-survey-on-data-center-outages>.

### ¿Por qué es tan compleja la composición?

Existen dos razones principales por las que la composición es tan desafiante. La primera es que los elementos utilizados para componer un conjunto integral y coherente pueden provenir de diferentes proveedores, tener diferentes capacidades o solo estar disponibles a través de diferentes API.

En segundo lugar, es posible que deba componer su infraestructura para brindar una gran variedad de servicios, cada uno con múltiples aspectos funcionales, como la accesibilidad, la seguridad, la calidad de la experiencia y la disponibilidad.

A medida que crea instancias de múltiples servicios, las interacciones entre estos componentes pueden causar interrupciones en el servicio, a menos que comprenda bien cómo asignar estos servicios a los mecanismos de cumplimiento. A medida que las capacidades de su infraestructura evolucionan con el tiempo, debe poder aprovechar e introducir innovaciones sin infringir los sistemas existentes.

### Apstra aborda el desafío con diseños de referencia

El concepto clave que utiliza la arquitectura de Apstra para abordar el desafío de la composición es el diseño de referencia. Un diseño de referencia es un contrato de conducta en que se define cómo la intención empresarial del usuario se asigna a los mecanismos de cumplimiento y qué expectativas se deben satisfacer para que la intención se considere cumplida.

Los diseños de referencia rigen lo siguiente:

- Las funciones y las responsabilidades de los componentes físicos y lógicos
- Cómo se asignan los servicios a los mecanismos de cumplimiento
- Las expectativas que se deben cumplir (es decir, situaciones que se deben advertir)

En un diseño de referencia, las funciones y las responsabilidades de los elementos físicos y lógicos están bien definidas, lo que, a su vez, une el alcance del modelado y permite la especificación de un modelo mínimo, pero completo. Un diseño de referencia también rige cómo se asigna la intención en los mecanismos de cumplimiento. La comprensión de esta asignación permite la automatización de la resolución de problemas y proporciona análisis eficientes basados en el conocimiento de cómo se compone el sistema, en lugar de la ingeniería inversa de lo que sucede en su red.

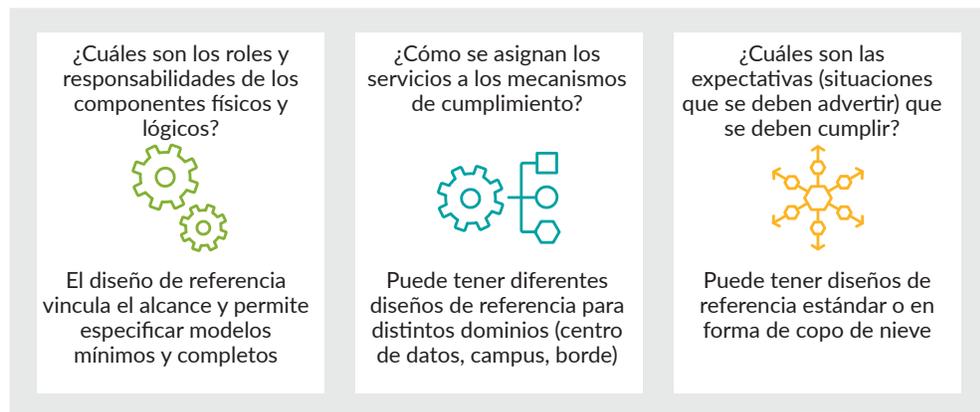


Figura 1: El diseño de referencia es un contrato de comportamiento rector.

En diferentes diseños de referencia, la misma intención puede imponerse con mecanismos diferentes, posiblemente más recientes e innovadores. La comprensión de esta asignación permite la identificación de la causa raíz, que dirige a los operadores a la causa de los problemas que afectan el servicio.

El diseño de referencia también especifica las expectativas que se deben validar. Por ejemplo, un diseño de referencia puede estipular que cada leaf tenga una sesión de protocolo de puerta de enlace de frontera con cada spine. De esta manera, las sesiones de protocolo de puerta de enlace de frontera perdidas e inesperadas se identifican fácilmente.

Definir un diseño de referencia es una actividad de valor agregado que debe completar un experto en redes. Con Apstra, la experiencia es una parte explícita del sistema en lugar de algo que vive solo en la mente de un experto. Esto permite que la experiencia se modele e inserte en el sistema, en lugar de estar codificada de forma rígida.

## El conocimiento es poder: cómo lidiar con el cambio de manera confiable

El cambio es una constante de los centros de datos, por lo que lidiar con él de manera confiable es siempre una prioridad. El cambio proviene de dos partes:

1. **La intención del operador.** Es posible que un operador quiera agregar un nuevo servicio (como una red virtual), agregar o eliminar un recurso de la infraestructura o realizar un cambio en algunas políticas. El operador tiene control sobre un cambio a medida que se inicia. Apstra utiliza la **orquestación con estado** para habilitar de manera confiable la intención de un operador.
2. **Fallas en una infraestructura administrada.** El cambio también puede provenir de la infraestructura administrada en forma de fallas (p. ej., caídas excesivas de paquetes o desequilibrios de tráfico). Un operador no tiene control sobre este cambio y el volumen de condiciones operativas suele ser abrumador. Apstra proporciona análisis basados en la intención para ayudar a los operadores a lidiar con los cambios en el estado operativo.

Tratar con ambos tipos de cambios de manera confiable depende de su conocimiento del estado de su infraestructura. Se deben cumplir dos condiciones:

1. El conocimiento debe interpretarse en contexto, es decir, se debe comprender la relación entre las condiciones y las expectativas, como se describe en la sección **Modelo de contexto**.
2. El conocimiento debe ser oportuno, es decir, debe reflejar las condiciones actuales, como se describe en la sección **Monitoreo y notificación en tiempo real**.

## Orquestación con estado

La orquestación con estado permite que los cambios que provienen del operador sean confiables mediante la intención. En la Figura 2, se puede ver un flujo de acciones de orquestación con estado.

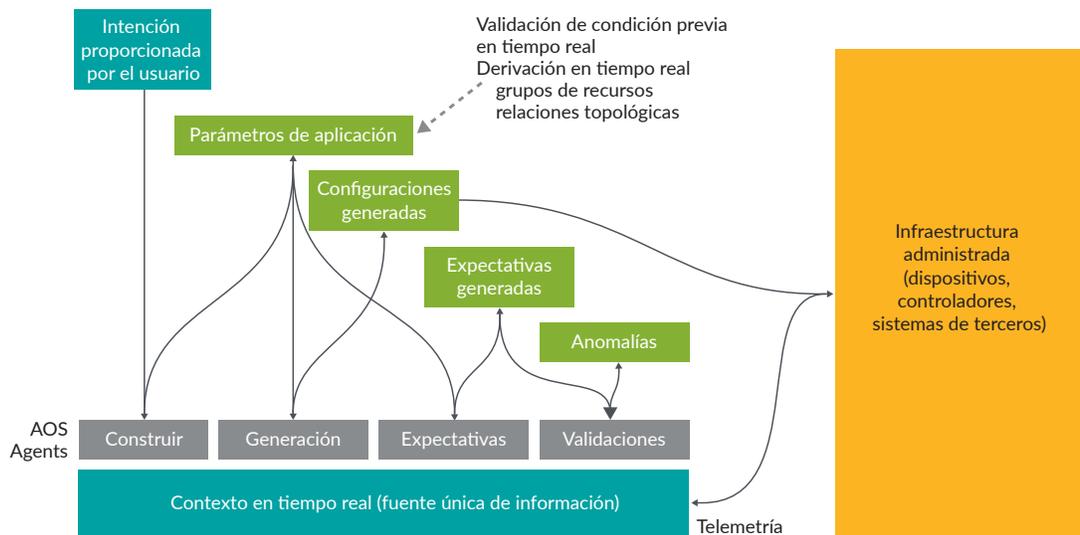


Figura 2: Flujo de orquestación con estado

Si desea realizar un cambio independiente de su sistema completamente funcional existente, la orquestación con estado le permite al usuario proporcionar solo la intención del cambio. Esto se hace de forma independiente a la implementación, lo que permite que la especificación de la intención sea más simple y menos propensa a errores. Un conjunto completo de pasos ejecutados durante la orquestación con estado incluye lo siguiente:

*“No es la especie más fuerte la que sobrevive, ni la más inteligente, sino la que responde mejor al cambio”.*

- Charles Darwin, 1809

**Paso 1: validación de la condición previa en tiempo real.** ¿Esta nueva solicitud infringirá algunas políticas? Por ejemplo, ¿tiene permiso para crear esta red virtual o esto creará algunas brechas de seguridad? ¿O puede poner un dispositivo en modo de mantenimiento sabiendo que otros dispositivos ya están fuera de línea y desconectar este dispositivo hará que su sistema quede vulnerable? Con la orquestación con estado, todas estas preguntas se formulan y validan en tiempo real con el gráfico de contexto.

**Paso 2: derivación de parámetros de aplicación en tiempo real.** Para implementar la intención, se deberán suministrar y activar parámetros específicos de los mecanismos de cumplimiento apropiados en elementos administrados específicos. El uso de Apstra de un diseño de referencia para explicar cómo se implementará la intención, junto con su comprensión del contexto actual, permite que el sistema realice automáticamente la derivación de parámetros de aplicación en tiempo real. No hay posibilidad de que un operador inserte accidentalmente un comando, una interfaz, una dirección IP o una VLAN incorrectos.

**Paso 3: implementación de configuraciones múltiples en tiempo real.** Apstra realiza la implementación de múltiples ajustes en tiempo real, lo que automatiza la implementación de la configuración en posiblemente múltiples elementos que pueden tener API específicas de diferentes proveedores y tecnologías.

**Paso 4: generación de expectativas en tiempo real.** Un diseño de referencia que actúa como un contrato de comportamiento le permite a Apstra generar expectativas en tiempo real, las cuales describen las condiciones que se deben cumplir para declarar que el resultado ha cumplido cabalmente con la intención.

**Paso 5: validación de expectativas en tiempo real.** Luego, Apstra activa una recopilación de telemetría y análisis operativos habilitados por contexto para realizar la validación de expectativas en tiempo real.

**Paso 6: resultado de servicio validado.** Al final de este proceso, el usuario puede observar un resultado de servicio validado en términos inequívocos. Cada cambio en la intención o el estado operativo previsto se refleja en el modelo de contexto en tiempo real, y cualquier componente que deba ser consciente del cambio también recibe una notificación al respecto en tiempo real. Apstra extrae el conocimiento pertinente de los datos operativos sin procesar mediante **análisis basados en la intención**.

Con la orquestación sin estado, faltan muchos de los pasos (como se muestra en la Figura 3). El resultado de la implementación de la configuración es de lo que usualmente se trata la orquestación sin estado: migrar la configuración a posibles múltiples sistemas y validar que los elementos administrados hayan aceptado las configuraciones. Y en este punto normalmente se detiene la orquestación sin estado; no hay noción de expectativas de servicio o validación de que estas expectativas se cumplan. La validación automatizada del resultado del servicio está ausente y, en cambio, los sistemas separados le presentan al usuario un punto de observación único con respecto al estado operativo sin procesar. Depende del usuario revisar visualmente para averiguar si se logró el resultado. Este punto de observación único generalmente incluye demasiada información y carece de un contexto específico de implementación, lo que hace que la detección visual sea muy difícil y subjetiva.

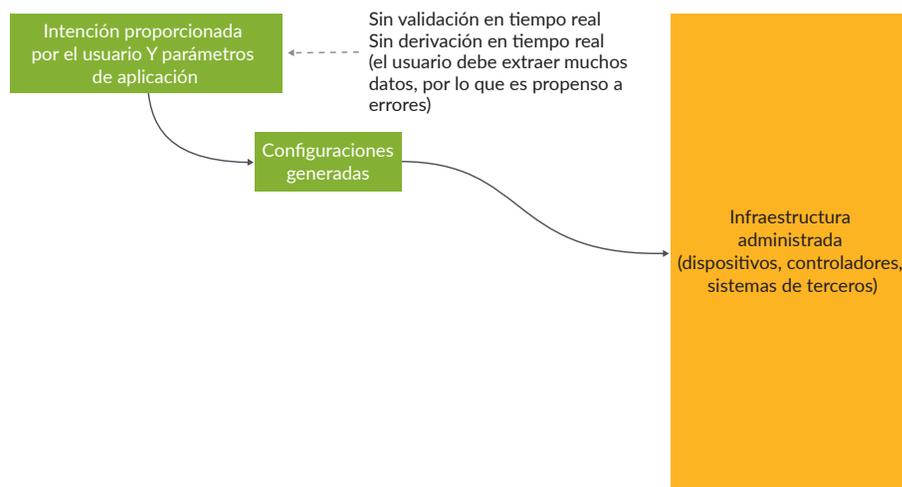


Figura 3: Orquestación sin estado típica

## Análisis basado en la intención

El análisis basado en la intención (IBA) ayuda a los operadores a lidiar con los cambios de estado operativo en su infraestructura mediante la extracción de conocimiento de los datos de telemetría sin procesar. Como se mencionó anteriormente, debe tener un contexto que pueda consultar en tiempo real antes de poder usar un IBA.

La extracción de conocimiento consta de dos pasos:

- 1. Detección de condiciones de interés** (situaciones que se deben advertir). La detección de condiciones se realiza mediante sondas de IBA.
- 2. Automatización de clasificación de condiciones de interés y derivación de sus relaciones.** Las condiciones varían en su contenido semántico. En otras palabras, algunas de ellas son más importantes que otras. Es vital comprender las relaciones entre las condiciones para que pueda identificar condiciones procesables importantes (causas raíz) y comprender qué condiciones son meras consecuencias que desaparecerán cuando se aborden las causas raíz importantes. La clasificación de condiciones y la derivación de causalidad se realizan mediante el componente de identificación de la causa raíz del IBA.

## Modelado de condiciones de interés

IBA modela las condiciones de interés mediante cuatro categorías: anomalía, síntoma, impacto y causa raíz.

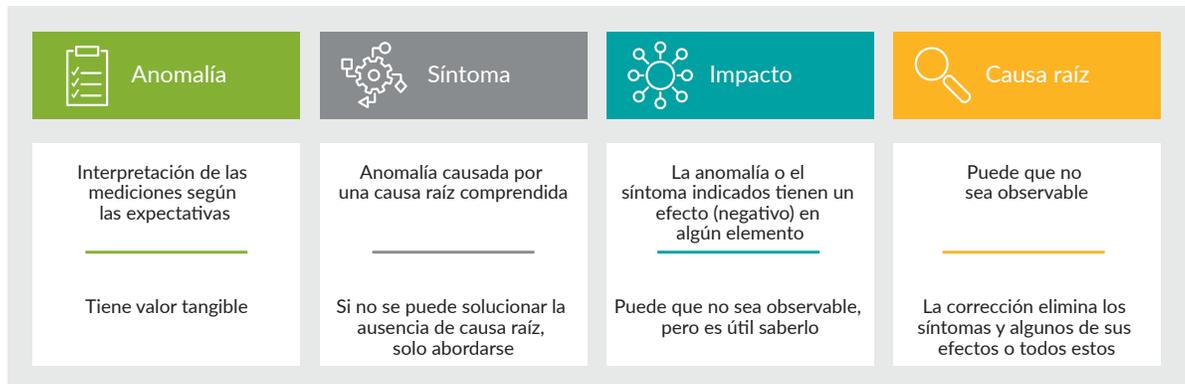


Gráfico 4: Categorización de condiciones

### Anomalías

Las anomalías representan esencialmente una interpretación de la medición, o algún agregado de esta, frente a alguna expectativa y, como tal, tienen un valor más tangible que una simple medición.

### Síntomas

Los síntomas son anomalías causadas por una causa raíz que se entiende bien. Por lo general, se observan con facilidad, pero no se pueden reparar, sino solo tratar, como dar una aspirina a un paciente para tratar la fiebre, pero no la enfermedad en sí. Si no trata la enfermedad, la fiebre regresará. Es importante destacar que un síntoma desaparecerá cuando se solucione la causa raíz, por lo que los síntomas no son procesables; simplemente son útiles para diagnosticar la causa raíz.

### Impactos

Los impactos indican que algo se produjo como resultado de una anomalía. Es posible que los impactos no siempre sean observables, pero es útil conocerlos. Por ejemplo, es posible que desee saber que un cliente importante se verá afectado por una falla en un dispositivo o una pérdida excesiva de paquetes antes de que lo llamen para reclamar.

Cuando los impactos no son observables, se pueden calcular con base en el conocimiento de la intención y los mecanismos de aplicación utilizados para implementar la intención. Comprender los impactos ayuda a los operadores a priorizar qué causas raíz y anomalías tienen el mayor efecto en la operación.

### Causas raíz

Las causas raíz son las más importantes de todas las condiciones. Es posible que no siempre sean observables, lo que dificulta su diagnóstico, pero causan muchos síntomas e impactos. Son procesables, ya que, si se soluciona la causa raíz, se eliminan los síntomas y los impactos asociados.

### Sondas de IBA: Con el conocimiento obtenido

Hay una cita famosa que se aplica a las redes de centros de datos: “Lo que te mete en problemas no es lo que sabes, sino lo que crees saber, pero no es verdad”. La intención caracteriza y formaliza lo normal: lo que sabemos. Mediante el IBA, nos aseguramos de que lo que sabemos es, de hecho, como debería ser.

*“Lo que te mete en problemas no es lo que sabes, sino lo que crees saber, pero no es verdad”. -Mark Twain*

Las sondas de IBA son responsables de detectar condiciones de interés y están impulsadas por un contrato de comportamiento que forma parte del diseño de referencia. Las sondas de IBA obtienen datos, aplican algunas tareas de procesamiento y, luego, comparan el resultado con las expectativas. Una sonda de IBA es esencialmente una canalización de procesamiento de datos configurable que les permite a los usuarios establecer condiciones de interés (es decir, situaciones que se deben advertir).

### Fuentes de datos y consultas

Las etapas iniciales de una sonda suelen ser fuentes de datos responsables de obtener los datos de telemetría sin procesar. La configuración de la etapa de fuente de datos incluye una consulta que puede expresar con precisión qué datos recopilar, lo que es una característica muy potente.

Por ejemplo, supongamos que un operador está interesado en analizar un desequilibrio de enrutamiento multirruta de igual costo (ECMP) que se aplica solo a las interfaces de estructura, y hubo informes de que una versión específica del sistema operativo introdujo un error en el algoritmo hash de ECMP. Una consulta puede expresar la necesidad de recopilar contadores de interfaz en cada conmutador de la parte superior del bastidor, pero solo para las interfaces de la estructura y los conmutadores que ejecuten una versión particular x.y.z del sistema operativo del conmutador. Ahora que esta situación que debe advertir está configurada, el operador no debe preocuparse por los cambios. Si se agrega un nuevo enlace de estructura a un interruptor que coincida con los criterios, se incluirá automáticamente en el análisis. Si se agrega un nuevo conmutador, se incluirá automáticamente. Si alguien actualiza otro conmutador a la versión x.y.z, se incluirá automáticamente. La sonda de IBA no requiere mantenimiento.

### Procesadores de etapas

En muchas situaciones, el operador no está interesado en valores instantáneos de datos de telemetría sin procesar, sino en una agregación o tendencias. El IBA contiene procesadores de etapas que agregan información, como el cálculo del promedio, los valores mínimo/máximo, la desviación estándar, etc. Luego, un operador puede comparar estos agregados con las expectativas para identificar si la métrica agregada está dentro o fuera de un rango específico, en cuyo caso se marca una anomalía.

Luego, es posible que el operador desee verificar si esta anomalía se mantiene durante un período que exceda un umbral específico y marcar la anomalía solo cuando se exceda el umbral para evitar marcar anomalías por condiciones transitorias o temporales. El operador puede lograr esto simplemente configurando una etapa posterior que contenga lo que se denomina como procesador de tiempo en estado.

### Más allá de los datos numéricos

Las sondas no funcionan solo con datos numéricos. Por ejemplo, se pueden utilizar para validar la exactitud del plano de control y datos. Un operador puede, por ejemplo, configurar un procesador de fuente de datos con una consulta para construir una tabla de enrutamiento o reenvío prevista, según la intención.

En un entorno de Ethernet VPN (EVPN), la intención contendrá información como las redes virtuales que existen, dónde están sus puntos de conexión, e información sobre los mecanismos de aplicación. Esta tabla prevista se puede comparar con la que proviene de la telemetría, y las anomalías se marcan cuando se encuentra una incongruencia.

También se puede configurar una sonda para rastrear cambios repentinos en los tamaños de las tablas de enrutamiento y reenvío, y alertar cuando las tendencias no son las previstas. El umbral de "lo previsto" también se puede calcular dinámicamente a partir de la intención, ya que puede ser una función de la cantidad de redes virtuales, la cantidad de puntos de conexión, la cantidad de puntos de conexión de túnel virtual (VTEP), la cantidad de VTEP con algunos problemas, etc. Las posibilidades son infinitas.

Las sondas se pueden crear, activar y desactivar de forma declarativa con una simple llamada a la REST o mediante el uso de una GUI. La activación de la sonda también sirve como desencadenante de la telemetría. En otras palabras, los datos de telemetría específicos se recopilan solo si hay una sonda interesada en estos, por lo que las sondas sirven esencialmente como un mecanismo de configuración de recopilación de telemetría. Las consultas en los procesadores de fuentes de datos permiten que la configuración sea tan detallada y precisa como sea necesario, lo que evita el "desorden por la acumulación de datos", que ocurre cuando se recopilan toneladas de datos sin tener una idea clara de qué hacer con ellos. Esto, a su vez, eleva por las nubes el costo de almacenar y procesar datos.

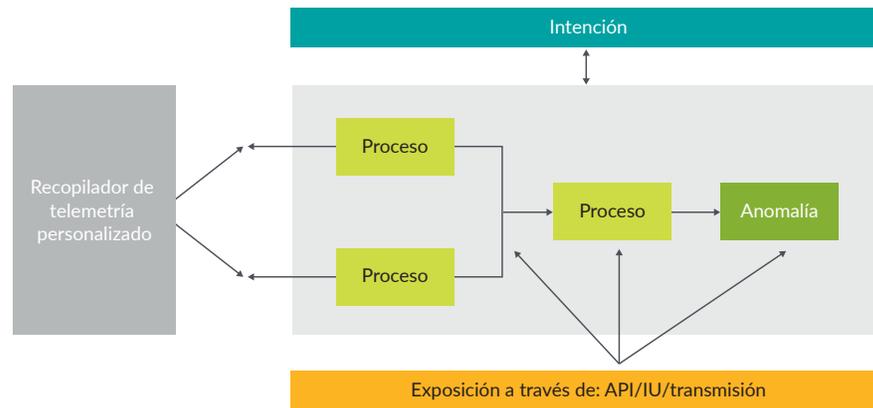


Figura 5: Recopilación de telemetría

Una vez que se configura una sonda, el resultado de cada etapa está disponible a través de la API, la GUI y los puntos de conexión de transmisión. Apstra tiene un conjunto de sondas integradas, y hay un repositorio de sondas de código abierto en GitHub. El usuario también puede crear nuevas sondas desde cero.

### Identificación de la causa raíz

La identificación de la causa raíz (RCI) es un mecanismo para automatizar la clasificación y la derivación de las relaciones de causalidad entre las condiciones identificadas en la infraestructura. El principal beneficio de la RCI es develar las condiciones de causa raíz que requieren la acción del operador de la gran cantidad de condiciones no procesables que son meras consecuencias de fallas de causa raíz. Por ejemplo, supongamos que un operador ha equipado bien la infraestructura y está observando un registro de condiciones anómalas en una consola de "punto de observación único" (consulte la Figura 6). Los puntos rojos indican diferentes tipos de condiciones esparcidas por la infraestructura que ocurren en muchos elementos diferentes.

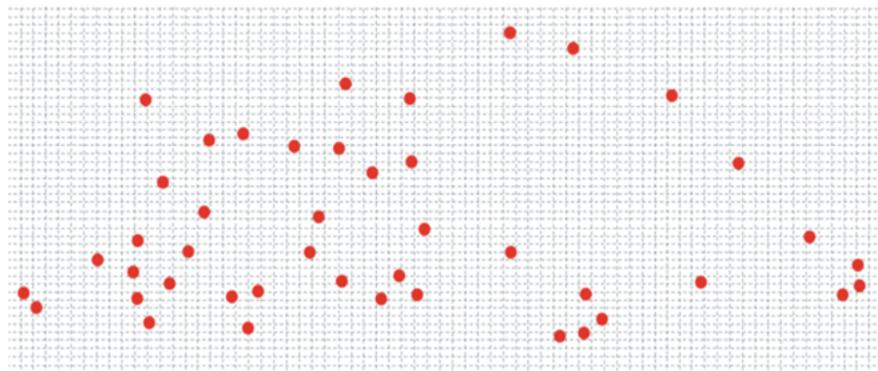


Figura 6: Consola de punto de observación único

El problema con esta imagen es que no es procesable; es simplemente algo inentendible. En ausencia de una clasificación, cada condición compite por la atención del operador, pero ¿en cuál se debe actuar? En este punto entra en juego la RCI. Con el conocimiento del contexto, que indica cuál es el servicio, cómo se implementa y asigna en los mecanismos de aplicación, y cómo se relacionan los elementos en la infraestructura administrada, la RCI identifica y clasifica las causas raíz y los síntomas e impactos relacionados. Como se muestra en la Figura 7, los resultados del análisis que utilizan la RCI son los siguientes:

- La causa raíz fue una fuga de memoria en un conmutador llamado “spine\_1” (causa raíz: gran punto rojo).
- Esta fuga de memoria hizo que el eliminador de procesos quedara sin memoria y actuara y eliminara algunos procesos, uno de los cuales era un proceso de enrutamiento (síntomas: puntos en azul oscuro).
- Esto, a su vez, provocó que las sesiones de BGP fallaran en este y en los dispositivos de emparejamiento, además de que faltaran las entradas esperadas de la tabla de enrutamiento (síntomas: puntos en azul oscuro).
- Como resultado, los puntos de conexión pertenecientes a los clientes X e Y experimentaron problemas de conectividad (impactos: puntos en celeste).

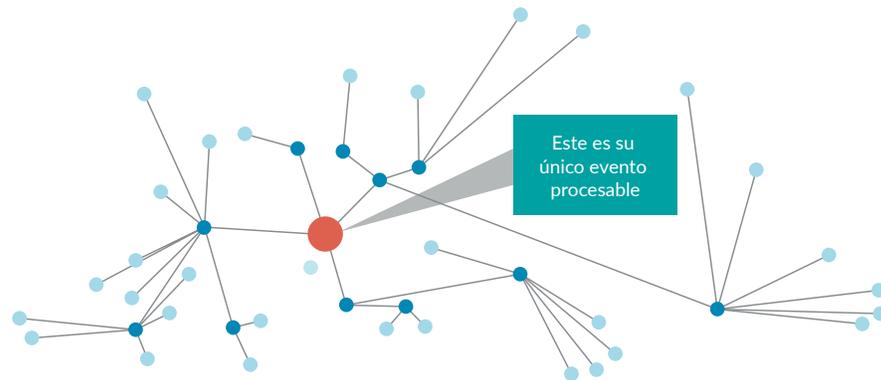


Figura 7: Consola de identificación de la causa raíz

La RCI automatiza el complejo proceso mental involucrado en un “punto de observación único”, que sobrecarga a los operadores con montañas de información y los responsabiliza de realizar la detección visual y la correlación. En cambio, la RCI le presenta al operador un punto de observación simple, que solo identifica las acciones necesarias.

### Modelo de contexto

Tener todos los datos no significa que tenga todas las respuestas. Y recopilar grandes cantidades de datos sin automatizar la extracción de conocimiento es una invitación a una explosión de costos operativos, ya que sus expertos dedicarán su valioso tiempo a trabajar para que estos datos tengan sentido. Los datos en sí tienen un valor limitado, a menos que le brinden conocimiento o respondan las preguntas correctas.

Los datos de telemetría sin procesar generalmente se almacenan en almacenes de valor clave, que son adecuados para el escalamiento horizontal y el particionamiento. Pero no admiten consultas que no sean búsquedas clave simples. Para extraer conocimiento, necesitaría desarrollar soporte para consultas y tener un contexto a fin de estructurar estas consultas.

Por otro lado, los almacenes de datos SQL sí admiten consultas complejas. El problema es que las tablas SQL están diseñadas en torno a consultas anticipadas, que se crean durante el diseño. Si desea hacer alguna otra pregunta durante el tiempo de ejecución, puede desnormalizar la base de datos, o las consultas pueden terminar con muchas uniones, lo que hará que el rendimiento se detenga. Pero la extracción de conocimiento operativo tiene que ver con consultas que surgen durante el tiempo de ejecución. En este punto se destaca la implementación de datos contextuales basada en gráficos, que admite consultas complejas y arbitrarias con un gran número de “uniones” durante el tiempo de ejecución. El modelo de gráfico también se puede ampliar de forma fácil y sencilla si se agregan más instancias de bloques de construcción básicos, nodos y relaciones.

Echemos un vistazo más de cerca a lo que ocurre en este contexto, ya que ayudará a explicar el poder que desaten las consultas.

- **Errores de diseño:** primero, el contexto del gráfico de intenciones contiene errores de diseño. Por ejemplo, si está diseñando una red del centro de datos, puede contener el factor de sobresuscripción deseado, la cantidad deseada de servidores y la configuración de alta disponibilidad (HA) deseada (conexión única o doble).
- **Asignación de recursos:** contiene decisiones relacionadas con la asignación de recursos. Rige si está administrando su infraestructura como **“ganado”** o **“mascotas”**.
- **Políticas de aislamiento:** esto también especifica las políticas de aislamiento. ¿Está permitido reutilizar ciertos recursos (direcciones IP, ASN, VLAN)?
- **Políticas de segmentación:** contiene políticas de segmentación que especifican qué puntos de conexión y cargas de trabajo pueden comunicarse entre sí y en qué condiciones.
- **Información de aplicación:** contiene información de aplicación sobre la ubicación (en qué dispositivos físicos o virtuales) y la forma (mediante qué mecanismos) con que se implementaron realmente las políticas de accesibilidad y segmentación.
- **Políticas externas de accesibilidad:** en el caso del centro de datos, también puede contener políticas de accesibilidad externa, que especifiquen qué puntos de conexión internos pueden ser vistos por el mundo exterior y viceversa. También contiene qué inquilinos son propietarios de qué segmentos, los niveles de servicio prometidos a cada uno de ellos y los objetivos de nivel de servicio.
- **Perspectiva empresarial:** aquí se abarca la perspectiva comercial, como los acuerdos de nivel de servicio y cuánto le costará si no los cumple.

#### Vista en 8-D

Aunque todos estos errores estén representados en un único gráfico, esa única fuente de información es lógicamente un único espacio multidimensional, con al menos las siguientes ocho dimensiones (derivadas de manera específica de este ejemplo; y podría haber más):

- Diseño
- Recursos
- Aislamiento
- Segmentación
- Cumplimiento
- Accesibilidad externa
- Niveles de servicio
- Perspectiva empresarial

Como resultado, cuando las cosas salen mal, puede hacer que una sola consulta opere en esta vista en 8-D y responda preguntas complejas como las siguientes: “Dada la condición de falla reciente en un elemento, ¿siguen siendo válidos mis objetivos de diseño?” “¿Se utilizan los recursos correctos en los puntos de aplicación correctos con las políticas de aislamiento deseadas?” “¿Afecta de alguna manera la política de segmentación?” “¿Cumple con el objetivo de nivel de servicio?” “¿Qué precio pagaré por esta falla?”

La existencia de esta vista en 8-D no es “agradable”, pero es un requisito para un funcionamiento confiable. Con el sistema basado en intenciones de Apstra, la vista en 8-D está integrada. Sin Apstra, debe reconstruir esta vista en 8-D con una capa muy compleja, que abarca múltiples fuentes de información y conocimiento interno que existen en las mentes de sus expertos. Construir esta capa, en un entorno en que las fuentes individuales de información no se desarrollaron teniendo en cuenta la integración y tienen diferentes semánticas y comportamientos, es un trabajo pesado indiscriminado en el mejor de los casos y una pesadilla inmanejable en el peor de los casos.

### Monitoreo y notificación en tiempo real

El objetivo principal de Apstra es extraer conocimiento sobre la intención y el estado operativo resultante de su infraestructura. Este conocimiento es poder si es oportuno, es decir, si refleja las condiciones actuales. En la esencia de Apstra se encuentra la capacidad de configurar consultas en tiempo real, que les permiten a los clientes suscribirse a condiciones de interés y recibir notificaciones en tiempo real cuando se cumplen las condiciones. Las condiciones en este contexto están relacionadas tanto con la intención como con el estado operativo. Este es un requisito absoluto para lidiar con los cambios de manera confiable.

Como verá en la sección [Visión general de la arquitectura](#), el mismo paradigma de publicación/suscripción que se presenta a nivel de usuario es compatible con un mecanismo equivalente de bajo nivel de publicación/suscripción. Este mecanismo lo implementa el almacén de datos distribuidos que actúa como un canal de comunicación lógica centrado en datos para los procesos del sistema Apstra que implementan la lógica de la aplicación.

### Operación automática

Por último, pero no menos importante, existe la necesidad de automatizar la reacción ante algunos eventos para remediar problemas, registrarlos para el análisis forense o realizar un desglose del siguiente nivel para recopilar telemetría que ayude con el análisis de la causa raíz.

Una reacción automatizada:

- Reduce el riesgo, ya que los parámetros de corrección se derivan automáticamente de la única fuente de información actualizada y no están sujetos a una mala configuración ni a datos obsoletos
- Mejora la experiencia del cliente, ya que la corrección ocurre de manera oportuna
- Reduce el costo operativo, ya que evita la necesidad de resolución de problemas manual y ejecución manual de la guía de correcciones

### *Los obstáculos para una red autónoma surgen más de la resistencia organizacional o individual que de las limitaciones tecnológicas, ya que la funcionalidad requerida ya existe.*

En última instancia, una reacción automatizada permite la operación y reparación automáticas de la red. Los obstáculos para una red autónoma surgen más de la resistencia organizacional o individual que de las limitaciones tecnológicas, ya que la funcionalidad requerida ya existe.

## Extensibilidad: adaptabilidad a los futuros cambios tecnológicos de las redes del centro de datos

La extensibilidad tiene tres dimensiones:

1. **Extensibilidad del diseño de referencia.** Esto rige cómo las piezas de la infraestructura funcionan en conjunto para cumplir con la intención. Incluye complementos que permiten modificaciones del modelo gráfico, así como modificaciones relacionadas con la forma en que se asigna la intención a los mecanismos de aplicación en la infraestructura.
2. **Extensibilidad de análisis.** Esto está relacionado con la definición de nuevas condiciones o situaciones que se deben advertir y le permite al usuario definir nuevas validaciones, así como clasificarlas y relacionarlas.
3. **API de servicio flexible.** Proporcionan la capacidad de ampliar las definiciones de servicio de nivel superior.

### Diseño de referencia

Como se mencionó anteriormente, un diseño de referencia es un contrato de conducta en que se define cómo la intención se asigna a los mecanismos de aplicación y cuáles expectativas se deben satisfacer para que la intención se considere cumplida. Cualquier aspecto de este contrato se puede modificar o ampliar. Se pueden definir nuevos tipos de relaciones y nodos, y se pueden modificar las plantillas de configuración y los mecanismos de asignación de recursos.

## Extensibilidad del análisis

La nueva funcionalidad de análisis se puede introducir a través de dos mecanismos:

1. Se pueden definir **nuevas sondas de IBA** que detecten nuevas condiciones de interés. Pueden incluir nuevos recopiladores de telemetría, así como canalizaciones de procesamiento de datos de condiciones específicas. Las sondas también se pueden publicar y, luego, importar desde un repositorio público.
2. Los **nuevos modelos de RCI** se pueden definir y cargar en un sistema. Un modelo de RCI es esencialmente una asignación de un nuevo tipo de causa raíz a un conjunto de síntomas que crea. Una vez que se define y carga este modelo, el sistema Apstra automatiza la identificación de la causa raíz en función de los síntomas observados.

## API de servicio

Apstra expone las API de nivel de servicio en forma de políticas basadas en grupos, a fin de brindar la flexibilidad para admitir una amplia gama de servicios y políticas de manera independiente de la implementación.

Con las políticas basadas en grupos, la intención se expresa como un gráfico que representa los puntos de conexión que se colocan en grupos (relaciones de miembros) con el fin de expresar la intención de un comportamiento común. Las políticas se resuelven en instancias y se relacionan con grupos o puntos de conexión individuales para definir ese comportamiento.

Las políticas se pueden relacionar con un grupo de manera direccional (relación de remitente o destinatario) o no direccional (que se aplica). Las políticas son una recopilación de reglas. Las reglas pueden tener una relación de continuación cuando el orden entre las reglas es importante. Los grupos pueden estar compuestos por otros grupos (jerarquía). Los grupos también pueden tener una relación con otros grupos para expresar algunas restricciones (como “estos puntos de conexiones/grupos están detrás de estos grupos de puertos”). Los puntos de conexión, los grupos, las políticas y las reglas se pueden considerar como bloques de construcción para expresar la intención de conectividad.

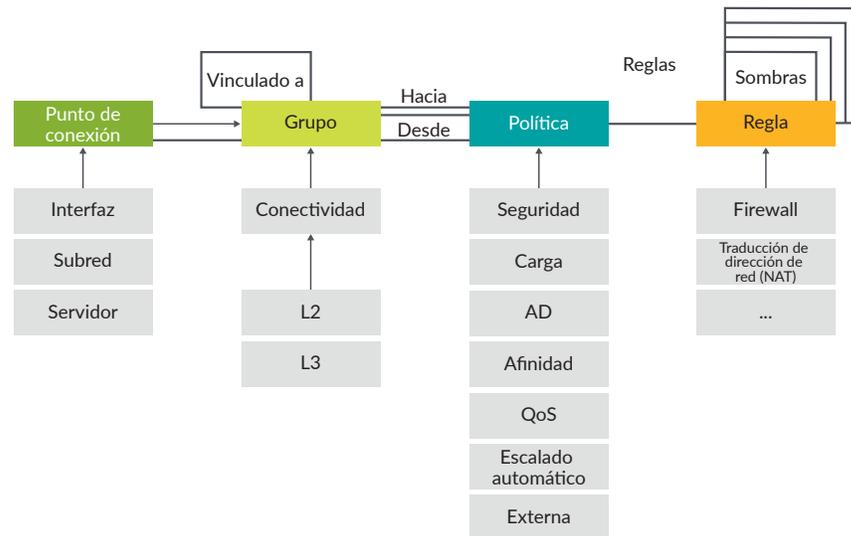


Figura 8: Políticas basadas en grupos

Los puntos de conexión son elementos de su infraestructura sujetos a políticas y, como tales, son bastante generales en su construcción. Pueden representar una interfaz (física o virtual), un servidor, una máquina virtual (VM), un contenedor o un punto de conexión de la aplicación. Las flechas azules en la Figura 8 indican una herencia “lógica”. Los puntos de conexión contienen parámetros que los definen con mayor precisión (p. ej., nombre de interfaz, número de puerto, número de serie y nombre de host, UUID de VM, dirección IP de contenedor, protocolo de aplicación, número de puerto UDP/TCP). Los puntos de conexión también pueden representar elementos no administrados por Apstra (puntos de conexión externos) y, como tales, se utilizan para expresar restricciones de los sistemas externos con los que Apstra necesita interactuar (p. ej., IP/ASN de enrutador externo).

Los puntos de conexión se colocan en grupos. Cualquier punto de conexión dado puede ser miembro de varios grupos, cada uno de los cuales expresa diferentes aspectos del comportamiento previsto. Por ejemplo, un grupo podría indicar lo siguiente:

- Un conjunto de servidores en la región “Este”
- Un grupo de servidores que se utilizarán para el equilibrio de carga
- Un grupo de servidores que forman un grupo de redundancia

Los grupos también pueden tener una semántica sobrecargada. Por ejemplo, los puntos de conexión miembros del grupo de dominio L2 son miembros de un dominio de difusión L2 (subred). De manera similar, los puntos de conexión miembros del dominio L3 tienen accesibilidad a L3.

Los grupos pueden estar compuestos por otros grupos. Los miembros de puntos de conexión del grupo pueden resolverse mediante instancias explícitamente, o puede haber una especificación de membresía dinámica como parte de una definición de grupo en que los puntos de conexiones estén implícitos en la especificación presente en el grupo.

Por ejemplo, un dominio L3 puede tener enrutamiento interdominio sin clase (CIDR) o una subred como propiedad; por lo tanto, todos los puntos de conexión con direcciones IP en ese rango/subred son implícitamente miembros del grupo. En ese caso, el diseño de referencia de L3 al servidor especifica una cantidad de subredes externas/internas. Aunque los puntos de conexión (contenedores) no se especifican ni administran explícitamente en Apstra, se da a entender que los contenedores pertenecen a los dominios L3 especificados. Pero, para modelar la segmentación detallada, también vamos a introducir especificaciones explícitas de puntos de conexión de contenedores y los servidores en los que están alojados.

Una política define un comportamiento común y puede contener parámetros para definir ese comportamiento con precisión. Las políticas se pueden aplicar al grupo de manera no direccional, o se puede indicar la dirección cuando sea necesario, como en las políticas de seguridad. Los ejemplos de políticas específicas incluyen seguridad, equilibrio de carga, alta disponibilidad, afinidad (p. ej., el deseo de colocar puntos de conexión o distribuirlos) y calidad del servicio (QoS).

Las políticas pueden contener reglas cuando sea necesario. Las reglas suelen seguir el patrón de “condición seguida de una acción”. Por ejemplo, en la política de seguridad, “coincidir” es una declaración de condición y la acción es “permitir/denegar/registrar”.

## Escalabilidad: crecimiento sin obstáculos

Apstra admite el acceso y la gestión de estado distribuido transparente a la red, mientras que la ejecución paralela es compatible con los procesos separados. La ejecución en tiempo real está respaldada por un modelo de ejecución asíncrono basado en eventos junto con la programación de la ejecución en tiempo real. La eficiencia y la previsibilidad son compatibles con la compilación a través de C++ como lenguaje intermedio para lograr la eficiencia a nivel de máquina. Hay tres dimensiones para escalar.

### Escalamiento del estado

La primera dimensión es escalar el estado. Los almacenes de datos de Apstra se escalan horizontalmente mediante la adición de más pares de servidores con alta disponibilidad. Los almacenes de datos de intención y telemetría están separados y se pueden escalar de forma independiente según sea necesario. También hay un aprovisionamiento para almacenes de datos jerárquicos, en los que un almacén de datos de nivel superior se suscribe (sincroniza) solo al subconjunto de estado requerido (según lo especificado por el diseñador) de los almacenes de datos del siguiente nivel, según sea necesario para correlacionar el estado a través de almacenes de datos.

### Procesamiento de escalamiento

La segunda dimensión del escalamiento es el procesamiento. Apstra puede lanzar múltiples copias de agentes de procesamiento (por tipo de agente) si es necesario para compartir la carga de procesamiento. Se pueden agregar más agentes si se agregan más servidores para alojarlos. También gestiona el ciclo de vida de un agente.

La arquitectura de publicación/subscripción basada en el estado del sistema les permite a los agentes reaccionar (proporcionar lógica de aplicación) a un subconjunto de estado bien definido. La cobertura de toda la intención se realiza a través de agentes separados delegados para tratar con diferentes subconjuntos del estado. Esto significa que, cuando hay un cambio en la intención o en el estado operativo, la reacción del agente es un “cambio incremental” y es independiente del tamaño de todo el estado.

Apstra emplea el enfoque tradicional para lidiar con el escalamiento y la complejidad asociada: descomposición. El enfoque de que “todos sepan todo” no se escala. Debe distribuir el conocimiento sobre el estado deseado y permitir que cada agente determine cómo llegar a ese estado para evitar la necesidad de una toma de decisiones centralizada. El soporte de Apstra para consultas gráficas en tiempo real implica que los clientes como la IU puedan solicitar exactamente lo que quieran y obtener justo lo que necesitan y nada más, lo que permite tener un control detallado de la cantidad de datos que se obtienen desde el back-end.

### Escalamiento del tráfico de red

La tercera dimensión es escalar el tráfico de red. La comunicación entre los agentes y el almacén de datos utiliza un canal binario optimizado, lo que reduce significativamente la cantidad de tráfico en comparación con los protocolos basados en texto.

La tolerancia a errores se logra mediante la ejecución de una aplicación de Apstra como múltiples procesos, posiblemente ejecutándose en dispositivos de hardware separados conectados por una red y separando el estado del procesamiento con soporte para el estado replicado y la recuperación rápida del estado.

## Visión general de la arquitectura de Apstra

En la primera parte de este documento, comentamos algunos de los desafíos de la arquitectura de red del centro de datos y cómo funciona Apstra para resolver esos problemas. En este punto, profundizamos en los detalles de la arquitectura de Apstra.

Apstra se basa en una infraestructura de administración de estado distribuida, que se puede describir como una estructura de comunicación centrada en datos con un almacén de datos en una memoria escalable horizontalmente y tolerante a errores. Todas las funcionalidades de la aplicación de diseño de referencia específica se implementan a través de un conjunto de agentes sin estado. Los agentes se comunican entre sí mediante un canal de comunicación lógico basado en la publicación y la suscripción, y esencialmente implementan la lógica de la aplicación.

Cada aplicación de diseño de referencia de Apstra es simplemente una recopilación de agentes sin estado como se describió anteriormente. En términos generales, hay tres clases de agentes:

1. Los **agentes de interacción (web)** son responsables de interactuar con los usuarios, es decir, tomar la entrada del usuario y proporcionarle el contexto pertinente del almacén de datos.
2. Los **agentes de aplicación** son responsables de realizar transformaciones de datos específicas del dominio de la aplicación suscribiéndose a entidades de entrada y produciendo entidades de salida.
3. Los **agentes de dispositivos** residen (o son servidores proxy) en un sistema virtual o físico administrado, como un conmutador, un servidor, un firewall, un equilibrador de carga o, incluso, un controlador. Se utilizan para escribir configuraciones y recopilar telemetría mediante interfaces nativas (específicas de los dispositivos).

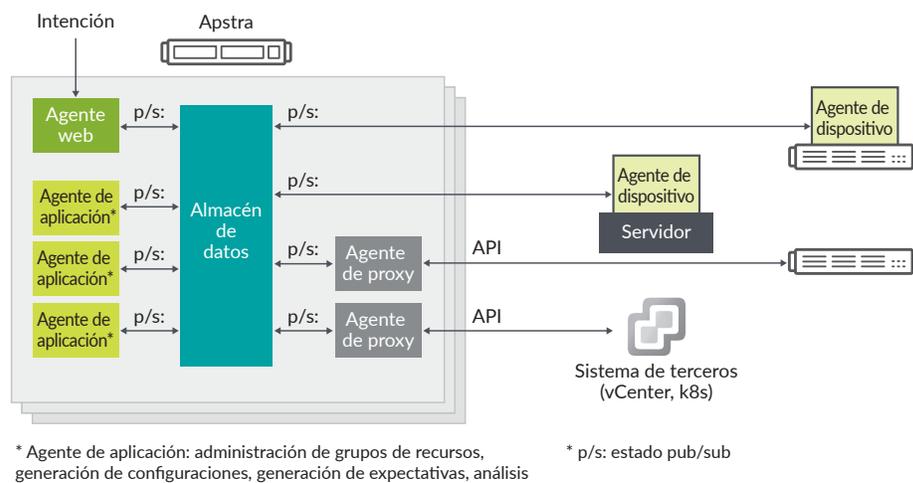


Figura 9: Agentes de Apstra

Esta interacción se puede ilustrar con un ejemplo que describe una parte de la aplicación de diseño de referencia de redes del centro de datos de Apstra.

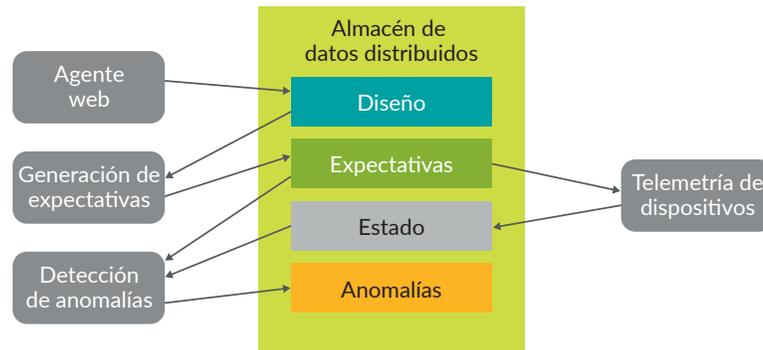


Figura 10: Aplicación de diseño de referencia de redes del centro de datos del sistema basado en intenciones de Apstra

El agente web toma la entrada del usuario, en este caso, un diseño de una estructura L3 Clos que contiene la cantidad de spines, leafs y enlaces entre ellos, así como los grupos de recursos que se deben utilizar para las direcciones IP de la estructura y los números de la notación de sintaxis abstracta (ASN). El agente web publica esta intención en el almacén de datos como un conjunto de relaciones y nodos gráficos y sus respectivas propiedades.

El agente de compilación se suscribe a esta intención y:

- Realiza validaciones de corrección e integridad
- Asigna recursos de grupos de recursos

Luego, suponiendo que se aprueben las validaciones, el agente de la compilación publica esa intención junto con las asignaciones de recursos en el almacén de datos:

1. El agente de representación de configuración se suscribe a la salida del agente de compilación.
2. Por cada nodo, el agente de configuración obtiene los datos pertinentes, incluidos los recursos, y los fusiona con las plantillas de configuración.
3. El agente de expectativas también se suscribe a la salida del agente de compilación y genera expectativas que deben cumplirse para validar el resultado.
4. El agente de telemetría del dispositivo se suscribe a la salida del agente de expectativas y comienza a recopilar la telemetría pertinente.
5. Las sondas de IBA procesan la telemetría sin procesar, la comparan con las expectativas y publican las anomalías.
6. El agente de la RCI analiza las anomalías y las clasifica en síntomas, impactos y causas raíz identificadas.

Los agentes se comunican a través de interfaces basadas en atributos (de ahí el término “centrado en datos”) mediante la publicación de entidades y la suscripción a los cambios en las entidades. El concepto “centrado en datos” también implica que la definición de datos es parte del marco y se implementa mediante la definición de las entidades, a diferencia de, por ejemplo, los sistemas basados en mensajes.

El sistema de publicación-suscripción centrado en datos no tiene los problemas de los sistemas basados en mensajes. En un sistema basado en mensajes, tarde o temprano la cantidad de mensajes excede la capacidad del sistema para almacenarlos o consumirlos. Lidar con esto es difícil, ya que se debe reproducir el historial de mensajes para llegar a un estado coherente.

El sistema centrado en datos es resistente a las oleadas de cambios de estado, ya que depende fundamentalmente solo del último estado. Este estado captura el contexto importante y abstrae todas las posibles (e irrelevantes) secuencias de eventos que condujeron a este. El código escrito con el paradigma de máquina de estado es más fácil de leer, mantener y depurar.

Los problemas complejos (p. ej., la elasticidad y la tolerancia a errores) se resuelven una vez y en nombre de todos los agentes. Entonces, la arquitectura típica consta de una serie de agentes sin estado que se pueden reiniciar en caso de falla y que retoman el estado en que los dejaron simplemente releendo el estado al que se suscribieron desde la base de datos del sistema (SysDB).

## Beneficios de la arquitectura de Apstra

La arquitectura de Apstra brinda beneficios significativos, puesto que resuelve algunos de los problemas de redes del centro de datos más complejos, como se describe en este documento. La arquitectura:

- **Ayuda a los operadores a lidiar con el cambio de manera confiable.** Esto es posible a través de la intención de consulta en tiempo real y el contexto operativo.
- **Simplifica todos los aspectos del ciclo de vida del servicio de red,** incluidas las operaciones de los días 0, 1 y 2. Y la simplicidad reduce la probabilidad de error del operador.
- **Reduce el riesgo operativo a través de la orquestación con estado,** que aprovecha las validaciones de condiciones previas, las validaciones de condiciones posteriores, la representación de configuración automatizada y la validación de expectativas automatizada.

### Los beneficios especiales de la identificación de la causa raíz y el análisis basado en la intención

Los componentes de análisis operativo de Apstra (identificación de la causa raíz y análisis basado en la intención) permiten lo siguiente:

- **Reducir el tiempo promedio de reparación y la carga de trabajo de los expertos en la materia (SME)** a través de análisis operativos simplificados, a fin de liberar a sus recursos más capacitados para dedicar su tiempo a mejorar e innovar en lugar de combatir crisis.
- **Extraer más conocimiento** recopilando y almacenando menos datos. Impulsado por el contrato de comportamiento de diseño de referencia, Apstra sabe lo que busca y recopila solo esa información. Este procesamiento sobre la marcha puede generar una reducción de cinco a seis órdenes de gran magnitud en las necesidades de almacenamiento y procesamiento posterior de datos no interesantes. Esto le permite ejecutar su infraestructura de manera más eficiente y seguir siendo competitivo mientras controla los costos.
- **Ver los problemas de forma rápida y sencilla.** Apstra identifica eventos significativos y procesables en un “punto de observación simple” y evita el problema de los síntomas que son simplemente errores de una causa raíz identificada.
- **Automatizar flujos de trabajo complejos.** El sistema Apstra le permite automatizar flujos de trabajo de resolución de problemas ricos en contexto (que, de lo contrario, serían engorrosos, ineficaces, lentos y costosos).
- **Lograr un mantenimiento sin contacto.** Dado que está en constante sincronización con la intención, Apstra permite un mantenimiento sin contacto y sin costo en presencia de cambios. Como tal, es resistente, responde automáticamente a los cambios y ahorra elevados costos asociados con el mantenimiento de las canalizaciones de procesamiento de datos.
- **Eliminar el costoso desarrollo propio.** El desarrollo propio de los esfuerzos de integración de canalización de procesamiento de datos es costoso y frágil, le quita tiempo y recursos a su negocio principal, y requiere mucho trabajo pesado de los expertos en la materia.
- **Ofrecer resultados muy precisos** en comparación con los enfoques de aprendizaje automático/inteligencia artificial.
- **Emplear un enfoque independiente del proveedor,** a fin de darle libertad de elección entre los mejores proveedores y capacidades.
- **Utilizar API comunes** en su infraestructura de nube pública/privada/híbrida.

### Escale desde el día 1: estudio de caso

Apstra se creó teniendo en cuenta la escalabilidad desde el día 1. Vea a este cliente que la usó para realizar una validación extensa con resultados impresionantes:

- **Infraestructura física:** 6000 validaciones de estado de interfaz; 1000 validaciones de cableado; 36 000 contadores de errores; 10 000 validaciones de métricas de alimentación, temperatura y tensión
- **Plano de datos L2/L2:** 12 000 contadores de caída de cola; corrección de cientos de MLAG; 3000 validaciones de protocolo de árbol de expansión (STP); notificaciones de cambios de estado
- **Plano de control:** estado de 1500 sesiones de BGP; alrededor de 500 siguientes saltos esperados/rutas predeterminadas
- **Planificación de capacidad:** alrededor de 500 análisis de tendencias con umbrales configurados para el uso de tablas de enrutamiento; tablas del protocolo de resolución de direcciones (ARP); tablas multidifusión por enrutamiento y reenvío virtual (VRF); 6000 validaciones de uso de enlaces

- **Cumplimiento:** garantizar que las versiones del sistema operativo esperadas se ejecuten en los aproximadamente 100 conmutadores
- **Multidifusión:** 2500 validaciones de los vecinos esperados del módulo de interfaz física (PIM); 300 verificaciones de puntos de encuentro; 25 validaciones con respecto a la detección de patrones anormales en el recuento de fuentes, grupos, pares fuente-grupo en puntos de encuentro

Esencialmente, en lugar de un solo punto de observación con 82 000 entradas, al cliente se le presentó un punto de observación simple que solo mostraba anomalías, categorizadas en un tablero según las especificaciones del cliente, las cuales:

- Fueron 100 % precisas (y no se infirieron estadísticamente)
- No requerían mantenimiento continuo, ya que estaban en constante sincronización con la topología y la intención del cliente
- Proporcionaron el contexto procesable pertinente (lo que se desvió, la naturaleza de la desviación, el estado deseado)
- Permitieron ahorrar en necesidades de almacenamiento y procesamiento (solo se requirieron 9 GB de RAM y un disco de 96 GB)
- Eliminaron el peligro de bloqueo interno y legado por escribir canalizaciones de procesamiento de datos complejas y frágiles

## Conclusión

La incapacidad de realizar cambios confiables en su infraestructura de TI es un obstáculo importante para el crecimiento y la innovación. Apstra elimina ese miedo y permite erradicar la temida "infraestructura heredada" para siempre, lo que le permite realizar cambios de manera confiable y, a su vez, le permitirá innovar de manera confiable y mantenerse competitivo.

## Acerca de Juniper Networks

Juniper Networks simplifica las redes con productos, soluciones y servicios que conectan al mundo. A través de nuestras innovaciones técnicas, eliminamos los obstáculos y la complejidad de la gestión de redes en la era de la nube para resolver las dificultades que nuestros clientes y socios afrontan a diario. En Juniper Networks, creemos que la red es un medio para compartir el conocimiento y el progreso que ayudan a cambiar el mundo. Nuestro compromiso es imaginar formas innovadoras de ofrecer redes automatizadas, escalables y seguras que permitan moverse a la velocidad de los negocios.

### Sede corporativa y de ventas

Juniper Networks, Inc.  
1133 Innovation Way  
Sunnyvale, CA 94089 EE. UU.  
**Teléfono: 888.JUNIPER (888.586.4737)**  
o +1.408.745.2000  
**www.juniper.net**

### Sedes en APAC y EMEA

Juniper Networks International B.V.  
Boeing Avenue 240  
1119 PZ Schiphol-Rijk  
Ámsterdam, Países Bajos  
**Teléfono: +31-0-207-125-700**

