New AAF Examples

Barcelona Developers Conference

12 Nov 2001

Jim Trainor, AAF Association

Goals

- Improve quality of example code.
- Create the files with the following object types:
  - Metadata
    - Mobs with TaggedValues and KLVData.
  - Essence
    - Audio/Video data access.
  - Composition
    - Two track audio/video composition with transition.

Goals

- Cleanly build one example program on the results of previous.
- Well structured, clean implementation, as might be found in a “real” application.
  - Modern C++
- Provide an example of “good use”.

Goals

- Identify opportunities for reuse. Isolate this code in a separate library.
  - Reusable code: an investment
  - Nonresusable code: an expense

Goals

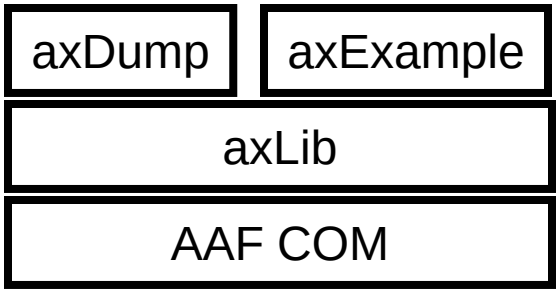
- YAD – Yet Another Dump program
  - Expose the object model structure.
  - Example of generic example of file input.
  - Reuseable components as a first step toward a graphical AAF file browser.

- Useful tool to study the evolution of the file structure as the content becomes increasingly complex.

What Exists Today

- Prefix "**Ax**" – **A**AF E**x**amples
- One program with three distinct modules:
  - axExample –metadata
  - axExample –essence
  - axExample –composition
- Dump program: axDump
- Reusable library: axLib

What Exists Today



- axDump and axExample access the AAF COM interface (nearly) exclusively via axLib
- axLib = A high level interface.

From Santa Fe DevCon

What We're Still Working On

- Types need for extensions
- Codecs (read and write essence)
- Object Manager support for multiple containers
- Testing/packaging
- Semantic validator
- Completing documentation
- **More high-level interfaces**

Slide Ref: "State of The AAF", AAF Santa Fe Dev. Con, Jeffery Bedell (Bold emphasis added).

Generated AAF Files

- Example code generates file with:
  - 4 Master Mobs
    - Each with TaggedValue objects.
    - Each with KLVData objects.
  - 4 Essence Data
    - 2 Audio
    - 2 Video
  - Composition
    - Video and Audio tracks, each with a transition.

Generated AAF Files

- Create empty file:

  axExample -file ax.aaf

- Dump objects only.
- Only Header and ContentStorage objects are present in the file.
- Dictionary also exists, but is not shown.

Level	Desc.	Detail
1	Object	Header
4	Object	ContentStorage

Generated AAF Files

- Add meta data

axExample -file ax.aaf -metadata

- Four MasterMobs
- Each has two Editor comments (TaggedValue).
- Each has two KLVData objects.

```
Level   Desc.      Detail
1       Object     Header
4       Object         ContentStorage
8       Object             MasterMob
12      Object                 TaggedValue
12      Object                 TaggedValue
12      Object                 KLVData
12      Object                 KLVData
8       Object             MasterMob
12      Object                 TaggedValue
12      Object                 TaggedValue
12      Object                 KLVData
12      Object                 KLVData
8       Object             MasterMob
12      Object                 TaggedValue
12      Object                 TaggedValue
12      Object                 KLVData
12      Object                 KLVData
8       Object             MasterMob
12      Object                 TaggedValue
12      Object                 TaggedValue
12      Object                 KLVData
12      Object                 KLVData
```

Generated AAF Files

- The essence examples add audio and video essence:

  axExample -file ax.aaf -metadata -essence

- Note the appearance of TimelineMobSlot, SourceClip, EssenceDescriptor, and SourceMob objects in the dump.
- These objects are created by a call to IAAFMasterMob::CreateEssence()

SDK - New Examples

Level	Desc.	Detail
1	Object	Header
4	Object	ContentStorage
8	Object	MasterMob
12	Object	TimelineMobSlot
15	Object	SourceClip
12	Object	TaggedValue
12	Object	TaggedValue
12	Object	KLVData
12	Object	KLVData

-
-
-

8	Object	SourceMob
12	Object	TimelineMobSlot
15	Object	SourceClip
11	Object	WAVEDescriptor
8	Object	SourceMob
12	Object	TimelineMobSlot
15	Object	SourceClip
11	Object	WAVEDescriptor
8	Object	SourceMob
12	Object	TimelineMobSlot
15	Object	SourceClip
11	Object	CDCIDescriptor
8	Object	SourceMob
12	Object	TimelineMobSlot
15	Object	SourceClip
11	Object	CDCIDescriptor
8	Object	EssenceData
8	Object	EssenceData
8	Object	EssenceData
8	Object	EssenceData

SDK - New Examples

- The composition example creates an audio/video sequence with a transition. The composition references the existing MasterMobs.

  axExample -file ax.aaf -metadata -essence -composition

- Note the appearance of CompositionMob, Sequence, Transition, and OperationGroup objects in the file.

SDK - New Examples

```
Level  Desc.    Detail
1      Object   Header
4      Object       ContentStorage
8      Object           MasterMob
12     Object               TimelineMobSlot
15     Object                   SourceClip
12     Object               TaggedValue
12     Object               TaggedValue
12     Object               KLVData
12     Object               KLVData
                        .
                        .
                        .
8      Object           SourceMob
12     Object               TimelineMobSlot
15     Object                   SourceClip
11     Object               WAVEDescriptor
                        .
                        .
                        .
8      Object           SourceMob
12     Object               TimelineMobSlot
15     Object                   SourceClip
11     Object               CDCIDescriptor
```

```
8      Object   CompositionMob
12     Object       TimelineMobSlot
15     Object           Sequence
19     Object               SourceClip
19     Object               Transition
22     Object                   OperationGroup
19     Object               SourceClip
12     Object       TimelineMobSlot
15     Object           Sequence
19     Object               SourceClip
19     Object               Transition
22     Object                   OperationGroup
19     Object               SourceClip
8      Object   EssenceData
8      Object   EssenceData
8      Object   EssenceData
8      Object   EssenceData
```

COM Observations

- COM suites its purpose – a binary interface standard.

BUT

COM Observations

- COM works fine in C and C++ programs. Even Visual Basic!

BUT

COM Observations

- We could all continue using the plain vanilla AAF COM interface…

BUT

COM Observations

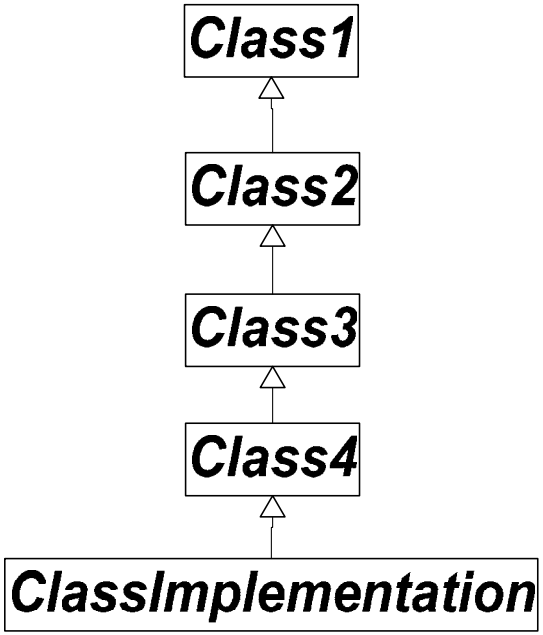
- If you plan on using only C++ to build applications…

WHY?

…. stick with vanilla COM.

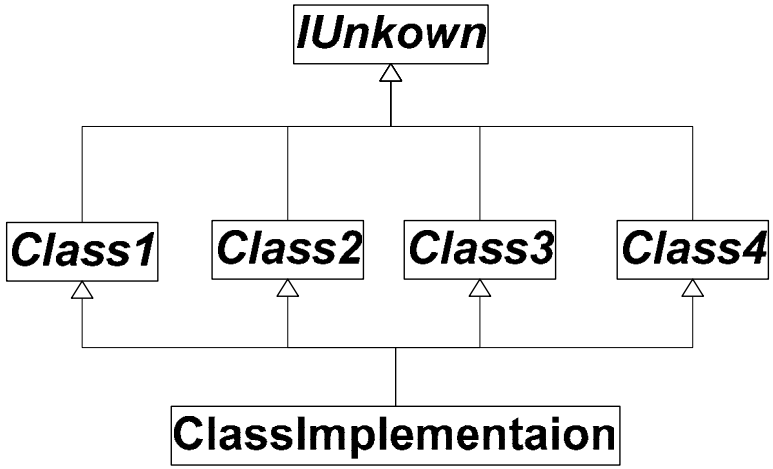
COM Observations

- In the AAF SDK, and in general, COM takes an object model like this:



COM Observations

- … and turns it into this:



- Applications only “see” Inknown and Class[1,2,3,4]. Never the entire interface!

COM Observations

- To Be Fair:
  - COM, and multiple inheritance, make sense when you don't know *a priori* what interfaces an object might implement.

- But, in the AAF COM interface we know what interfaces are implemented by every COM object.

COM Observations

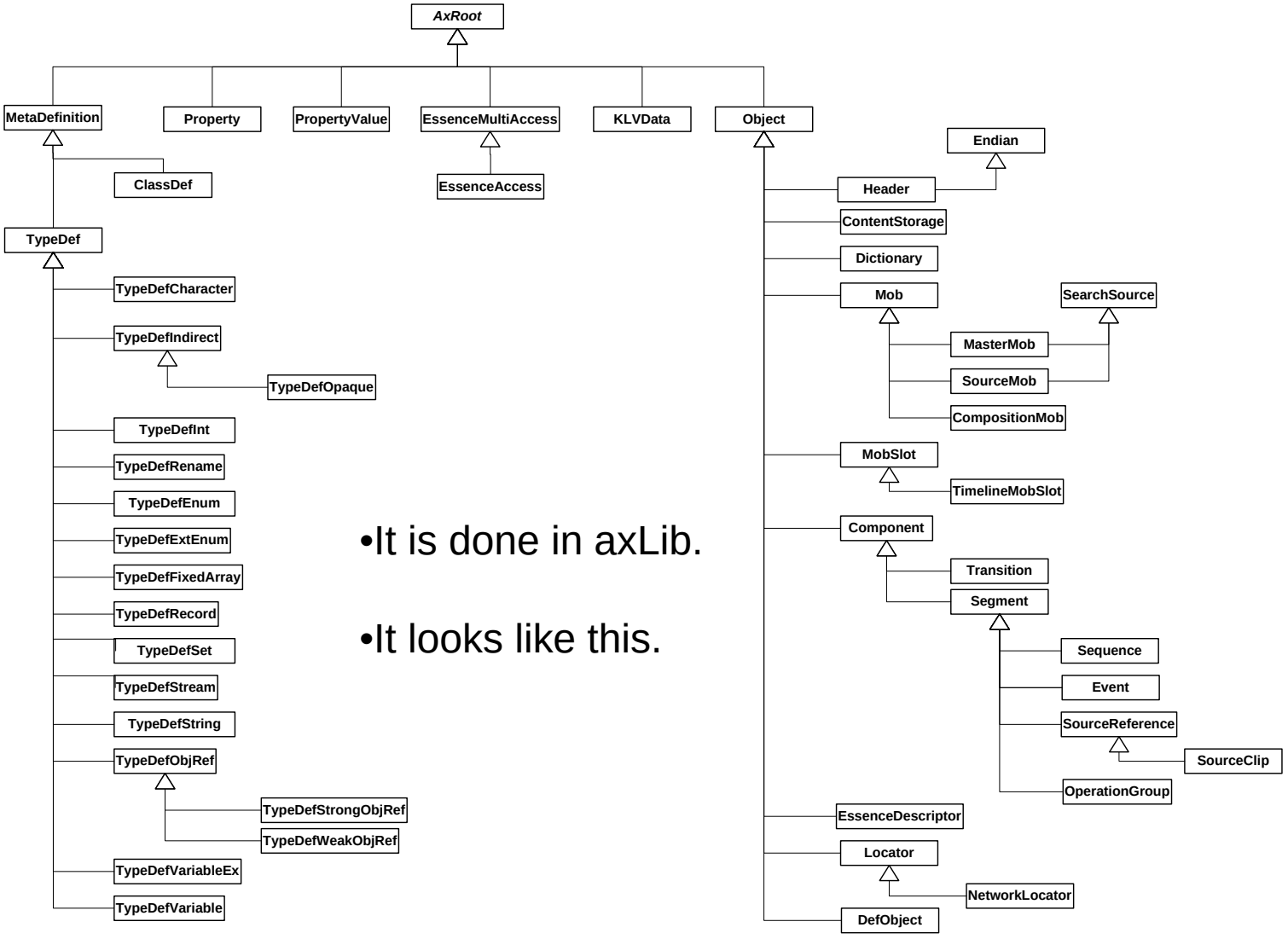
- Interface implementations are documented in AAF.idl:

```
//
// Objects that implement IAAFSourceClip also implement the following interfaces:
//  - IAAFSourceReference
//  - IAAFSegment
//  - IAAFComponent
//  - IAAFObject
//
```

COM Observations

- The COM interface implementations use a “normal” single inheritance graph.
- It is possible to “untangle” the COM interface and recover a “normal” single inheritance graph.
- Two exceptions to the single inheritance model: IAAFEndian, and IAAFSearchSource.

COM Observations



- It is done in axLib.
- It looks like this.

COM Wrappers

- We can:
  - Recover interface inheritance.
  - Emulate polymorphic behavior using implicit type cast operators.
  - Eliminate many QueryInterface() calls!

COM Wrappers

- All wrappers store a single data value: a AAF smart pointer to the underlying COM object.
- Their constructors always take a single argument: an AAF smart pointer.
- Other objects pointers are always returned as: AAF smart pointers.
- Result: Loosely coupled. Coupled only by inheritance.

COM Wrappers

- Also, implements a status check/exception throw for every COM call.
- Throws an axLib class that provides:
  - The file and line number.
  - The HRESULT value.
  - The HRESULT value mapped to a string.

COM Wrappers

- Most wrapper class interfaces match the underlying COM interface nearly exactly.
- Major Difference:
  - Return values not set via a pointer – they're returned by value:

```
hr = obj->GetValue ( &result );
        becomes
result = axObj.GetValue()
```

COM Wrappers

- Interfaces are changed only where it makes code safer.
- Example, returning string values:

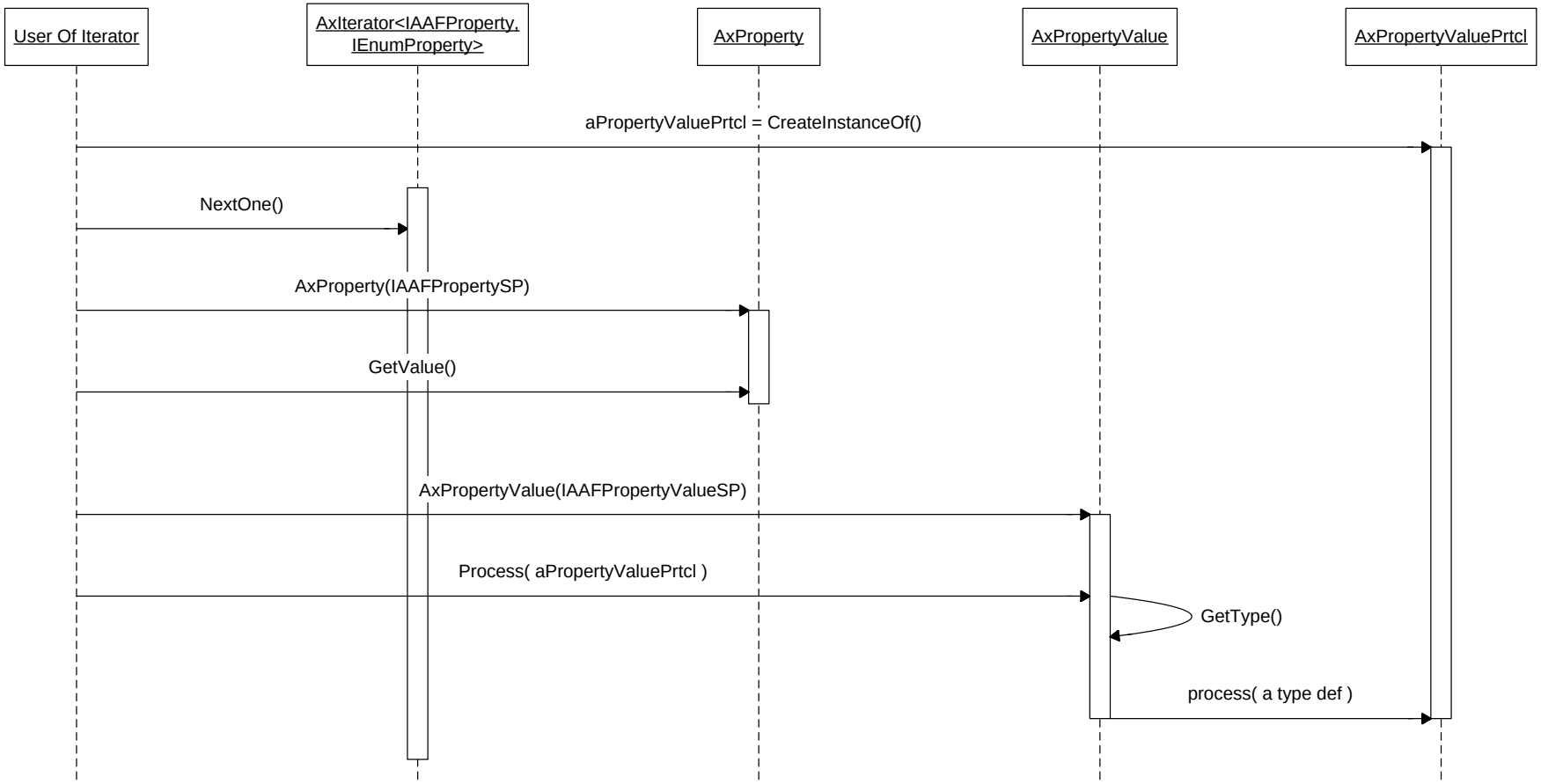
```
Obj->GetSize( &size );
buf = new wchar_t [size]
Obj->GetName( buf, size );
delete [] buf;

becomes

AxString name = wrappedObj.GetName();
```

COM Wrappers

- Wrappers extend the COM interface when common patterns of use justify it.
- Example: Property value processing.
- All code that processes property values must implement a type lookup. Factor this out! Make it reusable.

COM Wrappers

Call Sequence: Property Value Processing



Iterators

- Just another wrapper.
- Most basic iterator simply wraps IEnum COM interface objects.
- Can be implemented using a template because all IEnum's COM interfaces vary only by type.
- Study the axLib code. (See AxIterator<>)

COM Observations

- The problem with QueryInterface:
  - It is not type safe.
  - Compilers cannot enforce valuable C++ type safety features.
- User must deal with Yet Another Type System – Interface IIDs
- There are too many IDs in AAF programs!

COM Observations

- C++ compilers see no problem with this:

```
IAAFMasterMob* masterMob = CreateMasterMob();
int* mob;
masterMob->QueryInterface( IID_AAFMOB,
                           reinterpret_cast<void**>(&mob) );
```

- Or this:

```
IAAFSourceClip* sourceClip = CreateSourceClip();
IAAFComponent* component;
masterMob->QueryInterface( IID_AAFSegment,
                           reinterpret_cast<void**>(&component) );
```

COM Observations

- Some clever templates can make QueryInterface() type safe. Example:

```
IAAFMasterMob* masterMob;
IAAFMob* mob;
hr = masterMob->QueryInterface( IID_AAFMOB,
                                reinterpret_cast<void**>(&mob).
```

becomes

```
IAAFMasterMobSP masterMob;
IAAFMobSP mob;
AxQueryInterface( masterMob, mob );
```