Collabora Productivity

Image Handling Rework

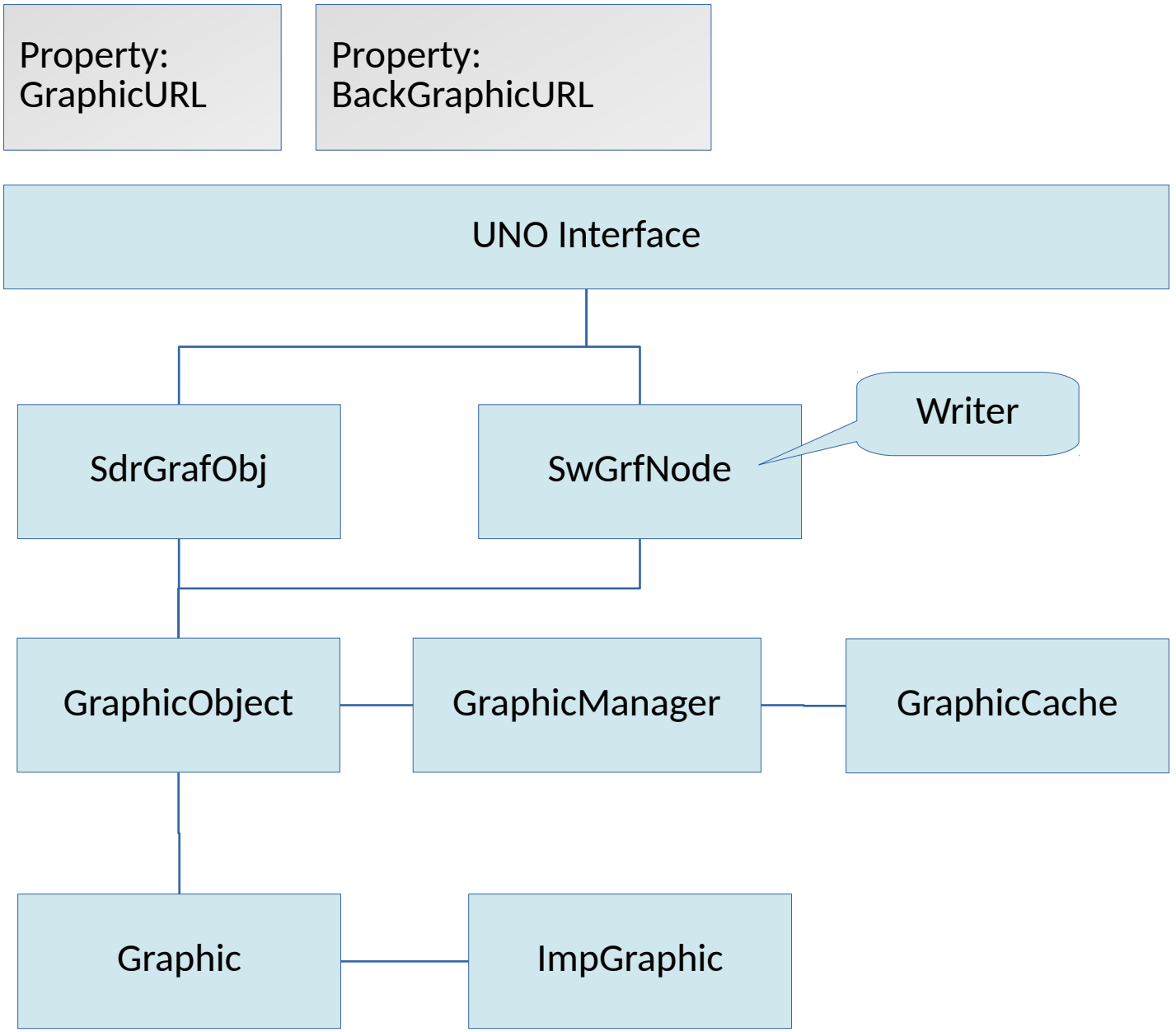
By Tomaž Vajngerl

Many thanks to TDF to make this work possible.

Problematic image life-cycle

- UNO API uses properties (typically GraphicURL) which takes a string attribute to internal (or external) identifier of graphic

  “vnd.sun.star.GraphicObject:<HEX ID>” (GraphicObject URL)
- GraphicCache objects stores and manages the ID
- GraphicObject can “recreate” a Graphic (XGraphic) from the ID


Property:
GraphicURL
Property:
BackGraphicURL
UNO Interface
SdrGrafObj
SwGrfNode
Writer
GraphicObject
GraphicManager
GraphicCache
Graphic
ImpGraphic

Problematic image life-cycle - Problems

- No way to know if a Graphic is used or not.
  - We have a GraphicObject URL - does the Graphic still exist?
  - We have a Graphic – is it still being used?
- This is a bad life-cycle which is calling for a disaster to happen.
- Can cause loss of images – it did happen before.

Life-cycle solution

- Remove all uses of GraphicObject string URL and replace it with XGraphic – reference counted object
- References are counted so we know when it is used and when it is not used any more
- More exact accounting of memory

The rework

Done in two phases:

- Phase 1 – change all usages of GraphicURL (and similar) property to use XGraphic
- Phase 2 – rework Graphic memory handling
- Phase 3 – On demand loading, various fixes

The rework – Phase 1

- Big chunk of work
  - Every GraphicObject URL needs to be changed to XGraphic
  - Most important usages in import/export filters
- Needs UNO API change
- Risk of something breaking – want to avoid that

The rework – API changes

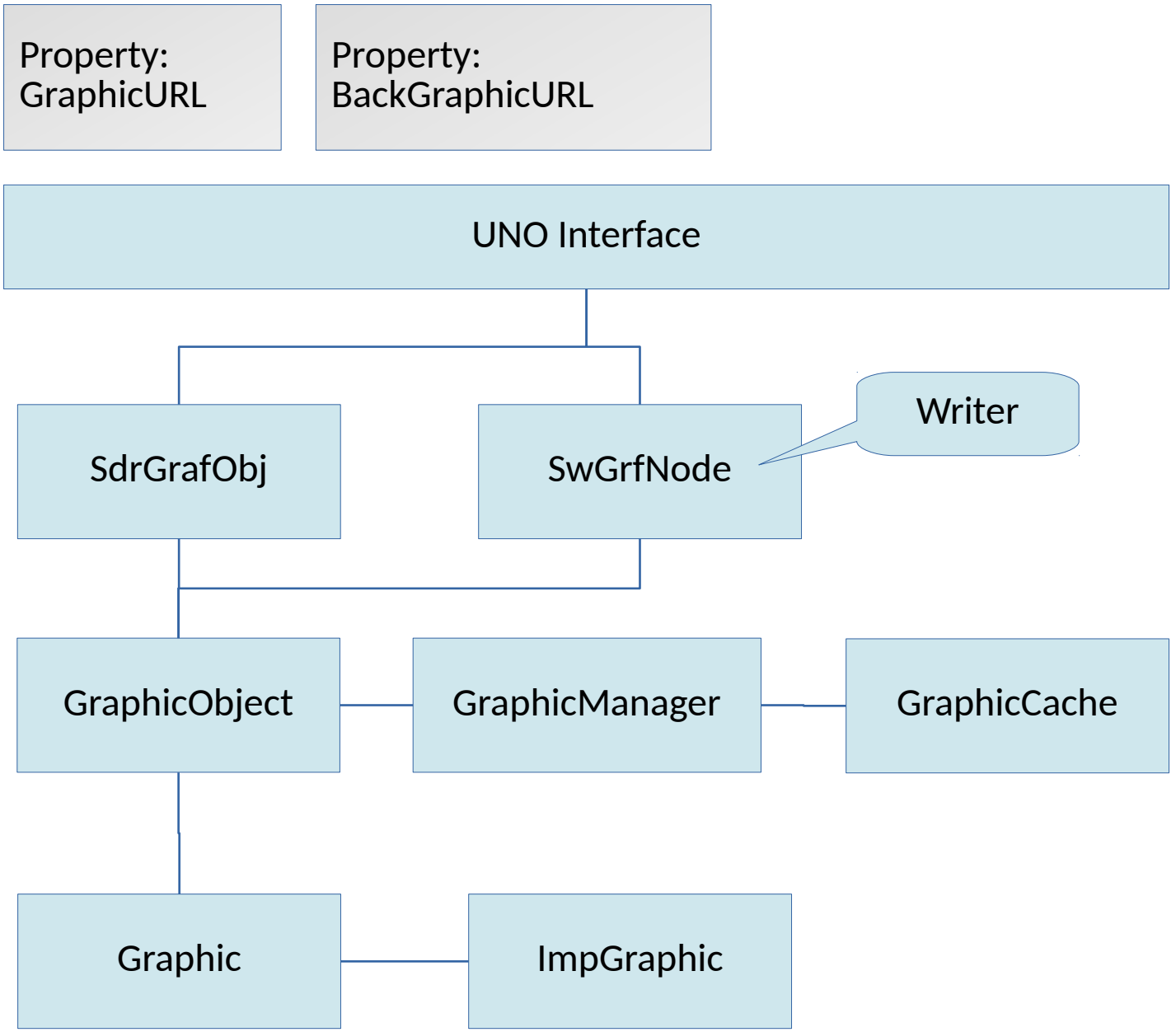
- GraphicURL -> Graphic (type XGraphic) and GraphicBitmap (type XBitmap) for bullets
- BackGraphicURL -> BackGraphic (type XGraphic)
- HeaderBackGraphicURL -> HeaderBackGraphic (type XGraphic)
- FooterBackGraphicURL -> FooterBackGraphic (type XGraphic)
- ParaBackGraphicURL -> ParaBackGraphic (type XGraphic)
- ThumbnailURL -> Thumbnail (type XGraphic)
- ReplacementGraphicURL -> ReplacementGraphic (type XGraphic)
- FillBitmapURL -> FillBitmap (type XBitmap)

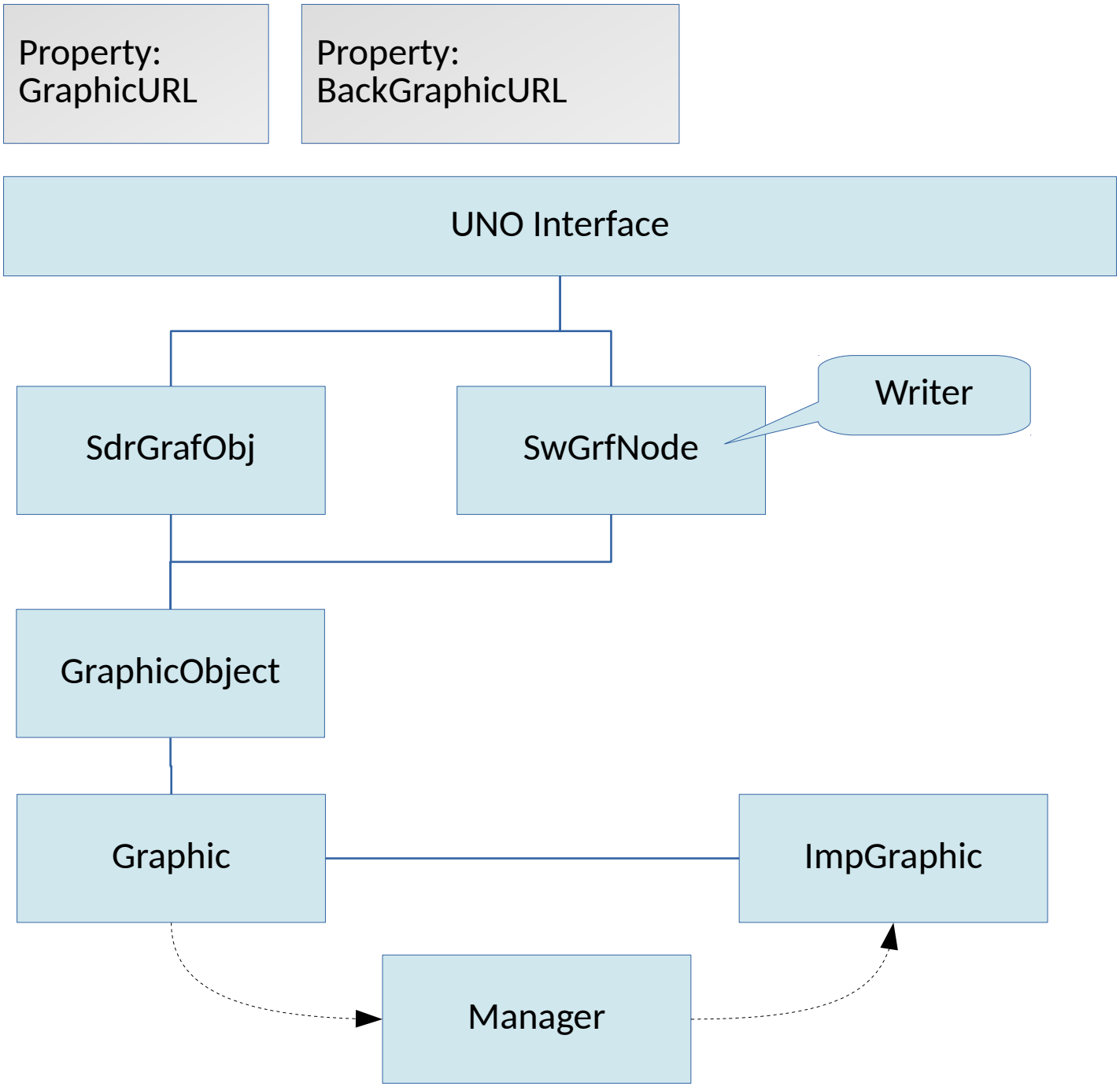
The rework – API changes

- For backwards compatibility previous properties (GraphicURL,...) are still working for setting.
- Only possible for “external” URL only.
- Getting is not possible any more - throws an exception.
  - When setting previous property, it loads and sets the corresponding new property.

The rework – Phase 2

- Remove most of responsibilities from GraphicObject and move it to Graphic
  - Swapping only inside Graphic, not outside
- Encapsulate Graphic
  - Don’t expose how images are handled internally
  - Makes the life-cycle much easier to control


Property:
GraphicURL
Property:
BackGraphicURL
UNO Interface
SdrGrafObj
SwGrfNode
Writer
GraphicObject
GraphicManager
GraphicCache
Graphic
ImpGraphic


Property:
GraphicURL
Property:
BackGraphicURL
UNO Interface
SdrGrafObj
SwGrfNode
Writer
GraphicObject
Graphic
ImpGraphic
Manager

The rework – Phase 2 (Cont.)

- Memory management - Swapping
  - Use a simple algorithm first, extend later
  - Prevent unneeded swapping (swap loop)
  - Allow to exceed memory limits, to prevent swap cycling
  - In future – implement more aggressive multi level algorithm

The rework – Phase 3

- On demand image loading
  - Read only image metadata (width, height) and don’t load the image into memory
  - Swap the image immediately to disk
  - Load the image only when needed
  - Previously implemented for ODF already, now also for other fiters (OOXML)
- UNO tests for Writer
  - Move from JAVA to C++

Future improvements

- Multitude of possibilities open because of Graphic encapsulation
- Improvements to GraphicFilter (loading of images)
- Farter graphic rendering connected with smarter swapping
  - Large images are usually scaled down
  - Use original size image swapped
  - Create half sized, quarter sized images which use less memory and are rendered faster

Collabora Productivity

Thank you for listening!

More details on https://tomazvajngerl.blogspot.com/

By Tomaž Vajngerl