CORE ENGINE | GENERIC DATA PIPELINE ENGINE & FRAMEWORK

1.0 Introduction

The Core Engine aims to re-envision how data Pipeline operations are executed for an M&E production environment. An evaluation of our production challenges resulted in a collection of requirements that drove our design choices towards discarding industry legacy in favour of modern technology solutions.

2.0 Requirements

Requirements for the Core Engine are defined by a series of clear objectives. These aim to give the studio efficiency, stability and an expansion of operational capabilities.

2.1 Objectives

- **Scalability** for productions to shrink/grow
- **Stability** which guarantees 99.9% uptime
- **Multisite** operations across studio locations
- **Data Integrity** to guarantee reproducibility
- **Data Versioning** for incremental updates
- **Data Relationships** to track dependencies
- **Data Management** via context and needs
- **Event Automation** for workflow efficiency
- **Infrastructure Independent** operations
- **Developer Practices** to improve releases
- **Remote Workers** to support any scenario
- **Cloud Support** for using IaaS when needed
- **Security Compliance** for production safety
- **Studio Analytics** providing BI insights
- **Supportability** using in-house teams
- **Marketability** for potential revenue stream

2.2 Technology Stack

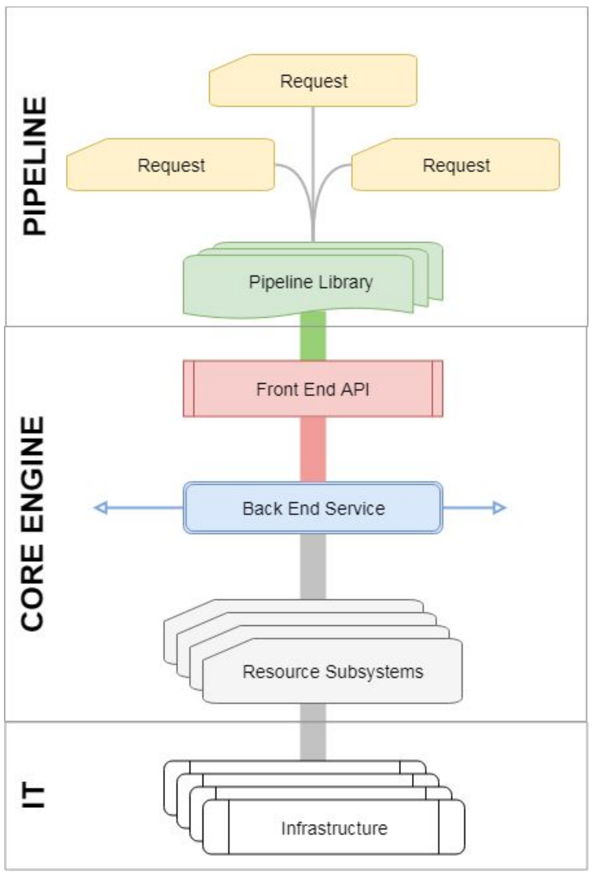
We selected the best tools to accomplish our goals. Our tech stack below are proven in other sectors and solve specific challenges.

Golang, Python, GraphQL, Elasticsearch, Bazel, NATS, Docker and Kubernetes

This selection allowed us to design software which would be performant, maintainable and scalable across all physical locations. Managing services, providing fast results to queries for information, ensuring consistent and tested releases, and virtualizing environments were all viable solutions because of the technology selections made.

3.0 Design Overview

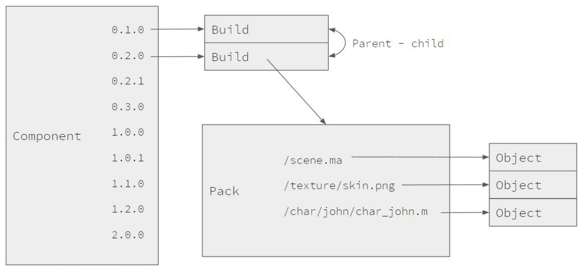
The main philosophy behind the Core Engine is that Pipelines should not be responsible for interacting with infrastructure directly; they provide tooling which supports workflows. Core Engine provides Pipelines standardized methods for storing and retrieving information through a front-end API Resource schema. These requests qualified, resolved across the whole cluster and translated to get data from the available infrastructure.



This diagram shows a simplified view of the layers of software a request traverses and the areas of responsibility for each. Make note of the arrows on the back-end, indicating other clusters and/or physical locations.

3.1 Data Types

Generalized data types are defined by the Core Engine to allow building Pipeline workflows using common logical concepts rather than having workflow code access data directly from infrastructure; an abstraction layer.



3.1.1 Resource Data Types

Component
Single unit of production data (“asset”) that is responsible for change tracking via versions.

Build
Single realization of a Component (“version”) that tracks dependencies and environments.

Pack
Collection of Objects that is responsible for describing shape of data when unpacked.

Object
Arbitrary blob of data identified by the hash of its contents; a store of information.

Task
Single unit of work to complete - in a tree structure - producing one Component.

Actor
Human or machine that can perform Actions, responsible for username and authentication.

Group
A set of Actors allowing for Permission control.

Permission
An endorsement to an Actor or Group, allows an Action or set of Actions to be perform.

3.1.2 Non-Resource Data Types

Tag
Text string assigned to any Resource, giving Pipelines the ability to add/remove context for search and organizational purposes.

Action
An execution within the Actor system that are used to enforce Permissions, track history for security audits and allow for automation.

Workspace
Localized environment where Packs can be opened, updated and constructed. Contains the working files and dependent Components.

4.0 Features

The implementation of Core Engine offers the business a number of features and capabilities to meet the requirements previously outlined.

Per-Production Instancing
Each production will have a separate instance of the deployed software running.

- Granular control of software releases to allow for scheduled updates per show.
- Data security by guaranteeing Actions cannot execute across instances.
- Reducing single points of failure.

Multi-site Deployments
Software architecture allows for clustered deployments across physical locations.

- All location have standardized workflows and access to multi-site data & operations.
- Redundancy across entire cluster.
- Authenticated services for secure multisite.
- Elastic infrastructure procurement.

Workspaces
The mechanics behind the Workspace were designed to offer a “work in progress” solution.

- Software, pipeline tools, configuration and environments are tracked with work in progress to ensure reproducibility.

- Localization of work data on user's machine as part of a "check out/in" system. Fast data interactions and guarantees ability to work if network interruptions occur until publish.
- Reducing load on network infrastructure.
- Checkpoints in work data capture a work in progress version of all files in progress.

Components
The mechanics behind the Component were designed to offer an "Asset" solution.

- Versioning sets of data with any number of concrete files needed (ex: image sequence)
- Immutable data guaranteeing no data can be modified or destroyed in a Workspace
- Dependency tracking of specific versions of other Components, to ensure all required data can be unpacked in a Workspace

Hierarchical Tasks
Task trees can be built to define the expected output Components for any work done.

- Standardizes workflows within a Pipeline and guarantees expected outputs are produced
- Customizable workflows, allowing unique solutions for hero shots and other scenarios requiring deviation from standards

Infrastructure Abstraction
Separating Pipeline operations from the site specific infrastructure yields tangible benefits.

- Avoids vendor lock-in which allows for flexibility in migrating to other solutions
- Expansion for burst capacity into cloud resources simply requires new integrations to be built, rather than workflow changes
- Each physical location can have different infrastructure without impacting Pipelines

Action Logging & Authentication
Every Action performed on each instance is both authenticated and logged indefinitely.

- Hooks for trigger automation scripts.
- Records for use in regular security audits.

5.0 Looking Forward

At the time of writing this document, the Core Engine is on target to have a Pipeline ready release in January 2019. The scope includes all essential features for building a Pipeline and managing multi-site data. Active development of a proof of concept Pipeline using the Core Engine is in progress as an initial step towards developing a refreshed solution for the studio.

Beyond the initial set of required features, we'll aim to expand the Core Engine into the future with features that could include:

- Building integrations for cloud providers to allow for remote workstations, burst capacity rendering and backup or data archival
- Data warehousing and analytics integrations for supporting business intelligence reports with a production and operations focus. Potentially even expanding this into forecasting studio performances and predicting needs, such as render capacity.
- Putting together a "studio in a box" solution for working with outsource partners. Having a shippable server which is pre-configured with direct access to a production's cluster could allow for faster delivery of outsource content and possibly even native access to our Pipeline if that's what is necessary.
- Productizing the solution as a general purpose Pipeline Engine and Framework. Design considerations were made early on to ensure the possibility of licensing our solution would be viable with minimal amounts of re-development.

6.0 Contact Information

If you would like to learn more about the project, or engage our studio in any way, please direct your inquiries to:

Grant Moore
gmoore@bardel.ca
Pipeline Supervisor, Core Engine