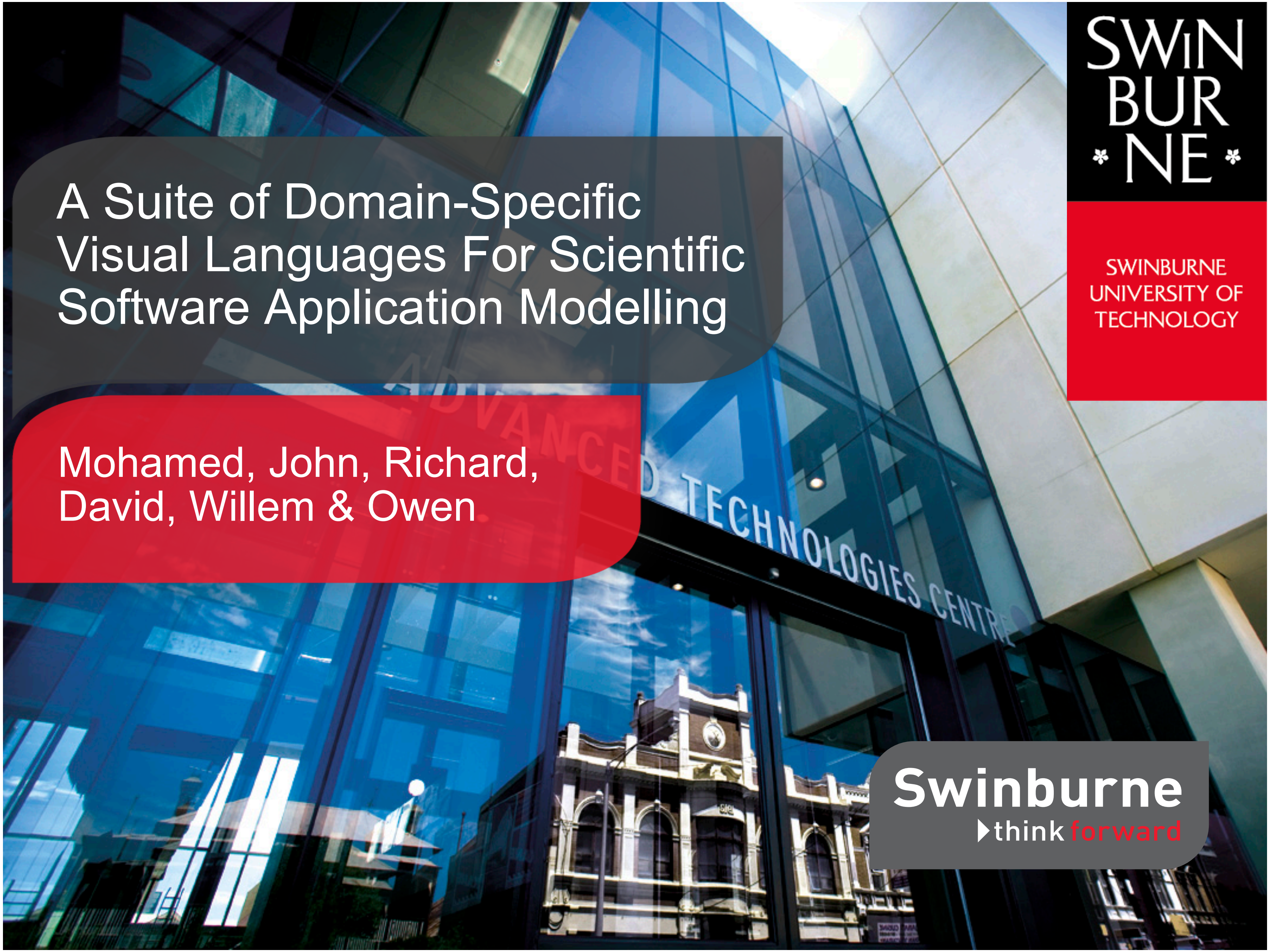


A Suite of Domain-Specific Visual Languages For Scientific Software Application Modelling

Mohamed, John, Richard, David, Willem & Owen

SWINBURNE UNIVERSITY OF TECHNOLOGY

Swinburne think forward

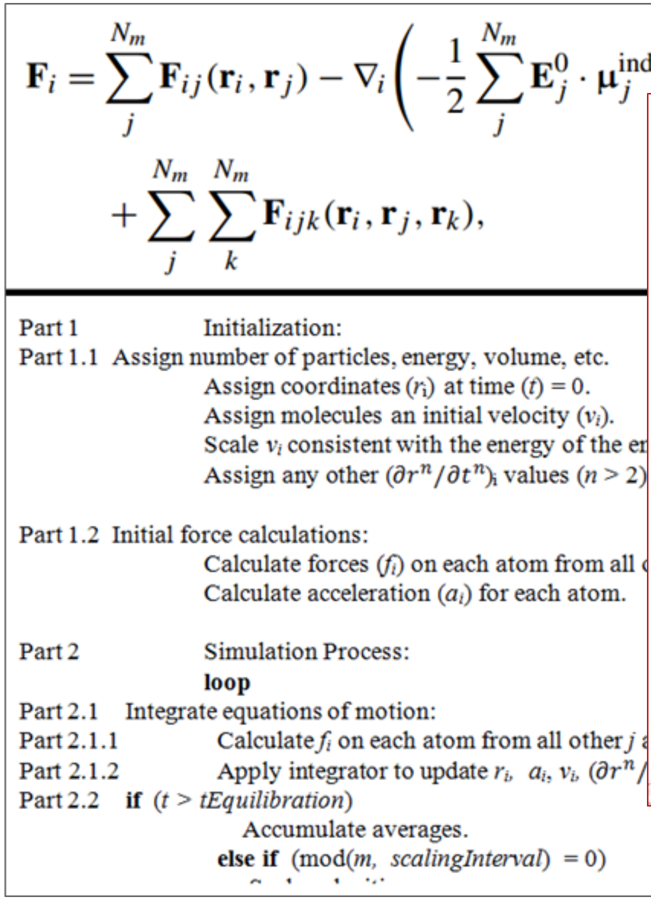
Outline

- Problem / Motivation
- Approach
- Example Usage
- Future work

Problem

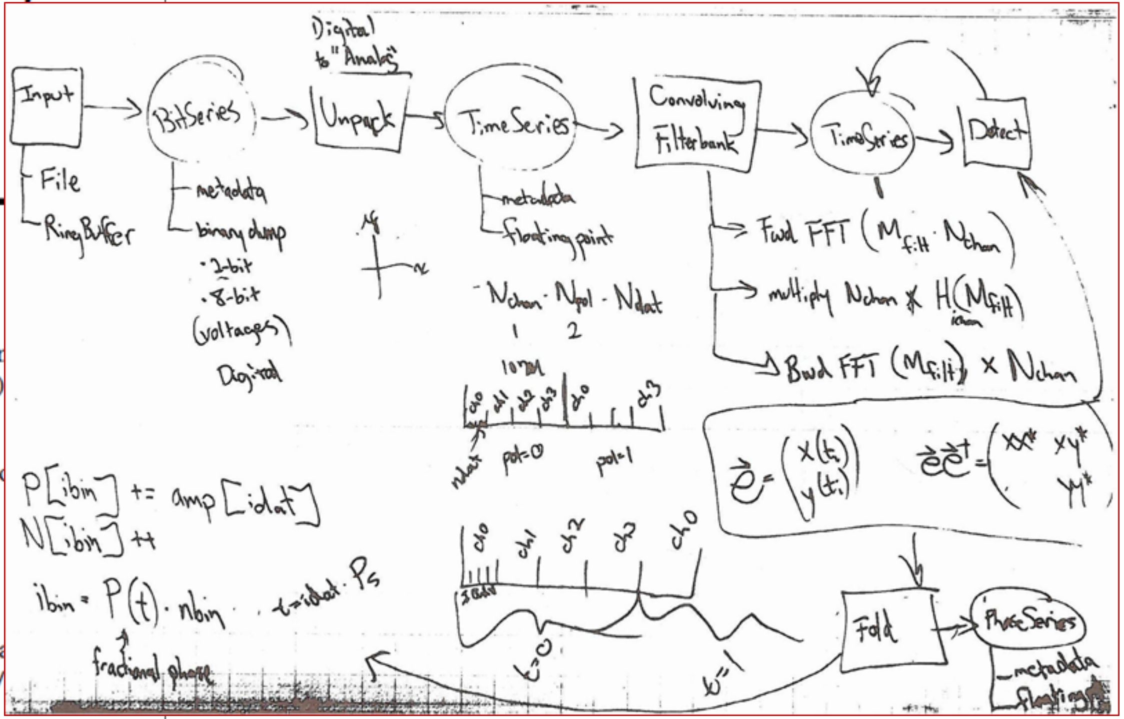
- Scientists are increasingly developing complex software for data analysis
- Most are not trained programmers
- Many are using complex software platforms and techniques e.g. distributed & parallel programming, GPUs, etc - that are hard even for experienced CS grads to use
- Approaches to address range from packaged software (Lack flexibility), DSLs (Also Flexibiity/Domain issues), programming patterns and toolkits (still complex)
- Still lack high-level, human-centric support; still really hard to develop high quality software for scientific apps

Motivation – how scientists design their applications…



$$\mathbf{F}_i = \sum_j^{N_m} \mathbf{F}_{ij}(\mathbf{r}_i,\mathbf{r}_j) - \nabla_i\left(-\frac{1}{2}\sum_j^{N_m}\mathbf{E}_j^0\cdot\mu_j^{\text{ind}}\right) + \sum_j^{N_m}\sum_k^{N_m}\mathbf{F}_{ijk}(\mathbf{r}_i,\mathbf{r}_j,\mathbf{r}_k),$$

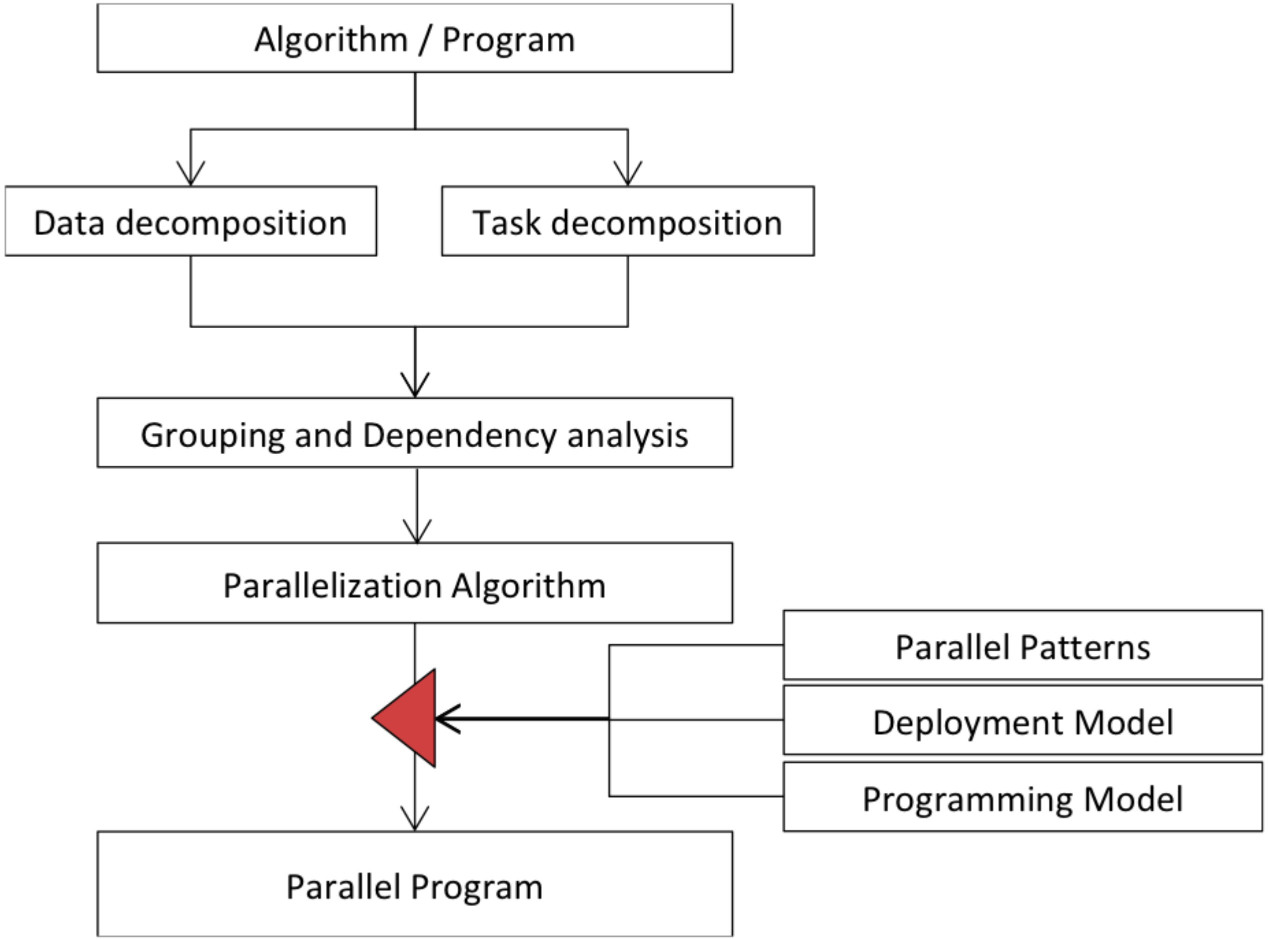
```
Part 1          Initialization:
Part 1.1  Assign number of particles, energy, volume, etc.
                Assign coordinates ($r_i$) at time ($t$) = 0.
                Assign molecules an initial velocity ($v_i$).
                Scale $v_i$ consistent with the energy of the er
                Assign any other $(\partial r^n/\partial t^n)_i$ values ($n > 2$)

Part 1.2  Initial force calculations:
                Calculate forces ($f_i$) on each atom from all o
                Calculate acceleration ($a_i$) for each atom.

Part 2          Simulation Process:
                loop
Part 2.1    Integrate equations of motion:
Part 2.1.1          Calculate $f_i$ on each atom from all other $j$ a
Part 2.1.2          Apply integrator to update $r_i$, $a_i$, $v_i$, $(\partial r^n/$
Part 2.2    if ($t$ > tEquilibration)
                    Accumulate averages.
                else if (mod($m$, scalingInterval) = 0)
```



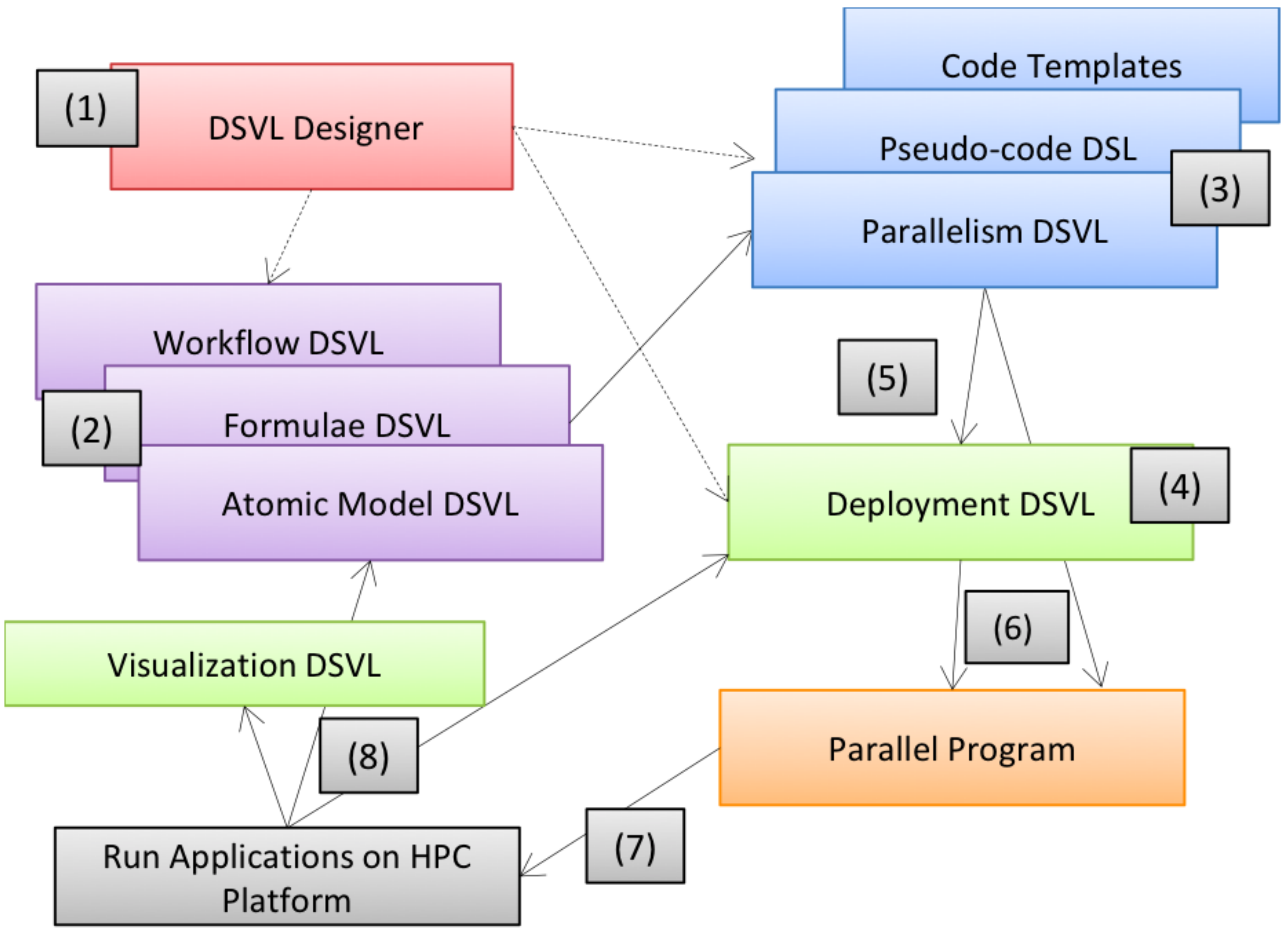
Approach

- Support scientists – and developers! – to model their applications at multiple levels of abstraction – domain right down to detailed C/C++/GPU kernel code
- Use set of user-defined and reusable DSVLs to model
- Provide web-based envrionment including DSVLs designer, coding, debugging, linking DSVL views
- Provide semi-automated support for generating lower-level models, generate code/code annotations, reverse-engineer higher-level models from (existing) code

Development approach to support

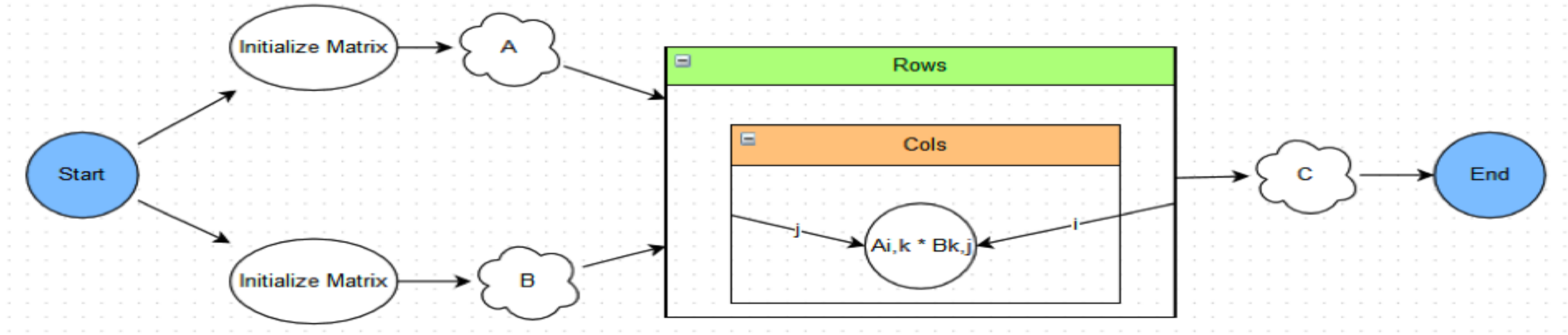


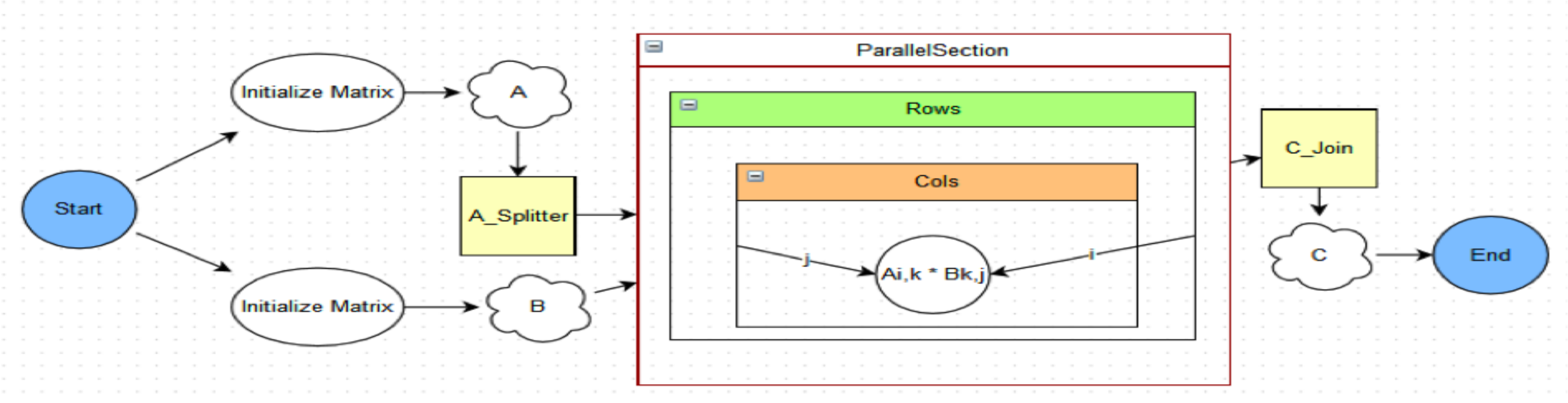
Our Approach



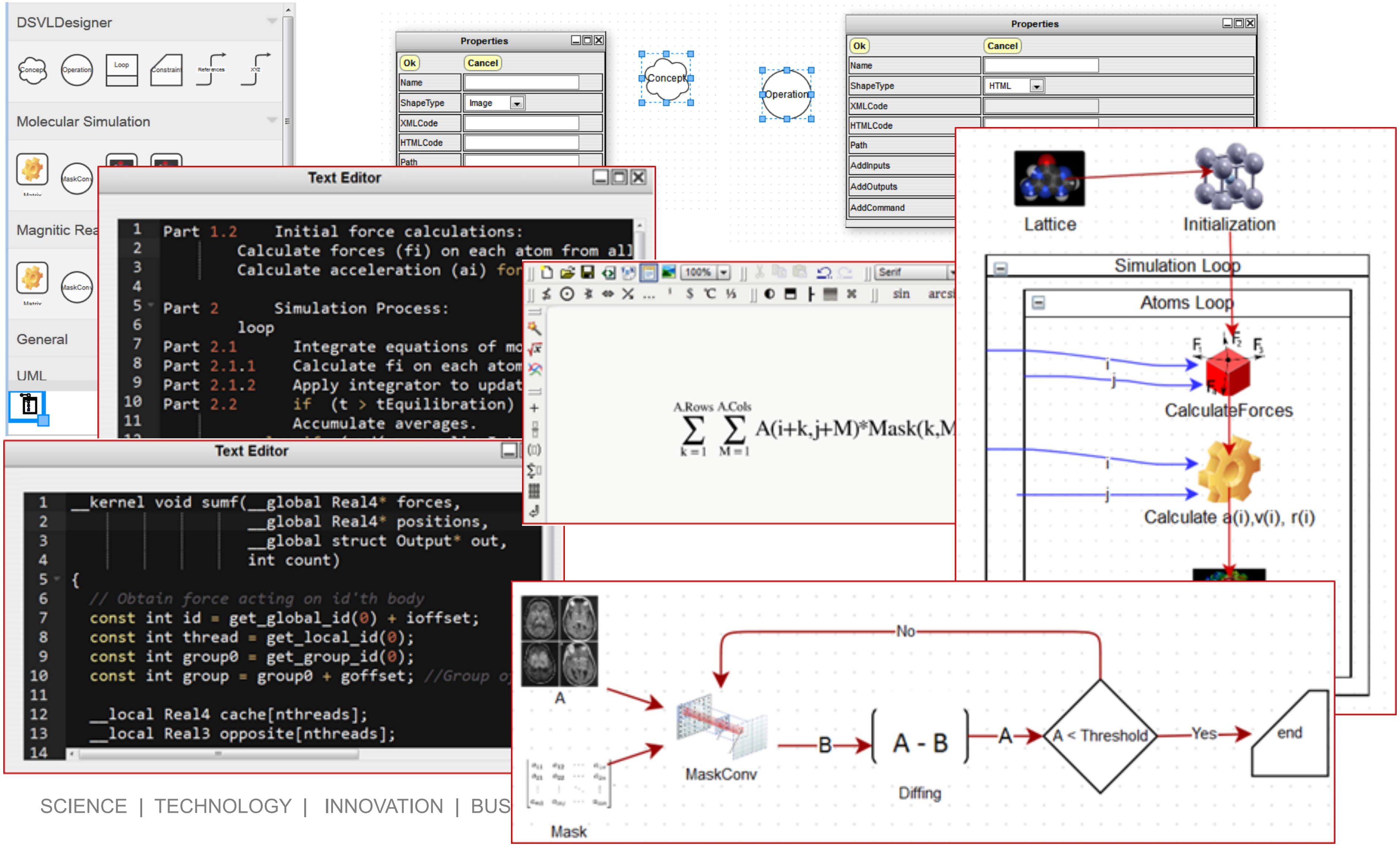
Example

$$C = A \times B\ ,\ C_{i,j} = \mathrm{Sum}(\ A_{i,k} * B_{k,j}\ )$$





Web-based development tool



Properties

Ok Cancel

Name	
ShapeType	Image
XMLCode	
HTMLCode	
Path	

Properties

Ok Cancel

Name	
ShapeType	HTML
XMLCode	
HTMLCode	
Path	
AddInputs	
AddOutputs	
AddCommand	

Text Editor

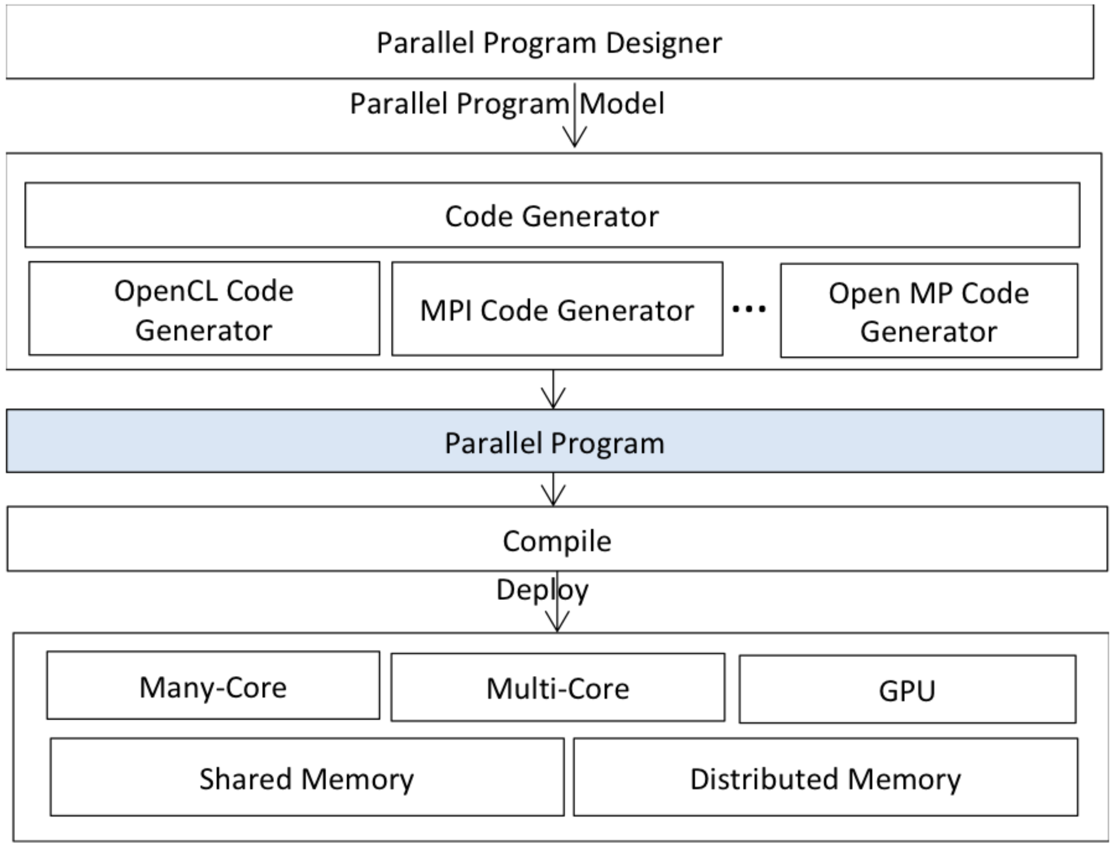
```
Part 1.2     Initial force calculations:
                 Calculate forces (fi) on each atom from all
                 Calculate acceleration (ai) for
                 
Part 2       Simulation Process:
                 loop
Part 2.1         Integrate equations of mo
Part 2.1.1       Calculate fi on each atom
Part 2.1.2       Apply integrator to updat
Part 2.2         if  (t > tEquilibration)
                     Accumulate averages.
```

$$\sum_{k=1}^{A.Rows} \sum_{M=1}^{A.Cols} A(i+k,j+M)*Mask(k,M$$

Text Editor

```
__kernel void sumf(__global Real4* forces,
                   __global Real4* positions,
                   __global struct Output* out,
                   int count)
{
  // Obtain force acting on id'th body
  const int id = get_global_id(0) + ioffset;
  const int thread = get_local_id(0);
  const int group0 = get_group_id(0);
  const int group = group0 + goffset; //Group o

  __local Real4 cache[nthreads];
  __local Real3 opposite[nthreads];
```

Tool


Parallel Program Designer
Parallel Program Model
Code Generator
OpenCL Code Generator
MPI Code Generator
…
Open MP Code Generator
Parallel Program
Compile
Deploy
Many-Core
Multi-Core
GPU
Shared Memory
Distributed Memory

Evaluation

- Define set of DSVLs to model domain (mollecular similation and signal processing for radio telecopes), architecture (GPU and MPI-based CPU), parallelism, processing models (MapReduce etc)
- Model hand-implemented programs at multiple levels of abstraction including links between models
- Generate C and OpenCL code
- Reverse-engineer (parts of) models from C code
- 3 scientists validating approach iteratively

Summary & Future work

- Integrated web-based development environment for GPU-based (and other) scientific applications
- Flexible DSVL designer with pre-packaged DSVLs and user-defined DSVLs
- Semi-automated roundtrip engineering support: model-> code-> model
- Working on:
  - Improve generation/reverse engineering – C/C++, OpenCL
  - Patterns and critics to guide users, analyse models/code
  - Visualisation of running GPU code in models
  - “Liveness”…? ☺

Questions?

Mohamed Almorsy

malmorsy@swin.edu.au

References

- Almorsy, M., Grundy, J.C. and Ruegg, U. HorusCML: Context-aware Domain Specific Visual Languages Designer, 2014 IEEE Symposium on Visual Languages and Human-Centric Computing, Melbourne, Australia, July 27-1 Aug 2014, IEEE CPS

- Almorsy, M. and Grundy, J.C. SecDSVL: A Domain-Specific Visual Language To Support Enterprise Security Modelling, 2014 Australasian Conference on Software Engineering (ASWEC 2014), Sydney, Australia, April 2014, IEEE CS Press.

- Grundy, J.C., Hosking, J.G., Li, N., Li, L., Ali, N.M., Huh, J. Generating Domain-Specific Visual Language Tools from Abstract Visual Specifications, IEEE Transactions on Software Engineering, vol. 39, no. 4, April 2013, pp. 487 - 515.

- Almorsy, M., Grundy, J.C., Sadus, R., Barnes, D., Kaluza, O., van Straten, W., A Suite of Domain-Specific Visual Languages For Scientific Software Application Modelling, 2013 IEEE Symposium on Visual Languages and Human-Centric Computing, San Jose, CA, USA, Sept 15-19 2013, IEEE CPS.

- Almorsy, M., Grundy, J.C., Sadus, R., van Straten, W., Barnes, D., Kaluza, O., Scientific Software Development - Why code when you can draw?, 2013 Accelerated Computing Workshop (ACW 2013), Melbourne, Australia, 9-10 December, 2013.)