



# Using the Sun Grid Engine for feature visual effects work

experiences and conclusions at Rising Sun Pictures

Matthew Landauer

Sun Grid Engine Workshop 2007  
Regensburg, Germany





# About Rising Sun Pictures

- Australian visual effects company
- Focus entirely on producing visual effects for feature films
- Founded 10 years ago







# About Rising Sun Pictures

- Offices in Adelaide and Sydney
- One hundred people currently
- In the last five years the vast majority of our work has been for US clients







# Visual Effects

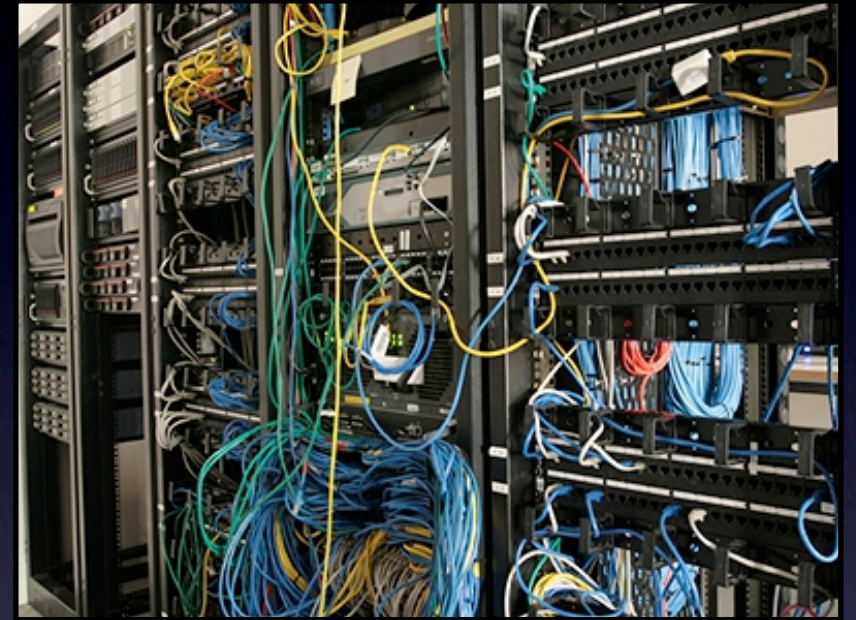
Seamlessly integrating photorealistic computer generated imagery into live action (real) footage





# Why do we use Grid Engine?

- We have a large renderfarm which we need to manage.
- Well established with a long history of use
- Extremely flexible and configurable
- Scales to very large number of machines
- Open source and free





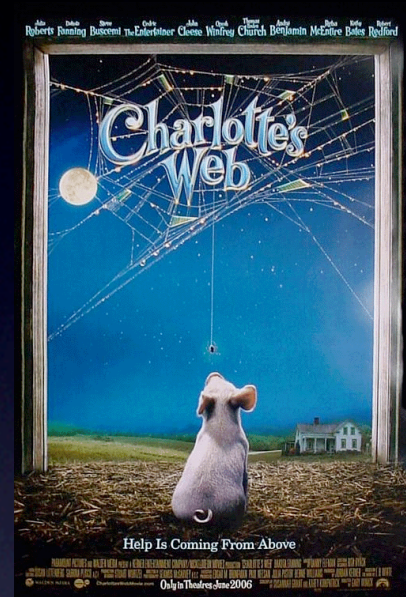
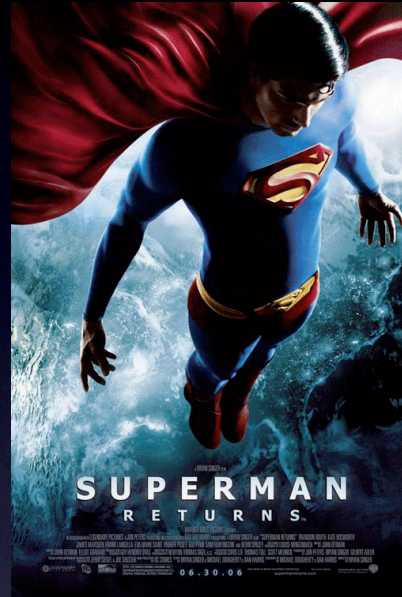


# Movies that used Grid Engine

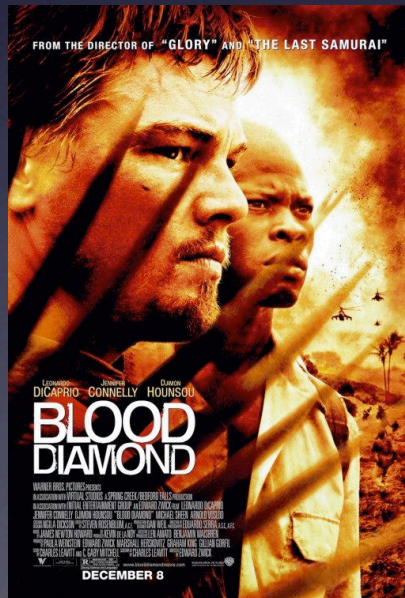




# Movies that used Grid Engine



Harry Potter  
and the Goblet of Fire  
Elephant Tales  
Superman Returns  
Charlotte's Web  
Blood Diamond  
The Last Mimzy  
28 Weeks Later



Harry Potter  
The Order of the Phoenix

Just finished:  
The Seeker

Currently in production:  
Australia  
The Ruins





[ RISING SUN PICTURES ]





# Example element breakdown



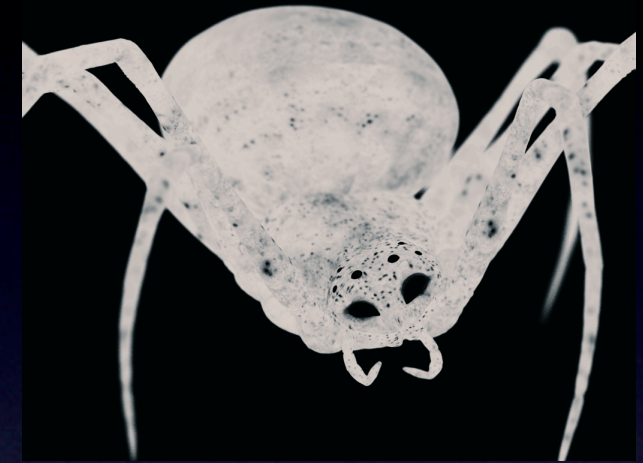
Ambient Occlusion



Hair



Diffuse



Freckle texture



Final Render





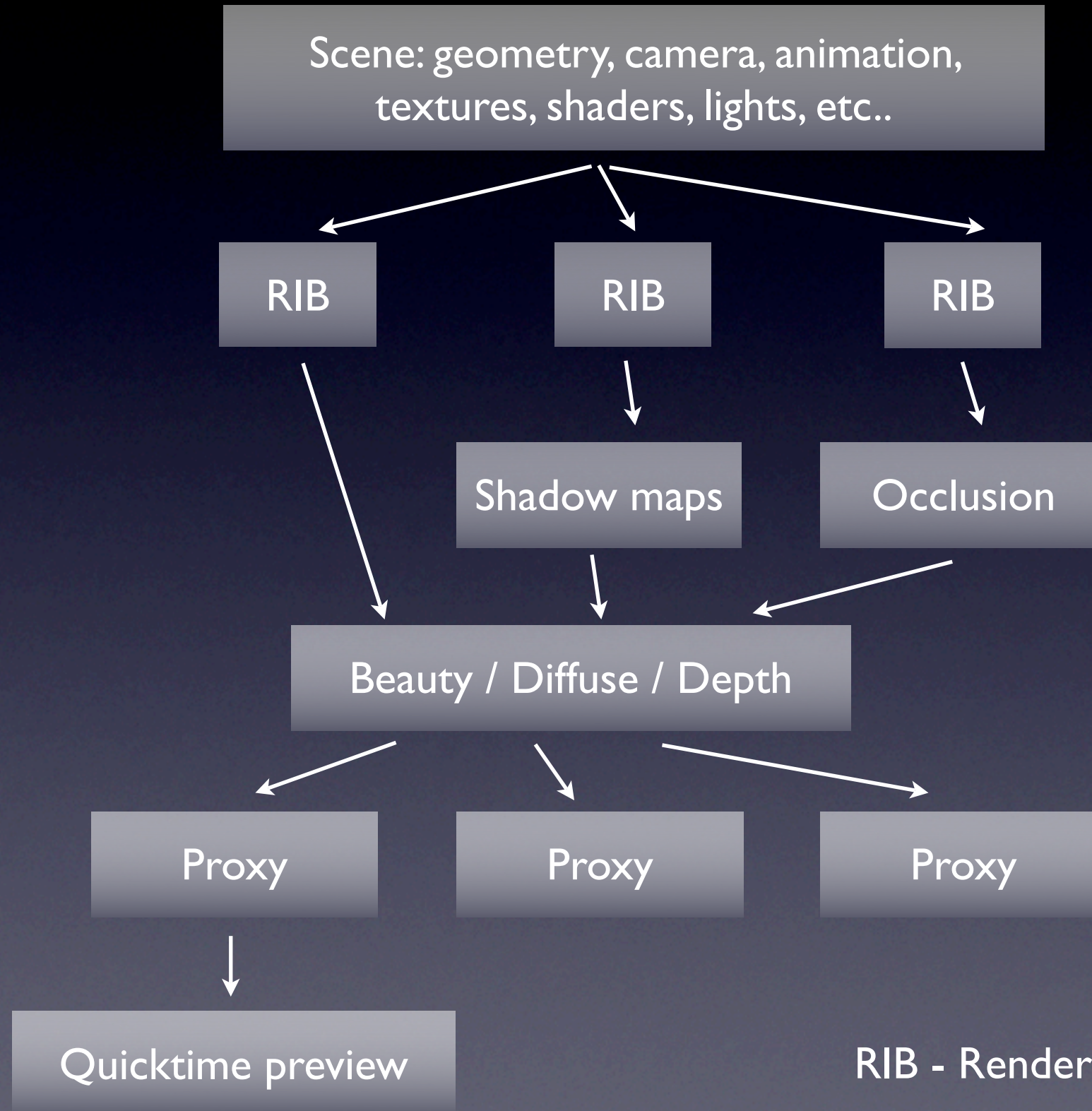
# Software

- 3d applications: Maya from Autodesk, XSI from Softimage, Houdini from Sidefx
- 3d renderers: 3Delight from DNASoft, Mental Ray from Mental Images
- Compositing: Shake from Apple
- Lens distortion: Hype from Visual Appliance
- And a whole host of proprietary stuff...





# A “typical” 3d render



RIB - Renderman Interface Binary





# Array jobs

- In visual effects and digital content creation we are typically repeating the same calculation for multiple time steps (frames)
- Array jobs are ideal for this
- Submission time for jobs is very low
- Memory footprint on qmaster is low





# Dependencies and array jobs

- Submit an array job A
- Submit an array job B which is dependent on job A (“held”)

Job B won't start until the whole array job A has finished.





# Dependency workarounds

## I. Ignore the problem

- Low throughput - it takes a long time to see a single rendered frame. This is a big problem for a creative driven process.





# Dependency workarounds

## 2. Do all the tasks for a single frame in a single job

- No simple visibility on the progress in Grid Engine.
- Doesn't work if we have to run some of the tasks on different machines.
- Reduced throughput when using “chunking” (explained later)





# Dependency workarounds

## 3. Break the array job into a set of array jobs

- Complex
- High submission times
- Multiple jobs in Grid Engine relating to a single high level task.





# Tools on top of Grid Engine

We have abstracted away this complexity from the user by creating our own front end to Grid Engine job submissions which we call “Job Engine”.





# Job Engine

- Job Engine is our front-end submission API which handles complex dependencies between multiple “tasks” as a single unit.
- A whole workflow is described in a single xml file.
- Job Engine is used by all our user-side tools





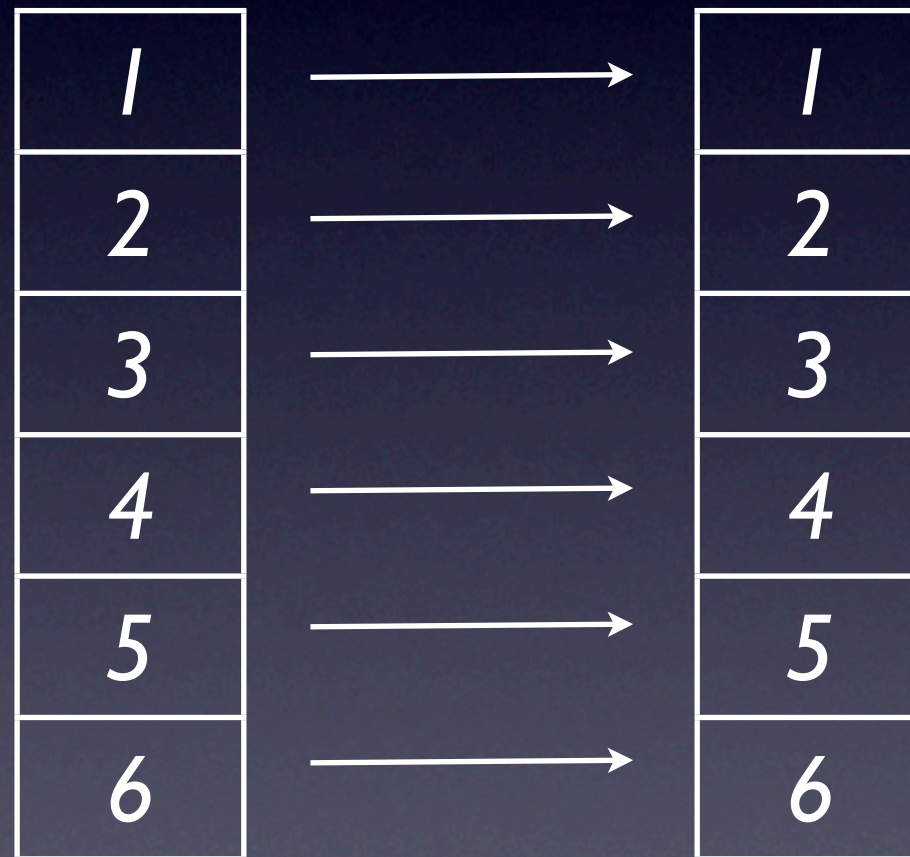
# Job Engine

- This has worked for us but is not ideal.
- The sensible next step: Improve the flexibility of dependencies between array jobs within Grid Engine itself.





# How we would like dependencies to work with array jobs

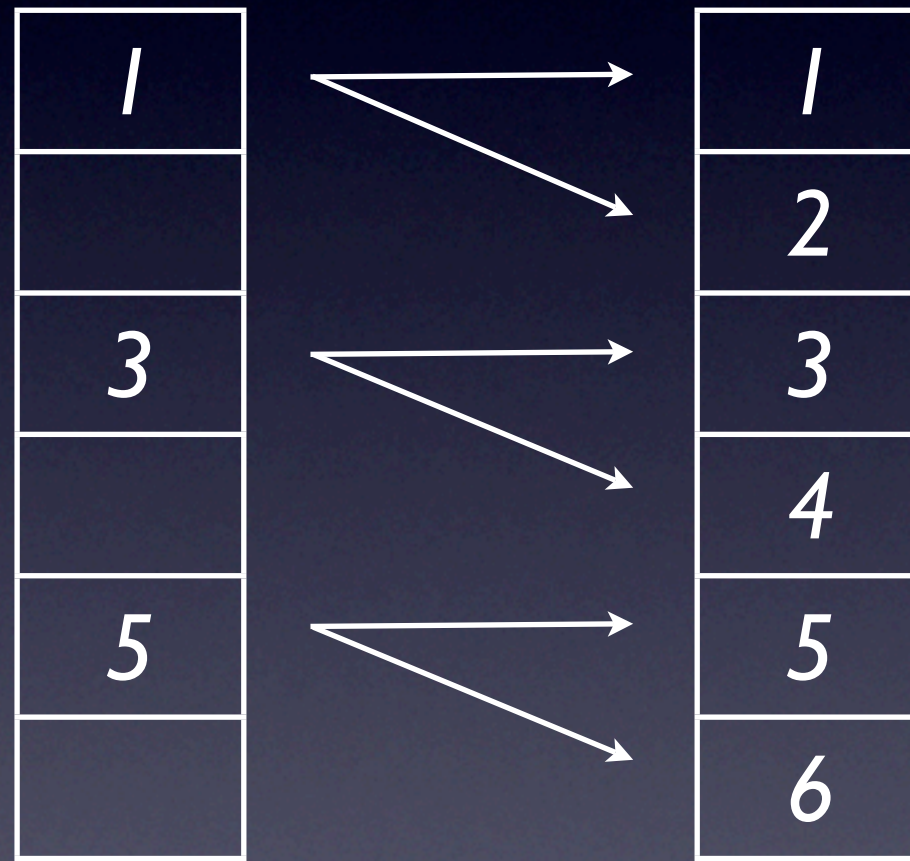






# Frame Chunking

In many cases it is more efficient to render several frames at once - we call this “chunking”.



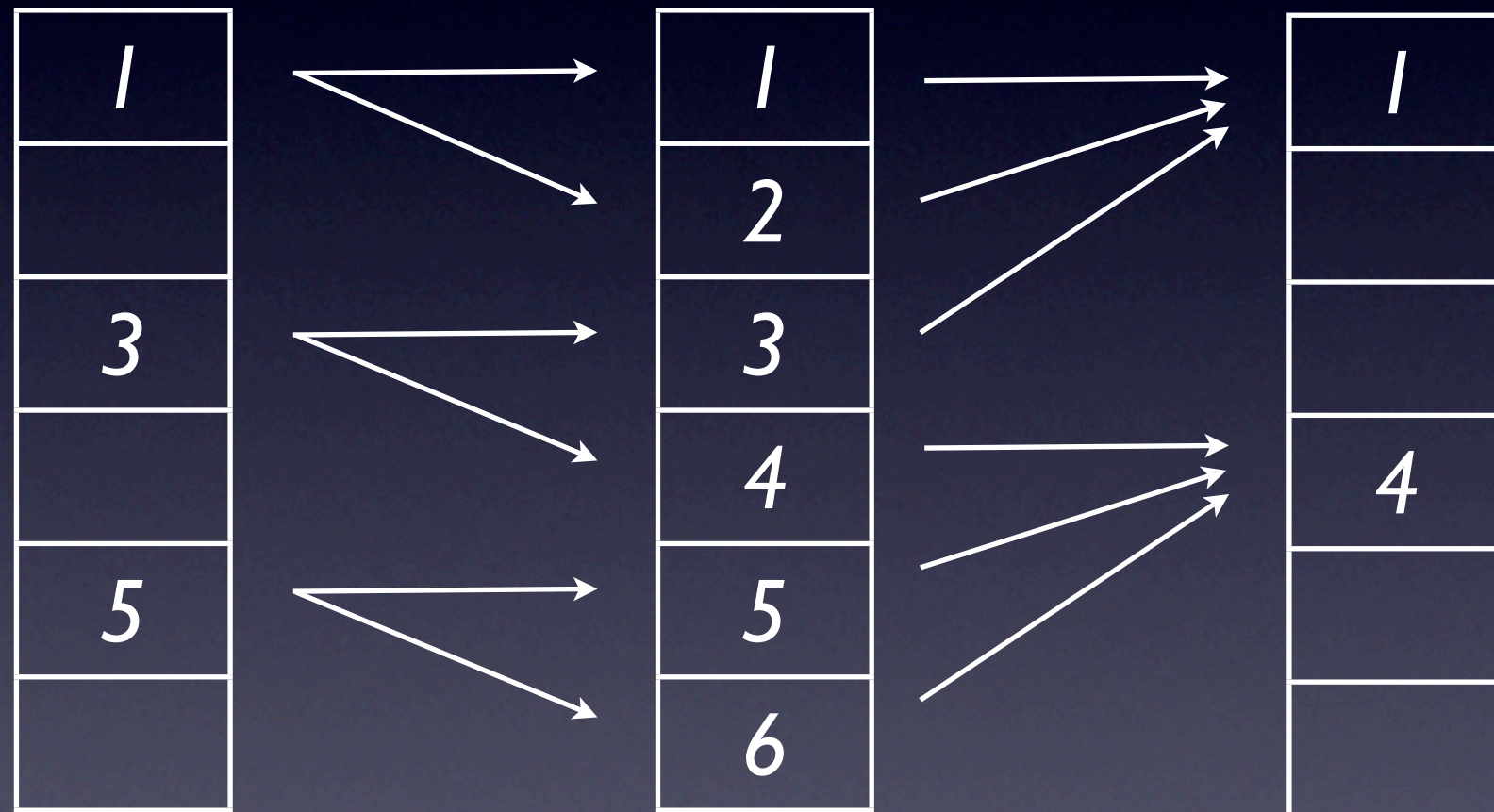
2 frames rendered  
each step





# Frame Chunking

We want to be able to mix and match chunking sizes




2 frames rendered  
each step

3 frames rendered  
each step





Powered by Black Minimalism Theme ★



Blog About Projects

## Grid Engine Array Task Dependency Specification

### Project Overview

#### Project aim

The aim of this project is to extend the Grid Engine job dependencies to support dependencies between sub-tasks in an array job. The aim is to target the most common use cases for digital content creation and digital visual effects.

The issues targeted with this project are:

- [gridengine issue 552: Array job interdependencies](#)
- [gridengine issue 1832: Support array sub job dependencies/holds](#)

Also, see related email threads:

- [Subject: Array dependencies](#)
- [Subject: Array sub job dependencies](#)

#### Expected benefit

The expected benefit of these changes is to make the grid engine easier to use for digital visual effects and digital content creation where typically "renders" consist of the processing of sets of images (frames in an image sequence) which is ideally carried out as an array task.

As an example:  
In a 3d render, several passes of renders are carried out in a specific order to produce a final rendered image. For instance, ribs need to be generated, followed by shadow maps, followed by the main render.

The time to submit renders of large numbers of frames will be reduced by at least an order of magnitude. The number of simultaneous running jobs on the grid engine will be increased. It will be easier to manage jobs as all frames will be "grouped" at the grid engine level. The speed of the grid in general should increase, including qstat when monitoring all jobs.

#### Functionality

The suggested changes in this document are designed to be fully backward compatible so with no change in usage there will be no change in behaviour.

#### The hold array dependency option

Add an option, when submitting an array job, where we specify a list of jobs that we make this job dependent on (in a similar way to `-hold_jid`) except we treat the dependency at the sub-task level rather than the job level. This is best explained through an example.

For example:

Currently, when an array task (B), consisting of three sub tasks is submitted and is dependent on another array task (A) consisting of three sub tasks as well:

```
$ qsub -t 1-3 A
$ qsub -hold_jid A -t 1-3 B
```

```
| A.1 | | B.1 |
| A.2 | --> | B.2 |
```

Read the specification at  
[http://open.rsp.com.au/?page\\_id=11](http://open.rsp.com.au/?page_id=11)



# Implementation of array dependency modifications

- Contributed by RSP back to the community under the “Sun Industry Standard Source License”
- Checked into a special branch on CVS on [gridengine.sunsource.net](http://gridengine.sunsource.net)
- V61u1\_ARI\_BRANCH branched from v6.1u1 release with bug fixes from v6.1u2 merged back.



# Implementation of array dependency modifications

- Courtesy binaries available for download today!
- Please test and provide feedback!





# Conclusions

- Grid Engine has worked well for us
- We hope our modifications will increase the adoption of Grid Engine in the visual effects and digital content creation industries





# Future work

Would love to see co-operation between visual effects companies to build an open source solid rendering infrastructure around Grid Engine that is artist friendly, geared to our needs and flexible.