



TECHNICAL
OVERVIEW

The Visionael OSS Suite

Integration & Customization Suite Version 10

April 2010



Table of Content:

[Integration in the Visionael Suite.....3](#)

What's an ESB?.....3

The SPI Integration Pattern.....3

[Customization and integration using Systems Orchestrator4](#)

[3rd party integration examples.....5](#)

Fault alarms enrichment.....5

Network activation integration6

[Steps to implement and deploy an SPI implementation.....8](#)

Integration in the Visionael Suite

The Visionael OSS Suite has been specifically designed to handle integration scenarios with other OSS applications in a straight-forward fashion, and without compromising the upgrade path of the Visionael Products. Integration is therefore performed “outside” of the product scope, to avoid instrumentation of core product functionality. Such integration and customization can be carried out on two different levels, either by SPI wiring and implementation, or by using the Visionael Systems Orchestrator product. This document covers customization and integration scenarios addressed by both approaches.

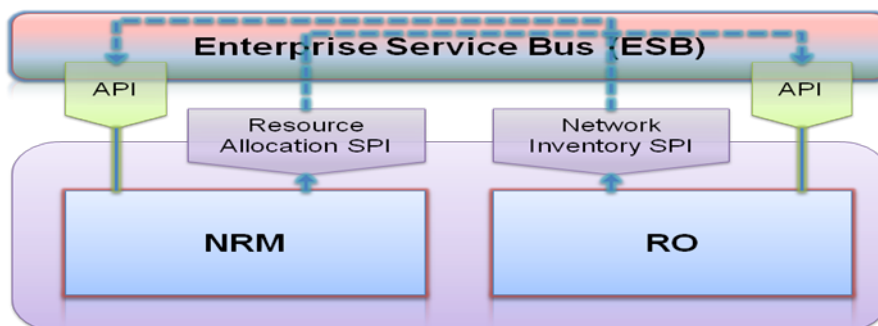
What's an ESB?

An ESB, or Enterprise Service Bus, is an architectural construct as well as software implementations fulfilling that construct. Visionael uses the Apache Service Mix ESB implementation to expose API's and carry out integration and customization. Using the ESB, API's may be exposed using a myriad of different transport protocols allowing legacy applications to integrate seamlessly.

The SPI Integration Pattern

As the Visionael products are decoupled and deliberately are designed to allow for 3rd party integration they don't integrate directly towards other applications in a 1-1-mapping. Instead, each application that either requires, or has the option of, integration with other *functional domains* defines Service Provider Interfaces (SPI's). An SPI constitutes a “perfect API” from the stakeholder applications point of view. It defines all methods and operations desired from the functional domain allowing the applications, in essence, to disregard how those requirements are really met. Using the ESB, the SPI's are wired towards application API's. This means that several applications can fulfill the SPI of another application, completely transparent.

Applications may even be integrated in what would seem a traditional 1-1-mapping, where they in reality have their domain SPI's wired to the opposite application API. One such example is NRM and RO, where NRM only defines its requirements within the resource allocation domain in its resource allocation SPI. If a deployment is using RO, that SPI can be wired to the RO API (out-of-the-box). RO in turn can use NRM for resource tracking. This way Visionael provides a true COTS offering, in allowing the use of the Visionael products integrated out of the box, but readily prepared for integration with other domains and products.



The SPI integration pattern also provides better support for scenarios where the functional requirements



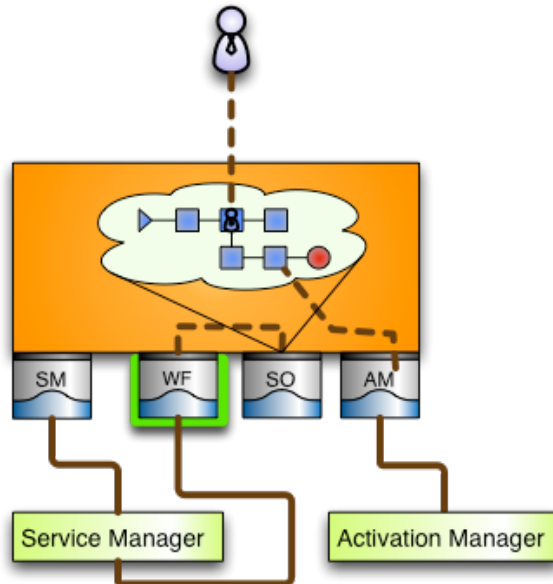
of an application need to be fulfilled by several others.

Customization and integration using Systems Orchestrator

There are cases where a workflow oriented approach to integration fits better than static implementations of SPI's. Such cases are for example:

- Long-running business processes where a process may have to hibernate
- Asynchronous processes that communicate with 3rd party systems
- Processes that require human interaction, task lists, escalation etc.
- Interaction that requires network communications protocols such as SNMP, telnet and SSH
- Open workflow definitions that allow on-going customization and versioning

When deploying modern OSS applications such as the Visionael product in a legacy OSS environment with a predominance of manual processes the transition to a completely automated environment may have to be incremental. In such scenarios SO can be used to include legacy manual steps in any process.



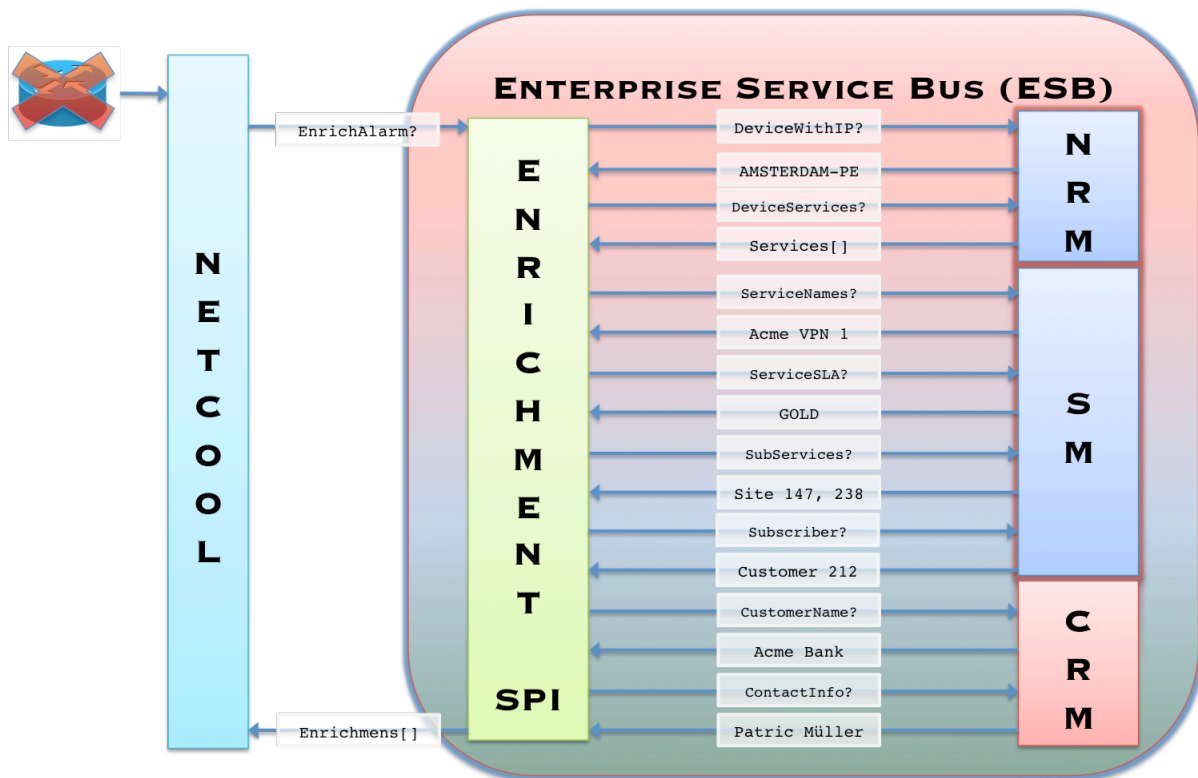
An example of a scenario where SM invokes a workflow, which carries out human interaction (such as delivering a CPE etc.) and then carries out activation using AM.

3rd party integration examples

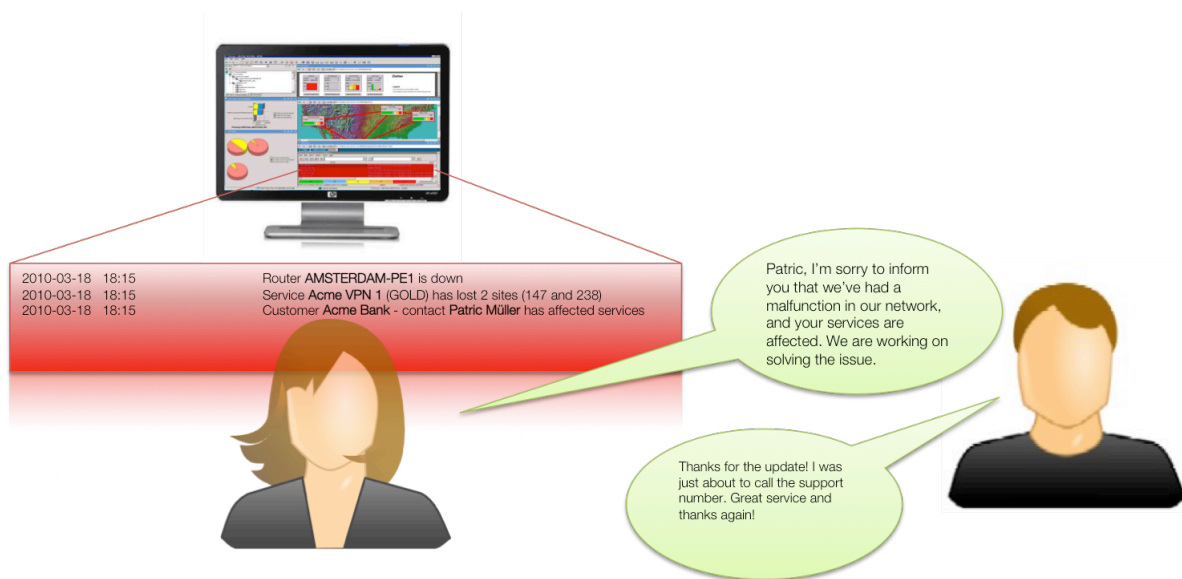
Fault alarms enrichment

A common integration scenario is a setup where a fault surveillance system, such as IBM Tivoli Netcool, enriches generated alarms with information from other OSS and BSS systems, such as network inventory systems, service provisioning systems and CRM systems. If a device fails in the network it is of great value to get information on which services that are affected, what SLA's they endure, and perhaps even contact information to the customer subscribing to the service.

The following example illustrates how the Visionael Suite and the ESB can be used to fulfill such a scenario. Please be advised that the alarm enrichment SPI is not part of the Visionael offering, and the SPI integration pattern is not in any way necessary to carry out integration, but it provides a better SOA and secures a future proof setup.



Netcool detects that a router is unreachable, uses an implementation of alarm enrichment SPI that in turn is wired to NRM, SM and an arbitrary CRM system. From Netcools point of view this is all transparent, it only uses its perfect enrichment API.



The NOC staff sees the alarms and contact the affected customers. One could even envision an integration solution where another system would send out SMS messages based on confirmation from the NOC staff.

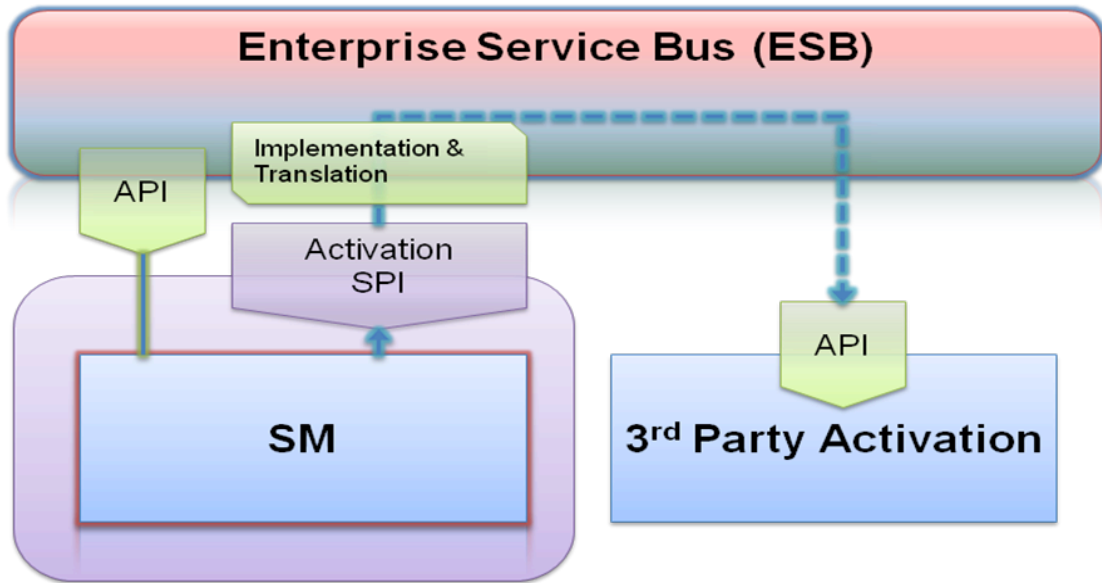
The basic NRM API allows any SPI or third party application to enrich alarms transparently. By navigating through different API methods any information contained within NRM can be used to enrich alarms. Likewise any other API in the Visionael Suite can be used to fulfill requirements of any 3rd party application.

Network activation integration

The Activation Manager product in the Visionael Suite covers network activation. However, there may be existing products responsible for network activation in a given OSS environment. In the Visionael Suite, there are mainly two products that have requirements on the functional domain of network activation, namely NRM and SM. NRM can be configured to allow network engineers to propagate changes in the network repository to the network, and SM is in most scenarios setup to activate provisioned services in the network automatically. NRM realizes this functionality through its Automation SPI. The Automation SPI is not only intended for network activation, but can be used for such. SM however, has a specific Activation SPI defined.

The following example will cover how to integrate Visionael Service Manager with a 3rd party network activation product. The SM activation SPI is fairly simple. It merely expects SM to pass on the references to any services or resources that it wants activated in the network. The default implementation of the activation SPI would query SM back for information on these services and resources generate a complete network resource representation expressed in VisION format (JSON) and pass that on to Activation Manager.

If one would want to replace the standard implementation of the activation SPI one would have to create similar logic, and possibly translation logic, to send of information to a 3rd party activation application.

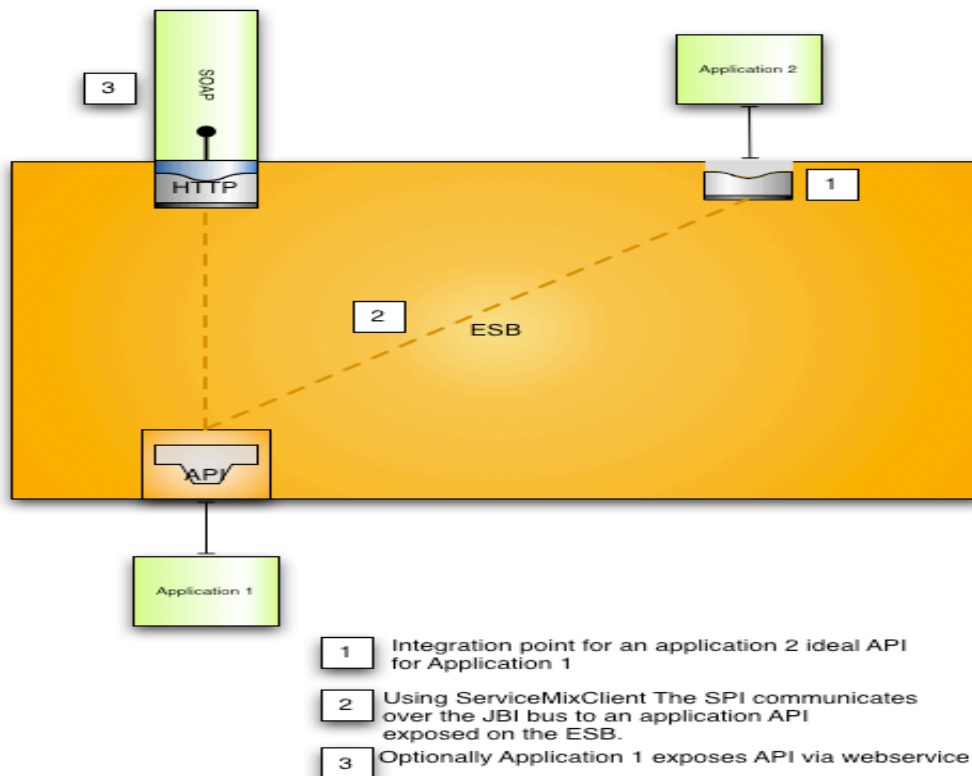


A typical translation scenario would be to generate an XML representation of the services and resources and pass that on to the activation application.



Steps to implement and deploy an SPI implementation

In the following scenario Application 1 (app1) has exposed an API on the ESB. Application 2 (app2) wants to integrate with app1. App2 has defined an SPI that represents an ideal integration API for a functional domain, like the one app1 is providing, and installs that as a service on the ESB that app2 will communicate with. This new ideal API is implemented and translates those ideal API calls to app1's API across the ESB using the ServiceMixClient provided by ServiceMix. This client is a convenience wrapper for intercommunication across the ESB (JBI Bus).



ServiceMixClient example:

```
ClientFactory factory = (ClientFactory) new InitialContext()
    .lookup(ClientFactory.DEFAULT_JNDI_NAME);

ServiceMixClient client = factory.createClient();
QName service = new QName("http://namespace", "servicename");
EndpointResolver resolver = client.createResolverForService(service);
client.send(resolver, null, null, "<operation><param>value</param></operation>");
```



This page is intentionally left blank.

**Corporate Headquarters**

Visionael Corporation
201 San Antonio Circle, Suite 235
Mountain View, CA 94040
Phone: +1-650-963-0960
Fax: +1-650-941-4456

Technical Support / Training Center

Visionael Corporation
9717 East 42nd Street, Suite 200
Tulsa, OK 74146-3618
Phone: +1-918-770-4452
Fax: +1-918-663-1456

R&D Branch office Sweden

Visionael Sweden AB
Forsta Langgatan 22
SE-413 28 Goteborg
Phone: +46 31 360 99 00
Fax: +46 31 144 288

For more information
Go to

<http://www.visionael.com>

