Reach Next Level!

CEDEC 2015

FINAL FANTASY XV -EPISODE DUSCAE-の エフェクトはこうして作られた

～Luminous VFX Editorの紹介～

株式会社スクウェア・エニックス
長谷川 勇　野副 竜太　小野 哲平

本セッションのゴール

- 伝えたいこと
  - 次世代機でのエフェクトデザインの一例
  - ノードベースエフェクトツールのメリット・効果的な運用
  - 実現時の課題とその解決方法

- 対象者
  - エフェクト開発に関わる人
    （現在関わっていないが興味がある人を含む）

アジェンダ

- 次世代機でのエフェクトデザインの一例 ← アーティスト向け
  - FFXV-EPISODE DUSCAE-におけるエフェクトデザイン
  - Luminous VFX Editorの紹介
- ノードベースエフェクトツールのメリット・効果的運用
  - ノードベースのメリット
  - Luminous VFX Editorのツール設計 ← アーティスト・TA・プログラマ向け
- 実現時の課題とその解決方法
  - 実装時の技術的課題
  - Luminous VFX Editorにおける解決方法 ← プログラマ向け

アジェンダ

- 次世代機でのエフェクトデザインの一例 ← アーティスト向け
  - FFXV-EPISODE DUSCAE-におけるエフェクトデザイン
  - Luminous VFX Editorの紹介
- ノードベースエフェクトツールのメリット・効果的運用
  - ノードベースのメリット
  - Luminous VFX Editorのツール設計
- 実現時の課題とその解決方法
  - 実装時の技術的課題
  - Luminous VFX Editorにおける解決方法

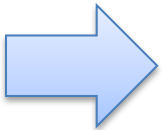
FINAL FANTASY XV-EPISODE DUSCAE-
におけるエフェクトデザイン

ゲームエフェクトとは

- ゲームエフェクトは
  ゲーム世界の中に存在するもの

ゲームエフェクトとは

- ゲームエフェクトは
  ゲーム世界の中に存在するもの



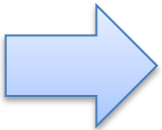
ゲームの世界観はエフェクトの
デザイン上で重要な情報

FINAL FANTASY XVの世界観

- フォトリアルに描くファンタジー世界

世界観を踏まえた上で…

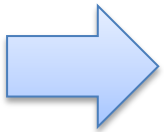
- FINAL FANTASY XVで目指すべき
  エフェクトデザインは



フォトリアルに描く
ファンタジーエフェクト

フォトリアルに描くファンタジーとは

- FINAL FANTASY XVエフェクトの解釈
  - ファンタジックな表現においても
    自然現象や物理現象をモデルとして描く



自然な表現を目指すことで
フォトリアリスティックとする

フォトリアルの問題点

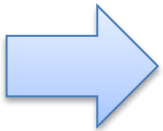
- フォトリアルだけだと不足する演出
  - 過剰に演出したい場面
  - 記号的にエフェクトを表示したい場面

適度なフェイクを取り入れる

FINAL FANTASY XVエフェクトの作例

フォトリアルに対する新たなアプローチ

- さらなるフォトリアルを
  目指すために実践したこと



環境の影響により
動的に変化するエフェクト

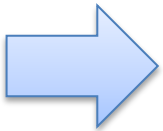
ゲーム内で変化する環境

- FINAL FANTASY XVにおける環境変化
  - 時間変化
    - 朝夕の切り替わり
  - 天候変化
    - 風の変化
    - 濃霧

環境変化を受けるエフェクトの作例

まとめ

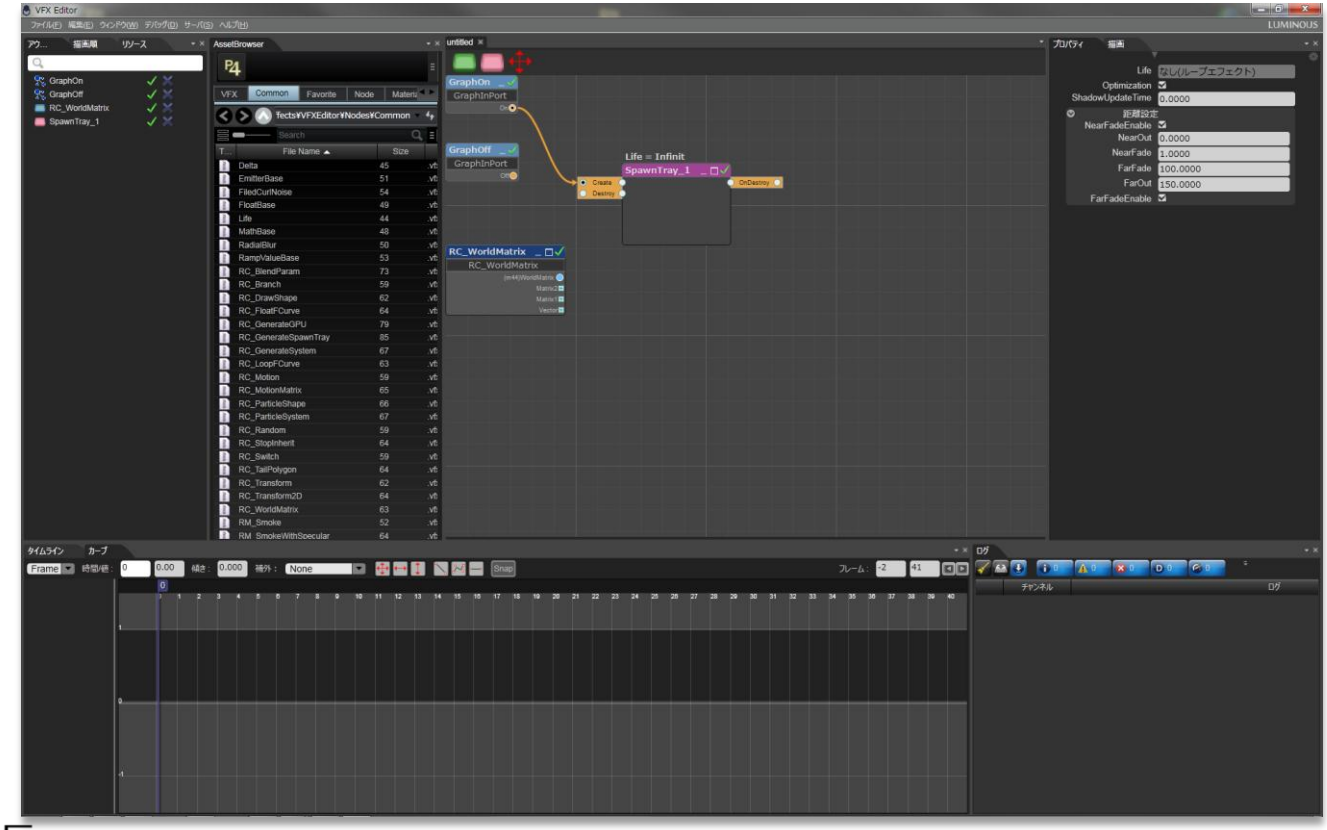
- FINAL FANTASY XVでは
  - ファンタジー表現を自然、物理現象の延長として捉える
  - フォトリアルが破綻しない程度のフェイク
  - 環境の影響で変化するエフェクト表現



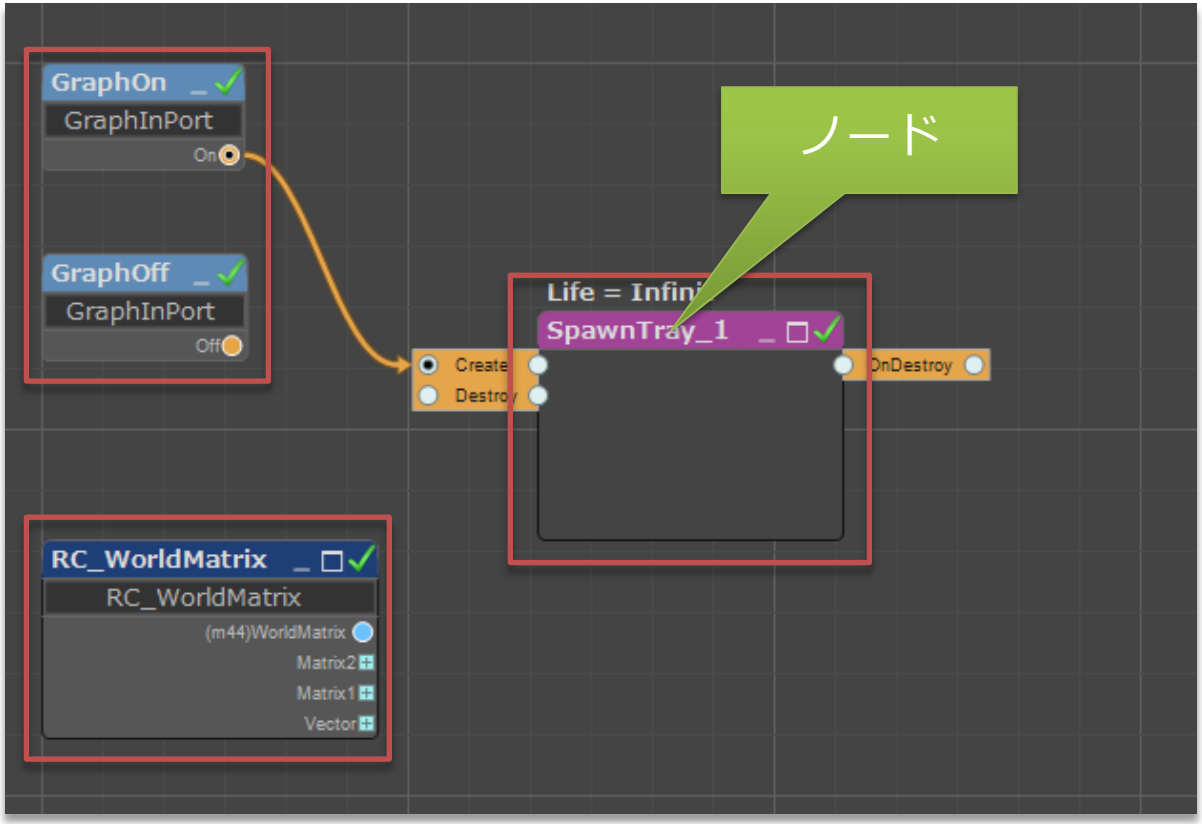
フォトリアルに描く
ファンタジーエフェクト

エフェクト開発環境
LUMINOUS VFX EDITORの紹介

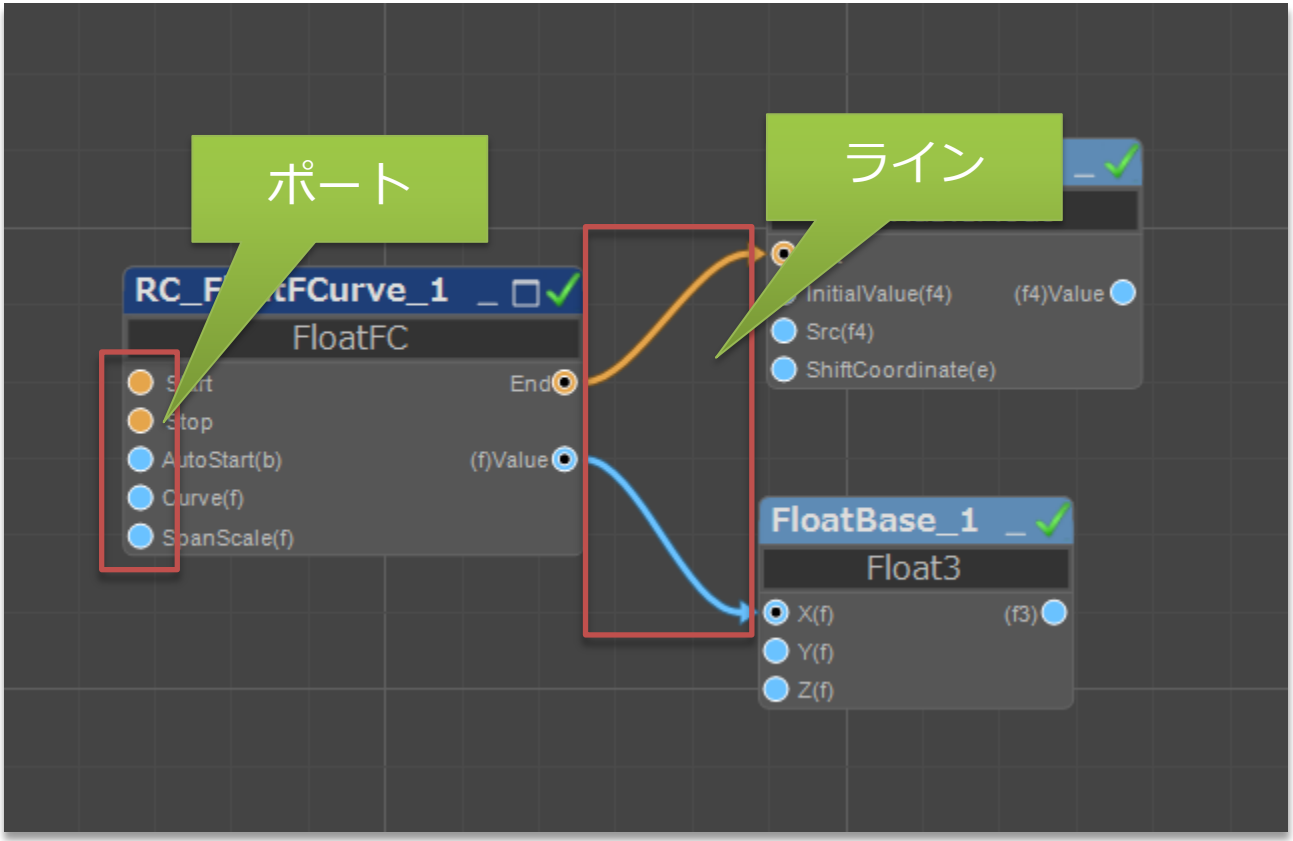
Luminous VFX Editorの紹介



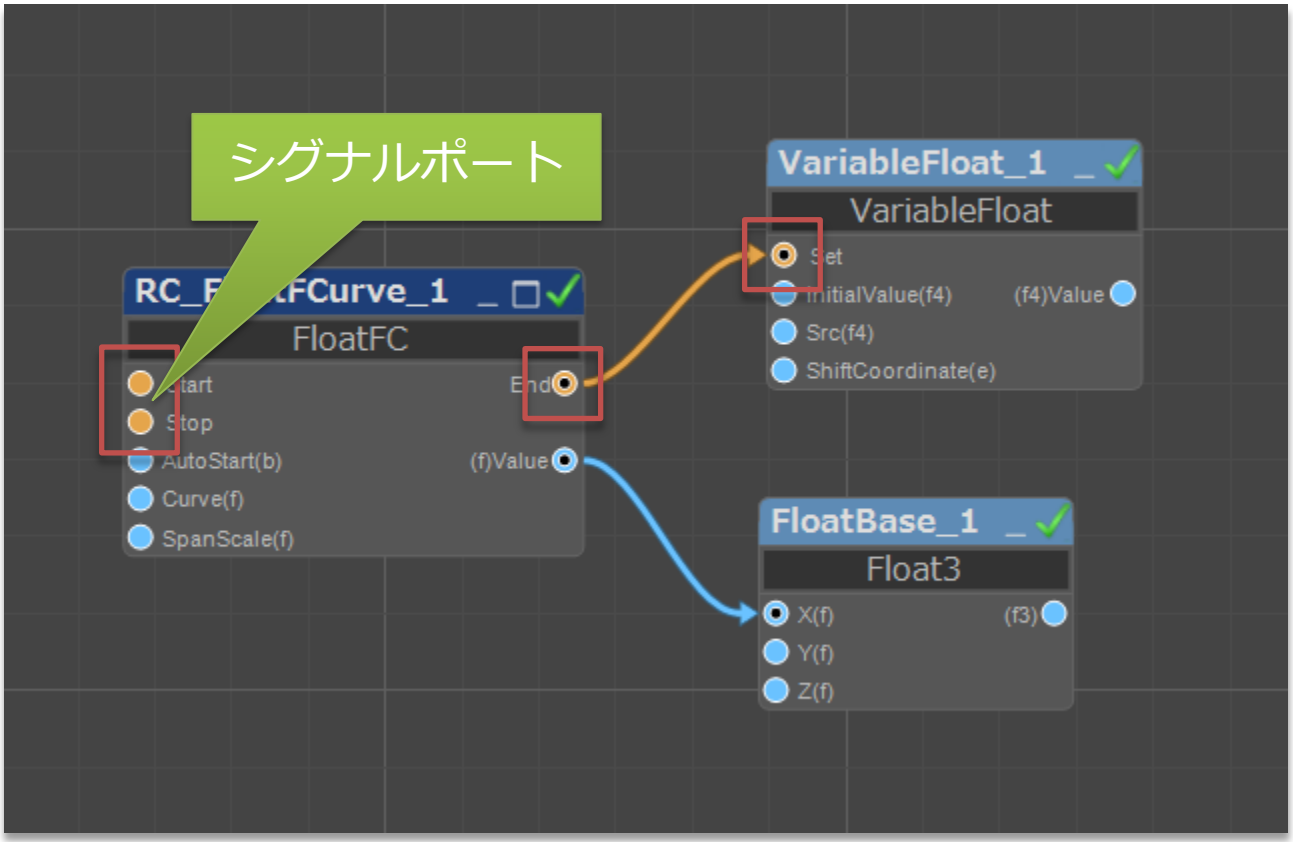
Luminous VFX Editorの紹介



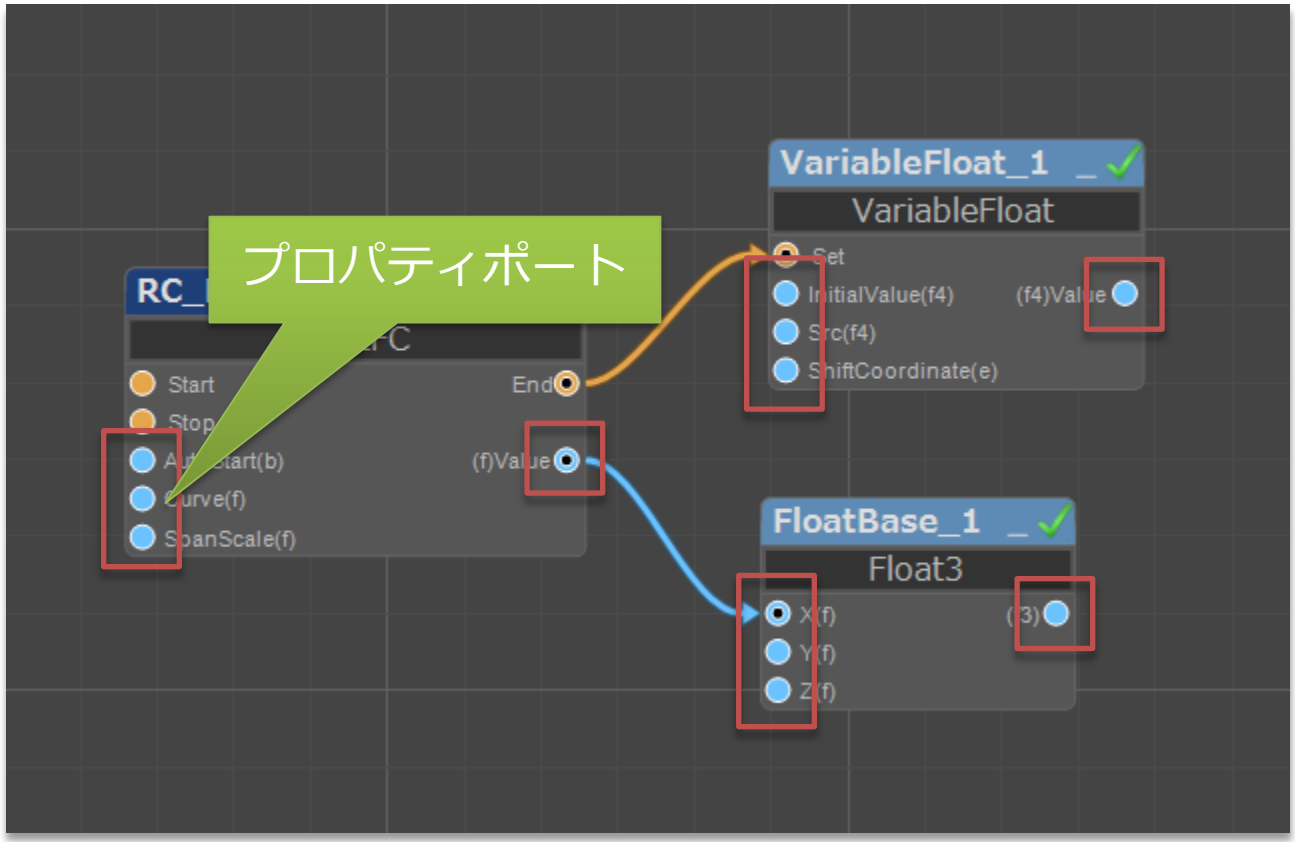
Luminous VFX Editorの紹介



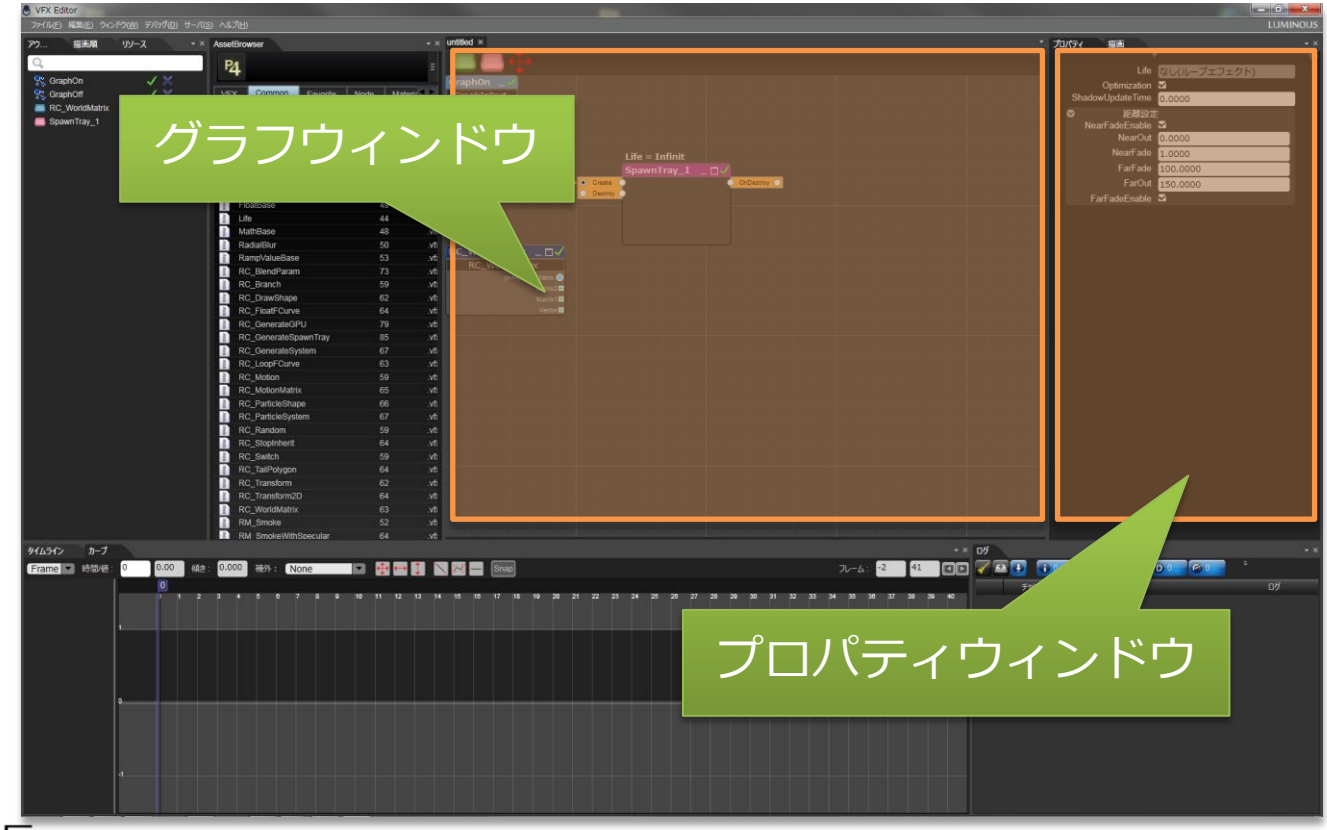
Luminous VFX Editorの紹介



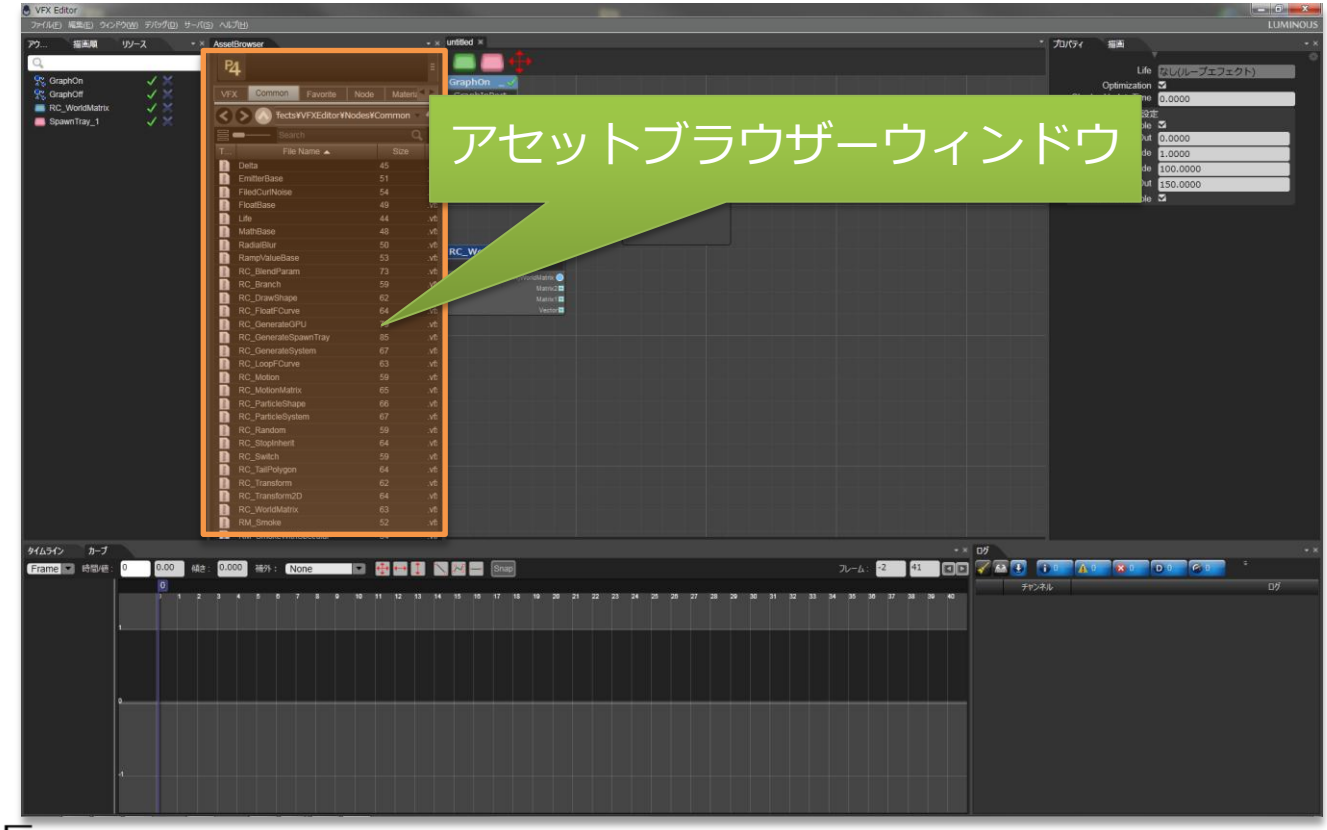
Luminous VFX Editorの紹介



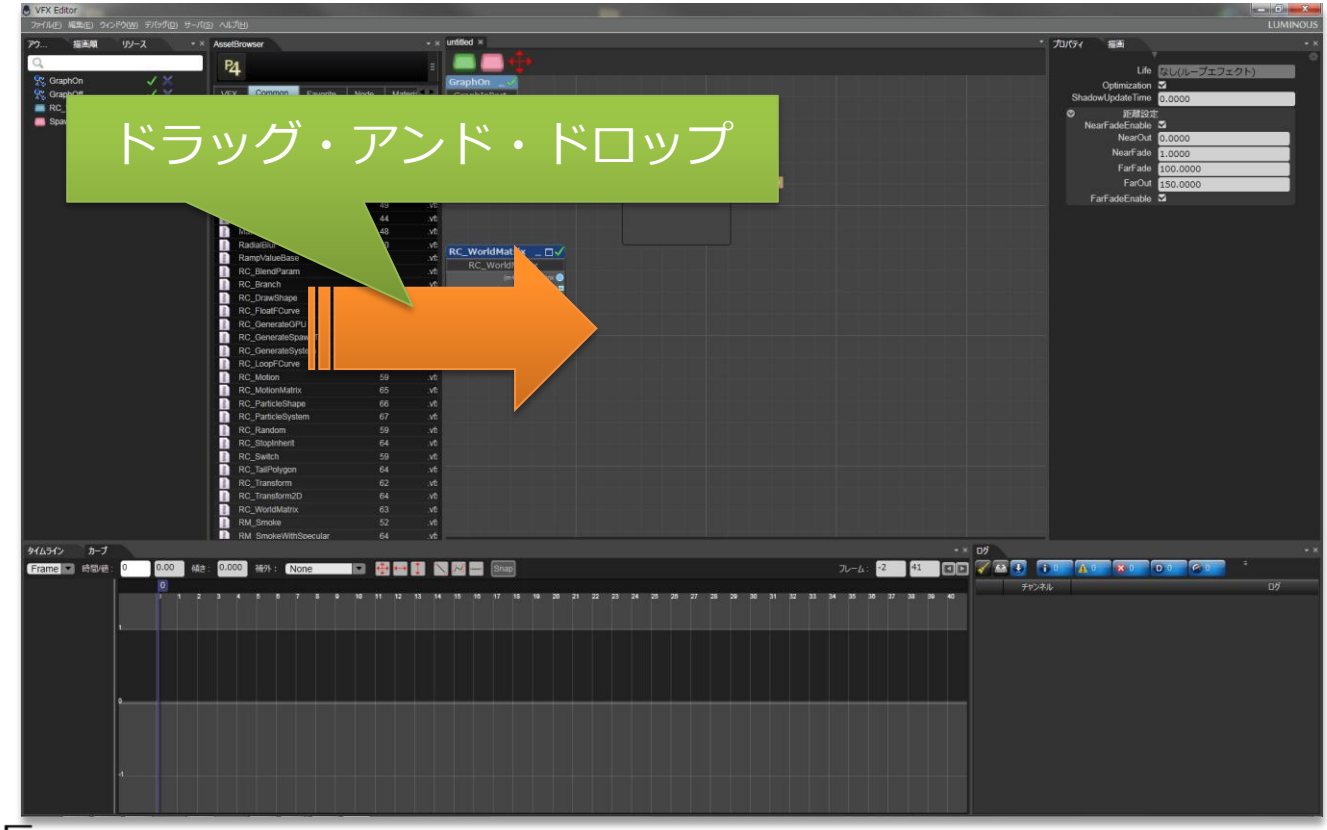
Luminous VFX Editorの紹介



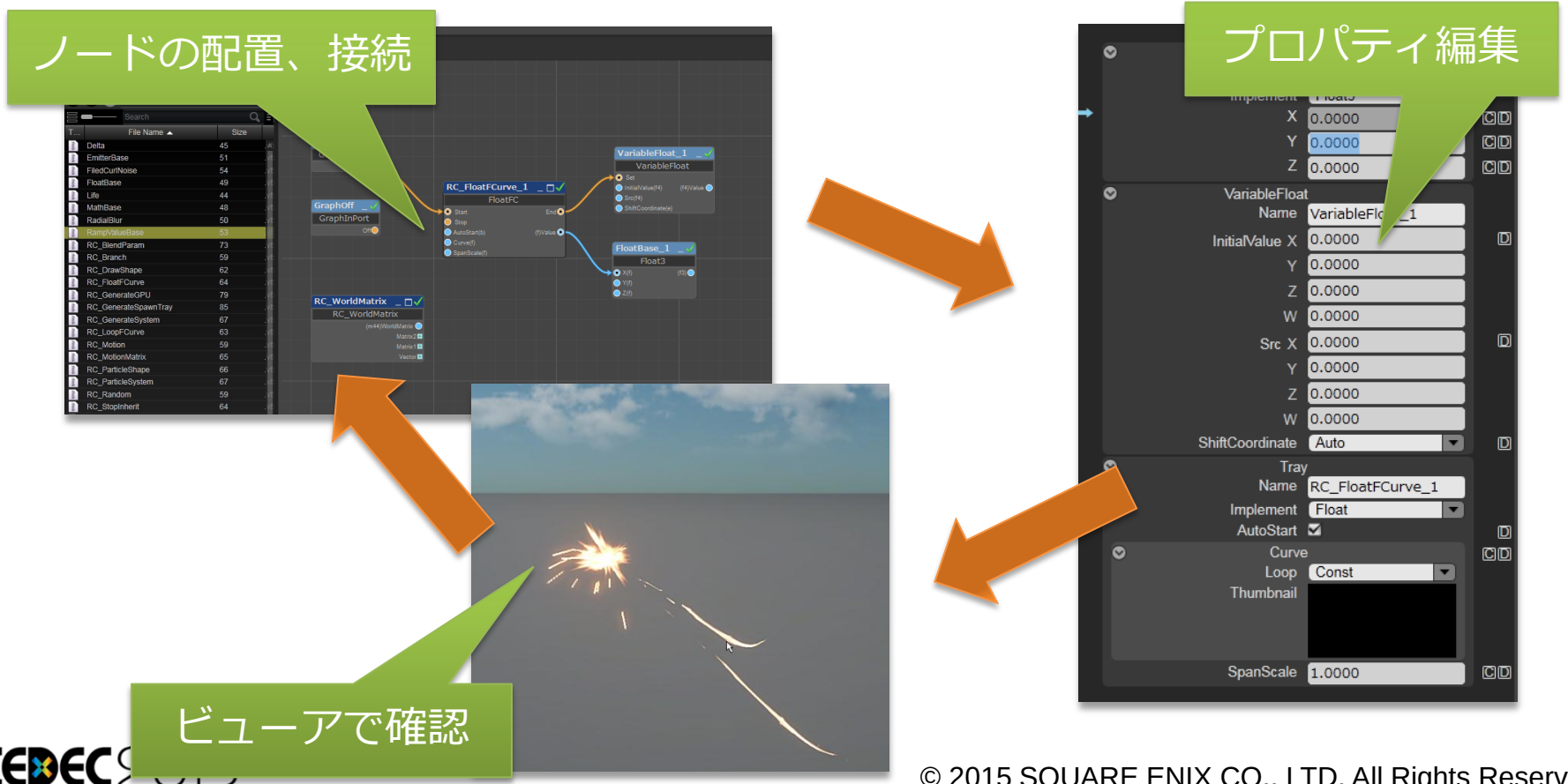
Luminous VFX Editorの紹介



Luminous VFX Editorの紹介

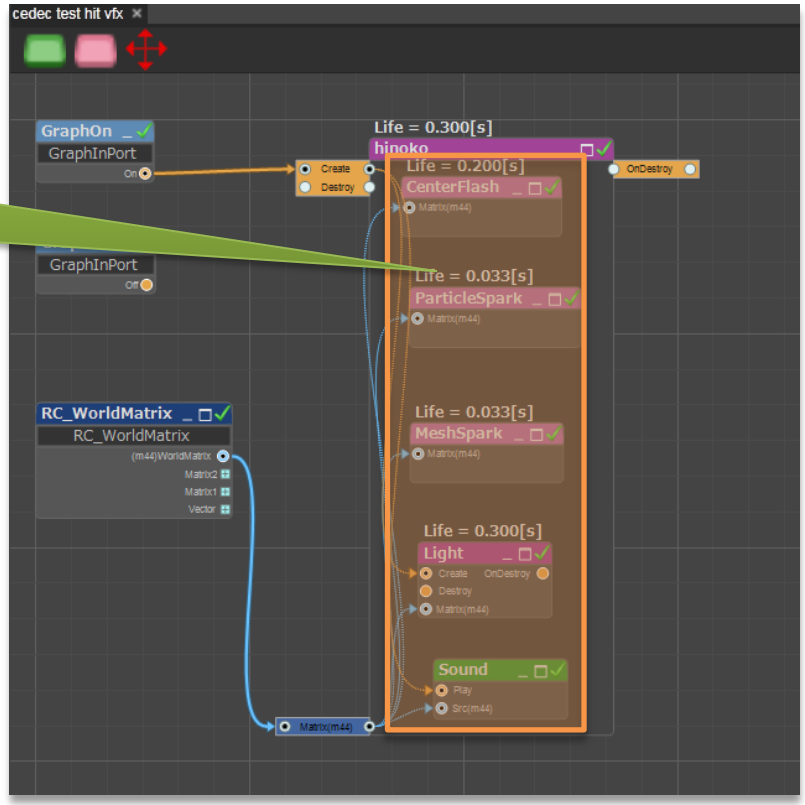


Luminous VFX Editorの紹介



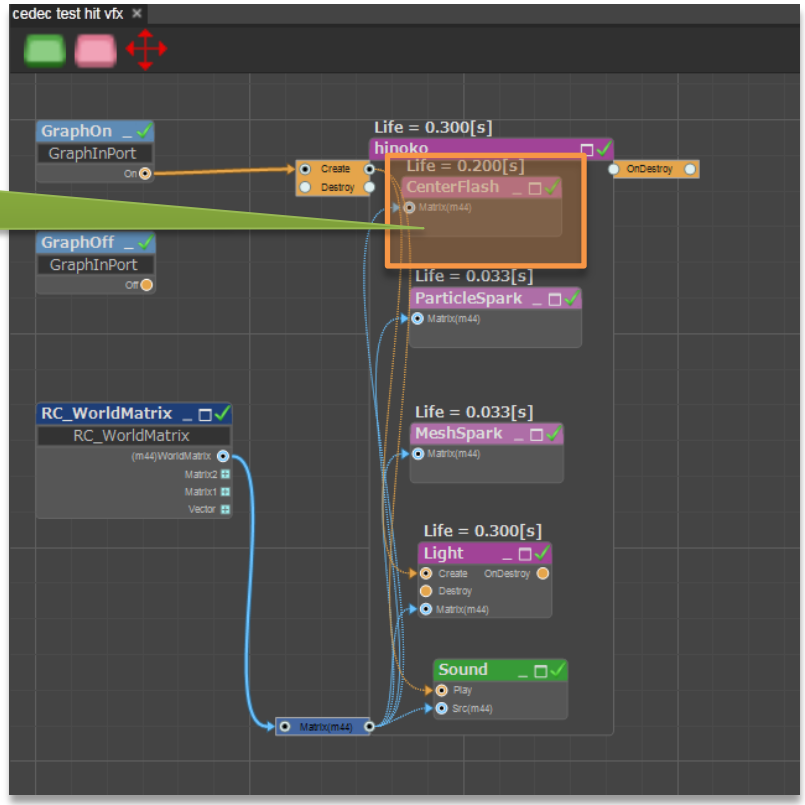
FFXVのエフェクトデータ紹介

エフェクトを構成するパーツ



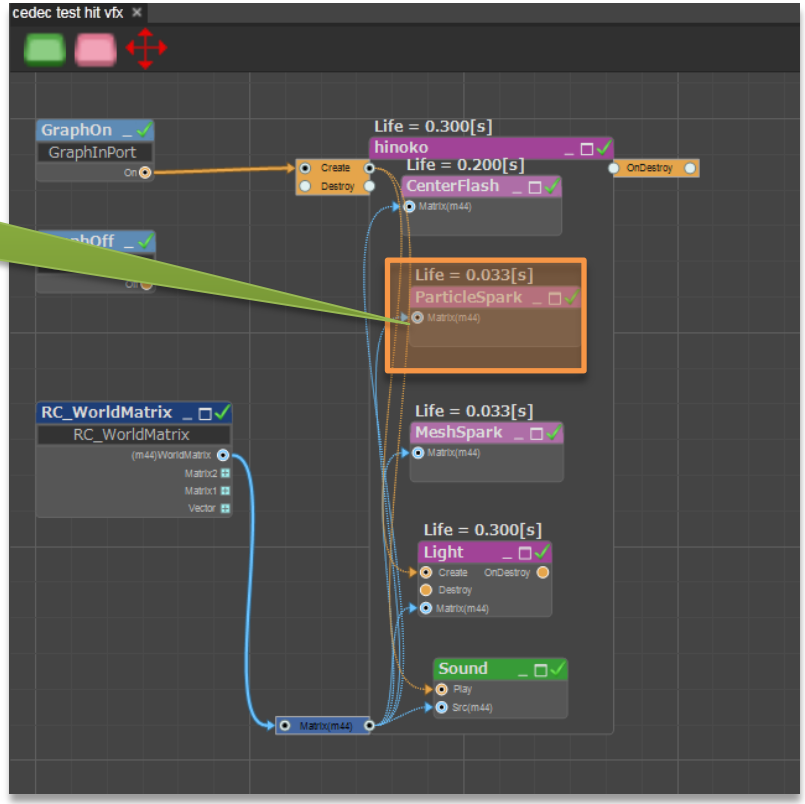
FFXVのエフェクトデータ紹介

中心発光パーツ



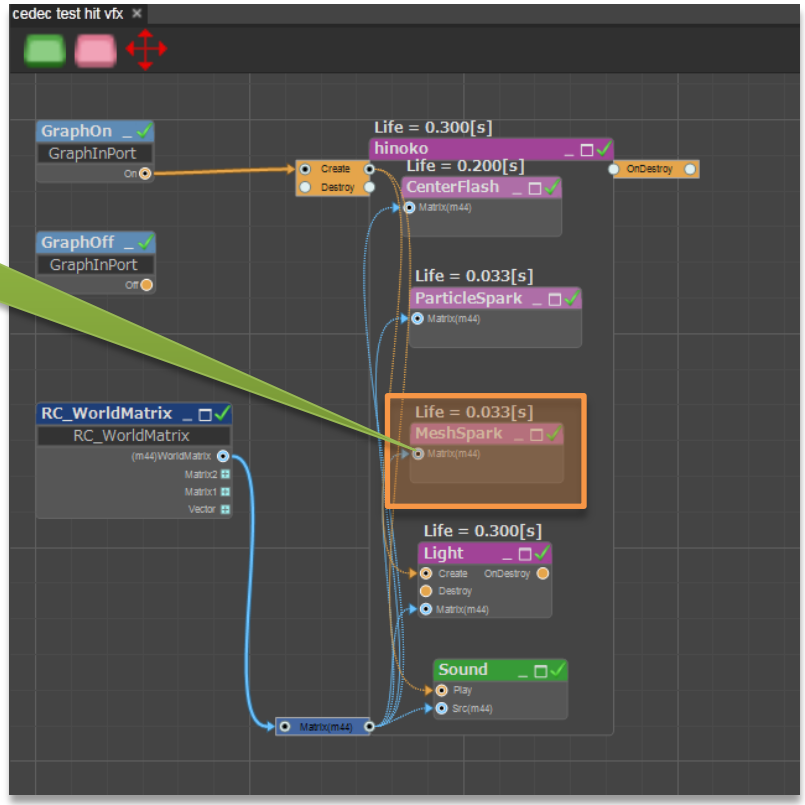
FFXVのエフェクトデータ紹介

Particleの火花



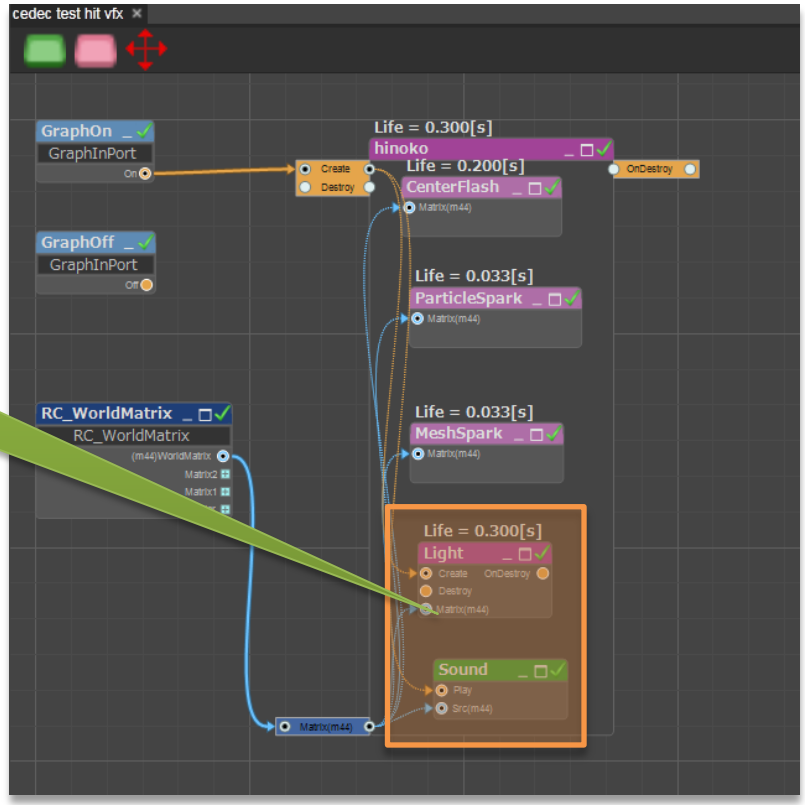
FFXVのエフェクトデータ紹介

火花のライン

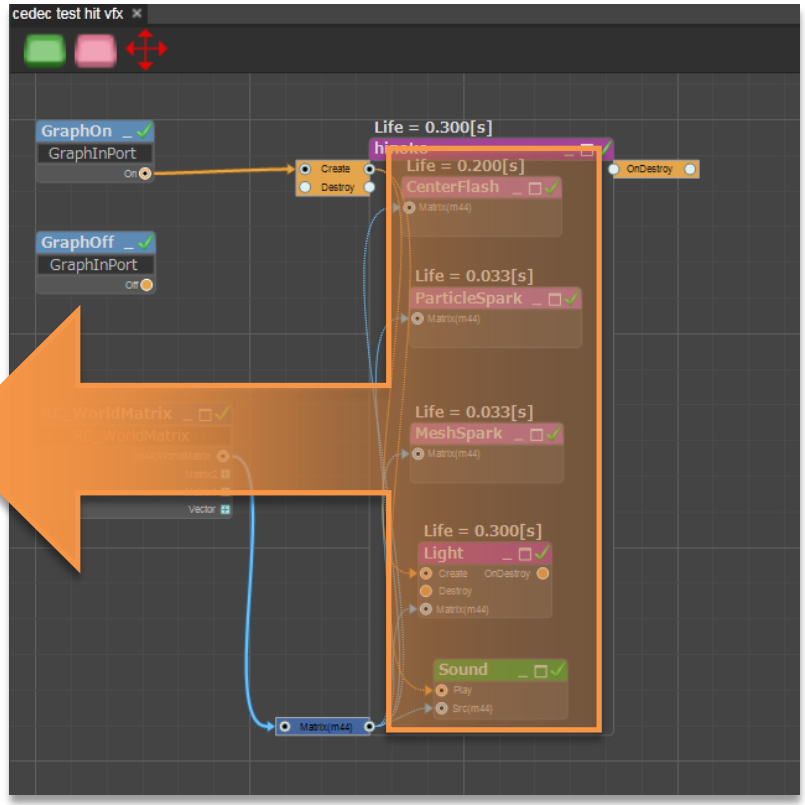


FFXVのエフェクトデータ紹介

ライト、サウンド

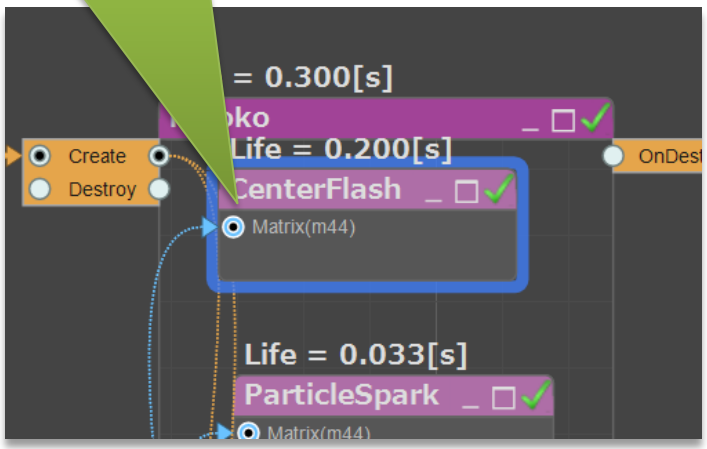


FFXVのエフェクトデータ紹介



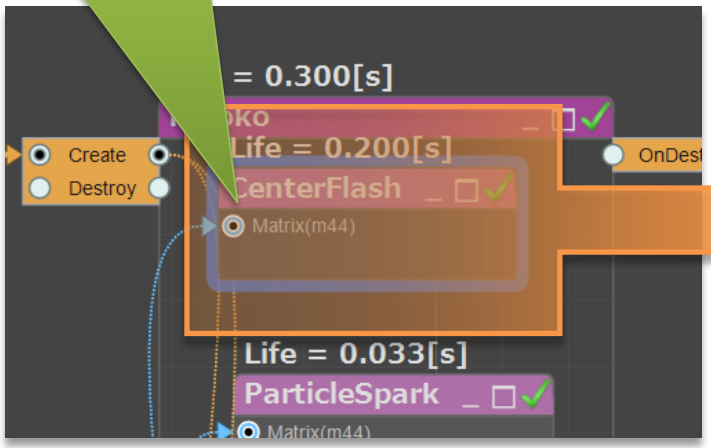
FFXVのエフェクトデータ紹介

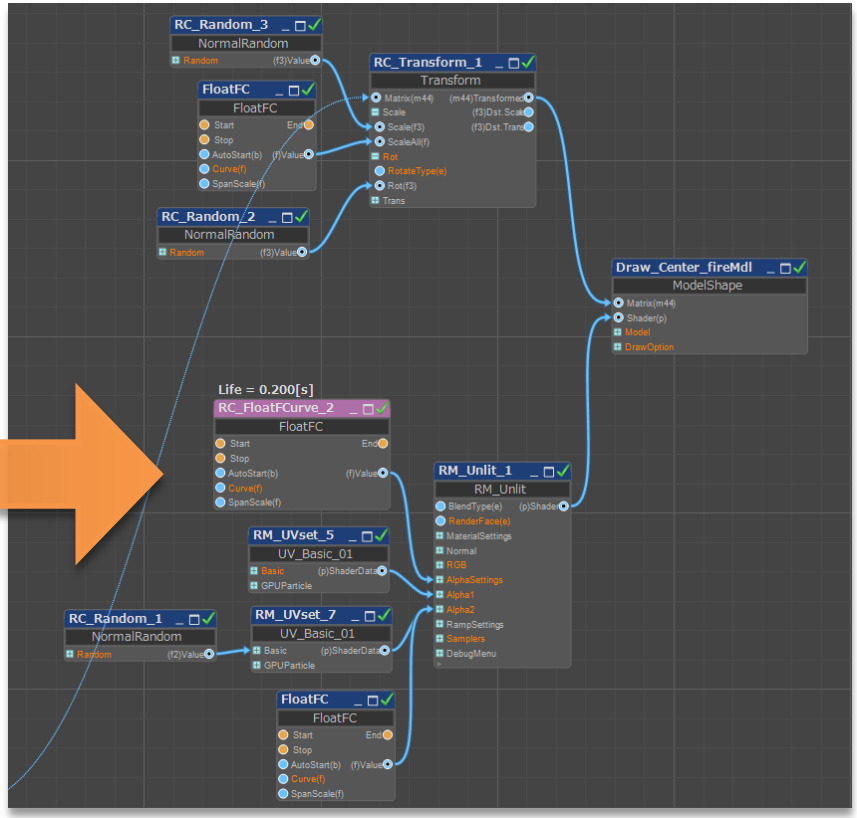
中心発光パーツ



FFXVのエフェクトデータ紹介

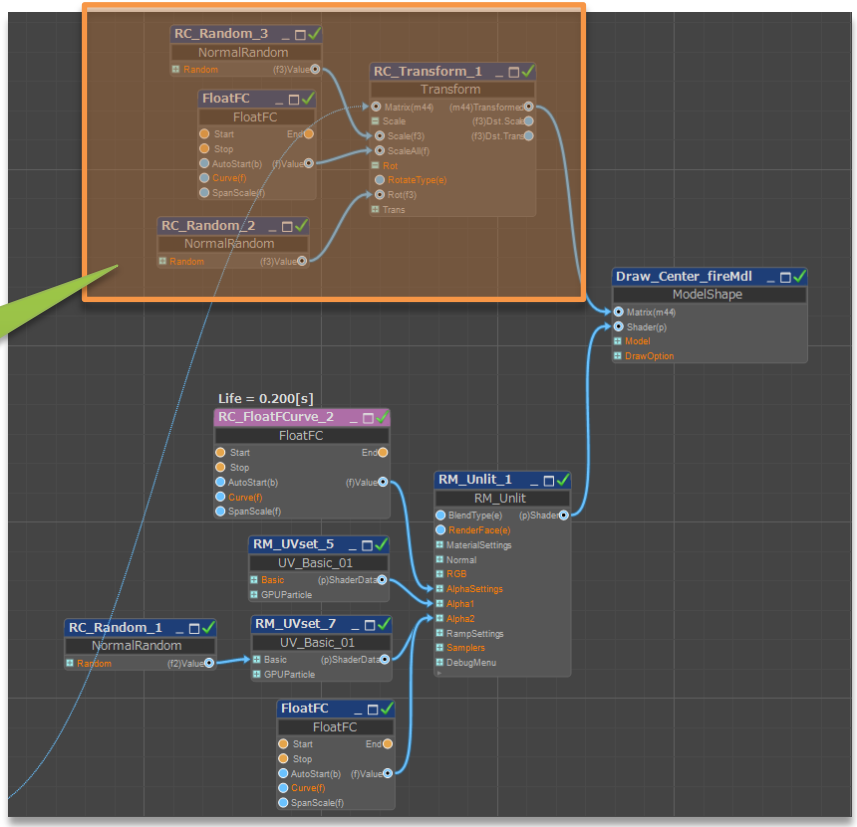
フォルダオープン





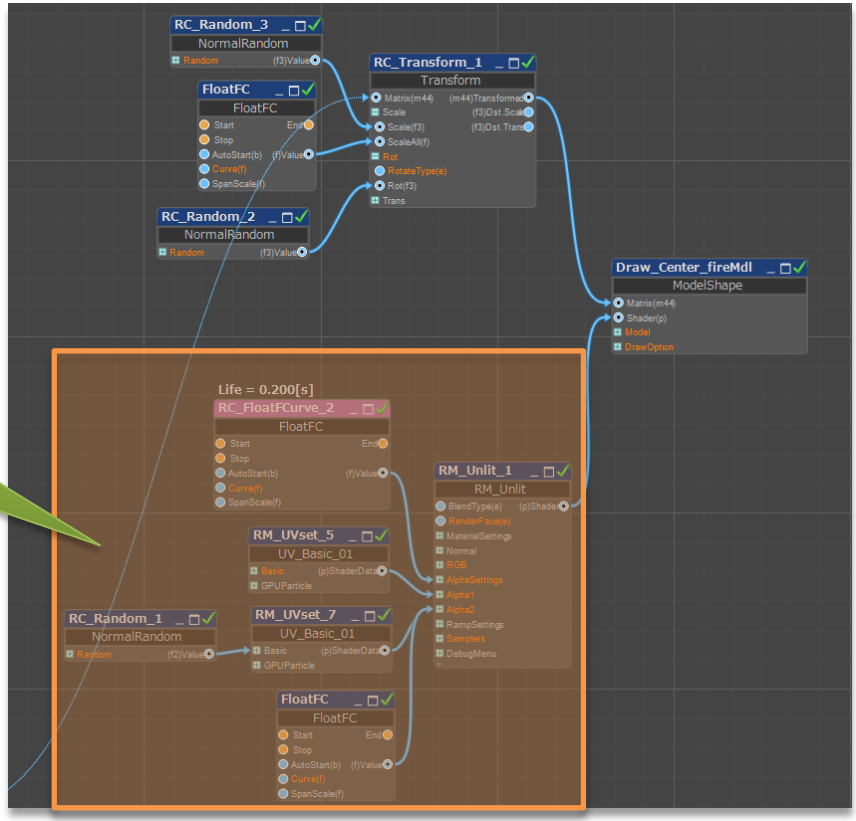
FFXVのエフェクトデータ紹介

Matrix計算
Scale、 Rotate、 Transの設定



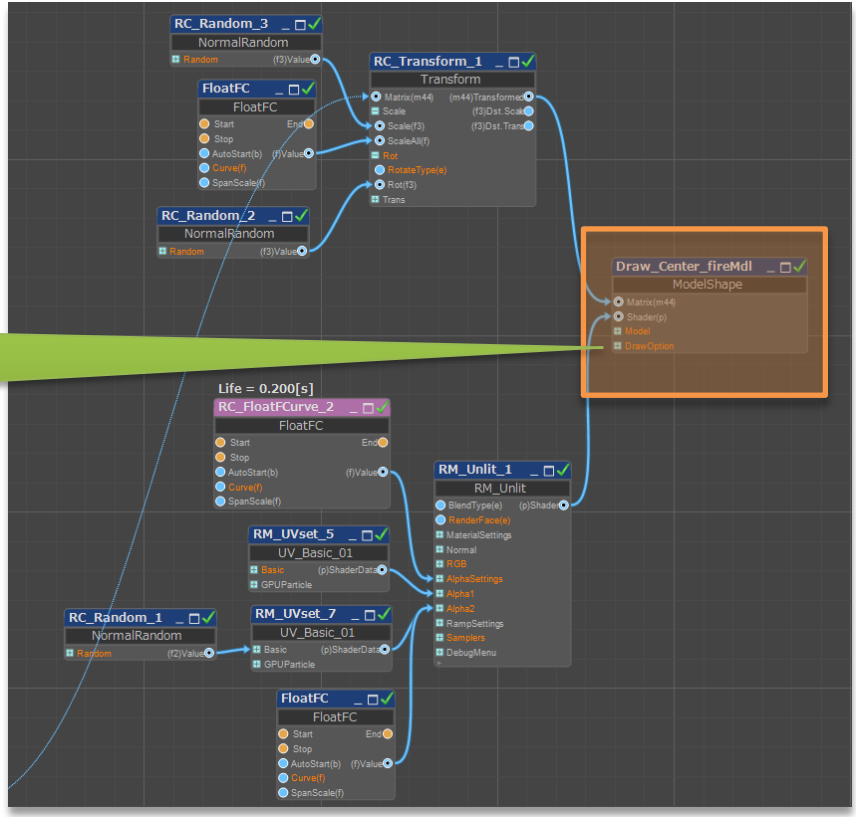
FFXVのエフェクトデータ紹介

Material計算
ColorやTextureなどの設定

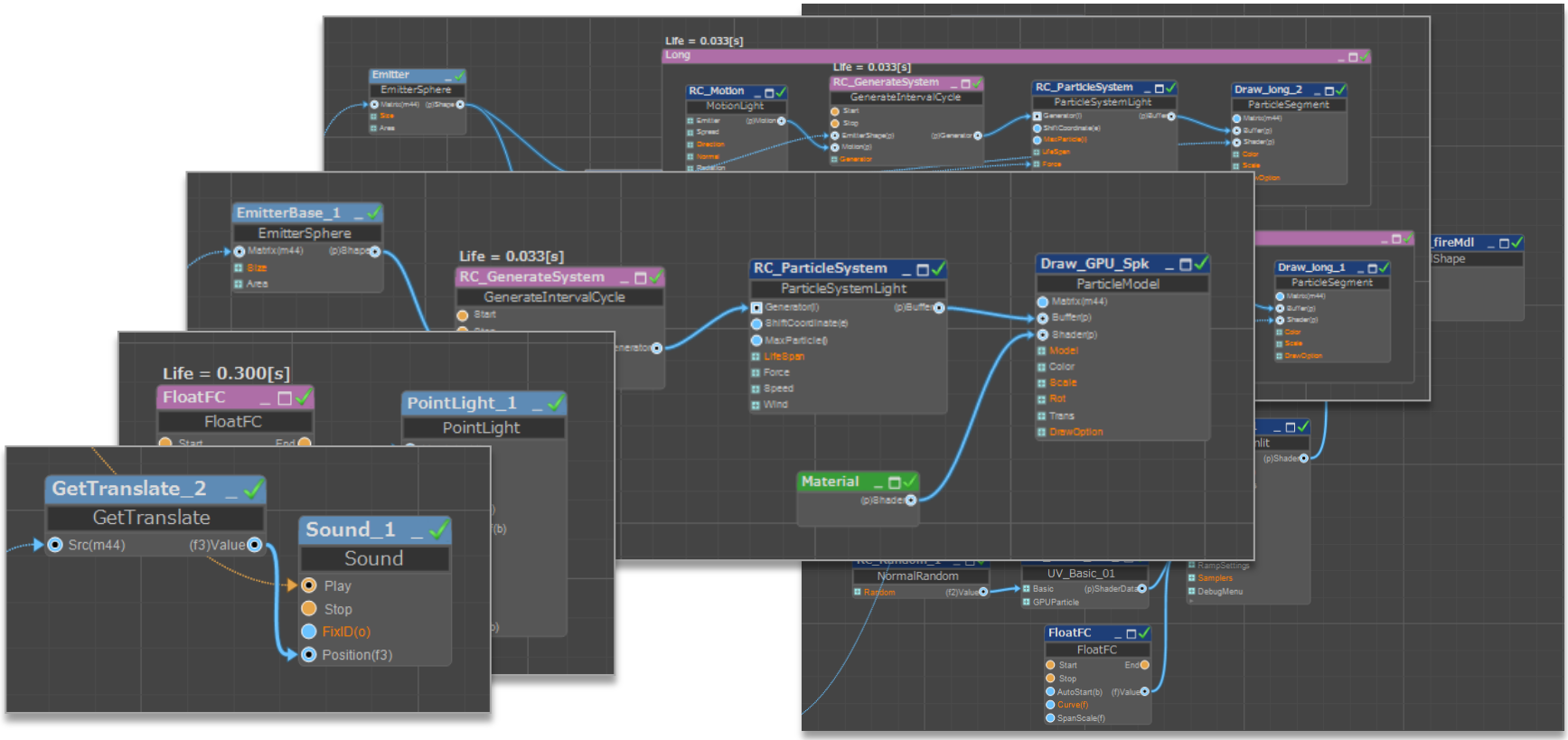


FFXVのエフェクトデータ紹介

描画計算
何を描画するか
Particle、Mesh、Spriteなど

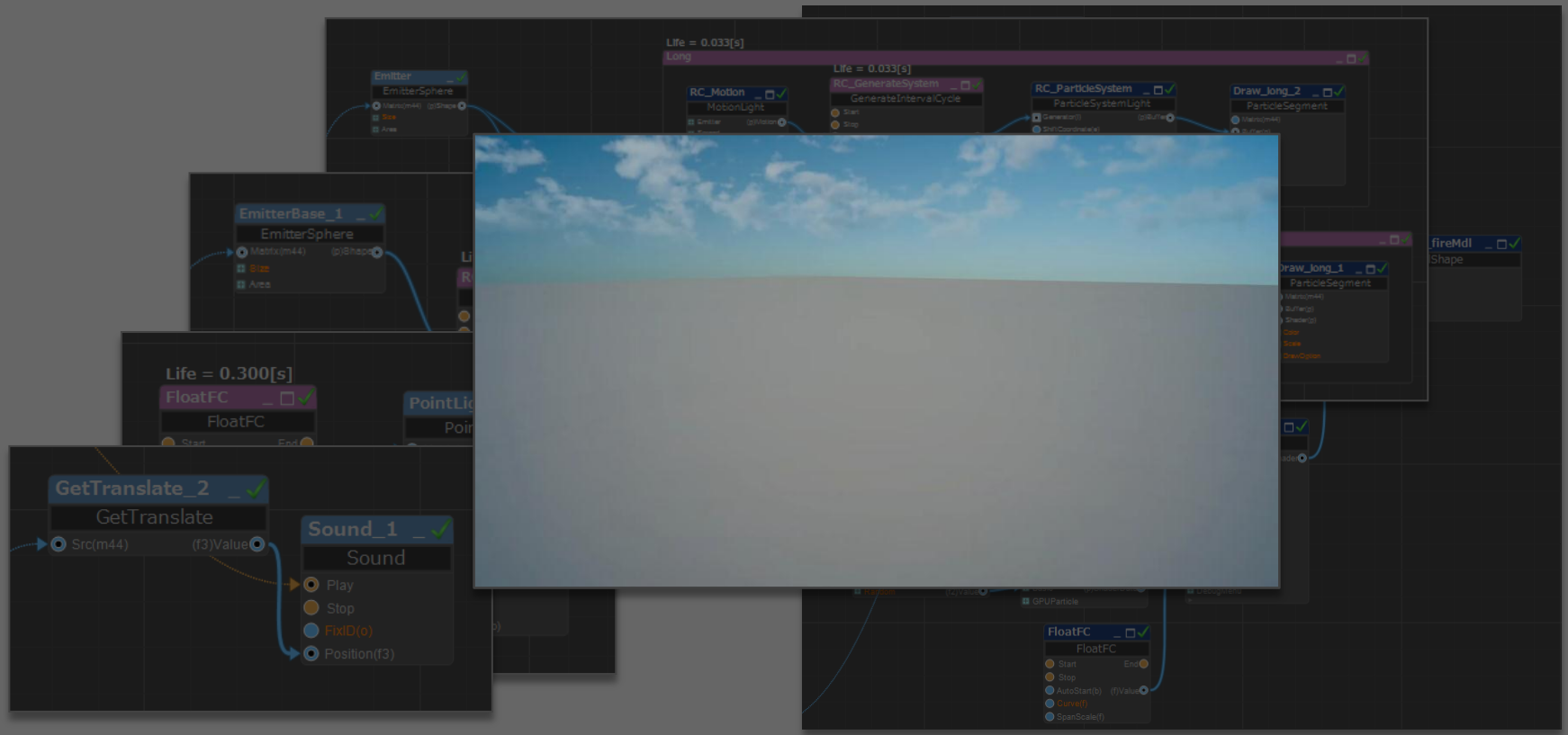


FFXVのエフェクトデータ紹介



CEDEC2015
Computer Entertainment Developers Conference

FFXVのエフェクトデータ紹介



アジェンダ

- 次世代機でのエフェクトデザインの一例
  - FFXV-EPISODE DUSCAE-におけるエフェクトデザイン
  - Luminous VFX Editorの紹介
- ノードベースエフェクトツールのメリット・効果的運用
  - ノードベースのメリット
  - Luminous VFX Editorのツール設計

← アーティスト・TA・プログラマ向け

- 実現時の課題とその解決方法
  - 実装時の技術的課題
  - Luminous VFX Editorにおける解決方法

ノードベースのメリット

ノードベースのメリット

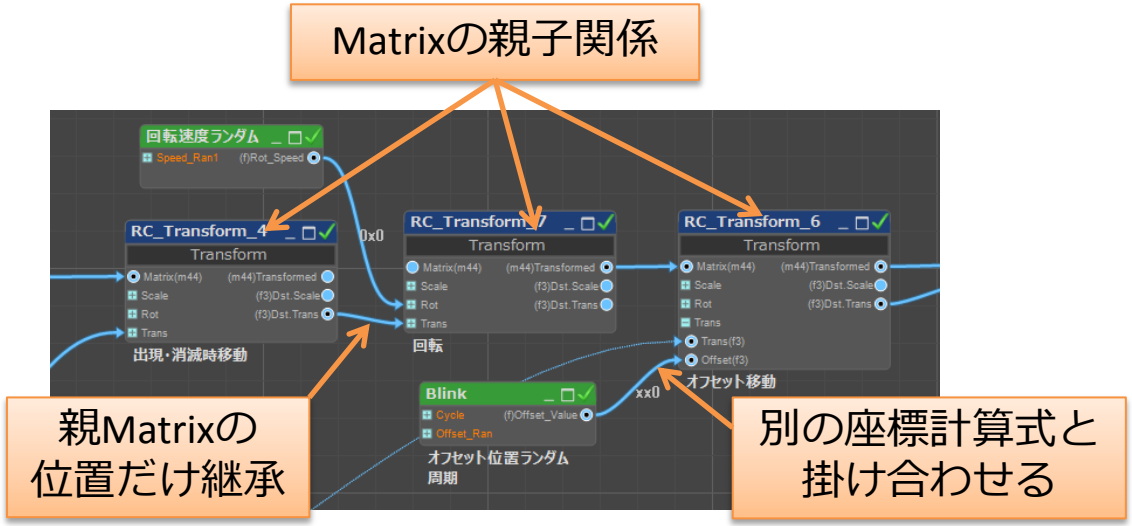
1．柔軟性・自由度が高い

2．外部とのやり取りが簡単

3．プログラマとアーティストの
　　作業分担がしやすい

メリット１：柔軟性・自由度が高い

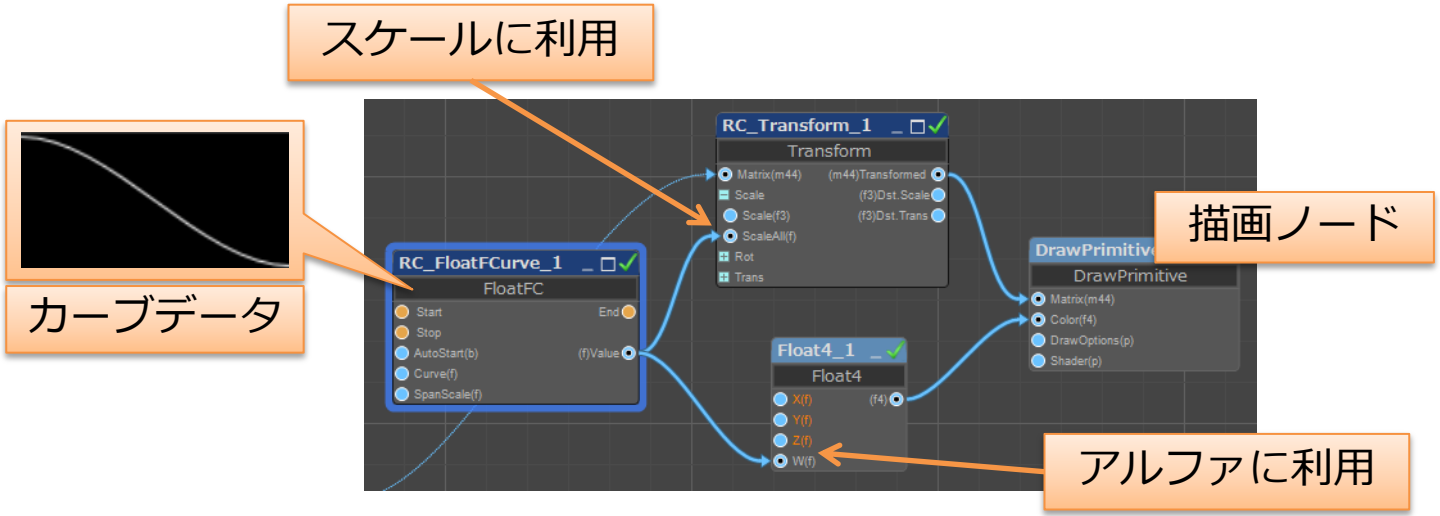
- つなぎ方次第でどんな計算もできる



例：蝶の座標計算（一部抜粋）

メリット１：柔軟性・自由度が高い

- 例）Fcurveノード



例：スケールとアルファを同じカーブでフェードさせる

メリット２：外部とのやり取りが簡単

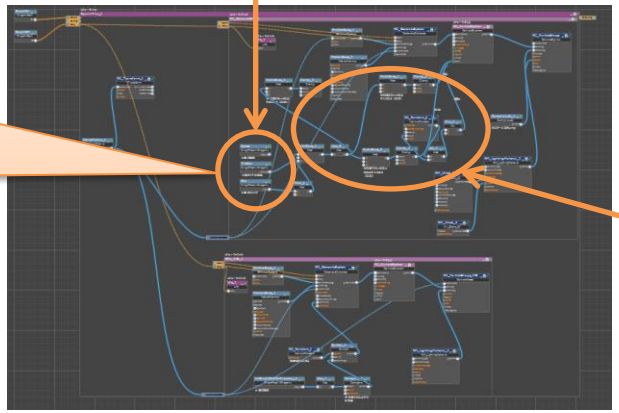
- 外部パラメータの例
  - 入力：時間、風の強さ、車のスピード、…
  - 出力：エフェクトの位置、描画領域、…
- 外部シグナルの例
  - 入力：フェードアウト開始タイミング、…
  - 出力：サウンドを鳴らすタイミング、…

メリット2：外部とのやり取りが簡単

- 外部パラメータ/シグナル入出力ノード

外部から値を
もらうためのノード

グラフ上の必要に
なった箇所に置く

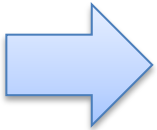


外部入力値を計算で
自由に加工できる

例：車の砂煙エフェクト

メリット３：作業分担がしやすい

- プログラム更新なしで新機能を作れる
  - アーティストがプログラマに頼らず、新しいエフェクトを作れる
  - 試作しやすく、プログラマにとっても便利



無駄なイテレーションが発生しない

メリット３：作業分担がしやすい

- 例）揚陸艇が起こす風煙
  - アーティスト要望

「煙を地面の傾き方向に
流したいけど、
起伏が激しいところでは
いい感じに無視したい」

プログラム更新なしにツール上で実装

ノードベースのメリット（まとめ）

1．柔軟性・自由度が高い

2．外部とのやり取りが簡単

3．プログラマとアーティストの
　　作業分担がしやすい

LUMINOUS VFX EDITORの
ツール設計

ツール開発で目指したこと

ノードベースのメリットを活かした
柔軟性の高いツール

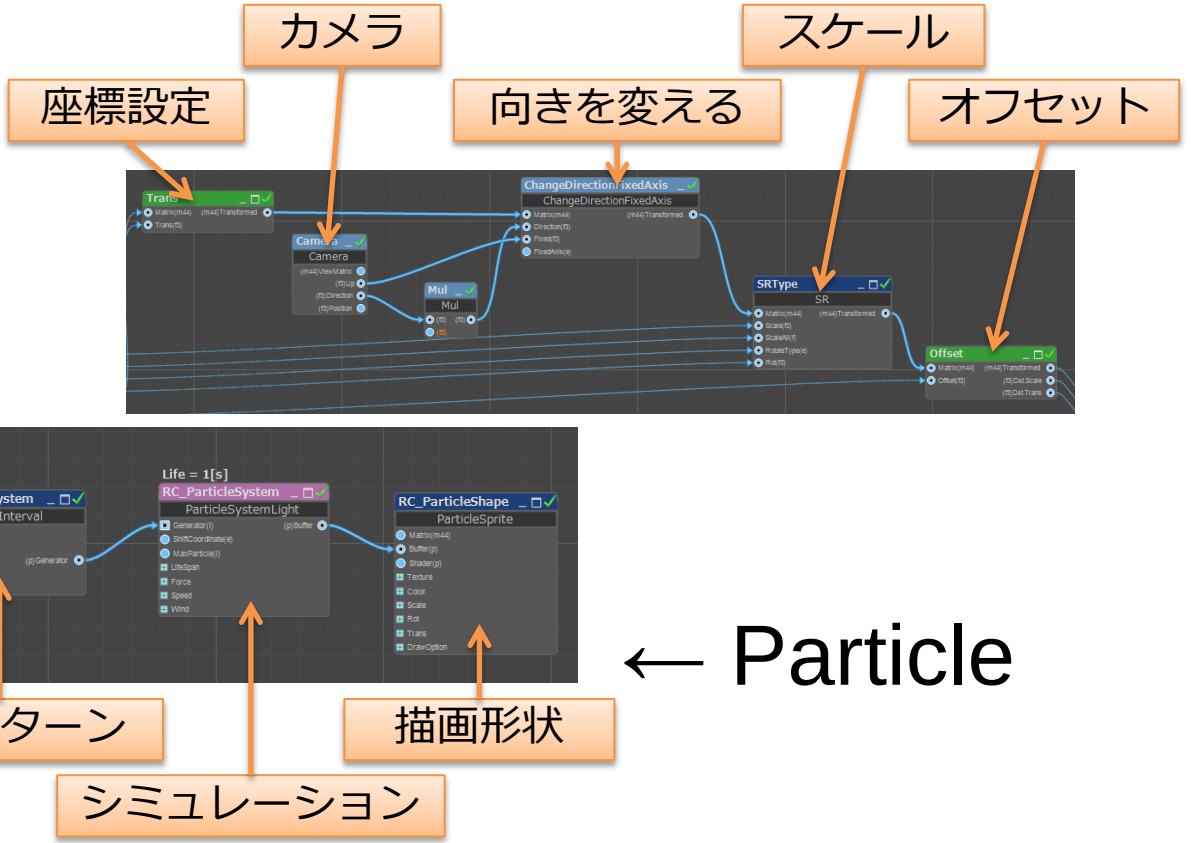
アーティストが学習しやすく
使いやすいツール

ノードベースのメリットを活かすためには

- 個々のノードをできるだけ小さく設計
  - いろんな場所で使いまわせるように
- 大きな機能はノードを組み合わせて作る
  - 巨大なノード設計は大抵失敗する…
    - あるノードに入れた機能が他でも欲しくなる

ノード分離の例

Billboard →



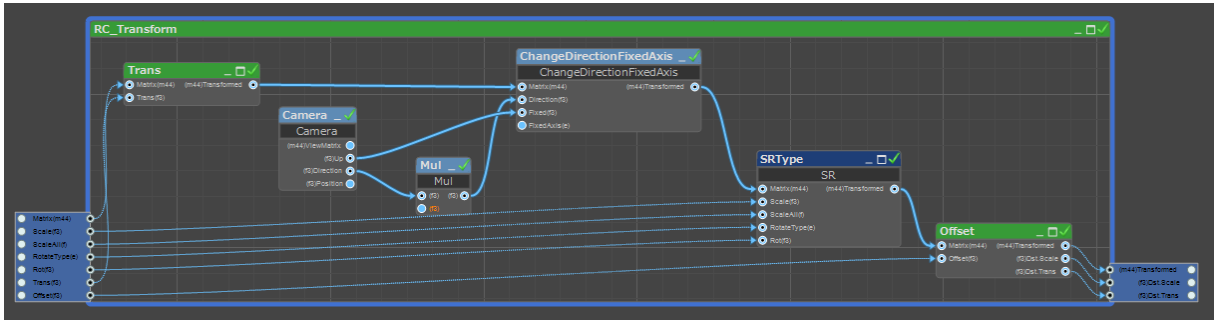
← Particle

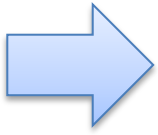
課題

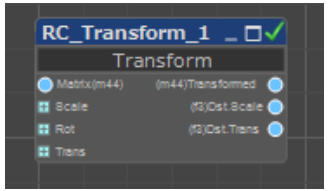
- 個々のノードを細かくすると…
  - データを作る際の手間が増えてしまったり
  - ツールの学習が難しくなったり
    - たくさんのノードを覚えないといけない
    - ノード単体では動かない

課題解決のアプローチ

- 「Reference Control」を導入
  - 複数のノードを１つのノードとして扱う機能







ReferenceControl化

Reference Controlのメリット

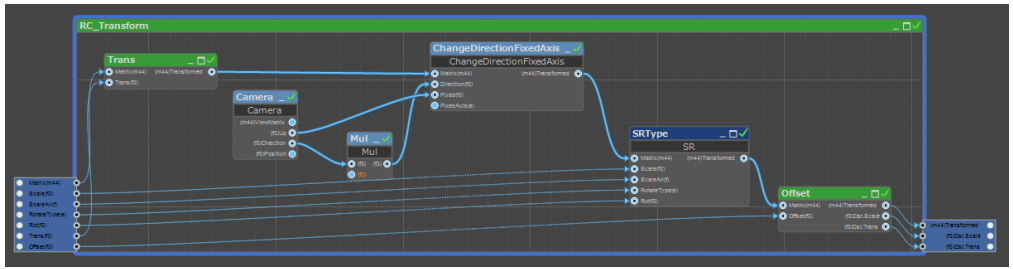
- 自由度を選べる
  - １つを大きくして座標計算から描画まで行う
  - １つを細かくして自由度を高くする
  - アーティストが目的に合わせて作れる

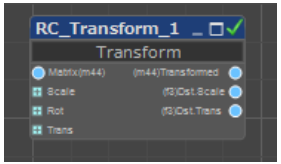
Reference Controlの機能を充実させていくことで
ツールの利便性が格段に上がる

Reference Controlの機能拡充（その１）

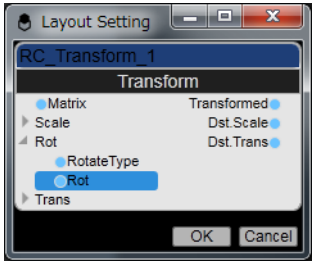
- ReferenceControlの編集機能を追加

ツールのグラフ上で作成できる





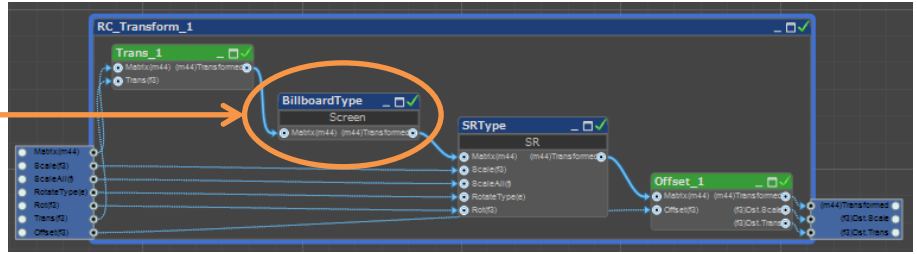
ポート編集機能
並び順, グループ化, 非表示



Reference Controlの機能拡充（その２）

- 「ベースノード」の概念を追加

向きを変える部分を
ベースノード化



中身のノードを選択して
動作を変えることができる

プロパティウィンドウ

結果

- プログラマは
  ノードを汎用性の高い設計にできる
- アーティストは
  ノードを扱いやすいように設計できる

ノードベースの利点を活かしつつ、
使いやすいツールになった！

アジェンダ

- 次世代機でのエフェクトデザインの一例
  - FFXV-EPISODE DUSCAE-におけるエフェクトデザイン
  - Luminous VFX Editorの紹介
- ノードベースエフェクトツールのメリット・効果的運用
  - ノードベースのメリット
  - Luminous VFX Editorのツール設計
- 実現時の課題とその解決方法
  - 実装時の技術的課題
  - Luminous VFX Editorにおける解決方法

← プログラマ向け

実装時の技術的課題

ノードベースとリアルタイムエフェクト

- ノードベースのツール自体は前例あり
  ⇒ 実装は問題なくできるはず

「リアルタイム」「エフェクト」を
扱うノードベースツールはまだ少数

- Luminous VFX Editor開発時、リアルタイムエフェクト特有の課題に遭遇

実装時の課題概要

- ノードベースのエフェクトツールの課題
  - エフェクト特有の寿命の概念と相性が悪い①
- ランタイムを実装する上での課題
  - 重い（メモリコスト） ②
  - 遅い（処理コスト） ③

実装時の課題概要

- ノードベースのエフェクトツールの課題
  - エフェクト特有の寿命の概念と相性が悪い
- ランタイムを実装する上での課題
  - 重い（メモリコスト）
  - 遅い（処理コスト）

エフェクト特有の「寿命」の問題

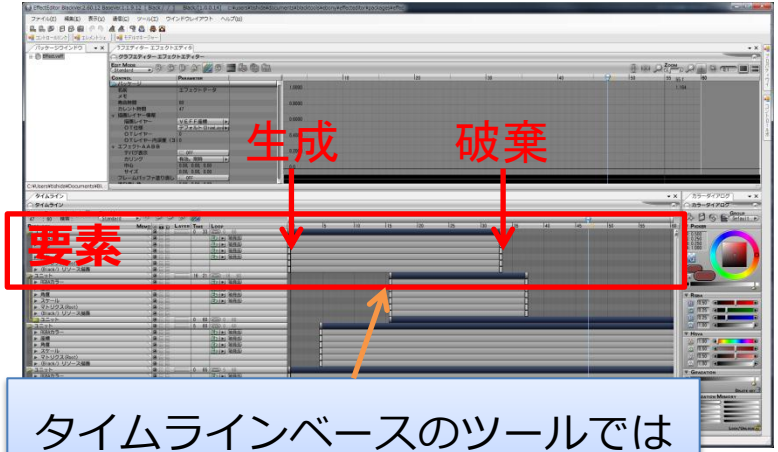
- エフェクトの要素の特徴
  - 短寿命
  - 大量

ゲームアセットの
中では特徴的

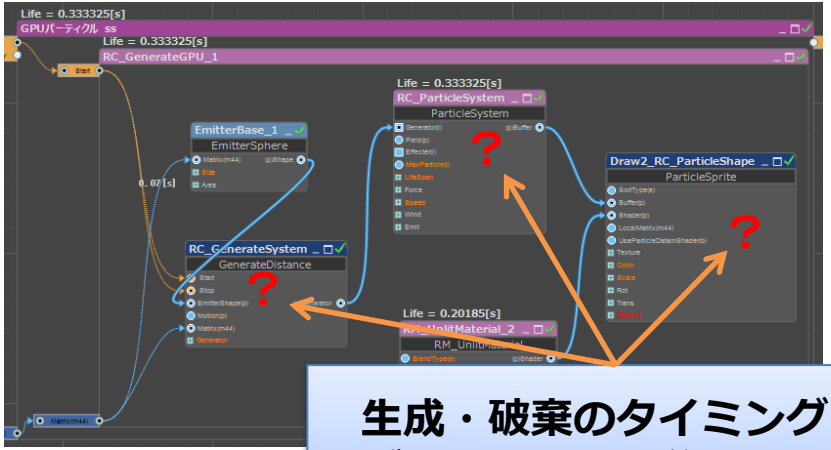
個々の火花は数フレー
ムで消えるが、大量

ノードベースは寿命の概念と相性が悪い

- エフェクトツールにおける寿命の表示は



タイムラインベースのツールでは
タイミングは一目瞭然



生成・破棄のタイミングは
グラフを見ても分からない

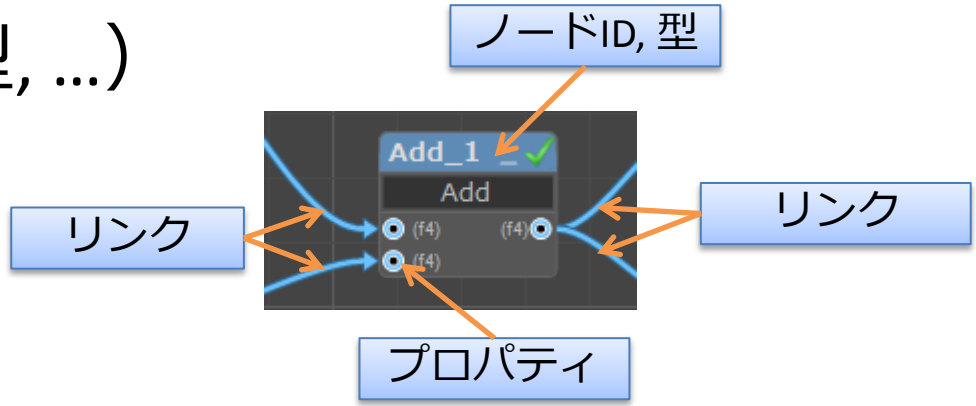
ノードベースでエフェクトは扱いにくい

実装時の課題概要

- ノードベースのエフェクトツールの課題
  - エフェクト特有の寿命の概念と相性が悪い
- ランタイムを実装する上での課題
  - 重い（メモリコスト）
  - 遅い（処理コスト）

ノードの消費メモリ

- ノード1個に必要なメモリ量
  - ノード情報（ID, 型, …）
  - プロパティ
  - ポート
  - リンク情報
  - …



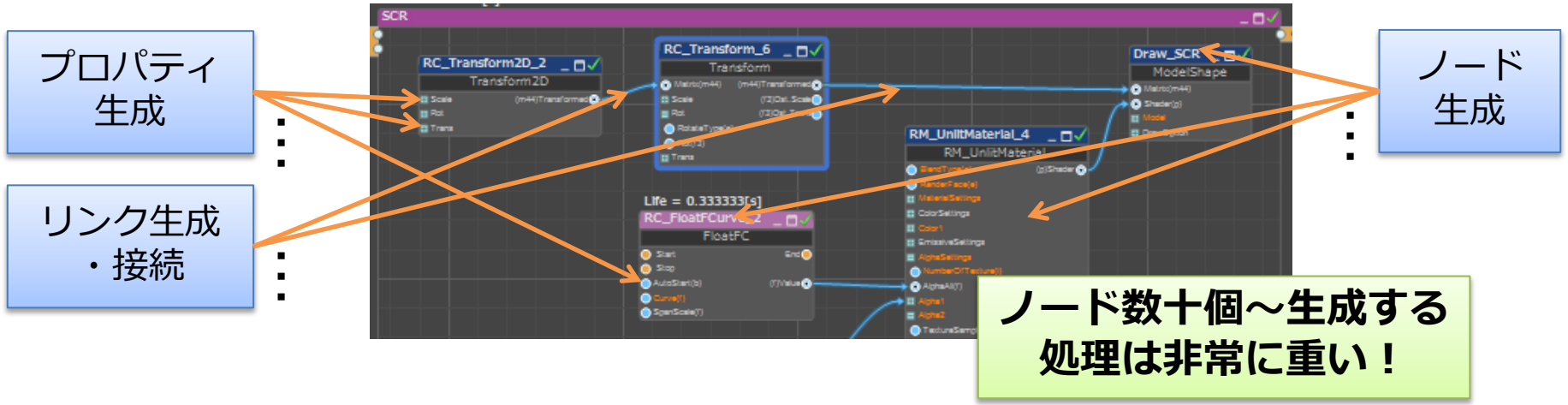
高々足し算一個の情報には大きすぎる！

実装時の課題概要

- ノードベースのエフェクトツールの課題
  - エフェクト特有の寿命の概念と相性が悪い
- ランタイムを実装する上での課題
  - 重い（メモリコスト）
  - 遅い（処理コスト）

ノード生成の処理コスト

- ノード生成処理をナイーブに実装すると…



生成はロード時に行いたいが…

リアルタイムエフェクトと動的生成

エフェクト内の要素は
その数もランダムに
決まるものがある

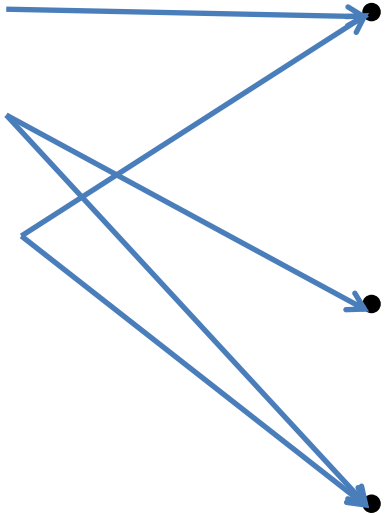
ノードベースのエフェクトツールでは
重いノード動的生成処理は避けられない

LUMINOUS VFX EDITOR における解決方法

Luminous における解決策概要

課題

- ノードと寿命
- メモリコスト
- 処理コスト



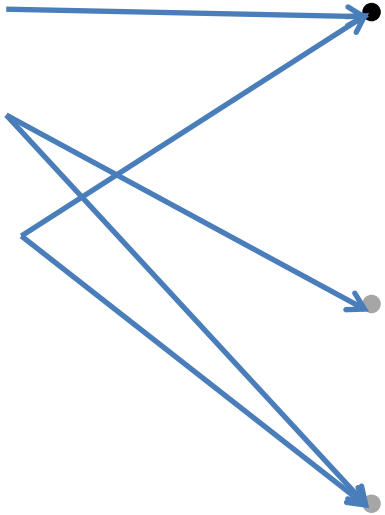
解決策

- ノードのグループ化
  - 寿命管理
  - 生成破棄コスト削減
- データ構造の階層化
  - メモリコスト削減
- ノード群の式化
  - コスト削減

Luminous における解決策概要

課題

- ノードと寿命
- メモリコスト
- 処理コスト

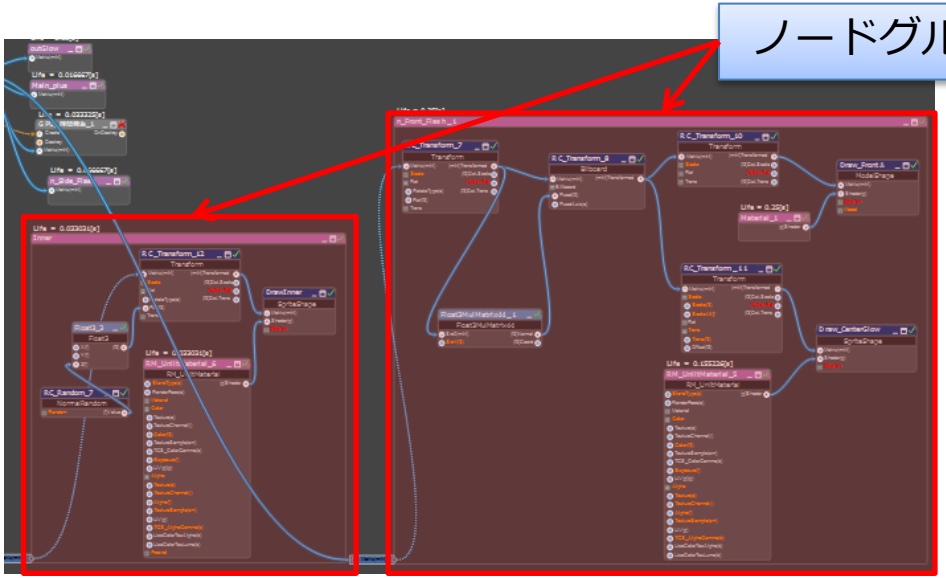


解決策

- ノードのグループ化
  - 寿命管理
  - 生成破棄コスト削減
- データ構造の階層化
  - メモリコスト削減
- ノード群の式化
  - コスト削減

ノードのグループ化によるコスト減

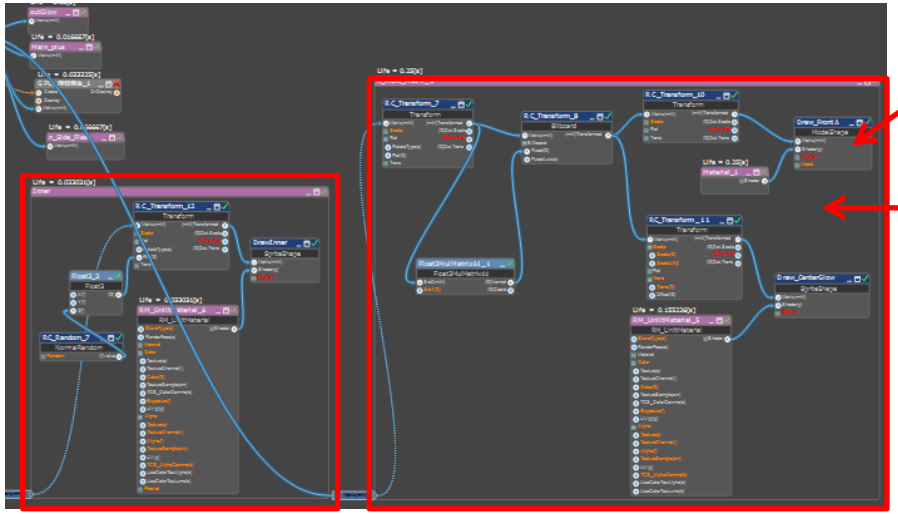
- ノードを個別に生成・削除は高コスト
  ⇒ノードのグループ単位で生成・削除



- ノード個数分のメモリ確保･解放 ⇒ 1回に
- 解放時のリンク解除が不要に

ノードグループ化のトレードオフ

- グループ単位で生成・削除ということは…



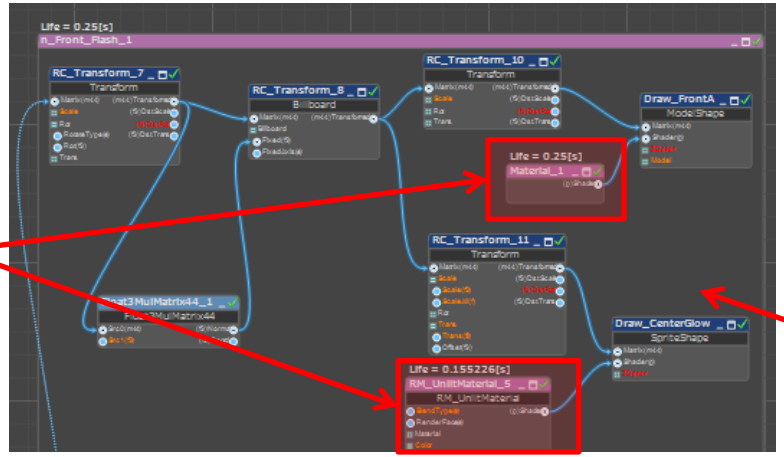
ノードグループ内で、個別にノードの生成・削除は不可

動的にノード種別を決定するような、動的グラフも不可

柔軟性低下だが、
トレードオフとして許容

寿命管理のアプローチ

- 従来の寿命管理⇒ 個々の要素に設定
- Luminous VFX Editorでのアプローチは…

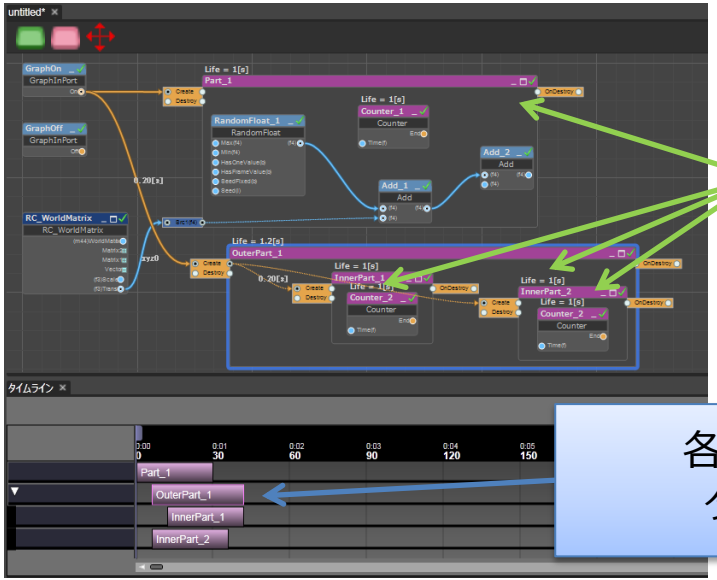


必要な部分の
寿命の設定

残りのノードと
全体の寿命は自動管理

タイムラインビューによる寿命の可視化

- 寿命の分かりにくさへの対応
  - タイムラインビューで寿命を可視化



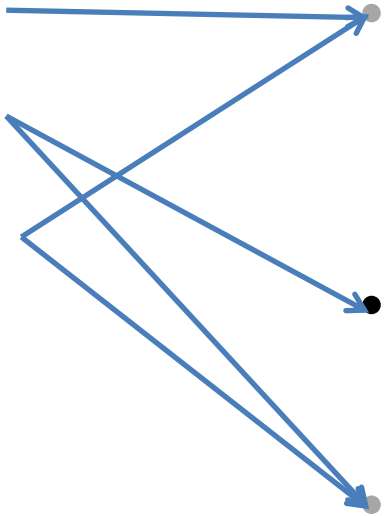
ノードグループ

各ノードグループの寿命を
タイムライン形式で表示

Luminous における解決策概要

課題

- ノードと寿命
- メモリコスト
- 処理コスト

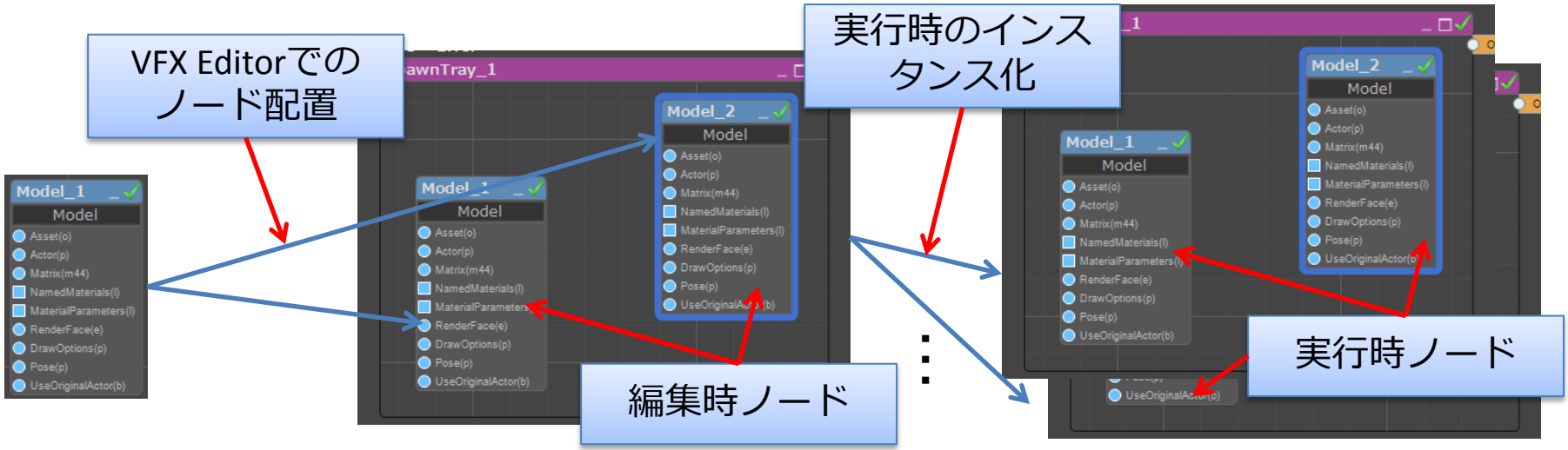


解決策

- ノードのグループ化
  - 寿命管理
  - 生成破棄コスト削減
- データ構造の階層化
  - メモリコスト削減
- ノード群の式化
  - コスト削減

VFX Editorのデータレイヤー

ノード　　編集時グラフ　　実行時グラフ

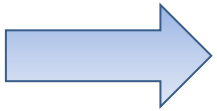


個数１：　　N　　：　　N×M

データ階層化によるコスト減

ノード：編集時ノード：実行時ノード
⇒ 個数は１：Ｎ：Ｎ×Ｍ

- 実行時ノードに必要なデータは一部
  - 例: プロパティの初期値は編集時ノードでOK

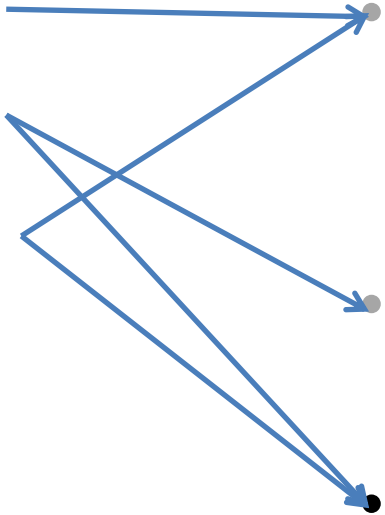


- データを適切な階層に持たせ、コスト減
- 編集時グラフはロード時に構築できる

Luminous における解決策概要

課題

- ノードと寿命
- メモリコスト
- 処理コスト

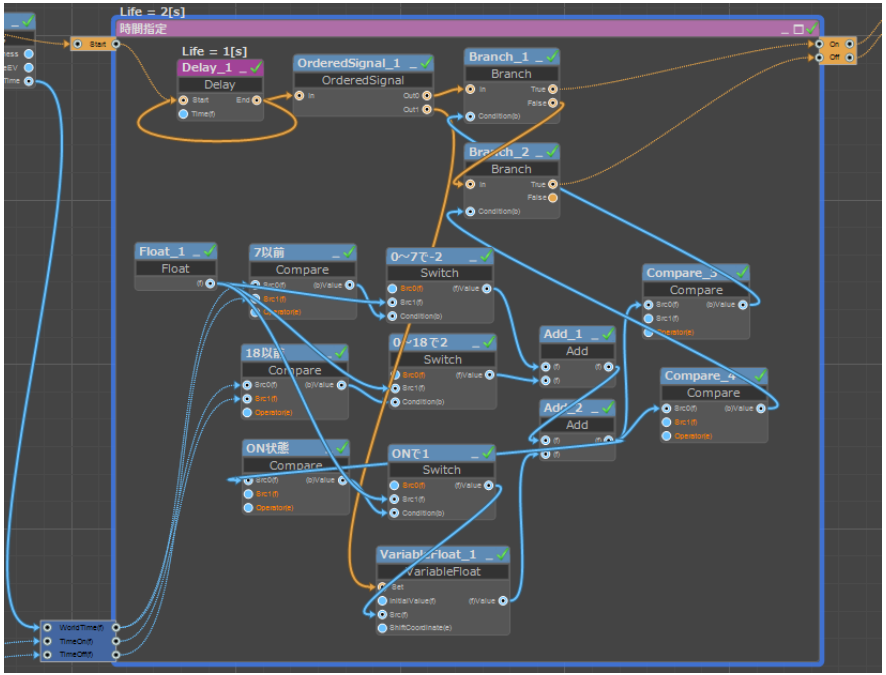


解決策

- ノードのグループ化
  - 寿命管理
  - 生成破棄コスト削減
- データ構造の階層化
  - メモリコスト削減
- ノード群の式化
  - コスト削減

ノードグループの式化

- 式を書くと大量のノードができてしまう



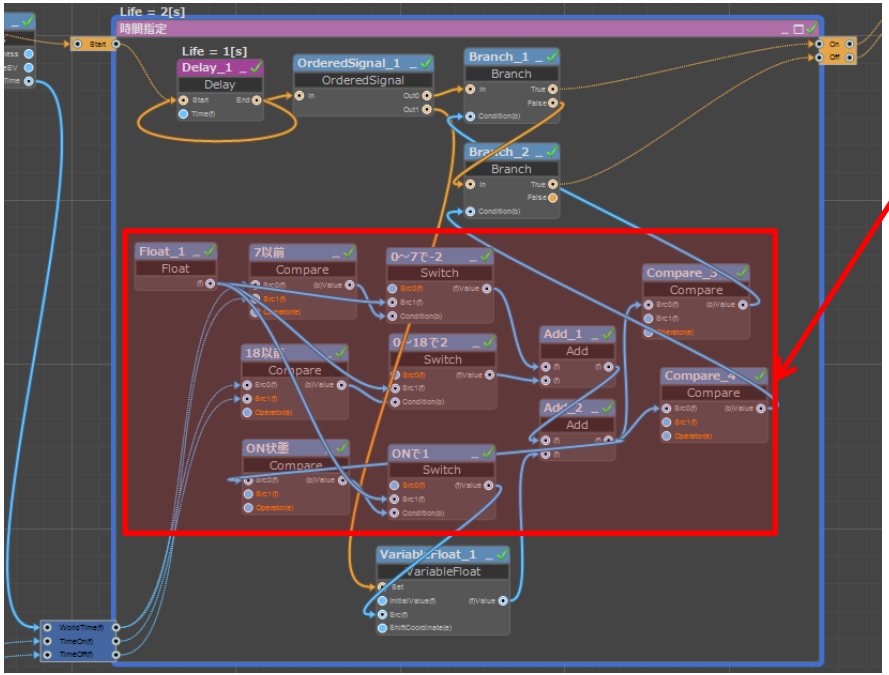
ノードの数が増えると…

- 消費メモリ増
- 処理時間増

例：蝶の出現ロジック（一部抜粋）

ノードグループの式化

- ツールの最適化機能により、計算ノード群を変換



仮想的な計算ノードとして扱う

実行時は1ノードになり

- 消費メモリ減
- 処理時間減

本セッションのまとめ

- 次世代機でのエフェクトデザインの一例
  ⇒ フォトリアルに描くファンタジーエフェクト
  ⇒ エフェクトが環境の影響を受ける別軸からのリアリティ表現

- ノードベースエフェクトツールのメリット・効果的運用
  ⇒「柔軟性・自由度」「外部とのやり取り」「容易な作業分担」
  ⇒ アーティストが学習しやすく使いやすいツール（ ReferenceControl ）

- 実現時の課題とその解決方法
  課題:「寿命」「重い」「遅い」
  解決方法:「グループ化」「データ構造階層化」「式化」

Q&A