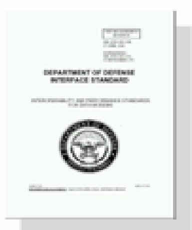
Computational Design Using MATLAB® and Simulink®

Pieter J. Mosterman
Senior Research Scientist

Introduction — Model-Based Design

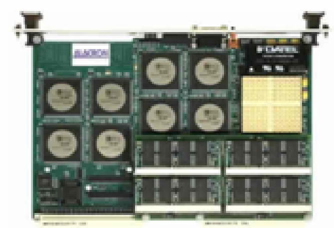
- Computational models are key!

Requirements & Specifications

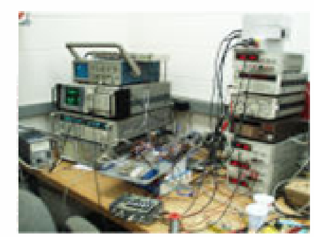


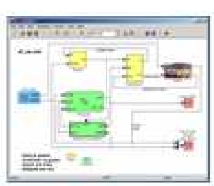
Design

Implementation



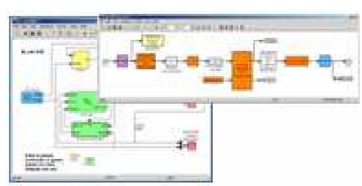
Test & Verification





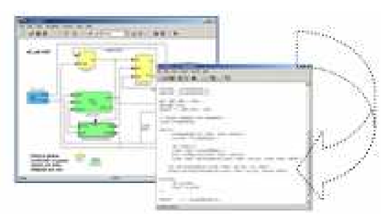
Executable Specifications

- Reduce Ambiguity Avoid Re-Work



Design With Simulation

- Rapid Design Iterations



Automatic Code Generation

- Minimizes Coding Errors

Continuous Verification

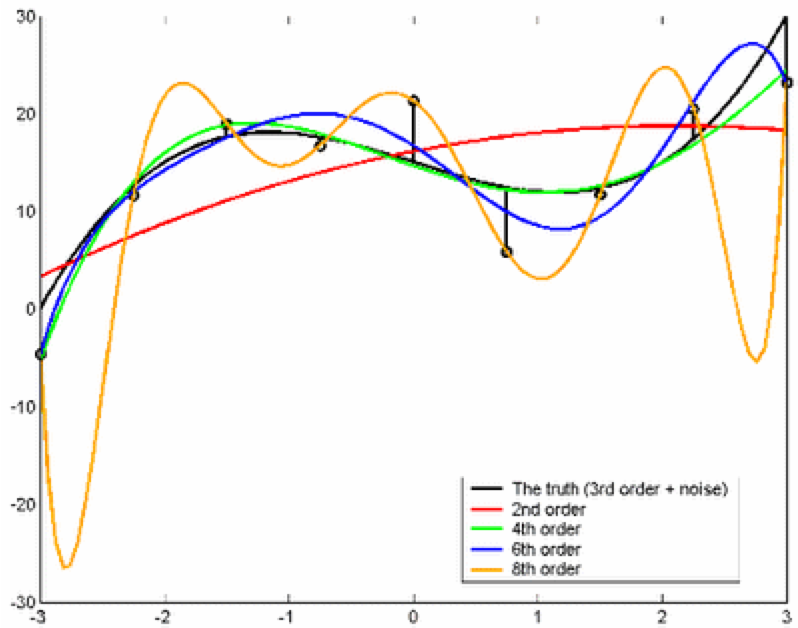
- Detect Errors Earlier

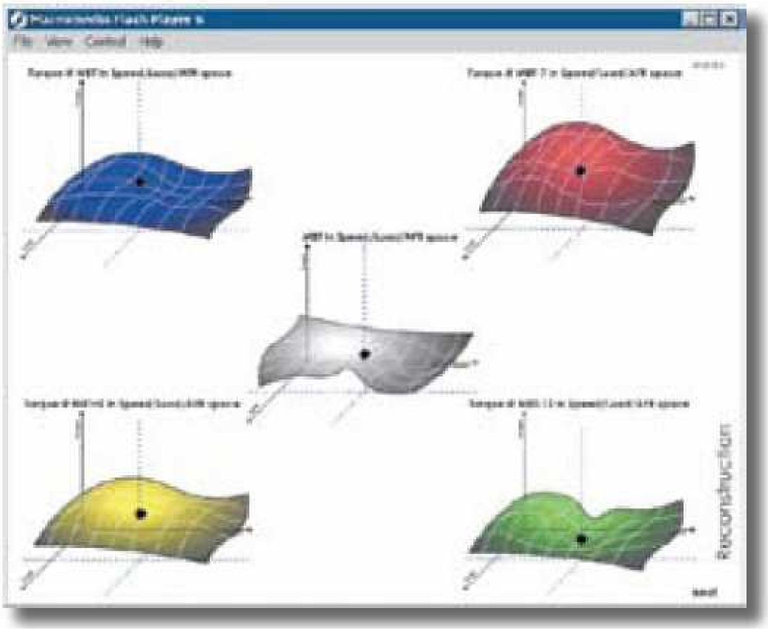
Agenda

- Computational design example (~5 min.)
  - Engine calibration
- Accelerated simulation (~5 min.)
  - The Embedded MATLAB Simulink block
- Distributed Simulation (~10 min.)
  - Distributed MATLAB
- Conclusions (~5 min.)

Engine Calibration

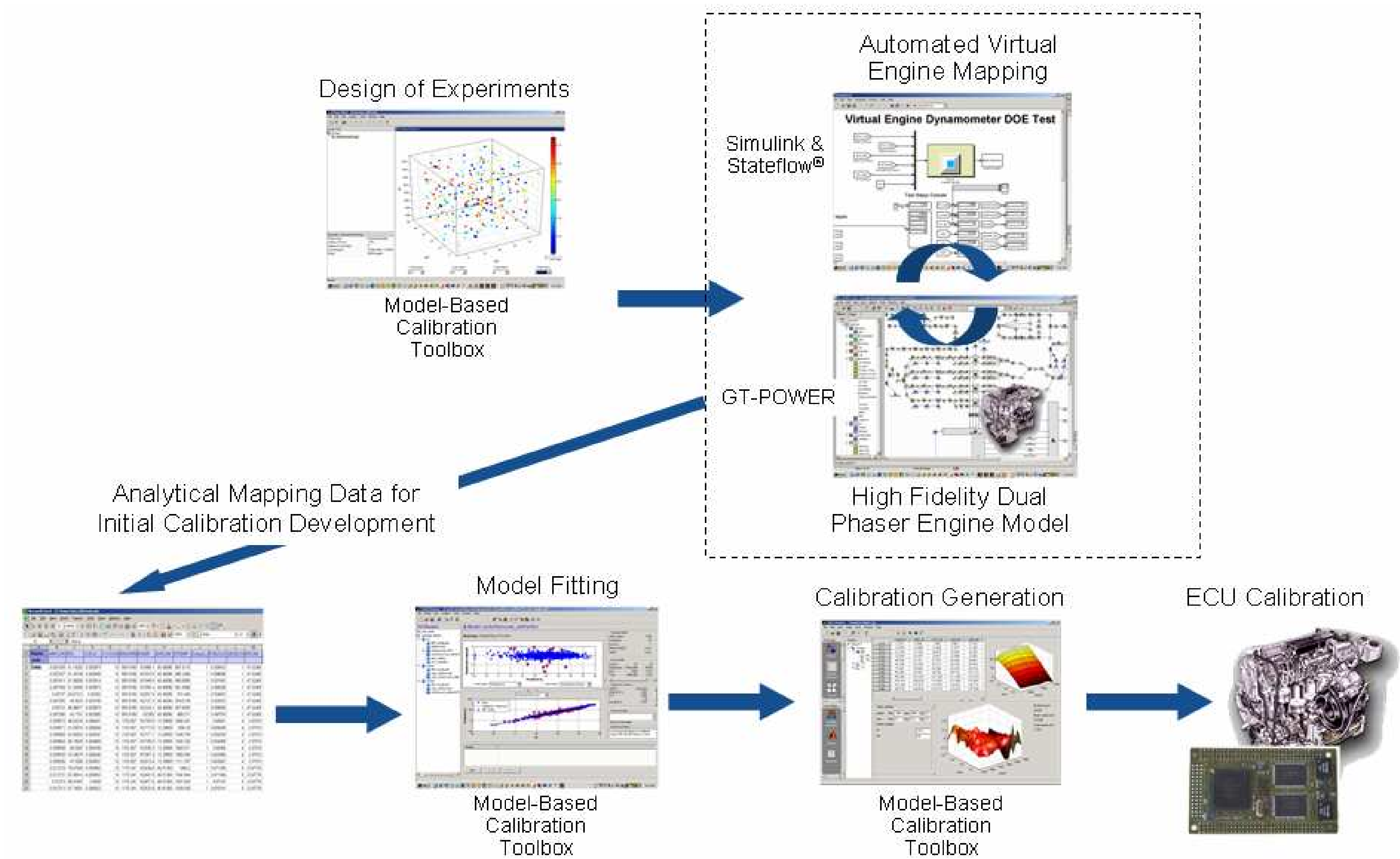
- Derive extensive calibration tables for control
  - Determine experiments and their density
  - Obtain optimal representation





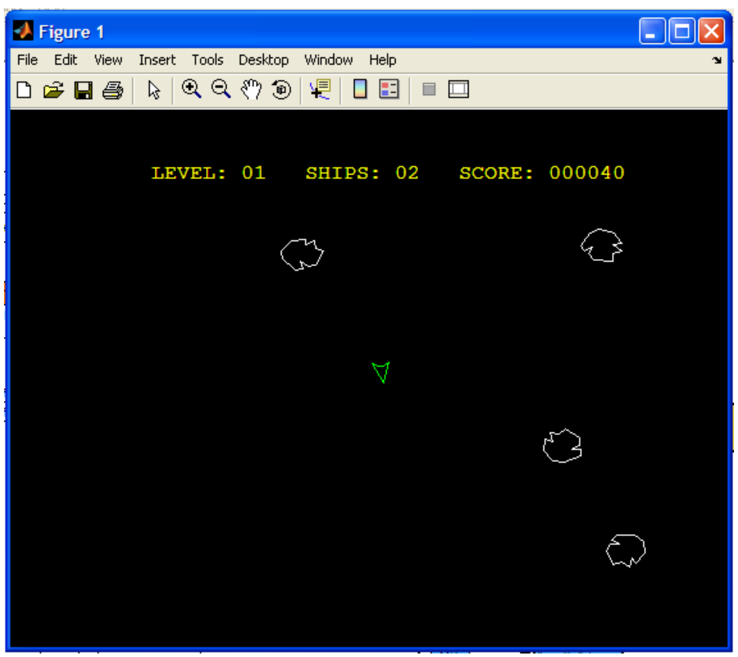
- Typically towards the end of the design process
  - Desire to implement as a concurrent activity
  - Apply computational models!

Model-Based Engine Calibration



Accelerated Simulation

- Compiled simulation model
  - M code in Simulink?
  - Embedded MATLAB block!
- Asteroids in Simulink



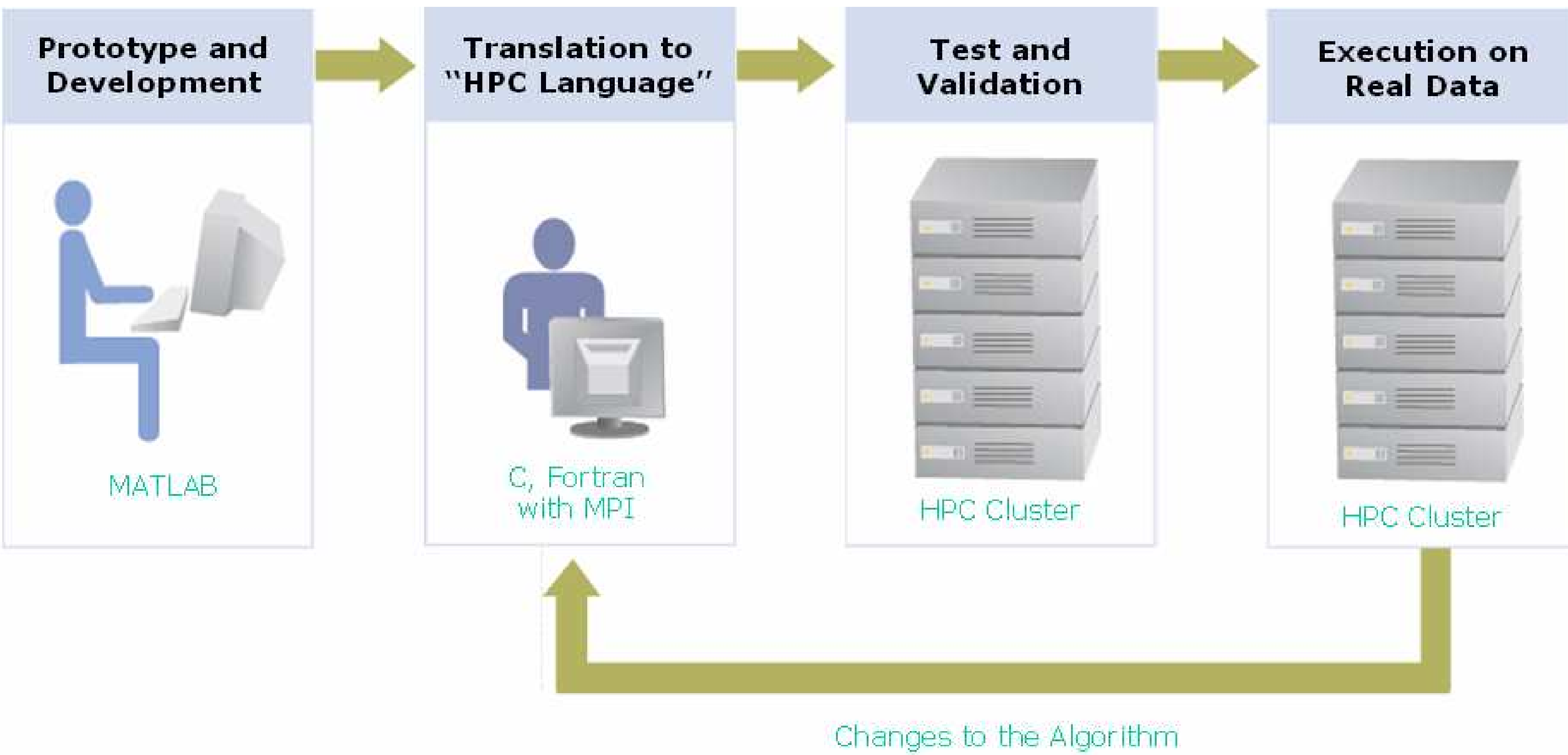

```
% 1. Compute Phi, Q, and R
Phi = [1 deltat 0 0; 0 1 0 0 ; 0 0 1 deltat; 0
Q =  diag([0 .005 0 .005]);
R =  diag([300^2 0.001^2]);

% 2. Propagate the covariance matrix:
P = Phi*P*Phi' + Q;

% 3. Propagate the track estimate:
xhat = Phi*xhat;

% 4 a). Compute observation estimates:
Rangehat = sqrt(xhat(1)^2+xhat(3)^2);
Bearinghat = atan2(xhat(3),xhat(1));

% 4 b). Compute observation vector y and linea
yhat =  [Rangehat;
         Bearinghat];
M = [ cos(Bearinghat)          0 sin(Bearingha
     -sin(Bearinghat)/Rangehat 0 cos(Bearingha

% 4 c).  Compute residual (Estimation Error)
residual = meas - yhat;

% 5. Compute Kalman Gain:
W = P*M'*inv(M*P*M'+ R);

% 6. Update estimate
```

Distributed Simulation

- The calibration process is ‘embarrassingly parallel’
  - Large number of independent experiments
  - Distributed approach!
- Convenient and efficient implementation
  - Powerful language constructs desired
    - Dynamic typing
    - …

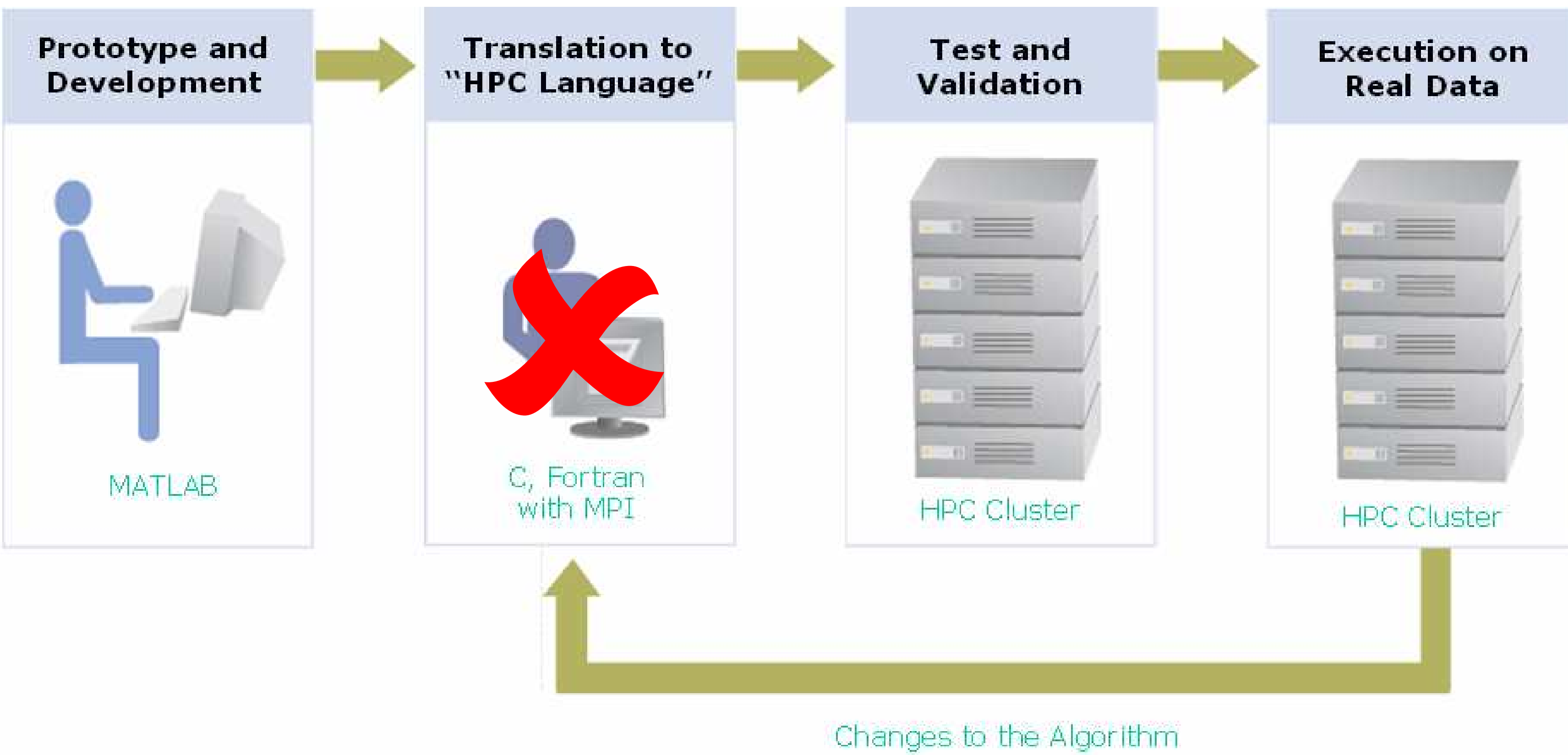
Conventional High Performance Computing Workflow

- Without the distributed computing tools



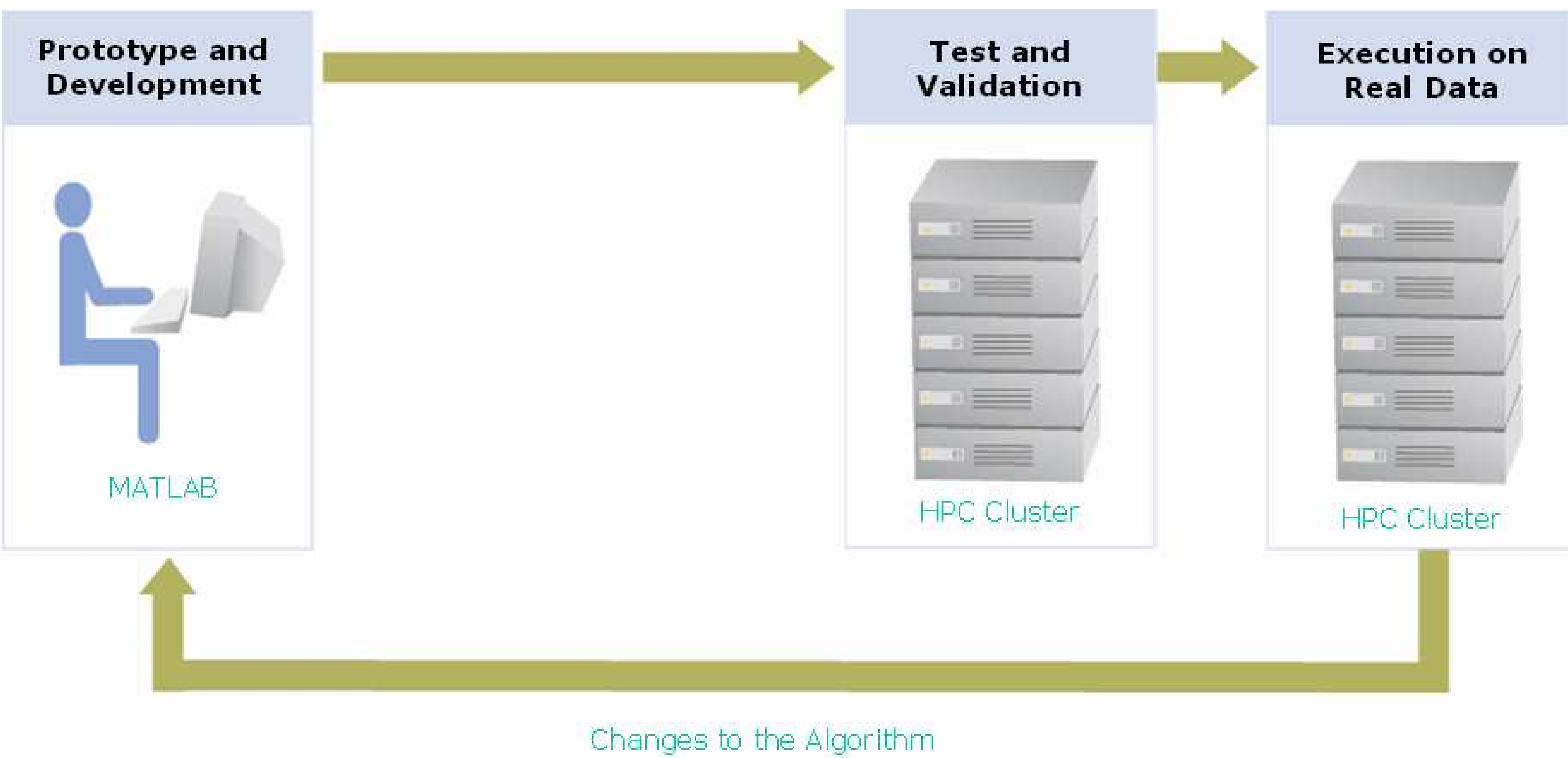
Conventional High Performance Computing Workflow

- Without the distributed computing tools



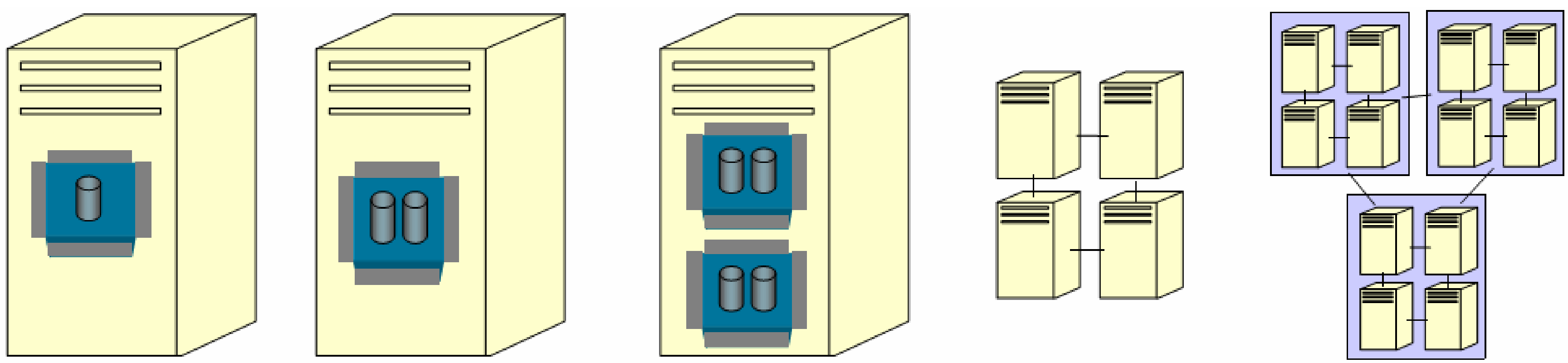
High *Productivity* Computing Workflow

- Using distributed computing tools



Enabling the High Productivity Workflow

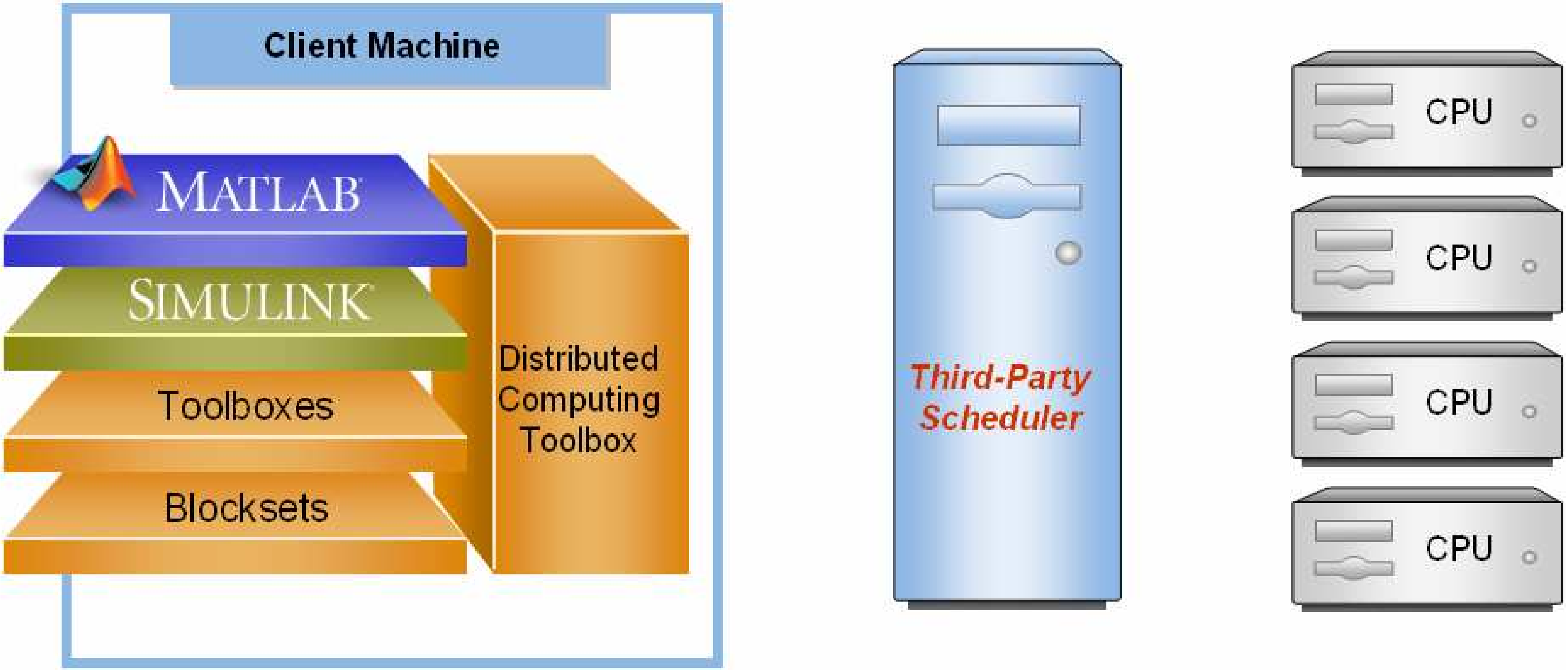
- Availability of hardware with super computing power from our desktop
  - Multi-core
  - Multi-processor
  - Cluster
  - Grid



- Ability to take advantage of this for highly productive computational design

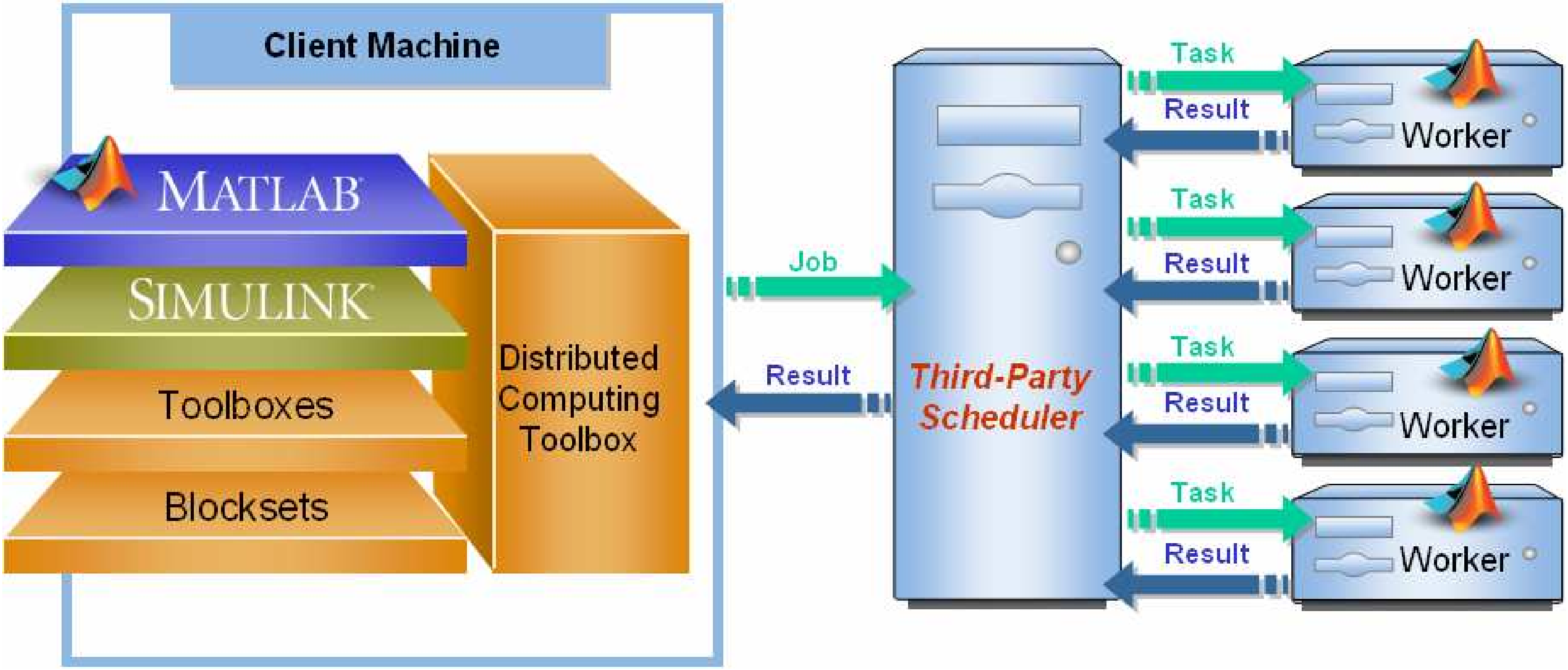
How did The MathWorks do it?

- Pluggable scheduler



How did The MathWorks do it?

- Pluggable scheduler



Benefit of Pluggable Scheduler

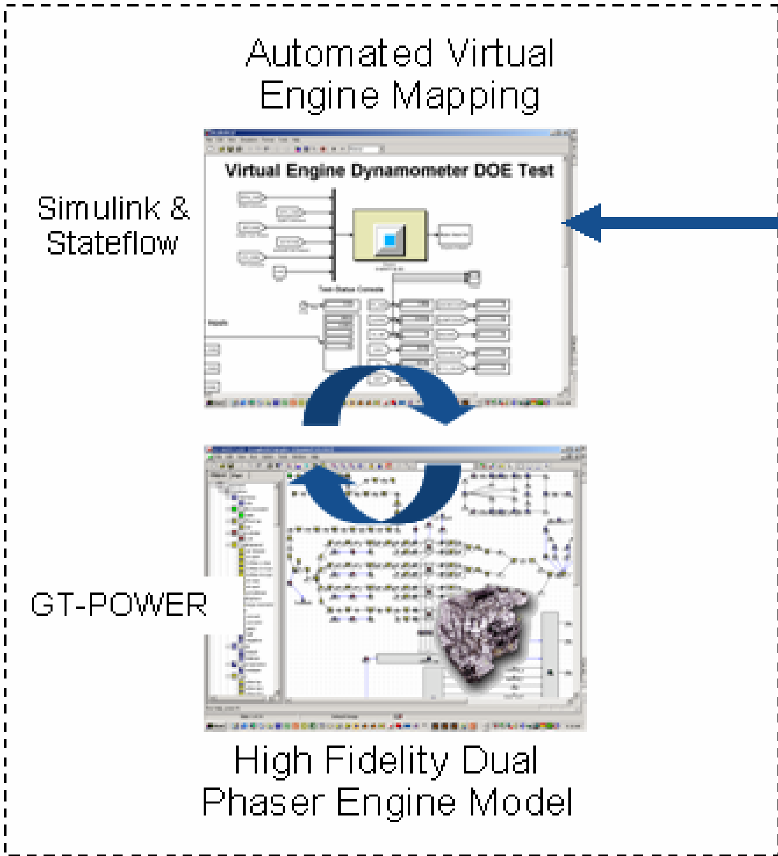
- Integration of distributed computing tools with existing cluster environment of customer
- Heterogeneous clusters
- MATLAB and other applications run on the same cluster
  - Increased throughput
  - Reduced cost of ownership
- Exploit unique capabilities of schedulers
  - Advanced scheduling
  - Batch workflow support
  - Utilization and performance increase
  - Scalability, reliability, and security

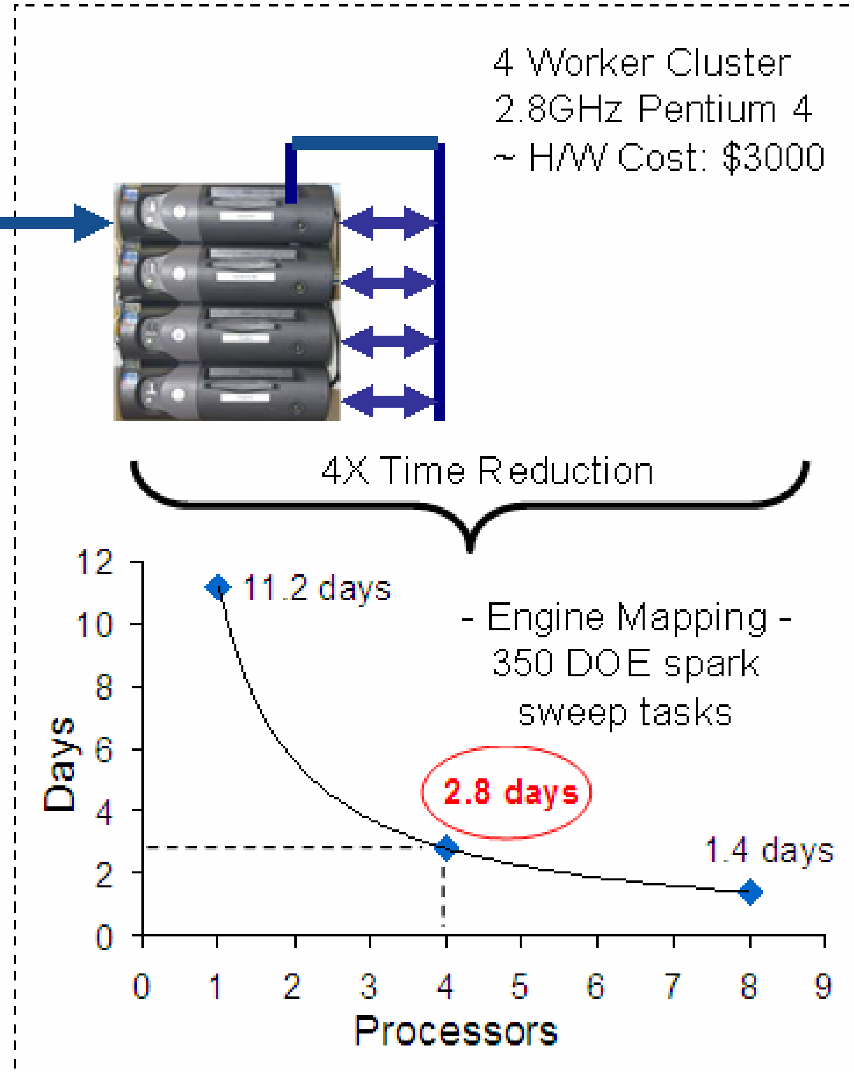
Object-oriented and Functional Interface

```
>> % Find a job manager and create a job
>> jm = findResource('jobmanager', 'Name', 'MyJobManager');
>> job1 = createJob(jm);
>>
>> % Create tasks for job1
>> createTask(job1, @rand, 1, {1});
>> createTask(job1, @rand, 1, {2});
>> createTask(job1, @rand, 1, {3});
>>
>> % Submit job1 and wait for it to finish
>> submit(job1);
>> waitForState(job1,'finished');
>>
>> % Get results of job1 and display them
>> results = getAllOutputArguments(job1);
>> for i = 1:3
disp(results{i})
end
    0.9501

    0.2311    0.4860
    0.6068    0.8913

    0.7621    0.8214    0.7919
    0.4565    0.4447    0.9218
    0.0185    0.6154    0.7382
```

```
>> % Call distributed version of FEVAL function
>> results = dfeval(@rand, {1 ; 2 ; 3});
>>
>> % Display results
>> for i = 1:3
disp(results{i})
end
    0.1763

    0.4057    0.9169
    0.9355    0.4103

    0.8936    0.8132    0.2028
    0.0579    0.0099    0.1987
    0.3529    0.1389    0.6038
```

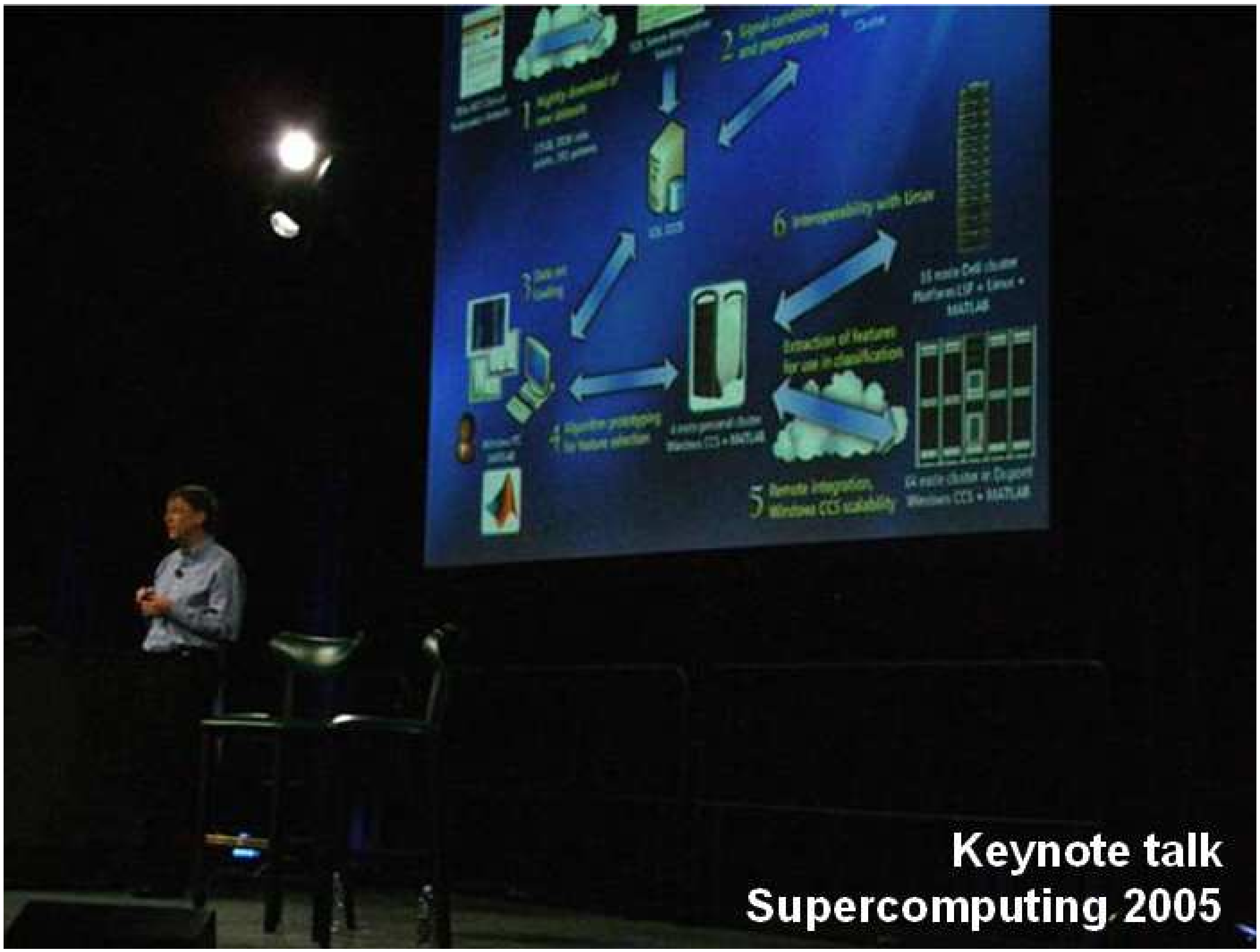
Distributed Simulation and GT-Power





Resource Management Trends

- Microsoft Cluster Computing Server

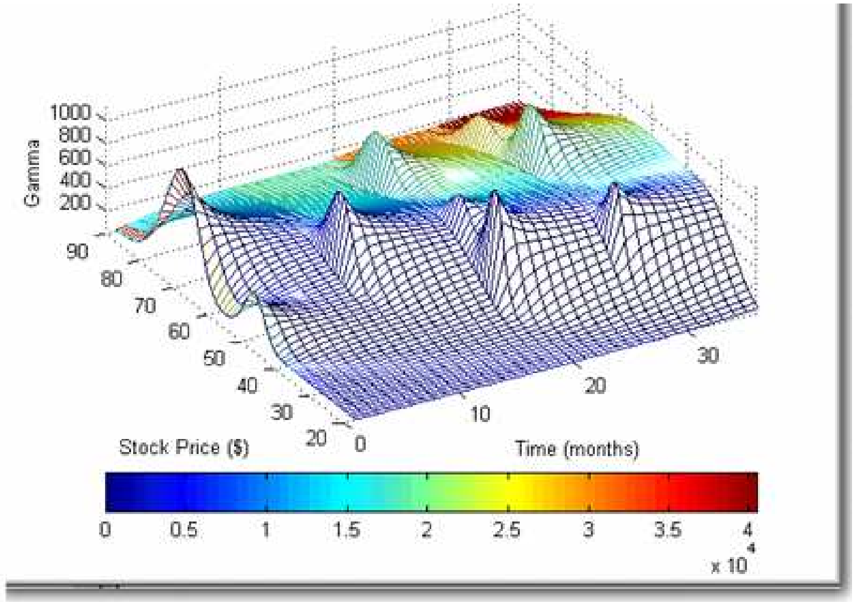


Keynote talk
Supercomputing 2005

Other Applications — EIM Group

MATLAB Based System measures overall risk of portfolios.

Simulates hedge fund realizations using MATLAB distributed computing tools.



"We reduced execution time from 6 to 1.5 hours using 3 dual-processor machines"

Dr. Stéphane Daul, EIM Group
Switzerland

Why no parallel MATLAB before?

- Cleve’s Corner in 1995
  - It did not make *business* sense at the time…

Cleve’s Corner

Why there isn’t a parallel MATLAB

Our experience has made us skeptical

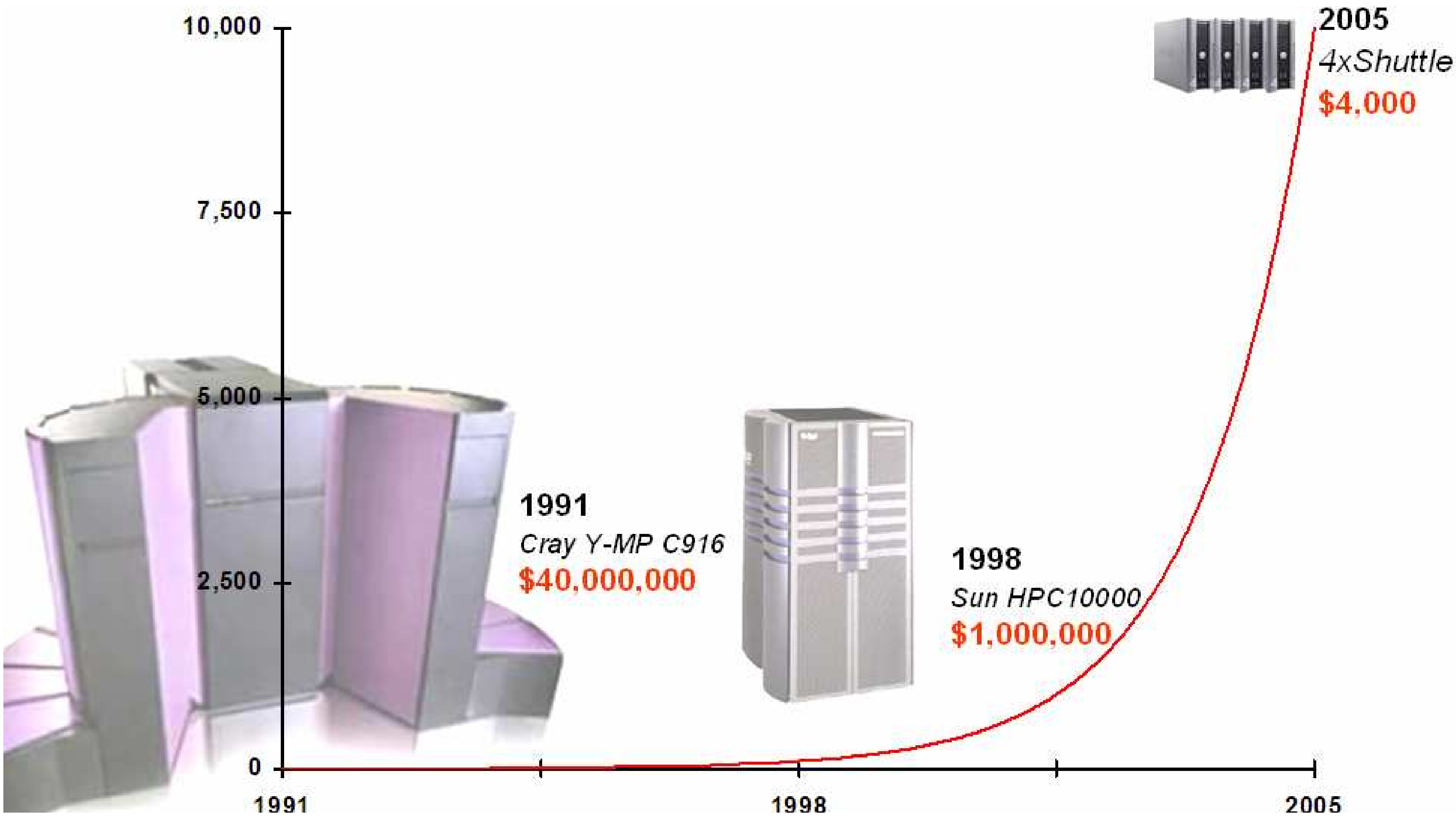
by Cleve Moler

There actually have been a few experimental versions of MATLAB for parallel computers. None of them has been effective enough to justify development beyond the experimental prototype. But we have learned enough from these experiences to make us skeptical

MATLAB is a lot bigger, and parallel computers a
But distributed memory is still a fundamental dif
MATLAB’s most attractive features is its memory
are no declarations or allocations—it is all handle
ically. The key question is: *Where are the matrices*

Market Trend

- The 10 GFLOP Personal Computer!



Recap

- High *productivity* computing

Conclusions

- Hardware
  - Less expensive
  - Networked
- Software infrastructure
  - Operating system support
  - Scheduling software
- Software applications
  - Inherent support for distributed computing
  - Think matrices not messages!

Interactive programming - re-use of existing applications
Distributed/parallel engineering software tools
Schedulers Standard OSs Libraries
Low-cost Hardware