Who am I?

- Tiberiu ‘Tibi’ Covaci
- Software engineer, 25 years experience
- CTO for Learning Connexions
- MCT since 2004, teaching .NET
- Telerik MVP & Insider
- MVP in Azure/ASP.NET
- Geek
- @tibor19 / #vslive
Agenda

- Parallel Programming
  - Definitions
  - Why
  - Laws
  - As-is
  - To-be
Definitions

- Pillars of Concurrency
  - Responsiveness and Isolation
    - Asynchronicity
  - Throughput and scalability
    - Concurrency
  - Consistency
    - Sharing
Definitions

- Asynchronous
  - Active waiting
- Concurrency
  - Several stuff at the same time
- Multithreading
  - Multiple threads of executions
- Parallel
  - Multiple computations concurrently
Why Concurrent Programming?

- The free lunch is over
- We need to do more with more
- We have to take responsibility for the way we write our code
Laws affecting Parallel Programming

- Moore’s Law
- Amdahl’s Law
- Gustafson’s Law
Moore’s Law (1965)

The complexity for minimum component cost has increased at a rate of roughly a factor 2 every 2
Amdahl’s Law (1967)

- The speedup of a program using multiple processors in parallel computing is limited by the time needed for the sequential fraction of the program.

\[
\text{SpeedUp} = \frac{1}{S + \frac{P}{N}}
\]

- If \( S = 10\% \) => Max Speedup = \( x10 \)

<table>
<thead>
<tr>
<th>Buy the ingredients</th>
<th>Prepare the food</th>
<th>Cook it</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Gustafson’s Law (1988)
• Any sufficiently large problem can be efficiently parallelized.

Is your problem large enough?

Buy the ingredients  Prepare the food  Cook it
Parallel Programming today

- Threads
  - Single threaded processes
  - Multi threaded processes
- Thread pools
- Synchronizations Objects
Threads

- The unit of execution in a process
- Heavyweight
  - Resource intense
  - Simultaneous threads cause context switches
Single threaded processes

- One application at the time
- To achieve multitasking
  - Hardware Interrupts
  - Terminate and Stay Resident Programs
Multi threaded processes

- Several threads in a process
- Single Core - Time sharing
- Multi Core - Machine aided parallelism
Thread pools

- Threads are pre-created
- Fewer threads - Less thread overhead
- Less control - Threads are bundled
Synchronization Objects

- Monitors (lock / SynkLock)
- Manual/Auto Reset Events
- The Interlock class
- Mutexes
- Semaphores
- Reader Writer Locks
Parallel Programming tomorrow

- Task parallelism
- Imperative data parallelism
- Declarative data parallelism
- Asynchronous programming
- Synchronization/cooordination objects
Task parallelism

- Task/Task<T>
- AggregateException
Imperative data parallelism

- Parallel.For
- Parallel.ForEach
- Parallel.Invoke
Declarative data parallelism

- Parallel Linq (PLinq)
  - AsParallel
  - AsOrdered
Asynchronous programming

- APM
  - BeginInvoke/EndInvoke
- EAP
  - MethodAsync/MethodCompleted event
- Async/Await
  - The compiler will rewrite your code to take care of the asynchronous portion of your program
Coordination objects

- SpinLock
- SpinWait
- SemaphoreSlim
- ManualResetEventSlim
- LazyInit
- CountdownEvent
- WriteOnce
Synchronization

- Memory synchronization / fences
- Barriers
- Mutual exclusion
Sharing State and Synchronization

- Don’t share!
- Read only data
- Data isolation
- Synchronization
Deadlocks

- Mutual exclusion
- Hold and wait
- No preemption
- Circular wait
Agenda

- Parallel Programming
  - Definitions
  - Why
  - Laws
  - As-is
  - To-be
Thank you!

- Please fill out the evaluation!

- Contact me by
  - Email tibi@covaci.se
  - Twitter tibor19

And Don’t Forget to Write!