Procedural Block-Based USD Workflows in Conduit

Chris Rydalch
Blue Sky Studios

Colvin Kenji Endo
Blue Sky Studios

Wayne Wu
Blue Sky Studios

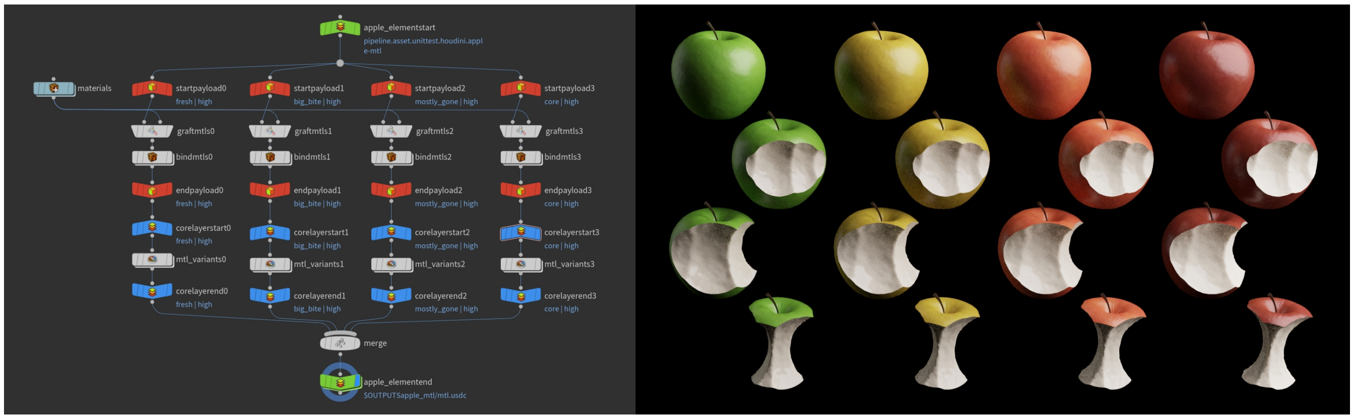

Figure 1: Artist adding materials to variants of an asset with block nodes.

ABSTRACT

We present a procedural block-based approach for USD pipelines that minimizes up-front USD knowledge requirements while ensuring users can still leverage the power of native USD. Building on USD and Conduit, we define fundamental workflow principles and philosophies on artist-interaction that guide our modular Houdini-based toolsets. Finally, we discuss the successes and challenges in scaling these workflows into production.

CCS CONCEPTS

• **General and reference** → **Design**; • **Software and its engineering** → *Software design engineering*; • **Computing methodologies** → **Animation**.

KEYWORDS

pipeline, production, USD, procedural, animation

ACM Reference Format:
Chris Rydalch, Colvin Kenji Endo, and Wayne Wu. 2021. Procedural Block-Based USD Workflows in Conduit. In *Proceedings of SIGGRAPH '21. ACM, New York, NY, USA, 2 pages.*. ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

In 2019, we introduced Conduit [1] and Universal Scene Description[1] (USD) as the foundation of a new modern pipeline at Blue Sky Studios. Since then, we have further integrated Conduit in all applications and deployed new production workflows. The introduction of Houdini Solaris[2], enabled us to explore and architect powerful procedural tools for artists to work with USD and Conduit.

While all production disciplines at Blue Sky deliver USD content, we explore the disciplines that natively author in USD, bringing forth our design philosophies and experience building procedural workflows in Solaris. As Solaris provides a native USD context, we aimed to expose as much flexibility of USD as possible to users. However, our early prototypes were confusing for artists, particularly in understanding their contributions to the USD and Conduit pipeline. We reformulated a new modular toolset that attempts to ensure USD features are easily accessible while requiring minimal USD knowledge to begin work. These workflows were tested and shown effective in an active studio production workflow, and the design principles can apply to other USD pipelines.

2 BACKGROUND: CONDUIT USD

Conduit provides repository containers, *Products*, that form the basis of the studio's USD pipeline implementation. Products can contain any type of pipeline data, typically USD. An *Entity* is a product composed of contributions from different disciplines. Assets, Scenes, and Shots are all entities in our pipeline, corresponding to *asset.usd*, *scene.usd*, and *shot.usd*. An *Element* is a product containing the individual discipline contributions, and elements compose to create an Entity. Elements usually contain the files that artists work with directly, such as *mtl.usd*, *anim.usd*, or *fx.usd*.

3 CORE PRINCIPLES

Our Solaris workflows are guided by four key design requirements for artist-interaction: Artists should be able to

(1) Work top-down similar to other Houdini contexts.

[1]https://www.openusd.org

[2]https://www.sidefx.com/products/houdini/solaris

(2) Understand their contributions to the pipeline; a composed working stage in LOPs should be consistent with the ultimate exported layer.
(3) View and flexibly switch their working context (e.g. prop, set, shot, or scene) based on discipline workflows.
(4) Build node graphs procedurally and modularly, and leverage expressions to make setups reusable and programmable.

To achieve these, we define the following three steps to characterize the central workings of our solution:

(1) *Scope* - Identify the artist's working root layer and area of interest, which may be nested within another scope.
(2) *Configure* - Prepare the target primitives for editing and configure them for scene hierarchy organization.
(3) *Compose* - Re-construct the scene to prepare for future edits using appropriate composition arcs, adhering to USD strength ordering rules.

4 IMPLEMENTATION

In Houdini's LOP context, we introduce a block workflow for artists to define the scope of their work. The block nodes come in pairs, start and end, to visually encapsulate an artist's work, inspired by code brackets and HTML tags. First, we scope each block pair to correspond to a USD layer. Blocks may contain multiple nested blocks to divide a layer's construction. Next, the start node configures the scope for editing, by adjusting the primitive hierarchy, de-instancing, or isolating primitives as needed. Finally, the end node composes and completes the artist's work on that scope, preparing USD layers and the stage for downstream operators and export.

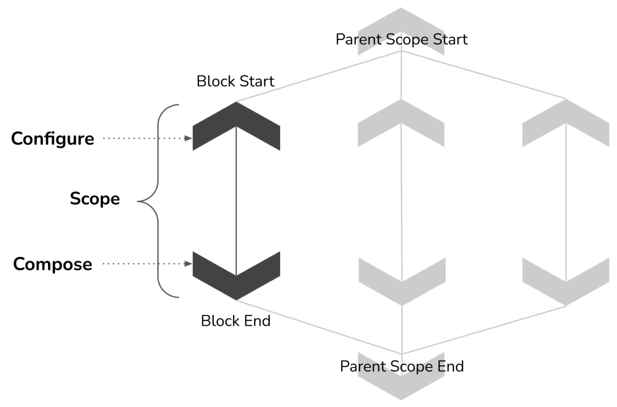

Figure 2: Breakdown of block nodes, with nested pairs.

This core block toolset forms the basis of nearly all Solaris-based workflows. Many cross-departmental tools are implemented following the block structure paradigm, regardless of asset, scene, or shot work. For example, all Houdini workflows that interface with Conduit employ the *Element Block*, the primary block for editing and contributing USD pipeline data. Element Block is designed to allow artists to work in context of an Entity, with consistent composition in both the artist's working stage and on export. For asset work, we introduce additional blocks that are tailored for our asset structure. These blocks are nested within the Element Block and scope an artist's work either in or out of an asset variant, and in or out of the USD payload.

5 RESULTS

Our block workflow proved successful and intuitive for artists to use. Explicit scoping and construction made USD exports significantly more predictable and reliable than a previous version of the toolset. Artists could preview their contributions in context, before export, composed with USD data from other disciplines within the same entity, or preview their contribution (e.g. asset) to separate entities (e.g. a shot containing the asset) by broadcasting the changes.

Particularly, with many new to USD, the combination of the block workflows and Solaris helped both artists and TDs to understand USD, asset structure, and Conduit concepts more visually. Artists were able to step through and visually understand how their contributions would compose with and fit into an existing asset, scene, or shot. The block workflows ensured that advanced USD knowledge was not a requirement to contribute to the pipeline.

Workflows proved intuitive for artists with a baseline understanding of USD, and experienced users were able to leverage USD fully to deviate and build upon typical workflows for advanced use cases. For example, we introduced tools to broadcast asset edits across a stage, and a node to cache out heavy USD data and perform stitching operations. All workflows were fully procedural and purposely compatible with Houdini's PDG[3] architecture, which many users utilized.

A drawback to our implementation was the overhead that each end block imposed. These nodes composed the artist's work but did so into a new stage. In addition to the costs behind opening a new stage, Hydra would clear itself out and re-sync the primitives from the new stage. These composition factors contributed to poor interactivity in heavy scenes.

6 FUTURE WORK

Minimizing stage re-composition throughout the block tools, potentially at the cost of their explicitness, would improve interactivity for artists using the scene viewer. Introducing scope-based population and load masks to isolate sections of complex scenes per block would improve artist efficiency. Integrating procedural tools within the block nodes, such as PDG, would accelerate parallel multi-asset and multi-shot workflows.

ACKNOWLEDGMENTS

Our work would not have been possible without the fantastic artists, TDs, and engineers at Blue Sky Studios. We would also like to thank SideFX, particularly Mark Tucker and his team, for providing invaluable development insights.

REFERENCES

[1] Oliver Staeubli, Tim Hoff, Ryan Bland, Rebecca Hallac, Josh Smeltzer, Chris Rydalch, Karyn Buczek Monschein, and Mark McGuire. 2019. Conduit: A Modern Pipeline for the Open Source World. In *ACM SIGGRAPH 2019 Talks* (Los Angeles, California) *(SIGGRAPH '19)*. Association for Computing Machinery, New York, NY, USA, Article 47, 2 pages. https://doi.org/10.1145/3306307.3328175

[3]https://www.sidefx.com/products/pdg/