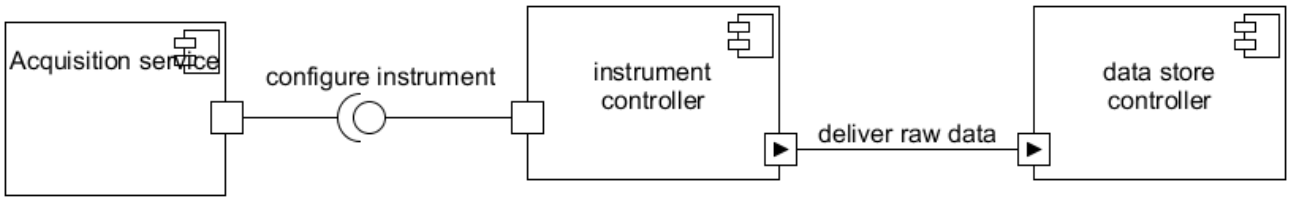
How to use the Model (Computational Viewpoint)

The computational viewpoint of the Model identifies a standard set of components and interfaces from which can be derived a standard set of interactions that a research infrastructure design should address. The Model does *not* specify how those interactions should be implemented – indeed, over the course of the lifetime of a research infrastructure, implementations may change. Nevertheless, the set of the most important interactions should remain constant regardless of implementation changes.

Someone trying to apply the Computational Viewpoint of the Model to their existing or planned research infrastructure should conduct two primary activities: mapping agents and services to computational objects, and defining the interactions that should occur when two or more interfaces are bound together.

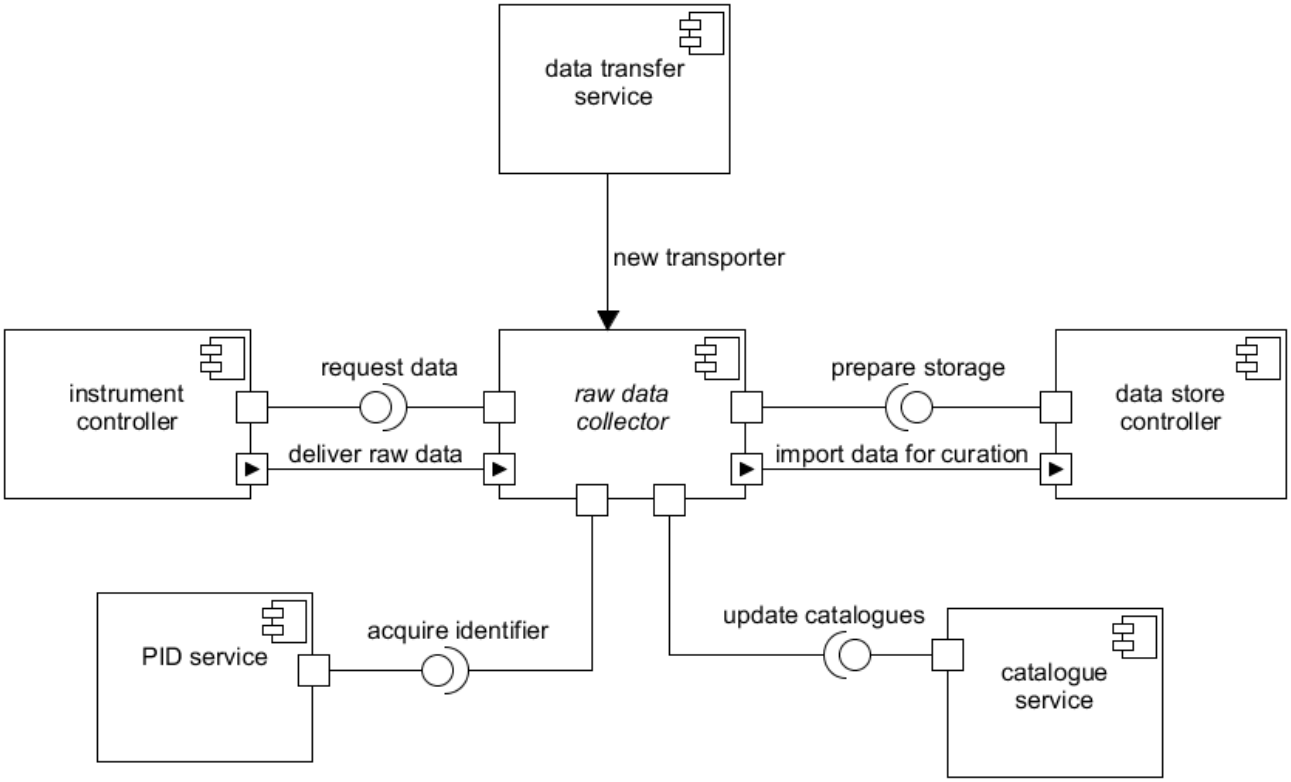
For each computational object in the Model, there should be at least one component or service (or group thereof) provided by the infrastructure that can provide the functions described – depending on the architecture of the infrastructure, there may be multiple candidate, particularly for federated infrastructures. Every such candidate could provide an instantiation of the given object. If no candidates exist, then either (a) the infrastructure does not provide the service embodied by the computational object (and it should be clearly understood that this is indeed the case) or (b) the infrastructure is missing functionality that should be implemented to bring it in compliance with the Model.

For each compatible pair of interfaces (operational or stream), there exists an interaction that should occur given a binding between those two interfaces. The Model does *not* prescribe these interactions, instead simply providing the means to identify them. A compliant research infrastructure should in principle have a well-defined description for *every* possible binding between interfaces on objects that it provides an implementation for.



In the above diagram, an (operational) primitive binding has been established between the *configure instrument* interfaces of an *acquisition service* object and an *instrument controller* object, as well as a (stream) primitive binding between the *deliver raw data* interfaces of the *acquisition service* and a *data store controller* (see How to read the Model (Computational Viewpoint) to understand the above notation and terms). Thus, assuming a Model-compliant research infrastructure that provides at least one acquisition service and instrument controller, there should be a specification of what happens when a 'configure instrument' binding occurs between an acquisition service and instrument controller. Likewise, there should be a specification of how raw data is delivered from an instrument (represented by the instrument controller) to a data store (represented by its own controller).

Many *primitive* (two-interface) bindings are linked in that the establishment of one binding will necessarily lead to the establishment of other bindings, implying a unified interaction description. This is particularly true for *compound* bindings where a particular binding object is created to establish pairwise primitive bindings with multiple computational objects that must all contribute to the given interaction. A compliant research infrastructure must therefore identify all such compound bindings and should define how any binding objects created to coordinate interactions are instantiated (generally as either an oversight service or as 'abstractly' as a distributed process involving agents / services participating in the resulting interaction.



In the above diagram, there exist multiple primitive bindings to a central binding object (the *raw data collector*) that nonetheless all relate to a single compound interaction (describing how the transfer of data from an instrument to a data store is configured and managed). It is very important to properly describe the relationship between the individual bindings and how the compound interaction between the various computational objects involved is produced if constructions like in the diagram above are to be properly understood. In the reference material for the Model, a number of 'core' reference interactions have been described informally to provide a Computational Viewpoint for Model implementors.

Interaction specifications (whether for primitive or compound interface bindings) can take any form deemed suitable by the developers of the infrastructure – for example, UML diagrams such as activity or sequence diagrams may be appropriate, as might be a formal logic model or BPEL workflow, or even natural language if the interaction is simple enough.