Settings – Server-side Services

Dynatrace Training Module

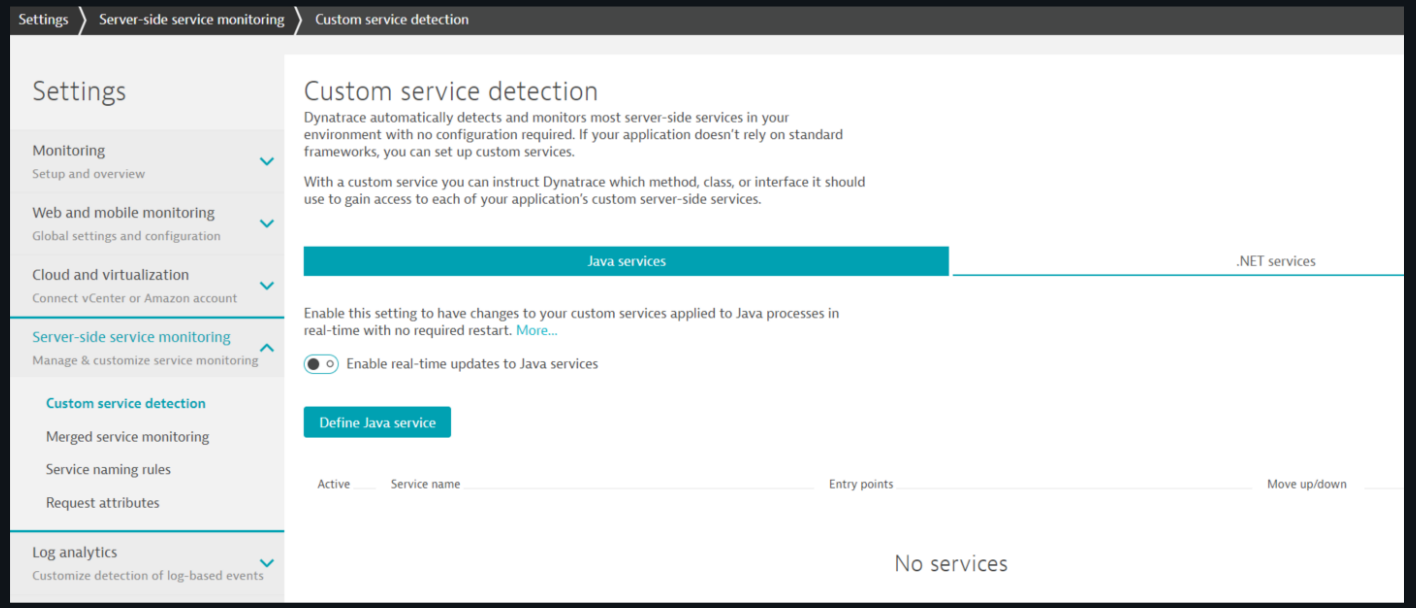
Agenda

- Custom Services
- Merged Service Monitoring
- Request Attributes
- API Detection Settings
- Deep Monitoring

Custom Services

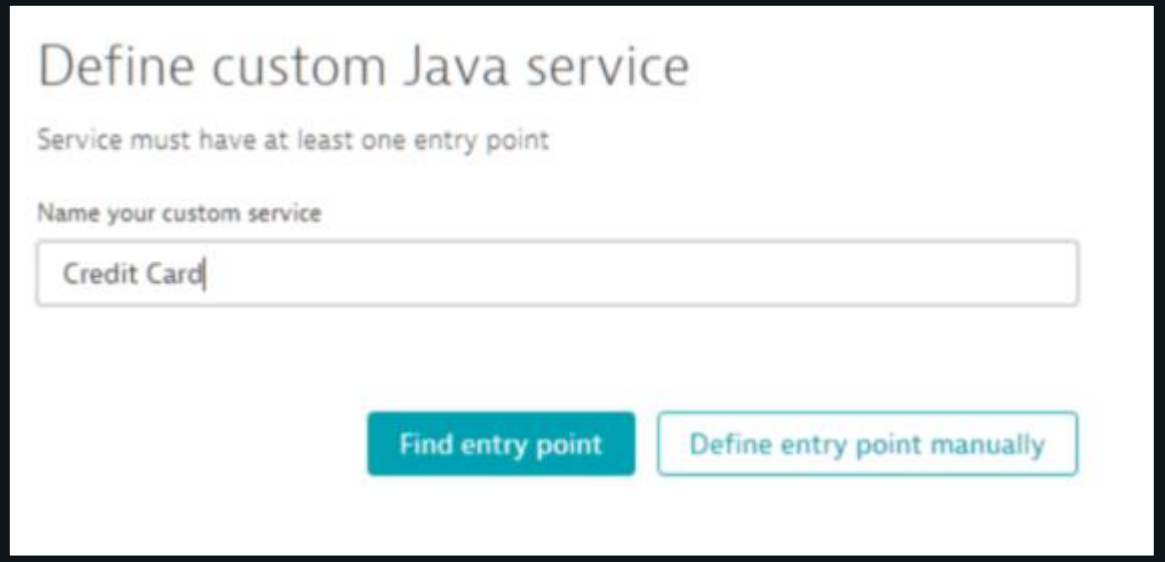
Custom Services

- Dynatrace automatically detects and monitors most server-side services in your environment with no configuration required
- If your application doesn't rely on standard frameworks, you can set up custom services
- With a custom service you can instruct Dynatrace which method, class, or interface it should use
- Only Java and .Net is currently supported



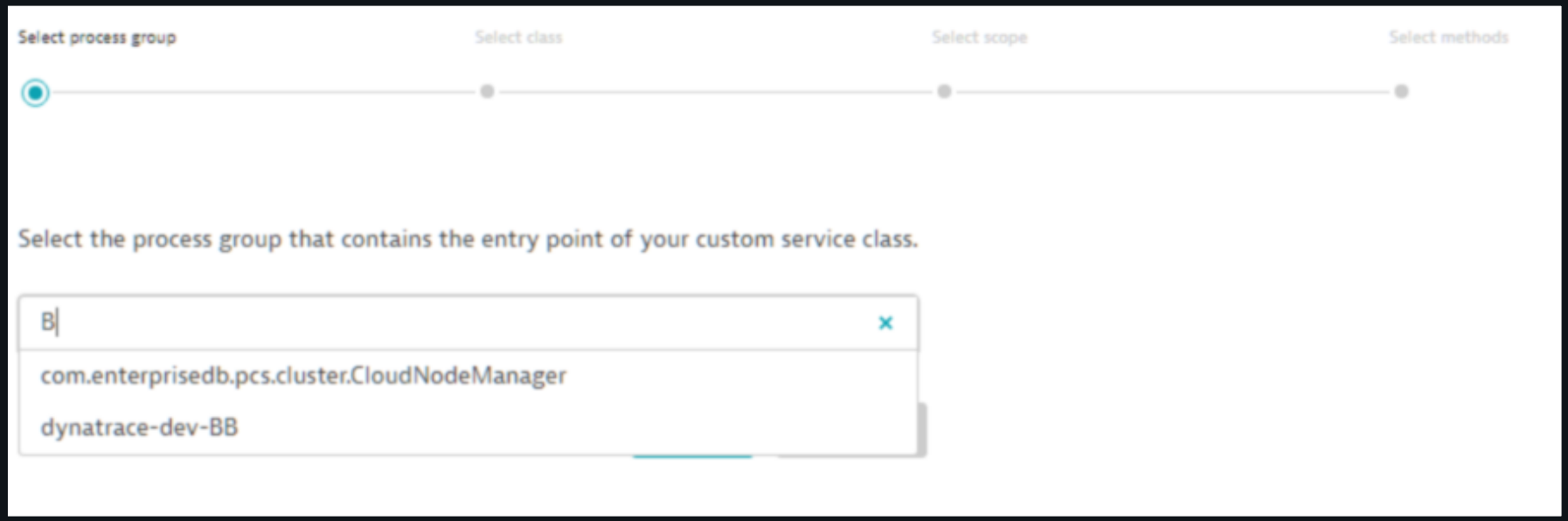
Custom Services

- Provide a Name for your custom service
- Click the Find entry point button



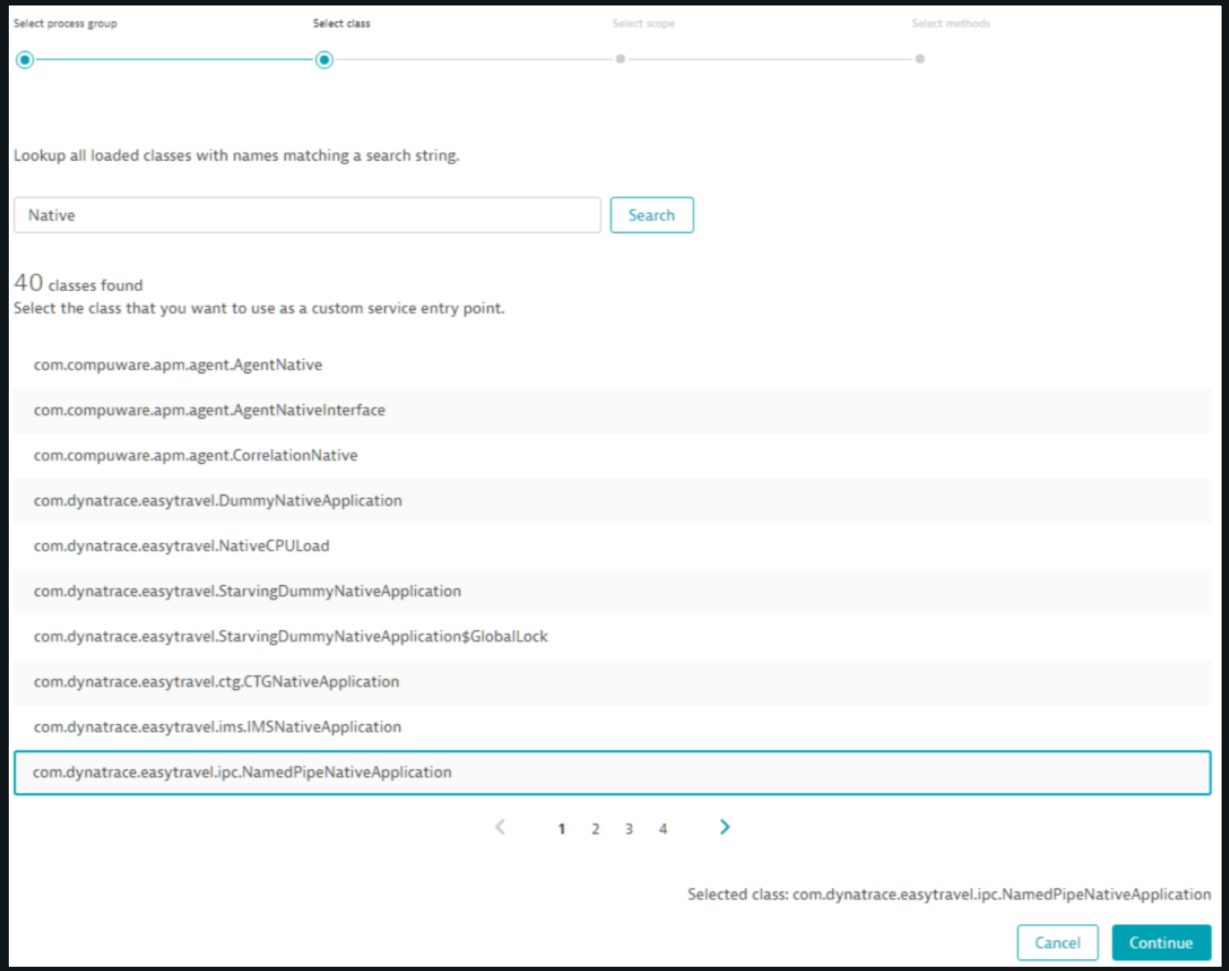
Custom Services

- Select the process group that contains the entry point of the custom service you want to monitor.
- Begin typing in the Select the process group text field to filter the process group list



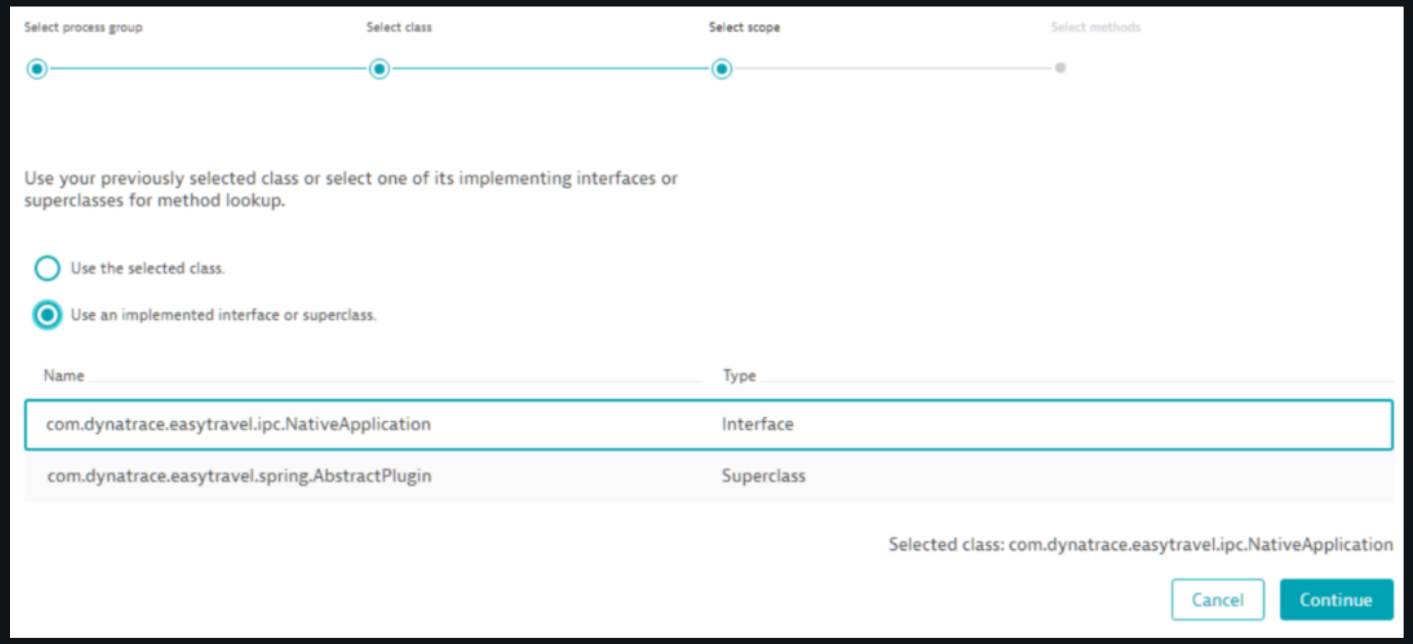
Custom Services

- Search for the class
- Type in part of the class name and click the Search button to locate the class
- Refine your search if the class you want isn't listed



Custom Services

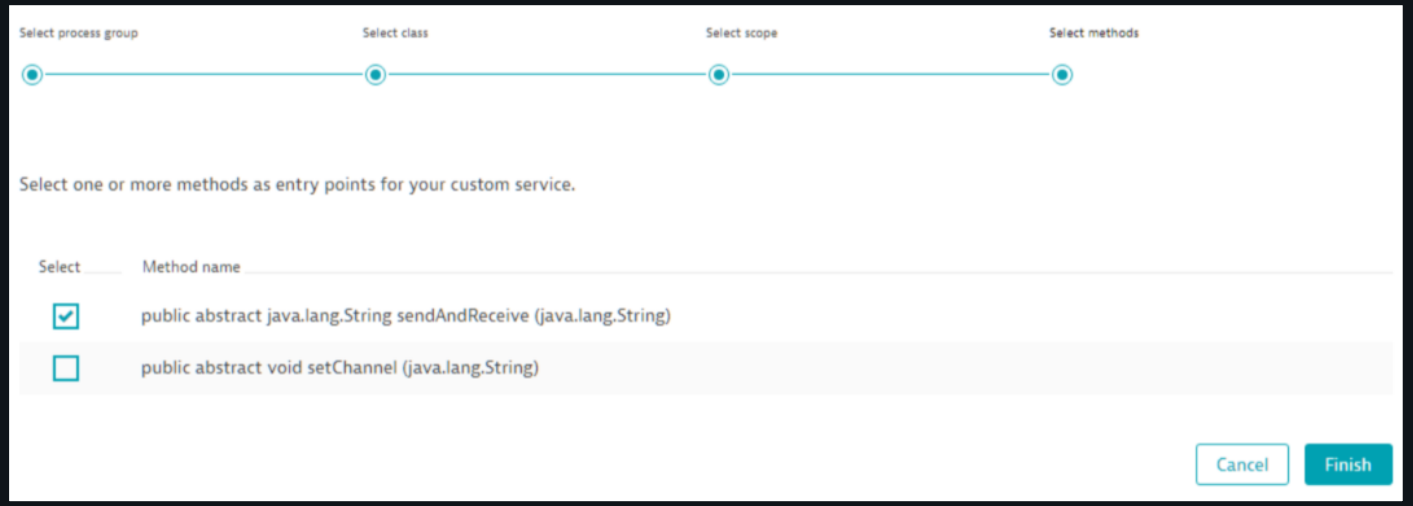
- (optional)
- If the selected class has super classes or an interface, select the Use an implemented interface or superclass option button to use one of these instead of the implementation you selected
- Otherwise, select Use the selected class



Custom Services

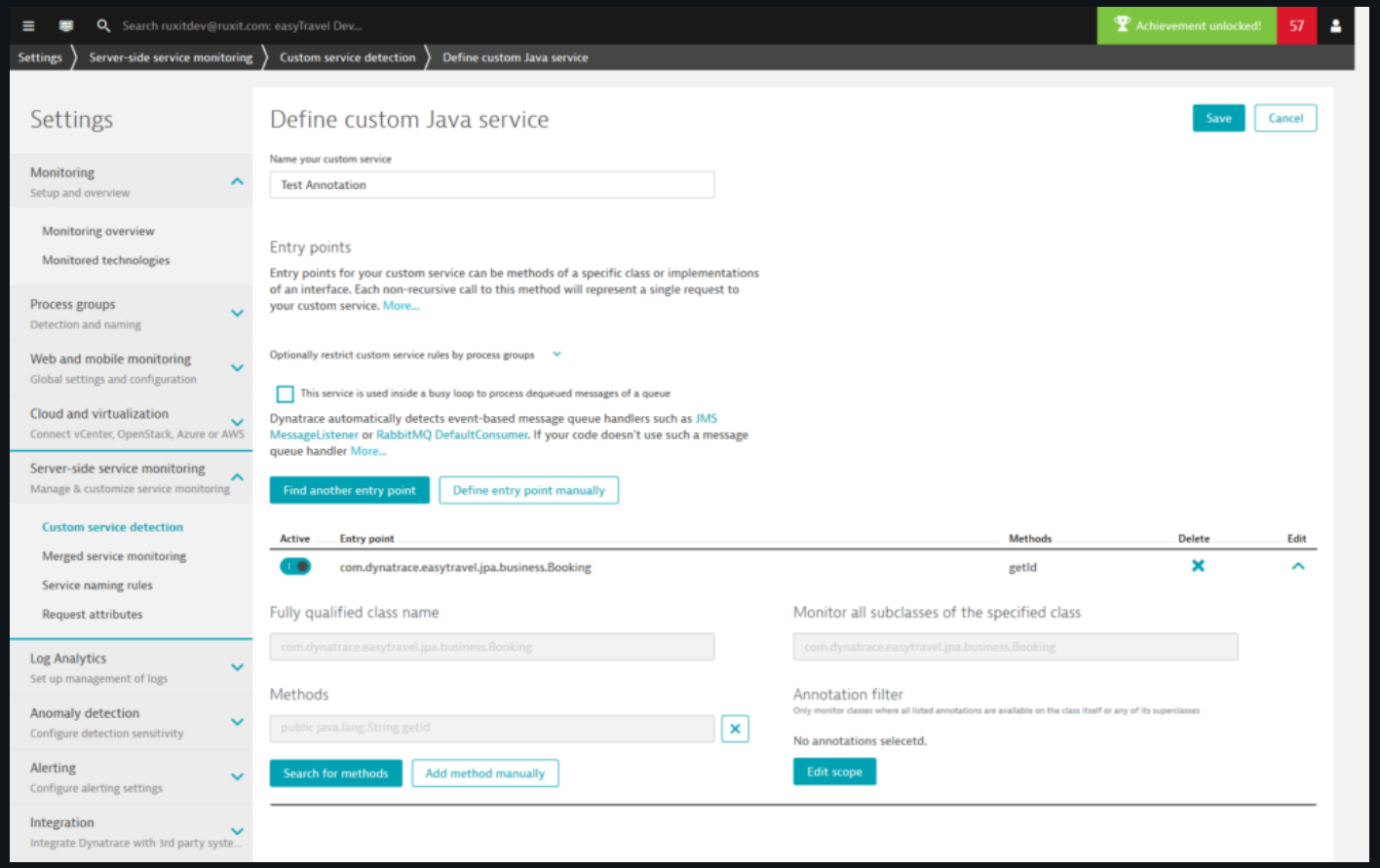
- Select the methods of the class that should act as entry points to the custom service
- Click Finish to save the service's entry point configuration

- Java only
  - Set the "Enable real-time updates to Java services" switch to the On position



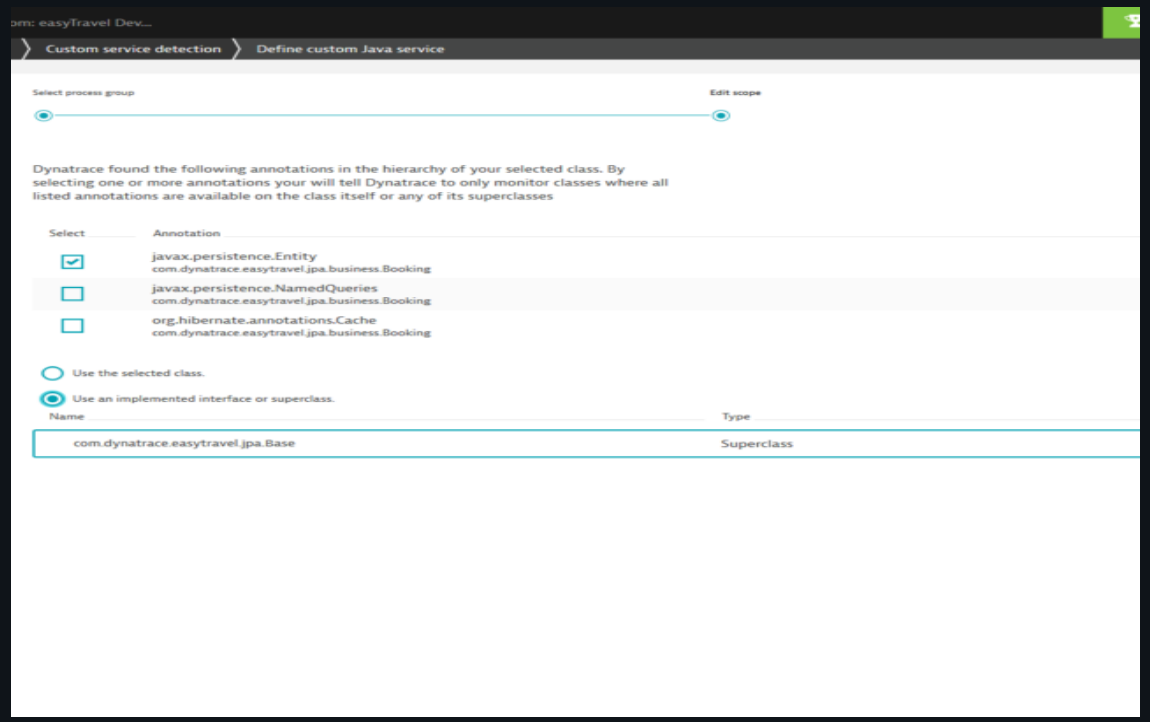
Use Java Annotations to define a custom service

- Many frameworks use interfaces but more and more rely on annotations to designate components.
- Instead of thinking about annotations right away simply define your custom service based on one particular incarnation of your component.
- After the custom service is created a new option to 'edit scope' will appear.



Java Annotations

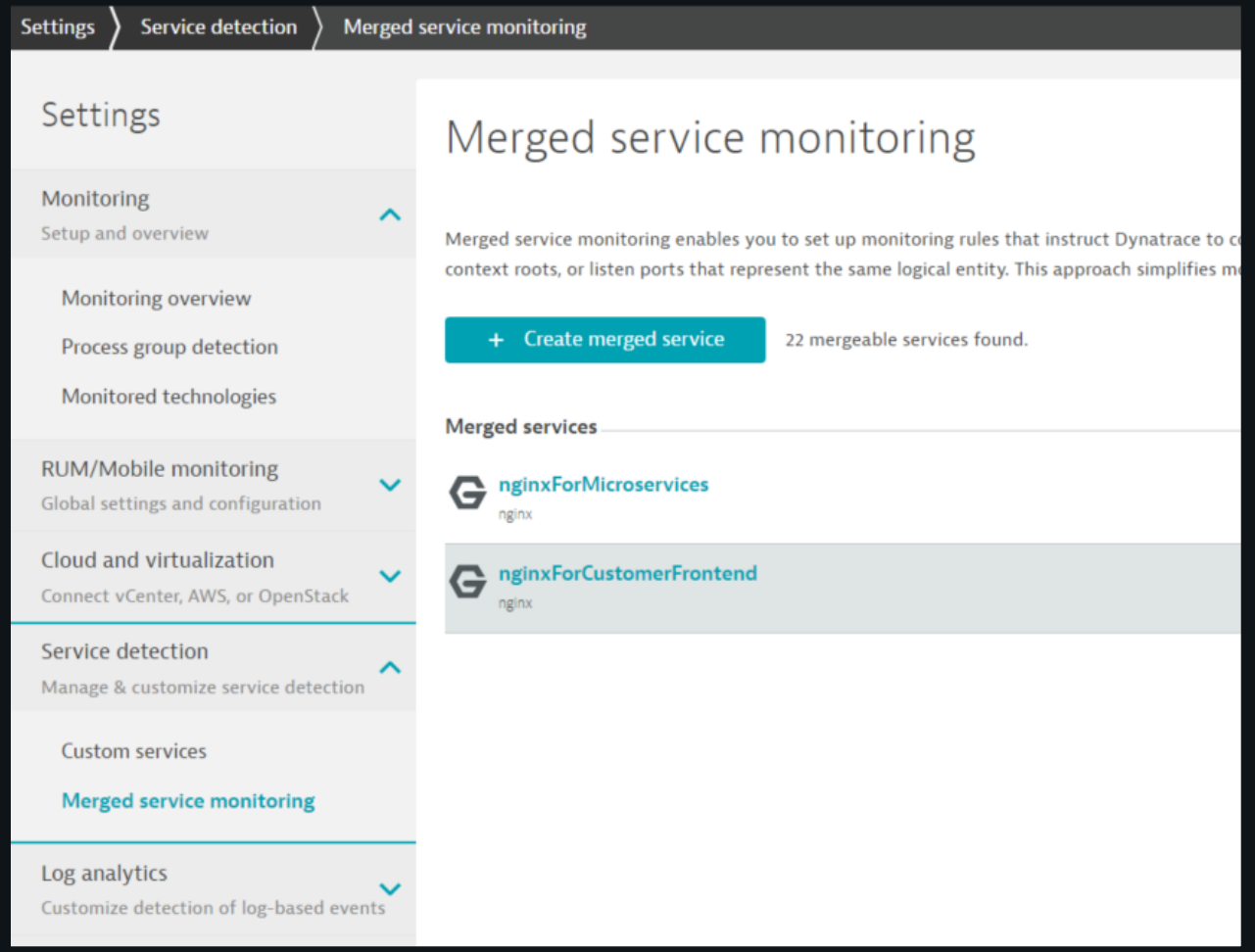
- Next, choose the process you want to ask for class meta data.
- Dynatrace will then show you all the annotation it found on the class.
- Most importantly this feature looks for all specific annotations in all super classes. This enables indirect matching scenarios where the annotation is on a base class not directly related to the interface match.



Merged Service Monitoring

Merged Service Monitoring

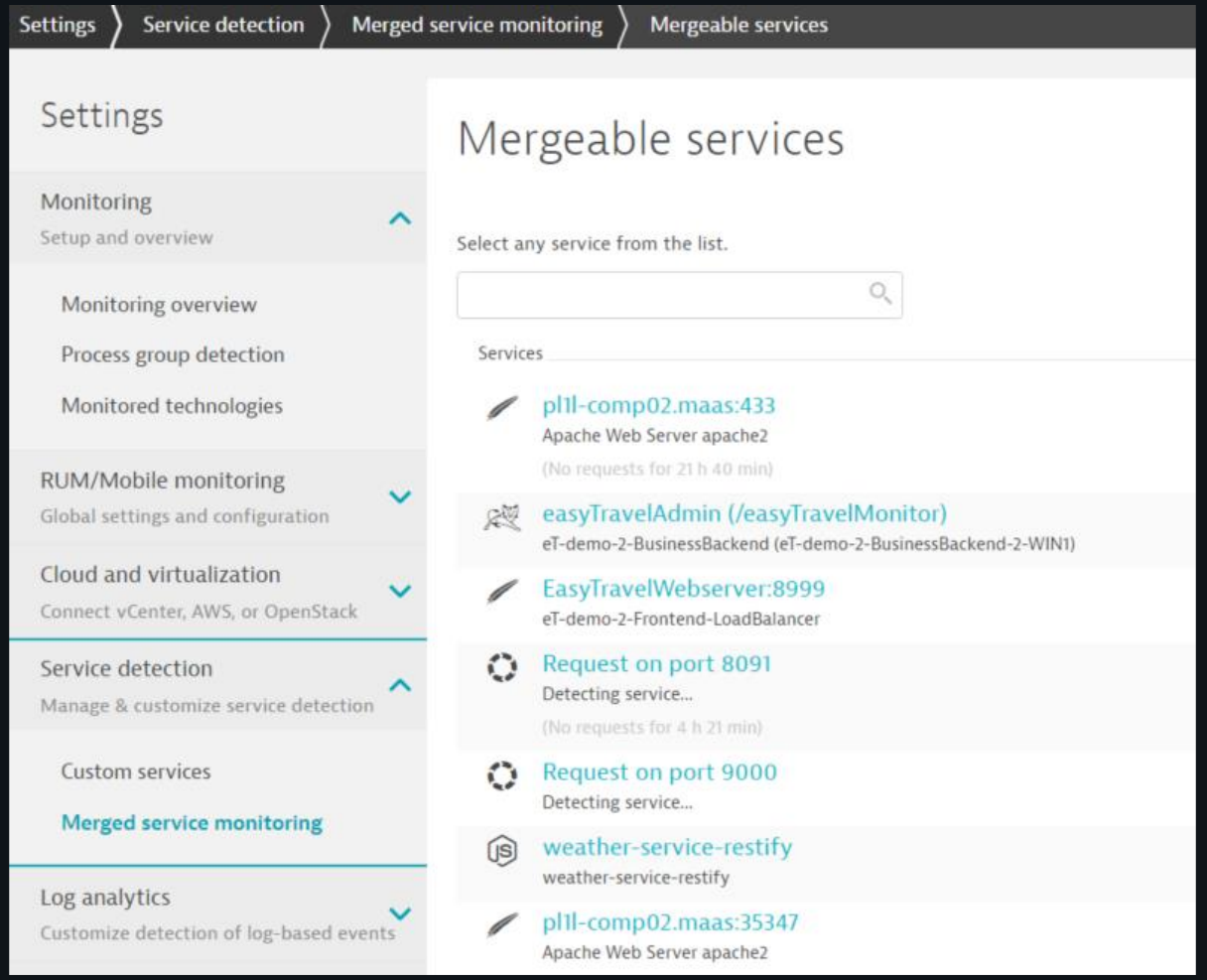
- Merged service monitoring enables you to set up monitoring rules that instruct Dynatrace to consider separately detected services as being logically the same service
- This is useful in cases where you have multiple virtual hosts, context roots, or listen ports that represent the same logical entity
- This approach simplifies monitoring and has no effect on the underlying services themselves



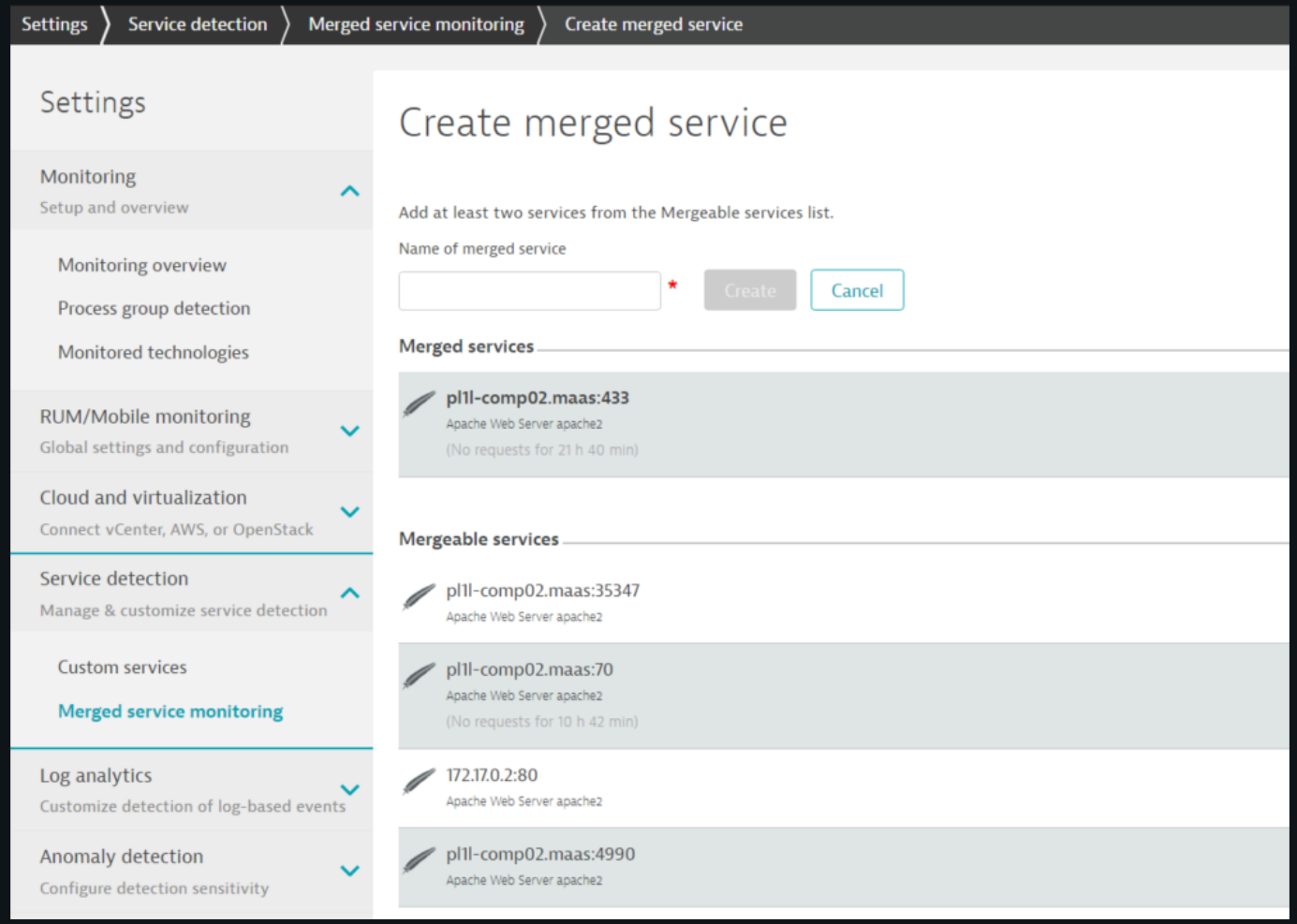
Merged Service Monitoring

- Say for example that you have an Apache web server with multiple virtual host definitions for *dynatrace.com*, *dynatrace.de*, and *dynatrace.at*
- From the perspectives of Apache and Dynatrace these are independent virtual hosts and services
- For your monitoring purposes however, you might want to view these services as a single logical service called *Dynatrace web page*

Merged Service Monitoring



Merged Service Monitoring



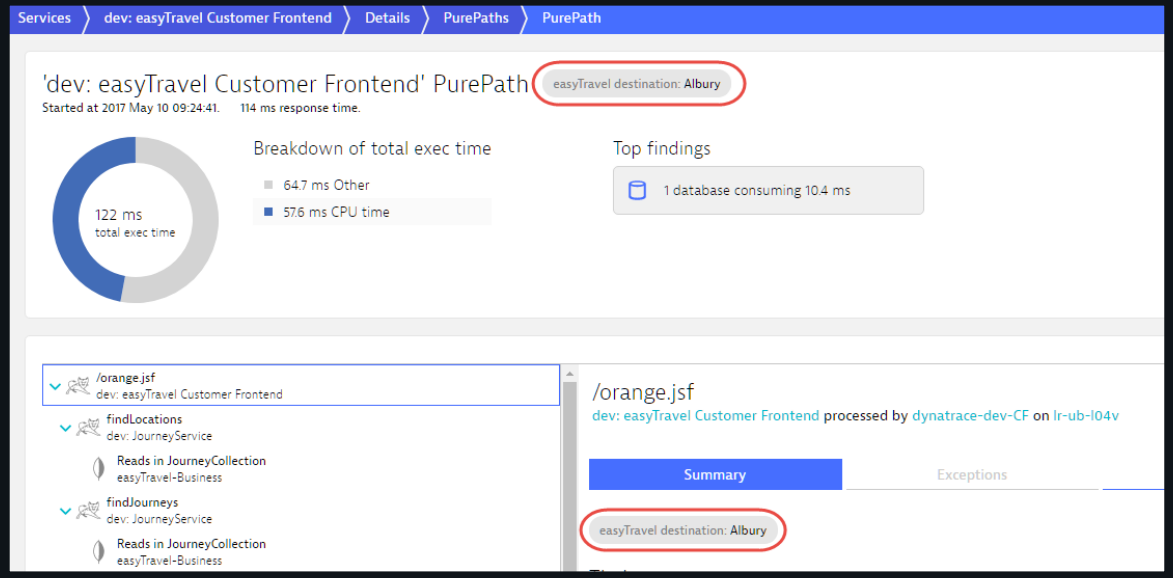
Request Attributes

Request Attributes

- Dynatrace tracks all requests from end-to-end and automatically monitors the services that underlie each transaction
- The performance and attributes of each request can be analyzed in detail
- You can even create custom, multi-faceted filters that enable you to analyze call sequences from multiple angles
- With such advanced request filtering, Dynatrace enables you to slice and dice your way through your requests to find the proverbial "needle in a haystack."
- Until now such filtering was only possible on certain predefined attributes
- You can now configure custom request attributes that you can use to improve filtering and analysis of problematic web requests

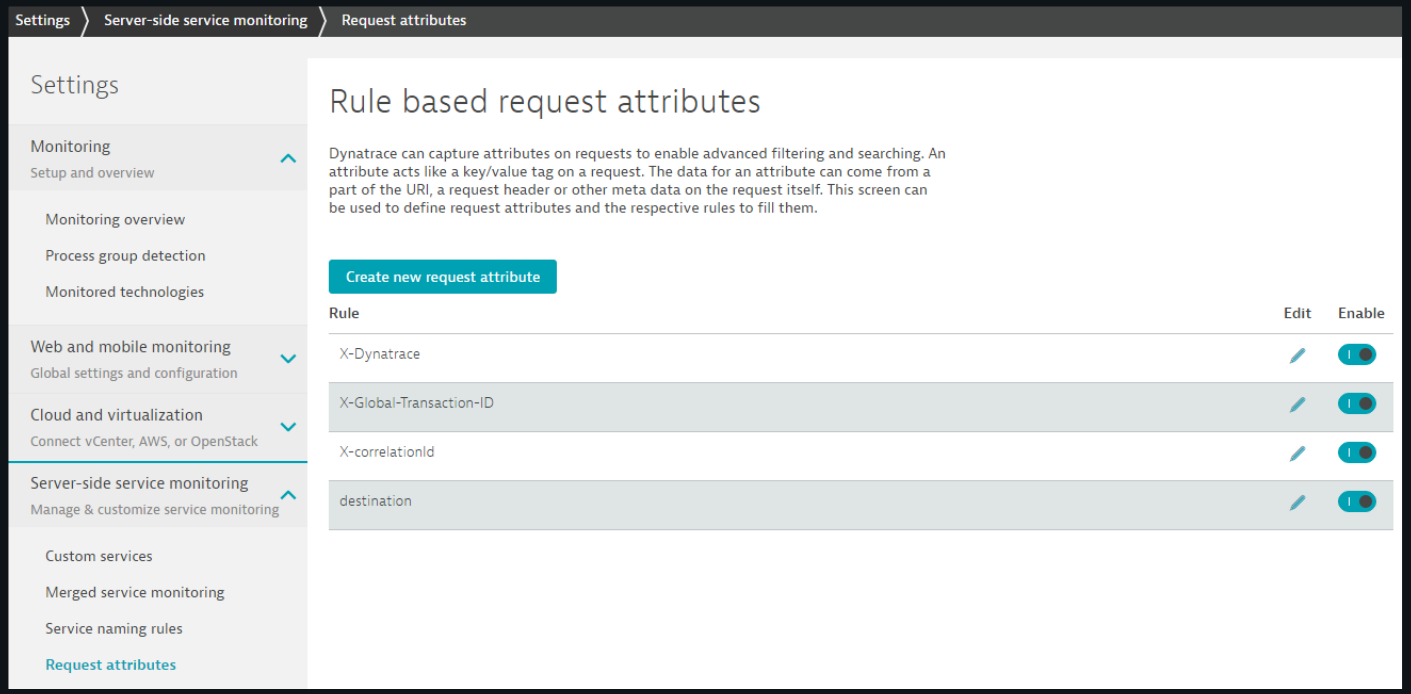
Request Attributes

- Request attributes are essentially key/value pairs that are associated with a particular service request
- For example, if you have a travel website that tracks the destinations of each of your customers' bookings, you can set up a destination attribute for each service request
- The specific value of the destination attribute of each request is populated for you automatically on all calls that include a destination attribute



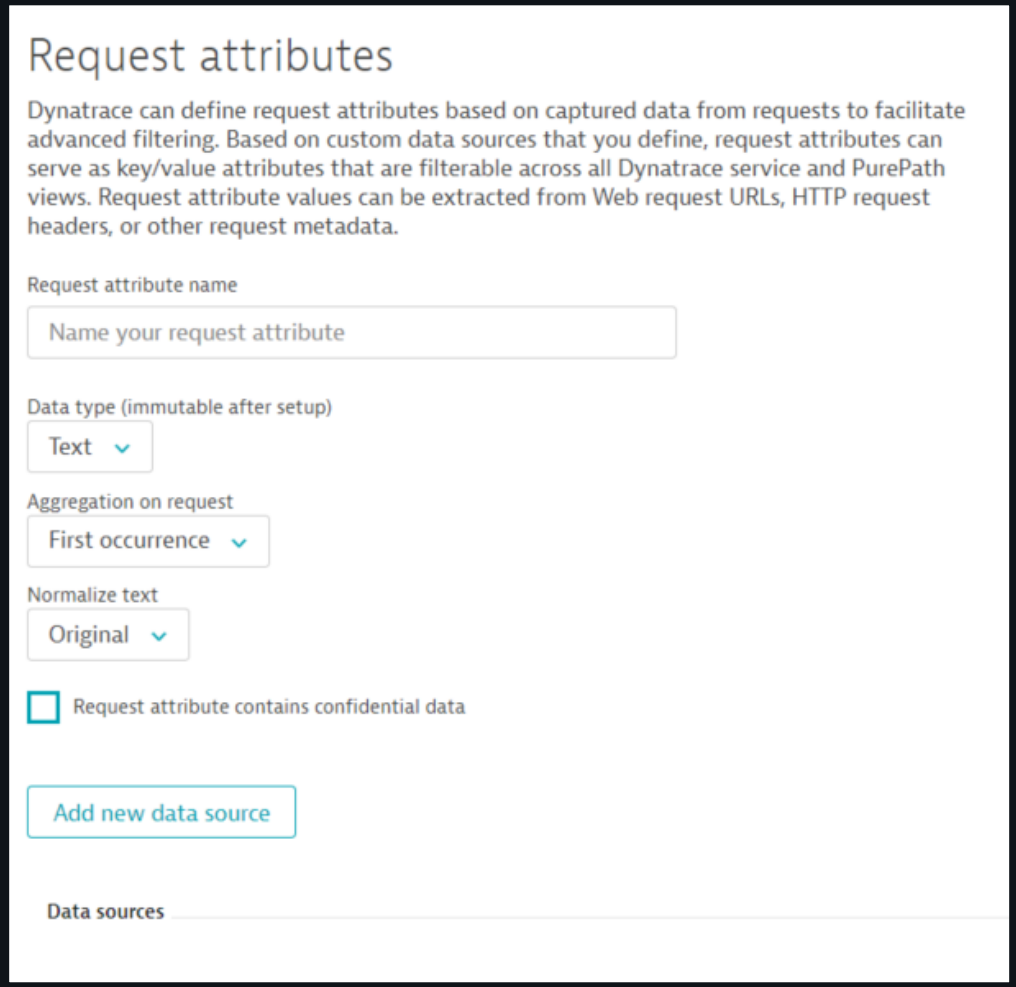
Create a Request Attribute

- Go to Settings
- Click the Create new request attribute button
- Provide a unique Request attribute name
- Request attributes can have one or more rules
  - Rules define how attribute values are fetched



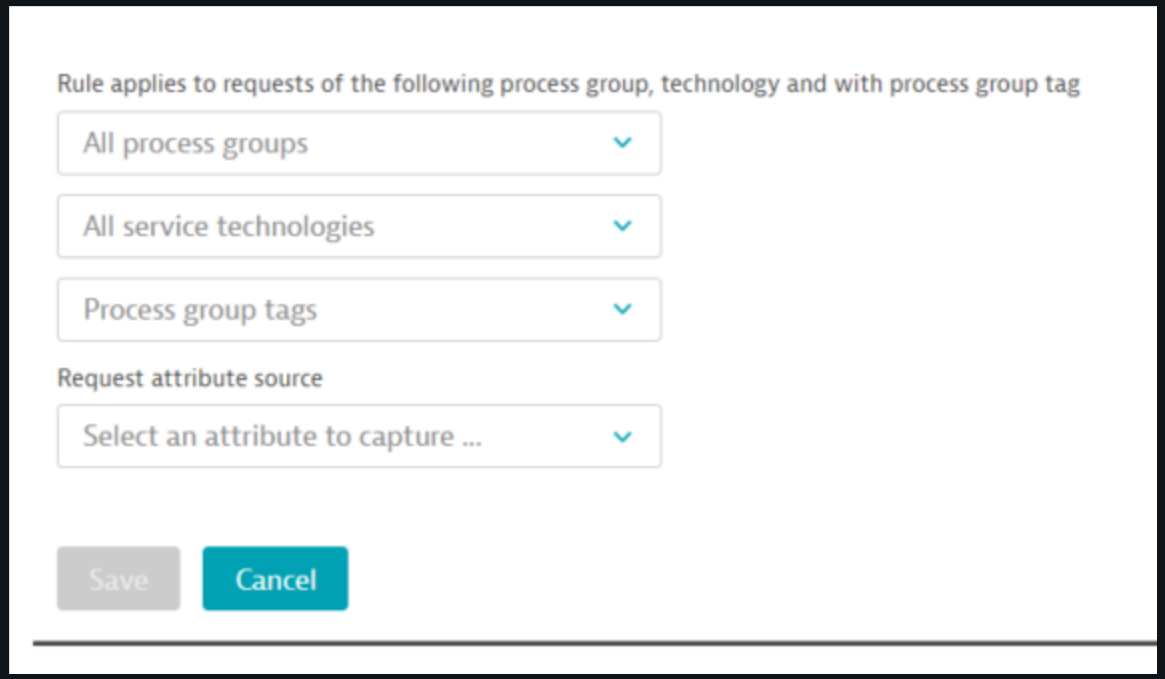
Create a Request Attribute

- Data type
  - Text, Integer, Double
- Aggregation
  - First/Last occurrence
  - Set of distinct values
  - Min/Max/Avg/Sum of captured values
  - Count occurrences/distinct
- Normalize text
  - Original/to upper/to lower



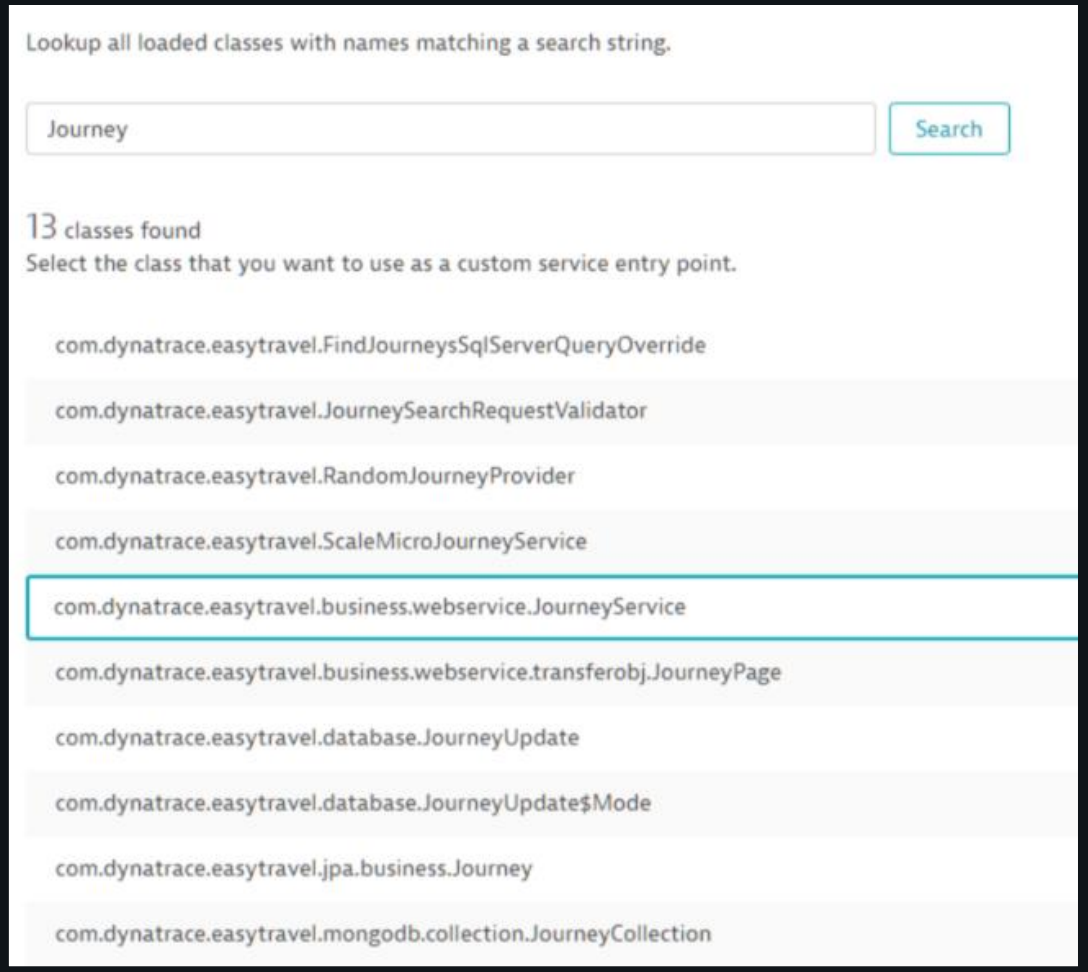
Create a Request Attribute

- Data Sources
  - HTTP Request/Response header
  - Web request URL
  - Web request URL query parameter
  - HTTP POST parameter
  - Java method
  - .Net method



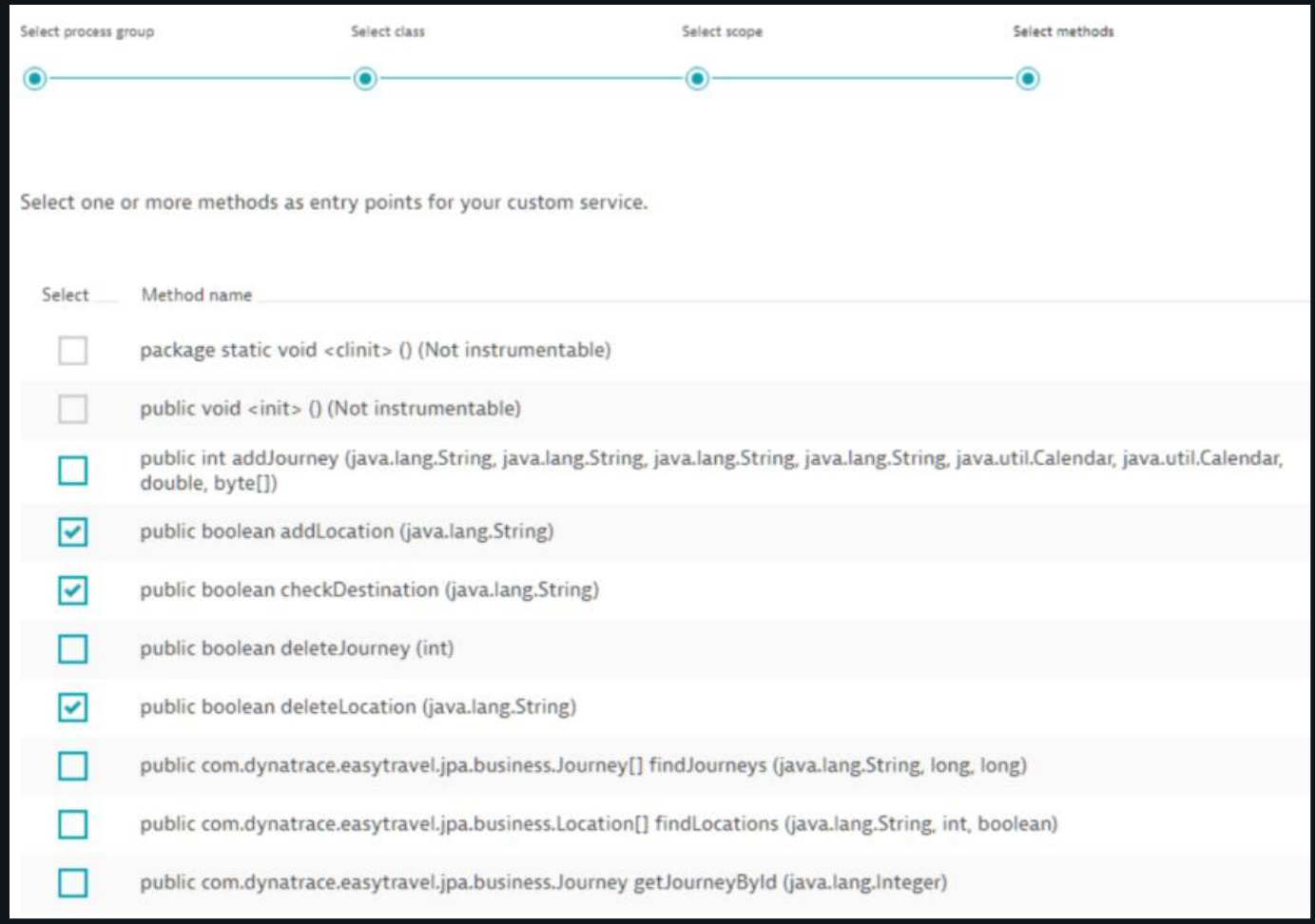
Capturing Method Arguments

- Select the process group that contains the classes or interfaces you're interested in
- Search for the class that includes the method you're interested in



Capturing Method Arguments

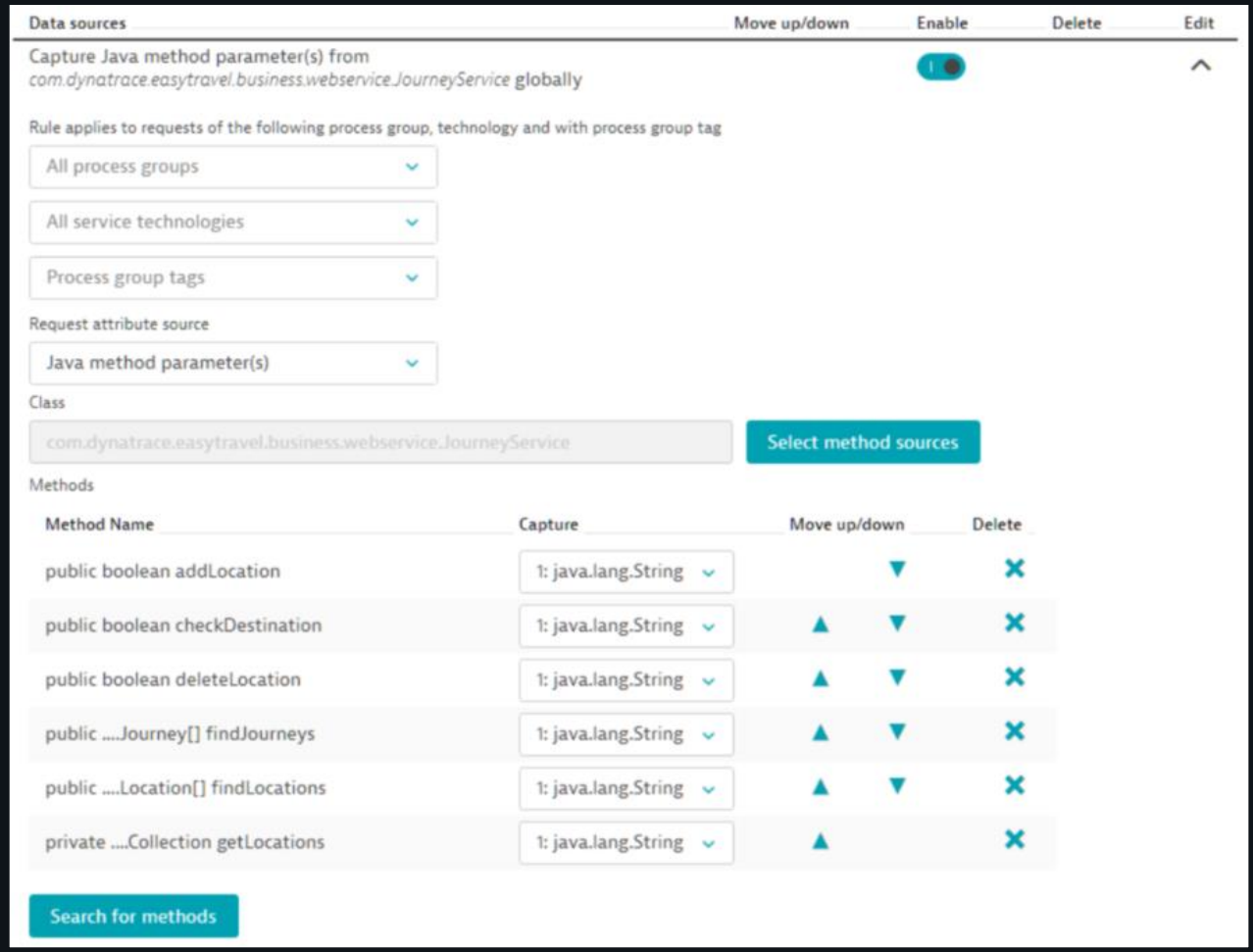
- Pick one or more methods that you want to capture parameters from



Capturing Method Arguments

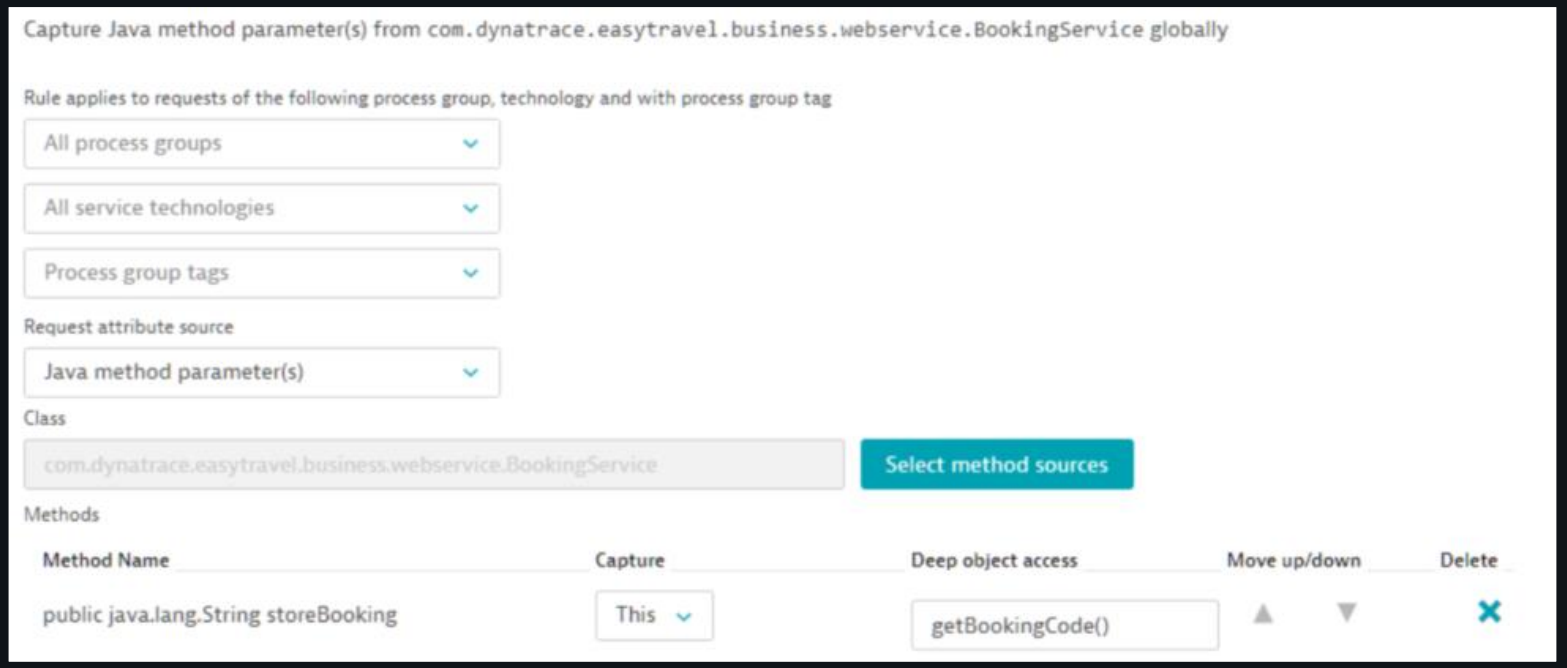
- Select which argument or return value you want to capture for each method

- Either restart the processes that you want this rule to apply to or ensure that real-time updates is enabled (Java only)



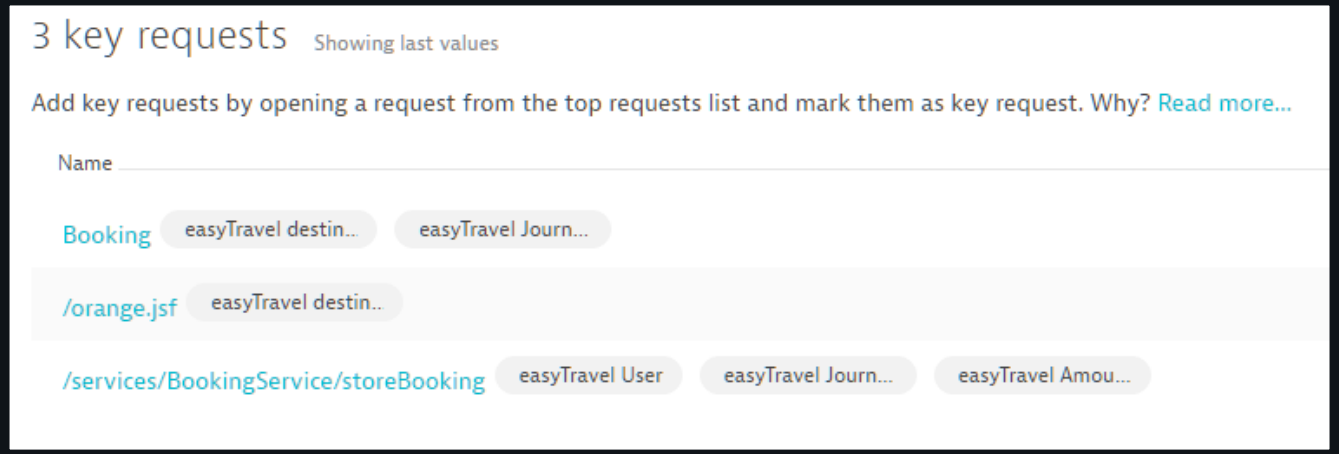
Deep Object Access

- You can access not only argument values and return values, but also the object itself
- Whenever an item to be captured (be it an argument or an object) is a complex object, you have the ability to define a method (chain) that enables deep access into the object



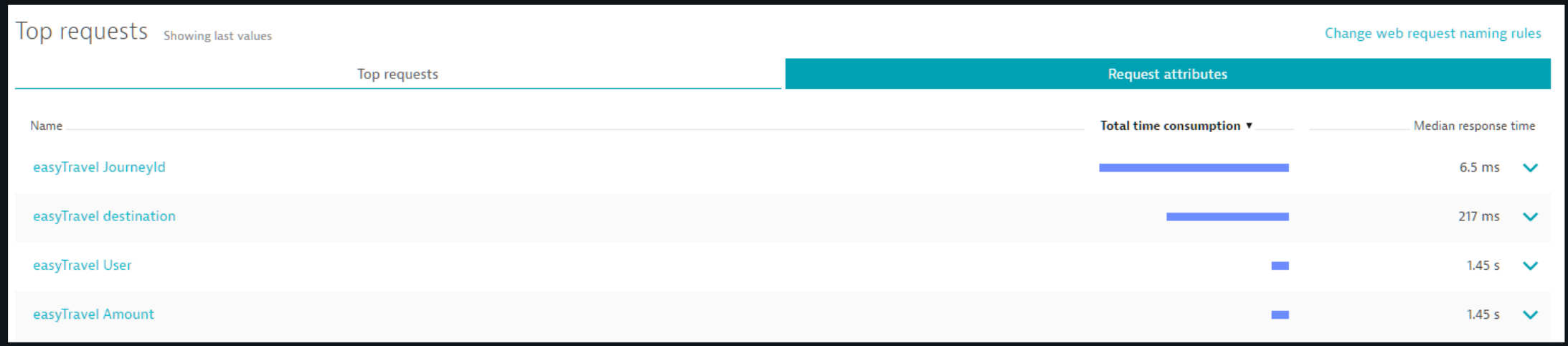
Viewing Request Attributes

- Once you've defined your attributes, go to any service page where you expect to see your defined request attributes
- Have a look at the Top requests section
- The requests now feature attribute labels indicating that at least some of respective requests contain the new request attribute
- Click any request attribute to filter the entire page view down to only those requests that carry the selected attribute



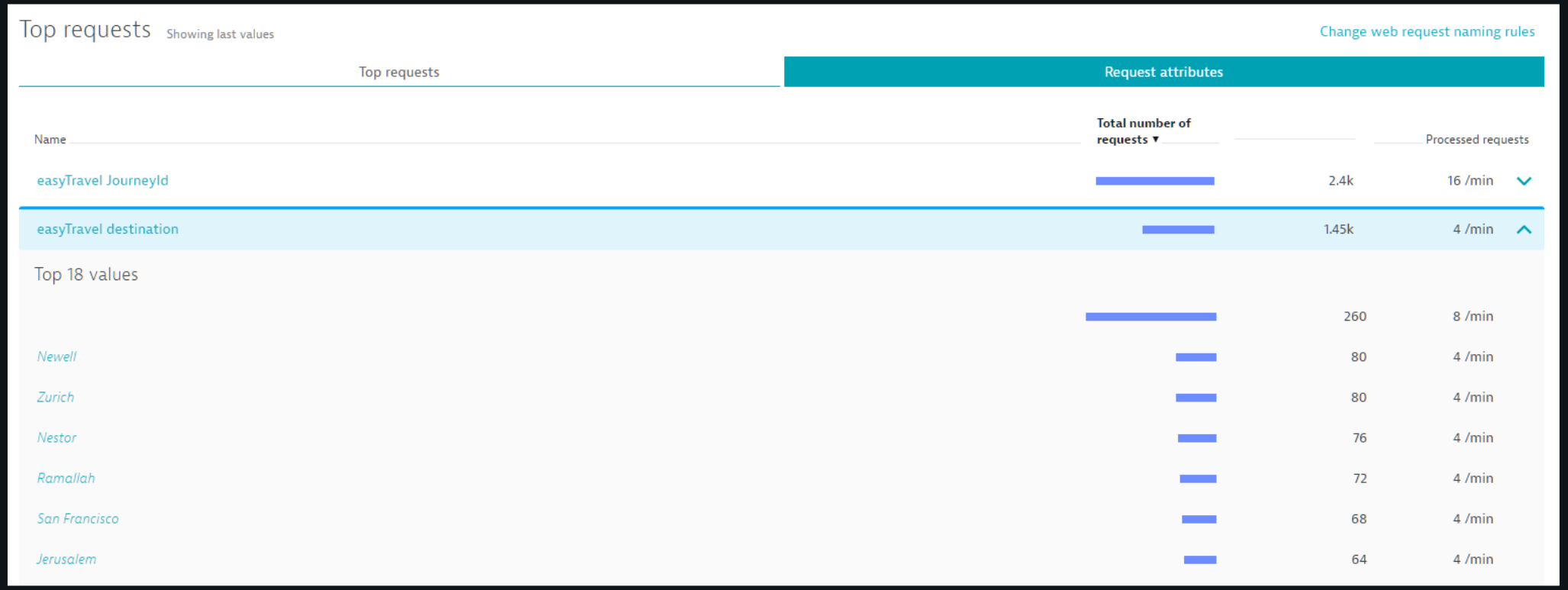
Viewing Request Attributes

- A new Request attributes tab has been added next to the Top requests tab
- This tab lists the request attributes that correspond to the request page
- This table reflects the current filter settings and shows the same metrics as the request table



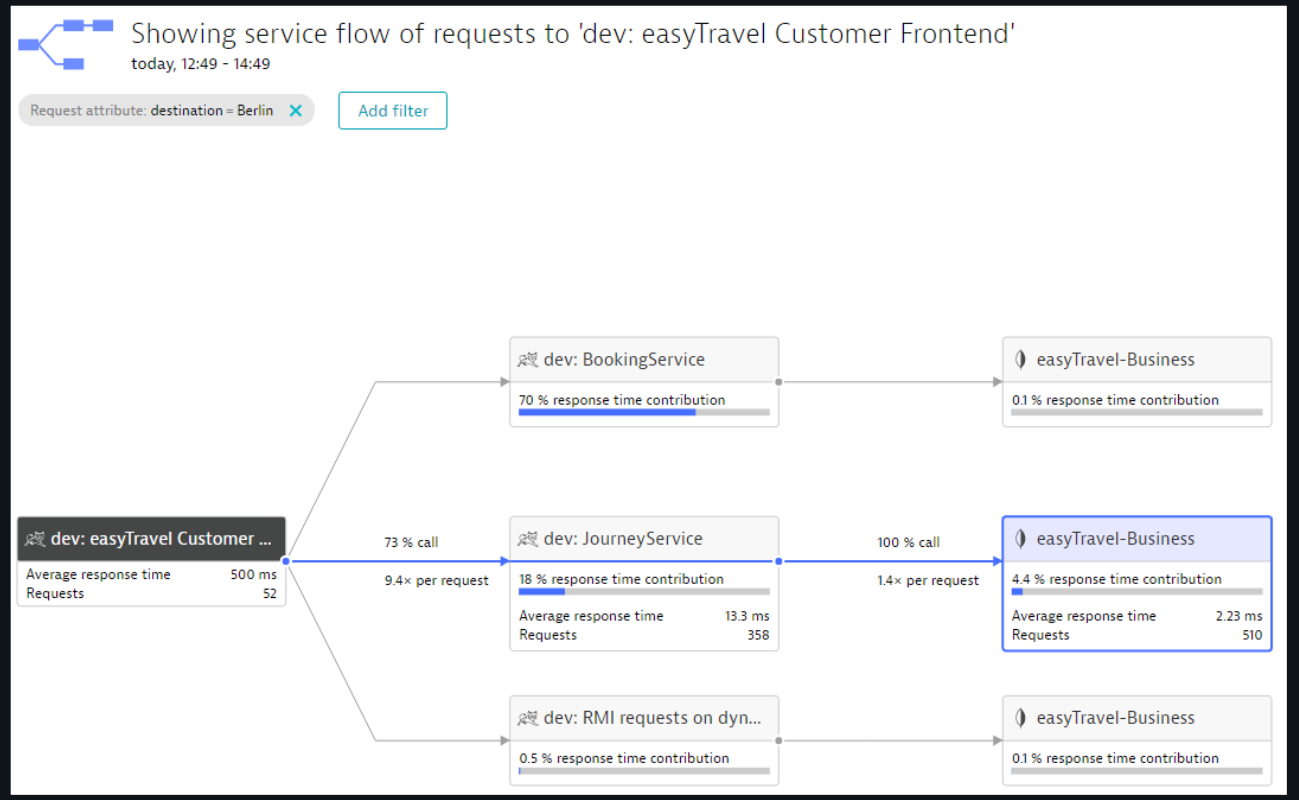
Viewing Request Attributes

- You can see the values by expanding any attribute row



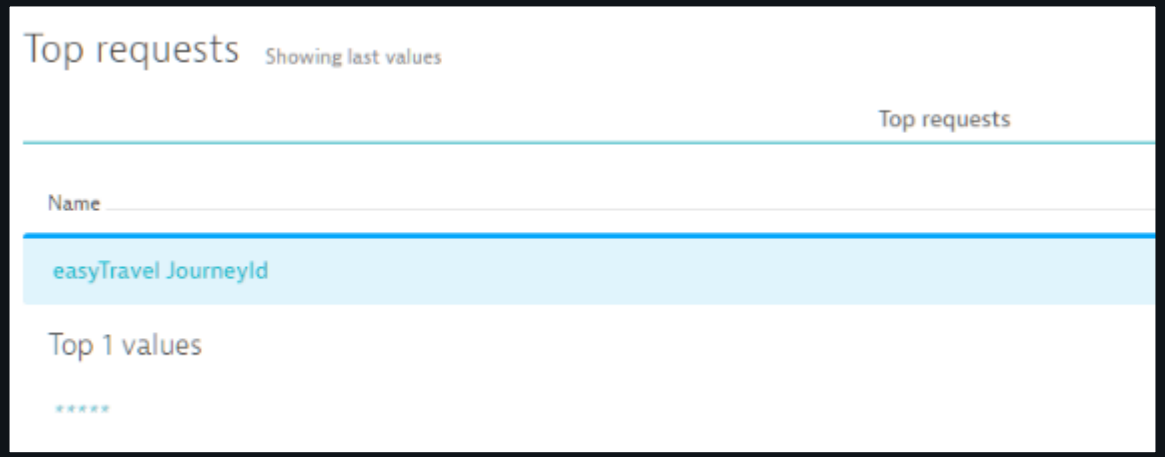
Viewing Request Attributes

- Request attributes can be leveraged across all service analysis views
- The service flow is filtered with the request attribute key/value pair destination = Berlin



Viewing Request Attributes

- Because request attributes can include confidential values, Dynatrace makes it possible to hide sensitive data from certain user groups
- To define or edit a request attribute, users must have the Configure capture of sensitive data permission
- If the confidential data check box is selected for a Request attribute, only users who have the View sensitive request data permission will be able to see the values of the attribute and use the attribute as a filter

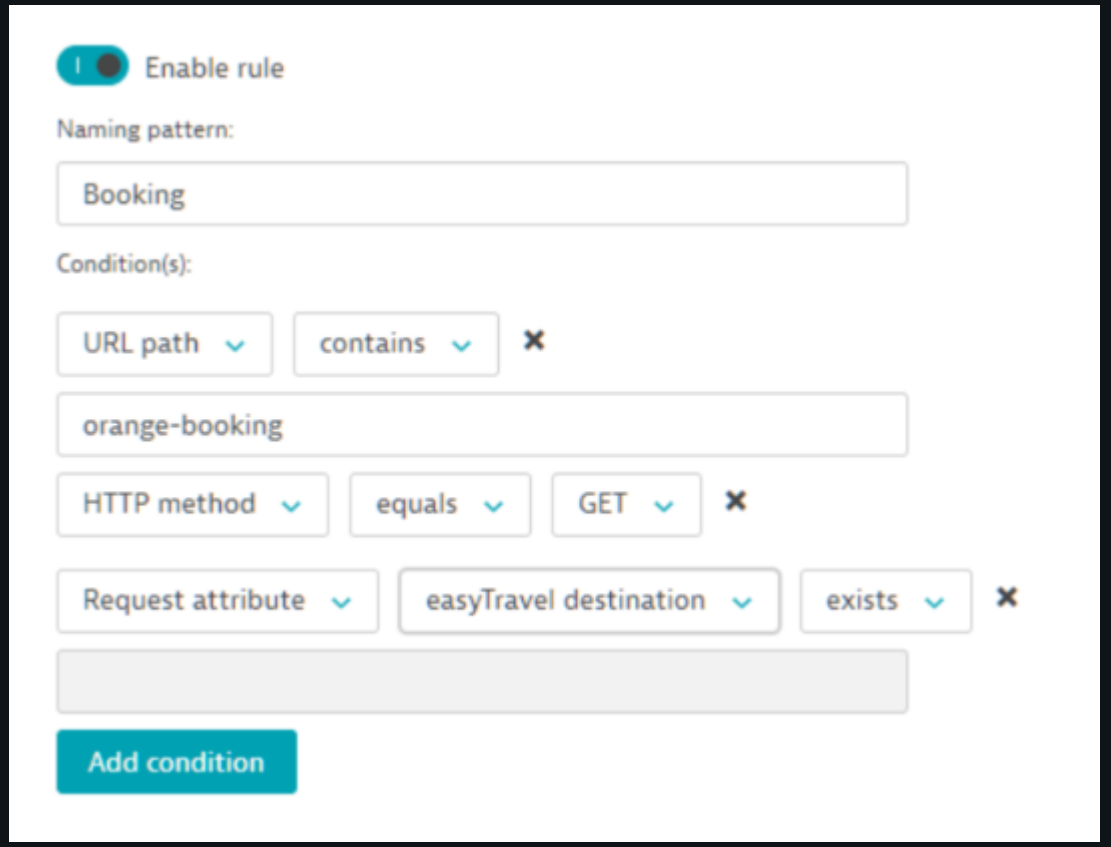


Request Naming Rules

- You can use request naming rules to adjust how your web requests are tracked and to define business transactions in your customer-facing workflow
- Request Attributes enables you to capture even more context around your requests and to use this to slice and dice through your data

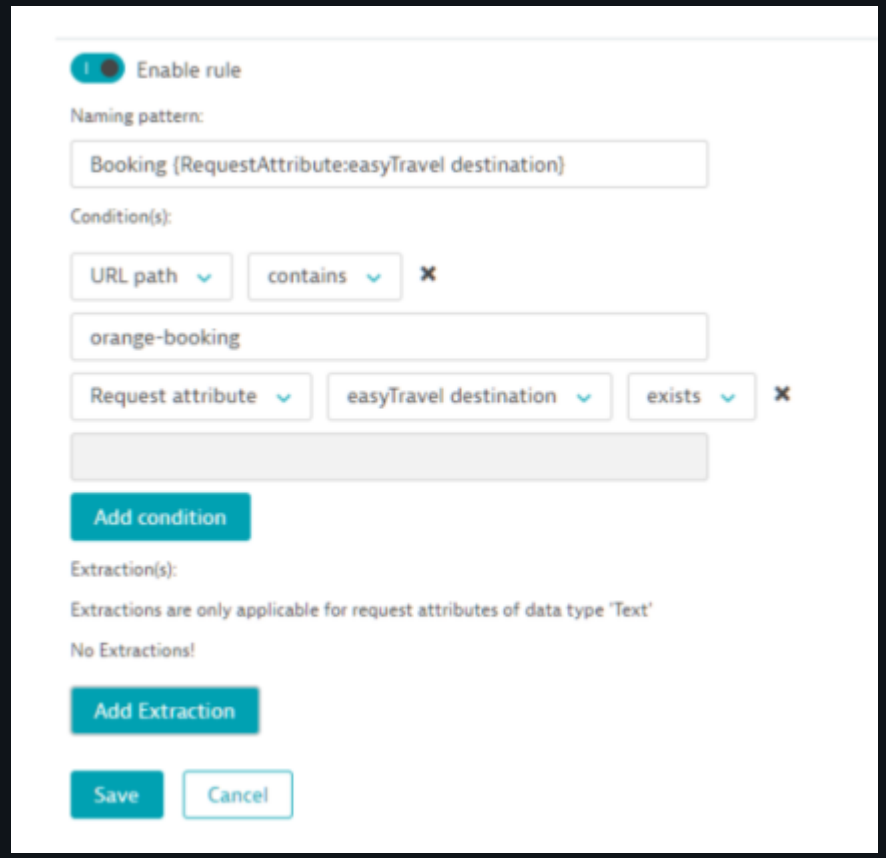
Request Naming Rules

- Define a set of conditions that represent the criteria of your business transaction
- The first condition specifies that:
  - the URL path must contain the string orange-booking
- The second condition specifies that:
  - the request must be an HTTP method of type GET
- The last condition specifies that:
  - the request must have the attribute easyTravel destination.

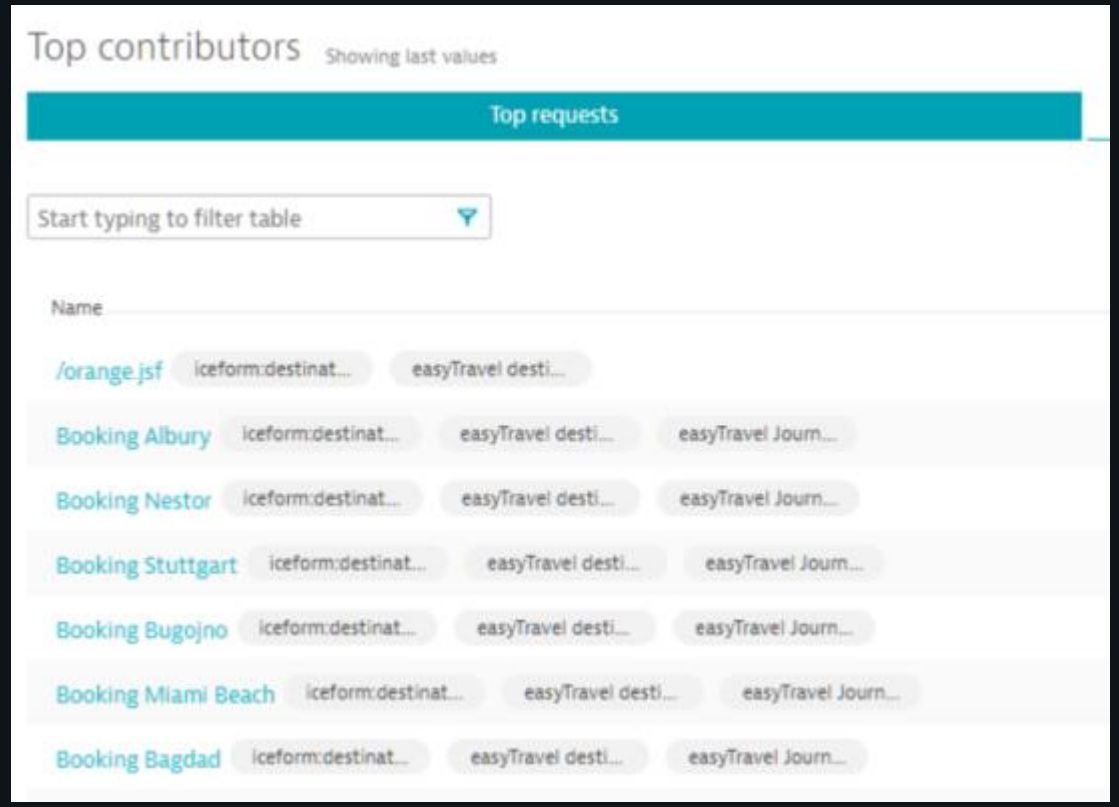


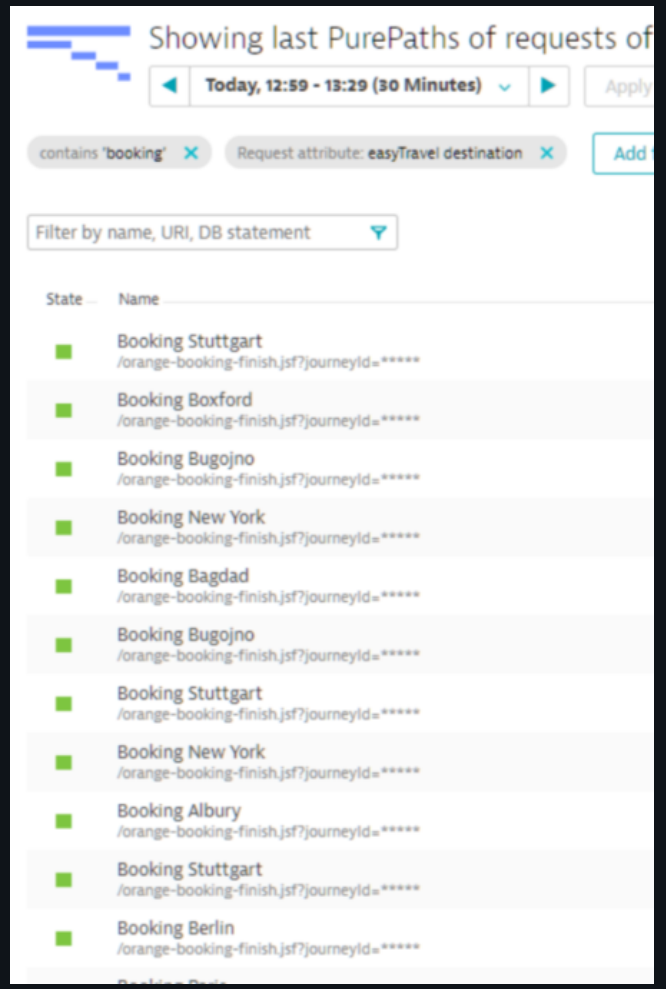
Request Naming Rules

- You may decide to include the value of a request attribute as part of the request name, to create intuitive variant names for this request
- Using the request attribute easyTravel destination, you can create a different request name for each unique value of the attribute



Request Naming Rules



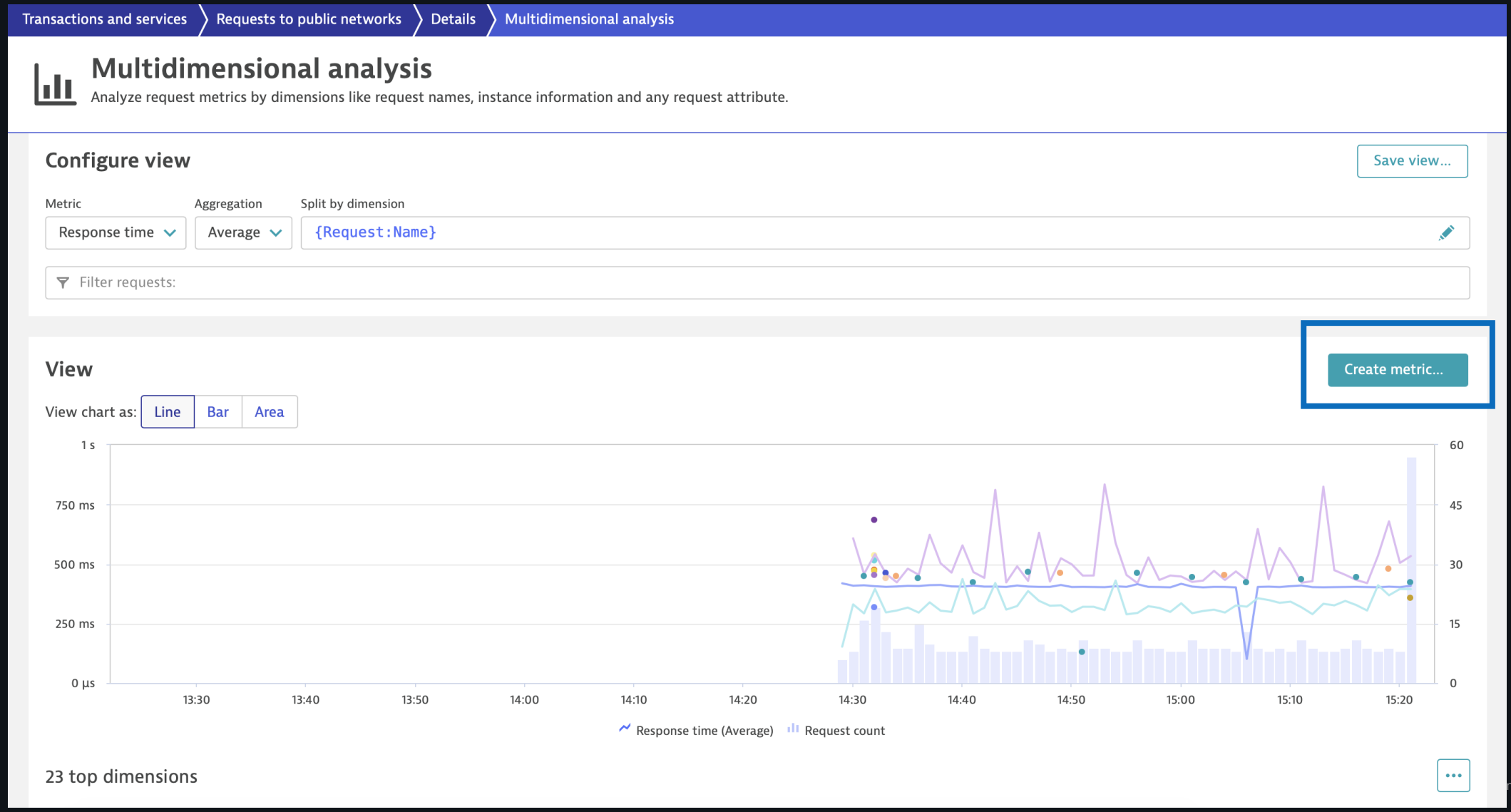


Calculated Service Metrics

Calculated Service Metrics

- Calculated Service Metrics allows us to promote captured values from Request Attributes into full-fledged metrics
- This allows us to:
  - Chart the metrics on a dashboard
  - Export the metrics' timeseries and dimensions over the API
  - Set thresholds and alerts based on the metrics
- Calculated service metrics can be created in two ways:
  1. From a multi-dimensional analysis screen.
  2. From the settings menu.

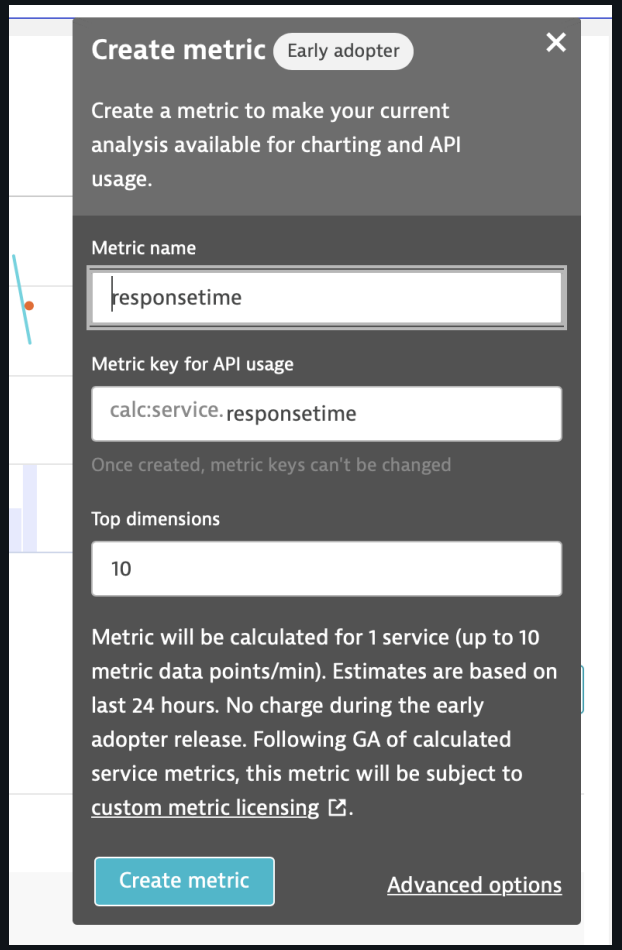
Creating Calculated Service Metrics: From Multi-Dimensional Analysis



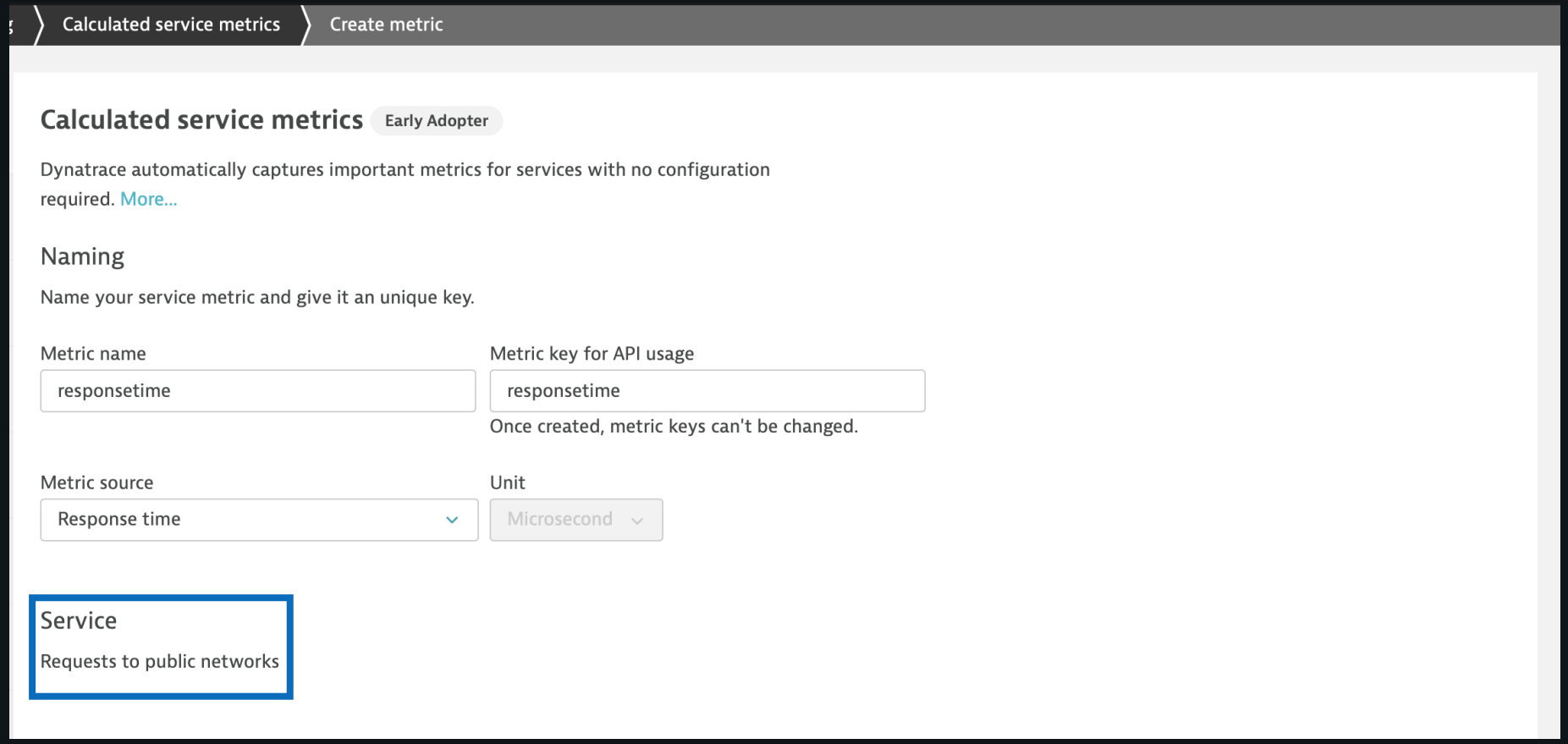
Creating Calculated Service Metrics: From Multi-Dimensional Analysis

- Fill in the name of the metric and the metric key
- The Metric key is used internally to store your metric data, and is also used when retrieving metric data over the API.
  - It cannot be changed after metric creation.

- Clicking ‘Create Metric’ instantly creates your metric based on the filter and dimension defined in the MDA.
- Clicking ‘Advanced Options’ opens the settings screen.



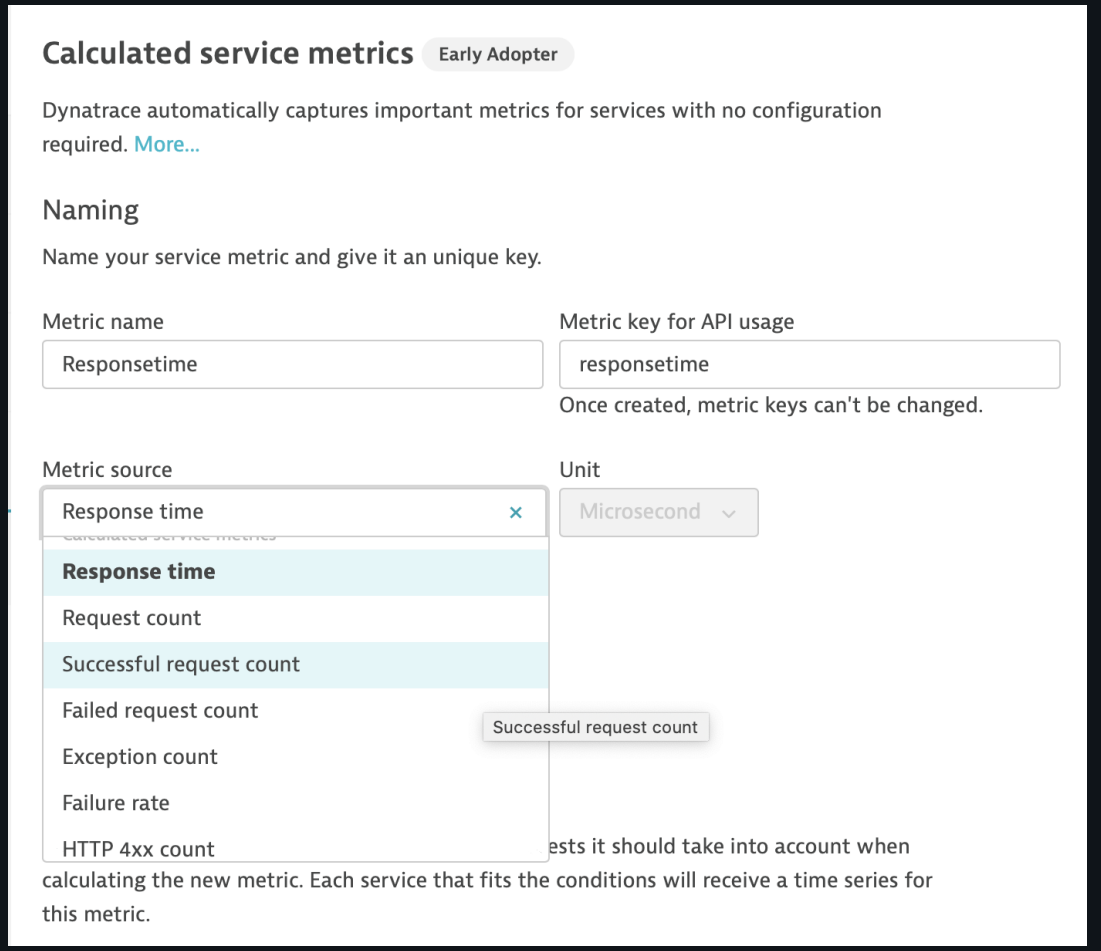
Creating Calculated Service Metrics: From Multi-Dimensional Analysis



Please note that the service cannot the changed if you create the metric this way

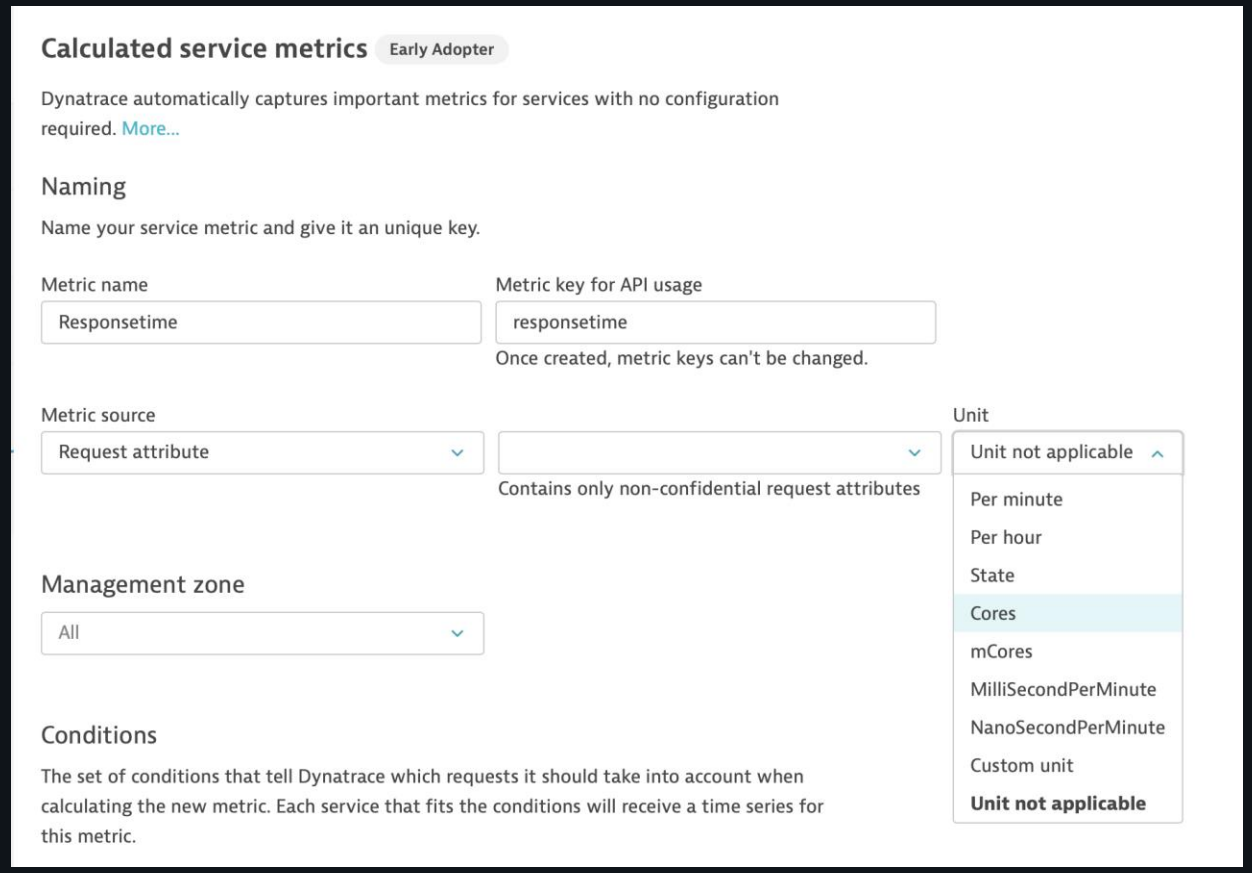
Creating Calculated Service Metrics: From Settings page

- Name the metric and provide the metric key
- Select the metric source
- When selecting a built-in metric source, the unit that Dynatrace uses is selected for you.



Creating Calculated Service Metrics: From Settings page

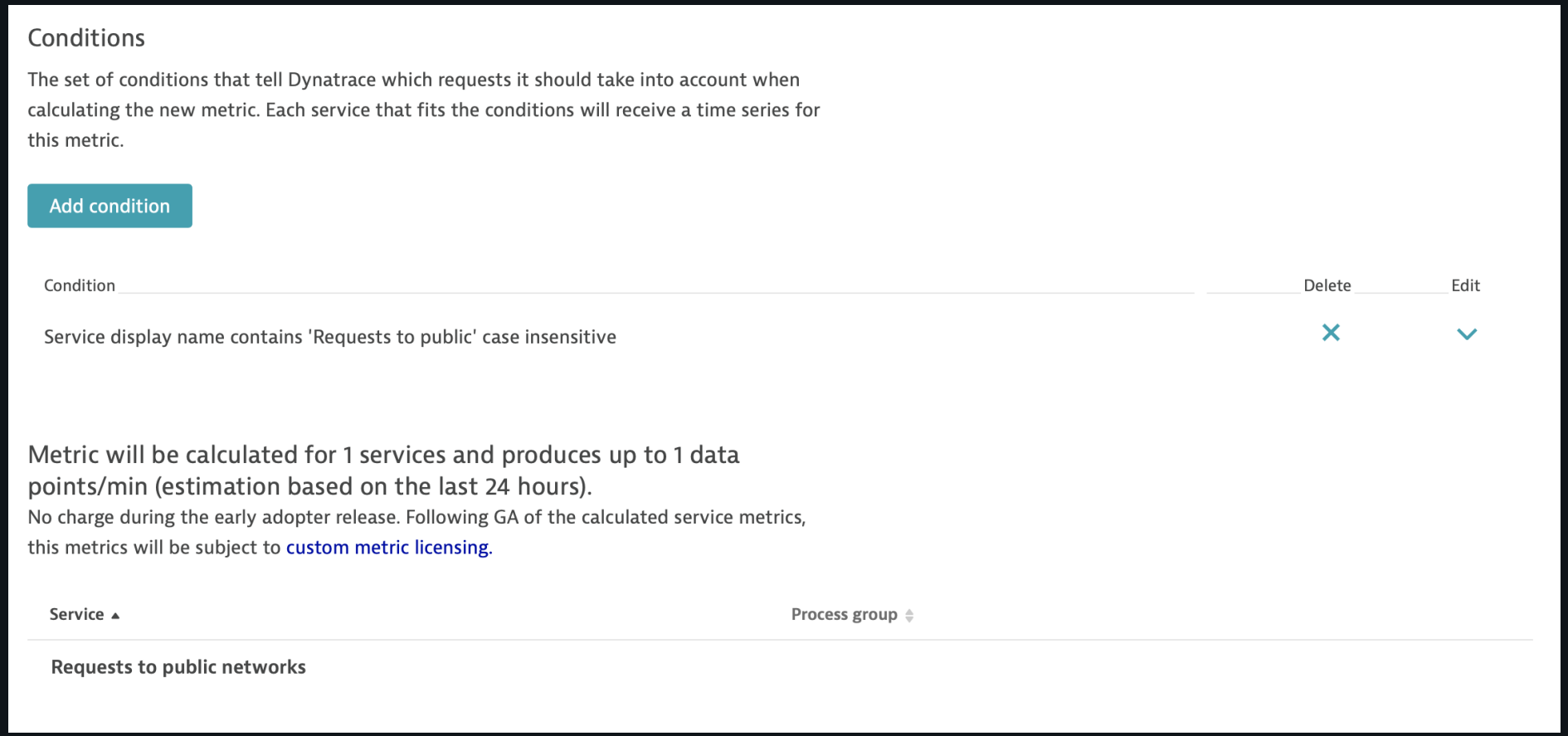
- When you want to use a request attribute, you can only use non-confidential attributes.
- You also have to select the unit, what your metric represents



Creating Calculated Service Metrics: From Settings page

- A metric can belong to at most 100 services. To reach this number, you must apply at least one filter (even if your environment has less than 100 services)
- You must provide a filter to limit your metric. This can be a combination of:
  - Management zone
  - Service Tag
  - Service Name
  - Actor system
  - Azure function function name
  - Azure function site name
  - CTG gateway URL
  - CTG server name
  - ESB application name
  - Public domain name
  - Remote endpoint
  - Remote service name
  - Web application id
  - Web context root
  - Web server name
  - Web service name
  - Web service namespace

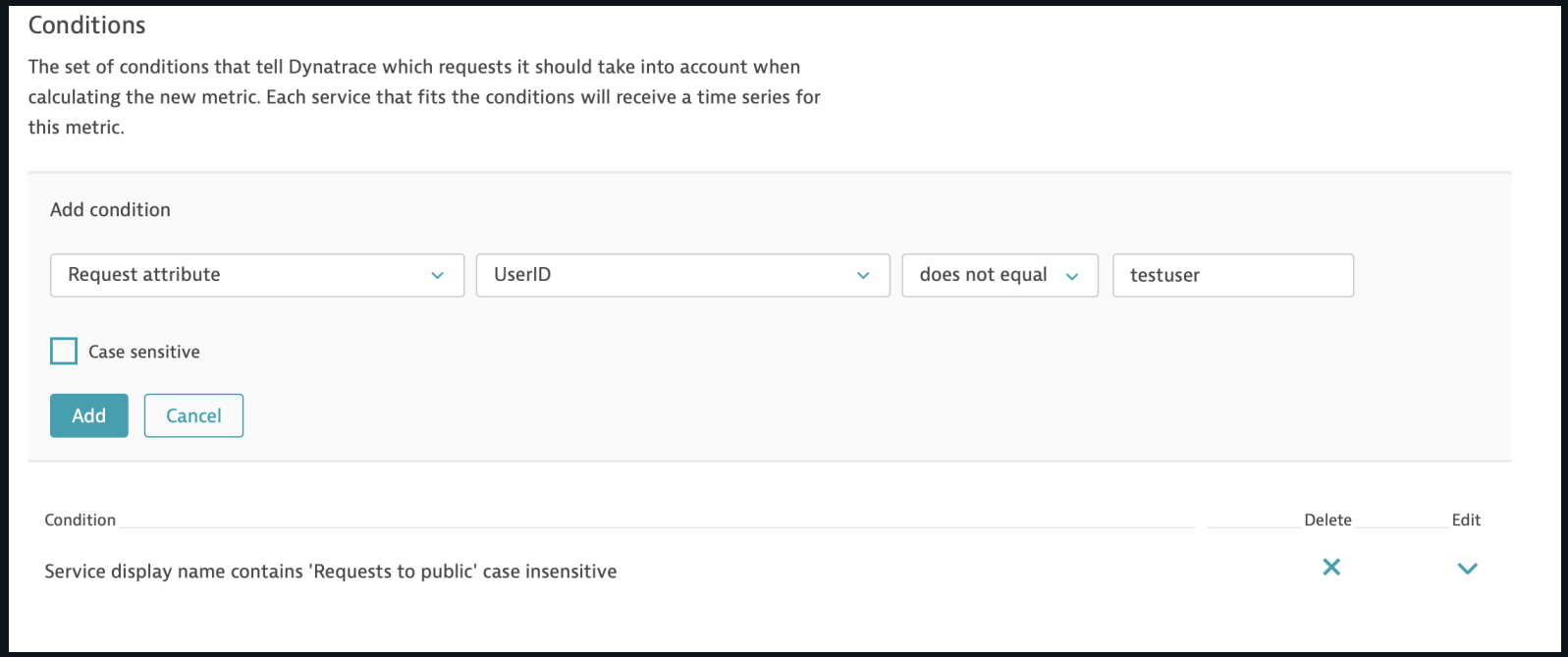
Creating Calculated Service Metrics: Conditions



Once you set a filter for certain services, the list of services this metric would apply to is displayed

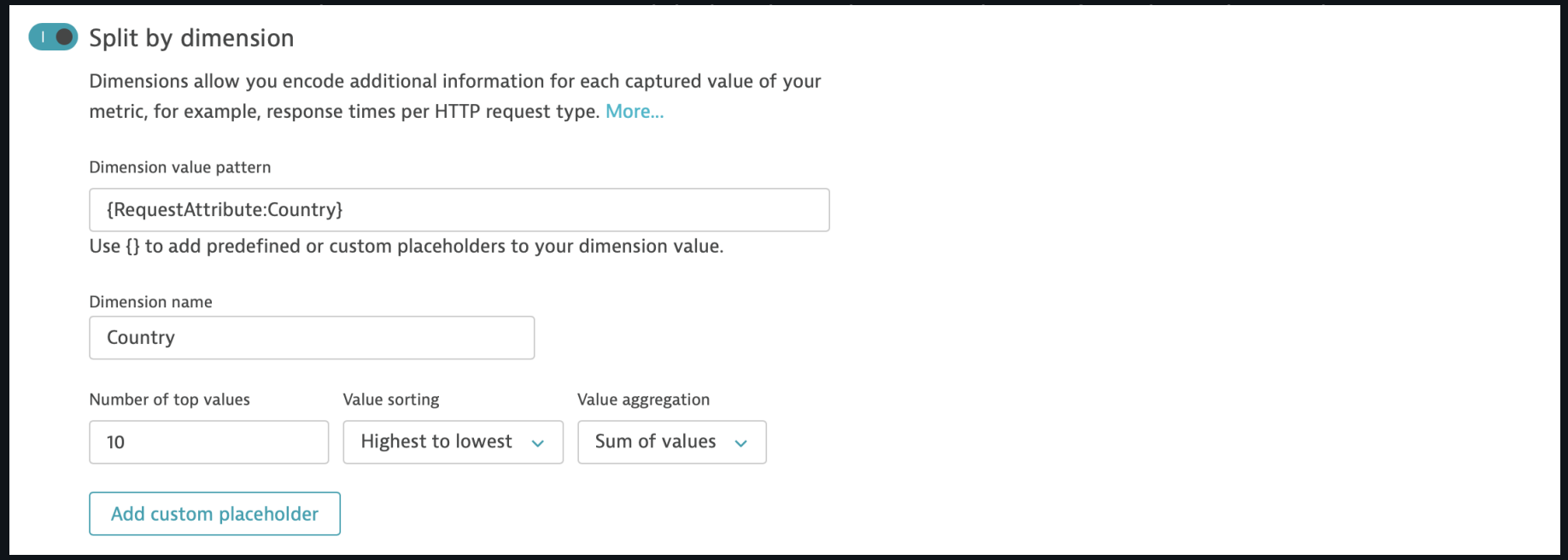
Creating Calculated Service Metrics: Conditions

- The conditions screen can also be used to filter requests inside services.
- Examples:
  - Requests that take longer than 10 seconds
  - Requests with a certain URL pattern
  - Requests where the attribute ‘amount’ is larger than 100



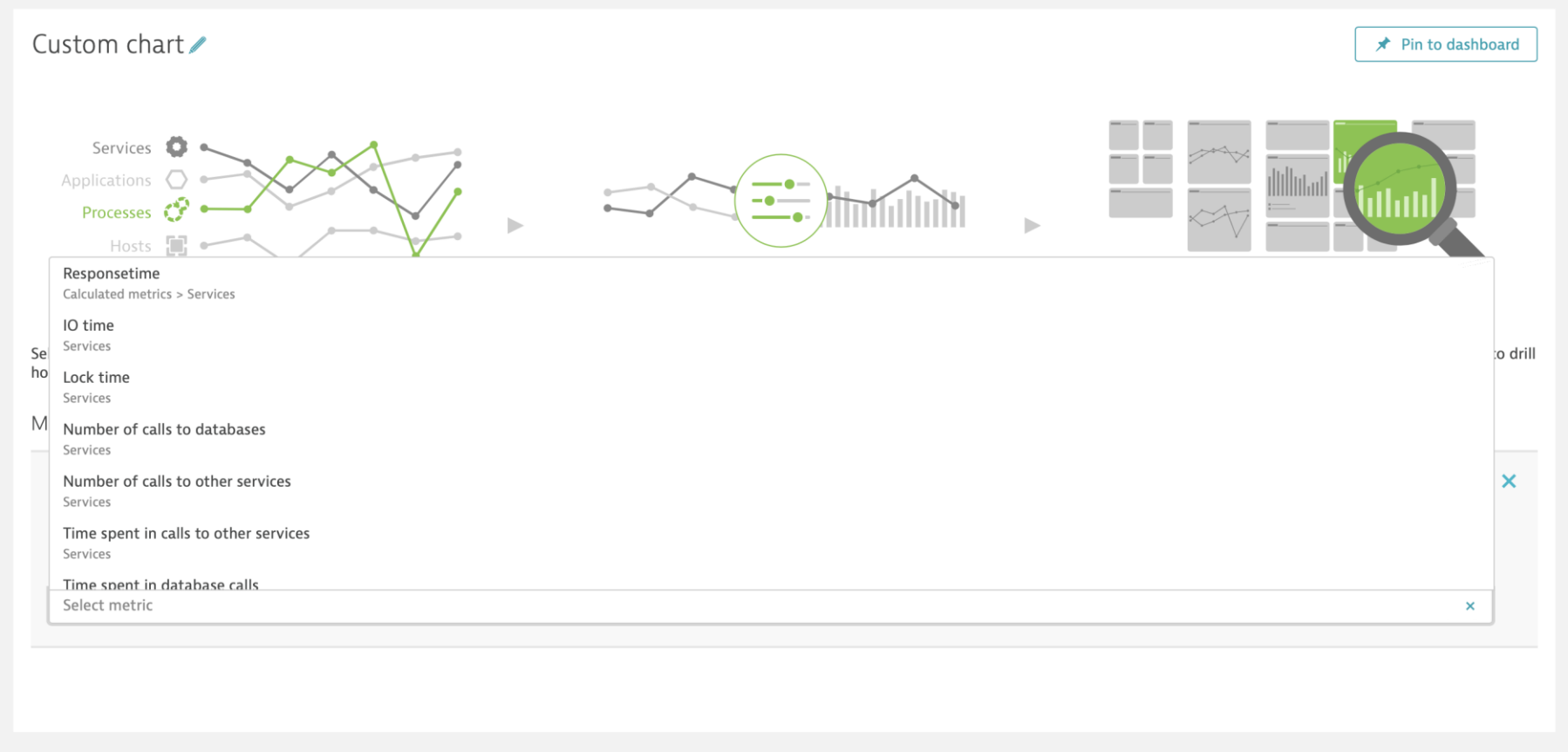
Creating Calculated Service Metrics: Dimensions

- Finally, we can opt to split our metric into dimensions
- This is not always necessaryif I want to see one final number for a service (e.g. total #products ordered), I do not need a dimension as the default metric will display the total number of ordered products.
- If I want to see the total number of products per country, I would create a dimension and split on



Calculated Service Metrics

- Calculated Service Metrics can be accessed in the ‘Service’ portion when creating a custom chart, or in the **v2** version of the Environment API (metrics endpoint)



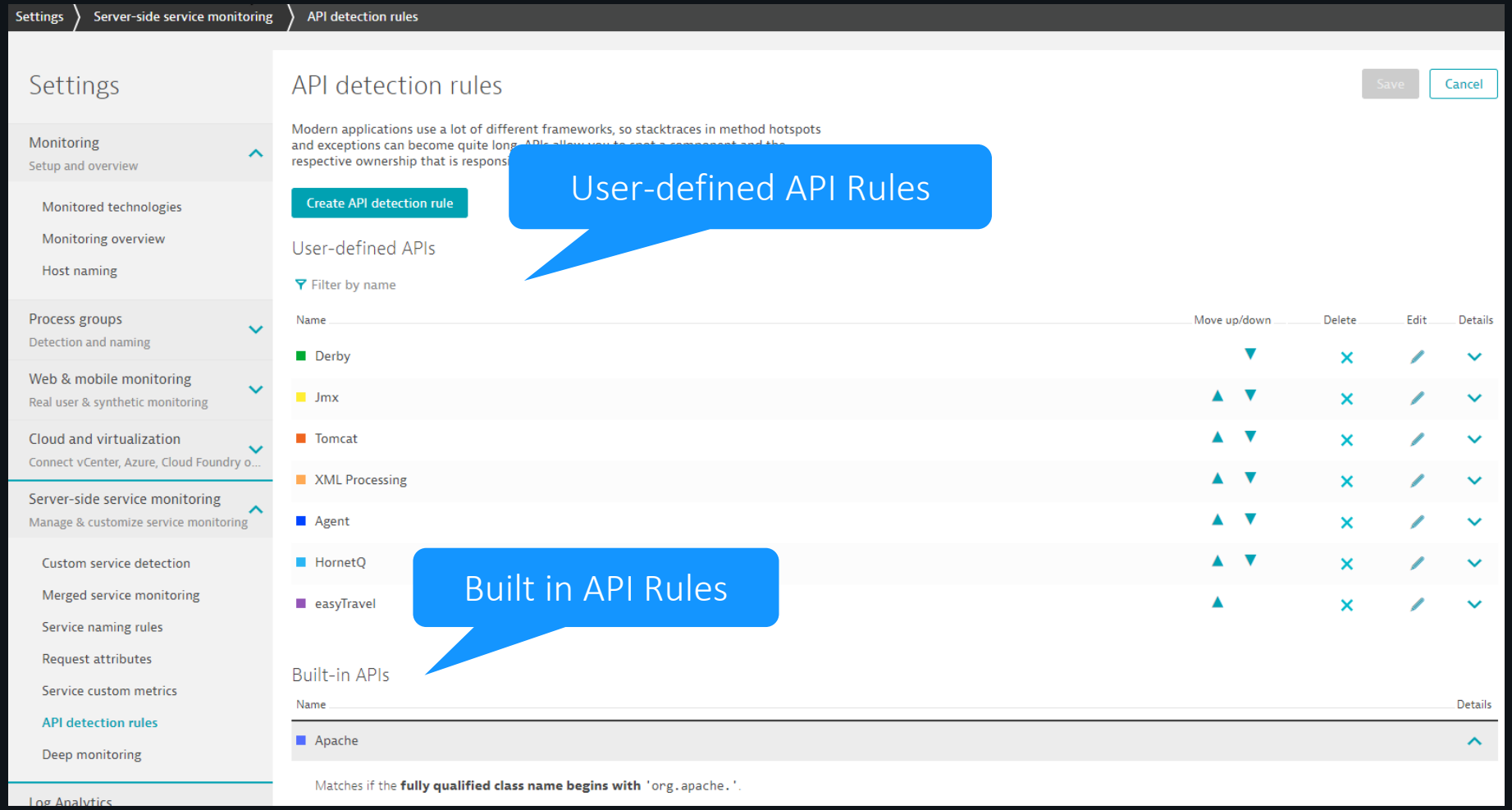
Limitations

- Creating a metric directly from a multidimensional analysis view, binds the metric to that service.
  - If the underlying process group changes (e.g. different hostgroup), the service changes and the metric will no longer collect data.
  - For this reason, we recommend creating them from the settings screen.

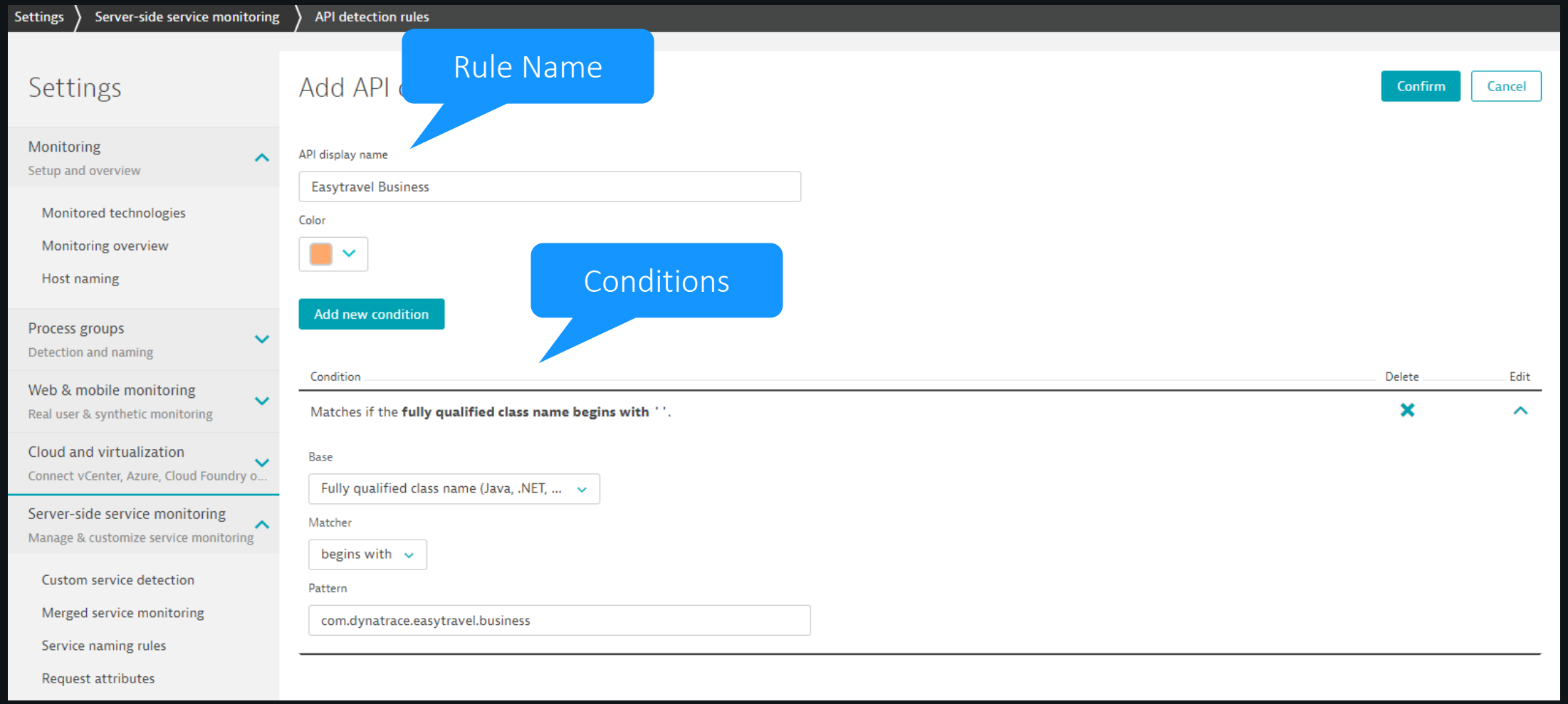
- Out-of-the box, you can:
  - Create a total of 500 calculated service metrics
  - Every metric can apply to a maximum of 100 services
  - A metric can have a maximum of 100 dimensions per service.
    - A metric with 100 dimensions over 100 services, would have a maximum of 10.000 different datapoints.

API Detection Settings

API Detection



Create a new API definition

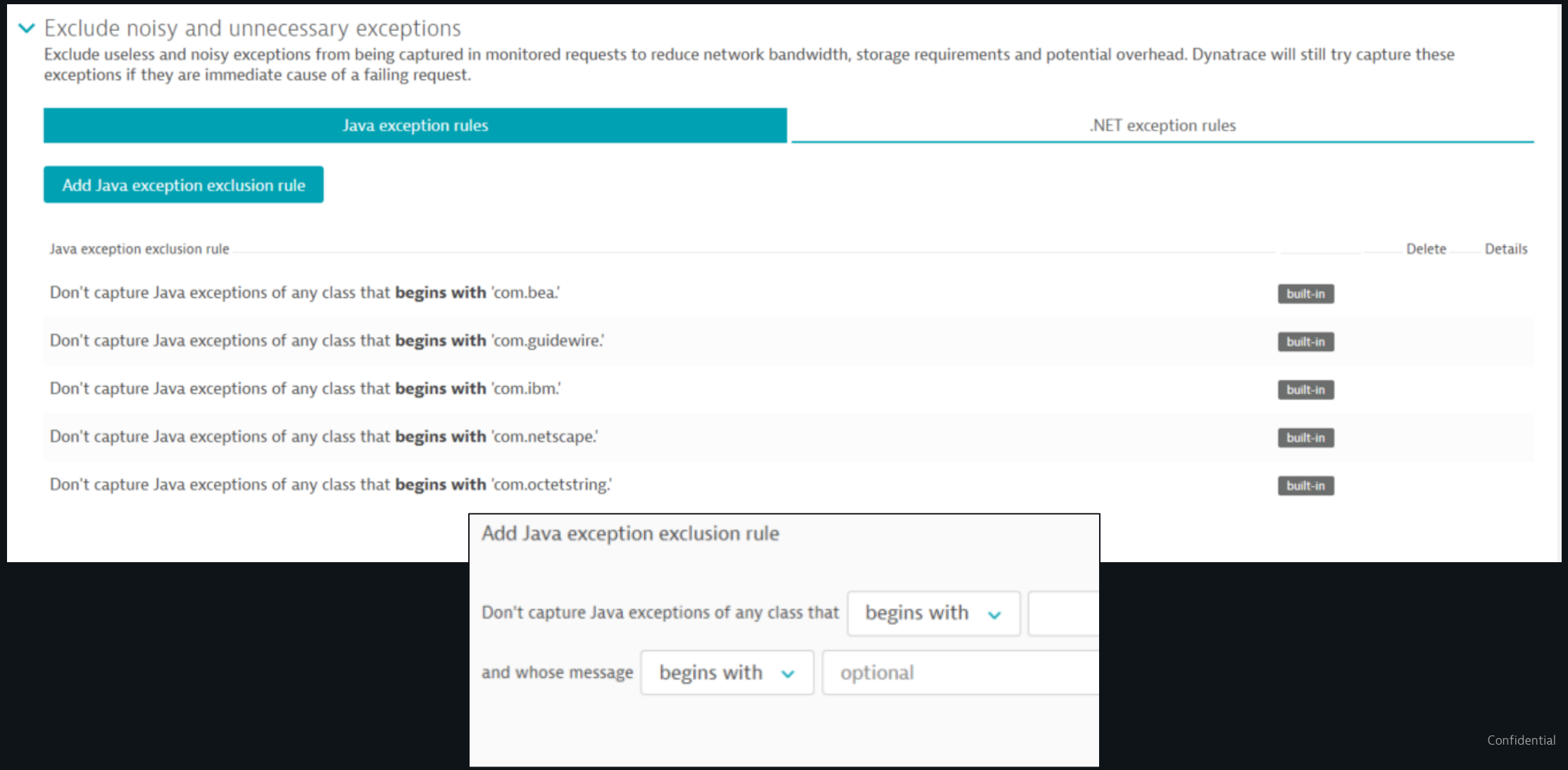


Deep Monitoring

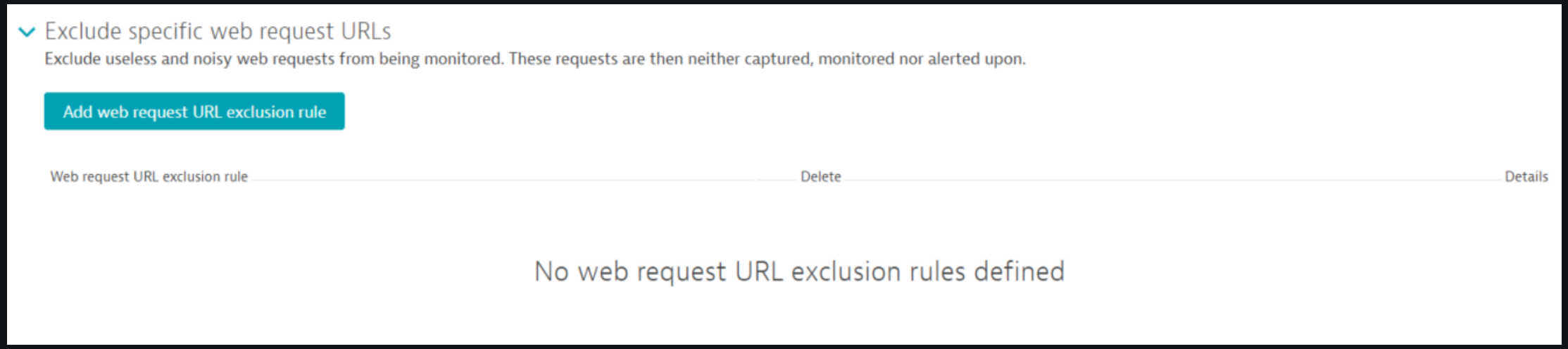
Deep Monitoring

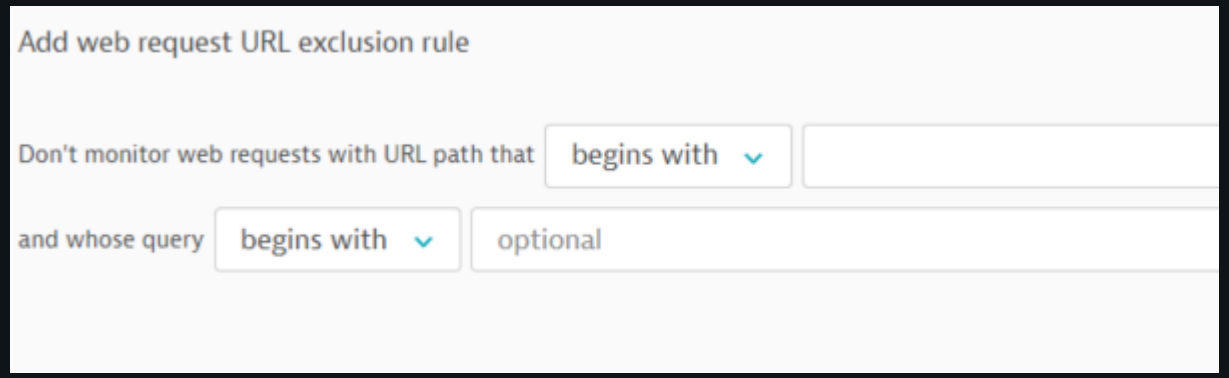
- Dynatrace OneAgent automatically traces each detected transaction end-to-end, capturing all related WebRequest calls, database calls, remoting calls, exceptions, and more
- Use the settings here to exclude certain URLs or exceptions from capture to reduce network bandwidth, storage requirements, misleading alerts, or other monitoring noise
- The troubleshooting settings enable you to disable certain OneAgent functionality at a fine-grained level to facilitate issue resolution when working with Dynatrace Support

Deep Monitoring

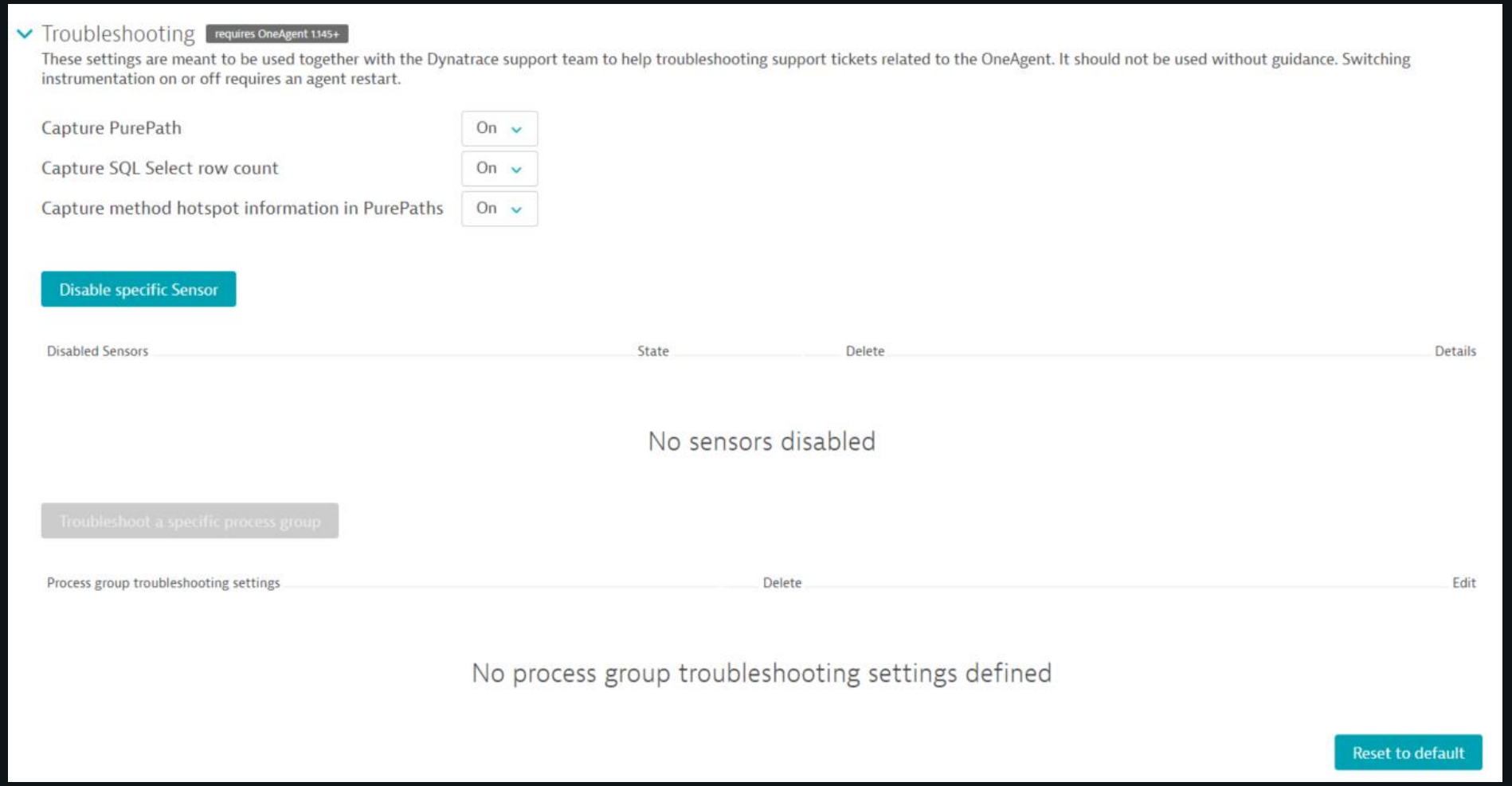


Deep Monitoring



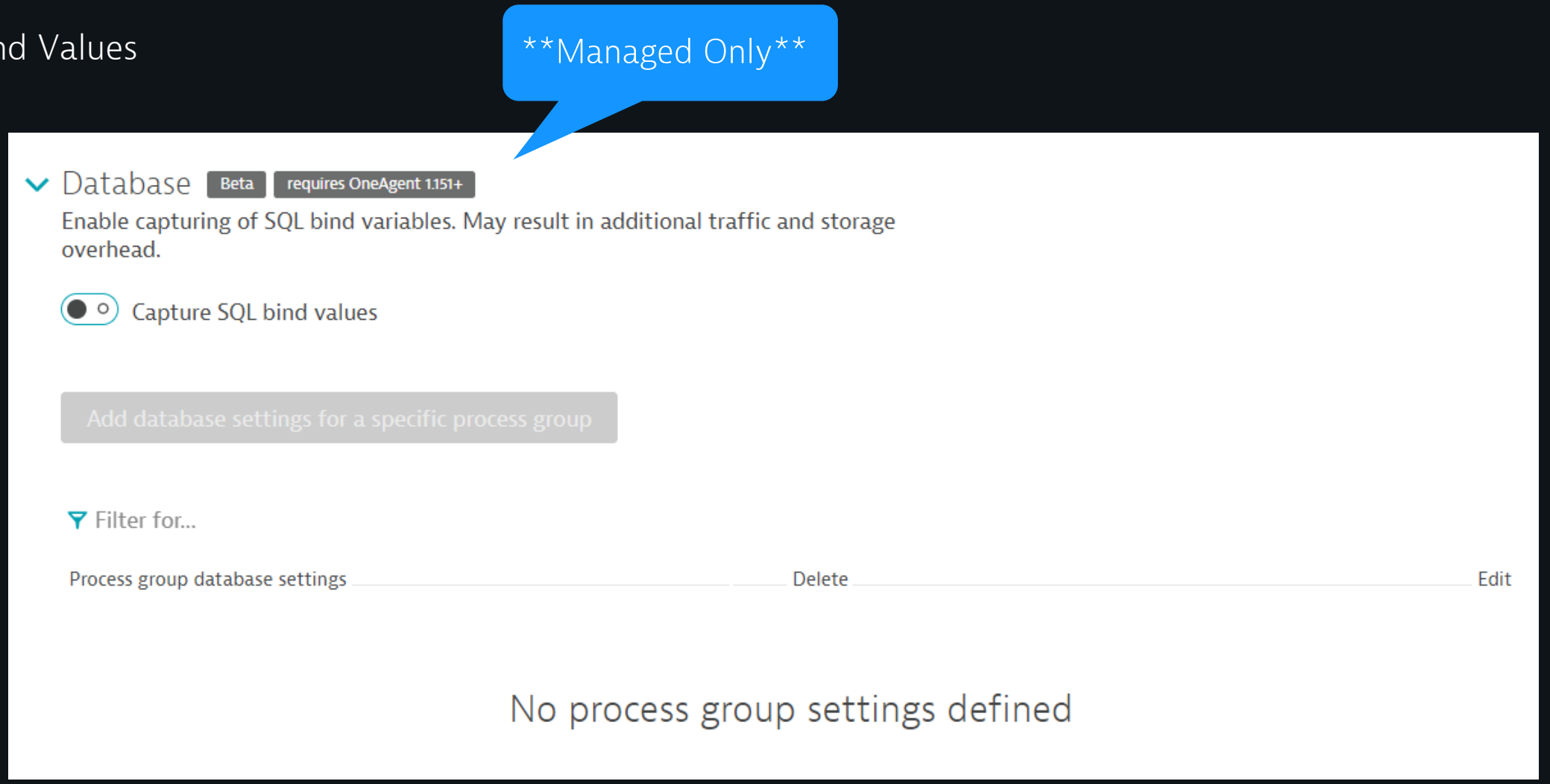


Deep Monitoring



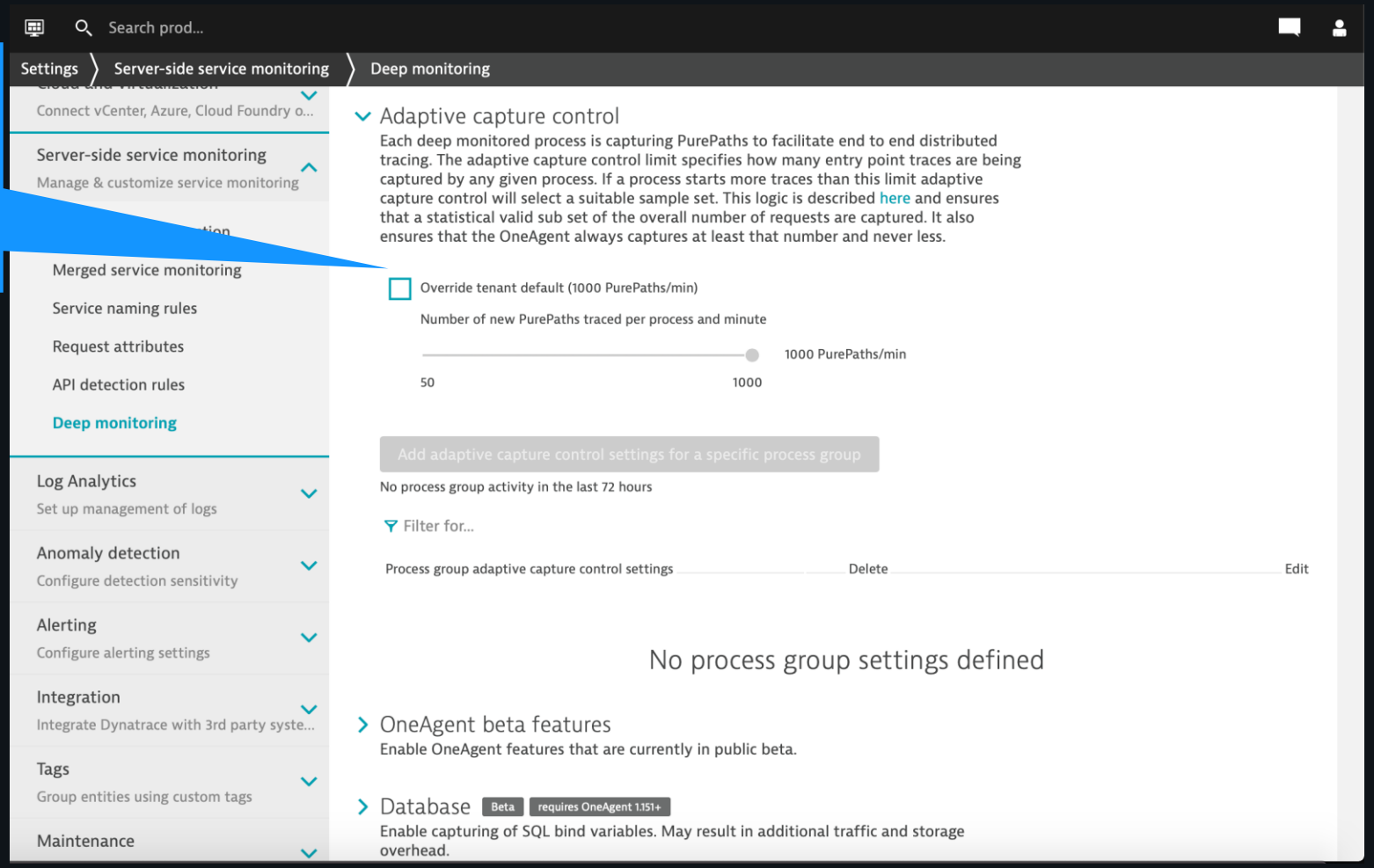
Deep Monitoring

- Bind Values



Adaptive Capture Control

Control the number of entry points captured per Environment



Questions?

Simply smarter clouds