Parallel Programming Made Easy

ITDevCon

Delphi European Conference

Primož Gabrijelčič, primoz@gabrijelcic.org

 www.thedelphigeek.com

November 17/18/19 '10 VERONA

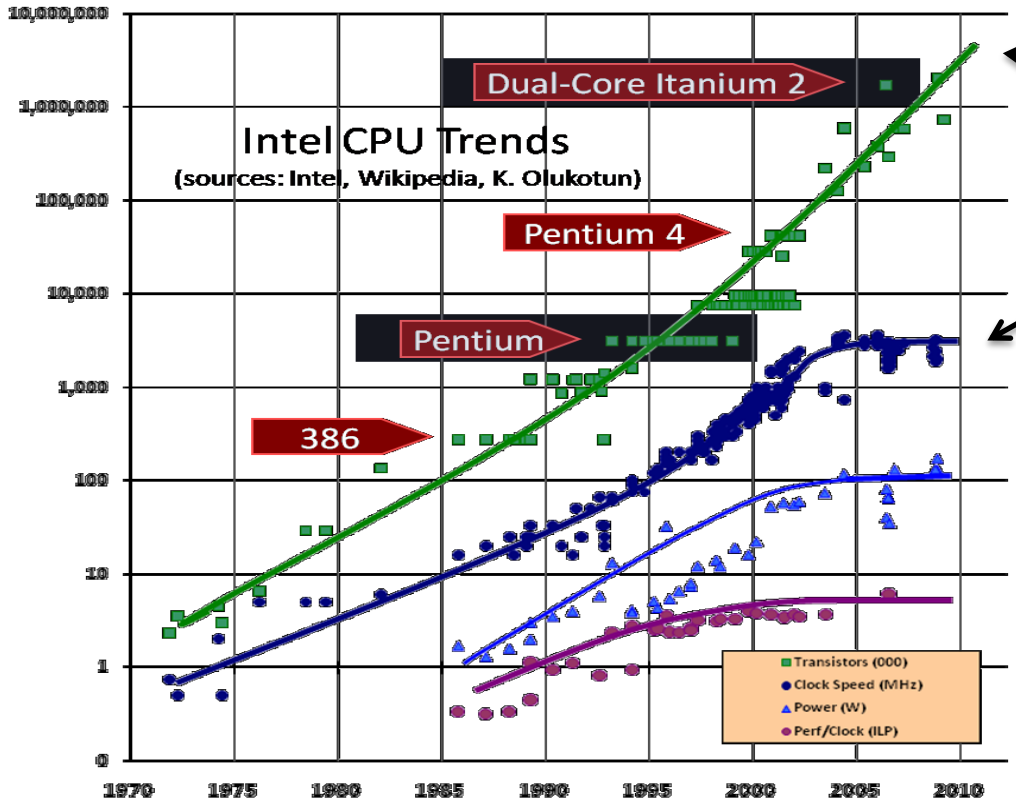
bit Time

Why and How

ITDevCon

The End of Free Lunch

ITDevCon



What we want

What we have

© Herb Sutter,
www.gotw.ca/publications/concurrency-ddj.htm

Three Paths …

- The Delphi Way
  - `TMyThread = class(TThread)`
- The Windows Way
  - `FHandle := BeginThread(nil, 0, @ThreadProc, Pointer(Self), 0, FThreadID);`
- The Lightweight Way (AsyncCalls)
  - `TAsyncCalls.Invoke(procedure begin DoTheCalculation; end);`
  - andy.jgknet.de

… Plus One

- The No-Fuss Way (OmniThreadLibrary)
  - `thread := TOmniFuture<integer>.Create(YourFunction);`
    `// do some work`
    `Show(thread.Value);`
  - otl.17slon.com
  - code.google.com/p/omnithreadlibrary
  - … D2007/2009⇨ only ☹

When To Use

- Slow background process
- Background communication
- Executing synchronous API
- Multicore data processing
- Multiple clients

OmniThreadLibrary

ITDevCon

OmniThreadLibrary is …

- … VCL for multithreading
  - Simplifies programming tasks
  - Componentizes solutions
  - Allows access to the bare metal
- … trying to make multithreading possible for mere mortals
- … providing *well-tested* components packed in *reusable* classes with *high-level* parallel programming support

Today‘s Topic

- Futures
- Join
- Parallel *for*

Futures

- Wikipedia
  - "They (futures) describe an object that acts as a proxy for a result that is initially not known, usually because the computation of its value has not yet completed."
  - Start background calculation, wait on result.

- How to use?
  - `Future := TOmniFuture<type>.Create(calculation);`
  - `Query Future.Value;`

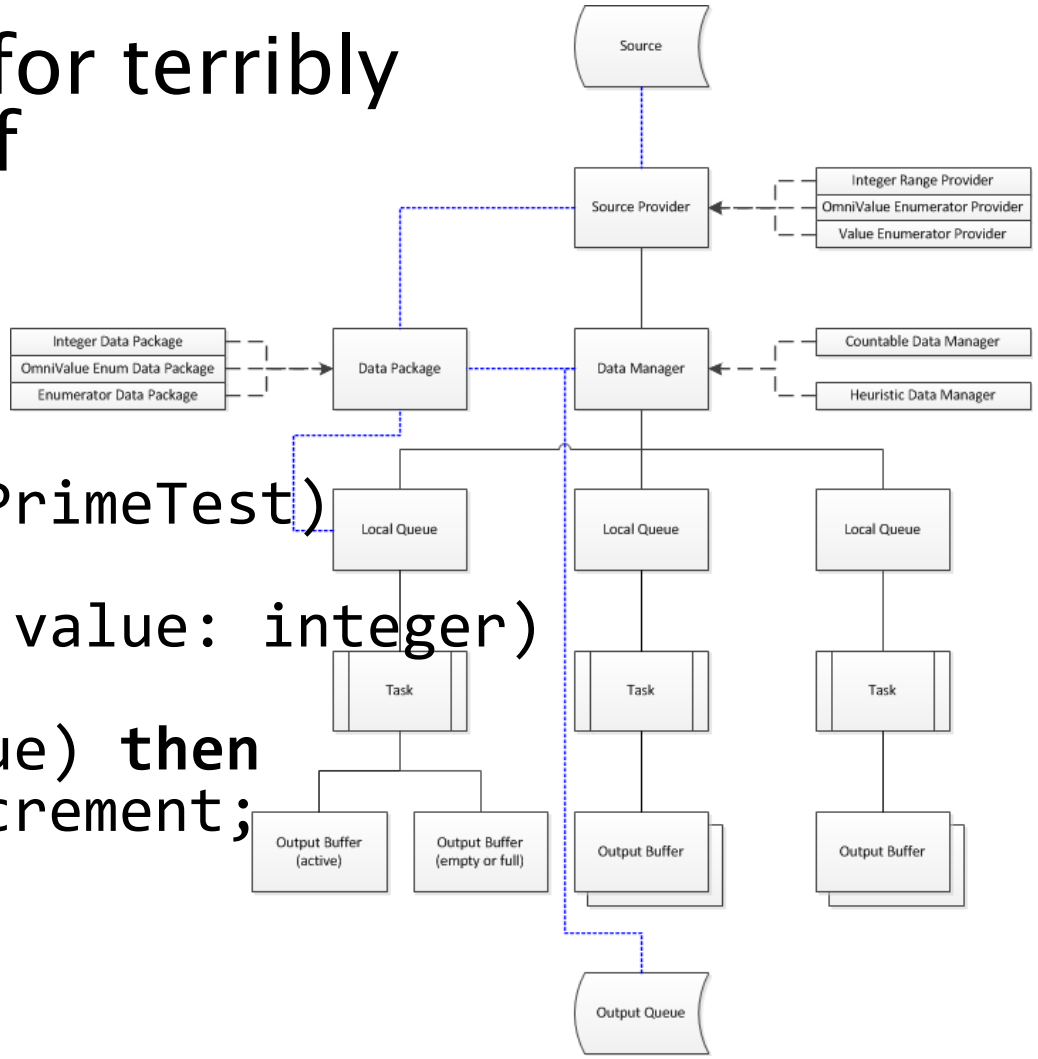
Join

- “Fork and Wait”
- Similar to Future
  - Start multiple background calculations
  - Wait for all to complete
  - No result is returned (directly)
- Two basic forms
  - `Join(task1, task2);`
  - `Join([task1, task2, task3, … taskN]);`

Parallel For

- A simple façade for terribly complicated stuff

```
Parallel
  .ForEach(1, CMaxSGPrimeTest)
  .Execute(
    procedure (const value: integer)
    begin
      if IsPrime(value) then
        numPrimes.Increment;
    end);
```



Parting Notes

ITDevCon

Be Afraid

- Be very afraid!

- Designing parallel solutions is hard
- Writing multithreaded code is hard
- Testing multicore apps is hard
- Debugging multithreading code is pure insanity

Keep in Mind

- Don’t parallelize everything
- Don’t create thousands of threads
- Rethink the algorithm
- Prove the improvements
- Test, test and test

Q & A

ITDevCon