digitalekultur e.V.

Seminars at Evoke 2008

by Jerome Muffat-Meridol, Senior Application Engineer at Intel

"Dual-core was easy, but you want me to use 8 threads?!?"

Multi-threading has become a stable of modern programmers vocabulary, however it still presents challenges to the most seasoned programmers. Ensuring your multi-threaded design continues to scale from Dual-Core to Quad-Core and even more is a big challenge. Threading Building blocks provide a task abstraction that generally leads to better scheduling and load balancing. Additionally, the Threading Building Blocks help to reduce synchronization by design and by more efficient synchronization primitives. We'll review the Threading Building blocks and show a real world case where Threading Building Blocks were applied to a demo game application.

"Multi-core & multi-threading performance analysis made easier"

Getting maximum performance used to be about counting cycles and moving asm instructions around to achieve the best performance. However with the growing availability of multi-core processors and the associated multi-thread programming, we've all found cycle counting to fall short in performance analysis when you have to deal with things like contention vs. granularity, load balance vs. synchronization cost. Here we'll discuss several tools that go beyond cycle counting and Intel VTune and provide you with the right tools to analyze multi-thread programming.

A short biography of Jerome Muffat-Meridol

Jerome Muffat-Meridol has been writing software for the past twenty years with a focus on applications with a graphic side to them. Before joining Intel, he wrote deepViewer a photo browser built on a very innovative point & zoom interface, applying the know-how gained in ten years of video games development: he previously was Technical Director at Bits Studios, a London based studio specialised in console games.