

Parallel Computing @ MS

Wolfgang Dreyer

HPC Product Manager

Microsoft Deutschland GmbH

HPC++

Microsoft's Vision for HPC

“Provide the platform, tools and broad ecosystem to reduce the complexity of HPC by making parallelism more accessible to address future computational needs.”

Reduced Complexity

Ease deployment for larger scale clusters

Simplify management for clusters of all scale

Integrate with existing infrastructure

Mainstream HPC

Address needs of traditional supercomputing

Address emerging cross-industry computation trends

Enable non-technical users to harness the power of HPC

Broad Ecosystem

Increase number of parallel applications and codes

Offer choice of parallel development tools, languages and libraries

Drive larger universe of end-users, developers, and system administrators

Looking Forward: Hard Problems

- Scaling distributed systems is hard
- Data sets are increasing
- Programming models are complex

$\frac{1}{x}$ $(0, +\infty)$ $\frac{1}{x^2}$ $(0, +\infty)$ $\frac{1}{x^3}$ $(-\infty, +\infty)$ $\frac{1}{x^4}$ $(0, +\infty)$ $\frac{1}{x^5}$ $(-\infty, +\infty)$ $\frac{1}{x^6}$ $(0, +\infty)$ $\frac{1}{x^7}$ $(-\infty, +\infty)$ $\frac{1}{x^8}$ $(0, +\infty)$ $\frac{1}{x^9}$ $(-\infty, +\infty)$ $\frac{1}{x^{10}}$ $(0, +\infty)$ $\frac{1}{x^{11}}$ $(-\infty, +\infty)$ $\frac{1}{x^{12}}$ $(0, +\infty)$ $\frac{1}{x^{13}}$ $(-\infty, +\infty)$ $\frac{1}{x^{14}}$ $(0, +\infty)$ $\frac{1}{x^{15}}$ $(-\infty, +\infty)$

Tedy $f'(x_0) = \lim_{h \rightarrow 0} \frac{f(x_0+h) - f(x_0)}{h}$ je okamžitá změna veličiny $y = f(x)$ v čase $x = x_0$. Pak $\frac{f(t+\Delta t) - f(t)}{\Delta t}$ je průměrná změna...

We need simpler programming models

$y = f(x) = k_1(x - x_0)$ kde $k_1 = \frac{f(x_0+h) - f(x_0)}{h}$ je směrnice tečny. Jestliže je bod B blízko k bodu A, $h \rightarrow 0$, přibližně se čísla na tečny grafu funkce f v bodě $A = [x_0, f(x_0)]$ rovnají $y - f(x_0) = k_2(x - x_0)$.

$k_2 = \lim_{h \rightarrow 0} \frac{f(x_0+h) - f(x_0)}{h} = f'(x_0)$ pokud $f'(x_0)$ existuje. Pokud existuje jen $f'_+(x_0) = k_+$ (resp. $f'_-(x_0) = k_-$), je k_+ (resp. k_-) rovnicí jednoramenné tečny zprava (resp. zleva) v bodě $A = [x_0, f(x_0)]$.

Je-li $f'(x_0) = 0$ (resp. $f'(x_0) = \pm\infty$) a f je spojitá v x_0 , je tečna v bodě A je přímkou, která prochází bodem dotyku A a je kolmá na tečnu k_+ (resp. k_-).

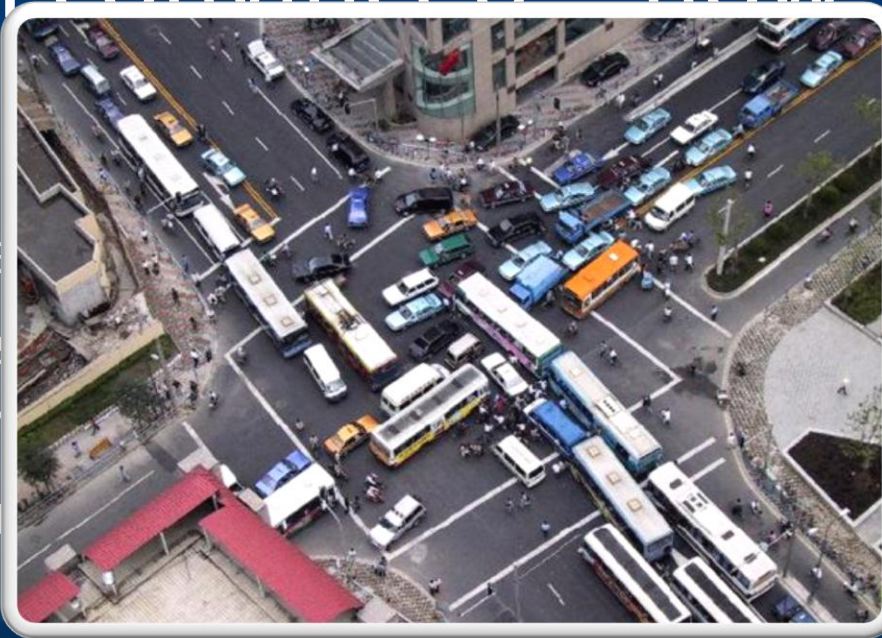
Je-li normála je rovnoběžná s osou y (resp. x).

Def 6.2. (i) Funkce f má v bodě $x_0 \in D(f)$ lokální minimum (resp. lokální maximum) pokud $f(x_0) \leq f(x)$ (resp. $f(x_0) \geq f(x)$) pro všechna x v nějakém okolí bodu x_0 . (ii) ostře lokální minimum (resp. ostře lokální maximum) pokud $f(x_0) < f(x)$ (resp. $f(x_0) > f(x)$) pro všechna x v nějakém okolí bodu x_0 .

Def 6.6 (Fermatova) Necht f má v bodě x_0 lokální extrém. Pak $f'(x_0) = 0$ (resp. $f'(x_0) = \pm\infty$).

I'm convinced now what?

- Multithreading is not a silver bullet today
 - Doable
 - Parallelism is not a silver bullet
 - Implementing parallelism is not well known, nor easy to do
 - So many people are still using single-threaded code
- Businesses have little desire to “go deep”
 - Best devs should focus on business value, not concurrency
 - Need simple ways to allow all devs to write concurrent code



HPC++

Crossing the Chasm

- Embrace existing programming models
 - We heart MPI
- Increase reach of existing codes
 - Cluster SOA, .NET/WCF, Excel integration
- Invest in mainstream parallel dev tools
 - Unlock multi/many-core for breadth developers
 - Evolve hybridized and scale-out models
- Seek opportunities for “automatic” parallelism
 - F#
 - DryadLINQ



Example: “Race Car Drivers”

```
IEnumerable<RaceCarDriver> drivers = ...;
var results = new List<RaceCarDriver>();
foreach(var driver in drivers)
{
    if (driver.Name == queryName &&
        driver.Wins.Count >= queryWinCount)
    {
        results.Add(driver);
    }
}
results.Sort((b1, b2) =>
    b1.Age.CompareTo(b2.Age));
```

Manual Parallel Solution

```
IE... > drivers = ...;  
var... CarDriver>();  
int... ProcessorCo...  
int... Count;  
var... ator();  
try {  
    using (var...  
        for(int...  
            Thread...  
                while...  
                    Rac...  
                        lock...  
                            break;  
                                su...  
                                    .Decrement(re...  
                                        ((b1, b2) => b1.Age.CompareTo(b2.A...  
                                            }  
                                                }  
                                                    finally { if (enumerator is IDisposable) ((IDisposable)enumerator
```

Synchronization Knowledge

Inefficient locking

Lack of foreach simplicity

Manual aggregation

Tricks

Lack of thread reuse

Heavy synchronization

Non-parallel sort

LINQ Solution

```
var results = from driver in drivers.AsParallel()  
              where driver.Name == queryName &&  
                  driver.Wins.Count >= queryWinCount  
              orderby driver.Age ascending  
              select driver;
```

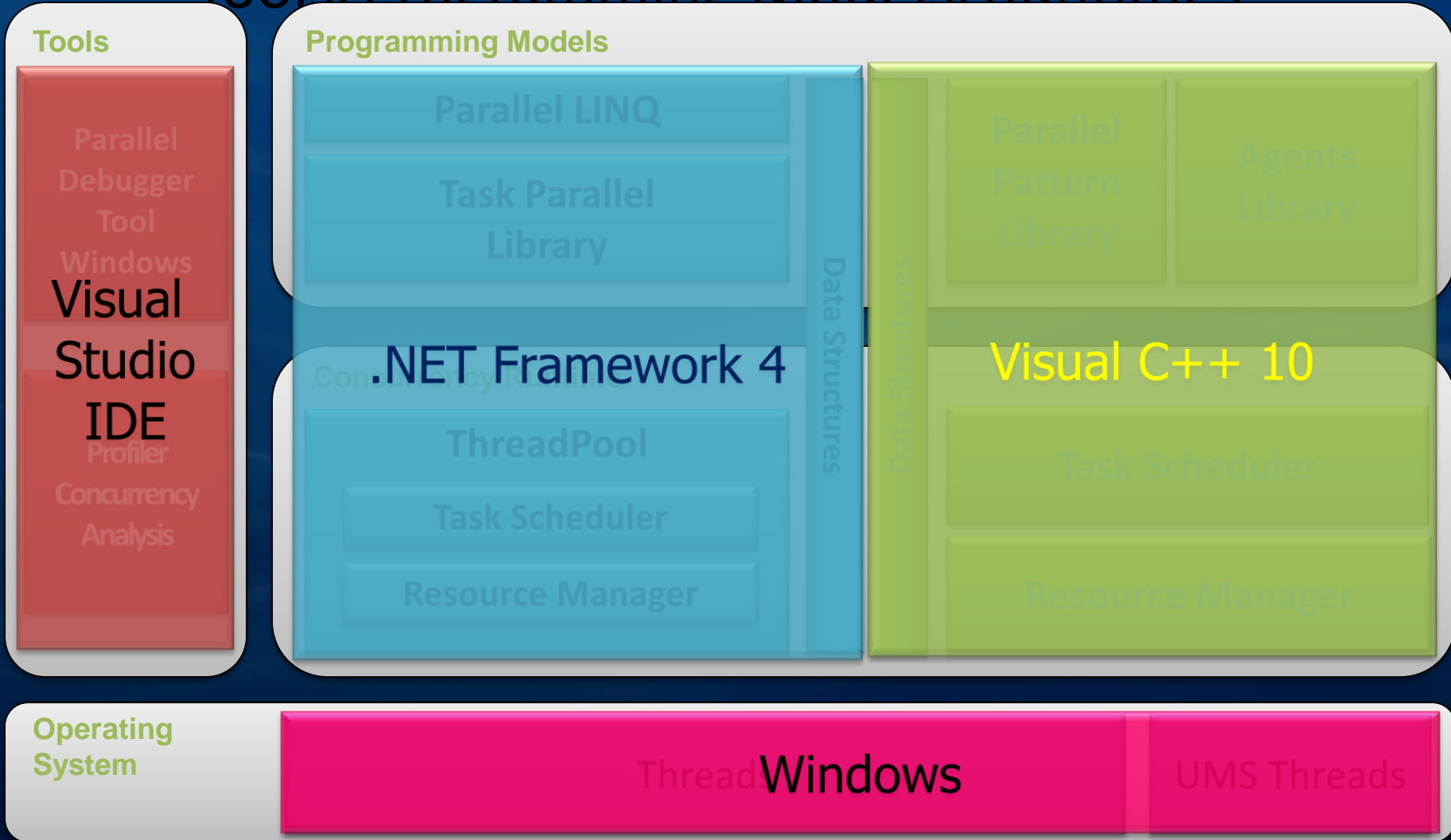
PLINQ

- Parallel implementation of LINQ-to-Objects
 - Supports all of the .NET Standard Query Operators
 - Aggregate, All, Any, Average, Cast, Concat, Contains, Count, DefaultIfEmpty, Distinct, ElementAt, ElementAtOrDefault, Empty, Equals, Except, First, FirstOrDefault, GroupBy, GroupJoin, Intersect, Join, Last, LastOrDefault, LongCount, Max, Min, OfType, OrderBy, OrderByDescending, Range, Repeat, Reverse, Select, SelectMany, SequenceEqual, Single, SingleOrDefault, Skip, SkipWhile, Sum, Take, TakeWhile, ThenBy, ThenByDescending, ToArray, ToDictionary, ToList, ToLookup, Union, Where
 - Knobs
 - AsParallel, AsSequential, AsOrdered, AsUnordered
 - WithCancellation, WithDegreeOfParallelism, WithExecutionMode, WithMergeOptions
 - Works for any `IEnumerable<T>`
 - Optimizations for other types (`T[]`, `IList<T>`)
 - Supports custom partitioning (`Partitioner<T>`, `OrderedPartitioner<T>`)
 - Built on top of the rest of Parallel Extensions

HPC++

Visual Studio 2010

Tools/Programming Models/Runtimes



Key:

Managed

Native

Tooling

Developer Accessibility

- Program to tasks, not threads
- Focus on problem domain
- C++, .NET, STM, agents,
- Embrace standards

Make it easier to express and manage the correctness,

ability of all

- Concurrency Runtime
- Enable latent parallelism
- Core OS changes (UMS)
- Scale-out runtimes (Dryad)

- Debugger speaks prog models
- Debugger must scale out
- Correctness analysis
- Accessible profiling

Enable developers to express parallelism easily and focus on the problem to be solved

Improve the efficiency and scalability of parallel applications

Simplify the process of designing and testing parallel applications

HPC++

- Microsoft HPC Web site
 - <http://www.microsoft.com/hpc>
- Parallelism at Microsoft
 - <http://msdn.microsoft.com/concurrency>
- Visual Studio 2010 Beta
 - <http://msdn.microsoft.com/visualstudio>
- Windows HPC TechCenter
 - <http://technet.microsoft.com/en-us/hpc/default.aspx>
- HPC on MSDN
 - <http://code.msdn.microsoft.com/hpc>


Microsoft[®]

Your potential. Our passion.[™]

© 2008 Microsoft Corporation. All rights reserved. Microsoft, Windows, Windows Vista and other product names are or may be registered trademarks and/or trademarks in the U.S. and/or other countries. The information herein is for informational purposes only and represents the current view of Microsoft Corporation as of the date of this presentation. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information provided after the date of this presentation.

MICROSOFT MAKES NO WARRANTIES, EXPRESS,
IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS PRESENTATION.

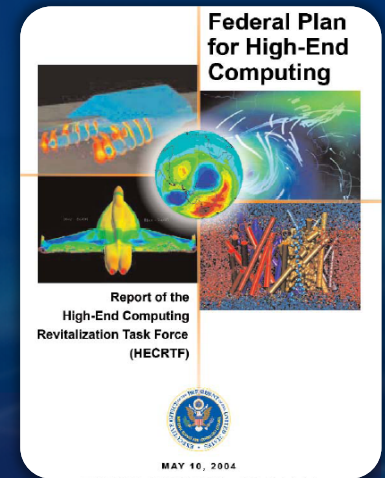
HPC++

 $t_2 - t_1 = g$
 $(t_2 = t_1)$
 $(0.5 \times V_2)^2 + (0.5 \times V_2)^2 = (0.5 \times V_1)^2$

$(0.5 \times V_2)^2 + (0.5 \times V_2)^2 = (0.5 \times V_1)^2$
 $(0.5 \times V_2)^2 = (0.5 \times V_1)^2$
 $V_2 = V_1$
 $= \text{constant}$
 $= PC$

HPC++

Challenge: High Productivity Computing



“Make high-end computing easier and more productive to use.

Emphasis should be placed on time to solution, the major metric of value to high-end computing users...

A common software environment for scientific computation encompassing desktop to high-end systems will enhance productivity gains by promoting ease of use and manageability of systems.”

2004 High-End Computing Revitalization Task Force
Office of Science and Technology Policy, Executive Office of the President



What's new in Windows HPC Server 2008?

- New System Center UI
- PowerShell for CLI Management
- High Availability for Head Nodes
- Windows Deployment Services
- Diagnostics/Reporting
- Support for Operations Manager

Systems Management

- Support for SOA and WCF
- Granular resource scheduling
- Improved scalability for larger clusters
- New Job scheduling policies
- Interoperability via HPC Profile

Job Scheduling

Microsoft® HPC Pack 2008

Networking & MPI

- NetworkDirect (RDMA) for MPI
- Improved Network Configuration Wizard
- Shared Memory MS-MPI for multi-core
- MS-MPI integrated with Windows Event Tracing

Storage

- Improved iSCSI SAN & parallel file system Support in Win2008
- Improved Server Message Block (SMB v2)
- New 3rd party parallel file system support for Windows
- New Memory Cache Vendors

HPC Server at scale

- Data center integration
 - HPC continues to integrate with the general computing fabric
 - Partnerships with System Center, MOAB, and more
- Commitment to performance
 - Open source contributions to ANL, OpenFabrics, and more
 - GP-GPU integration and tuning
 - Performance tuning labs
 - User mode and kernel mode RDMA
- Cloud integration
 - Offload or remote computation sometimes makes sense
 - Must solve problems with data locality, licensing, code installation

Approaching the storage problem

- Segmentation models

- Structure models:

Size	Locality	Structure
<ul style="list-style-type: none"> • Large files • Lots of small files 	<ul style="list-style-type: none"> • Server-read/server-write • Server-read/local-write • Schedule for local read (data locality) 	<ul style="list-style-type: none"> • Unstructured (files) • Semi-structured (HDF5, etc.) • Structured (SQL)

- Programming models:

Matrix Operations	Pseudo Task Parallel	LINQ-like
<ul style="list-style-type: none"> • GEMM: $C \leftarrow \alpha AB + \beta C$ 	<ul style="list-style-type: none"> • foreach (file in dir) • parallel.foreach (...){ xxxx } 	<ul style="list-style-type: none"> • array{...} • map (func, array)



Insights

- Many apps mine data without SQL
 - Enormous amounts of data in unstructured flat files
 - Queries are too convoluted for SQL statement or SQL knowledge is limited
 - But... predictive analytics and general business intelligence are starting to use HPC infrastructure
- No good *general* out of the box solutions
 - Pre-canned prog model and storage system works for some problems
 - Expensive PFS works well for existing and MPI apps
 - Some customers decide to build their own

Strategic issues

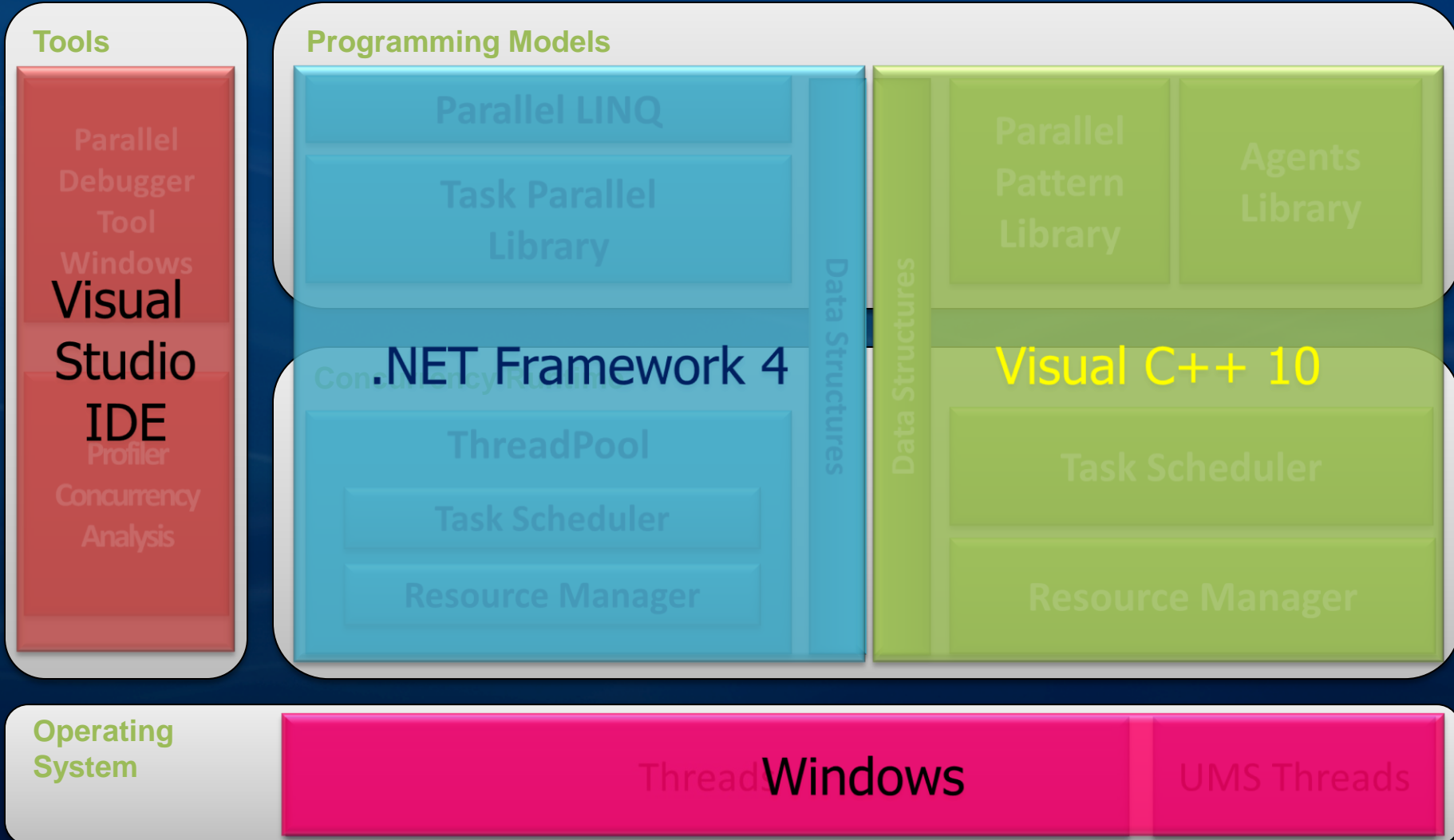
- Our long term storage strategy is governed by three tensions:
 - Success and timing of NFS v4.1
 - Stuff like Dryad/LINQ is loved by Emerging/Finance
 - Locality is critical
 - Compute in the cloud makes the data locality problem especially complex
- General purpose vs. model specific
 - Model specific storage models are easier to build
 - So many models to choose from...
 - MPI requires parallel file systems, not model based systems

Models

- We love MPI
 - MPI cited as primary programming model for over 60% of users
 - We're contributing to the MPI Forum
 - Open source contributions to MPICH at ANL are largest in MS history
- SOA & Horizontal applications
 - SOA innovations will help conveniently parallel applications scale to clusters
 - SOA infrastructure is designed to help horizontal apps scale
 - ISVs and developers can focus on writing code, not building middleware
 - Users focus on doing science rather than learning job control languages
 - Excel is used practically everywhere
- Accessible, “mass market” parallelism
 - How to leverage increasing core count on nodes and workstations
 - Parallel Computing developer technology shipping from MS soon
 - More to come: deeper integration of hybrid development models

Visual Studio 2010

Tools/Programming Models/Runtimes



Key:

Managed

Native

Tooling