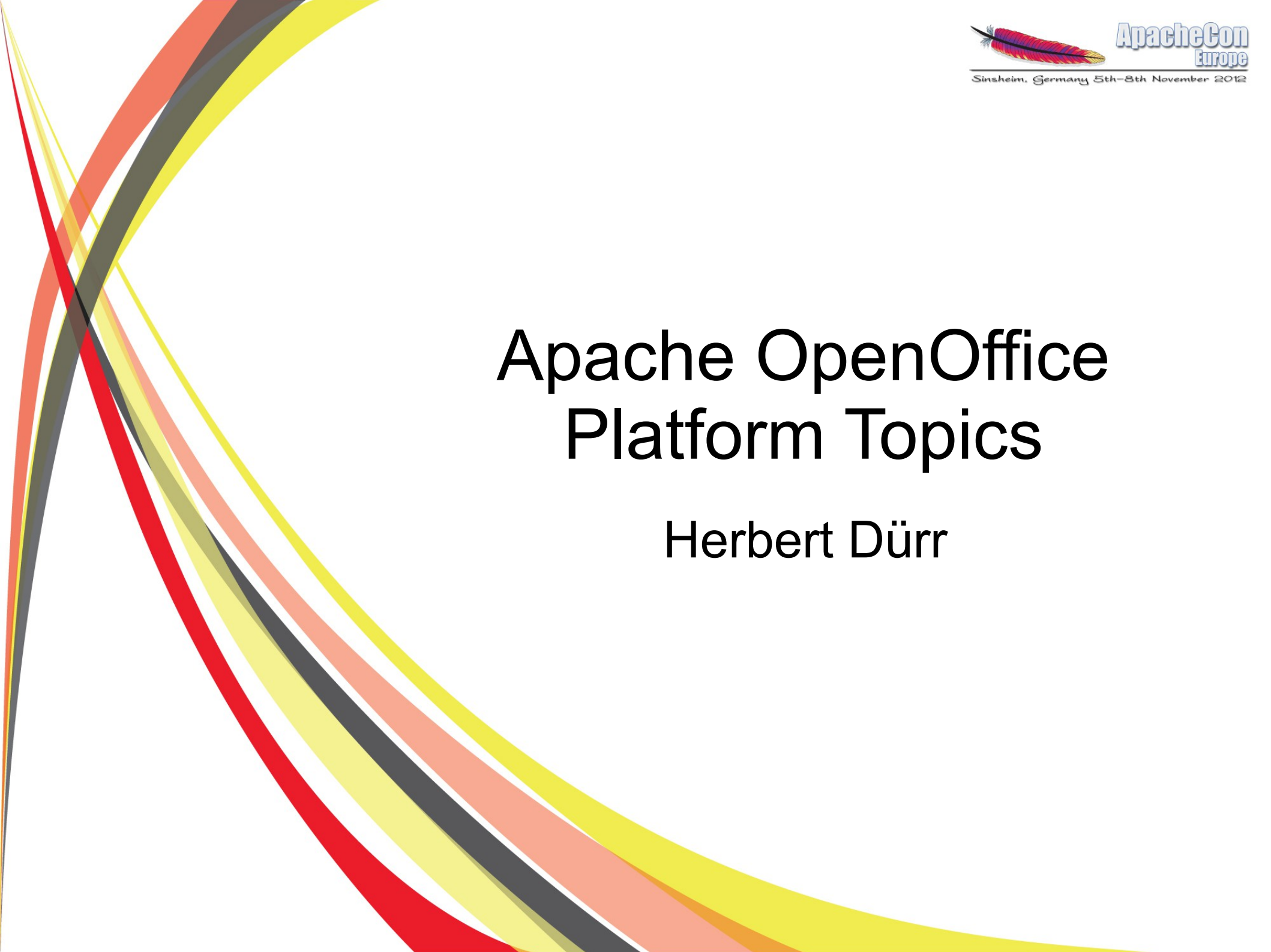


Apache OpenOffice Platform Topics

Herbert Dürr

OpenOffice
Much More Than the Apps

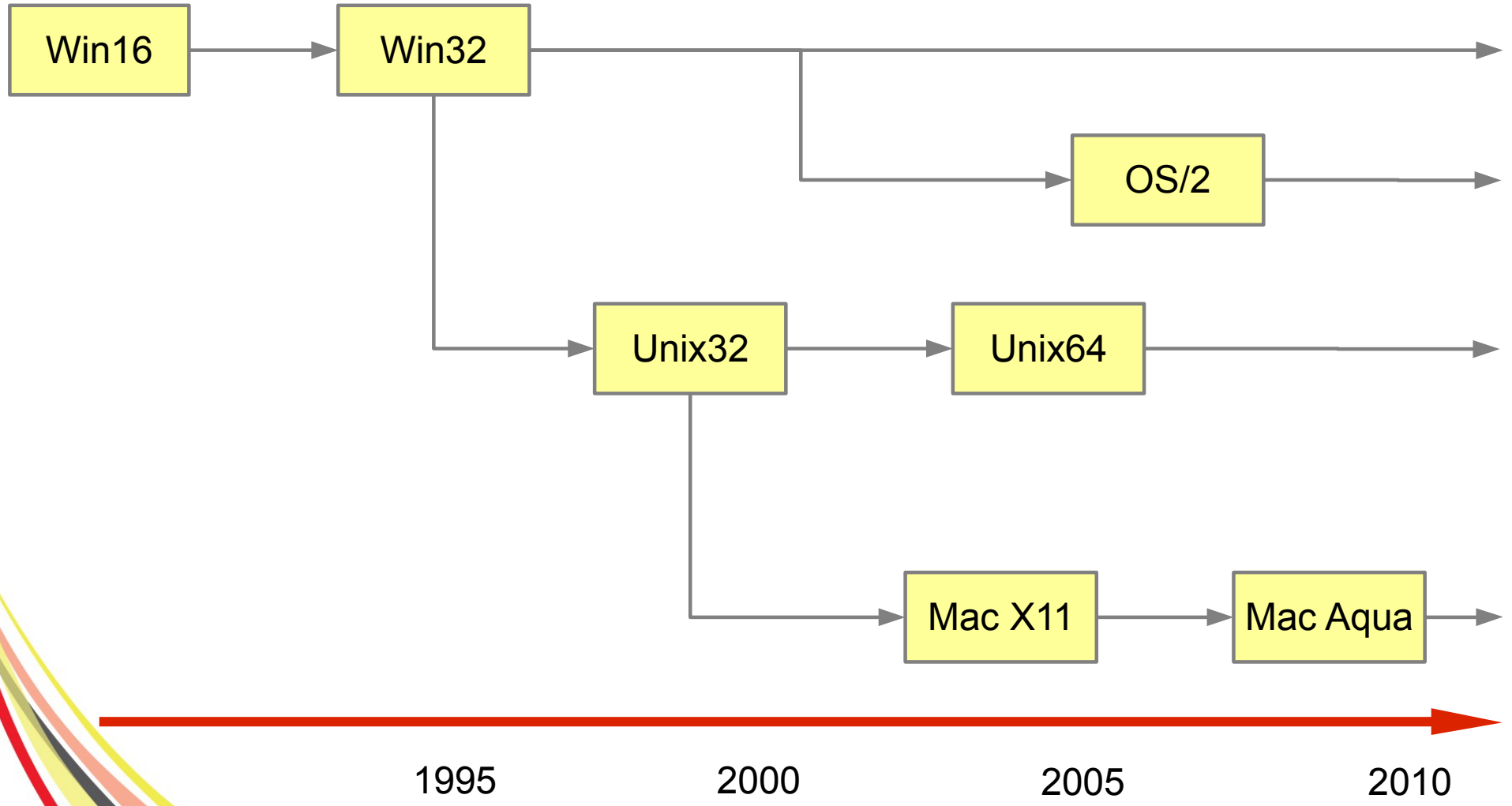
- OpenOffice and its Platforms
- Historic Background
- the Current Situation,
  Challenges and Opportunities
- Summary

OpenOffice and its Platforms

- as an end-user oriented application
  - target for the audience's platforms
  - with system integration
  - with full localization
- as a productivity suite
  - used by knowledge workers
  - with tightly coupled productivity programs

➔ we have to fulfill very high expectations

Historic Background



64bit General Considerations

- more address space
  - shared address space between documents
  - few 32bit applications remaining for sharing
  - better caching
  - helps to find wild pointers
  - allows better instrumenting
- better performance
  - more registers and better calling conventions
  - PC indirect addressing benefits ASLR
  - SSE floating point math

64bit Platform Specifics

- Linux/FreeBSD
  - missing 32bit libraries on distributions
- MacOSX
  - 32bit APIs are being deprecated
  - debugging on 32bit is more difficult
- Windows
  - 32bit Shell Extension are not supported on a normal 64bit Windows
  - 64bit libraries for better AppStore compliance

Platform Processing Power

- Number of Processors grows exponentially
- General Purpose Performance per Processor flattens out

➢

- Multiple Processes
  - per document document processes
  - increased reliability and stability
- Multi-Threading

OpenOffice Threading Overview

- It already runs many threads
  - main thread (message loop)
  - many worker threads
    (gdiplus, drag&drop, RPC, OLE, ...)
  - UI thread / worker are usually not separated
- But it does not run client threads

OpenOffice Threading Problems

- Does not scale with multiple clients
- A stability problem in a thread can take down the whole process
- The UNO API has thread safety issues
- Granularity of locks is quite coarse (global mutex, pool mutex, solar mutex, refcounters...)
- many more details are available: http://wiki.openoffice.org/wiki/Analysis/Multi-Threading

General Platform Opportunities

- separate processes per document
- deeper system integration
- improved Update Mechanism
- reduced Barrier of Entry for Development
- allowing the replacement of build oddities
  - e.g. the ancient Mozilla 1.7
- improved Graphics
  - native renderers
  - color depth, color spaces

Opportunities for Better System Integration on Mac

Opportunities for Better Deployment on MacOSX

- DiskImage vs. Package installations
- Packaging Java with the Download
- App Store and App Store Updates!
- Code Signing for “Gatekeeper”
- 64bit Porting

Opportunities on Windows

- AppStore: Deployment and Updates
- Improved Accessibility Integration
- The input language as the text language
- Improved OpenType Support
- Windows Address Book Integration
- 64bit Porting

Developer Perspective of the Situation on Linux/FreeBSD

- Works almost out of the box
- Dependency on GCC (>= 4.2)
- Eventual Alternative is Clang
- Desktop Environments
  - Plain X11
  - Gnome
  - KDE

Challenges for OpenOffice Developers on Windows

- Build based on Cygwin
- Visual Studio 2008 as the only compiler
- Odd binary artifacts needed for building
- Target baseline is Windows XP
- not yet compliant with the Win8 AppStore

Development Perspective of the Current Situation on Mac

- target baseline is still OSX 10.4
- dependency on XCode 3
  - installing that on a recent OSX is quite hard
- dependency on 32bit only APIs
- based on the Cocoa framework
- graphics based on Quartz, some OpenGL and ATSUI

Standard Template Library

- AOO as C++ application has to interface with C++ extensions and system libraries
  - stlport was used since the beginning of OOo
  - GNU libstdc++
  - Clang libc++
  - Apache stdcxx
  - Boost TR1
  - MSVC STL

➔ prefer system template libraries to allow direct use of system C++ libraries

Mobile Devices

- Better Support for Touch Interface Devices needed
- Model, View and Controller are not as cleanly separated as expected in new applications
  - classic VNC-like screen forwarding
  - web-based HTML content forwarding
  - major refactoring for MVC needed
    - work on this MVC refactoring, the new drawing primitives and their renderers open a whole new world of opportunities for a better experience

Touch Interfaces

- User Interaction Concepts Need Rethinking
  - hit targets need to be bigger
  - dialogs become much more rare
  - toolbar count has to be reduced
  - window menus are almost gone
  - context menus become popups
  - keyboard accelerators are gone
  - scrollbars become mostly obsolete

Touch Gestures

- Already supported
  - tap, double tap, scroll, pinch, spread, two finger right click
- Only partially supported
  - long press, page forward/backward, chiral scrolling
- Not yet supported
  - momentum, pan, rotate, flick, back, tap to zoom, three finger drag

Other Platform Trends

- The Cloud
  - for storing
  - for sharing
  - for live interactions
- GPU computing
  - allows faster image/video manipulations
  - allows faster solver

Summary

- Platforms are moving faster than ever
  - New Platforms are attractive to our Users
  - There are many exciting new software features that we should seamlessly interact with
  - Hardware Technologies like multi-core CPUs, high-resolution displays, powerful GPUs and new interactions offer plenty of opportunities

→ Be part of it, help out and join the FUN!