# Works, Part 1. Generic works

## Scope of document & Introduction

This document discusses the issues in using bf:Work in the context of multiple recordings of multiple performances of the same generic musical work, the issues in the current modeling structure, and a suggested two-tier work modeling that resolves some of these issues and moves BIBFRAME into closer compatibility with materials modeled in FRBR/IFLA-LRM/RDA. It also clarifies the relationship between the BIBFRAME work and the BIBFRAME event (particularly performances, concerts, etc.) in the context of performed music.

## Modelling Works in BIBFRAME

BIBFRAME organizes resource descriptions into 3 core levels of abstraction: Work, Instance, and Item.[[1]](#footnote-1) A bf:Work is defined as “The highest level of abstraction, a Work, in the BIBFRAME context, reflects the conceptual essence of the cataloged resource:  authors, languages, and what it is about (subjects)” while the instance is defined in relation to the work: “A Work may have one or more individual, material embodiments, for example, a particular published form. These are Instances of the Work.  An Instance reflects information such as its publisher, place and date of publication, and format.”

A bf:Work has multiple subclasses: bf:Audio, bf:Cartography, bf:Dataset, bf:MixedMaterial, bf:Multimedia, bf:NotatedMusic, bf:NotatedMovement, bf:Object, bf:StillImage, and bf:Text, which cluster works in how they are intended to be perceived. These subclasses are basically equivalent to the FRBR/LRM expression in that they combine a conceptual-type work with a closer clustering (“constellation” in LRM) of attributes of a particular representation of a work.

### bf:Work subclasses vs bf:Content

In RDA and its abstract models the divisions of bf:Work represented by its subclasses are instead modelled as attribute of the work. Theses attributes can be recorded in BIBFRAME with the property/class bf:content/bf:Content, using vocabulary from RDA.

If one uses RDA vocabulary in bf:Content, there is considerable overlap of there terms with the subclasses of bf:Work (e.g. bf:Text, rdaco:text), which may have some potential for confusion to users and to machine inferencing or search strategies. Given that potential, it is worth considering whether to *not* use one or the other in implementing BIBFRAME, i.e., not using the subclasses of bf:Work, especially in a relatively pure implementation of RDA, or not using bf:Content in a non-RDA context.

On the other hand, in the particular case of performed music, there are benefits in retaining both the bf:Work subclass and the bf:Content vocabularies. Unlike other subclasses of bf:Work, the relevant subclass, bf:Audio, is not at the same level of granularity as that of the RDA vocabularies, which include rdaco:performed music, rdaco:spoken word, and rdaco:sounds. There are major benefits to having both the broader and narrower terms available for searching, and we would suggest that both bf:Content and the subclass bf:Audio be used, when cataloging with RDA. These Works papers will use the subclasses of bf:Work (and an understanding of bf:Content with a value of “performed music” or an equivalent), but with the understanding that in some implementations, only bf:Work will be used.

## Relating BIBFRAME Works to BIBFRAME events

The BIBFRAME model for event content of an event, e.g., a music performance is at its most basic as follows:



 A bf:Event is bf:eventContent of a bf:Audio (and inversely, the bf:Audio has bf:eventContentOf a bf:Event

:e1 a bf:Event ;

 bf:eventContent :a1 .

:a1 a bf:Audio ;

 bf:eventContentOf :e1 .

The bf:Event is bf:eventContentOf a bf:Audio (inversely, the bf:Audio has bf:eventContent a bf:Event).

If the bf:Event is a performance of a pre-existing work, then the bf:Audio is created through the combination of a bf:Event and a bf:Work (generally, the exact instance is not known). This relationship is not modeled in BIBFRAME. There is also a relationship between the pre-existing bf:Work and the bf:Audio. In FRBR terms, this would be an expression of an expression, and is covered by bf:hasExpression/bf:expressionOf.



:e1 a bf:Event ;

 bf:eventContent :a1 ;

 xxx:[undefinedProperty] :w1 .

:a1 a bf:Audio ;

 bf:eventContentOf :e1 ;

 bf:expressionOf :w1 .

:w1 a bf:Work ;

 xxx[undefinedProperty] :e1 ;

 bf:hasExpression :a1 .

The properties bf:expressionOf/hasExpression, however, state that they are “[f]or use to connect Works under FRBR/RDA rules. Because of this stated restriction, the property cannot really be used as a generalized link between a performance event and the work that is being performed. Unless LC removes this restriction, another property needs to be created; PMO has chosen pmo:realizedIn/realizationOf.

To make the model more expressive, PMO subclasses bf:Event, in this case the relevant subclass being pmo:Performance. The relationship between the bf:Performance and bf:Audio is expressed through the inverse properties pmo:hasRecording/pmo:recordingOf (subclasses of bf:eventContentOf/ bf:eventContent). The relationship between the bf:Performance and bf:Work is expressed through the inverse property pmo:performanceOf/pmo:hasPerformance.



:p1 a pmo:Performance ;

 pmo:performanceOf :w1 ;

 pmo:hasRecording :a1 .

:w1 a bf:Work ;

 pmo:hasPerformance :p1 ;

 pmo:realizedIn :a1 .

:a1 a bf:Audio ;

 pmo:recordingOf :p1 ;

 pmo:realizationOf :w1 .

If there is no pre-existing work before the performance (such as a fully improvised performance), the bf:Work is excluded.

A further comment is necessary here about the use of the generic bf:Work for the pre-existing work that is performed. In many cases, a performance of a pre-existing work will make use of a particular instance of that work (a published score, for example) in the performance. It is rare, however, that the cataloger will have this information, since most recordings do not record what score was used in performance or even whether any score at all was used; the cataloger or listener only knows that it is a performance of a particular “work”. Thus the emphasis is on “work” rather than “instance of the work” or a particular “expression of a work”. Because of this, the PMO model generally uses the generic bf:Work to represent the pre-existing work in the performance/pre-existing work/recording relationship.

### Relating BIBFRAME works to each other

A major aspect in modeling and cataloging performed music, and music in general, is the proliferation of LRM or FRBR expressions for FRBR/LRM works, which are then modeled as bf:Works or their subclasses, in this case bf:Audio or bf:NotatedMusic. Most books have only one expression, so the work-expression relationship is usually one-to-one. Beethoven’s 5th symphony, however, has had innumerable performances, and multiple notated music editions, each of which is a separate expression or BIBFRAME work.



:a1 a bf:Audio ;

 bf:hasInstance :i1 ;

 bf:hasInstance :i2 ;

 bf:hasInstance :i3 .

:a2 a bf:Audio ;

 bf:hasInstance :i3 ;

 bf:hasInstance :i4 ;

 bf:hasInstance :i5 .

Take the case above. Let us say that these are two recordings (each modelled as a separate bf:Audio) of the same musical work, Rimsky-Korsakov’s Firebird, each with 3 instances. The bf:Audio works are clearly related, each being the recording of a performance of the same work, but there is no way in BIBFRAME to clearly represent this. One can use the generic bf:relatedTo connect the two works as below, but that is neither informative or perhaps strictly true. They are not really related directly to one another, but through the work being performed.



:a1 a bf:Audio ;

 bf:hasInstance :i1 ;

 bf:hasInstance :i2 ;

 bf:hasInstance :i3 ;

 bf:relatedTo :a2 .

:a2 a bf:Audio ;

 bf:hasInstance :i3 ;

 bf:hasInstance :i4 ;

 bf:hasInstance :i5 ;

 bf:relatedTo :a1 .

PMO members decided there was a better way to express this relationship—by use of a generic bf:Work that links to all its “expressions” (bf:Audio, bf:NotatedMusic, bf:Video), for lack of a better word, relating all of them to the musical work that they express. This bf:Work may be a FRBR or LRM work, or some other type of generic work as defined by implementation.



:w1 a bf:Work ;

 bf:realizedIn :a1 ;

 bf:realizedIn :a2 .

:a1 a bf:Audio ;

 bf:realizationOf :w1 ;

 bf:hasInstance :i1 ;

 bf:hasInstance :i2 ;

 bf:hasInstance :i3 .

:a2 a bf:Audio ;

 bf:realizationOf :w1 ;

 bf:hasInstance :i3 ;

 bf:hasInstance :i4 ;

 bf:hasInstance :i5 .

This bf:Work is the same as that which appeared in the tripartite performance/work/recording model extended to include multiple expressions. The entire model, therefore, looks as below:



:w1 a bf:Work ;

 pmo:hasPerformance :p1 ;

 bf:realizedIn :a1 ;

 bf:realizedIn :a2 .

:p1 a pmo:Performance ;

 pmo:performanceOf :w1 ;

 pmo:hasRecording :a1 .

:p2 a pmo:Performance ;

 pmo:performanceOf :w1 ;

 pmo:hasRecording :a2 .

:a1 a bf:Audio ;

 pmo:recordingOf :p1 ;

 pmo:realizationOf :w1 ;

 bf:hasInstance :i1 ;

 bf:hasInstance :i2 ;

 bf:hasInstance :i3 .

:a2 a bf:Audio ;

 pmo:recordingOf :p2 ;

 pmo:realizationOf :w1 ;

 bf:hasInstance :i4 ;

 bf:hasInstance :i5 ;

 bf:hasInstance :i6 .

It is important to note that this extension of the bf:Work to two levels is not mandatory in PMO, and the ontology itself does not dictate what attributes attach to which level. For this, implementers should rely on outside models such as FRBR or LRM, or a consistent internal model.

1. Overview of the BIBFRAME 2.0 model, published April 21, 2016. <http://www.loc.gov/bibframe/docs/bibframe2-model.html> [↑](#footnote-ref-1)