Software Architecture for Identifying and Updating Parameters

Michael H. Scott
Oregon State University

OpenSees Developers Symposium
August 16, 2006

Department of Civil, Construction, and Environmental Engineering

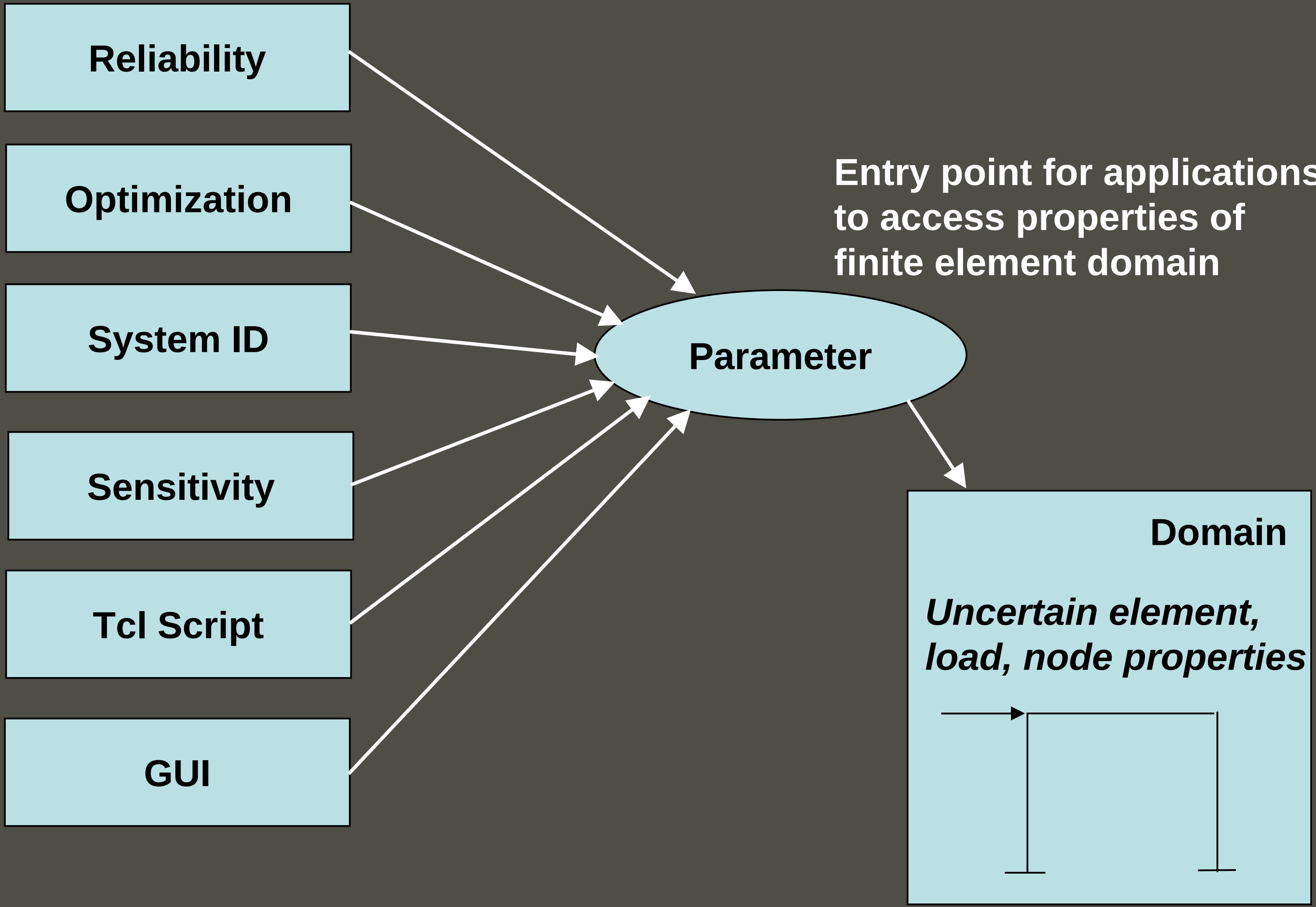
Structural Engineering

Applications Requiring Parameterization

- Structural reliability
- Optimization
- System identification
- Design/Response sensitivity
- Parametric studies (Tcl scripts)
- Graphical user interfaces

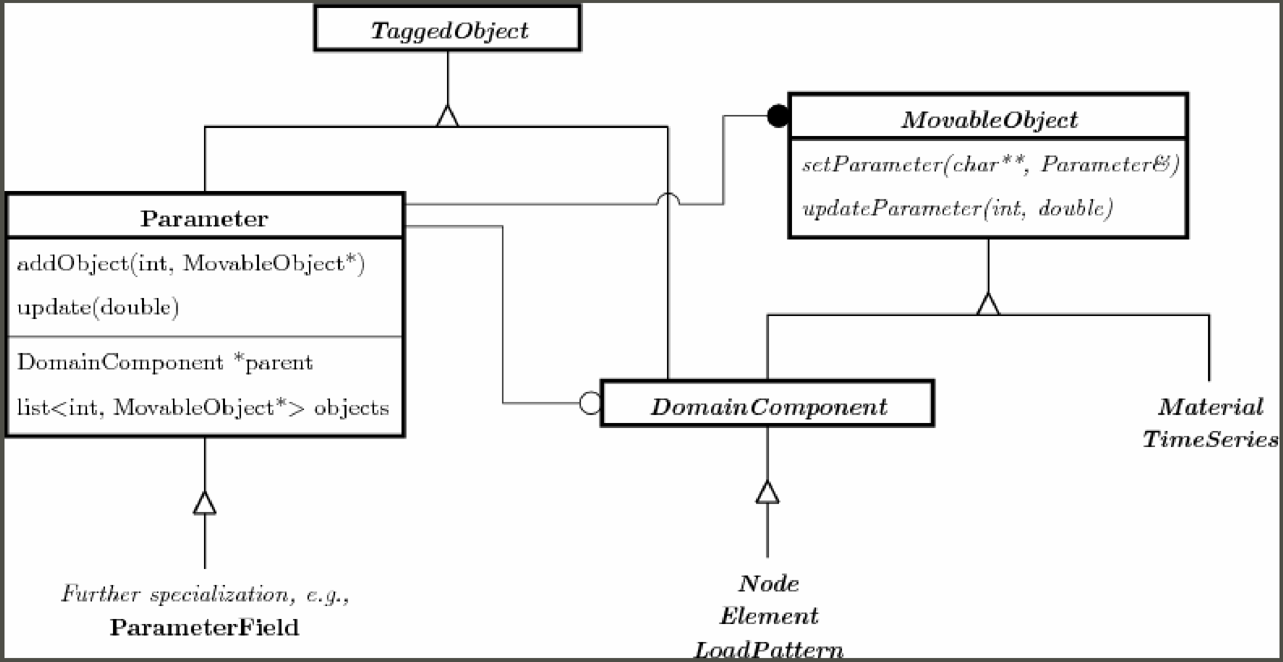
Department of Civil, Construction, and Environmental Engineering

Parameter Class



Entry point for applications to access properties of finite element domain

Parameter Class Diagram


TaggedObject
MovableObject
setParameter(char**, Parameter&)
updateParameter(int, double)
Parameter
addObject(int, MovableObject*)
update(double)
DomainComponent *parent
list<int, MovableObject*> objects
DomainComponent
Material
TimeSeries
Further specialization, e.g.,
ParameterField
Node
Element
LoadPattern

- Parameter contains pointers to MovableObjects (base class for most everything in OpenSees)
- Parameter::addObject adds a MovableOjbect to list

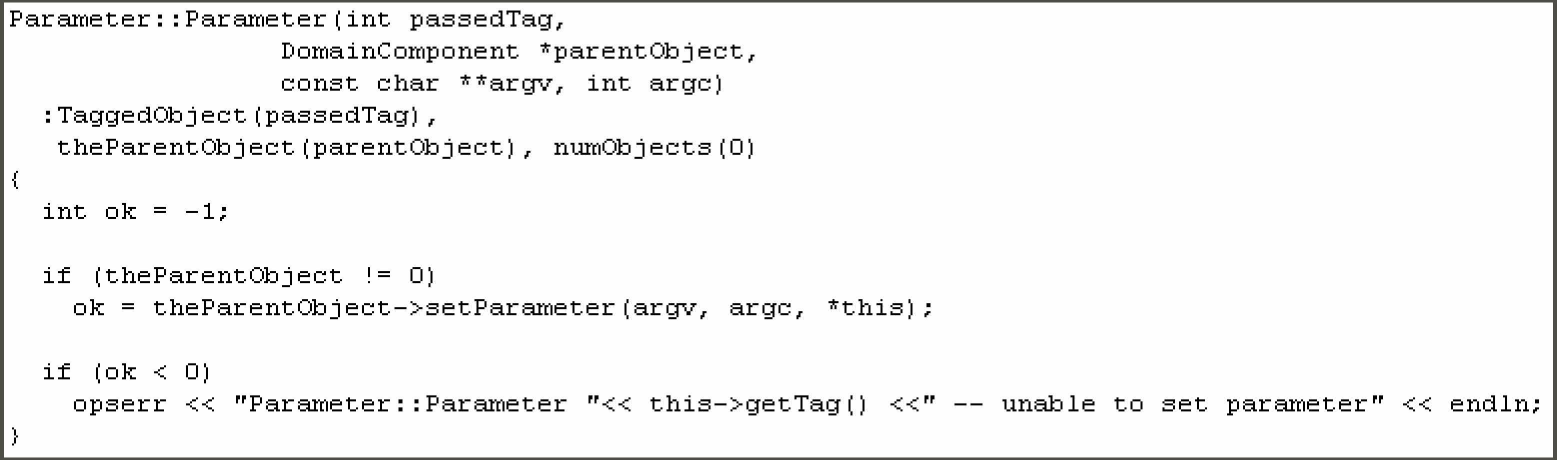
Department of Civil, Construction, and Environmental Engineering

Oregon State University

Department of Civil, Construction, and Environmental Engineering

Parameter Class Constructor

- Invoke setParameter on a DomainComponent
- “Chain of Responsibility” software pattern



```
Parameter::Parameter(int passedTag,
                     DomainComponent *parentObject,
                     const char **argv, int argc)
  :TaggedObject(passedTag),
   theParentObject(parentObject), numObjects(0)
{
  int ok = -1;

  if (theParentObject != 0)
    ok = theParentObject->setParameter(argv, argc, *this);

  if (ok < 0)
    opserr << "Parameter::Parameter "<< this->getTag() <<" -- unable to set parameter" << endln;
}
```

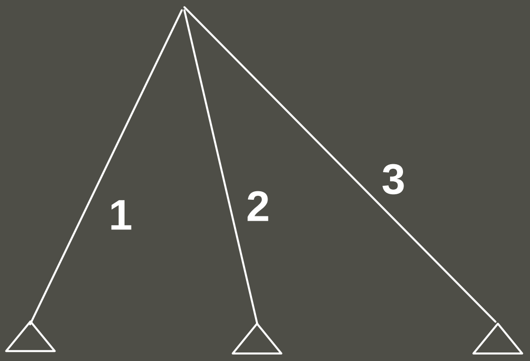
Example of Truss Elements with Elastic UniaxialMaterial

Oregon State University

Tcl commands in OpenSees

Model Building

```
uniaxialMaterial Elastic 1 29000.0
element truss 1 1 2  4.0 1
element truss 2 1 3  4.0 1
element truss 3 1 4  4.0 1
```



Parameter identification

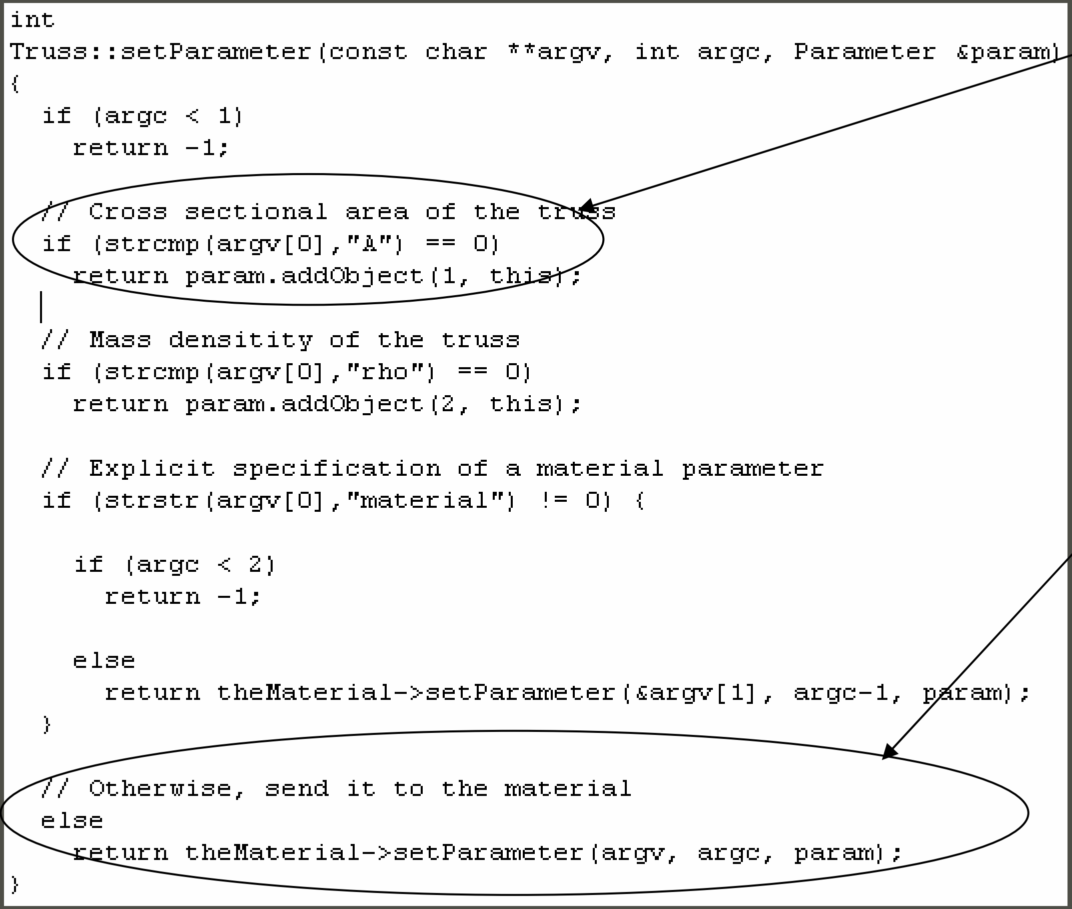
```
parameter 1 element 1 A
parameter 2 eleRange 1 3 E
```

Parameter update

```
updateParameter 1 5.0
updateParameter 2 30000.0
```

Code for parameter identification in element

parameter 1 element 1 A



```
int
Truss::setParameter(const char **argv, int argc, Parameter &param)
{
  if (argc < 1)
    return -1;

  // Cross sectional area of the truss
  if (strcmp(argv[0],"A") == 0)
    return param.addObject(1, this);

  // Mass densitity of the truss
  if (strcmp(argv[0],"rho") == 0)
    return param.addObject(2, this);

  // Explicit specification of a material parameter
  if (strstr(argv[0],"material") != 0) {

    if (argc < 2)
      return -1;

    else
      return theMaterial->setParameter(&argv[1], argc-1, param);
  }

  // Otherwise, send it to the material
  else
    return theMaterial->setParameter(argv, argc, param);
}
```

Add id and pointer to self to Parameter object

Forward request to UniaxialMaterial object if no parameters found

Department of Civil, Construction, and Environmental Engineering

Code for parameter identification in material

parameter 2 eleRange 1 3 E

```
int
ElasticMaterial::setParameter(const char **argv, int argc, Parameter &param)
{
  if (argc < 1)
    return -1;

  if (strcmp(argv[0],"E") == 0)
    return param.addObject(1, this);

  else if (strcmp(argv[0],"eta") == 0)
    return param.addObject(2, this);

  else
    return -1;
}
```

Add self to Parameter object

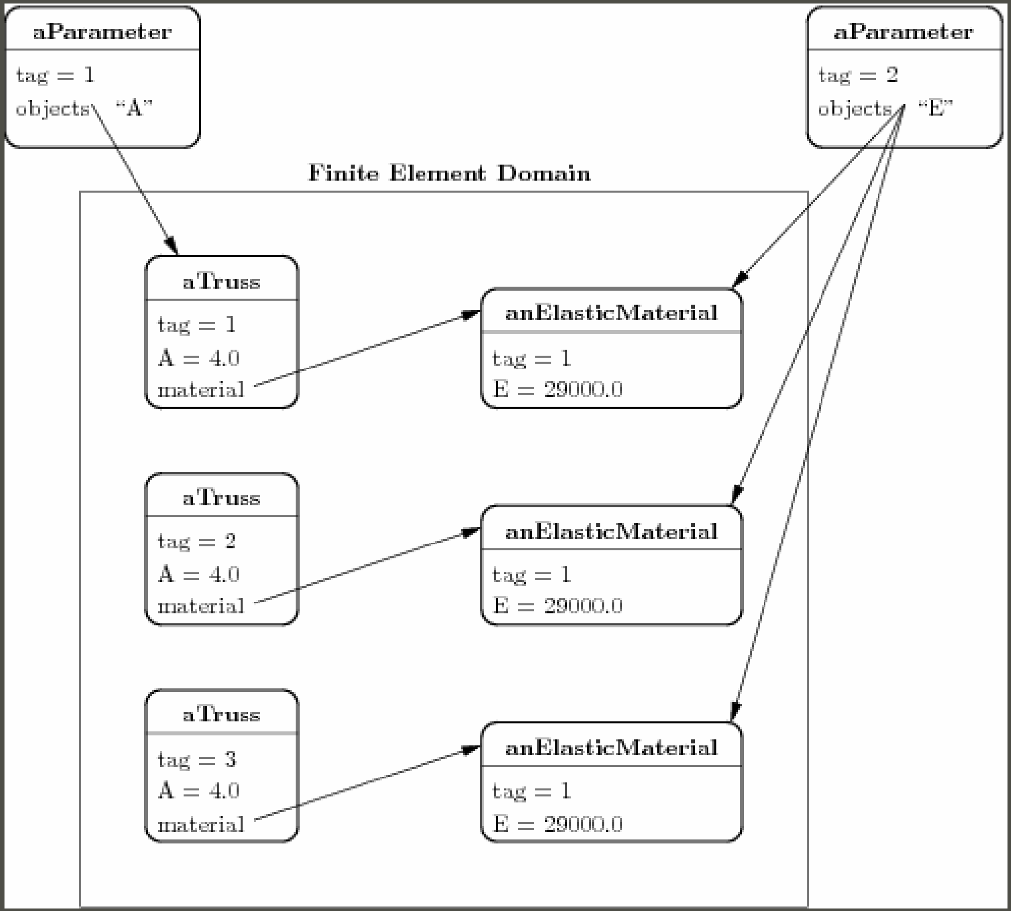
Else parameter not found (end of chain)

- Association of id to parameter is encapsulated by each class
- Developer decides what is a parameter and which id values to use

Department of Civil, Construction, and Environmental Engineering

Object Diagram After Parameters Identified

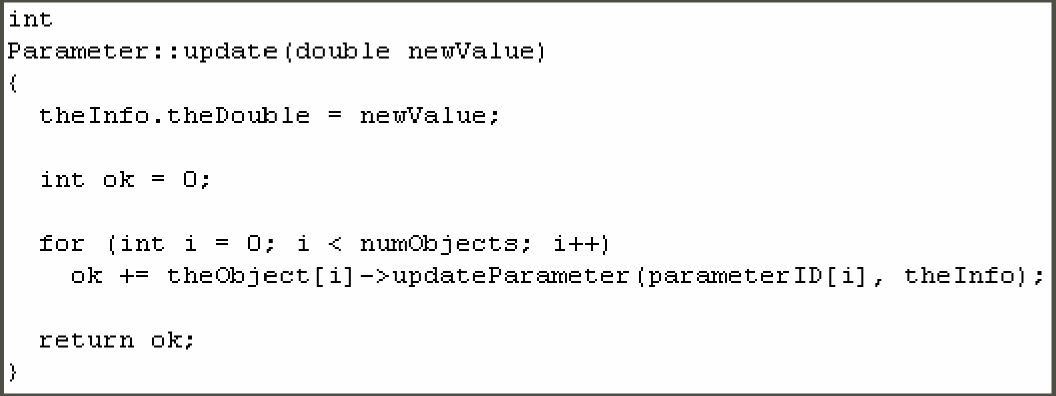
parameter 1 element 1 A
parameter 2 eleRange 1 3 E



Department of Civil, Construction, and Environmental Engineering

Code to update parameter values

updateParameter 1 5.0
updateParameter 2 30000.0



```
int
Parameter::update(double newValue)
{
  theInfo.theDouble = newValue;

  int ok = 0;

  for (int i = 0; i < numObjects; i++)
    ok += theObject[i]->updateParameter(parameterID[i], theInfo);

  return ok;
}
```

- Iterate over MovableObjects and invoke updateParameter with new parameter value and id number

Department of Civil, Construction, and Environmental Engineering

updateParameter Methods

updateParameter 1 5.0

```
int
Truss::updateParameter (int parameterID, Information &info)
{
  switch (parameterID) {
  case 1:
    A = info.theDouble;
    return 0;
  case 2:
    rho = info.theDouble;
    return 0;
  default:
    return -1;
  }
}
```

match the id, then update the parameter

updateParameter 2 30000.0

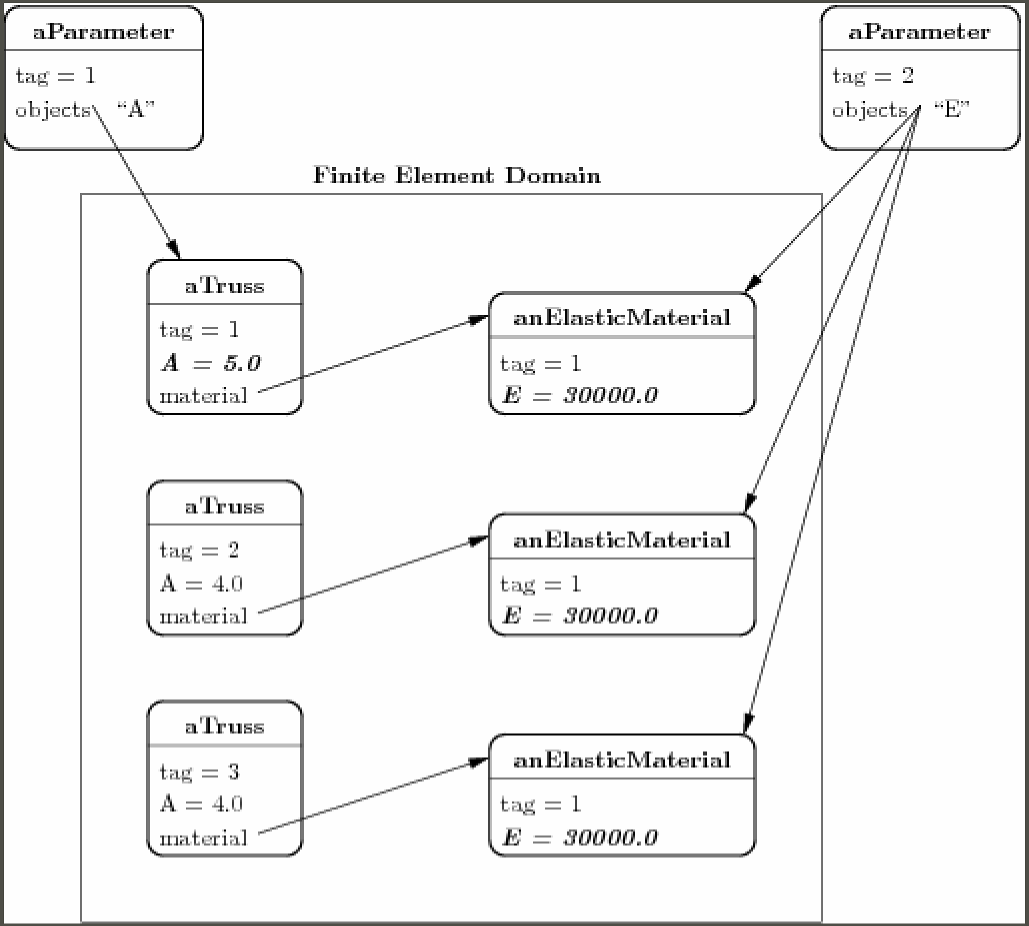
```
int
ElasticMaterial::updateParameter(int parameterID, Information &info)
{
  switch(parameterID) {
  case 1:
    E = info.theDouble;
    return 0;
  case 2:
    eta = info.theDouble;
    return 0;
  default:
    return -1;
  }
}
```

Oregon State UNIVERSITY

Department of Civil, Construction, and Environmental Engineering

Object Diagram After Parameters Updated

updateParameter 1 5.0
updateParameter 2 30000.0



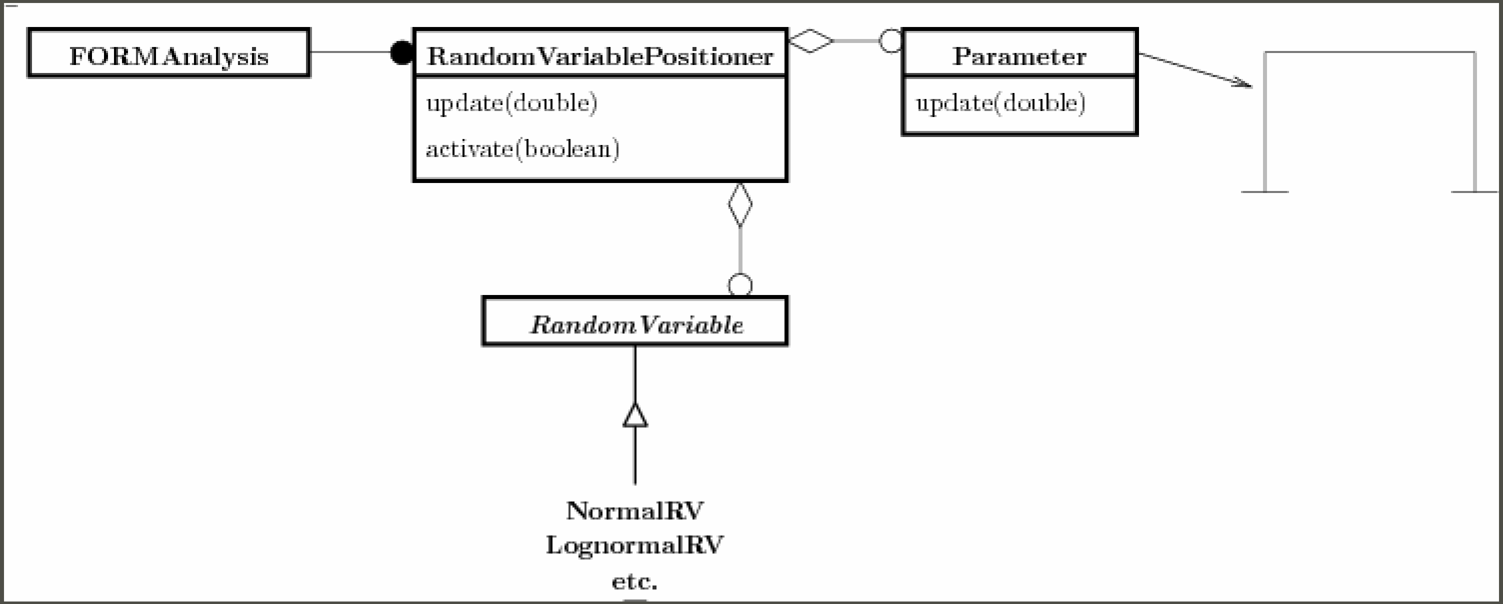
Department of Civil, Construction, and Environmental Engineering

Extensible Parameterization Framework

- Solid and beam-column elements
  - constitutive model at each integration point
- Fiber-discretized cross sections
- Nodal coordinates
- Nodal and element loads
- Any MovableObject…

Department of Civil, Construction, and Environmental Engineering

Random Variables in a Reliability Analysis



- RandomVariablePositioner aggregates a RandomVariable with a Parameter in order to model uncertain properties in the finite element domain (Terje Haukaas, UBC)
- Additional methods required to compute response sensitivity with respect to each parameter

Questions/Comments?

- Reference:

  Scott, M.H., Fenves, G.L., McKenna, F.T., and Filippou, F.C. "Software Patterns for Nonlinear Beam-Column Models" *Journal of Structural Engineering*, Under review, Submitted July 2006.

- e-mail:

  michael.scott@oregonstate.edu

Oregon State UNIVERSITY

Department of Civil, Construction, and Environmental Engineering