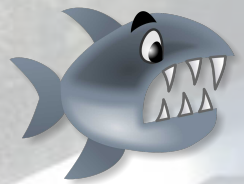
Schichten einer Engine: Vom Metall bis zum Live-Live-Editing in Shark 3D.

by Folker Schamel

spieleentwicklertreffenNRW • Düsseldorf • 19. Oktober 2009

Version: 23 October 2009



Shark 3D™ powers award winning Dreamfall on console and PC

DREAMFALL
THE LONGEST JOURNEY

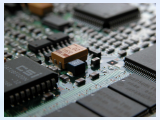
Layers in Shark 3D

Game Designers

- Artist
- Level Designer
- Programmer
- ...

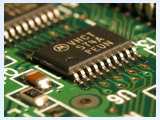
Engine Core

- Dependency Manager
- Renderer/Sound Physics/...
- Actors
- ...



Hardware Abstraction

- Thread Pools
- Shader Programs Manager
- Memory Manager
- IO
- Resource Manager



Hardware

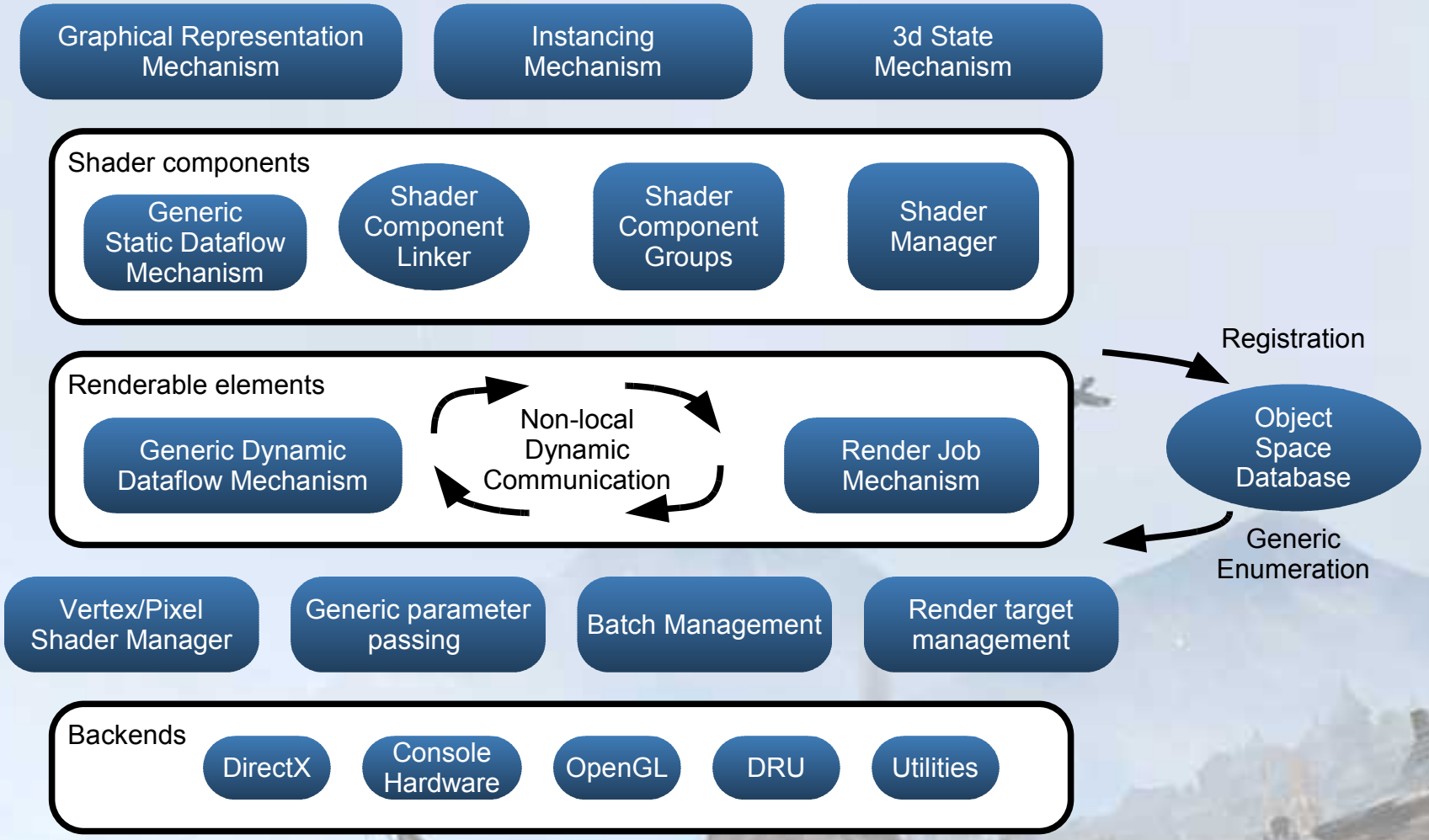
- CPU
- GPU
- Memory
- Network
- ...

Live Editing of Every Layer[1,2]

- Game development is designer/artist driven.
- Turn-around cycles are the bottleneck
  - especially for console games.

[1] Except hardware ;-)
[2] Not all game designers ;-)

Highly Optimized Renderer Layers



Shark 3D Shader Components

Shark 3D™

std

- drvlightenum
- meshenter
- animactu
- modeswitch
- lightparam
- animgen
- modelmesh
- lightenum
- bundle
- variants
- lightenter
- collexec
- translmat
- group
- coloralpha
- totex
- fog
- constfloat
- regionenter
- duptexchan
- constmat
- redirect
- drvlightcoll
- constvec
- rectmesh
- projmat
- directtexchan
- multi
- addvec
- mulmat
- paintmesh

special

- plain
- projtotex
- screenparam
- filter
- particenter
- func
- multilight
- opticsdirect
- envmap
- opticstex
- billboardmesh

shadow

- enum
- lenparam
- perform
- occluder
- combineparam

user

- user 1
- user 2
- user 3
- user 4
- user 4
- user 5

Modularity Sample: Main rendering code is generic

Shark 3D™

Shark 3D's main rendering code:

```
s3d_CEngGfxTaskArray TaskArray;
s3d_CEngGfxCycle *Cycle = CollectNewCycle(
        Run, Cam, m_Trigger, TaskArray);

s3d_CEngUtilGfxElemJobBegin::AddGfxBegin(
        m_MsgHandler, m_Info.GetChars(), Cycle,
        m_DestProp, m_ClearParam, TaskArray, BeginMain);

s3d_CEngUtilGfxUtil::ExecTaskArray(TaskArray, 0);
```

The main rendering code is completely independent from particular advanced rendering features.

Modularity Sample: Rendering features in modules

Generic interface for implementing rendering modules in Shark 3D:

```
class s3d_CEngGfxElem: public s3d_CUtilRecogEyeBase
{
public:
    s3d_CEngGfxElem();

    virtual void GfxElemExec(
            s3d_CUtilRecogBase *GfxElemCtx,
            s3d_CUtilAtom *Trigger,
            s3d_CDrvVarBlk_cr ParamVarBlk,
            s3d_CEngGfxTaskArray &TaskArray);
};
```

Even advanced, non-local rendering techniques can be implemented in separate modules.

Examples: Different lighting techniques; multiple passes; rendering order; simple shadow volumes & shadow maps; advanced soft shadowing techniques; dynamic mirroring (planar, environment map etc.); post rendering effects; various render-to-texture techniques; effects requiring complex scene enumeration; PVS; ...

Layers in Shark 3D

Game Designers

- Artist
- Level Designer
- Programmer
- ...

Editors

- Game Editor
- Shader Editor
- Resource Editor

Tools Infrastructure

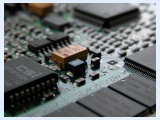
- On-the-fly Compilation
- Off-line Compilation
- Live-Live Editing

Live Editing

- Resource Manipulation
- Simple Actor Protocol (SAP)
- Editing Run-time controls

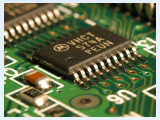
Engine Core

- Dependency Manager
- Renderer/Sound Physics/...
- Actors
- ...



Hardware Abstraction

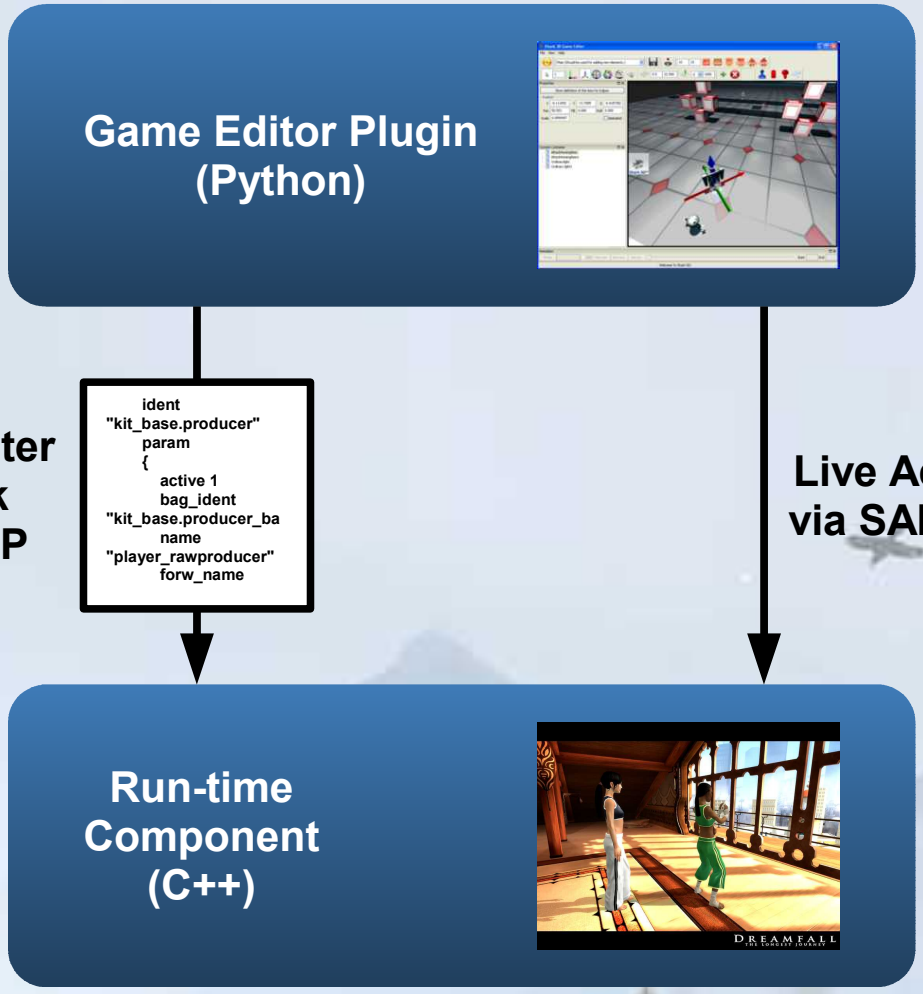
- Thread Pools
- Shader Programs Manager
- Memory Manager
- IO
- Resource Manager



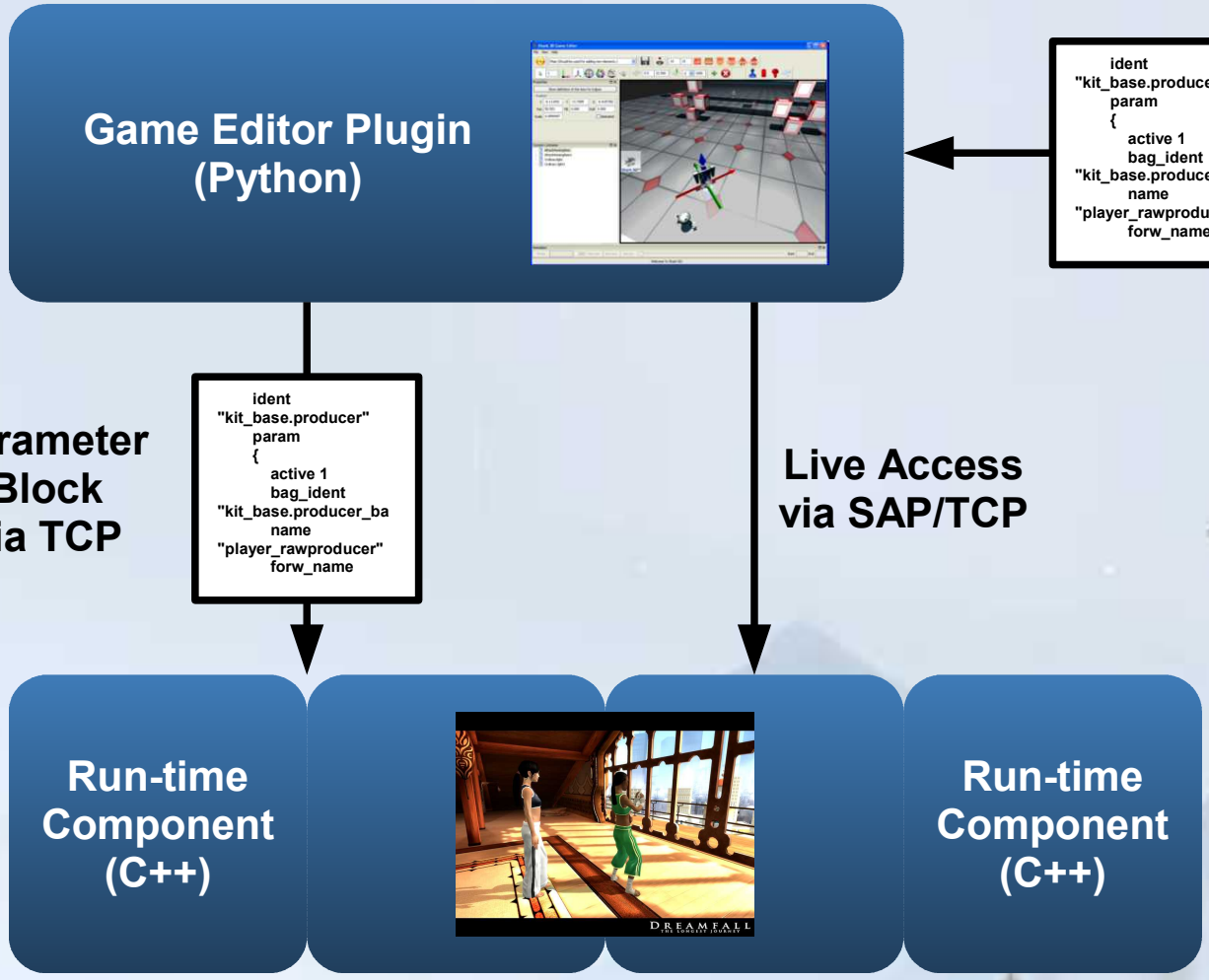
Hardware

- CPU
- GPU
- Memory
- Network
- ...

Implementing a Game Feature



Compositing a Game Feature



Why Python?

- Coding speed matters:
  Up to 10x faster than C++.
  Up to 5x faster than C#/Java.

- Proven for tools:
  For example Maya, XSI.

- Powerful, but very easy to learn for simple tasks.

- Python is cool!

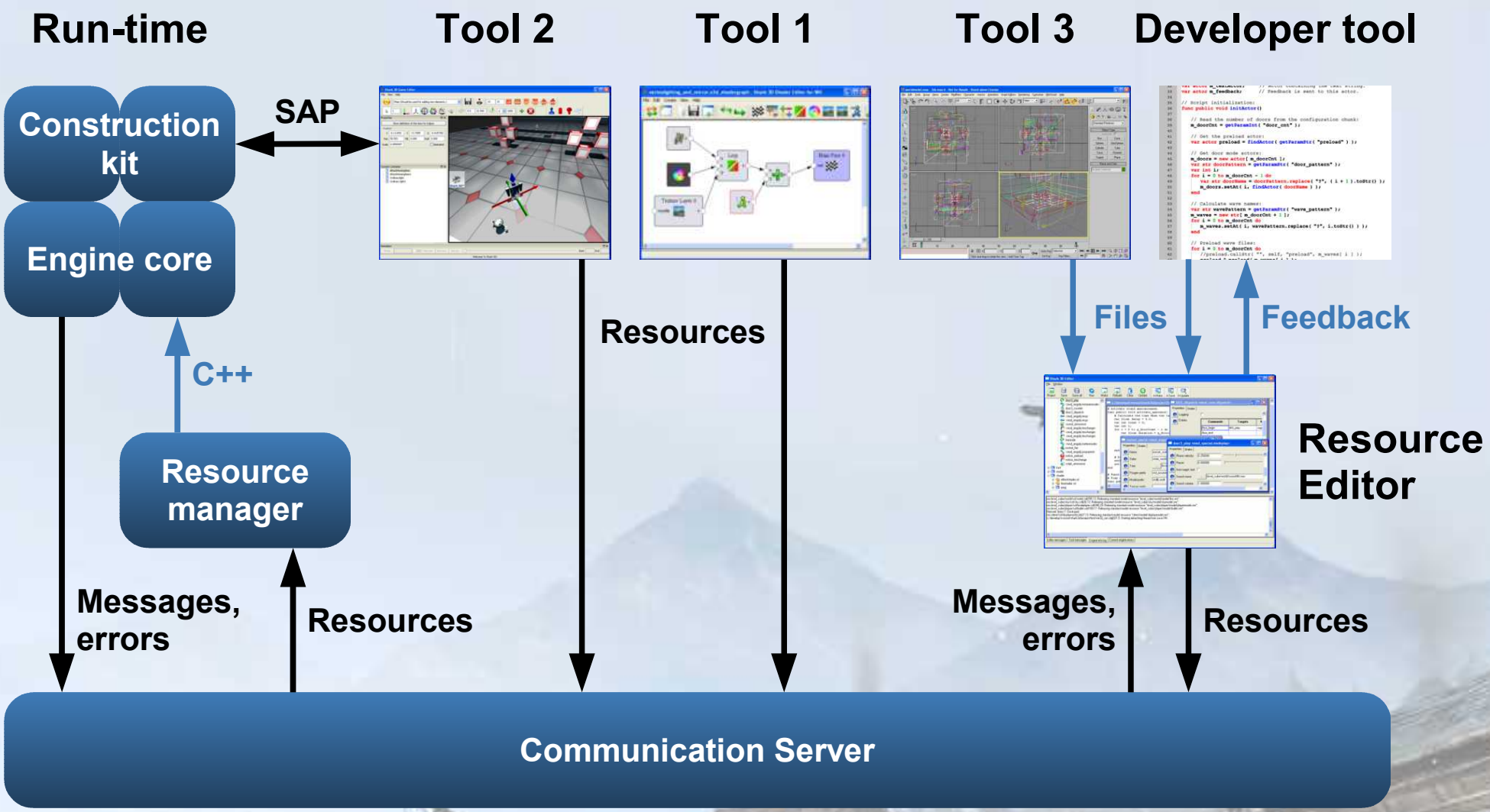
Advantages of the Python Layer

- Design: Separation of editor GUI and run-time.
- Beyond parameters: Many game features require GUI logic.
- Agility: Easy tool & GUI customizing for the game designer.
- Reusability: Better re-use of run-time components.
- Final run-time data differs from editor data.

Advantages of Parameter Block Templates

- Hide complex run-time features to the designer,
- Exposing parameters to designer later as needed.
- C++ developers are encouraged to re-usable components.
- C++ developers are encouraged to flexibility.
- Easily change functionality without recompiling C++ code.
- Simple editor data creates complex run-time data.

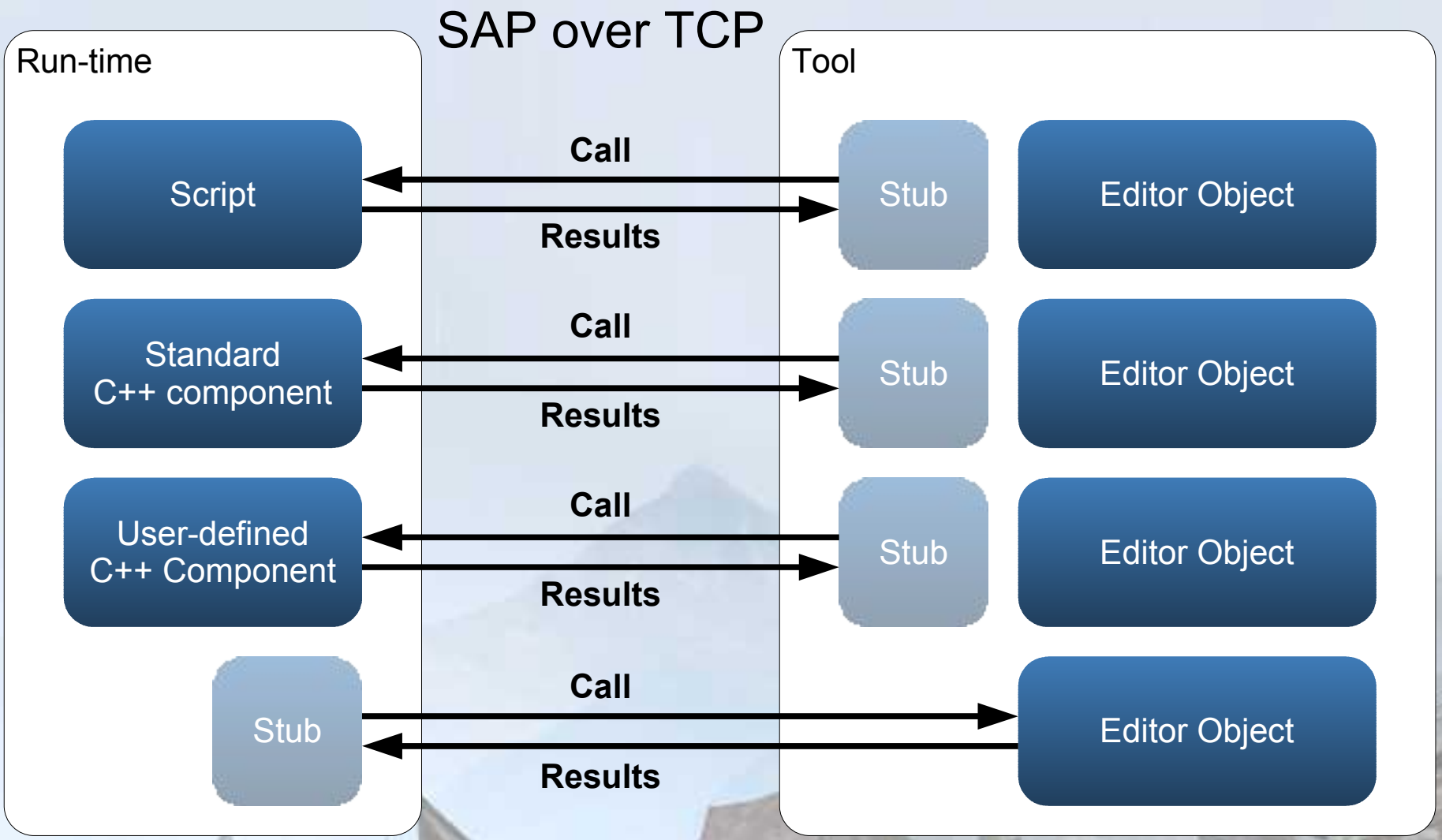
Live Editing Infrastructure



Simple Actor Protocol

- Generic network protocol for calling methods on high-level components
- Accessing the same features as game components and game scripts
- Transparent generic stub mechanism in Python
- Run-time components can call methods of tool objects

SAP Stub Objects in Python



Live Editing: Comparison Shark 3D to CryEngine 3

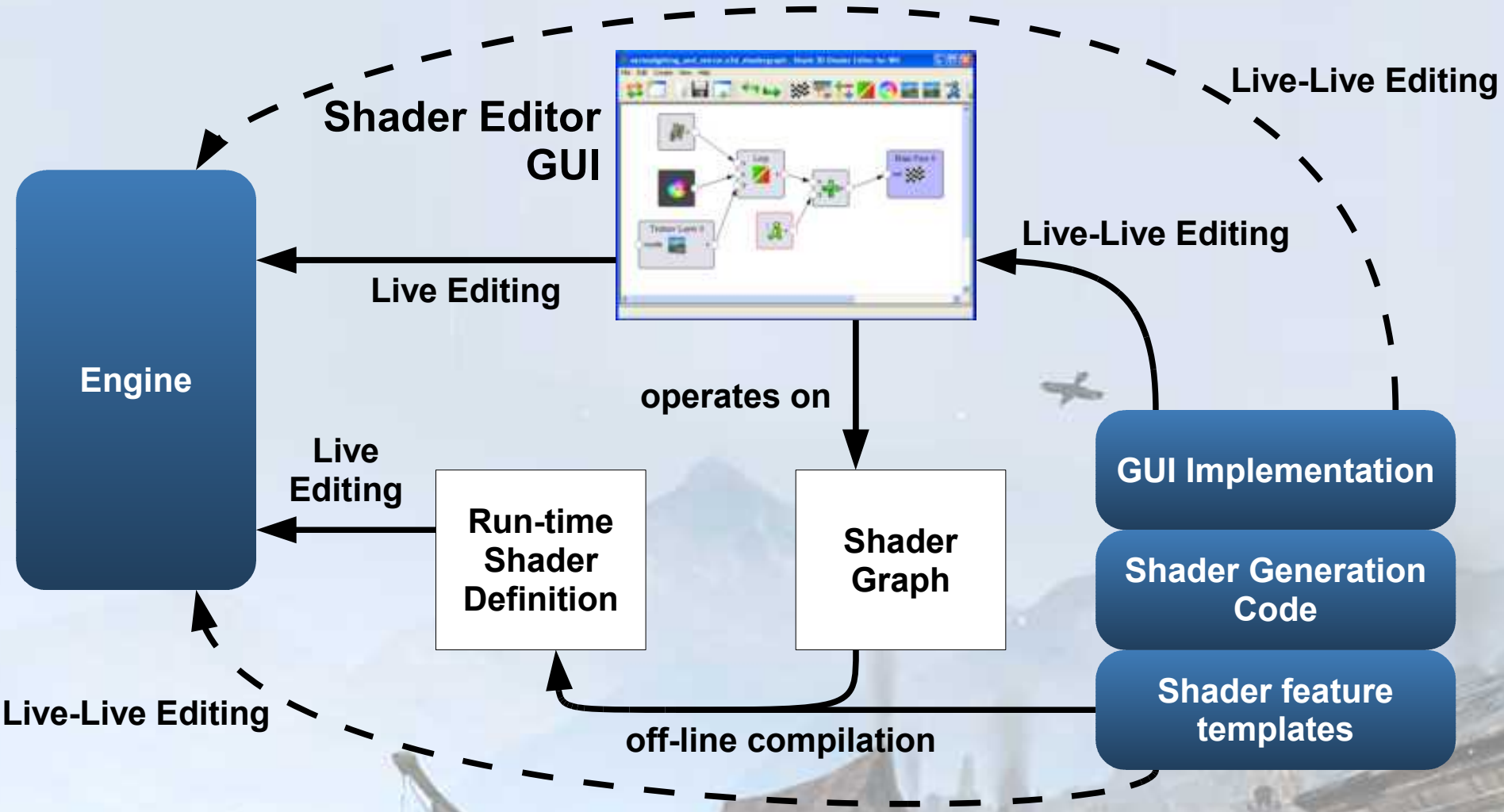
Shark 3D live editing for the user	**CryEngine 3** live editing for the user
→ WYSIWYG in the game editor	→ WYSIWYG in the game editor
→ Live update to consoles	→ Live update to consoles
→ Live update also from ext. tools	→ Within the level editor
→ Works with all resource types	→ Works for level editor features
→ Out-of-the-box & customizable	→ Focus on out-of-the-box solution

Shark 3D live editing technology	**CryEngine 3** live editing technology
→ Data flow / connection oriented	→ Data / database oriented
→ Duality: editing vs. run-time	→ Editing equals run-time
→ “Infrastructure follows features”	→ “Features follow infrastructure”
→ Persistence via any file format	→ Persistence via database

Live-Live Editing

- Editing your tool live without restarting the tool
- Eliminating turn-around cycles when customizing tool
- Live editing makes the designer happy.
  Live-live editing makes the tool programmer/scripter happy.
  → His/her results make the designer even more happy.
- We ourselves are heavily using live-live editing.

Live-Live Editing the Shark 3D Shader Editor



Thank you!

Shark 3D™

www.spinor.com