```
(*****************************************************************************************************************************)
(* GeneralizedArrowFromReification:                                                                                          *)
(*                                                                                                                           *)
(*   Turn a reification into a generalized arrow                                                                             *)
(*                                                                                                                           *)
(*****************************************************************************************************************************)

Generalizable All Variables.
Require Import Preamble.
Require Import General.
Require Import Categories_ch1_3.
Require Import Functors_ch1_4.
Require Import Isomorphisms_ch1_5.
Require Import ProductCategories_ch1_6_1.
Require Import OppositeCategories_ch1_6_2.
Require Import Enrichment_ch2_8.
Require Import Subcategories_ch7_1.
Require Import NaturalTransformations_ch7_4.
Require Import NaturalIsomorphisms_ch7_5.
Require Import BinoidalCategories.
Require Import PreMonoidalCategories.
Require Import PreMonoidalCenter.
Require Import MonoidalCategories_ch7_8.
Require Import Coherence_ch7_8.
Require Import Enrichment_ch2_8.
Require Import Enrichments.
Require Import RepresentableStructure_ch7_2.
Require Import Reification.
Require Import GeneralizedArrow.

Section GArrowFromReification.

  Definition binoidalcat_iso `{bc:BinoidalCat}{a1 b1 a2 b2:bc} (i1:a1≅a2)(i2:b1≅b2) : (a1⊗b1)≅(a2⊗b2) :=
    iso_comp
      (functors_preserve_isos (- ⋉ b1) i1 )
      (functors_preserve_isos (a2 ⋊ -) i2).

  Context  `(K           : SurjectiveEnrichment)
           `(CMon        : MonicEnrichment C)
            (CM          : MonoidalEnrichment C)
            (reification : Reification K C (pmon_I (enr_c_pm C))).
```

```
Fixpoint garrow_fobj (vk:senr_v K) : C :=
  match vk with
  | T_Leaf None     => enr_c_i C
  | T_Leaf (Some a) => match a with (a1,a2) => reification_r reification a1 a2 end
  | t1,,t2          => bin_obj(BinoidalCat:=enr_c_bin C) (garrow_fobj t1) (garrow_fobj t2)
  end.

Fixpoint homset_tensor_iso (vk:enr_v_mon K) : reification vk ≅ enr_c_i C ~~> garrow_fobj vk :=
  match vk as VK return reification VK ≅ enr_c_i C ~~> garrow_fobj VK with
  | T_Leaf None     => (mf_i(PreMonoidalFunctor:=reification))^-1 >>≅>> (mf_i(PreMonoidalFunctor:=CM))
  | T_Leaf (Some a) => match a as A
                         return reification (T_Leaf (Some A)) ≅ enr_c_i C ~~> garrow_fobj (T_Leaf (Some A)) with
                         (s,s0) => iso_inv _ _ (ni_iso (reification_commutes reification s) s0)
                       end
  | t1,,t2          => (ni_iso (@mf_first _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ reification _) _)^-1 >>≅>>
                       (binoidalcat_iso (homset_tensor_iso t1) (homset_tensor_iso t2)) >>≅>>
                       (ni_iso (mf_first(PreMonoidalFunctor:=CM) (garrow_fobj t2)) _)
  end.

Definition HomFunctor_fullimage := FullImage CM.

(* R' is a functor from the domain of the reification functor
 * to the full subcategory in the range of the [host language's] Hom(I,-) functor *)
Instance R' : Functor (FullImage (reification_rstar reification)) HomFunctor_fullimage garrow_fobj :=
  { fmor := fun a b (f:a~~{FullImage (reification_rstar reification)}~~>b) =>
    (#(homset_tensor_iso a)^-1 >>> f >>> #(homset_tensor_iso b))
  }.
  abstract (intros; simpl;
            apply comp_respects; try reflexivity;
            apply comp_respects; try reflexivity;
            auto).
  abstract (intros; simpl;
    setoid_rewrite right_identity;
    apply iso_comp2).
  abstract (intros;
    simpl;
    repeat setoid_rewrite <- associativity;
    apply comp_respects; try reflexivity;
    repeat setoid_rewrite associativity;
    apply comp_respects; try apply reflexivity;
```

```
    apply comp_respects; try apply reflexivity;
    eapply transitivity; [ symmetry; apply associativity | idtac ];
    eapply transitivity; [ idtac | apply left_identity ];
    apply comp_respects; try apply reflexivity;
    apply iso_comp1).
    Defined.

(* the "step2_functor" is the section of the Hom(I,-) functor *)
Definition step2_functor :=
  ff_functor_section_functor _ (me_full(MonicEnrichment:=CMon)) (me_faithful(MonicEnrichment:=CMon)).

Definition garrow_functor :=
  RestrictToImage (reification_rstar reification) >>>> (R' >>>> step2_functor).

Lemma iso_id_lemma1 `{C':Category}(a b:C')(f:a~~{C'}~~>b) : #(iso_id a) >>> f ~~ f.
  simpl.
  apply left_identity.
  Qed.

Lemma iso_id_lemma2 `{C':Category}(a b:C')(f:b~~{C'}~~>a) : f >>> #(iso_id a) ~~ f.
  simpl.
  apply right_identity.
  Qed.

Lemma full_roundtrip : forall a b (f:a~>b), me_homfunctor \ (ff_functor_section_fmor me_homfunctor me_full f) ~~ f.
  intros.
  unfold ff_functor_section_fmor.
  set (me_full a b f) as full.
  destruct full.
  apply e.
  Qed.

Opaque UnderlyingFunctor.
Instance garrow_first a :
  (garrow_functor >>>> bin_first(BinoidalCat:=enr_c_bin C) (R' a)) <~~~>
  (bin_first(BinoidalCat:=enr_v_pmon K) a >>>> garrow_functor) :=
  { ni_iso := fun a => iso_id _ }.
  intros.
    etransitivity.  apply iso_id_lemma1.  symmetry.
    etransitivity.  apply iso_id_lemma2.  symmetry.
```

```
Opaque Underlying.
unfold garrow_functor.
  unfold functor_comp at 1.
  unfold functor_comp at 1.
  Opaque functor_comp. simpl. Transparent functor_comp.

symmetry.
  eapply transitivity.
  apply (functor_comp_assoc (RestrictToImage reification) (R' >>>> step2_functor) (ebc_first (R' a)) f).
  unfold functor_comp at 1.
  unfold functor_comp at 1.
  Opaque functor_comp. simpl. Transparent functor_comp.

symmetry.
  eapply transitivity.
  set (ni_commutes (mf_first(PreMonoidalFunctor:=reification_rstar reification) a) f) as qq.
  unfold functor_comp in qq.
  simpl in qq.
  apply iso_shift_right' in qq.
  apply (fmor_respects(R' >>>> step2_functor) _ _ qq).

apply (me_faithful(MonicEnrichment:=CMon)).
  symmetry.
  unfold fmor at 1.
  eapply transitivity.
  set (ni_commutes (mf_first(PreMonoidalFunctor:=CM) (R' a))) as zz.
  unfold functor_comp in zz; unfold functor_fobj in zz; simpl in zz.
  set (zz _ _ ((R' >>>> step2_functor) \ (reification \ f))) as zz'.
  apply iso_shift_right' in zz'.
  apply zz'.

unfold functor_comp; simpl.

symmetry.
  eapply transitivity.
  set full_roundtrip as full_roundtrip'.
  unfold fmor in full_roundtrip'.
  simpl in full_roundtrip'.
  apply full_roundtrip'.

set (@iso_shift_right') as q. simpl in q. apply q. clear q.
```

```
set (@iso_shift_left) as q.   simpl in q. apply q. clear q.

symmetry.
  eapply transitivity.
  set full_roundtrip as full_roundtrip'.
  unfold fmor in full_roundtrip'.
  simpl in full_roundtrip'.
  apply (fun a' b' f z => fmor_respects (bin_first(BinoidalCat:=enr_v_bin C) z) _ _ (full_roundtrip' a' b' f)).
  symmetry.

intros.
  unfold bin_obj.
  unfold senr_v_bobj.

setoid_rewrite <- associativity.
  setoid_rewrite <- associativity.
  setoid_rewrite <- associativity.
  setoid_rewrite <- associativity.
  simpl.
  setoid_rewrite <- associativity.
  etransitivity.
  apply iso_comp1_left.

eapply transitivity.
  eapply comp_respects; [ idtac | reflexivity ].
  eapply comp_respects; [ idtac | reflexivity ].
  eapply comp_respects; [ idtac | reflexivity ].
  eapply comp_respects; [ idtac | reflexivity ].
  apply iso_comp1_right.

eapply symmetry.
  eapply transitivity.
  setoid_rewrite <- fmor_preserves_comp.
  setoid_rewrite <- fmor_preserves_comp.
  eapply reflexivity.
  eapply symmetry.

eapply transitivity.
  apply associativity.
  eapply transitivity.
```

```
  eapply comp_respects; [ reflexivity | idtac ].
  apply (centralmor_first(CentralMorphism:=commutative_central(CommutativeCat:=enr_v_mon C) _)).
  eapply transitivity.
  eapply symmetry.
  apply associativity.
  apply comp_respects; try apply reflexivity.

eapply transitivity.
  eapply comp_respects; [ idtac | reflexivity ].
  eapply comp_respects; [ idtac | reflexivity ].
  eapply symmetry.
  eapply associativity.
  eapply transitivity.
  eapply comp_respects; [ idtac | reflexivity ].
  eapply comp_respects; [ idtac | reflexivity ].
  eapply comp_respects; [ idtac | reflexivity ].
  apply iso_comp1_left.

eapply transitivity.
  eapply comp_respects; [ idtac | reflexivity ].
  eapply transitivity.
  eapply comp_respects.
  eapply symmetry.
  eapply associativity.
  eapply reflexivity.
  apply iso_comp1_left.

eapply transitivity.
  eapply comp_respects; [ idtac | reflexivity ].
  eapply comp_respects; [ idtac | reflexivity ].
  eapply symmetry.
  apply (centralmor_first(CentralMorphism:=commutative_central(CommutativeCat:=enr_v_mon C) _)).
  eapply transitivity.
  eapply comp_respects; [ idtac | reflexivity ].
  eapply transitivity.
  apply associativity.
  eapply comp_respects; [ reflexivity | idtac ].
  eapply symmetry.
  apply (centralmor_first(CentralMorphism:=commutative_central(CommutativeCat:=enr_v_mon C) _)).
  eapply transitivity.
  eapply transitivity.
```

```
    apply associativity.
    eapply comp_respects; [ reflexivity | idtac ].
    eapply transitivity.
    apply associativity.
    eapply transitivity; [ idtac | apply right_identity ].
    eapply comp_respects; [ reflexivity | idtac ].
    eapply transitivity.
    unfold functor_fobj.
    apply fmor_preserves_comp.
    setoid_rewrite iso_comp2.
    apply fmor_preserves_id.

  apply comp_respects.
    reflexivity.
    reflexivity.
    Defined.

Instance garrow_second a :
     (garrow_functor >>>> bin_second(BinoidalCat:=enr_c_bin C) (R' a))
  <~~~>  (bin_second(BinoidalCat:=enr_v_pmon K) a >>>> garrow_functor) :=
  { ni_iso := fun a => iso_id _ }.

  intros.
    etransitivity.  apply iso_id_lemma1.  symmetry.
    etransitivity.  apply iso_id_lemma2.  symmetry.

  Opaque Underlying.
  unfold garrow_functor.
    unfold functor_comp at 1.
    unfold functor_comp at 1.
    Opaque functor_comp. simpl. Transparent functor_comp.

  symmetry.
    eapply transitivity.
    apply (functor_comp_assoc (RestrictToImage reification) (R' >>>> step2_functor) (ebc_second (R' a)) f).
    unfold functor_comp at 1.
    unfold functor_comp at 1.
    Opaque functor_comp. simpl. Transparent functor_comp.

  symmetry.
    eapply transitivity.
```

```
  set (ni_commutes (mf_second(PreMonoidalFunctor:=reification_rstar reification) a) f) as qq.
  unfold functor_comp in qq.
  simpl in qq.
  apply iso_shift_right' in qq.
  apply (fmor_respects(R' >>>> step2_functor) _ _ qq).

apply (me_faithful(MonicEnrichment:=CMon)).
  symmetry.
  unfold fmor at 1.
  eapply transitivity.
  set (ni_commutes (mf_second(PreMonoidalFunctor:=CM) (R' a))) as zz.
  unfold functor_comp in zz; unfold functor_fobj in zz; simpl in zz.
  set (zz _ _ ((R' >>>> step2_functor) \ (reification \ f))) as zz'.
  apply iso_shift_right' in zz'.
  apply zz'.

unfold functor_comp; simpl.

symmetry.
  eapply transitivity.
  set full_roundtrip as full_roundtrip'.
  unfold fmor in full_roundtrip'.
  simpl in full_roundtrip'.
  apply full_roundtrip'.

set (@iso_shift_right') as q. simpl in q. apply q. clear q.

set (@iso_shift_left) as q.   simpl in q. apply q. clear q.

symmetry.
  eapply transitivity.
  set full_roundtrip as full_roundtrip'.
  unfold fmor in full_roundtrip'.
  simpl in full_roundtrip'.
  apply (fun a' b' f z => fmor_respects (bin_second(BinoidalCat:=enr_v_bin C) z) _ _ (full_roundtrip' a' b' f)).
  symmetry.

intros.
  unfold bin_obj.
  unfold senr_v_bobj.
```

```
setoid_rewrite <- associativity.
  setoid_rewrite <- associativity.
  setoid_rewrite <- associativity.
  setoid_rewrite <- associativity.
  simpl.
  setoid_rewrite <- associativity.
  etransitivity.
  eapply transitivity.
    apply associativity.
    eapply transitivity; [ idtac | apply right_identity ].
    apply comp_respects; [ reflexivity | idtac ].
    etransitivity.
    apply comp_respects; [ idtac | reflexivity ].
    apply (mf_consistent(PreMonoidalFunctor:=CM)).
    apply iso_comp1.

eapply transitivity.
  eapply comp_respects; [ idtac | reflexivity ].
  eapply comp_respects; [ idtac | reflexivity ].
  eapply comp_respects; [ idtac | reflexivity ].
  eapply comp_respects; [ idtac | reflexivity ].
  eapply transitivity.
    eapply symmetry.
    eapply associativity.
  eapply transitivity; [ idtac | apply left_identity ].
  eapply comp_respects; [ idtac | reflexivity ].
  eapply transitivity.
    eapply comp_respects; [ idtac | reflexivity ].
    eapply symmetry.
    apply (mf_consistent(PreMonoidalFunctor:=CM)).
    apply iso_comp1.

eapply symmetry.
  eapply transitivity.
  setoid_rewrite <- fmor_preserves_comp.
  setoid_rewrite <- fmor_preserves_comp.
  eapply reflexivity.
  eapply symmetry.

apply comp_respects; try reflexivity.
```

```
eapply transitivity.
  apply associativity.
  eapply transitivity.
  apply associativity.
  eapply transitivity.
  apply associativity.
  eapply transitivity.
  apply associativity.
  apply comp_respects; try reflexivity.

eapply transitivity.
  eapply comp_respects; [ reflexivity | idtac ].
  eapply transitivity.
  eapply comp_respects; [ idtac | reflexivity ].
  apply mf_consistent.
  eapply transitivity.
  eapply comp_respects; [ reflexivity | idtac ].
    apply associativity.
  apply iso_comp1_right.

eapply transitivity.
  eapply comp_respects; [ reflexivity | idtac ].
  eapply transitivity.
    apply associativity.
  eapply comp_respects; [ reflexivity | idtac ].
  eapply transitivity.
    eapply symmetry.
    apply associativity.
  eapply transitivity; [ idtac | apply left_identity ].
  eapply comp_respects; [ idtac | reflexivity ].
  eapply transitivity.
    eapply comp_respects; [ idtac | reflexivity ].
    eapply symmetry.
    eapply (mf_consistent(PreMonoidalFunctor:=reification)).
  apply iso_comp1.

eapply transitivity.
  eapply comp_respects; [ reflexivity | idtac ].
  eapply symmetry.
  apply (centralmor_first(CentralMorphism:=commutative_central(CommutativeCat:=enr_v_mon C) _)).
  eapply transitivity; [ idtac | apply right_identity ].
```

```
    eapply transitivity.
      eapply symmetry.
      apply associativity.
    eapply transitivity.
      eapply comp_respects; [ idtac | reflexivity ].
      unfold functor_fobj.
      apply (fmor_preserves_comp (bin_first(BinoidalCat:=enr_v_bin C) (reification_rstar_obj reification A))).

    apply symmetry.
      eapply transitivity.
      apply right_identity.
      apply symmetry.
      eapply transitivity; [ idtac | apply left_identity ].
      apply comp_respects; [ idtac | reflexivity ].

    eapply transitivity.
      Focus 2.
      eapply (fmor_preserves_id (bin_first(BinoidalCat:=enr_v_bin C) (reification_rstar_obj reification A))).
      idtac.
      apply (fmor_respects (bin_first(BinoidalCat:=enr_v_bin C) (reification_rstar_obj reification A))).
      apply iso_comp2.
    Defined.

Implicit Arguments mf_first [[Ob] [Hom] [C1] [bin_obj'] [bc] [I1] [PM1] [Ob0] [Hom0] [C2] [bin_obj'0] [bc0] [I2] [PM2] [fobj] [F]].
Implicit Arguments mf_second [[Ob] [Hom] [C1] [bin_obj'] [bc] [I1] [PM1] [Ob0] [Hom0] [C2] [bin_obj'0] [bc0] [I2] [PM2] [fobj] [F]].
Implicit Arguments mf_i [[Ob] [Hom] [C1] [bin_obj'] [bc] [I1] [PM1] [Ob0] [Hom0] [C2] [bin_obj'0] [bc0] [I2] [PM2] [fobj] [F]].

  Lemma assoc_lemma1 : forall a b c,
       iso_backward ((mf_first reification c) (a ⊗ b)) >>>
       iso_backward ((mf_second reification a) b) ⋉ reification c >>>
      #(pmon_assoc(PreMonoidalCat:=enr_v_mon C) (reification a) (reification c) (reification_rstar_obj reification b)) >>>
      reification a ⋊ #((mf_first reification c) b) >>>
      #((mf_second reification a) (b ⊗ c)) ~~
      reification \ #((pmon_assoc(PreMonoidalCat:=enr_v_mon K) a c) b).

    intros.
      eapply transitivity.
        apply associativity.
        eapply transitivity.
        apply associativity.
```

```
      eapply transitivity.
      apply associativity.
      apply symmetry.
      set (@iso_shift_right') as q.
      simpl in q.
      apply q.
      clear q.

    eapply symmetry.
      eapply transitivity.
      apply comp_respects; [ reflexivity | idtac ].
      set (mf_assoc(PreMonoidalFunctor:=reification)) as q.
      eapply transitivity.
      eapply symmetry.
      apply associativity.
      apply q.

    eapply transitivity.
      eapply symmetry.
      apply associativity.
      apply comp_respects; try reflexivity.
      eapply transitivity.
      eapply symmetry.
      apply associativity.
      eapply transitivity; [ idtac | apply left_identity ].
      apply comp_respects; try reflexivity.

    eapply transitivity.
      apply (fmor_preserves_comp (bin_first(BinoidalCat:=enr_v_bin C) _)).
    eapply transitivity; [ idtac |
      apply (fmor_preserves_id (bin_first(BinoidalCat:=enr_v_bin C) _)) ].
    apply (fmor_respects (bin_first(BinoidalCat:=enr_v_bin C) _)).
    apply iso_comp2.
    Qed.

Lemma assoc_lemma2 : forall a b c,
     iso_backward ((mf_first CM (garrow_fobj c)) _) >>>
     (bin_first(BinoidalCat:=enr_v_bin C) _ \ iso_backward ((mf_second CM (garrow_fobj a)) (garrow_fobj b))) >>>
     (#((pmon_assoc(PreMonoidalCat:=enr_v_mon C) _ _) _) >>>
      (bin_second(BinoidalCat:=enr_v_bin C) _ \
       #((mf_first CM (garrow_fobj c)) (garrow_fobj b)))) >>>
```

```
     #(mf_second CM (garrow_fobj a)) _) ~~
     HomFunctor C _ \ #(pmon_assoc(PreMonoidalCat:=enr_c_pm C) (garrow_fobj a) (garrow_fobj c)) (garrow_fobj b)).
  intros.
    eapply transitivity.
      apply associativity.
      eapply transitivity.
      apply associativity.
      set (@iso_shift_right') as q.
      apply symmetry.
      simpl in q.
      apply q.
      clear q.

    eapply symmetry.
      eapply transitivity.
      apply comp_respects; [ reflexivity | idtac ].
      set (mf_assoc(PreMonoidalFunctor:=CM)) as q.
      apply q.

    eapply transitivity.
      eapply symmetry.
      apply associativity.
      apply comp_respects; try reflexivity.
      eapply transitivity.
        eapply symmetry.
        apply associativity.
      eapply transitivity; [ idtac | apply left_identity ].
      apply comp_respects; try reflexivity.

    eapply transitivity.
      apply (fmor_preserves_comp (bin_first(BinoidalCat:=enr_v_bin C) _)).
    eapply transitivity; [ idtac |
      apply (fmor_preserves_id (bin_first(BinoidalCat:=enr_v_bin C) _)) ].
    apply (fmor_respects (bin_first(BinoidalCat:=enr_v_bin C) _)).
    apply iso_comp2.
    Qed.

Lemma assoc_coherent (a b c : enr_v K) :
 (#(pmon_assoc(PreMonoidalCat:=enr_c_pm C)
   (garrow_functor a) (garrow_functor c)) (garrow_fobj b)) >>> garrow_functor a ⋊ #(garrow_first c) b)) >>>
 #(garrow_second a) (b ⊗ c)) ~~
```

```
(#(garrow_second a) b) ⋉ garrow_functor c >>>
  #(garrow_first c) (a ⊗ b))) >>> garrow_functor \ #(pmon_assoc(PreMonoidalCat:=enr_v_mon K) a c) b).
 Opaque Underlying.
 eapply transitivity.
   eapply transitivity; [ idtac | apply right_identity ].
   eapply comp_respects; [ eapply reflexivity | idtac ].
   reflexivity.

 apply symmetry.
   eapply transitivity.
   eapply transitivity; [ idtac | apply left_identity ].
   eapply comp_respects; [ idtac | eapply reflexivity ].
   eapply transitivity; [ idtac | apply right_identity ].
   apply comp_respects.
   simpl.
   apply (fmor_preserves_id (ebc_first (garrow_functor c))).
   reflexivity.

 symmetry.
   eapply transitivity.
   eapply transitivity; [ idtac | apply right_identity ].
   apply comp_respects.
   reflexivity.
   simpl.
   apply (fmor_preserves_id (ebc_second (garrow_functor a))).
   symmetry.

 unfold functor_fobj.
   unfold garrow_functor.
   unfold step2_functor.
   Opaque R'.
   Opaque ff_functor_section_functor.
   unfold functor_comp.
   simpl.
   Transparent R'.
   Transparent ff_functor_section_functor.
   idtac.
   apply (me_faithful(MonicEnrichment:=CMon)).
   eapply transitivity; [ eapply full_roundtrip | idtac ].

 unfold fmor at 1.
```

```
unfold R'.
unfold me_homfunctor.
eapply transitivity.
  eapply comp_respects; [ idtac | apply reflexivity ].
  eapply comp_respects; [ apply reflexivity | idtac ].
  eapply symmetry.
  apply assoc_lemma1.

apply symmetry.
  eapply transitivity.
  eapply symmetry.
  apply assoc_lemma2.

simpl.
  eapply transitivity.
  eapply comp_respects; [ apply reflexivity | idtac ].
  eapply symmetry.
  eapply (mf_consistent(PreMonoidalFunctor:=CM)).
  apply symmetry.
  eapply transitivity.
  eapply symmetry.
  apply associativity.
  apply comp_respects; [ idtac | apply reflexivity ].

eapply transitivity.
  apply associativity.
  eapply transitivity.
  apply associativity.
  symmetry.
  eapply transitivity.
  apply associativity.
  apply comp_respects; [ reflexivity | idtac ].

apply symmetry.
  eapply transitivity.
  apply comp_respects; [ reflexivity | idtac ].
  eapply transitivity.
  eapply symmetry.
  apply associativity.
  eapply transitivity.
  apply comp_respects; [ idtac | reflexivity ].
```

```
eapply transitivity.
apply associativity.
eapply transitivity.
eapply transitivity.
apply comp_respects; [ reflexivity | idtac ].
apply comp_respects; [ idtac | reflexivity ].
eapply symmetry.
apply (mf_consistent(PreMonoidalFunctor:=reification)).
apply comp_respects; [ reflexivity | idtac ].
apply iso_comp1.
apply right_identity.
eapply transitivity.
eapply symmetry.
apply associativity.
eapply transitivity.
apply comp_respects; [ idtac | reflexivity ].
eapply transitivity.
apply associativity.
apply comp_respects; [ reflexivity | idtac ].
eapply symmetry.
eapply (centralmor_first(CentralMorphism:=commutative_central(CommutativeCat:=enr_v_mon C) _)).
eapply transitivity.
eapply transitivity.
apply comp_respects; [ reflexivity | idtac ].
eapply transitivity.
  eapply symmetry.
  apply (fmor_preserves_comp (bin_second(BinoidalCat:=enr_v_mon C) _)).
apply comp_respects; [ idtac | reflexivity ].
eapply symmetry.
  apply (fmor_preserves_comp (bin_second(BinoidalCat:=enr_v_mon C) _)).
apply comp_respects; [ reflexivity | idtac ].
  apply associativity.
eapply transitivity.
  eapply symmetry.
  apply associativity.
apply comp_respects; [ idtac | reflexivity ].
eapply transitivity.
eapply transitivity.
apply associativity.
apply comp_respects; [ reflexivity | idtac ].
eapply transitivity.
```

```
  apply associativity.
  eapply transitivity; [ idtac | apply right_identity ].
  apply comp_respects; [ reflexivity | idtac ].
  eapply transitivity.
  apply (fmor_preserves_comp (bin_second(BinoidalCat:=enr_v_mon C) _)).
  eapply transitivity; [ idtac |
  apply (fmor_preserves_id (bin_second(BinoidalCat:=enr_v_mon C) _)) ].
  apply (fmor_respects (bin_second(BinoidalCat:=enr_v_mon C) _)).
  apply iso_comp1.
  apply reflexivity.

eapply transitivity.
  eapply symmetry.
  apply associativity.
  eapply transitivity.
  eapply symmetry.
  apply associativity.
  eapply symmetry.
  eapply transitivity.
  eapply symmetry.
  apply associativity.
  apply comp_respects; [ idtac | reflexivity ].

apply symmetry.
  eapply transitivity.
  apply comp_respects; [ idtac | reflexivity ].
  eapply transitivity.
  apply associativity.
  apply comp_respects; [ reflexivity | idtac ].
  eapply transitivity.
  eapply symmetry.
  apply associativity.
  apply comp_respects; [ idtac | reflexivity ].
  eapply transitivity.
  apply comp_respects; [ reflexivity | idtac ].
  apply associativity.
  apply iso_comp1_right.

eapply transitivity.
  apply comp_respects; [ idtac | reflexivity ].
  eapply transitivity.
```

```
  apply comp_respects; [ reflexivity | idtac ].
  apply associativity.
  eapply transitivity.
  eapply symmetry.
  apply associativity.
  apply comp_respects; [ idtac | reflexivity ].
  eapply transitivity.
  apply comp_respects; [ idtac | reflexivity ].
  eapply transitivity.
  apply comp_respects; [ reflexivity | idtac ].
  eapply transitivity.
  eapply symmetry.
  apply (fmor_preserves_comp (bin_first(BinoidalCat:=enr_v_bin C) _)).
  eapply transitivity.
  apply comp_respects; [ reflexivity | idtac ].
  eapply symmetry.
  apply (fmor_preserves_comp (bin_first(BinoidalCat:=enr_v_bin C) _)).
  eapply symmetry.
  apply associativity.
  eapply symmetry.
  apply associativity.
  apply reflexivity.
  eapply transitivity.
  apply comp_respects; [ idtac | reflexivity ].
  apply comp_respects; [ idtac | reflexivity ].
  eapply transitivity.
    apply associativity.
  eapply transitivity; [ idtac | apply right_identity ].
  apply comp_respects; [ reflexivity | idtac ].
  eapply transitivity.
  apply (fmor_preserves_comp (bin_first(BinoidalCat:=enr_v_bin C) _)).
  eapply transitivity; [ idtac | apply (fmor_preserves_id (bin_first(BinoidalCat:=enr_v_mon C) _)) ].
  apply (fmor_respects (bin_first(BinoidalCat:=enr_v_mon C) _)).
  eapply transitivity.
    apply comp_respects; [ idtac | reflexivity ].
    eapply mf_consistent.
    apply iso_comp1.

eapply transitivity.
  apply associativity.
  eapply transitivity.
```

```
  apply comp_respects; [ idtac | reflexivity ].
  eapply transitivity.
  eapply symmetry.
  apply associativity.
  apply comp_respects; [ idtac | reflexivity ].
  eapply symmetry.
  eapply (centralmor_first(CentralMorphism:=commutative_central(CommutativeCat:=enr_v_mon C) _)).
  eapply transitivity.
  apply associativity.
  eapply transitivity.
  apply associativity.
  set (fun a b => isos_forward_equal_then_backward_equal _ _ (mf_consistent(PreMonoidalFunctor:=CM) a b)) as q.
  apply symmetry.
  eapply transitivity.
  apply comp_respects; [ idtac | reflexivity ].
  set (q (garrow_fobj b) (garrow_fobj a)) as q'.
  simpl in q'.
  set (fun qq => fmor_respects (bin_first(BinoidalCat:=enr_v_bin C) qq) _ _ q') as q''.
  eapply symmetry.
  apply q''.
  apply comp_respects; [ reflexivity | idtac ].

apply symmetry.
  eapply transitivity.
  apply comp_respects; [ reflexivity | idtac ].
  apply comp_respects; [ reflexivity | idtac ].
  apply comp_respects; [ idtac | reflexivity ].
  set (ni_commutes (pmon_assoc_rr(PreMonoidalCat:=enr_v_mon C) (reification b) (reification c))
                   #(homset_tensor_iso a)) as xx.
  unfold functor_comp in xx.
  simpl in xx.
  set (pmon_coherent_r(PreMonoidalCat:=enr_v_mon C) (reification a) (reification b) (reification c)) as yy.
    set (isos_forward_equal_then_backward_equal _ _ yy) as yy'.
    eapply transitivity.
    apply comp_respects; [ idtac | reflexivity ].
    apply symmetry in yy'.
    eapply transitivity; [ idtac | apply yy' ].
    eapply symmetry.
    apply iso_inv_inv.
  clear yy' yy.
    apply iso_shift_right' in xx.
```

```
   apply symmetry in xx.
   idtac.
   assert (#((pmon_assoc_rr(PreMonoidalCat:=enr_v_mon C) (reification b) (reification c)) (reification a)) ^-1 >>>
#(homset_tensor_iso a) ⋉ (reification b ⊗ reification c)
  ~~
(#(homset_tensor_iso a) ⋉ reification b) ⋉ reification c >>>
#((pmon_assoc_rr(PreMonoidalCat:=enr_v_mon C) (reification b) (reification c)) _)^-1).
   apply iso_shift_left.
   eapply transitivity.
   apply associativity.
   apply xx.
   apply H.
   clear q.
 eapply transitivity.
   apply comp_respects; [ reflexivity | idtac ].
   eapply transitivity.
   apply comp_respects; [ idtac | reflexivity ].
   eapply symmetry.
   apply (fmor_preserves_comp (bin_first(BinoidalCat:=enr_v_mon C) _)).
   eapply transitivity.
   apply associativity.
   apply comp_respects; [ reflexivity | idtac ].
   eapply transitivity.
   eapply symmetry.
   apply associativity.
   apply comp_respects; [ idtac | reflexivity ].
   eapply transitivity.
   eapply symmetry.
   apply associativity.
   eapply transitivity; [ idtac | apply left_identity ].
   apply comp_respects; [ idtac | reflexivity ].
   eapply transitivity.
     apply (fmor_preserves_comp (bin_first(BinoidalCat:=enr_v_bin C) _)).
   eapply transitivity; [ idtac |
     apply (fmor_preserves_id (bin_first(BinoidalCat:=enr_v_bin C) _)) ].
   apply (fmor_respects (bin_first(BinoidalCat:=enr_v_bin C) _)).
   eapply transitivity.
     apply (fmor_preserves_comp (bin_first(BinoidalCat:=enr_v_bin C) _)).
   eapply transitivity; [ idtac |
     apply (fmor_preserves_id (bin_first(BinoidalCat:=enr_v_bin C) _)) ].
   apply (fmor_respects (bin_first(BinoidalCat:=enr_v_bin C) _)).
```

```
    apply iso_comp2.

set (fun a' => ni_commutes (pmon_assoc(PreMonoidalCat:=enr_v_mon C) a' (reification c))
  (iso_backward (homset_tensor_iso b))) as xx.
  unfold fmor in xx; unfold functor_comp in xx; simpl in xx.
  eapply transitivity.
  apply comp_respects; [ reflexivity | idtac ].
  eapply transitivity.
  eapply symmetry.
  eapply associativity.
  apply comp_respects; [ idtac | reflexivity ].
  eapply transitivity.
  apply comp_respects; [ reflexivity | idtac ].
  eapply transitivity.
    eapply isos_forward_equal_then_backward_equal.
    apply (pmon_coherent_r(PreMonoidalCat:=enr_v_mon C)).
    apply iso_inv_inv.
    eapply symmetry.
    eapply xx.
    clear xx.
  eapply transitivity.
    apply comp_respects; [ reflexivity | idtac ].
    eapply transitivity.
    eapply associativity.
    apply comp_respects; [ reflexivity | idtac ].
    eapply transitivity.
    apply comp_respects; [ reflexivity | idtac ].
    eapply symmetry.
    apply (fmor_preserves_comp (bin_second(BinoidalCat:=enr_v_bin C) _)).
    eapply transitivity.
    eapply symmetry.
    eapply associativity.
    eapply transitivity; [ idtac | apply left_identity ].
    apply comp_respects; [ idtac | reflexivity ].
    eapply transitivity.
      apply (fmor_preserves_comp (bin_second(BinoidalCat:=enr_v_bin C) _)).
      eapply transitivity; [ idtac |
      apply (fmor_preserves_id (bin_second(BinoidalCat:=enr_v_bin C) _)) ].
      apply fmor_respects.
    eapply transitivity.
      apply (fmor_preserves_comp (bin_first(BinoidalCat:=enr_v_bin C) _)).
```

```
        eapply transitivity; [ idtac |
        apply (fmor_preserves_id (bin_first(BinoidalCat:=enr_v_bin C) _)) ].
        apply (fmor_respects (bin_first(BinoidalCat:=enr_v_bin C) _)).
      apply iso_comp2.

  set (fun a b => ni_commutes (pmon_assoc_ll(PreMonoidalCat:=enr_v_mon C) a b)) as xx.
    unfold functor_comp in xx.
    simpl in xx.
    eapply transitivity.
    apply comp_respects; [ reflexivity | idtac ].
    eapply transitivity.
      apply comp_respects; [ idtac | reflexivity ].
      eapply symmetry.
      apply (pmon_coherent_l(PreMonoidalCat:=enr_v_mon C)).
      apply xx.
    eapply transitivity.
      eapply symmetry.
      apply associativity.
    apply symmetry.
      eapply transitivity.
      eapply symmetry.
      apply (pmon_coherent_l(PreMonoidalCat:=enr_v_mon C)).
    apply symmetry.
    eapply transitivity; [ idtac | apply left_identity ].
      apply comp_respects; [ idtac | reflexivity ].
    eapply transitivity.
      apply (fmor_preserves_comp (bin_second(BinoidalCat:=enr_v_bin C) _)).
      eapply transitivity; [ idtac |
      apply (fmor_preserves_id (bin_second(BinoidalCat:=enr_v_bin C) _)) ].
      apply fmor_respects.
      apply iso_comp2.
  Qed.

Lemma cancell_lemma `(F:PreMonoidalFunctor) b :
  iso_backward (mf_i F) ⋉ (F b) >>> #(pmon_cancell (F b)) ~~
  #(mf_first F b) _) >>>  F \ #(pmon_cancell b).
  set (mf_cancell(PreMonoidalFunctor:=F) b) as q.
  setoid_rewrite associativity in q.
  set (@comp_respects) as qq.
  unfold Proper in qq.
  unfold respectful in qq.
```

```
  set (qq _ _ _ _ _ _ (iso_backward (mf_i F) ⋉ F b) (iso_backward (mf_i F) ⋉ F b) (reflexivity _) _ _ q) as q'.
  setoid_rewrite <- associativity in q'.
  clear q qq.
  setoid_rewrite (fmor_preserves_comp (-⋉ F b)) in q'.
  eapply transitivity; [ apply q' | idtac ].
  clear q'.
  setoid_rewrite <- associativity.
  apply comp_respects; try reflexivity.
  symmetry.
  apply iso_shift_left.
  setoid_rewrite iso_comp1.
  symmetry.
  eapply transitivity; [ idtac | eapply (fmor_preserves_id (-⋉ F b))].
  apply (fmor_respects (-⋉ F b)).
  apply iso_comp2.
  Qed.

Lemma cancell_coherent (b:enr_v K) :
 #(pmon_cancell(PreMonoidalCat:=enr_c_pm C) (garrow_functor b)) ~~
 (#(iso_id (enr_c_i C)) ⋉ garrow_functor b >>>
  #((garrow_first b) (enr_v_i K))) >>> garrow_functor \ #(pmon_cancell(PreMonoidalCat:=enr_v_mon K) b).
  Opaque Underlying.
  Opaque fmor.
  intros; simpl.
    setoid_rewrite right_identity.
    symmetry.
    eapply transitivity; [ idtac | apply left_identity ].
    apply comp_respects.
    apply (fmor_preserves_id (ebc_first (garrow_functor b))).
    unfold garrow_functor.
    unfold step2_functor.
    Transparent fmor.
    unfold functor_fobj.
    unfold functor_comp.
    simpl.

    apply (me_faithful(MonicEnrichment:=CMon)).
    eapply transitivity; [ eapply full_roundtrip | idtac ].

    eapply transitivity.
    apply comp_respects; [ idtac | reflexivity ].
```

```
apply comp_respects; [ idtac | reflexivity ].
apply comp_respects; [ reflexivity | idtac ].
apply comp_respects; [ idtac | reflexivity ].
apply comp_respects; [ reflexivity | idtac ].
eapply symmetry.
apply (fmor_preserves_comp (bin_first(BinoidalCat:=enr_v_bin C) (reification b))).

apply symmetry.
apply iso_shift_left.

symmetry.
eapply transitivity.
eapply transitivity.
apply associativity.
apply comp_respects; [ reflexivity | idtac ].
eapply transitivity.
apply associativity.
eapply transitivity.
apply associativity.
apply comp_respects; [ reflexivity | idtac ].
eapply transitivity.
apply associativity.
apply comp_respects; [ reflexivity | idtac ].
eapply symmetry.
set (mf_cancell(PreMonoidalFunctor:=reification) b) as q.
eapply transitivity; [ idtac | apply associativity ].
apply q.

apply iso_shift_left'.
eapply transitivity.
apply associativity.
symmetry.
set (@iso_shift_right') as qq.
simpl in qq.
apply qq.
clear qq.
unfold me_homfunctor.
eapply transitivity.
symmetry.
apply (cancell_lemma CM (garrow_fobj b)).
```

```
    apply symmetry.
    eapply transitivity.
    apply comp_respects; [ idtac | reflexivity ].
    eapply transitivity.
    eapply symmetry.
    eapply associativity.
    apply comp_respects; [ idtac | reflexivity ].
    eapply symmetry.
    eapply (centralmor_first(CentralMorphism:=commutative_central(CommutativeCat:=enr_v_mon C) _)).

    eapply transitivity.
    apply associativity.
    eapply transitivity.
    apply associativity.
    apply comp_respects; try reflexivity.

    unfold functor_fobj.
    unfold pmon_I.

    set (ni_commutes (pmon_cancell(PreMonoidalCat:=enr_v_mon C))) as q.
    eapply transitivity.
    eapply symmetry.
    apply associativity.
    eapply transitivity.
    apply comp_respects; [ idtac | reflexivity ].
    eapply symmetry.
    apply q.
    clear q.
    unfold fmor; simpl.

    eapply transitivity.
    apply associativity.
    eapply transitivity; [ idtac | apply right_identity ].
    apply comp_respects; try reflexivity.
    apply iso_comp2.
    Qed.

Lemma cancelr_lemma `(F:PreMonoidalFunctor) b :
  (F b) ⋊ iso_backward (mf_i F)>>> #(pmon_cancelr (F b)) ~~
  #((mf_first F _) _) >>>  F \ #(pmon_cancelr b).
  set (mf_cancelr(PreMonoidalFunctor:=F) b) as q.
```

```
  setoid_rewrite associativity in q.
  set (@comp_respects) as qq.
  unfold Proper in qq.
  unfold respectful in qq.
  set (qq _ _ _ _ _ _ (F b ⋊ iso_backward (mf_i F)) (F b ⋊ iso_backward (mf_i F)) (reflexivity _) _ _ q) as q'.
  setoid_rewrite <- associativity in q'.
  clear q qq.
  setoid_rewrite (fmor_preserves_comp (F b ⋊ -)) in q'.
  eapply transitivity; [ apply q' | idtac ].
  clear q'.
  setoid_rewrite <- associativity.
  apply comp_respects; try reflexivity.
  symmetry.
  apply iso_shift_left.
  eapply transitivity.
    apply comp_respects; [ idtac | reflexivity ].
    apply mf_consistent.
  setoid_rewrite iso_comp1.
  symmetry.
  eapply transitivity; [ idtac | eapply (fmor_preserves_id (F b ⋊ - ))].
  apply (fmor_respects (F b ⋊ -)).
  apply iso_comp2.
  Qed.

Lemma cancelr_coherent (b:enr_v K) :
 #(pmon_cancelr(PreMonoidalCat:=enr_c_pm C) (garrow_functor b)) ~~
 (garrow_functor b ⋊ #(iso_id (enr_c_i C)) >>>
  #((garrow_second b) (enr_v_i K))) >>> garrow_functor \ #(pmon_cancelr(PreMonoidalCat:=enr_v_mon K) b).
  Opaque Underlying.
  Opaque fmor.
  intros; simpl.
    setoid_rewrite right_identity.
    symmetry.
    eapply transitivity; [ idtac | apply left_identity ].
    apply comp_respects.
    apply (fmor_preserves_id (ebc_second (garrow_functor b))).
    unfold garrow_functor.
    unfold step2_functor.
    Transparent fmor.
    unfold functor_fobj.
    unfold functor_comp.
```

```
simpl.

apply (me_faithful(MonicEnrichment:=CMon)).
eapply transitivity; [ eapply full_roundtrip | idtac ].

eapply transitivity.
apply comp_respects; [ idtac | reflexivity ].
apply comp_respects; [ idtac | reflexivity ].
apply comp_respects; [ reflexivity | idtac ].
apply comp_respects; [ idtac | reflexivity ].
apply comp_respects; [ idtac | reflexivity ].
eapply symmetry.
apply (fmor_preserves_comp (bin_second(BinoidalCat:=enr_v_bin C) _)).

apply symmetry.
apply iso_shift_left.

symmetry.
eapply transitivity.
eapply transitivity.
apply associativity.
apply comp_respects; [ reflexivity | idtac ].
eapply transitivity.
apply associativity.

set (mf_cancelr(PreMonoidalFunctor:=reification) b) as q.
setoid_rewrite associativity in q.

eapply transitivity.
apply associativity.
eapply transitivity.
apply associativity.
apply comp_respects; [ reflexivity | idtac ].
eapply transitivity.
eapply symmetry.
apply associativity.
eapply transitivity.
apply comp_respects; [ idtac | reflexivity ].
symmetry.
eapply (centralmor_first(CentralMorphism:=commutative_central(CommutativeCat:=enr_v_mon C) _)).
eapply transitivity.
```

```
apply associativity.
apply comp_respects; [ reflexivity | idtac ].
eapply transitivity.
apply comp_respects; [ reflexivity | idtac ].
apply comp_respects; [ idtac | reflexivity ].
apply mf_consistent.
eapply symmetry.
apply q.

apply iso_shift_left'.
eapply transitivity.
apply associativity.
symmetry.
set (@iso_shift_right') as qq.
simpl in qq.
apply qq.
clear qq.
unfold me_homfunctor.
eapply transitivity.
symmetry.
apply (cancelr_lemma CM (garrow_fobj b)).

unfold functor_fobj.
unfold pmon_I.

set (ni_commutes (pmon_cancelr(PreMonoidalCat:=enr_v_mon C))) as q.
apply symmetry.
eapply transitivity.
apply comp_respects; [ idtac | reflexivity ].
apply comp_respects; [ reflexivity | idtac ].
eapply symmetry.
apply q.
clear q.

eapply transitivity.
apply associativity.
apply comp_respects; try reflexivity.
simpl.

eapply transitivity.
apply associativity.
```

```
      eapply transitivity; [ idtac | apply right_identity ].
      apply comp_respects; try reflexivity.
      apply iso_comp2.
      Qed.

  Instance garrow_monoidal : PreMonoidalFunctor (enr_v_pmon K) (enr_c_pm C) garrow_functor :=
  { mf_first      := garrow_first
  ; mf_second     := garrow_second
  ; mf_i          := iso_id _ }.
    intros; reflexivity.
    intros; apply (reification_host_lang_pure _ _ _ reification).
    apply cancell_coherent.
    apply cancelr_coherent.
    apply assoc_coherent.
    Defined.

  Definition garrow_from_reification : GeneralizedArrow K C :=
    {| ga_functor_monoidal := garrow_monoidal
     ; ga_host_lang_pure   := reification_host_lang_pure _ _ _ reification
    |}.

End GArrowFromReification.
```