Appendix A

Notes
In this appendix the most important abstract and concrete classes are listed of the framework that was described in the dissertation. First remark that the classes listed here can differ in minor details of their counterpart in the dissertation itself. The names of some classes have been changed to avoid name collisions. Remark also that this is by no means a complete listing of all the classes in the framework.

Furthermore some notes must be made about the descriptions of these classes. In principle the language used is self-explanatory. It uses a Smalltalk-like message passing, and class descriptions. However, two notations have been introduced that deal with overloading. The superscript "+" is used to indicate in a (abstract) class that an argument can be — apart from being overridden — overloaded in later subclasses. In fact it indicates that the method is overloaded on the indicated argument with all subclasses of the formal parameter. Also, a formal parameter that takes the form of a "name:class" couple indicates overloading on that argument. Finally note that in the class descriptions no instantiation methods are listed. We presume the existence of instantiation methods for all classes. Instantiation methods take the form of a keyword message composed of the instance variable names.

Kernel Abstract Classes of the Framework

Abstract Expressions with Overloading on the Context Argument

```
class AbstractExpression
  methods
    abstract eval:StandardContext+ result AbstractMetaObject
endclass
```

Abstract Meta-Object with Overloading on the Pattern Argument

```
class AbstractMetaObject
  methods
   abstract send:AbstractPattern+ client:StandardClient
      result AbstractMetaObject
endclass
```

Abstract Pattern Class

```
class AbstractPattern
  methods
   abstract = pattern result: Boolean
endclass
```

Standard Client Object

```
class StandardClient
endclass
```

Standard Context Object

```
class StandardContext
endclass
```

Kernel Classes of the Simple Framework

Message Passing with Evaluation and Pattern Categories

```
class UnaryMessageExpression extends AbstractExpression
  abstract class attributes
   EmptyClient
  instance variables
   receiver:AbstractExpression
   pattern:AbstractPattern
  methods
   concrete eval:(context:StandardContext+) result AbstractMetaObject
     ^(receiver eval:context)
        send:(pattern asCategory:context)
        client:EmptyClient
endclass
```

Receiverless Message Passing with Evaluation and Pattern Categories

```
class PatternExpression extends AbstractExpression
  abstract class attributes
   EmptyClient
  instance variables
   pattern:AbstractPattern
  methods
   concrete eval:(context:StandardContext+) result AbstractMetaObject
     ^context send:(pattern asCategory:context) client:EmptyClient
endclass
```

Standard Client Object for Simple, Grouping all Send Arguments

```
class SimpleStandardClient extends StandardClient
 public instance variables
  private:AbstractMetaObject
endclass
```

Standard Context Object for Simple, Grouping All Evaluation Arguments

```
class SimpleStandardContext extends StandardContext
 public instance variables
  private:AbstractMetaObject
endclass
```

Adapted Abstract Pattern Class for Simple

```
class SimpleAbstractPattern extends AbstractPattern
  methods
   abstract asCategory:StandardContext result: AbstractPattern
endclass
```

Abstract Class for Slots

```
class Slot
  instance variables
   key:AbstractPattern
  methods
   abstract valueIn:context result AbstractMetaObject
   concrete key result AbstractPattern
     ^key
endclass
```

Other Classes of the Simple Framework

Simple Expression Class Hierarchy

```
AbstractExpression
  AbstractionExpression
   AbstractCompoundObjectExpression (ACompoundObject)
     CompoundObjectExpression (publicPart:AbstractExpression,
                               privatePart:AbstractExpression)
   AbstractBaseObjectExpression (ABaseObject)
     BaseObjectExpression (associations:Sequence(AssociationExpression))
  UnaryMessageExpression (receiver:AbstractExpression,
                          pattern:AbstractPattern)
  PatternExpression (pattern:AbstractPattern)
AbstractAssociationExpression (AAssociation)
  AssociationExpression (pattern:AbstractPattern, value:AbstractExpression)
   VariableAssociationExpression
   MethodAssociationExpression
AbstractPattern
  Pattern(name:String)
```

Simple Meta-Object Class Hierarchy

```
AbstractMetaObject
     CompoundObject (publicPart:AbstractMetaObject
                     privatePart:AbstractMetaObject)
     BaseObject (slots:slots:Sequence(Slot))
```

Simple Slot Class Hierarchy

```
Slot
  VariableSlot (value:AbstractMetaObject)
  MethodSlot (value:AbstractExpression)
```

Kernel Classes of the Agora Framework

Agora Message Passing

```
class MessageExpression extends AbstractExpression
  abstract class attributes
   StandardClient
  instance variables
   receiver:AbstractExpression,
   pattern:AbstractPattern
  methods
   concrete eval:(context:StandarContext+)
       result AbstractMetaObject
     local variables arguments:ArgumentList

     for each argument in pattern do
        arguments add:(argument eval:(context asFunctionalContext))

     ^(receiver eval:(context asFunctionalContext))
         send:(pattern asCategory:context)
         client:(StandardClient arguments:arguments)
endclass
```

Evaluation of Receiverless Messages

```
class PatternExpression extends AbstractExpression
  abstract class attributes
   StandardClient
  instance variables
   pattern:AbstractPattern
  methods
   concrete eval:(context:StandaredContext+)
       result AbstractMetaObject
     local variables arguments:ArgumentList

     for each argument in pattern do
        arguments add:(argument eval:(context asFunctionalContext))

     ^(context privatePart)
         send:(pattern asCategory:context)
         client:(StandardClient arguments:arguments)
endclass
```

Pattern Classes Used in Expressions Versus Pattern Classes Used in Messages

```
class AbstractPattern
  methods
   abstract asCategory:StandardContext+ result AbstractCategoryPattern
endclass
```

```
class AbstractCategoryPattern
  methods
   abstract = AbstractCategoryPattern+ result Boolean
endclass
```

Agora Standard Client

```
class AgoraStandardClient extends StandardClient
  public instance variables
   arguments:ArgumentList
endclass
```

Agora Standard Context

```
class AgoraStandardContext extends StandardContext
  methods
   abstract asFunctionalContext result StandardContext
     ... return an instance of functional context with the same
     ... content
endclass
```

Agora Slot Class Hierarchy

```
Slot
  ReadVariableSlot (value:AbstractMetaObject)
  WriteVariableSlot (value:AbstractMetaObject)
  MethodSlot (value:AbstractExpression)
  MixinSlot (value:AbstractExpression)
  RiefierSlot (value:AbstractExpression)
```

Agora Expression Class Hierarchy

```
AbstractExpression
  MessageExpression (receiver:AbstractExpression, pattern:AbstractPattern)
  ReifierMessageExpression (receiver:AbstractExpression,
                                    pattern:AbstractReifierPattern)
  AggregateExpression (expressions:Array(AbstractExpression),
                           delimiter:Delimiter)
  LiteralExpression (value:LiteralValue)
  PatternExpression (pattern:AbstractPattern)
  ReifierPatternExpression (pattern:AbstractReifierPattern)

AbstractPattern
  UnaryPattern (id:Identifier)
  OperatorPattern (op:Operator, argument:AbstractExpression)
  KeywordPattern (keys:Array(Keyword), arguments:Array(AbstractExpression))
AbstractReifierPattern
  UnaryReifierPattern (id:Identifier)
  OperatorReifierPattern (op:Operator, argument:AbstractExpression)
  KeywordReifierPattern (keys:Array(Keyword),
                           arguments:Array(AbstractExpression))

Identifier (name:String)
Operator (name:String)
Keyword (name:String)
Delimiter (left:String, right:String)
```

Kernel Classes for Handling Inheritance in the Agora Framework

Wrapper Objects

```
class WrapperObject extends AbstractMetaObject
  abstract class attributes
   DelegationContext
  instance variables
   delegate: AbstractInternalObject
  methods
   concrete send:pattern client:client result AbstractMetaObject
     ^delegate delegate:pattern
               client:client
               context:(DelegationContext receiver:self
                                          public:delegate)
endclass
```

Root of the Abstract Internal Object Classes

```
class AbstractInternalObject
  methods
   abstract  delegate:CategoryPattern
             client:StandardClient
             context:DelegationContext+
       result AbstractMetaObject
endclass
```

Delegation Contexts Used in the Realisation of Mixin methods

```
class DelegationContext extends StandardContext
  methods
   concrete parent:AbstractInternalObject result DelegationContext
     -- returns a copy of the context with a new parent field
   concrete private:AbstractInternalObject result DelegationContext
     -- returns a copy of the context with a new private field
   concrete noParent result DelegationContext
     -- returns a copy of the context with a empty parent field
   concrete noPrivate result DelegationContext
     -- returns a copy of the context with a empty private field
   concrete receiver:AbstractMetaObject result DelegationContext
     -- returns a copy of the context with a new receiver field
   concrete public:AbstractInternalObject result DelegationContext
     -- returns a copy of the context with a new public field
endclass
```

Agora Delegation Contexts that Record the Public and Private Slots

```
class MixinContext extends DelegationContext
  methods
   concrete publicSlots:ExtensibleSimpleObjectresult DelegationContext
     -- returns a copy of the context with a new publicSlots field
   concrete privateSlots:ExtensibleSimpleObject result DelegationContext
     -- returns a copy of the context with a new privateSlots field
endclass
```

Concrete Internal Object Classes

```
DelegationContext
  ObjectWithParent (thisPart: AbstractInternalObject
                    parentPart: AbstractInternalObject)
  EncapsulatedObject (publicPart: AbstractInternalObject
                      privatePart: AbstractInternalObject)
  SimpleObject (thisPart:Sequence(Slot))
     ExtensibleSimpleObject
```