ARUP

USD & IFC

Coworkers, cousins or siblings???

David de Koning

August 29, 2024

ARUP

Narrative

ARUP

USD Background

ARUP

USD Birthplace

Pixar, VFX

VFX industry

- *Creative* field
- Tight production *deadlines*
- Big *teams*
- Many different *disciplines*
- Lots of *data*

AECO industry

- *Creative* field
- Tight production *deadlines*
- Big *teams*
- Many different *disciplines*
- Lots of *data*

ARUP

USD's starting mandate

- The mandate for the USD project, initiated in 2012, was to marry the (recently redesigned and improved) ***composition*** engine and ***low-level data model*** from Presto with the lazy-access, time-sampled ***data model*** and lightweight ***scenegraph*** from TidScene.

- Composition – composing model data from multiple authors into a single data set
- Scenegraph – a method of organizing objects in a model / scene
- Data model

ARUP

Why use USD?

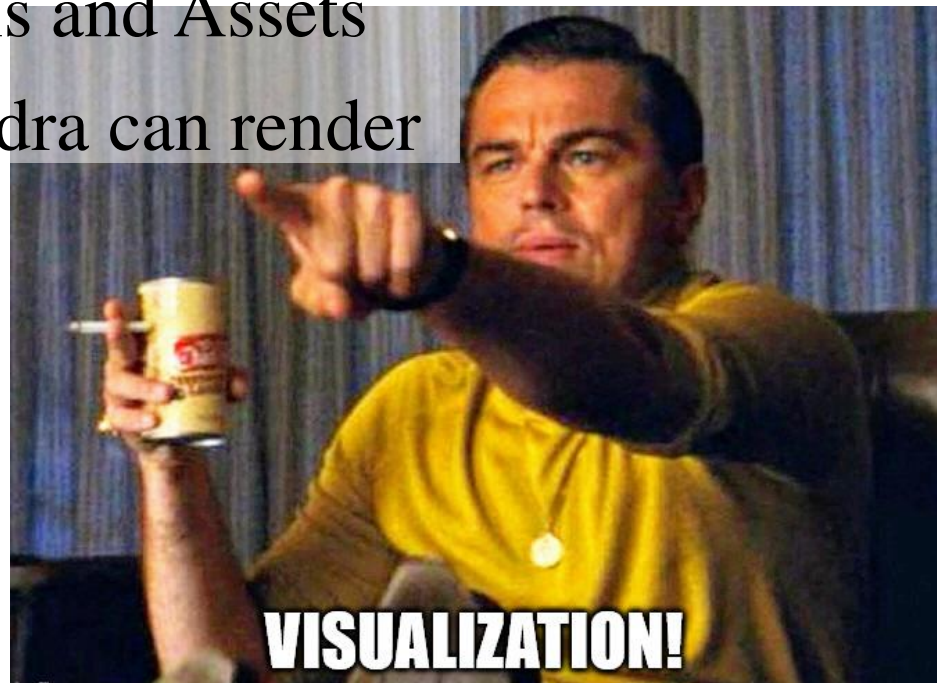
(in the VFX industry)

- Provide a rich, common language for defining, packaging, assembling, and editing 3D data, facilitating the use of *multiple digital content creation applications*.
- Allow *multiple artists to collaborate* on the same assets and scenes.
- *Maximize* artistic *iteration* by minimizing latency.

ARUP

What does USD claim to do?

- USD can compose and override
  - Across multiple authors
  - Across catalogues
- USD can be extended/customized
  - Asset Resolution
  - File Formats
  - Schemas
- USD can represent
  - Geometry
  - Shading
  - Models and Assets
- USD/Hydra can render



ARUP

Common aims

”time is money” and relying on a predictable toolset which promotes collaboration between departments in a non-destructive way, is fundamental to achieving feature film production deadlines. -Pixar

”time is money” and relying on a predictable toolset which promotes collaboration between *design and construction firms* in a non-destructive way, is fundamental to achieving *construction* deadlines.

ARUP

Recap

- USD exists to solve *data* and *teamwork* problems
- USD's core is a mechanism for non-destructive collaboration on structured data sets
- Geometry and visualization aspects of USD are built on top of the core
- USD is extensible to other domains

ARUP

Structure of a USD Stage

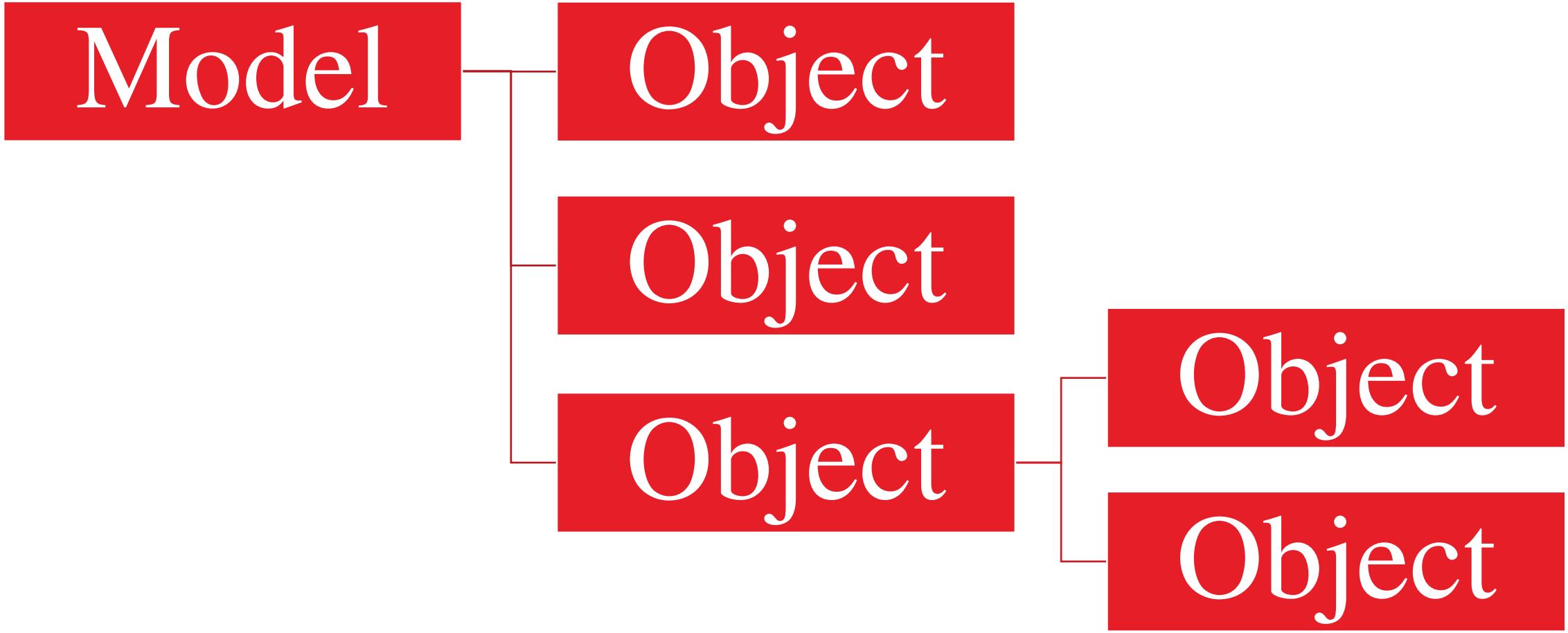
ARUP

Terminology

	IFC	ECS	USD
Model	Model / File	World ??	Stage
Object	Object	Entity	Prim

ARUP

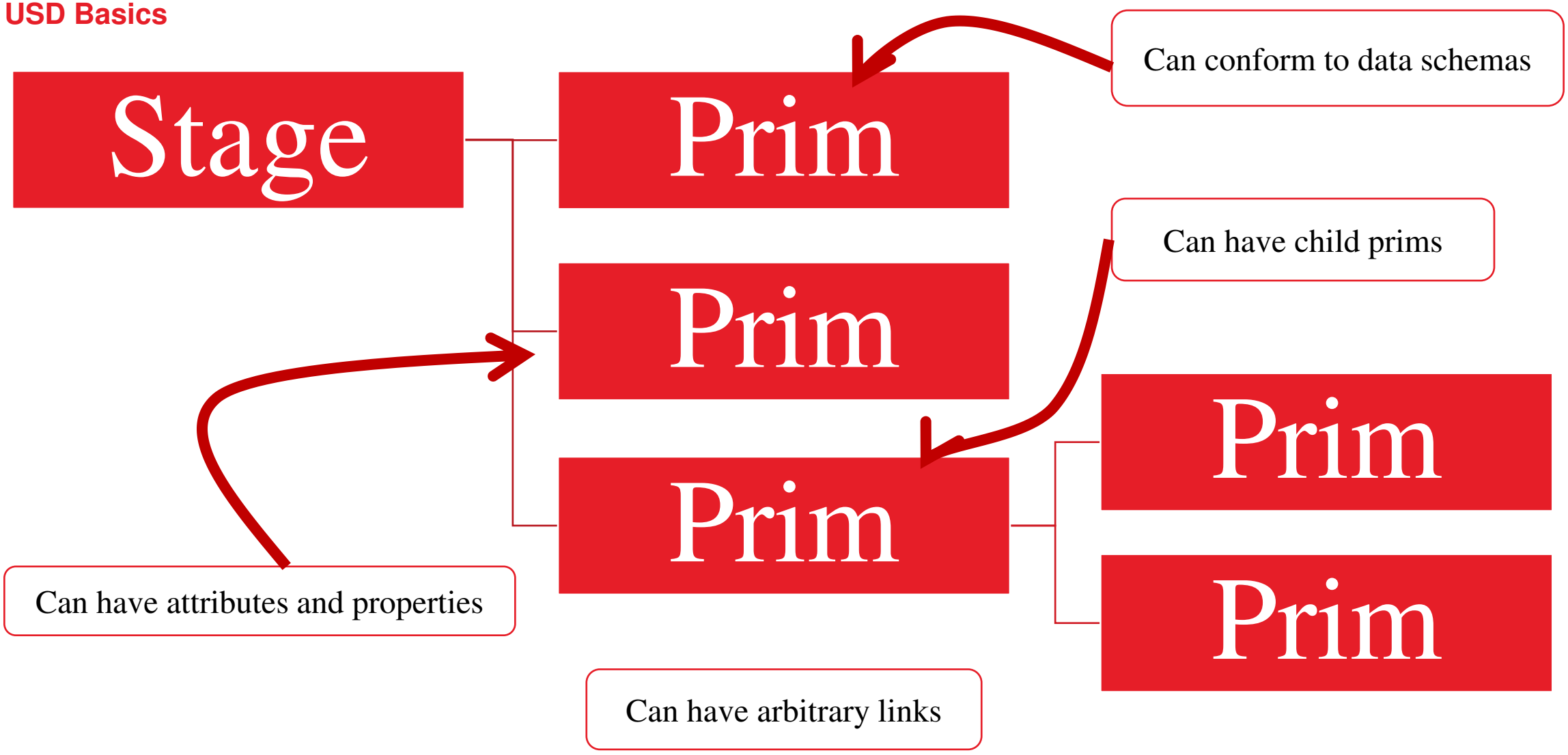
A model organized as a hierarchy of objects



ARUP

Reading / accessing USD data

USD Basics



ARUP

Parametric graphs vs. trees

Graph

- Set of nodes
- Related by edges
  - Directed or undirected
  - Arbitrarily many edges
- Computed values depend on arbitrary nodes (full graph traversal may be required)

Tree

- Set of nodes
- Related by directed edges
  - One parent
  - Multiple children
- Computed Values depend only on parents and children (strict subset)

ARUP

Semantic Parametric / Procedural Model

OpenExec – Pixar planning to open source

- Objects are defined semantically
- Geometry is computed / solved
- Semantic definition is a *directed graph* (not necessarily a tree)
- Parametric models have big advantages for authoring
- To render a parametric model you need to know:
  - The vocabulary
  - The meaning of the words
  - The algorithm to compute the geometry

ARUP

Explicit Geometry Scene Graph

USD

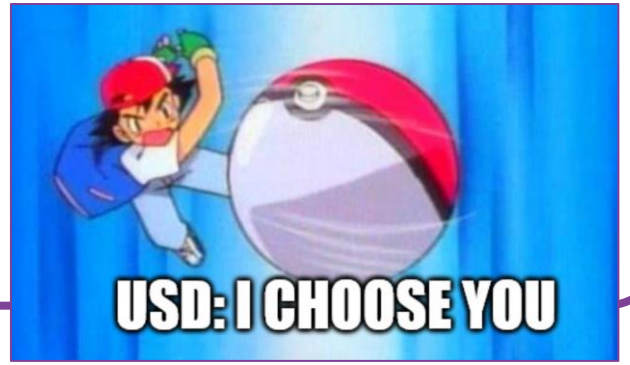
- All geometry is explicitly defined in cartesian space
- Single hierarchy for
  - Model Organization
  - Geometrical Parenting
- Stage is a strict tree (one parent per node)
- To render the model, you need to know
  - The geometry primitives
  - Parent Euclidian transforms

ARUP

Two model paradigms

Explicit Geometry

- Explicit geometry + Euclidian transforms
- Tree structure for straightforward scene traversal
- Semantics are optional overlay



"Parametric" Model

- Quick modelling of objects based on few parameters
- Changes propagate to dependent objects
- 'Smart' objects
- Scene rendering has extra step of solving geometry
- Semantic definition of model

ARUP

USD has taken a decision

And moved forward

https://openusd.org/release/glossary.html

Interchange is an important aspect of USD, and we are *not currently willing to tackle* the **conformance problem of different dataflow semantics** between different DCCs.

http://nickporcino.com/posts/last_mile_interchange.html

USD does not interchange **computational dataflow graphs** as found in programs such as Houdini and Maya.

ARUP

BUT…

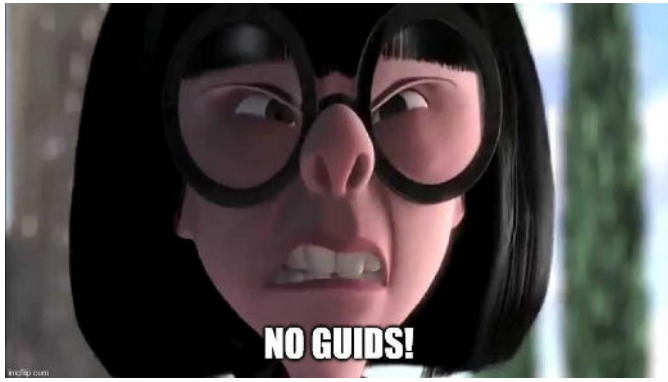
- USD can *represent* graphs
  - Relationship: a link from one Prim to another
  - Connection: a property value drawn from another Prim's property
- UsdShade Shading schema uses these. Shaders traverse the graph as part of the rendering process.

ARUP

Prim identifiers

- Prims are identified by their names, including parent prims
  - *e.g. Project/Site/Building/Storey/objectname*

"In past iterations of USD, Pixar used a form of GUID at the model/asset granularity, and after carefully weighing the pros and cons, we have decided that for us, the cost of occasional "namespace fix-up" operations run over a collection of assets is worth paying for the *ease of asset construction and aggregation*, and *readable text asset representations* that we get from namespace-paths as identifiers."



ARUP

Construction of a USD Stage

ARUP

Terminology

	IFC	ECS	USD
Model	Model / File	World ??	Stage
Object	Object	Entity	Prim
Group of Partial Objects	❌	‘Container’	Layer
Partial Object	❌	Component	PrimSpec

ARUP

ECS and IFC do the same thing

With a different frame of reference

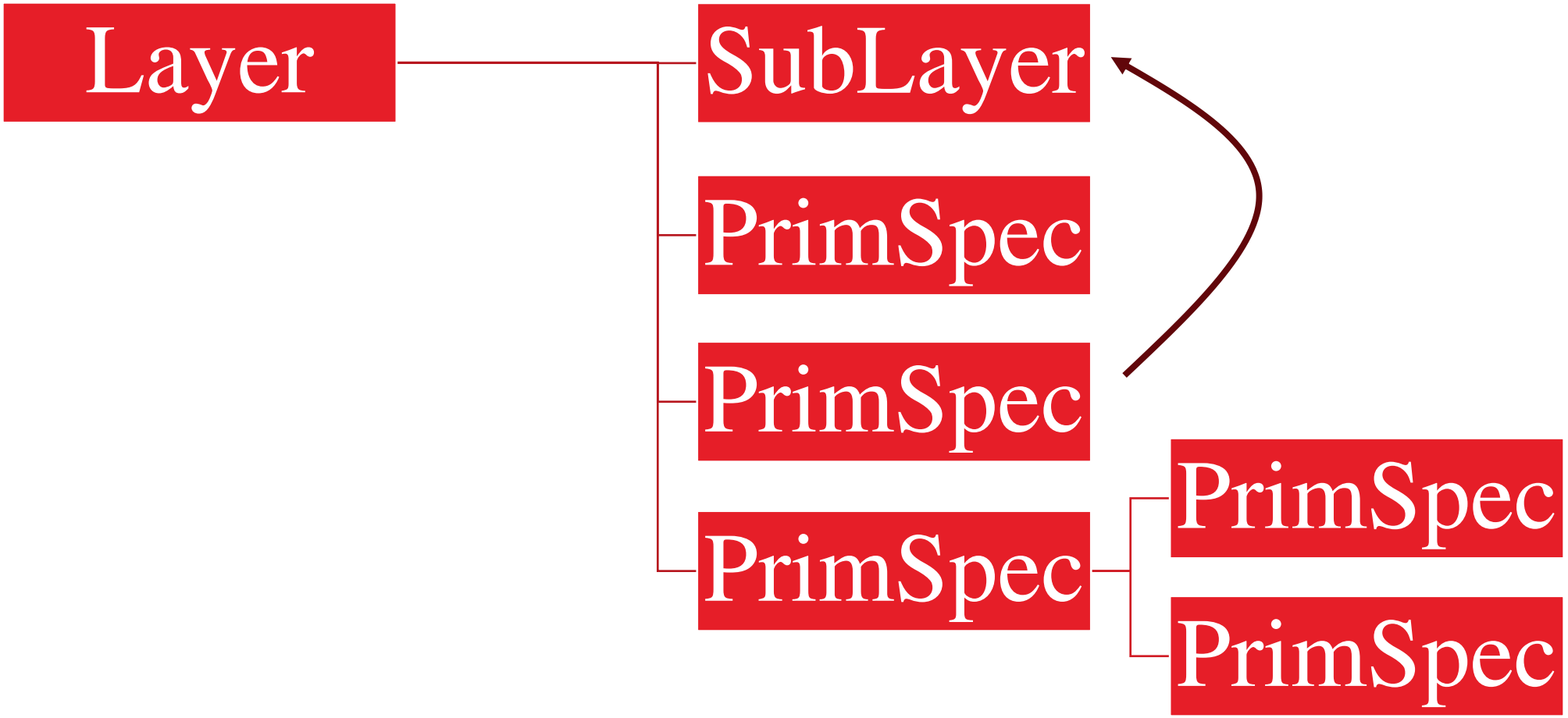
ECS bundles components

USD stacks layers

ARUP

Writing or sharing data

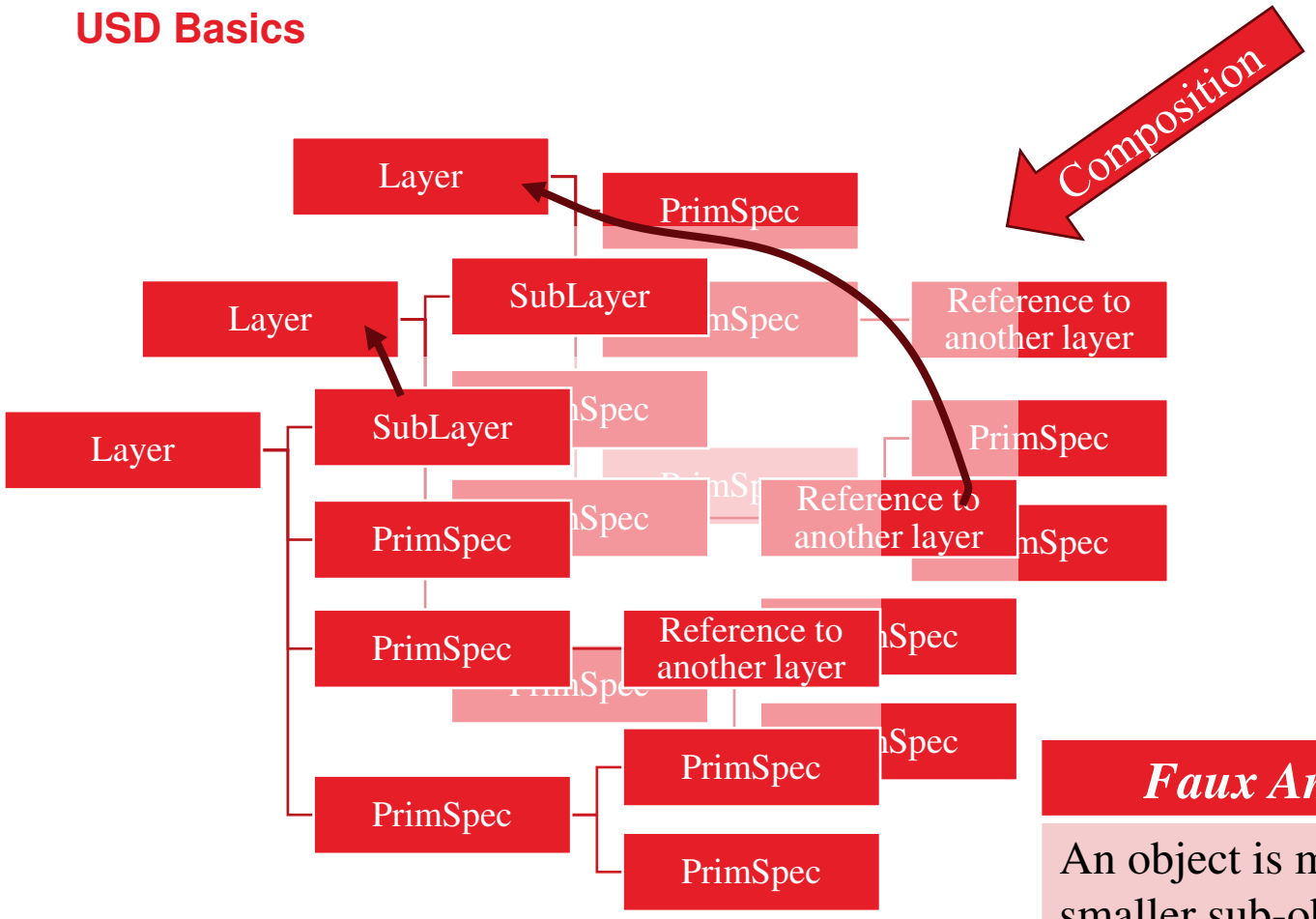
USD Basics



ARUP

Layers are composed into a stage

USD Basics



- Overlapping PrimSpecs define the final Prim
- Referenced Prims insert sub-trees

Faux Amis Alert!!	IFC	USD
An object is made up of smaller sub-objects	Composition	Model Hierarchy
Creating a model from multiple sources	Federation	Composition

ARUP

Specifiers

How to express information about a prim

- *def*: create and place a Prim on a Stage
- *over*: add information to a Prim that is def'ed elsewere
- *class*: define a Prim, but don't place it in the Stage

ARUP

Composition Arcs

- *subLayers*: include other layers in the composition process
- *references*: insert a Prim into the scenegraph
  - A `def` Prim can *reference* a `class` Prim
- *payload*, *inherits*, *specializes*: special cases of references
- *variantSet*: publish a discrete set of alternatives

Faux Amis Alert!!	**IFC**	**USD**
Placing a copy of a ‘catalogue object’ into a model	instance	reference
Telling the render pipeline that an object will not change and can be cached thoughout the pipeline	-	instance

ARUP

Example

- A *class* PrimSpec is defined in a library layer
- The library layer is referenced into a stage, nothing is placed
- A designer *def*s a PrimSpec that *references* the class from the library
  - This places the object into the stage

ARUP

USD Schema Domains

ARUP

Types of Schemas

- isA Schema: an OO class, an IFC4 entity (e.g. IfcBeam)
- API schema: an OO interface, an IFC4 Pset (e.g. PSet_ServiceLife)
  - Applied: part of the Prim's definition
  - Non-Applied: used only by someone reading the stage

ARUP

Controlling the data model - schemas

USD Basics

- Schemas define:
  - Names of properties
  - Default values
  - Some behaviour (optional)
- Grouped into *Schema domains*
  - UsdGeom, UsdShade, UsdRi, UsdPhysics

ARUP

Validation of a USD Stage

ARUP

Controlling the data model - validators

- “USD is very flexible and every site/studio … using USD can establish their own rules and patterns”
- Validation Framework
  - Mechanism to validate Prims, Stages and Layers
- Validation rules are specified as tests

ARUP

USD's organization

ARUP

USD's Organization

- Developed at Pixar, open sourced in 2016
- Alliance for Open USD (2023) is working towards ISO certification
- AECO Interest Group (hi Angel and Sean!!)

ARUP

Recap

ARUP

Recap

- USD provides
  - A robust mechanism for composing/federating a model from multiple authors, with sub-object granularity.
  - A performant, open-source core for working with massive models.
  - An extensible framework for defining specific data schemas / organizations.

ARUP

IFC 5 objectives

- Multiple authors can contribute to a shared data set
- Geometry and data is stored unambiguously
- Ease of implementing readers that can read all geometry and data
- Reference third party data into an IFC data set
- Modular data schemas (that allows validation against multiple data schemas separately)
- Ability to integrate data sets with “IDS 5” and “BCF 5”
- Express libraries of objects that can be placed in a data set
- Add properties to library objects on a per-instance, per-project or wider context
- Facilitate performant implementations
- Provide basis for partial, atomic transactions
- Express arbitrary relationships between objects
- Express the difference between two data sets
- Support audit trails for QA processes
- Open to extensions with new country-, client-, firm- and project-specific data schemas
- Ability to validate data set against IFC 5 schema

ARUP