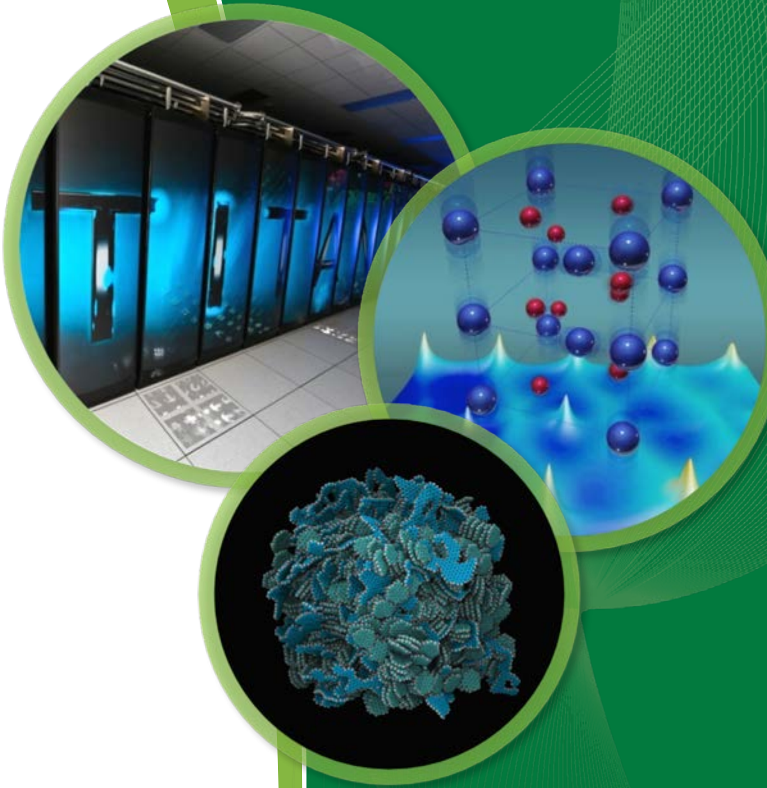
Template Engine Applied to Rapid Modeling



Robert A. Lefebvre

William J. Marshall

NCSD Topical Meeting
Carlsbad, NM, Sept 2017

Outline

- Introduction
  - Current application
  - Benefits
- Template Engine Capabilities
  - Workflow-based
  - Interactive

Introduction

- Templates are applied to everything
  - Presentations, Reports, Papers, Emails, webpages, etc. all originate from **formal** templates
  - Where are you using templates?
  - How frequently do you use templates that are **informal**?
    - E.g., Copy an existing file with the structural components to update it to your new needs.
- Benefits are numerous
  - Provides consistency
    - Parts in a template don’t need to be reproduced
  - Jumpstarts work
  - Encapsulates knowledge
    - Best practices
  - Consolidates input

Prevalence In Nuclear Modeling

- Why are formal templates not more prevalent?
- Specific applications have template-like constructs
  - E.g., MCNP's REPEATED STRUCTURE, SCALE's ALIASES
  - Pro
    - consolidates specific application (e.g., MCNP, SCALE) input
  - Cons
    - requires user have application specific and best practice knowledge
    - requires application specific software feature development
      - May not be consistently available throughout the application
- How to increase prevalence?
  - Develop template engine
    - What capabilities are needed?

Existing Use

- SCALE 6.2
  - Input model sampling for UQ analysis
  - In-document pattern evaluation capability within the Fulcrum user interface

- Used Nuclear Fuel-Storage, Transportation & Disposal Analysis Resource and Data System (UNF-ST&DARDS)
  - Streamline analysis workflow
  - Enables data-driven application input creation

Template Engine Capabilities

- Placeholder
  - Find a named-variable and substitute its value
  - Oldest and simplest construct
- Expressions
  - Find a parameterized expression, evaluate it and substitute its value
  - More complex more powerful
- Pattern repetition
  - Repeat a parameterized section as a function of data
  - Non-obstructive loop construct for repeating patterns of input
- Data-driven
  - Allows incorporation of templates into application workflows

Primary Goals

- Provide users an interactive templating construct
  - Placeholders to keep input consistent
  - Reduce redundancy via consolidated patterns
  - Allow users to request substitution/expansion on demand
  - Facilitates user-driven template

- Provide data-driven reusable templating construct
  - Separate application input format from analysis data
  - Allow reuse of data with different application input templates

Example 1 - Placeholder

- Enhances readability and maintainability

```
' gbc-32 - be buc 4/50; subtask 1.a.iii.2
'
'
'
read geom
unit 1
cylinder 101 1 0.3922 20.32 0
cylinder 401 1 0.4001 20.32 0
cylinder 201 1 0.4572 20.32 0
cuboid 301 1 0.6299 -0.6299 0.6299 -0.6299 20.32 0
unit 2
cylinder 102 1 0.3922 20.32 0
cylinder 402 1 0.4001 20.32 0
cylinder 202 1 0.4572 20.32 0
cuboid 302 1 0.6299 -0.6299 0.6299 -0.6299 20.32 0
unit 3
cylinder 103 1 0.3922 20.32 0
cylinder 403 1 0.4001 20.32 0
cylinder 203 1 0.4572 20.32 0
cuboid 303 1 0.6299 -0.6299 0.6299 -0.6299 20.32 0
unit 4
cylinder 104 1 0.3922 20.32 0
cylinder 404 1 0.4001 20.32 0
cylinder 204 1 0.4572 20.32 0
cuboid 304 1 0.6299 -0.6299 0.6299 -0.6299 20.32 0
unit 5
cylinder 105 1 0.3922 20.32 0
cylinder 405 1 0.4001 20.32 0
cylinder 205 1 0.4572 20.32 0
cuboid 305 1 0.6299 -0.6299 0.6299 -0.6299 20.32 0
```

```
' gbc-32 - be buc 4/50; subtask 1.a.iii.2
' fuelr=<fr=0.3922>, gapr=<gr=0.4001>,
' cladr=<cr=0.4572>, axialh=<ah=20.32>
' hpitch=<hp=0.6299>
read geom
unit 1
cylinder 101 1 <fr> <ah> 0
cylinder 401 1 <gr> <ah> 0
cylinder 201 1 <cr> <ah> 0
cuboid   301 1 <hp> -<hp> <hp> -<hp> <ah> 0
unit 2
cylinder 102 1 <fr> <ah> 0
cylinder 402 1 <gr> <ah> 0
cylinder 202 1 <cr> <ah> 0
cuboid   302 1 <hp> -<hp> <hp> -<hp> <ah> 0
unit 3
cylinder 103 1 <fr> <ah> 0
cylinder 403 1 <gr> <ah> 0
cylinder 203 1 <cr> <ah> 0
cuboid   303 1 <hp> -<hp> <hp> -<hp> <ah> 0
unit 4
cylinder 104 1 <fr> <ah> 0
cylinder 404 1 <gr> <ah> 0
cylinder 204 1 <cr> <ah> 0
cuboid   304 1 <hp> -<hp> <hp> -<hp> <ah> 0
unit 5
cylinder 105 1 <fr> <ah> 0
cylinder 405 1 <gr> <ah> 0
cylinder 205 1 <cr> <ah> 0
cuboid   305 1 <hp> -<hp> <hp> -<hp> <ah> 0
```

Example 2 - Expressions

- Expressions can be used to determine numerical values at runtime
- Example calculates ^{234}U, ^{236}U, and ^{238}U based on formulas provided in Polaris manual

```
' enrichment=<enr=4.8>
read comp

uo2 101 0.96  293  92234 <u234=0.007731*enr^1.0837>
                   92235 <enr>
                   92236 <u236=0.0046*enr>
                   92238 <100-u234-enr-u236> end
...
```

- Variables u234 and u236 will be available for subsequent use
- Numerical value substituted for ^{238}U weight percent, but not available for later use since no variable was defined

Example 3 – Pattern Repetition

- Significantly consolidate input
- Facilitates enhanced fidelity

```
' gbc-32 - be buc 4/50; subtask 1.a.iii.2
' fuelr=<fr=0.3922>, gapr=<gr=0.4001>,
' cladr=<cr=0.4572>, axialh=<ah=20.32>
' hpitch=<hp=0.6299>
read geom
unit 1
cylinder 101 1 <fr> <ah> 0
cylinder 401 1 <gr> <ah> 0
cylinder 201 1 <cr> <ah> 0
cuboid   301 1 <hp> -<hp> <hp> -<hp> <ah> 0
unit 2
cylinder 102 1 <fr> <ah> 0
cylinder 402 1 <gr> <ah> 0
cylinder 202 1 <cr> <ah> 0
cuboid   302 1 <hp> -<hp> <hp> -<hp> <ah> 0
unit 3
cylinder 103 1 <fr> <ah> 0
cylinder 403 1 <gr> <ah> 0
cylinder 203 1 <cr> <ah> 0
cuboid   303 1 <hp> -<hp> <hp> -<hp> <ah> 0
unit 4
cylinder 104 1 <fr> <ah> 0
cylinder 404 1 <gr> <ah> 0
cylinder 204 1 <cr> <ah> 0
cuboid   304 1 <hp> -<hp> <hp> -<hp> <ah> 0

...
```

```
' gbc-32 - be buc 4/50; subtask 1.a.iii.2
' fuelr=<fr=0.3922>, gapr=<gr=0.4001>,
' cladr=<cr=0.4572>, axialh=<ah=20.32>
' hpitch=<hp=0.6299>, axialc=<ac=18>
read geom
#for( n=1; n <= axialc; n=n+1)
unit <n>
cylinder <100+n> 1 <fr> <ah> 0
cylinder <400+n> 1 <gr> <ah> 0
cylinder <200+n> 1 <cr> <ah> 0
cuboid   <300+n> 1 <hp> -<hp> <hp> -<hp> <ah> 0
#endfor
```

Example 4 – Data-Driven Workflow

- Separate data from input template
- Facilitates UNF-ST&DARDS workflow
  - Database to template parameter set to application input

id	u235_wtpt	fuelr	gapr	cladr	pitch	height	npins
1	3.81	0.48	0.49	0.56	1.48	370.10	14
2	3.49	0.41	0.42	0.48	1.26	350.25	17
3	4.20	0.41	0.42	0.48	1.26	350.25	17
4	4.20	0.39	0.41	0.45	1.26	364.25	17

Critical Safety

SCALE/KENO

MCNP

Thermal Hydraulics

COBRA-SFS

...

...

???

...

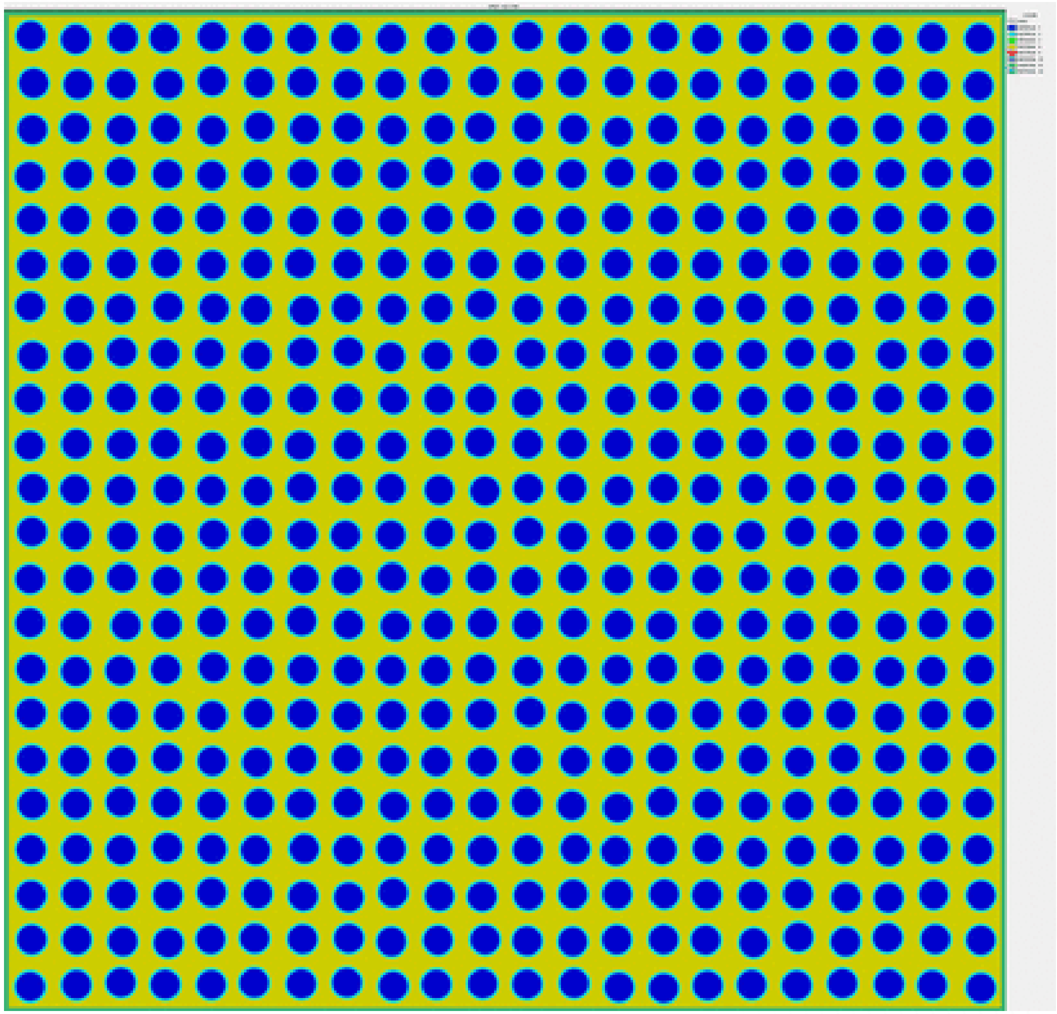
Example 5 – Data-Driven Model Perturbation

- Single fuel pin unit cell specified in KENO input
- Position of each rod is sampled uniquely using loop construct

```
#for(i=1; i<=1170; i=i+1)
 unit <10000+i>
' bottom end plug
 cylinder 3 1  0.5588 1.27 0  origin 0 0
 cylinder 5 1  0.6350 1.27 0  origin 0 0
 cuboid  10 1  0.842 -0.842  0.842 -0.842  1.27 0
'
 unit <20000+i>
' fueled section
 cylinder 1 1  0.5588 91.44 0  origin 0 0
 cylinder 6 1  0.6350 91.44 0  origin 0 0
 cuboid   9 1  0.842 -0.842  0.842 -0.842  91.44 0
'
 unit <30000+i>
' clad top end plug
 cylinder 2 1  0.5588 0.48 0  origin 0 0
 cylinder 4 1  0.6350 0.48 0  origin 0 0
 cuboid  10 1  0.842 -0.842  0.842 -0.842  0.48 0
'
 unit <40000+i>
' top end plug
 cylinder 2 1  0.6350 4.6 0  origin 0 0
 cuboid  10 1  0.842 -0.842  0.842 -0.842  4.6 0
#endfor
```

```
#for(i=1; i<=1170; i=i+1)
' sample displacement
read variable[displacement_<i>_042_001]
   distribution=normal
   value = 0  stddev = 0.0054
   minimum = -0.207  maximum = 0.207
   cases = Case1 end
end variable
read variable[theta_<i>_042_001]
   distribution=uniform
   value = 0  minimum = 0  maximum = 6.2831853
   cases = Case1 end
end variable
' calculate displacement in (x,y) - apply to origin
read variable[displacement_x_<i>_042_001]
   distribution=expression
   expression = "displacement_<i>_042_001 *
                 cos(theta_<i>_042_001)"
   cases = Case1 end
end variable
read variable[displacement_y_<i>_042_001]
   distribution=expression
   expression = "displacement_<i>_042_001 *
   sin(theta_<i>_042_001)"
cases = Case1 end
end variable
#endfor
```

Example 5 – Data-Driven Model Perturbation



75 realizations, looped to make animation

Summary

- Templating is not a new concept, but has generally been implemented previously within specific codes
  - MCNP pstudy
  - SCALE PRISM
- Going forward, the TemplateEngine will provide a powerful, code-agnostic capability to process input
- Facilitates automation of model construction based on existing data
  - UNF-ST&DARDS
- Enables construction of more complicated models with looping constructs
- Combination could enable much more realistic modeling for complicated systems