Etere
a consistent system

VIMN Custom Workflows

VIMN requested a way to call heterogeneous systems through Web Services (SOAP and REST). The purpose of the request is to facilitate integration with other systems through a configurable generic component, which will be useful for many integration scenarios, while there will be other more complex integrations that will require a custom integration, which is out of the scope of this document.

Viacom

Overview
VIMN requested a way to call heterogeneous systems through Web Services (SOAP and REST). The purpose of the request is to facilitate integration with other systems through a configurable generic component, which will be useful for many integration scenarios, while there will be other more complex integrations that will require a custom integration, which is out of the scope of this document.

The initial proposal made by VIMN is to have a generic/configurable web server client that should interact with external services by sending a configurable request, parse the return data and then executes some database action.

Our Concerns
While the idea to have a generic communication component is attractive, the implementation of the architecture as proposed may be very tricky especially if we think about the database actions to perform after a result has been acquired: this point has not been deeply considered but can be the weakest part.

Consider for example a scenario, already depicted, in which etere sends a request to DIVA. Suppose that after that action we have to update metadata for a file. As you may know the entity upon which Etere workflow runs are assets, physical files references are placed in the database as a many to many relations, so there's no way to know which file to update without having additional information. Such information should flow from a workflow block to another. So some context information must be kept in the database.

This means the Etere component must be stateful and not stateless, thus contradicting the fact that the web service component should be kept stateless.

Even if context info are available, there is much to do: to implement a generic update engine we should think of all the possible data you may want to update and all the ways to select a certain information.

If some web service requires a keep alive, there's no way to guarantee that such a method can be called at fixed times on within a given deadline.

If some web service requires some "session information" there's no easy way to store this context information and get it in another workflow block at a later time. This is not a trivial task and requires knowledge of the db and some programming skill.

Our Solution - Overview
What we propose is a different solution that leverages well established and documented technologies, cuts the developing times and enhances system stability, while keeping the system much more open to future enhancements and requirements.
The idea is to use windows powershell, the scripting engine by Microsoft, to perform all the needed tasks.
Powershell by default allows to

Etere a consistent system

■ Call any kind of web service and web method and get the result
■ Connect to any database engine supported by ADO, leveraging the full power of T-SQL
■ Using any component available in the Windows OS (XML Parsing, file copy, …)

Our Solution - Deeper
We'll add a component that has three basic configuration sections

Sect 1 – Input
The uses select which kind of variables it want to retrieve from the db and pass to the script. The only required ones are the database connection data and the job id, however for the sake of simplicity we may query the db for any field that links to the job/asset, such as a particular asset metadata. We put those data plus custom defined data in variables and pass them to the script

Sect 2 – Script
The user simpli puts the script and we'll execute

Sect 3 – Output
The script returns data back to the caller, we must receive at least an error code and the all the data the user wants and we may also do database update for simply-linked data such as asset metadata.

This should solve many scenarios without complex scripts and without need to know the database internals, however for complex scenarios where complex data retrieval is required, all the database work may be done in the script itself without the need for VIMN to ask Etere to develop additional components. The powershell script is powerful enough to support stateful webservices, keep-alive scenarios, you can also perform additional tasks such as XML parsing, ftp download, file copy, http requests and so on.