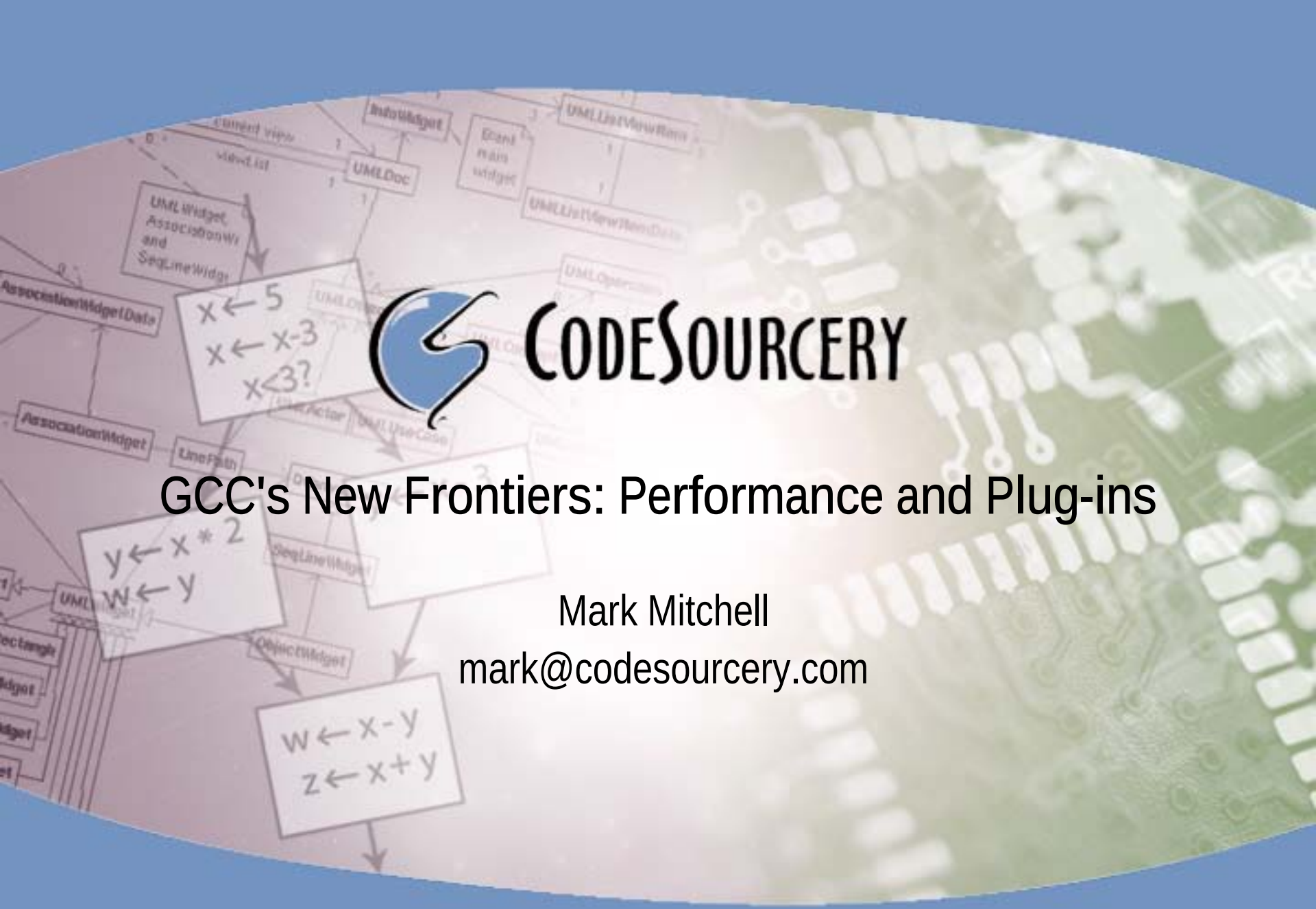
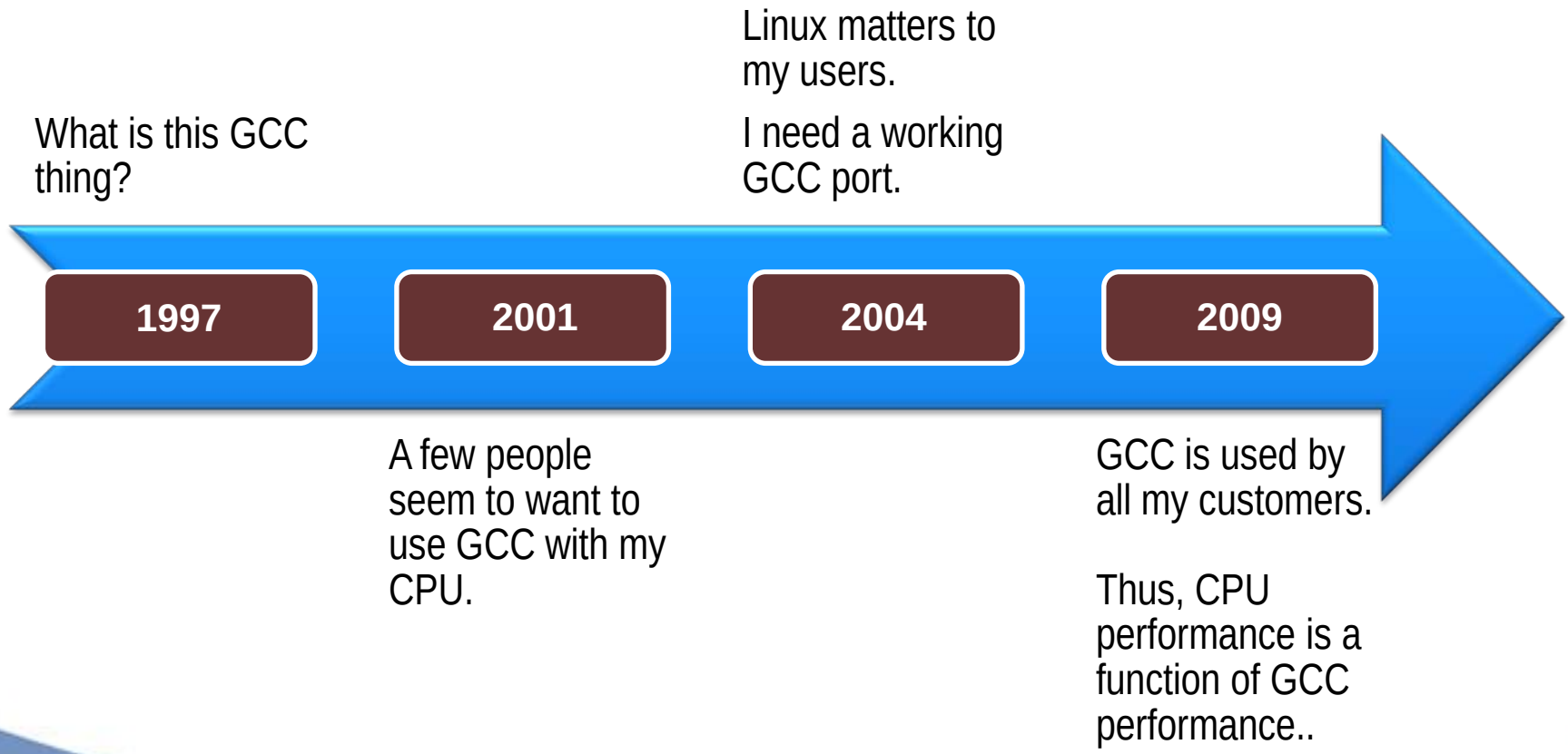


GCC's New Frontiers: Performance and Plug-ins

Mark Mitchell
mark@codesourcery.com

Performance

GCC Performance: Silicon Vendor Perspective



Link-Time Optimization: See The Big Picture

Traditional Optimization

Compile a.c
- Optimizing
- …g we can't
- …horrible

Compile b.c
- Just like a.c

Link
- No optimization
- Mechanical gluing only

Link-Time Optimization

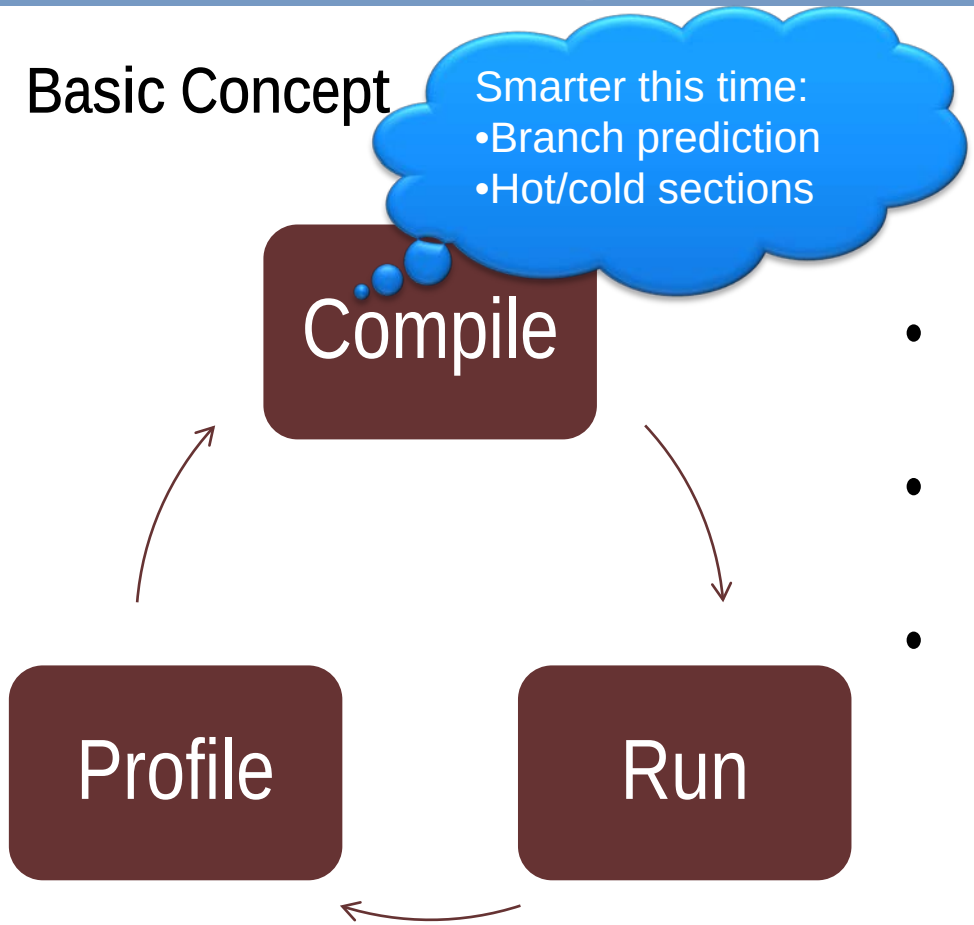
Compile a.c
- Optimize a bit
- And generate bytecode

Compile b.c
- Just like a.c

Link
- Load in all the bytecode
- Optimize it all at once

Profile-Guided Optimization

Basic Concept



Future Uses

Place data so as to maximize cache performance

- Minimize code size for cold code
- Optimistically replace variables by constants
- Auto-tune optimization parameters

Know more!
Guess less!

Thoughts for GCC Developers

Optimization is a quantitative exercise

- Optimization patches should not be posted without *quantitative* data

Benchmarking is part of the development process

- On multiple platforms
- In a scientific, methodical, reproducible way.

Tuning is vital

- Optimization work is not complete until all the parameters have been tuned
- In a scientific, methodical, reproducible way

Good performance requires awareness of the target machine

- Machine-specific oddities require machine-specific optimizations
- Machine-specific parameters are needed at all optimization stages

Thoughts for Silicon Vendors

Why You Must Invest

Your customers will not invest in compiler development.

Because GCC is Free Software, neither will ISVs.

Therefore, you must bear the cost.

Why You Must Invest Broadly

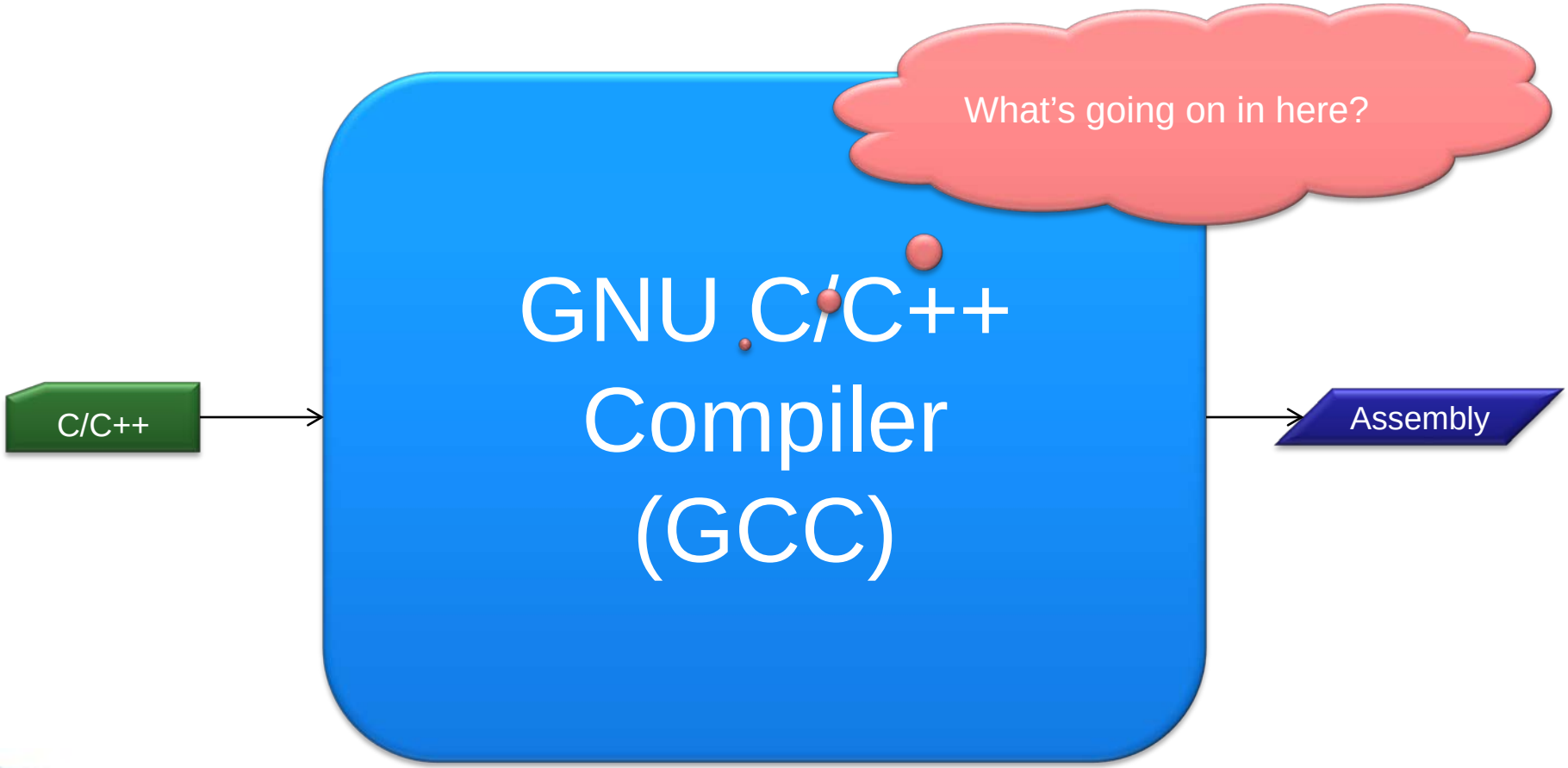
You would like to benefit only your CPUs.

Machine-independent improvements are needed to optimize for novel CPUs.

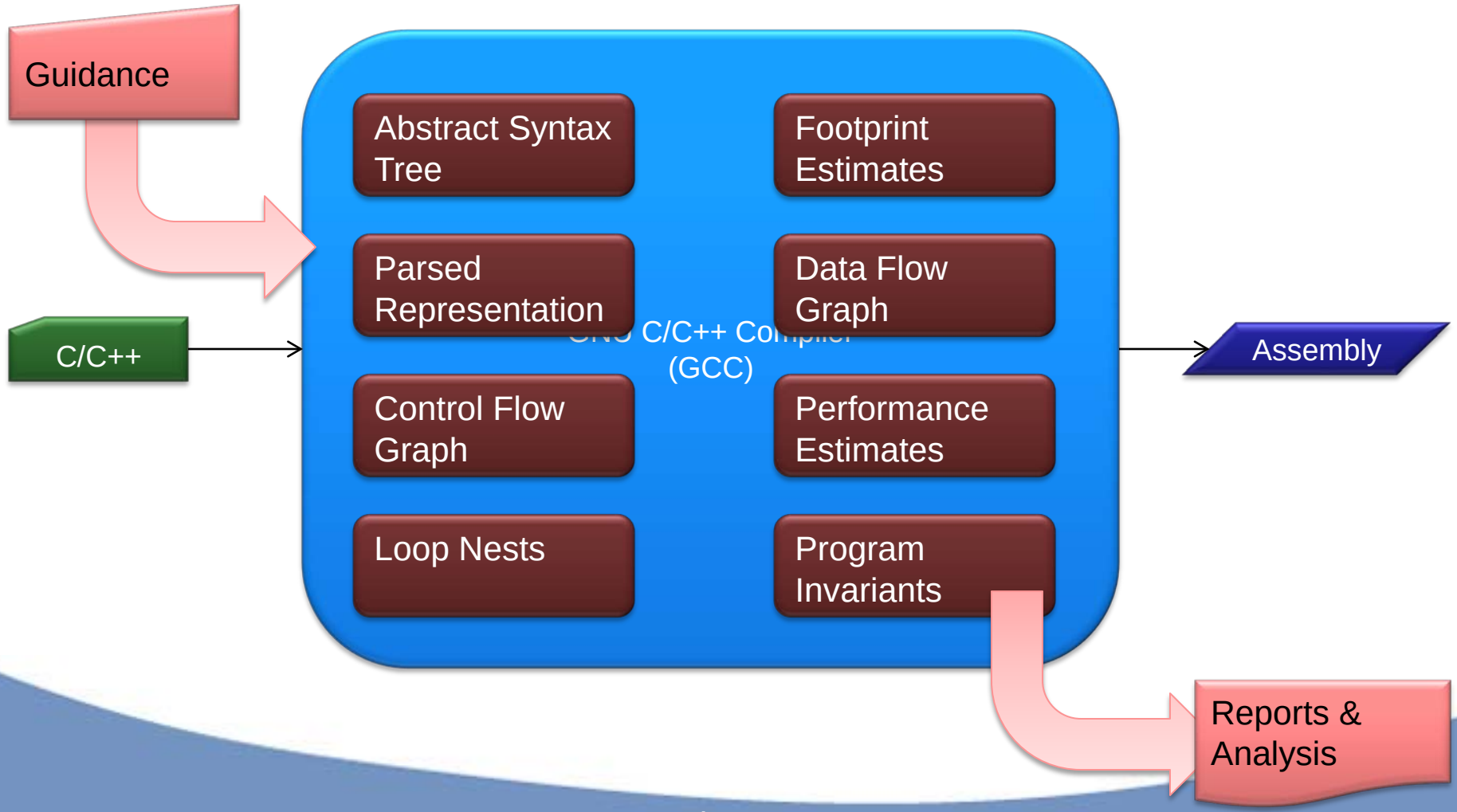
Therefore, you must invest broadly – or stop building novel CPUs.

Plug-Ins

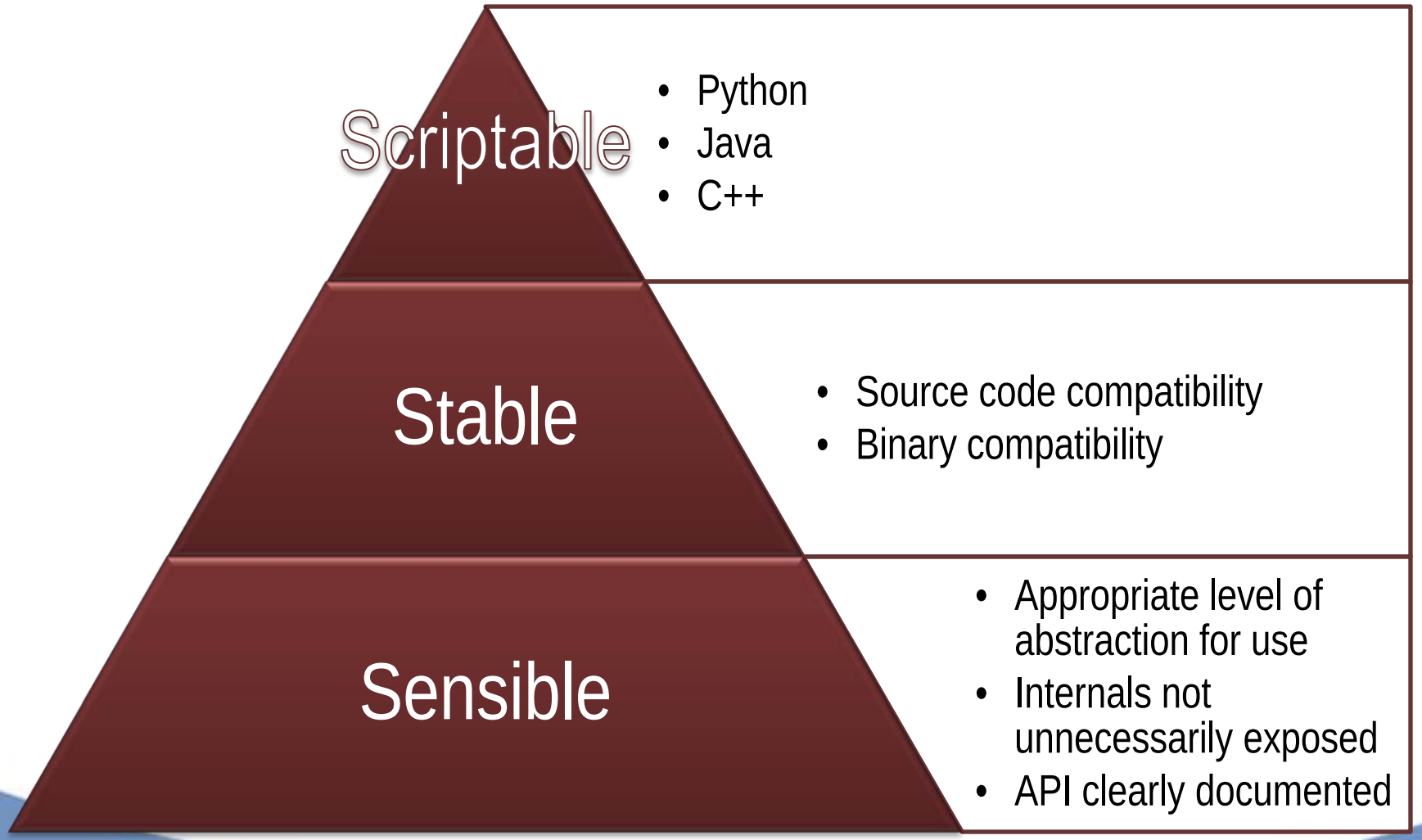
Compilers Are Black (and Blue?) Boxes

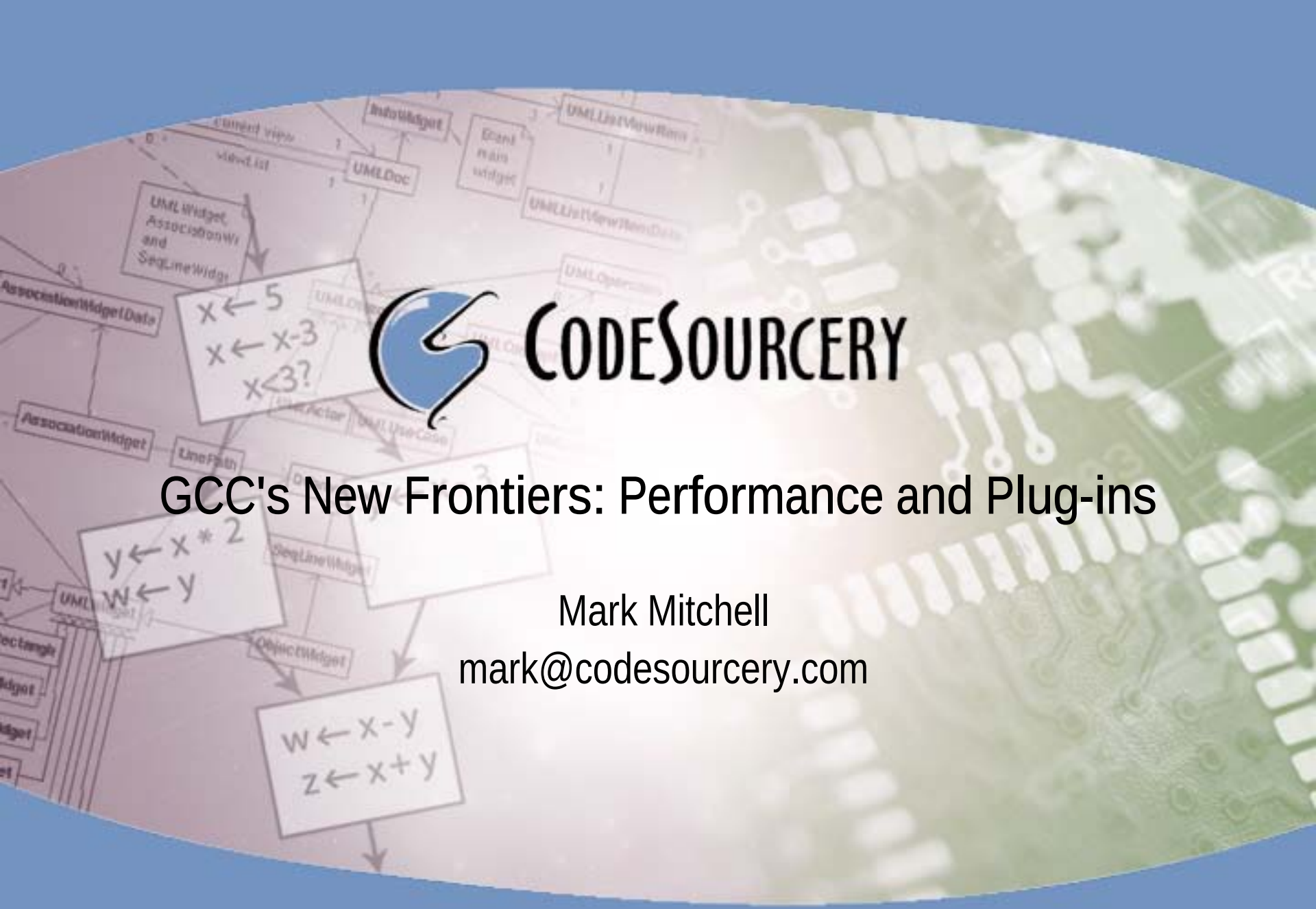


Compilers Should Be White Boxes



Requirements for A Plug-In API





GCC's New Frontiers: Performance and Plug-ins

Mark Mitchell
mark@codesourcery.com