# CLEANING UP OOo MULTI-THREADING

**Kay Ramme**
Senior Technical Architect
StarOffice/OpenOffice.org UDK Project Lead
Sun Microsystems

# Agenda

- Current Use of Multi-Threading …
- Opportunities for Improvement …
- In an Ideal World …
- Proposed Solution …
- Current Implementations …
- The Plan …
- Q&A …
- May be some Deep Diving …

# Current Use of Multi-Threading

- Basically single threaded, some dedicated threads:
    - > Windows clipboard
    - > Windows drag&drop
    - > VOS timer
    - > UCB-helper background loader
    - > Acceptor thread(s)
    - > Configuration flasher
    - > ICE thread
    - > ...
- Uno request threads.
- Everything implemented thread-safe.

# Opportunities for Improvement
**(see http://wiki.services.openoffice.org/wiki/Analysis/Multi-Threading for details)**

- Does not scale with multiple clients, CPUs, …

- No documentation regarding threading-model or -architecture.

- It is not (really) thread-safe.

- Fragile / no systematic approach to thread-safeness.

- Every developer has to take care of multi-threading.

- Hard to implement OLE/COM based components:

4

# Opportunities for Improvement (continued)

- Subtle dependencies against the "main-thread",
  - > because VCL being thread-affine.
- Performance penalties because of much locking etc. (e.g. Interlock counters).
- Increased code size and complexity because of multi-thread constraints (locking).
- Long lasting (slow) operations blocking the GUI (partly addressed with polling).
- Competing threading models, Uno <-> VCL

# In an Ideal World

**(see http://wiki.services.openoffice.org/wiki/Architecture/Goals_for_OOo_Threading-Model%26-Architecture for details)**

- Always Responsive GUI ...
- Scales with multiple clients, CPUs, ...
- Systematic approach to concurrency ...
- Simple to implement and use ...
- Exactly one threading-model
- Good Documentation ...

# Always Responsive GUI

- GUI is soft real time.
- Long lasting (slow) operations, e.g.
  - > loading,
  - > printing,
  - > saving, respectively
  - > I/O in general

  need to be offloaded.
- Want to use dedicated threads for this.
- Need to ensure scalable I/O (UCB).

# Scales with Multiple Clients, CPUs

- Scalability basically is about parallelism.
- OOo could scale on a ...
  - > application level – documents of different applications can be manipulated in parallel,
  - > document level – multiple documents can be manipulated in parallel,
  - > window level – every window can process events in parallel,
  - > ...
- Need to identify scaling sensitive code.

# Systematic Approach to Concurrency

**(See http://wiki.services.openoffice.org/wiki/Uno/Binary/Spec/Threading-Architecture for details)**

- Automatic External locking.
- Only few thread-aware code.
- Only well tested thread-aware code.
- Support for encapsulating thread-affinity.
- Defined scalability.

# Simple to Implement and Use

- In clients and services code.

- Be conservative, only require thread related programming where actually necessary.

- No surprise (thread-transparent):
  - > No call back by another thread.
  - > No asynchronous call backs.
  - > Every activity is triggered by the client.

- Code can just marked to be either
  - > thread-safe,
  - > thread-unsafe, or

# Documentation

- Have specifications.
- Have implementation Descriptions.
- Have Tutorials / Best Practices.
- Publicly provide implementation status.
- Document everything in the wiki.

# Proposed Solution

**(See http://wiki.services.openoffice.org/wiki/Uno/Binary/Spec/Threading-Architecture for details)**

- Drop VCL threading-model.

- Extend Unos threading-model .

- Switch all code to be thread-unsafe, except scaling sensitive parts (UCB, Config Manager).

- Fix thread-affinity.

- Introduce I/O threads.

- Enhance scalability step-wise, as needed only.

# Current State

- Proof of Concept in CWS UTF2.

- Asynchronous Dialogs ~80% (Intel, CH2000, Sun).

- Uno threading-model extension nearly ready - 90%.

- Thread-Affinity fix is on the way – about 80%.

- Switch to thread-unsafe is ongoing - about 85%.

- Introducing I/O threads – open.

- Enhance scalability – open.

13

# The Plan

- Finish&Integrate new threading-model / -architecture.

- Remove outdated thread related constructs.

- Introduce I/O threads.

- And finally, switch to an event driven architecture ...

# Questions & Answers

# Deep Diving

- Outlook ...

- How Uno is going to support thread related code ...

- Making VCL Thread-Transparent ...

- Switching OOo to thread-unsafe ...

- History ...

# Outlook

- Use a running office process not only for GUI, but also for other services (3$^{rd}$ party integrations).

- This would be a push for more scalability / parallelism.

# Unos Extended Threading-Model

- Background
  - > Environments – to manage objects of same OBI (and purposes)
  - > Mappings – to map from one environment to another
  - > Objects – the actual functionality
- Concrete
  - > Map thread-unsafe objects to become thread-safe
  - > Map thread-affine objects to become thread-safe
- Tutorials

18

# Unos Extended Threading-Model

- Purpose Environments: "<OBI>[:purpose]*"

- Environment Stacking

- Cascaded Mappings
  > "<OBI>[:purpose]*" <-> "<OBI>[:purpose]*"

- Two new, thread related purposes:
  > ":unsafe"
  > ":affine"

- Bootstrapping support

# Make VCL Thread-Transparent
**(see http://wiki.services.openoffice.org/wiki/Effort/Make_VCL_Thread-Transparent for details)**

- Problem:
  - > VCL inherits Windows thread-affinity.
  - > VCL provides the Solar-Mutex.
  - > The solar mutex becomes released wrongly, in some situations. Fixing this introduces regressions because of "Dialog::execute".
  - > DDE depends on the "main" thread ...

- Tasks:
  - > Encapsulate thread-affinity by using a dedicated thread.
  - > Remove the Solar-Mutex.
  - > Replace "Dialog::execute" where necessary

# Switching OOo to Thread-Unsafe

- Find Uno components a mark them as thread-unsafe.

- Find threads, make them use Unos extending threading-model.

- Take a look at the libraries / private APIs, mark them as thread-unsafe.

- FIND all EXCEPTIONS.

# History

- ~1997: "Horst" introduced multi-threading.
- ~1998: "Horst" and Markus introduced the beloved Solar-Mutex.
- 2000: Markus asked me briefly, to spend some thoughts on this and to (just) solve it.
- 2002: I heard the same from Jörg (Heilig).
- 2002: Kai (Sommerfeld) and I started our journey to finally solve this.
- 2005: Intel / Dhananjay (Keskar) jumped

# Some Links

- http://wiki.services.openoffice.org/wiki/Arc
- …/wiki/Uno
- …/wiki/Effort/Revise_OOo_Multi-Threading
- …/wiki/Effort/Make_VCL_Thread-Transparent
- …/wiki/Effort/Make_Dialogs_Asynchronous
- …/wiki/Effort/Encapsulate_the_Win32_thread_affinity
- …/wiki/Spec/Threading-Model
- …/wiki/Spec/Threading-Architecture

# CLEANING UP OOo MULTI-THREADING

**Kay Ramme**
Kay.Ramme@sun.com