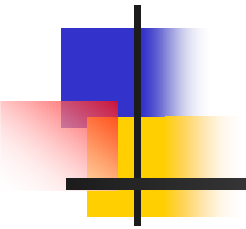
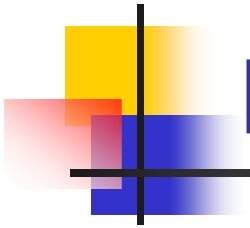


KerMeta
– a Meta-modelling Language

Jun Li
McGill University
March 2008

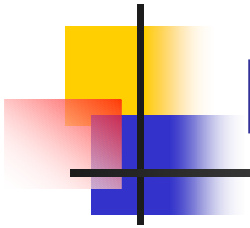


References

- http://www.kermeta.org/
- Pierre-Alain Muller, Franck Fleurey, and Jean-Marc Jézéquel, Weaving executability into object-oriented meta-languages. Proceedings of MODELS/UML'2005, volume 3713 of LNCS,
- Z. Altahat, T. Elrad, D. Vojtisek. Using Aspect Oriented Modeling to localize implementation of executable models, in: Models and Aspects workshop, at ECOOP 2007, July 2007

KerMeta – Kernel Metamodeling

- Created by Triskell team, IRISA - Institut de recherche en informatique et systèmes aléatoires.
- Model oriented
  - Meta-models' structural specifications
- EMOF compliant
  - An extension to the Essential Meta-Object Facilities (EMOF) 2.0
- Object-oriented (Eiffel, Java)
- Imperative
- Statically Typed (100% typesafe)
- Aspect-Oriented Modeling
- Transform from/to Ecore (EMF)



KerMeta

- Meta-models' structural specifications
- Specify operational semantics / action specifications
  - Static semantic (similar to OCL)
    - Constraints: pre & post conditions
    - Has algorithmic specification
  - Dynamic semantic (using operational semantic in the operation of the meta-model)
  - Domain specified: not Java, MOF

KerMeta Advantages

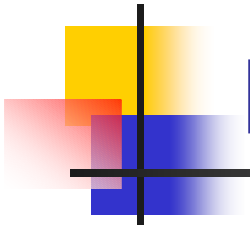
- EMOF operation's definition

  Operation ***isInstance(element:Element):Boolean***

  "Returns true if the element is an instance of this type or a subclass of this type. Returns false if the element is null"

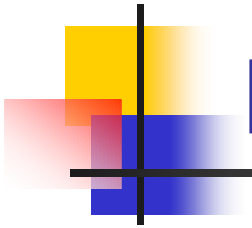
```
operation isInstance(element : Element) : Boolean is do
    // false if the element is null
    if element == void then result := false
    else
        // true if the element is an instance of this type
        // or a subclass of this type
        result := element.getMetaClass == self or
            element.getMetaClass.allSuperClasses.contains(self)
    end
end
```

Fig. 1. Executable specification of the *isInstance* operation of the EMOF *Type* class



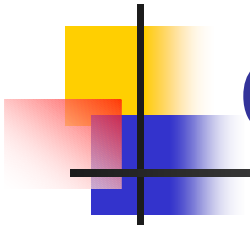
KerMeta - Environment

- Integrated with Eclipse
- KerMeta Text Editor
  - Syntax highlighting
  - Code auto-completion
  - Error reporting: parsing and type checking
  - Integrated with outline feature
- Model Editor
  - generated with the tool "Topcased"
- Debugger...
  - Set breakpoints,
  - Watch view, ...



Demonstrate in Eclipse

- Demonstrate creating a Kermeta meta-model in Eclipse
  - Modifying Ecore class diagram
  - Transferring it into kermeta model, km, kmt
  - Adding operational semantic code
    - Showing the Kermeta IDE, auto-complete, syntax high-lighting, debug environment, outline list…
  - Building FSA model based on meta-model
  - Running the Kermeta simulation, to show the trace of operations codes
  - Showing pre/post conditions and running it to show how it works
  - Model transformation from NFA to DFA



Conclusion

- KerMeta is an extension to the Essential Meta-Object Facilities (EMOF) 2.0
- Built on the top of Eclipse Modeling Framework; transform from/to Ecore
- Integrated with Eclipse; distributed as Eclipse plug-in,
- Executable meta-language
- Textual Simulation

Q & A