Python based Shaderpipeline in VRED

Ideenkultivierung GmbH

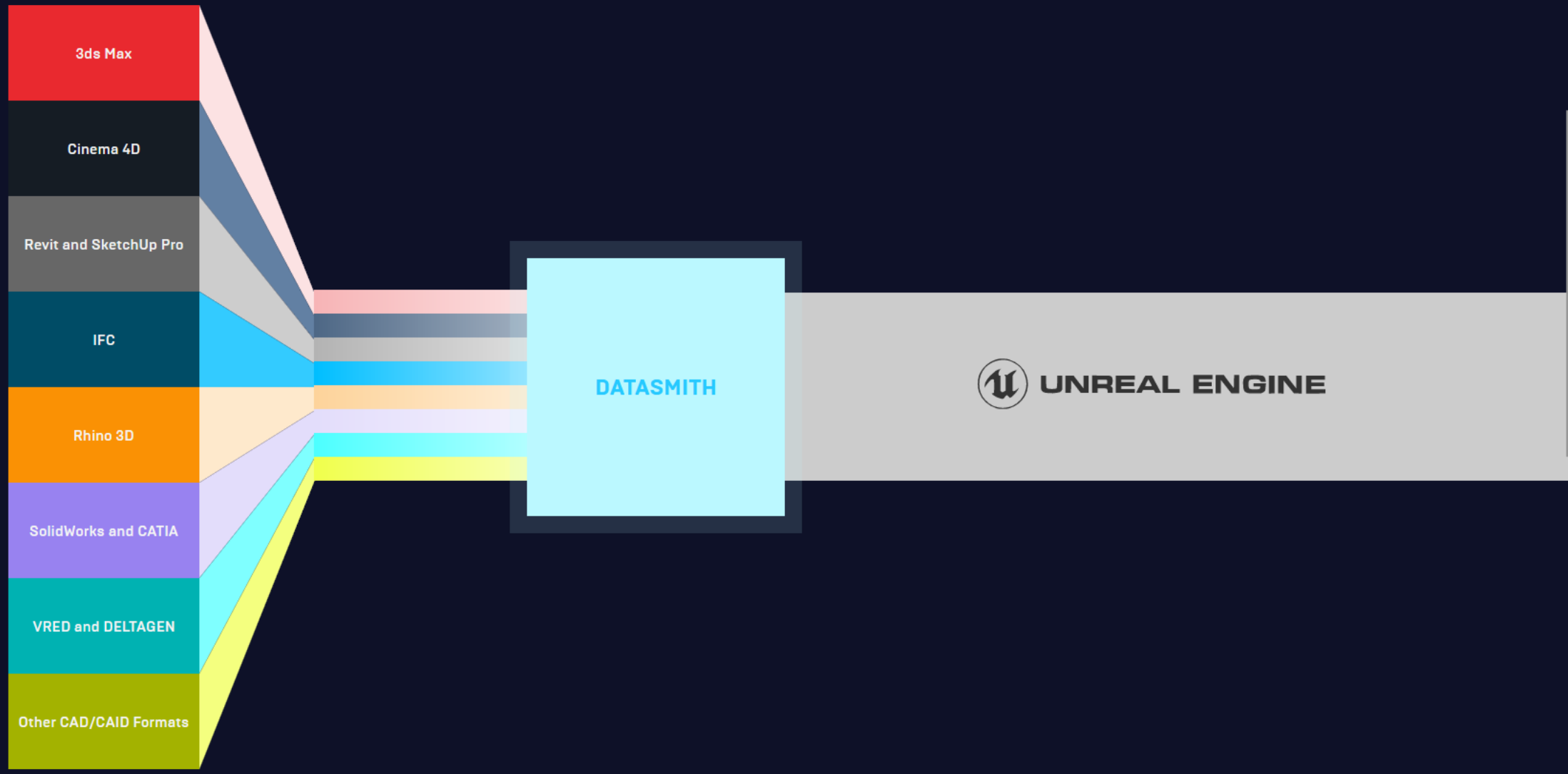
- 3D-Design
- Interactive Visualization
- Software-Development
- Trainings (e.g. CAD, VRED, Scripting)

Motivation

Bilder:

- Automodell ungeshaded
- Automodell geshaded
- Scenegraph ungeordnet

Examples in the wild

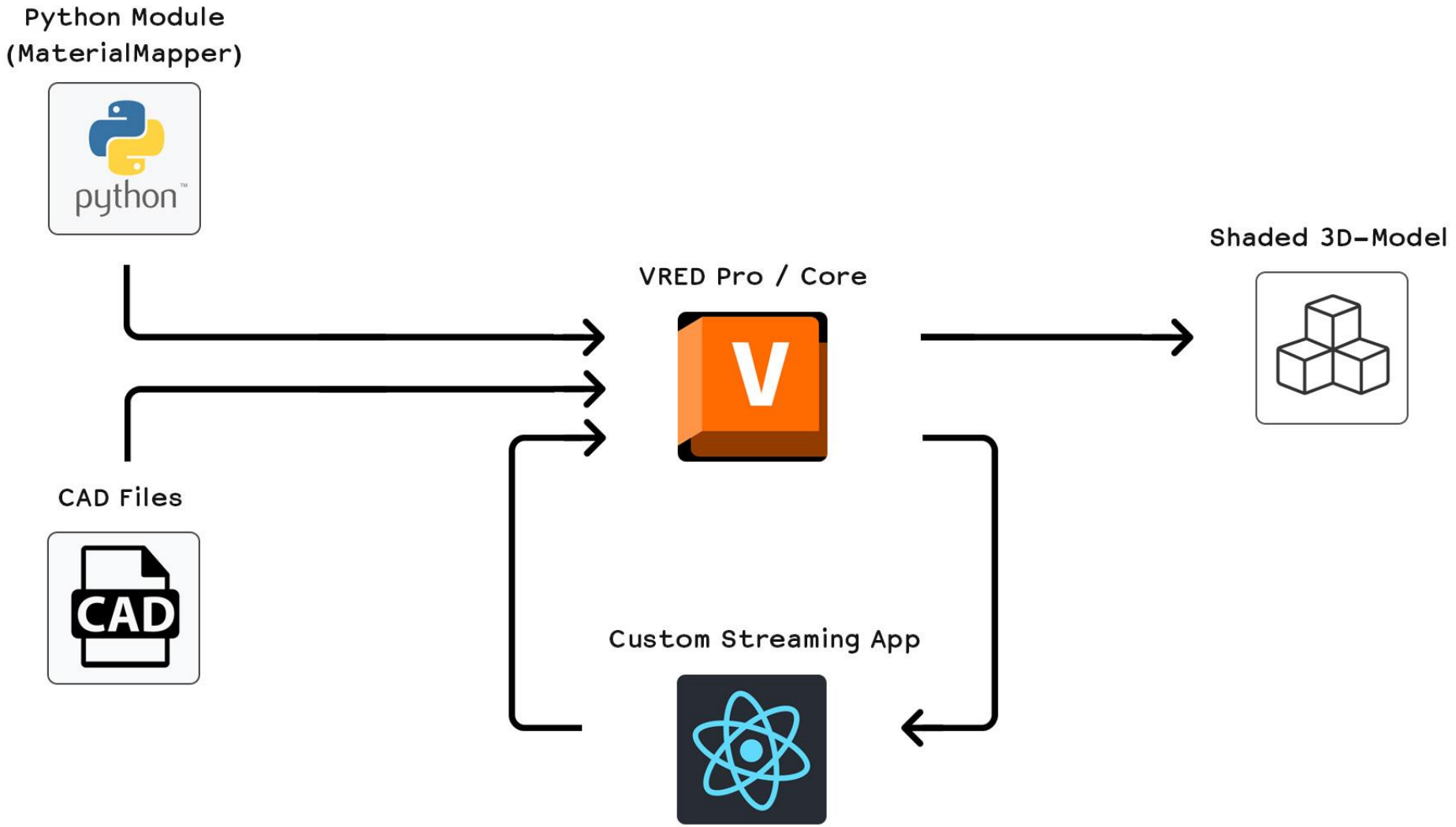


"With Datasmith, I can literally do the same thing I did in four weeks in one day and that is magic."

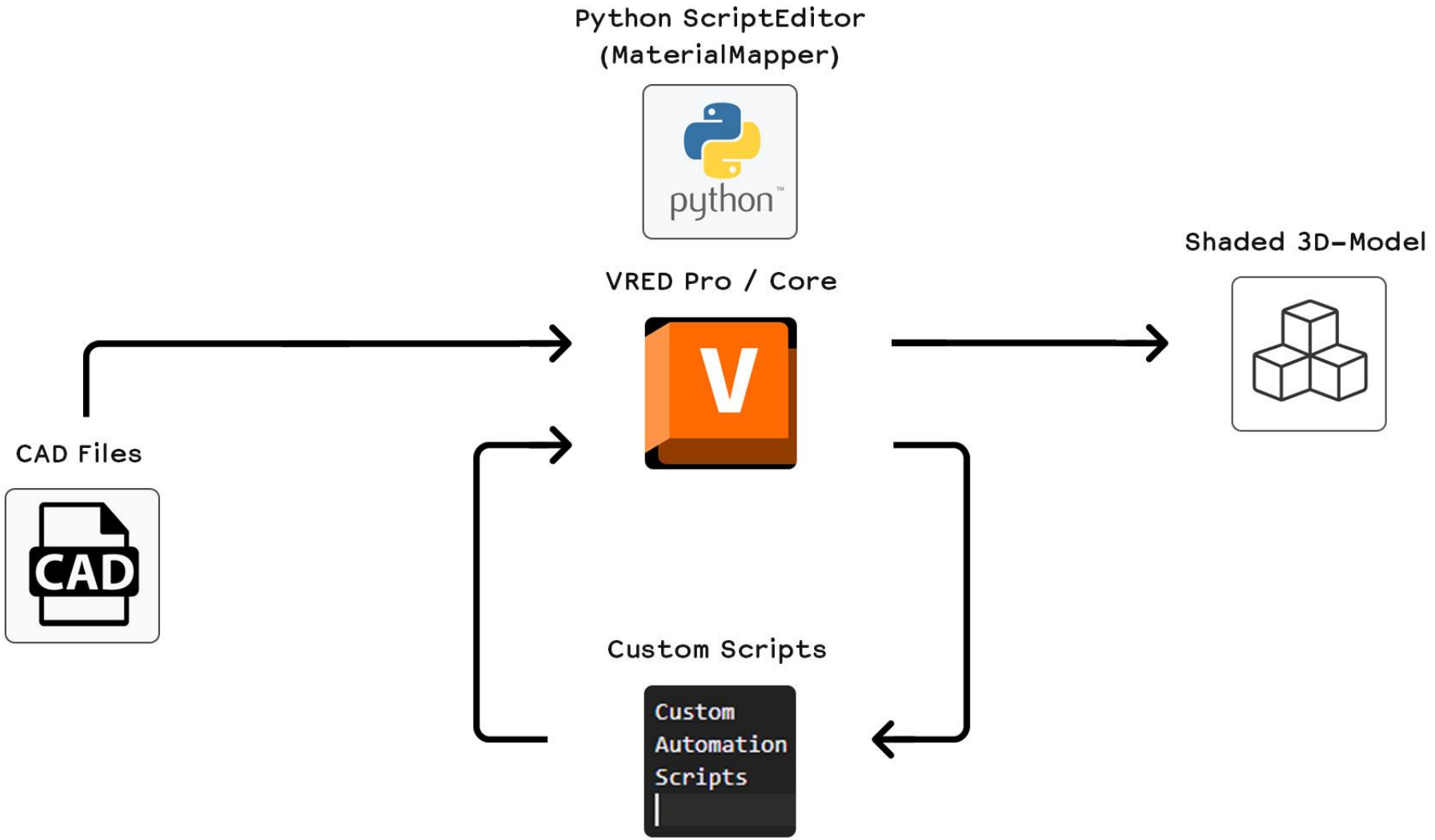
Implementing Shaderpipelines in VRED

- Use an automated shaderpipeline in VRED to do the time consuming work
- Automatically apply materials by using material libraries
- Use color coded base-materials or material names to shade the model
- Use fuzzy matching to shade even faster

Project Overview



Project Overview



Prerequisits: Common Standards

rgb(176, 30, 104)

↓

Carpaint

rgb(220, 53, 53)

↓

Chrome

rgb(244, 157, 26)

↓

Plastic

rgb(255, 225, 93)

↓

Glas

Prerequisits: Common Standards

carpaint_#1

↓

Carpaint

chrome_door_handle

↓

Chrome

plastic_black

↓

Plastic

glas_front_light

↓

Glas

Matching Colors

- Look at all materials in the imported file
- Get the RGB values of the diffuse channel using the field access
- Use lookup table to find a matching color
- If there is a matching color -> replace material!

Matching Names

- Look at all materials in the imported file
- Get the name of the material
- Use lookup table to find a matching name
- If there is a matching name -> replace material!

Fuzzy Matching

chrome-door

name-mapping.csv

```
body_main   -> Carpaint
chrome_door   -> Chrome
plastic_black   -> Plastic
glas_front_light   -> Glas
```

Exact match
100%
not possible!

Fuzzy Matching

chrome-door

name-mapping.csv

body_main -> Carpaint

chrome_door -> Chrome

plastic_black -> Plastic

glas_front_light -> Glas

Similar
> 90%
Match!

Automate Everything