Integrate USD the nodal way, a visual VFX pipeline

Lucille Caillaud
Software Engineer
lucille@rodeofx.com

Robin De Lillo
Technical Lead
rdelillo@rodeofx.com

Guillaume Laforge
Head of R&D
glaforge@rodeofx.com

Jean-Christophe Morin
Software Engineer
jcmorin@rodeofx.com

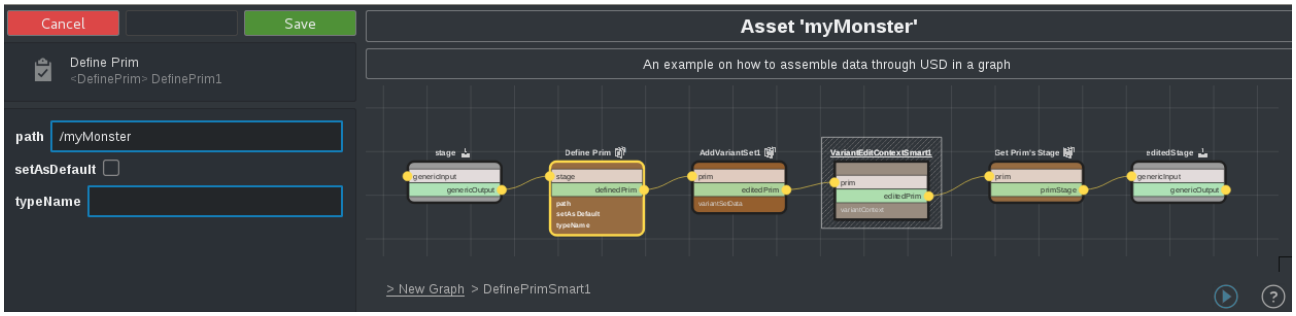

Figure 1: Construct USD data visually through a generic graph.

ACM Reference Format:
Lucille Caillaud, Robin De Lillo, Guillaume Laforge, and Jean-Christophe Morin. 2019. Integrate USD the nodal way, a visual VFX pipeline. In *Proceedings of SIGGRAPH '19 Talks.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3306307.3328157

1 INTRODUCTION

Pixar Universal Scene Description technology is well known in the VFX world. With its promising goal to unify and ease the interchanges of data between Digital Content Creation tools (DCC), it has convinced the industry since its initial open source release in 2016. The question is no more *Should USD be adopted ?* but *How to integrate it the good way in our pipeline ?*

Following the open-source release of Walter[1], the Rodeo FX suite of USD plugins for DCCs (announced at SIGGRAPH 2018[2]), this extended abstract aims to detail what have been the challenges of assemble USD workflows through a graph editor.

2 EXPECTATIONS AND CHALLENGES FROM A USD PIPELINE

From the initial brainstorm, here were our base principles:

- Bring the full power of USD to the artists without any restrictions or compromises
- Non-technical artists also needs to be able to control the exchanged data
- The pipeline is to be flexible and production driven at any time
- The technical support and involvement shall remain under control

And these are the challenges we foresaw:

- USD remains a programming API with required coding skills
- It takes time to be fully comfortable with USD technology
- USD often allows multiple solutions to a unique problem

3 USD BECOMES NODAL

3.1 Abstract Syntax Tree

We like to explain our USD graph as kind of Abstract Syntax Tree (AST) around the USD API. This decoupling can be compared to existing implementations:

- Javascript can interact directly with HTML with the DOM (Document Object Model).
- Control flow graphs in DCCs
- Composite programming patterns

Our graphs are organized with group container for scoped action, each leaf of the graph can be translated as a USD API call. The calls to the USD API are only made on graph execution. This allows any non-technical user to interact with the framework and remove the temptation hardcoded scripts.

3.2 Visualize and edit USD data seamlessly

The main benefit of such USD graphs is to provide a visual representation of USD calls. It abstracts any USD python script as a nodal graph, technical artists can then create, inspect or transform it in a secure and consistent way. This allows any prototyping/templating and also give full transparency to the DCC interchanges. The recipe or graph can also be exported separately for archive or delayed execution purposes. Thanks to this abstraction through graphs, software developers can provide production with default working

[1]Open source project: http://github.com/rodeofx/OpenWalter
[2]https://dl.acm.org/citation.cfm?id=3214772

SIGGRAPH '19 Talks, July 28 - August 01, 2019, Los Angeles, CA, USA

ACM ISBN 978-1-4503-6317-4/19/07.
https://doi.org/10.1145/3306307.3328157

templates of a vanilla pipeline then let the artists push it when required.

3.3 Decomposer

Another feature was implemented to simplify the transition toward USD, it is called the *decomposer*. It reverse-engineers any USD file to a nodal graph. This way, any kind of USD file, produced within or outside the studio, can be deconstructed as an editable node graph describing how it would have been created using the API. New node graphs, created very quickly, can then be used as a templates to generate new USD files. For example, we could replace the asset paths by new ones or add metadata on specific prims keeping the same USD structure. Resulting graph examples can also ease the learning of USD (as per the example on *Figure 2*).

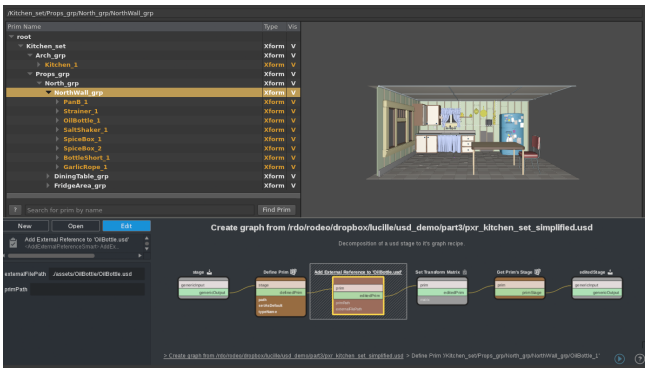

Figure 2: An USD file hierachy inspected through usdview tool along our graph editor.

4 UNLEASH USD POWER

4.1 Integration with our existing pipeline

Like many other VFX studios, Rodeo FX has developed its own nodal system[3] to run specific part of our pipelines. Currently, the ingest and delivery workflows as well as other render processes are already expressed as graphs. This means USD nodes can be fully combined with existing node libraries dedicated in rendering, encoding, database etc. to describe full end-to-end processes.

Existing nodes simply need to be tweaked to accept USD inputs when possible then whole processes can be switched to USD while keeping the full set of features already implemented. For a specific use case, a project with new requirements can easily deviate from the common workflow and investigate new USD usage. Later on, once experiments are proven working, the default process can be updated if needed.

4.2 Workflow example

Figure 3 is a demonstration of an end-to-end workflow going through multiple DCCs via USD interexchange format. From a specific shot, the goal is to render in Katana a turntable of a asset exported in differents parts by Maya.

[3] We also had interest in existing solutions such as Texels' *Kurtis* (commercial solution) and *Gaffer* from ImageEngine.

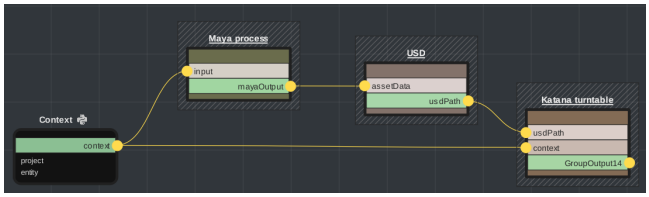

Figure 3: An end-to-end workflow built around USD.

Maya process group:

- Export the geometry, textures and UVs of an asset from a scene file
- Loop on different variant from the asset (*low*, *medium*, *high* quality) as separate files

USD group (see *Figure4*):

- Assemble all of the Maya outputs files in a common USD assembly
- Join it with some already existing shading data
- Output a single *self-contained* asset in a USD file

Katana group:

- Load the asset in Katana via Rodeo FX Walter
- Use a predefined template to create a turntable script
- Render the turntable image sequence as output

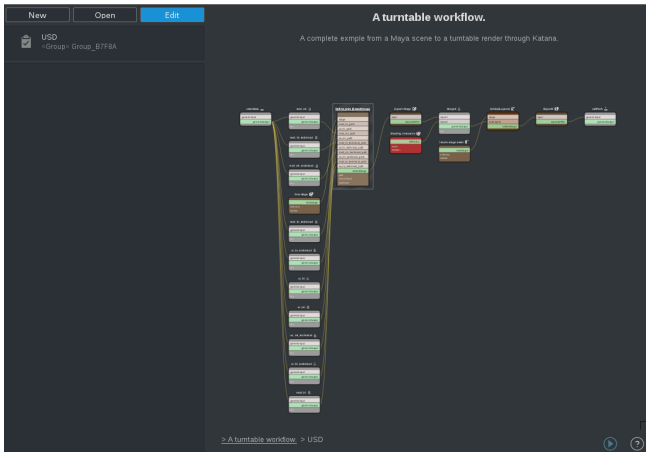

Figure 4: Within the USD group, the user can directly interact with USD based exchanged data.

5 CONCLUSION

As many other VFX studios, Rodeo FX is still in the process of integrate USD within its pipeline. We do hope that by exposing the core of USD data and features to any user interested in learning and editing the data via a nodal graph, this will ease the adoption of this upcoming standard. This flexible USD integration combined with Walter's tight integration with USD in DCCs constitute a solid foundation for the next generation pipeline at Rodeo FX.