Service-Oriented Data Structure

Author:
Cor Winkler Prins
CEO - 4me, Inc. - July 2019

Service-Oriented Data Structure

Where most service management tools talk a lot about services in their marketing collateral, in the tool itself there is not a separate record type for services. Organizations are expected to register their services as configuration items. These configuration item (CI) records are then flagged in the tool to indicate that they are not really configuration items, but services.

This is problematic primarily because it stops organizations from being able to break down their management reporting by service. To illustrate this, consider how an incident is registered. When the incident is initially reported, it is often not known which CI is causing it. At this point the incident is related with the affected service. After a specialist has resolved the incident, this specialist is expected link it to the CI that caused it. This then overwrites the link with the service. This, in turn, removes the ability for the organization to generate simple management reports such as 'Time Spent on Incidents by Service'.

4me is the first of a new generation of enterprise service management tools that does not center on the CMDB (the configuration management database). As enterprises rely more and more on SaaS solutions, the CI information becomes less and less relevant. Modern SaaS vendors push out new releases of their software on an almost daily basis. Their enterprise customers no longer see the need to register the version of the software in their CMDB. They are not able to know which servers this software is running on. Most SaaS providers now auto-scale their infrastructure. Keeping the server information up to date in a CMDB is useless for their customers. Their customers do care very much about the service, though.

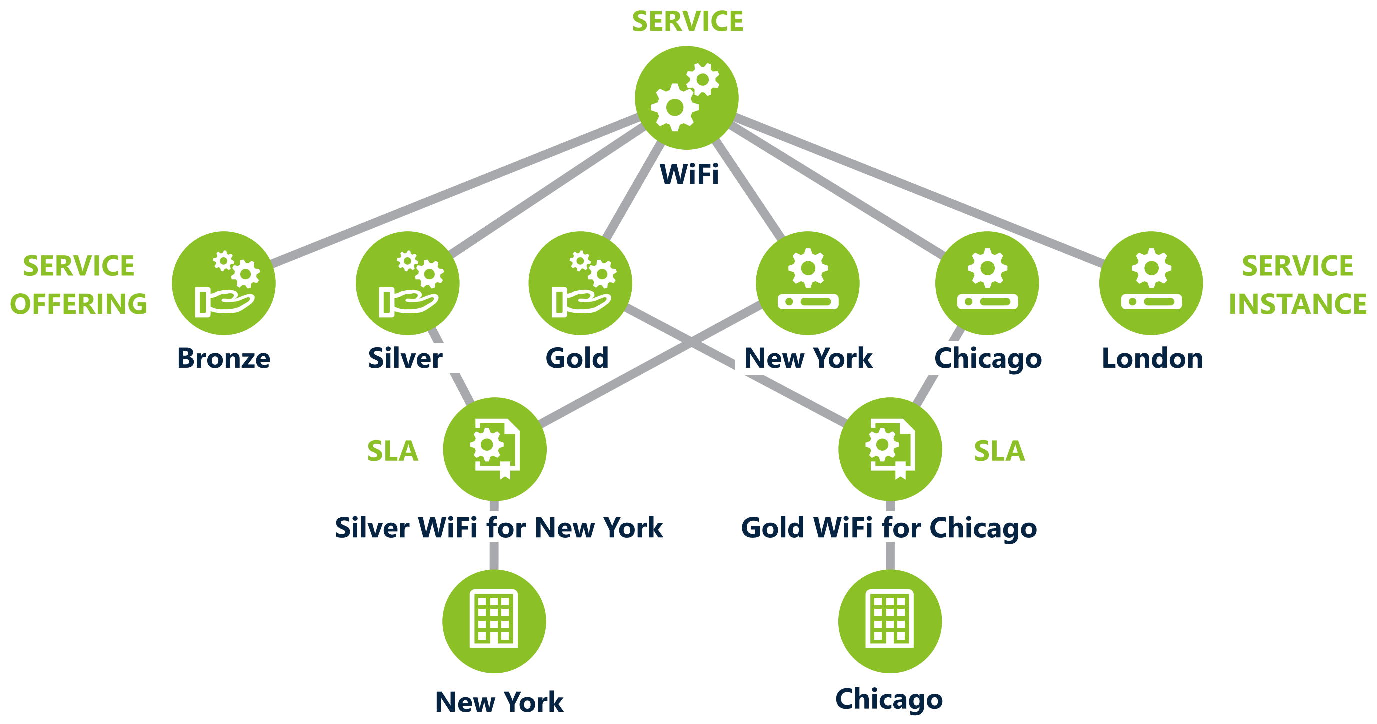
In 4me everything revolves around the record type 'service' and being able to provide accurate, real-time, management information that can be broken down by service. To enable this, simply adding the record type 'service' to the data structure of a service management application is not enough. That is because multiple instances of a service may exist. For example, an organization may have an SAP production environment, as well as a QA environment. Both of these environments provide essentially the same functionality, but they need to be monitored, and reported, separately. When the QA environment was down for a few hours, this should not show up in the SLA reporting that is delivered to the customers that rely on the SAP production environment.

Similarly, the enterprise may provide its employees with WiFi access. But when the WiFi is down at the New York headquarters, that outage should not be presented to management at the manufacturing facility in Chicago. The organization has one WiFi service, but separate instances of this service in each of its buildings. So apart from the record type 'service', there is also a need for the record type 'service instance'.

Because the organization may decide that the WiFi at the manufacturing facility should be supported 24x7 and that an outage at this plant needs to be resolved within 2 hours, while the WiFi at the headquarters needs to be support from Monday through Friday from 7am until 7pm.
These service level objectives could be defined (along with other objectives like the response target for each impact level, availability, reliability, supported standard service requests, etc.) in two separate SLAs, but the enterprise may have facilities all over the world, several of which require similar service level objectives for their WiFi service instance.

That is why 4me separates these objectives from the SLA records. In 4me the objectives are documented in a separate record type called 'service offering'. Rather than specifying all the details of an agreement in each SLA separately, these details can be defined in a service offering. For example, an organization may have a service offering 'Gold WiFi' for all its manufacturing facilities that operate around the clock, and a 'Silver WiFi' service offering for other important facilities like the organization's HQ. A 'Bronze WiFi' service offering can be added for less important facilities.

Once these 3 service offerings are in place, it becomes very easy to activate a new service level agreement (SLA). An SLA record is then essentially just a link between a customer organization, the service instance that this organization uses, and a service offering that defines all the SLA's targets.



To summarize, a proper service-oriented data structure for a service management application includes the following record types:

- Service
- Service instance
- Service offering
- Service level agreement

Services from External Providers

But 4me takes this one step further. When the WiFi is outsourced to an external provider, the enterprise could register all the service instances that this external provider manages on behalf of the enterprise. That allows the enterprise employees to submit requests to the IT department when there is an issue with one of the WiFi networks. The enterprise can use this information to track how well the provider is supporting the WiFi service.

But the managed service provider (MSP) also needs to register these service instances, and all other network environments they manage for other customers, in its service management tool. That makes it possible for the MSP to track the quality of service they are providing. When a request comes in from a customer they can link it to the affected service instance, which causes the request to be assigned to the team that is responsible for it.

What 4me allows an enterprise to do is to connect with its external providers. If an MSP also uses 4me, the MSP can register the SLAs for the enterprise customer. The enterprise customer can then accept these SLAs and specify which sites or organizations the SLAs cover. This ensures that the enterprise can link incidents to the provider's service instances without having to register these service instances; they can see these service instances in their 4me environment. And when there is an issue with one of these service instances, the enterprise can register an incident and link it to the affected service instance. This causes the incident to be assigned to the provider, without the need for any complex integrations.

About 4me

4me is an enterprise service management (ESM) solution for seamless collaboration between internal, external and outsourced teams.

4me is the first enterprise service management application specifically built to support the Service Integration and Management (SIAM) approach. 4me makes it possible for all internal departments, like IT, HR and Facilities, to work together seamlessly with each other, as well as with the managed service providers to which some services have been outsourced. In addition to supporting the ITIL processes, 4me also provides fully integrated knowledge management, time tracking and project management capabilities. For enterprise employees, 4me is the Self Service app that is always there for them whenever they need some help.