

# The IFF 2-Category (meta) Ontology

THE IFF NAMESPACE OF 2-CATEGORIES ..... 1  
*The Underlying Category* ..... 2  
*The Horizontal Category* ..... 3  
*The Vertical Source and Target Functors*..... 5  
*The Vertical Category*..... 6  
*Additional Properties*..... 8  
*Example: The Category of Categories*..... 8

This document axiomatizes the meta-ontology of 2-categories, which is situated in the third (top) metalevel. This will include namespaces for 2-categories, 2-functors and 2-natural transformations.

## The IFF Namespace of 2-Categories

2-cat

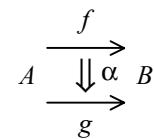
Define adjunctions and monads within a 2-category. Review the axiomatization in terms of a 2-category as a category object in  $\mathbf{Cat}$ , with  $\mathbf{K}$  as the object of objects, with  $\mathbf{K}_{\text{horiz}}$  as the object of morphisms, and with vertical composition as the morphism of composition.

The terminology of the 2-category namespace (part of the IFF-Top metalanguage) is listed in Table 1.

**Table 1: The 2-category part of the Top Metalanguage, introduced in the 2-category meta-ontology**

	KIF.COL\$collection	KIF.MOR\$function	Other
<b>2-cat</b>	object = 0-cell arrow = morphism = 1-cell 2-cell		2-category
	composable-pair	source target zeroth first composition identity	category composable-opspan composable
	horizontally- composable-pair	source-object = horizontal-source target-object = horizontal-target horizontal-zeroth horizontal-first horizontal-composition horizontal-identity	horizontal-category horizontally- composable-opspan horizontally-composable
		source-pair target-pair	vertical-source-functor vertical-target-functor
	vertically- composable-pair	source-arrow = vertical-source target-arrow = vertical-target vertical-zeroth vertical-first vertical-composition vertical-identity	vertical-category vertically-composable- opspan vertically-composable

A 2-category  $\mathbf{K}$  is directly described as follows.  $\mathbf{K}$  has a collection of *objects* or *0-cells*, a collection of *arrows*, *morphisms* or *1-cells*, and a collection of *2-cells*. There are *source* and *target* functions for arrows and 2-cells sufficiently indicated by Figure 1. In particular, the source and target objects of the source (and target) arrow of a 2-cell are its source and target objects, respectively. These definitions and constraints will be axiomatized below. The linear notation corresponding to the 2-cell in Figure 1 is  $\alpha : f \Rightarrow g : A \rightarrow B$ .



**Figure 1: 2-cell**

- (1) (UR\$object 2-category)
- (2) (UR\$morphism object) (UR\$morphism 0-cell)  
 (= 0-cell object)  
 (= (UR\$source object) 2-category)  
 (= (UR\$target object) KIF.COL\$collection)
- (3) (UR\$morphism arrow) (UR\$morphism morphism) (UR\$morphism 1-cell)  
 (= morphism arrow) (= 1-cell arrow)  
 (= (UR\$source arrow) 2-category)

```
(= (UR$target arrow) KIF.COL$collection)

(4) (UR$morphism 2-cell)
    (= (UR$source 2-cell) 2-category)
    (= (UR$target 2-cell) KIF.COL$collection)
```

## The Underlying Category

The objects and arrows form a (very large) category  $\underline{K}$  called the *underlying category* of  $K$ . The composition and identity functions in  $\underline{K}$  are renamed here as arrow composition and arrow identity.

```
(5) (UR$morphism category)
    (= (UR$source category) 2-category)
    (= (UR$target category) cat.3$category)
    (= (UR$composition [category cat.3$object]) object)
    (= (UR$composition [category cat.3$morphism]) arrow)

(6) (UR$morphism source)
    (= (UR$source source) 2-category)
    (= (UR$target source) KIF.MOR$function)
    (= (UR$composition [source KIF.MOR$source]) arrow)
    (= (UR$composition [source KIF.MOR$target]) object)
    (= source (UR$composition [category cat.3$source]))

(7) (UR$morphism target)
    (= (UR$source target) 2-category)
    (= (UR$target target) KIF.MOR$function)
    (= (UR$composition [target KIF.MOR$source]) arrow)
    (= (UR$composition [target KIF.MOR$target]) object)
    (= target (UR$composition [category cat.3$target]))

(8) (UR$morphism composable-opspan)
    (= (UR$source composable-opspan) 2-category)
    (= (UR$target composable-opspan) KIF.DGM.OSPN.OBJ$opspan)
    (= (UR$composition [composable-opspan KIF.DGM.OSPN.OBJ$opvertex]) object)
    (= (UR$composition [composable-opspan KIF.DGM.OSPN.OBJ$opzeroth]) target)
    (= (UR$composition [composable-opspan KIF.DGM.OSPN.OBJ$opzeroth]) source)
    (= composable-opspan (UR$composition [category cat.3$composable-opspan]))

(9) (UR$morphism composable)
    (= (UR$source composable) 2-category)
    (= (UR$target composable) KIF.REL$relation)
    (= (UR$composition [composable KIF.REL$collection0]) arrow)
    (= (UR$composition [composable KIF.REL$collection1]) arrow)
    (= (UR$composition [composable KIF.REL$extent]) composable-pair)
    (= composable (UR$composition [composable-opspan KIF.DGM.OSPN.OBJ$relation]))
    (= composable (UR$composition [category cat.3$composable]))

(10) (UR$morphism composable-pair)
    (= (UR$source composable-pair) 2-category)
    (= (UR$target composable-pair) KIF.COL$collection)
    (= composable-pair (UR$composition [category cat.3$composable-pair]))
    (= composable-pair (UR$composition [category cat.3$composable-pair]))

(11) (UR$morphism zeroth)
    (= (UR$source zeroth) 2-category)
    (= (UR$target zeroth) KIF.MOR$function)
    (= (UR$composition [zeroth KIF.MOR$source]) composable-pair)
    (= (UR$composition [zeroth KIF.MOR$target]) arrow)
    (= zeroth (UR$composition [category cat.3$zeroth]))

(12) (UR$morphism first)
    (= (UR$source first) 2-category)
    (= (UR$target first) KIF.MOR$function)
    (= (UR$composition [first KIF.MOR$source]) composable-pair)
    (= (UR$composition [first KIF.MOR$target]) arrow)
    (= first (UR$composition [category cat.3$first]))

(13) (UR$morphism composition)
    (= (UR$source composition) 2-category)
```

```

(= (UR$target composition) KIF.MOR$function)
(= (UR$composition [composition KIF.MOR$source]) composable-pair)
(= (UR$composition [composition KIF.MOR$target]) arrow)
(= composition (UR$composition [category cat.3$composition]))
(forall (?K (2-category ?K))
  (and (= (KIF.MOR$composition [(composition ?K) (source ?K)])
        (KIF.MOR$composition [(zeroth ?K) (source ?K)]))
        (= (KIF.MOR$composition [(composition ?K) (target ?K)])
          (KIF.MOR$composition [(first ?K) (target ?K)]))))

(14) (UR$morphism identity)
(= (UR$source identity) 2-category)
(= (UR$target identity) KIF.MOR$function)
(= (UR$composition [identity KIF.MOR$source]) object)
(= (UR$composition [identity KIF.MOR$target]) arrow)
(= identity (UR$composition [category cat.3$identity]))
(forall (?K (2-category ?K))
  (and (= (KIF.MOR$composition [(identity ?K) (source ?K)])
        (KIF.MOR$identity object))
        (= (KIF.MOR$composition [(identity ?K) (target ?K)])
          (KIF.MOR$identity object))))

```

Composition satisfies the usual *associative law*.

```

(15) (forall (?f (arrow ?f) ?g (arrow ?g) ?h (arrow ?h))
      (composable ?f ?g) (composable ?g ?h))
(= (composition [?f (composition [?g ?h])])
    (composition [(composition [?f ?g]) ?h]))

```

Identity satisfies the usual *identity laws* with respect to composition.

```

(16) (forall (?f (arrow ?f))
      (and (= (composition [(identity (source ?f)) ?f]) ?f)
            (= (composition [?f (identity (target ?f))] ?f))))

```

## The Horizontal Category

The objects and 2-cells form a (very large) category  $K_{\text{horiz}}$  called the *horizontal category* of  $K$ . The composition and identity functions in  $K_{\text{horiz}}$  are renamed here as horizontal composition and horizontal identity.

```

(17) (UR$morphism horizontal-category)
(= (UR$source horizontal-category) 2-category)
(= (UR$target horizontal-category) cat.3$category)
(= (UR$composition [horizontal-category cat.3$object]) object)
(= (UR$composition [horizontal-category cat.3$morphism]) 2-cell)

(18) (UR$morphism horizontal-source) (UR$morphism source-object)
(= source-object horizontal-source)
(= (UR$source horizontal-source) 2-category)
(= (UR$target horizontal-source) KIF.MOR$function)
(= (UR$composition [horizontal-source KIF.MOR$source]) 2-cell)
(= (UR$composition [horizontal-source KIF.MOR$target]) object)
(= horizontal-source (UR$composition [horizontal-category cat.3$source]))

(19) (UR$morphism horizontal-target) (UR$morphism target-object)
(= target-object horizontal-target)
(= (UR$source horizontal-target) 2-category)
(= (UR$target horizontal-target) KIF.MOR$function)
(= (UR$composition [horizontal-target KIF.MOR$source]) 2-cell)
(= (UR$composition [horizontal-target KIF.MOR$target]) object)
(= horizontal-target (UR$composition [horizontal-category cat.3$target]))

```

A pair of 2-cells  $\sigma$  and  $\tau$  is *horizontally composable* when the target object of the first is the source object of the second,  $\sigma : f_0 \Rightarrow g_0 : A_0 \rightarrow A_1$  and  $\tau : f_1 \Rightarrow g_1 : A_1 \rightarrow A_2$ . There is a *horizontal composition* 2-cell  $\sigma \circ \tau : f_0 \circ f_1 \Rightarrow g_0 \circ g_1 : A_0 \rightarrow A_2$  (Figure 2h).

```

(20) (UR$morphism horizontally-composable-opspan)
(= (UR$source horizontally-composable-opspan) 2-category)
(= (UR$target horizontally-composable-opspan) KIF.DGM.OSPN.OBJ$opspan)

```

# The IFF Namespace of 2-Categories

Robert E. Kent

Page 4

October 4, 2004

```
(= (UR$composition [horizontally-composable-opspan KIF.DGM.OSPN.OBJ$opvertex]) arrow)
(= (UR$composition [horizontally-composable-opspan KIF.DGM.OSPN.OBJ$opzeroth]) target-arrow)
(= (UR$composition [horizontally-composable-opspan KIF.DGM.OSPN.OBJ$opzeroth]) source-arrow)
(= horizontally-composable-opspan
  (UR$composition [vertical-category cat.3$composable-opspan]))

(21) (UR$morphism horizontally-composable)
(= (UR$source horizontally-composable) 2-category)
(= (UR$target horizontally-composable) KIF.REL$relation)
(= (UR$composition [horizontally-composable KIF.REL$collection0]) 2-cell)
(= (UR$composition [horizontally-composable KIF.REL$collection1]) 2-cell)
(= (UR$composition [horizontally-composable KIF.REL$extent]) horizontally-composable-pair)
(= horizontally-composable
  (UR$composition [horizontally-composable-opspan KIF.DGM.OSPN.OBJ$relation]))
(= horizontally-composable (UR$composition [vertical-category cat.3$composable]))

(22) (UR$morphism horizontally-composable-pair)
(= (UR$source horizontally-composable-pair) 2-category)
(= (UR$target horizontally-composable-pair) KIF.COL$collection)
(= horizontally-composable-pair
  (UR$composition [horizontally-composable-opspan KIF.LIM.PBK.OBJ$pullback]))
(= horizontally-composable-pair (UR$composition [vertical-category cat.3$composable-pair]))

(23) (UR$morphism horizontal-zeroth)
(= (UR$source horizontal-zeroth) 2-category)
(= (UR$target horizontal-zeroth) KIF.MOR$function)
(= (UR$composition [horizontal-zeroth KIF.MOR$source]) horizontally-composable-pair)
(= (UR$composition [horizontal-zeroth KIF.MOR$target]) 2-cell)
(= horizontal-zeroth
  (UR$composition [horizontally-composable-opspan KIF.DGM.OSPN.OBJ$projection0]))
(= horizontal-zeroth (UR$composition [vertical-category cat.3$zeroth]))

(24) (UR$morphism horizontal-first)
(= (UR$source horizontal-first) 2-category)
(= (UR$target horizontal-first) KIF.MOR$function)
(= (UR$composition [horizontal-first KIF.MOR$source]) horizontally-composable-pair)
(= (UR$composition [horizontal-first KIF.MOR$target]) 2-cell)
(= horizontal-first
  (UR$composition [horizontally-composable-opspan KIF.DGM.OSPN.OBJ$projection1]))
(= horizontal-first (UR$composition [vertical-category cat.3$first]))

(25) (UR$morphism horizontal-composition)
(= (UR$source horizontal-composition) 2-category)
(= (UR$target horizontal-composition) KIF.MOR$function)
(= (UR$composition [horizontal-composition KIF.MOR$source]) horizontally-composable-pair)
(= (UR$composition [horizontal-composition KIF.MOR$target]) 2-cell)
(= horizontal-composition (UR$composition [vertical-category cat.3$composition]))
(forall (?K (2-category ?K))
  (and (= (KIF.MOR$composition [(horizontal-composition ?K) (source-arrow ?K)])
        (KIF.MOR$composition [(source-pair ?K) (composition ?K)]))
        (= (KIF.MOR$composition [(horizontal-composition ?K) (target-arrow ?K)])
        (KIF.MOR$composition [(target-pair ?K) (composition ?K)]))
        (= (KIF.MOR$composition [(horizontal-composition ?K) (source-object ?K)])
        (KIF.MOR$composition [(horizontal-zeroth ?K) (source-object ?K)]))
        (= (KIF.MOR$composition [(horizontal-composition ?K) (target-object ?K)])
        (KIF.MOR$composition [(horizontal-first ?K) (target-object ?K)])))))
```

For any object  $A$  there is a *horizontal identity* 2-cell  $1_A : id_A \Rightarrow id_A : A \rightarrow A$ .

```
(26) (UR$morphism horizontal-identity)
(= (UR$source horizontal-identity) 2-category)
(= (UR$target horizontal-identity) KIF.MOR$function)
(= (UR$composition [horizontal-identity KIF.MOR$source]) object)
(= (UR$composition [horizontal-identity KIF.MOR$target]) 2-cell)
(= horizontal-identity (UR$composition [vertical-category cat.3$identity]))
(forall (?K (2-category ?K))
  (and (= (KIF.MOR$composition [(horizontal-identity ?K) (source-arrow ?K)])
        (identity ?K))
        (= (KIF.MOR$composition [(horizontal-identity ?K) (target-arrow ?K)])
        (identity ?K))
        (= (KIF.MOR$composition [(horizontal-identity ?K) (source-object ?K)]))
```

```
(KIF.MOR$identity (object ?K))
(= (KIF.MOR$composition [(horizontal-identity ?K) (target-object ?K)])
   (KIF.MOR$identity (object ?K)))
```

Horizontal composition satisfies the usual associative law.

```
(27) (forall (?a (2-cell ?a) ?b (2-cell ?b) ?c (2-cell ?h)
             (horizontally-composable ?a ?b) (horizontally-composable ?b ?c))
      (= (horizontal-composition [?a (horizontal-composition [?b ?c])])
         (horizontal-composition [(horizontal-composition [?a ?b]) ?c])))
```

Horizontal identity satisfies the usual identity laws with respect to horizontal composition.

```
(28) (forall (?a (2-cell ?a))
      (and (= (horizontal-composition [(horizontal-identity (horizontal-source ?a)) ?a]) ?a)
           (= (horizontal-composition [?a (horizontal-identity (horizontal-target ?a))] ?a))))
```

## The Vertical Source and Target Functors

There are two (very large) functors, the vertical source functor  $\partial_0 = \mathit{vert-src} : \mathbf{K}_{\text{horiz}} \rightarrow \mathbf{K}$  and the vertical target functor  $\partial_1 = \mathit{vert-tgt} : \mathbf{K}_{\text{horiz}} \rightarrow \mathbf{K}$ , which agree on objects. The composition part of the assertion of functoriality states that (1) the vertical source of horizontal composition is the arrow composition of the vertical sources of the components, and (2) the vertical target of horizontal composition is the arrow composition of the vertical targets of the components. The identity part of the assertion of functoriality states that (1) the vertical source of horizontal identity of an object is the arrow identity of the object, and (2) the vertical target of horizontal identity of an object is the arrow identity of the object. The graph (type-preservation) part of the assertion of functoriality states that (1) the source of the vertical source (target) is the horizontal source, and (2) the target of the vertical source (target) is the horizontal target.

```
(29) (UR$morphism vertical-source-functor)
(= (UR$source vertical-source-functor) 2-category)
(= (UR$target vertical-source-functor) func.3$functor)
(= (UR$composition [vertical-source-functor func.3$object])
   (UR$composition [object KIF.MOR$identity]))
(= (UR$composition [vertical-source-functor func.3$morphism]) vertical-source)
```

```
(30) (UR$morphism source-pair)
(= (UR$source source-pair) 2-category)
(= (UR$target source-pair) KIF.MOR$function)
(= (UR$composition [source-pair KIF.MOR$source]) horizontally-composable-pair)
(= (UR$composition [source-pair KIF.MOR$target]) composable-pair)
(= source-pair (UR$composition [vertical-source-functor func.3$composable-pair]))
(forall (?K (2-category ?K))
  (and (= (KIF.MOR$composition [(source-pair ?K) (zeroth ?K)])
          (KIF.MOR$composition [(horizontal-zeroth ?K) (source-arrow ?K)]))
        (= (KIF.MOR$composition [(source-pair ?K) (first ?K)])
           (KIF.MOR$composition [(horizontal-first ?K) (source-arrow ?K)]))))
```

```
(31) (UR$morphism vertical-target-functor)
(= (UR$source vertical-target-functor) 2-category)
(= (UR$target vertical-target-functor) func.3$functor)
(= (UR$composition [vertical-target-functor func.3$object])
   (UR$composition [object KIF.MOR$identity]))
(= (UR$composition [vertical-target-functor func.3$morphism]) vertical-target)
```

```
(32) (UR$morphism target-pair)
(= (UR$source target-pair) 2-category)
(= (UR$target target-pair) KIF.MOR$function)
(= (UR$composition [target-pair KIF.MOR$source]) horizontally-composable-pair)
(= (UR$composition [target-pair KIF.MOR$target]) composable-pair)
(= target-pair (UR$composition [vertical-target-functor func.3$composable-pair]))
(forall (?K (2-category ?K))
  (and (= (KIF.MOR$composition [(target-pair ?K) (zeroth ?K)])
          (KIF.MOR$composition [(horizontal-zeroth ?K) (target-arrow ?K)]))
        (= (KIF.MOR$composition [(target-pair ?K) (first ?K)])
           (KIF.MOR$composition [(horizontal-first ?K) (target-arrow ?K)]))))
```

```
(33) (forall (?K (2-category ?K))
      (and (= (KIF.MOR$composition [(horizontal-identity ?K) (vertical-source ?K)]))
```

```

(identity ?K)
(= (KIF.MOR$composition [(horizontal-identity ?K) (vertical-target ?K)])
(identity ?K)))
(34) (forall (?K (2-category ?K))
      (and (= (KIF.FTN$composition [(vertical-source ?K) (source ?K)])
              (horizontal-source ?K))
            (= (KIF.FTN$composition [(vertical-source ?K) (target ?K)])
              (horizontal-target ?K))
            (= (KIF.FTN$composition [(vertical-target ?K) (source ?K)])
              (horizontal-source ?K))
            (= (KIF.FTN$composition [(vertical-target ?K) (target ?K)])
              (horizontal-target ?K))))

```

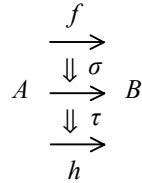


Figure 2v: Vertical Composition

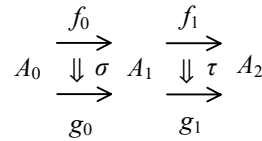


Figure 2h: Horizontal Composition

## The Vertical Category

The arrows and 2-cells form a (very large) category  $K_{\text{vert}}$  called the *vertical category* of  $K$ . The composition and identity functions in  $K_{\text{vert}}$  are renamed here as vertical composition and vertical identity.

```

(35) (UR$morphism vertical-category)
      (= (UR$source vertical-category) 2-category)
      (= (UR$target vertical-category) cat.3$category)
      (= (UR$composition [vertical-category cat.3$object]) arrow)
      (= (UR$composition [vertical-category cat.3$morphism]) 2-cell)

(36) (UR$morphism vertical-source) (UR$morphism source-arrow)
      (= source-arrow vertical-source)
      (= (UR$source vertical-source) 2-category)
      (= (UR$target vertical-source) KIF.MOR$function)
      (= (UR$composition [vertical-source KIF.MOR$source]) 2-cell)
      (= (UR$composition [vertical-source KIF.MOR$target]) arrow)
      (= vertical-source (UR$composition [vertical-category cat.3$source]))

(37) (UR$morphism vertical-target) (UR$morphism target-arrow)
      (= target-arrow vertical-target)
      (= (UR$source vertical-target) 2-category)
      (= (UR$target vertical-target) KIF.MOR$function)
      (= (UR$composition [vertical-target KIF.MOR$source]) 2-cell)
      (= (UR$composition [vertical-target KIF.MOR$target]) arrow)
      (= vertical-target (UR$composition [vertical-category cat.3$target]))

```

A pair of 2-cells  $\sigma$  and  $\tau$  is *vertically composable* when the target arrow of the first is the source arrow of the second,  $\sigma : f \Rightarrow g : A \rightarrow B$  and  $\tau : g \Rightarrow h : A \rightarrow B$ . The *vertical composition*  $\sigma \bullet \tau : f \Rightarrow h : A \rightarrow B$  of two vertically composable natural transformations (Figure 2v) is defined.

```

(38) (UR$morphism vertically-composable-opspan)
      (= (UR$source vertically-composable-opspan) 2-category)
      (= (UR$target vertically-composable-opspan) KIF.DGM.OSPN.OBJ$opspan)
      (= (UR$composition [vertically-composable-opspan KIF.DGM.OSPN.OBJ$opvertex]) arrow)
      (= (UR$composition [vertically-composable-opspan KIF.DGM.OSPN.OBJ$opzeroth]) target-arrow)
      (= (UR$composition [vertically-composable-opspan KIF.DGM.OSPN.OBJ$opzeroth]) source-arrow)
      (= vertically-composable-opspan
          (UR$composition [vertical-category cat.3$composable-opspan]))

(39) (UR$morphism vertically-composable)
      (= (UR$source vertically-composable) 2-category)
      (= (UR$target vertically-composable) KIF.REL$relation)
      (= (UR$composition [vertically-composable KIF.REL$collection0]) 2-cell)

```

# The IFF Namespace of 2-Categories

Robert E. Kent

Page 7

October 4, 2004

```
(= (UR$composition [vertically-composable KIF.REL$collection1]) 2-cell)
(= (UR$composition [vertically-composable KIF.REL$extent]) vertically-composable-pair)
(= vertically-composable
  (UR$composition [vertically-composable-opspan KIF.DGM.OSPN.OBJ$relation]))
(= vertically-composable (UR$composition [vertical-category cat.3$composable]))

(40) (UR$morphism vertically-composable-pair)
(= (UR$source vertically-composable-pair) 2-category)
(= (UR$target vertically-composable-pair) KIF.COL$collection)
(= vertically-composable-pair
  (UR$composition [vertically-composable-opspan KIF.LIM.PBK.OBJ$pullback]))
(= vertically-composable-pair (UR$composition [vertical-category cat.3$composable-pair]))

(41) (UR$morphism vertical-zeroth)
(= (UR$source vertical-zeroth) 2-category)
(= (UR$target vertical-zeroth) KIF.MOR$function)
(= (UR$composition [vertical-zeroth KIF.MOR$source]) vertically-composable-pair)
(= (UR$composition [vertical-zeroth KIF.MOR$target]) 2-cell)
(= vertical-zeroth
  (UR$composition [vertically-composable-opspan KIF.DGM.OSPN.OBJ$projection0]))
(= vertical-zeroth (UR$composition [vertical-category cat.3$zerorth]))

(42) (UR$morphism vertical-first)
(= (UR$source vertical-first) 2-category)
(= (UR$target vertical-first) KIF.MOR$function)
(= (UR$composition [vertical-first KIF.MOR$source]) vertically-composable-pair)
(= (UR$composition [vertical-first KIF.MOR$target]) 2-cell)
(= vertical-first
  (UR$composition [vertically-composable-opspan KIF.DGM.OSPN.OBJ$projection1]))
(= vertical-first (UR$composition [vertical-category cat.3$first]))

(43) (UR$morphism vertical-composition)
(= (UR$source vertical-composition) 2-category)
(= (UR$target vertical-composition) KIF.MOR$function)
(= (UR$composition [vertical-composition KIF.MOR$source]) vertically-composable-pair)
(= (UR$composition [vertical-composition KIF.MOR$target]) 2-cell)
(= vertical-composition (UR$composition [vertical-category cat.3$composition]))
(forall (?K (2-category ?K))
  (and (= (KIF.MOR$composition [(vertical-composition ?K) (source-arrow ?K)])
        (KIF.MOR$composition [(vertical-zeroth ?K) (source-arrow ?K)]))
        (= (KIF.MOR$composition [(vertical-composition ?K) (target-arrow ?K)])
          (KIF.MOR$composition [(vertical-first ?K) (target-arrow ?K)]))
        (= (KIF.MOR$composition [(vertical-composition ?K) (source-object ?K)])
          (KIF.MOR$composition [(vertical-zeroth ?K) (source-object ?K)]))
        (= (KIF.MOR$composition [(vertical-composition ?K) (target-object ?K)])
          (KIF.MOR$composition [(vertical-zeroth ?K) (target-object ?K)]))))
```

For any arrow  $f: A \rightarrow B$  there is a *vertical identity* 2-cell  $1_f: f \Rightarrow f: A \rightarrow B$ .

```
(44) (UR$morphism vertical-identity)
(= (UR$source vertical-identity) 2-category)
(= (UR$target vertical-identity) KIF.MOR$function)
(= (UR$composition [vertical-identity KIF.MOR$source]) arrow)
(= (UR$composition [vertical-identity KIF.MOR$target]) 2-cell)
(= vertical-identity (UR$composition [vertical-category cat.3$identity]))
(forall (?K (2-category ?K))
  (and (= (KIF.MOR$composition [(vertical-identity ?K) (source-arrow ?K)])
        (KIF.MOR$identity (arrow ?K)))
        (= (KIF.MOR$composition [(vertical-identity ?K) (target-arrow ?K)])
          (KIF.MOR$identity (arrow ?K)))
        (= (KIF.MOR$composition [(vertical-identity ?K) (source-object ?K)])
          (source ?K))
        (= (KIF.MOR$composition [(vertical-identity ?K) (target-object ?K)])
          (target ?K))))
```

Vertical composition satisfies the usual *associative law*.

```
(45) (forall (?a (2-cell ?a) ?b (2-cell ?b) ?c (2-cell ?h)
  (vertically-composable ?a ?b) (vertically-composable ?b ?c))
  (= (vertical-composition [(?a (vertical-composition [?b ?c])])
    (vertical-composition [(vertical-composition [?a ?b]) ?c])))
```

Vertical identity satisfies the usual *identity laws* with respect to vertical composition.

```
(46) (forall (?a (2-cell ?a))
      (and (= (vertical-composition [(vertical-identity (vertical-source ?a)) ?a]) ?a)
            (= (vertical-composition [?a (vertical-identity (vertical-target ?a))] ?a))))
```

## Additional Properties

For any object  $A$ , the horizontal identity  $1_A : id_A \Rightarrow id_A : A \rightarrow A$  is the vertical identity  $1_A = 1_{id_A}$  of the arrow identity  $id_A : A \rightarrow A$ .

```
(47) (forall (?K (2-category ?K))
      (= (horizontal-identity ?K)
         (KIF.MOR$composition [(identity ?K) (vertical-identity ?K)])))
```

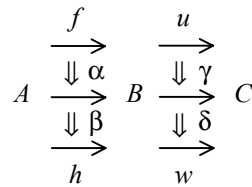


Figure 3: Interchangeable 2-cells

For any four 2-cells  $\alpha : f \Rightarrow g : A \rightarrow B$ ,  $\beta : g \Rightarrow h : A \rightarrow B$ ,  $\gamma : u \Rightarrow v : B \rightarrow C$  and  $\delta : v \Rightarrow w : B \rightarrow C$ , where the pairs  $(\alpha, \beta)$ ,  $(\gamma, \delta)$  and  $(\alpha \circ \gamma, \beta \circ \delta)$  are vertically composable, and the pairs  $(\alpha, \gamma)$ ,  $(\beta, \delta)$  and  $(\alpha \bullet \beta, \gamma \bullet \delta)$  are horizontally composable, we have the *interchange law*:  $(\alpha \bullet \beta) \circ (\gamma \bullet \delta) = (\alpha \circ \gamma) \bullet (\beta \circ \delta)$ .

```
(48) (forall (?K (2-category ?K))
      ?a ((2-cell ?K) ?a) ?b ((2-cell ?K) ?b) ?c ((2-cell ?K) ?c) ?d ((2-cell ?K) ?d)
      ((vertically-composable ?K) ?a ?b) ((vertically-composable ?K) ?c ?d)
      ((horizontally-composable ?K) ?a ?c) ((horizontally-composable ?K) ?b ?d)
      ((vertically-composable ?K)
        ((horizontal-composition ?K) [?a ?c]) ((horizontal-composition ?K) [?b ?d])))
      ((horizontally-composable ?K)
        ((vertical-composition ?K) [?a ?b]) ((vertical-composition ?K) [?c ?d])))
      (= ((horizontal-composition ?K)
          [(vertical-composition ?K) [?a ?b]) (vertical-composition ?K) [?c ?d]])
          ((vertical-composition ?K)
            [(horizontal-composition ?K) [?a ?c]) (horizontal-composition ?K) [?b ?d]))))
```

For a composable pair of arrows, the horizontal composite of the two vertical identities is the vertical identity of the arrow composite.

```
(49) (forall (?K (2-category ?K))
      ?f ((arrow ?K) ?f) ?g ((arrow ?K) ?g)
      ((composable ?K) ?f ?g))
      (= ((horizontal-composition ?K)
          [(vertical-identity ?K) ?f] ((vertical-identity ?K) ?g)))
          ((vertical-identity ?K) (composition ?K) [?f ?g])))
```

## Example: The Category of Categories

The following declaration should be made in the upper metalevel Category Theory (meta) Ontology (IFF-CAT). We make it again here for illustrative purposes. Verification of its properties uses the axiomatization in IFF-CAT. The category of (large) categories  $\mathbf{Cat}$  is a 2-category. The objects of  $\mathbf{Cat}$  are the (large) categories, the arrows of  $\mathbf{Cat}$  are the (large) functors, and the 2-cells of  $\mathbf{Cat}$  are the (large) natural transformations.

```
(50) (2-category Category)
      (= (object Category) CAT$category)
      (= (arrow Category) FUNC$functor)
      (= (2-cell Category) NAT$natural-transformation)
      (= (source Category) FUNC$source)
      (= (target Category) FUNC$target)
```

# The IFF Namespace of 2-Categories

Robert E. Kent

Page 9

October 4, 2004

```
(= (composition Category) FUNC$composition)
(= (identity Category) FUNC$identity)
(= (source-arrow Category) NAT$source-functor)
(= (target-arrow Category) NAT$target-functor)
(= (source-object Category) NAT$source-category)
(= (target-object Category) NAT$target-category)
(= (horizontal-composition Category) NAT$horizontal-composition)
(= (horizontal-identity Category) NAT$horizontal-identity)
(= (vertical-composition Category) NAT$vertical-composition)
(= (vertical-identity Category) NAT$vertical-identity)
```