

Integrating Requirements Management with IP Management

By Vishal Moondhra

Introduction

Increasing design and verification complexity in SoCs is leading many teams to adopt industry standard best-practices for managing their projects.

One of the key components to successfully managing a distributed, complex and time-critical SoC project is Requirements Management. Having a full set of requirements, easily managed and updated for the project makes it much easier to bring transparency and tracking to a project.

Once requirements management is in place, it is also essential to be able to tie requirements back to the context of the IPs being used in the SoC. Since IP management and requirements management are two separate systems, being able to connect these two is critical. This paper describes how a requirements management system like Jama can be integrated into the IP management platform ProjectIC, allowing users to see IP requirements right in the context of their SoC.

The Importance of Maintaining Data at Its Source

When integrating separate systems, it is important to maintain data in the native format required by each system and leave that data in its source. This eliminates the possibility of introducing data errors and also reduces any on-going maintenance required to keep the systems talking to each other.

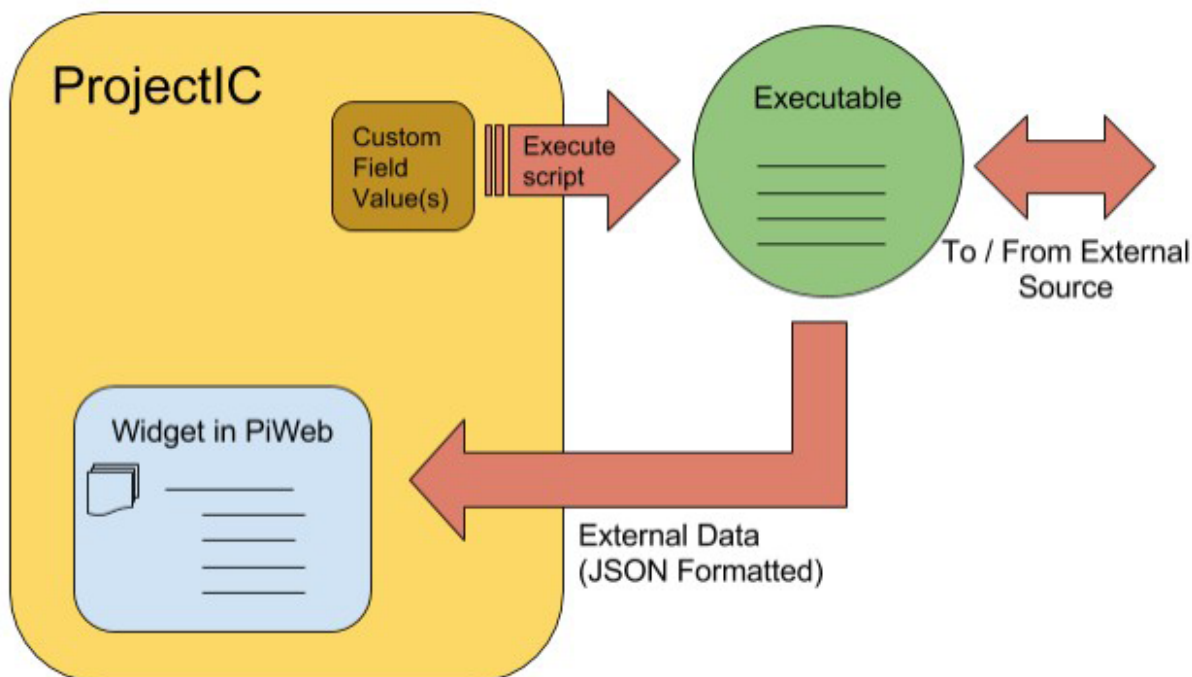
For this reason, the main philosophy of integrating external data into ProjectIC is to continue to leave data in its source, and provide a way to view the data in native formats, but in the context of hierarchical IPs. One example of this is ProjectIC's bug integration, where bugs are maintained in their native tool, but ProjectIC provides a view of the bugs associated with the IP hierarchy in context.

A similar approach can be taken when integrating a requirements management system into ProjectIC as well. Each IP will have its own requirements set up in a tool like Jama, where they may be actively managed/added/deleted etc. Then, from within ProjectIC, users will be able to see these requirements in summary format, extracted on-demand in the IP context.

Setting Up External Data Widgets in ProjectIC

ProjectIC can be configured to set up an external widget in any one of its pages - the home page, the IP page, or the IP View (IPV) page. Generally, a requirements management integration widget would be set up in the IP or the IPV page and displayed in the PiWeb interface within ProjectIC.

In addition to specifying the location of the widget, the user needs to specify an executable script that will be run to populate the widget with its data. This executable is usually a Perl or Python script that extracts the relevant information from an external system and presents it to ProjectIC in predefined JSON format.



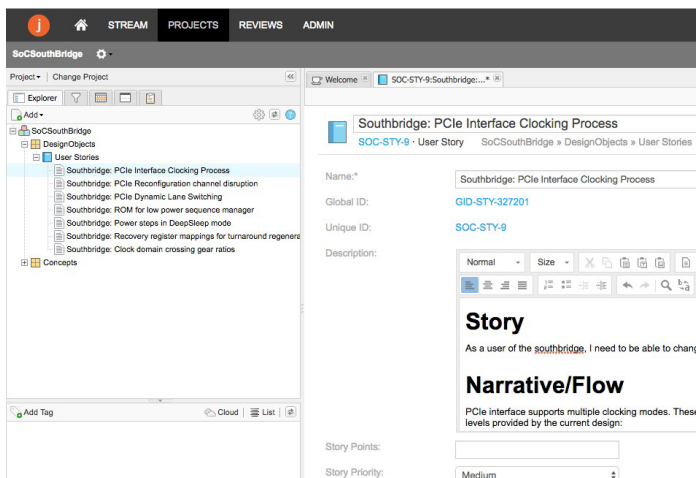
One key feature that enables the integration of this sort in ProjectIC is ‘Custom Fields’. Each installation of ProjectIC can be configured with custom fields. These custom fields extend the definition of an IP and allow users to customize any additional meta-data that they want to attach to an IP.

When setting up the executable script for the widget, users can select one or more custom fields as arguments to it. Thus, when an IP’s page is generated, ProjectIC will automatically pick up the value contained within the custom field for that IP or IPV, and pass it to the executable as an argument.

Sample Integration – Jama Requirements Management

As an example, consider an integration into a requirements management tool like Jama. This tool has a well defined API. Wrappers are available for Python and Perl to extract hierarchical data from Jama.

The following screenshot shows a sample project in the Jama Requirements Management tool.



The aim of the integration is to extract this data and make it visible within ProjectIC, in the context of the IP for which these requirements exist. To pull this data, a script needs to connect to the Jama API and provide an ID (usually an integer). Jama will then respond with a data structure that contains all the details that ID and any of its children.

Step 1: Set up a custom field in ProjectIC for the Jama ID.

Here, we first create a ProjectIC custom field called ‘JAMA_ID’. This can be an integer field or a string field, and is set at the IP level. We will use this as an argument to the executable script in step 3 below.

Once this field has been created, each IP needs to fill this out with the ID of the Jama user story hierarchy

that pertains to this IP. This is usually available from Jama itself, and is in the form of a number.

Step 2: The Executable Script

As mentioned before, each external widget needs to have an executable script that queries the API of the external data source and presents the response back in pre-formatted JSON.

The script below, called ‘jama_if.py’ is a very simple Python based example of what this might look like. This script presents the Jama ID to the Jama API, and returns up to 2 levels of hierarchy back to ProjectIC.

```
#!/usr/bin/env python """
Get a Jama tree up to 2 levels by ID ID
is passed as argument to the script """
from jama import API import json
import sys

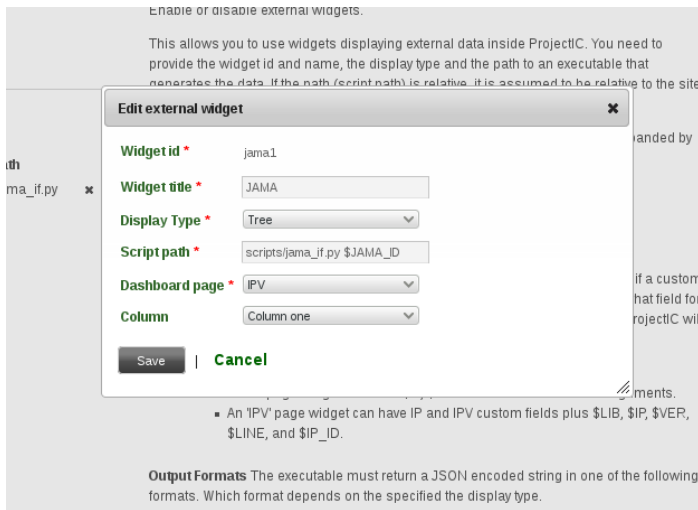
jama_id = sys.argv[1] api = API()
jfunc = 'getChildrenOfItem' item
= api(jfunc, int(jama_id))

res = {}
for i in item:
    res['title'] = i.name
    if i.hasChildren:
        res['isFolder'] = True
        kids = api(jfunc, i.id)
        rkids = []
        for k in kids:
            t = {}
            t['title'] = k.name
            t['isFolder'] = False
            rkids.append(t)
        res['children'] = rkids
    else:
        res['isFolder'] = False
print(json.dumps(res))
```

Step 3: Set up the Widget:

Once we have the script running as needed, we need to create the external widget in ProjectIC. This widget is created on the admin page. We need to provide a widget name, and a path to the script with ‘\$JAMA_ID’ as an argument. Note that if your custom field is named something else, you need to use that instead.

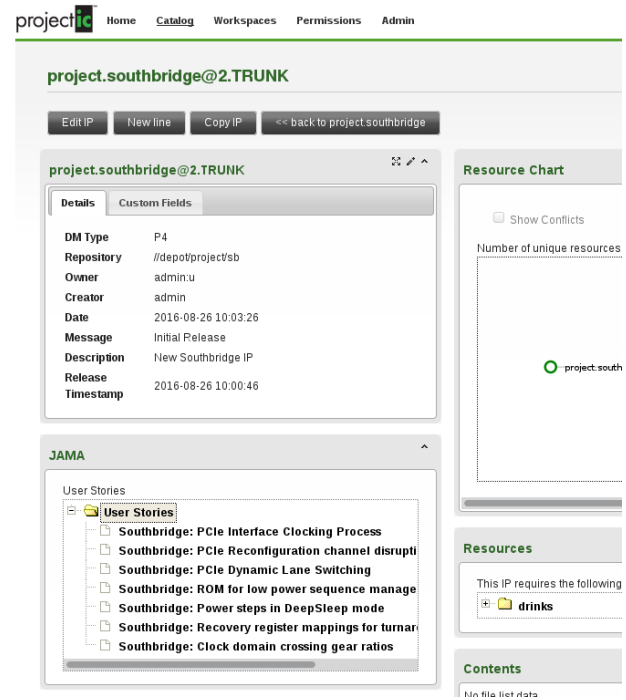




Now we have external data setup to connect to Jama and present the specific requirements of each IP in its own context.

Conclusion

Implementing a requirements management system like Jama with an IP Management system like ProjectIC allows fully traceable hierarchical SoC development during the IP Lifecycle Management process and closes the loop between system design requirements and IP implementation details.



This represents one more step forward to reaching the important goal of implementing IP reuse strategies that reduce development costs, improve time-to-market, and keep semiconductor companies profitable in today's highly competitive SOC marketplace.

About Methodics

Methodics, Inc. is a leading provider of design data management (DM) tools that improve the efficiency and collaboration of integrated circuit (IC) design. Its ProjectIC™ system is the first integrated platform for managing SoC realization, reducing the time, complexity and costs involved in IP-based design approaches. Its VersIC™ DM tools integrate industry-standard software configuration management (SCM) tools, such as the Perforce® and Subversion® version control systems into the hardware design environment to provide a more efficient global collaboration experience. The privately held company has offices in the USA, Europe and Asia. For further information, visit www.methodics.com.

About the Author

Vishal Moondhra is VP of Applications & Strategic Partnerships at Methodics Inc. and also sits on the Board of Directors there. Throughout his career, Vishal has led hardware design and technical teams specializing in data management, dev-ops, and hardware design. Today, Vishal helps lead the team dedicated to Methodics' ProjectC IP management platform while driving technical marketing and managing customer engagements with some of the world's largest technology companies

About Jama Software

Jama Software is the product development platform for companies building complex, smart and connected products. The Jama solution enables enterprises to accelerate development time, mitigate risk, slash complexity and verify regulatory compliance. More than 600 product-centric organizations, including NASA, Thales and Caterpillar, use Jama Software to modernize their process for bringing complex products to market. The company is headquartered in Portland, Oregon. For more information, visit www.jamasoftware.com.

