Large-Scale and Supercomputing

Tuomas Eerola

August 16, 2011

Outline

9:00-10:00 **Introduction to Techila**

- **Why was Techila developed?**
- **What is the position of Techila compared to other solutions?**
- **Evolution of the Techila technology.**
- **Where should I use Techila?**

10:00-11:00 **Use cases**

- **Physics**
- **Optimization**
- **Life Sciences**
- **Economics**

INTRODUCTION

Customer problem

” …Lack of computing capacity has been a big problem and a bottleneck in research projects…”

”…Energy efficiency is becoming an increasingly important item in the future…

…Supercomputers have a high energy consuption. In addition to this, supercomputers require efficient and remarkably expensive cooling. The Techila solution enables reconsidering expensive supercomputer investments...”

TECHiLA
TECHNOLOGIES

Rainer Wehkamp, CEO, Techila Technologies Ltd.

In my opinion, founding a company should always be based on solving a genuine customer problem. I found this problem close by, at the university I was attending, when I noticed that lack of computing capacity was a bottleneck for many research projects. The market and technology studies carried out with funding from TEKES, the Finnish Funding Agency for Technology and Innovation, showed that this was an extensive global phenomenon, and that there were not solid technology solutions available. We understood that our innovation could be the answer to the critical unavailability of computing resources; Techila Technologies was founded in 2006 to solve a real-life problem.

The Techila innovation is a highly automated distributed computing system that takes the idle capacity of computers and servers into utilization. Our solution is unique, even on the architecture level. From the very beginning, the foundation of our product development has been to understand the needs of our customers. The more impossible the requirements of our end-users seem to be, the harder we worked to solve them. To be able to solve a problem that at first seems insurmountable not only secures customer satisfaction and trust but also creates a clear competitive edge on the global market. Working together with demanding end-users, our technology has matured into a product which we, and our clients, consider a world-class solution.

The biggest hurdle on our path in the four-year history of Techila has been to achieve credibility. It is typical that radical innovations, such as ours, are greeted with skepticism and their feasibility and investment potential are questioned. When our clients do witness that our technology works, their disbelief turns into belief. One of our clients in the financial sector conducted a thorough test on our system before deciding to invest in our solution. They concluded in surprise that our product delivered everything we had promised. With our current clients and references, bringing our technology to new markets has eased considerably.

Private funding did not really have a lot of faith in us before, but the Finnish governmental funding agencies such as TEKES, Finnvera, and Avera have shown their trust, and helped us over the financial valley of death that has claimed many good innovations. We are deeply grateful for this sign of trust. With the same humble gratitude, but also with pride, we now accept the INNOFINLAND Prize. The prize will certainly grant Techila more of the credibility we have sought, and it will strengthen our position as we head to the international market.

Rector Markku Kivikoski of the Tampere University of Technology
TECHILA GRID HAS A BRIGHT FUTURE IN THE WORLD OF RESEARCH

Half of the departments in Tampere University of Technology are extremely high performance computing intensive areas due to their operations. The lack of computing capacity has been a big problem and a bottleneck for research projects. There has not been funding available for consistent IT upgrades, and in some cases research groups have had to use their time sourcing the required computing capacity from outside the university.

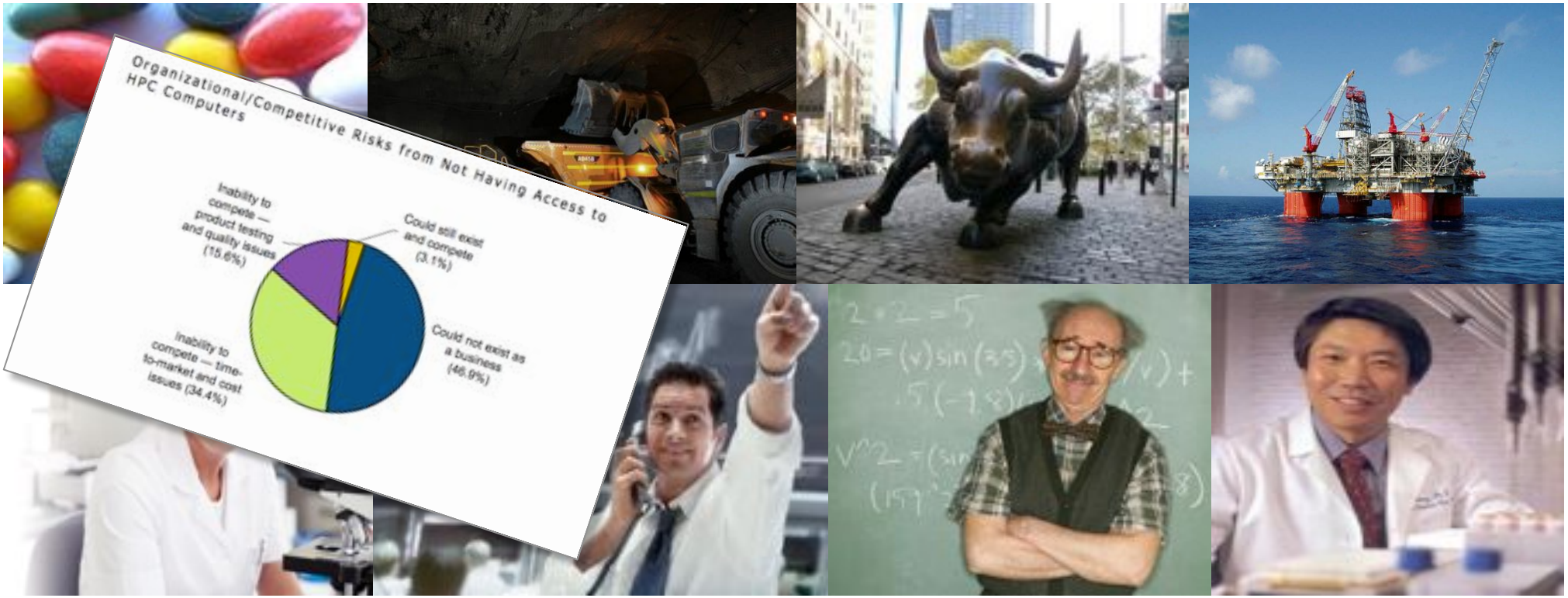
Techila Grid solution has changed this situation radically. The Techila solution has been able to convince and empower our researchers, as computing projects that used to take days can now be run in hours. Techila has been able to remove the computing bottleneck. We have added both PCs and server clusters to the grid. This has unleashed their idle capacity at a very reasonable cost. At the same time, distributed computing has freed native supercomputer capacity for projects, which can not be completed without these resources.

Energy efficiency is becoming an increasingly important item for the future. The grid uses PCs that would be running around the clock anyway, regardless of their utilization. Supercomputers have a high energy consumption. In addition to this, supercomputers require efficient and remarkably expensive cooling. The Techila solution enables reconsidering expensive supercomputer investments.

We at the Tampere University of Technology have always encouraged our students to engage in entrepreneurship. Techila Technologies is a fine example of this. We are proud to be pioneers in utilizing a solution developed by our own alumni. Techila's solution is a great benefit to our own research, and I am sure it will benefit the whole academia in the future.

The importance of computing to customers

- **97% of tier 1 industrial firms that have adopted HPC (High-Performance Computing) consider it indispensable for their ability to compete and survive (Council on Competitiveness Study of U.S. Industrial HPC Users, 2004)**



Pain-points still standing still

- Tools are hard to use and manage
- Systems are complex
- Software does not scale
- …

IDC HPC Market Update 06-2011

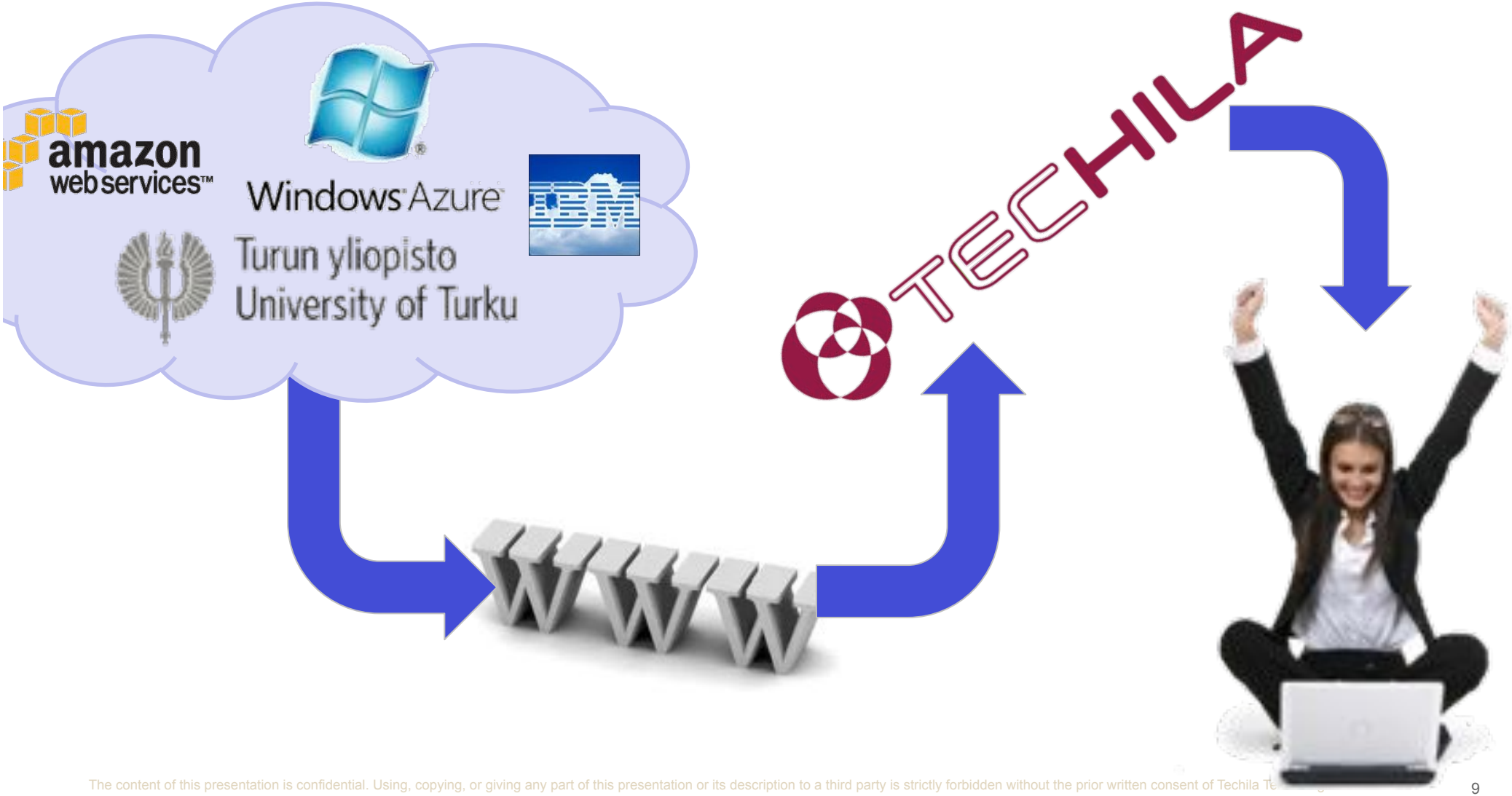
Enacts.org 2004

Insanity – continuing to do things the way they have always been done and expecting the results to be significantly different".

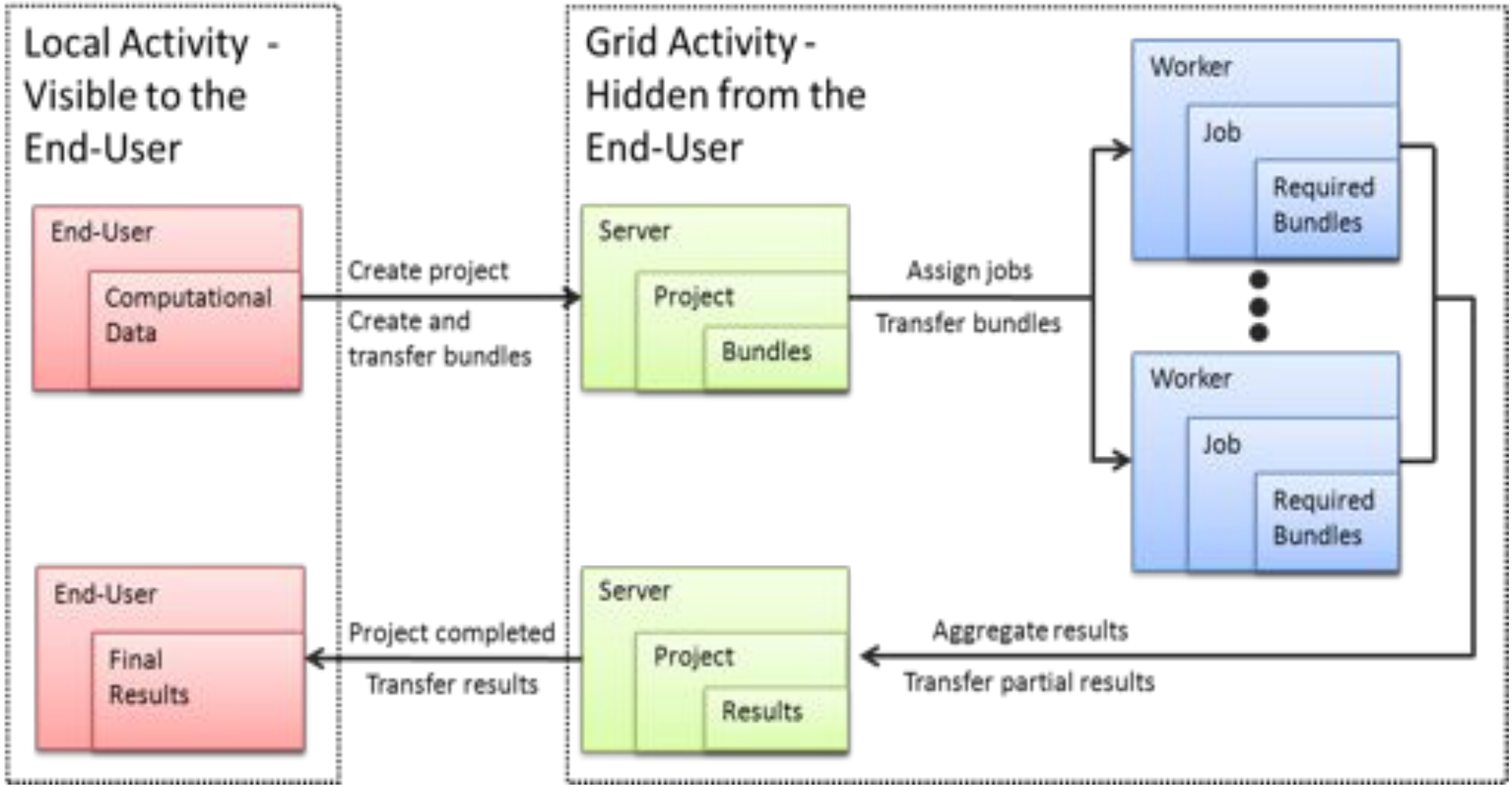
- Dr. W. Edwards Deming, the 'father of quality' -

Electricity

Computing Capacity



Efficient distributed computing is not easy to do



Ease-of-Use is the #1 item

- Techila has been designed for end-users in co-operation with end-users. This is an on-going process.
- Scientists do not want to be system administrators.

All feedback and improvement ideas will be appreciated

HISTORY

Key milestones

Year	Features	New languages
2006	First version	Java
	Distribution required writing ”only” a splitter, a result handler, an I/O wrapper for Workers and some PHP for the user interface.	MATLAB
2007	SGC (Simple Grid Client)	
	Joint-development with early adopters	
2008	GMK (Grid Management Kit) PEACH helper function	C/ C++
	Integration API Snapshotting	APL
2009	Streaming Callbacks	R, Perl
		Python
2010	Job-input files	
		Command Line Interface
2011	GridFor	

2006 – Java

Worker Module Activator

```
public class Activator implements BundleActivator {

    public void start(BundleContext bc) {}
    public void stop(BundleContext bc) {}

}
```

Module

```
public class Implementation extends
techilaclient.ModuleImpl implements Module {

                Vector cmdparams;
                static public LogRef log;
                BundleContext bc;
                String workdir;

                boolean interrupted=false;

                public void interrupt() {
                                interrupted=true;
                }

                publicImplementation(BundleContextbc){
                                this.bc=bc;
                                Implementation.log=newLogRef(bc);
                }

                public Implementation() {}
                …
```

Splitter Service

```
public class SplitterServiceSampleextends
ServiceBase implements SplitterService {

    public SplitterServiceSample(BundleContext
bc) {
        this.bc = bc;
    }

  public DatabaseService db() {
     return ServiceHandlers.db(bc);
  }
…
```

Splitter

```
public class Activator implements
BundleActivator {

  ServiceReference sr;

  public void start(BundleContext bc) {

    try {

    SplitterServiceSample ds=new
SplitterServiceSample(bc);

    Hashtable props = new Hashtable();
    props.put("service.pid", this.getClass
().getName().replace(".Activator", ""));
    props.put("splitter.categoryname",
"techilaclient.mod.shared.Sample");

    ServiceRegistration sreg
…
```

Result Handler Activator

```
public class Activator implements BundleActivator {

  ServiceReference sr;

  public void start(BundleContext bc) {

    try {

    ResultServiceLindroos ds=new ResultServiceLindroos
(bc);

    Hashtable props = new Hashtable();
…
```

Result Service

```
public class ResultServiceSample extends ServiceBase
implements ResultService {

    public ResultServiceSample(BundleContext bc) {
        this.bc = bc;
    }

  String dirname="/mnt/results";

  private DatabaseService db() {
        return ServiceHandlers.db(bc);
  }

  private class ResultFilenameFilter implements
FilenameFilter {

    String regexp;
…
```

2008 - PEACH

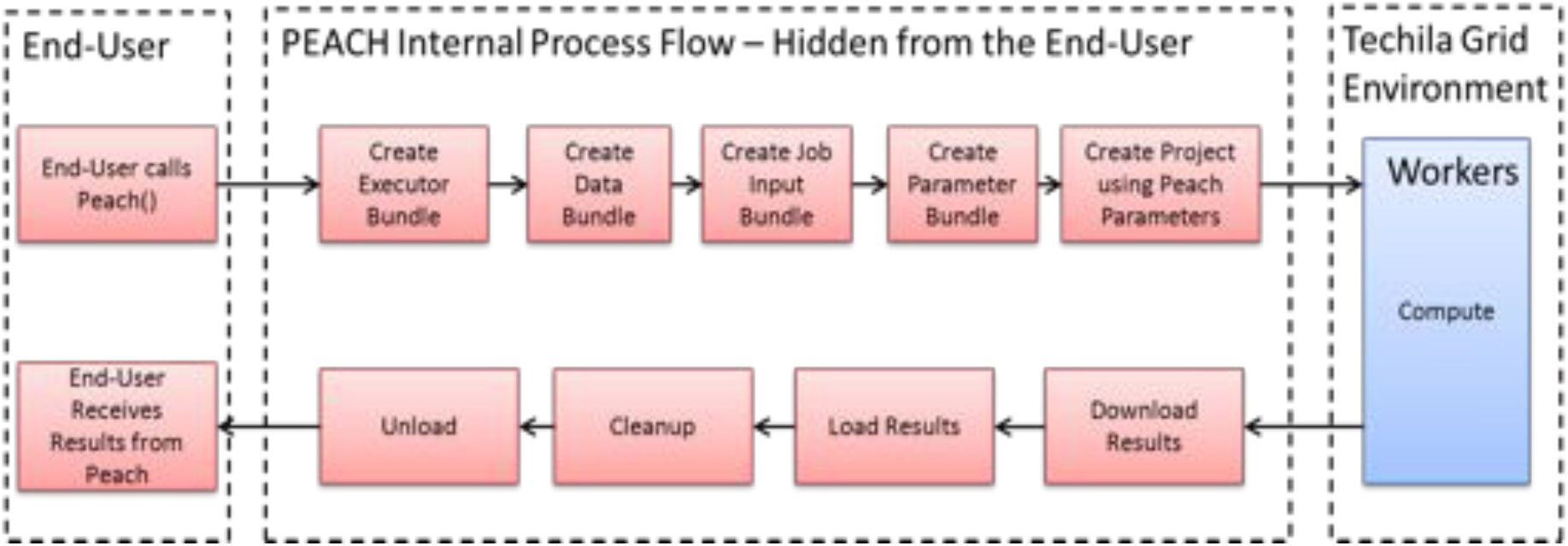
MATLAB as frontend

```
outputfile = [ 'outputdata/result_' strrep(strrep(datestr(now), ' ', '_'), ':', '') '.txt' ];

% Create peachvector of the jobs
header=1;
peachVector={};
for x=1:xslice:maxx
    xend=min(x+xslice-1,maxx);
    for y=1:yslice:maxy
        yend=min(y+yslice-1,maxy);
        peachVector=[ peachVector {{x, xend, y, yend, centery, header}} ];
        centery = centery + deltacentery;
        header=0;
    end
end

% Computes the project in Techila using given binaries.
files=peach('graphene', ... % Name for the project type, informative only
    {'<param1>', '<param2>', '<param3>', '<param4>', centerx, '<param5>', dividerx, dividery, '<param6>', potfile, datafile, 'resultfile', satifile}, ...
    {{potfile}, {datafile, satifile}}, ... % Files to be transferred to the grid
    peachVector, ... % "PeachVector", tells the number of jobs (length of vector) and the replacements for '<paramx>' parameters
    'ReturnResultFiles', true, ... % Returns the names of the result files
    'Binaries', binaries, ... % Binaries to be used as executables
    'OutputFiles', {'resultfile'}); % Result files to be transferred from the workers

% Concatenates the partial result files
dirname=fileparts(outputfile);
…
fclose(fo);
fprintf('Results written into %s\n', outputfile);
```

PEACH

- **Single command interface to Techila Grid:**

peach(funcname,params,peachvector)



PEACH

Main function

Computationally intensive function

```
function [price] = run_asian_locally

S0 = linspace(45,47,9);                 %The initial stock price
sigma0 = linspace(0.35, 0.4, 9);        %The initial volatility of stock return

%The parameters of stock price diffusion
M = 20000;      %The number of trajectories
N = 365;        %The number of data points in year
nn = 1;         %The number of time steps per sample point

%The parameters of volatility diffusion
rho = -0.5;                 %The correlation between the increments of stock price and volatility
kappa = 0.1;                %The speed of revision
psi = 0.5;                  %The standard deviation of volatility

%Other parameters
E = 70;         %Exercise price
T = 1;          %Maturity time
r = 0.05;       %Interest rate

price = zeros(length(S0),length(sigma0));

for x=1:length(S0)
    for y=1:length(sigma0)
                price(y,x) = asian_montecarlo(S0(x),sigma0(y)^2,M,nn,r,N,rho,kappa,psi,E,T);
    end
end

figure(1);
[S,sigma] = meshgrid(S0, sigma0)
mesh(S,sigma,price);
xlabel('Initial stock price'),ylabel('Initial volatility'),zlabel('Price of Asian Call')
end

%Function asian computes a single option price with Monte Carlo
function [price] = asian_montecarlo(S0,v0,M,nn,r,N,rho,kappa,psi,E,T)

Dt = 1/N/nn;                %The time step size

%Preallocate memory
x = zeros(N,1); %A single log-price trajectory
C = zeros(M,1); %The value of option for each trajectory
for m=1:M
    w = randn(N,nn);
    y = randn(N,nn);
    z = rho*w+sqrt(1-rho^2)*y;
    v = v0;
    x(1) = 0;
    xx = x(1);
    for j=1:N-1
                for k=1:nn
                            xx = xx+(r-0.5*v)*Dt+sqrt(v*Dt)*w(j,k);
                            v  = max(0,v+kappa*(v0-v)*Dt ...
                            +psi*sqrt(v*Dt)*z(j,k) ...
                            +0.25*psi^2*Dt*(z(j,k)^2-1));
                end
                x(j+1) = xx;
    end
    S = S0*exp(x);
    C(m) = max(sum(S)/N-E,0);
end
price = exp(-r*T)*sum(C)/M;
end
```

2011 – GridFor

Original Code

```
...
for x=1:length(S0)
  for y=1:length(sigma0)
     price(y,x) = sample_montecarlo(S0(x),sigma0(y)^2,M,nn,r,N,rho,kappa,psi,E,T);
   end
end
...
function [price] = sample_montecarlo(S0,v0,M,nn,r,N,rho,kappa,psi,E,T)
...
```

GridFor Code

```
...
gridfor x=1:length(S0)
  gridfor y=1:length(sigma0)
     price(y,x) = sample_montecarlo(S0(x),sigma0(y)^2,M,nn,r,N,rho,kappa,psi,E,T);
  gridend
gridend
...
function [price] = sample_montecarlo(S0,v0,M,nn,r,N,rho,kappa,psi,E,T)
...
```

WHERE SHOULD I USE TECHILA

Silver bullets and a lot of hype

"There is no single development… …which by itself promises even one order of magnitude improvement within a decade in productivity, in reliability, in simplicity"

Fred Brooks, IFIP Tenth World Computing Conference, 1986

- **Still, many people act as if there would be a silver bullet, which can meet any High-Performance Computing needs**

Cloud **Grid** **Cluster** **Supercomputer**

Techila **FPGA** **Condor** **Infiniband** **Linux**

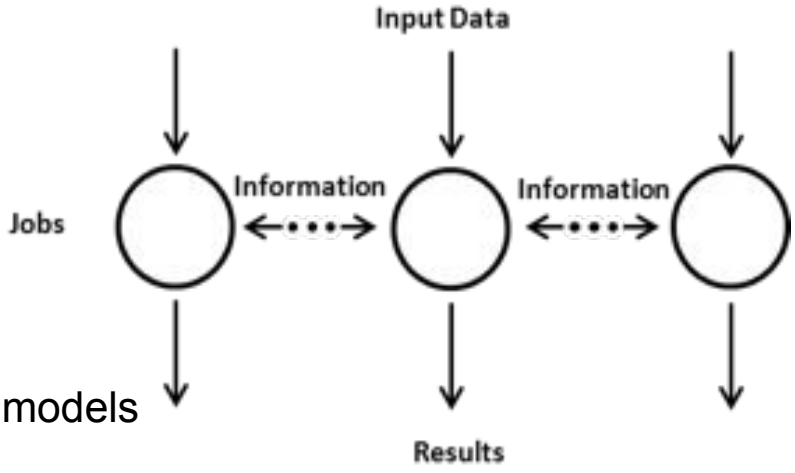
x86 **RISC**

Supermatrix **GPU** **Amazon**

Blue Gene **Petascale** **Vector** **Open Source**

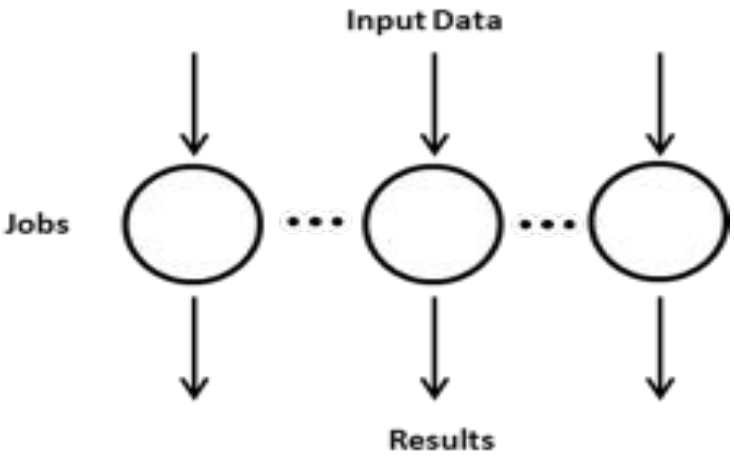
Distributed Computing Problems

- **Parallel Problems**
  - Jobs are depending on each other's states
  - Lot of communication between jobs
  - For example fluid dynamics or finite element models
  - Usually not suitable for distributed environments

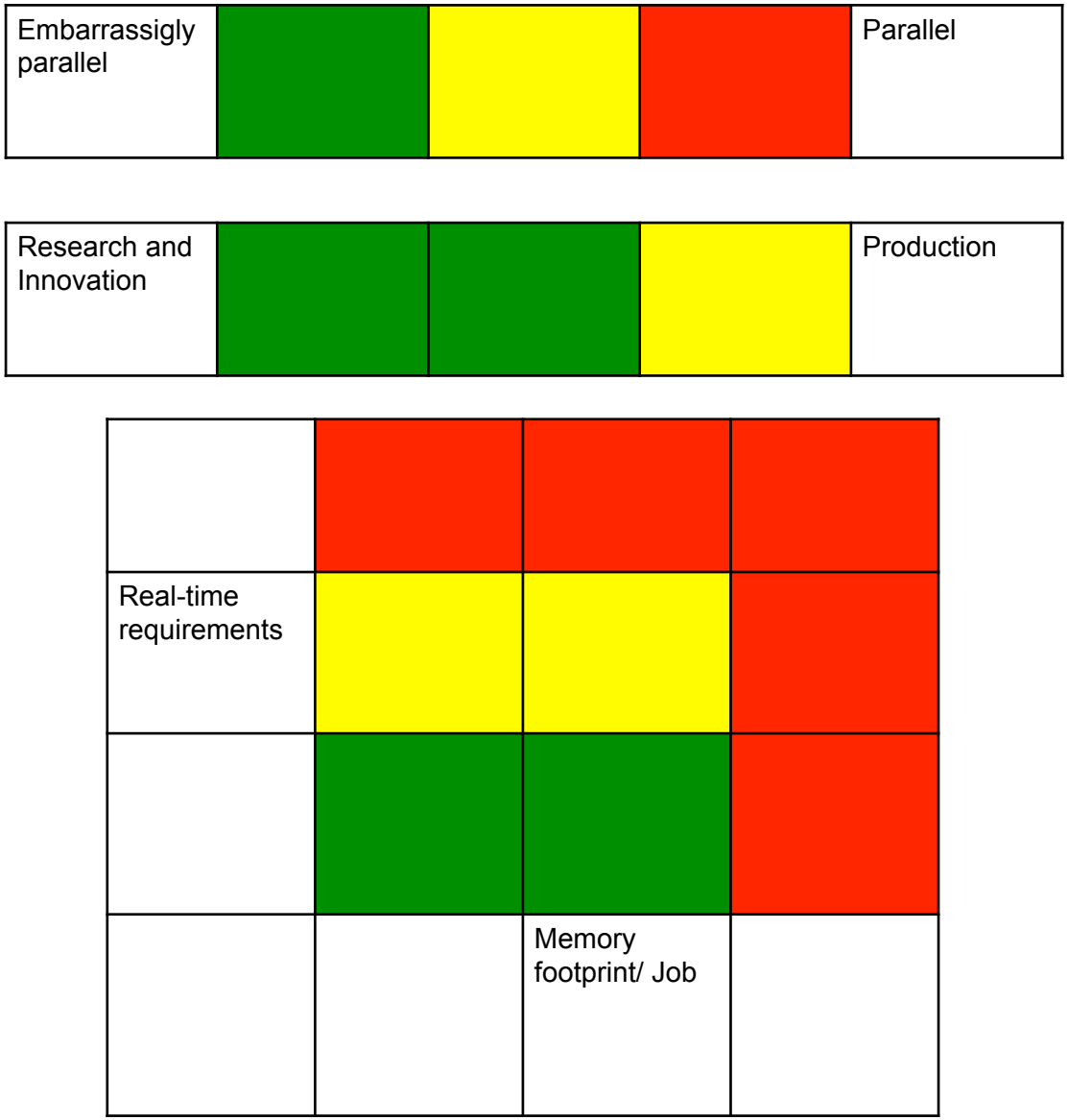


- **Embarrassingly Parallel Problems**
  - Jobs are totally independent
  - No communication between jobs
  - For example Monte Carlo

→ Distributed computing → Techila

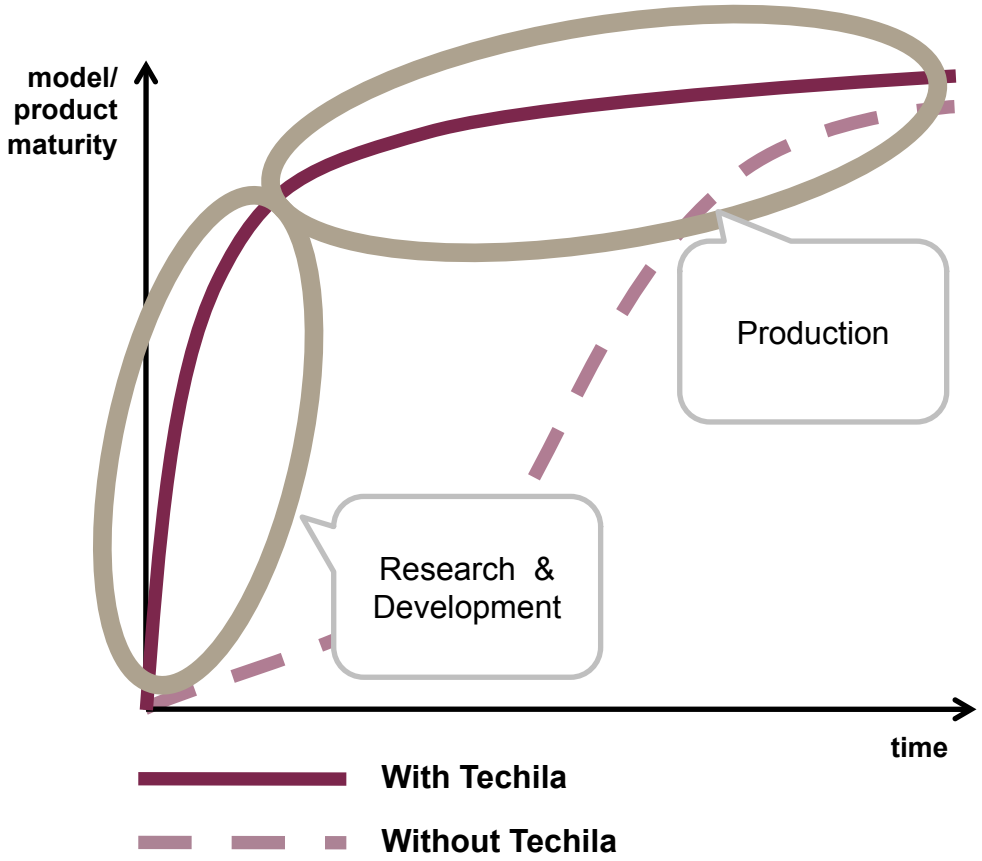


Position

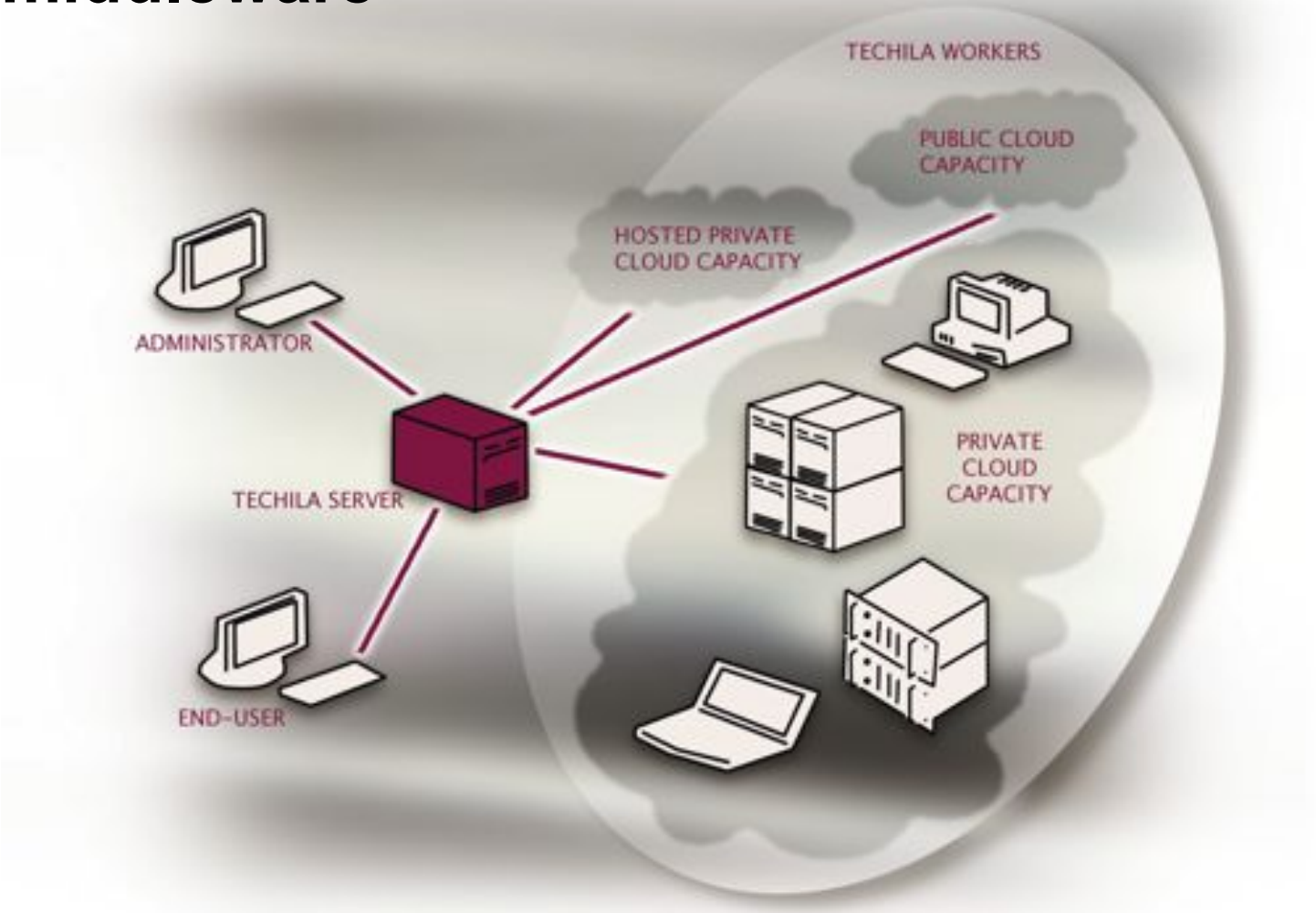


Combining the benefits of the “silver bullets”

- Techila <u>is not</u> a silver bullet. It is a great addition to your existing tool box
- Techila is the perfect solution to provide access to the power available in the “silver bullets”
- Techila is the perfect solution to horsepower the applications used for advanced research and development



Techila middleware



USE CASES

Surface science (1/2)

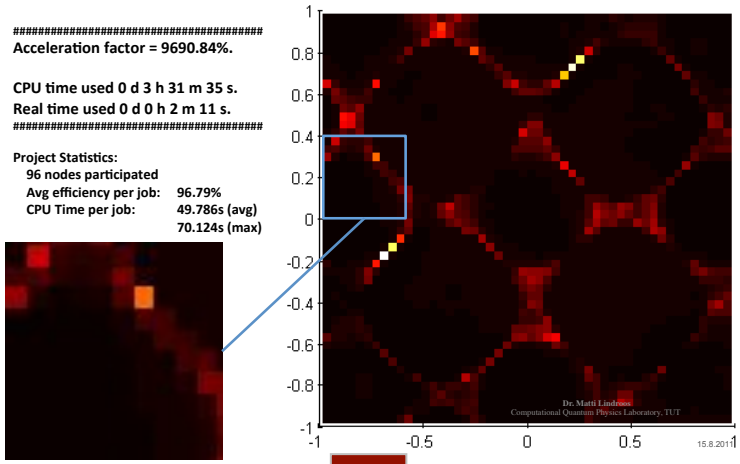
- **Surface science to determine geometrical and electronic structure of surface.**
- **Obtaining a comprehensive picture of the unconventional superconductors relies very much on electronic spectroscopes.**
- **Need to run computations with multiple variables, check the results with experimental predictions and adjust the combinations of parameters and run again.**
- **High-Performance Computers (HPC) with Message Passing Interface (MPI) was tested at NERSC**
- **Challenges:**
  - Scalability issues
  - Frustration resulting of HPC queuing and MPI's complexity

→Need for a simple, scalable and fast solution

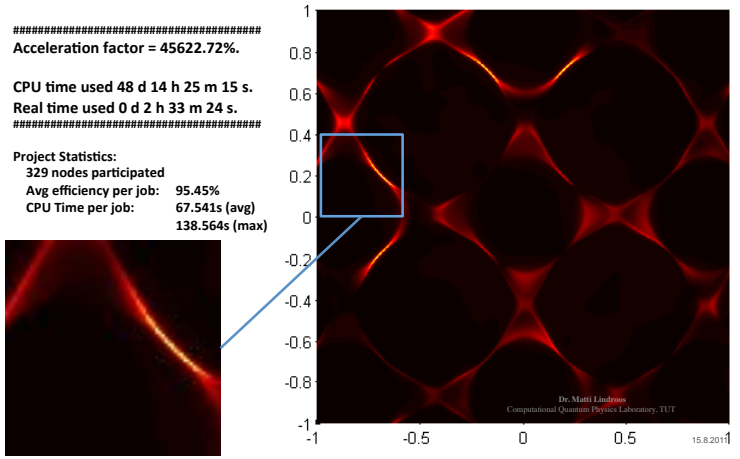
Surface science (2/2)

- **Originally a FORTRAN code.**
- **Code was changed from hardcoded to accept command line parameters.**
- **Currently using MATLAB as front-end; Preprocessing, Postprocessing**
- **Benefits:**
  - Ease of input and output handling
  - Modularity
  - Easy visualization

50x50 Matrix



800x800 Matrix

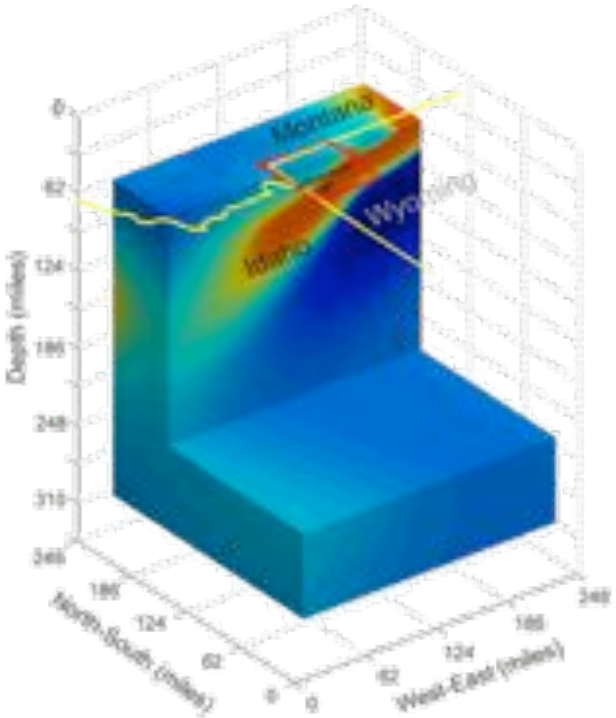


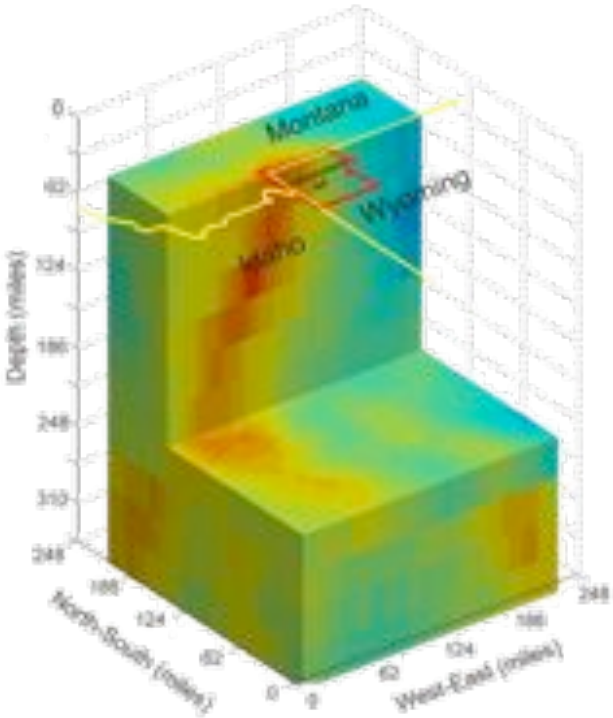
Bridge simulation

- Mitigation of Stay-Cable Vibration.
- Reduce wind hazards on long-span bridges.
- MATLAB application using Monte carlo methods

3D Inversion

- **Geophysics inverse problem.**
- **Hundreds of measurement points. Variable frequency.**
- **Distribution brings great benefits.**



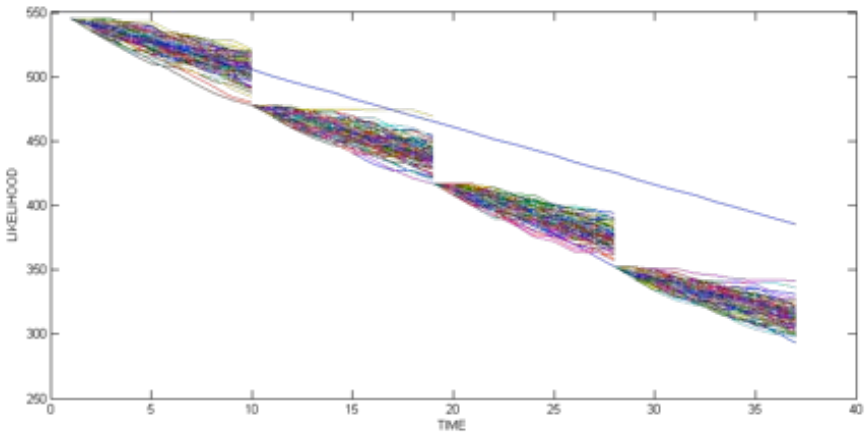


Optimization

- **Improved a deterministic non-distributable problem through stochstic approach and clever use of large pool of underutilized IT-infra**
- **Approach relies on the large number of grid nodes rather than on the actual raw computational power**
- **More nodes → better guesses → faster optimization.**
- **Estimates can now be found significantly faster by using Techila. 2 days vs. 2 months**

Techila benefits:

- Time limits (stop after specific time)
- Optimization (fastest resources automatically)
- Failure tolerance
- Ease of use

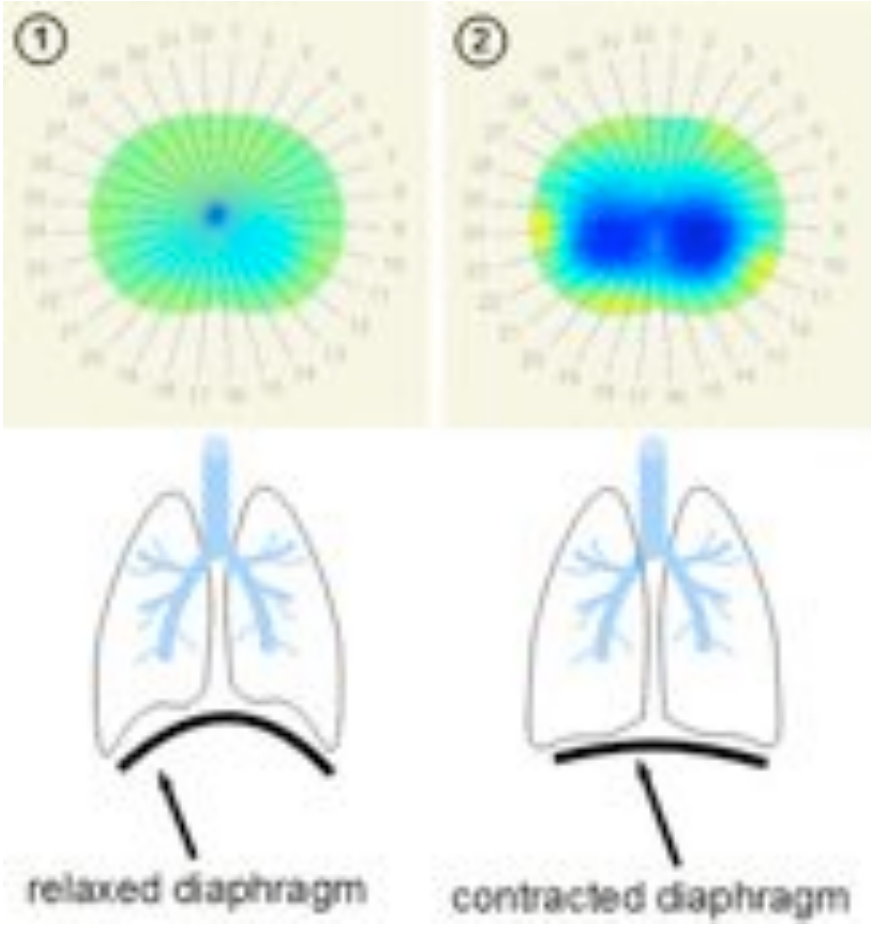


Enhanced p-value accuracy

- Bioinformatics
- Code written in R language.
- Using LME4 package, which is a package to fit linear and generalized linear mixed-effects models.
- Indepdendent slmulations. The more simulations, the more accurate p-value.
- Distributed the code using the Techila with R language integration.
- Techila took care of autonomous LME4 distribution to Workers.
- Techila provides a linear speed-up.

Medical imaging

- Inverse research applications on medical imaging.
- Creating imaging algorithms for EIT to build accurate, cheap and even portable devices that can save people's lives.
- Dentists always aim at implants that stay safely in place. The screw needed for that should be drilled in a deep as possible.



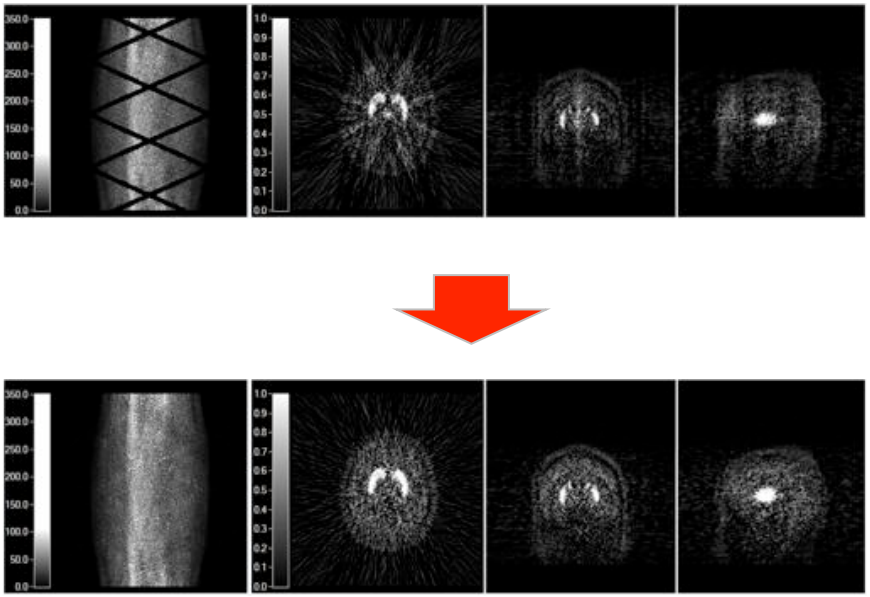
Hosted private cloud

- Image manipulation algorithms implemented with MATLAB
- Using in-house computer capacity to horsepower the application.
- Integration to a hosted product
- Secure access to on-demand computing power for peak workloads by a hosted private cloud provider.

Image reconstruction

- Scientists have had to accept inaccurate imaging in their research.
- Image enhancement applies stochastic models, which are computing-intensive
- Stochastic image enhancement has not been feasible. Takes 1 month.
- Techila supports scientific innovation.

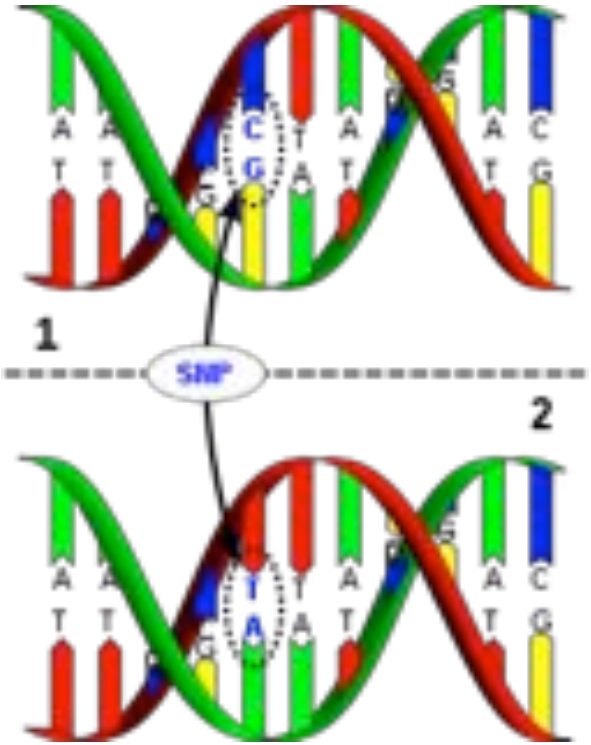
→Enables working on newest data as medical image reconstruction can be done in 5 minutes



Images: Uygar Tuna, Sari Peltonen, Ulla Ruotsalainen. Department of Signal Processing, Tampere University of Technology. Gap-Filling for the High Resolution PET Sinograms with a Dedicated DCT Domain Filter. 2009. Data acquired by the ECAT High Resolution Research Tomograph (HRRT, CTI PET Systems, Knoxville, TN, USA), located at Turku PET Centre.

Data analysis

- Systems Biology
- Pattern Discovery.
- Calculating statistics for SNP (single-nucleotide polymorphism) pairs.
- Relatively data-intensive.
- Still, 8955% speed-up with the idle capacity of ~100 computers.



Combinatory analysis

- Assessing the likelihood that a primary breast cancer tumour develops metastases.
- A lot of genes and their combinations that may contribute to the metastasis progression.
- Key challenge on finding the right combination of genes from among 25000 genes.
- Windows Azure with Techila integration used.
- 1200 Windows Azure instances running MATLAB code.
- 15-year project completed in 4,5 days.

Model calibration

- **Monte Carlo simulation has become an essential tool in financial engineering**
  - GARCH models and other volatility models
  - Exotic options
  - LIBOR models
  - Risk measures
- **Drawback: No closed-form analytical solution for the price**

→ Price available only through Monte Carlo simulation.

→Makes the model calibration computationally expensive

- **Calculating primes of the each subinterval separately, and then collectig the results enables distributed computing.**

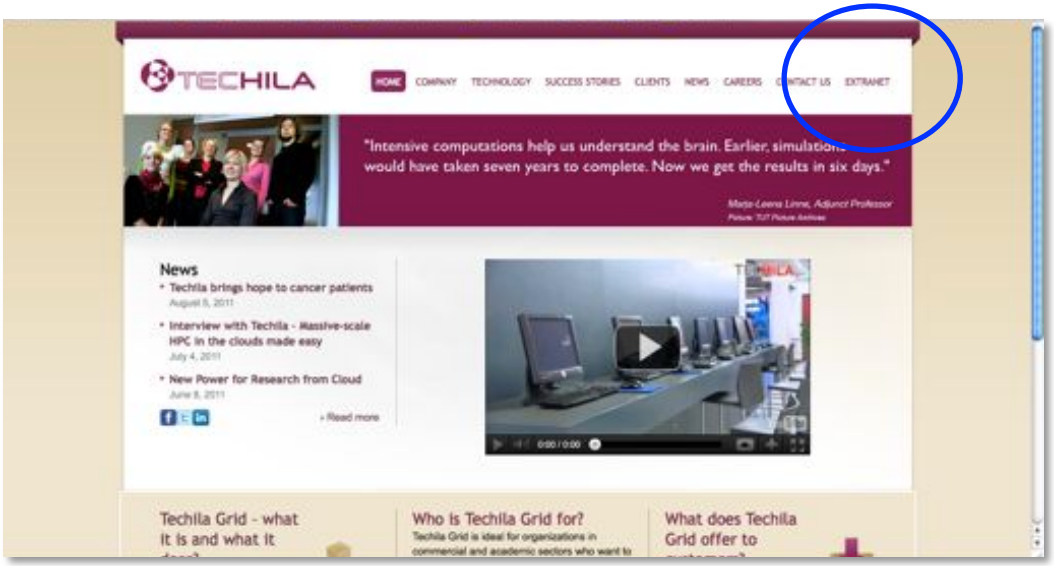
→Nearly linear speed-up

TECHILA DOCUMENTS & DOWNLOADS

Techila Extranet

Please register at Techila Extranet for developer tools and documents

www.techilatechnologies.com > Extranet > Register



Tuomas Eerola
Vice President, Partner
Techila Technologies Ltd

e: tuomas.eerola@techilatechnologies.com

t: +358 50 336 7730