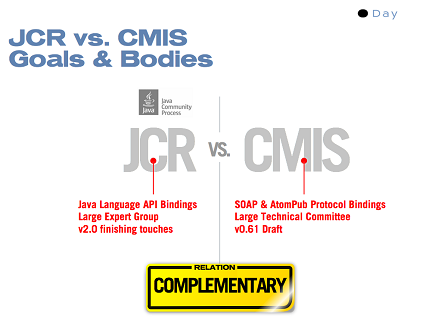
On the [CQ5 tour](http://www.day.com/content/day/en/registration/seminars.html) stop in [Milan](http://www.day.com/content/day/en/registration/seminars/cq5ontour-milan.html) I had the pleasure to talk about CMIS and JCR for the first time since I am serving as the official JCR / CMIS Liaison.

This brought the opportunity to compare and differentiate the two efforts for with a unique legitimacy. There seems to be a desire to discuss the relationship between CMIS and JCR similar to the desire to discuss JCR and WebDAV or, more recently, [JCR and Atom](http://dev.day.com/microsling/content/blogs/main/jcr-loves-atom.html). So far, JCR and CMIS have been compared in what I would call "a less educated fashion" for example in the CMIS v0.5 spec draft. Most of those comparisons have been corrected in the meantime.

Feel free to view the [entire slideshow here](http://www.slideshare.net/uncled/content-management-standards), but here are the main points:

**API vs. Protocol**

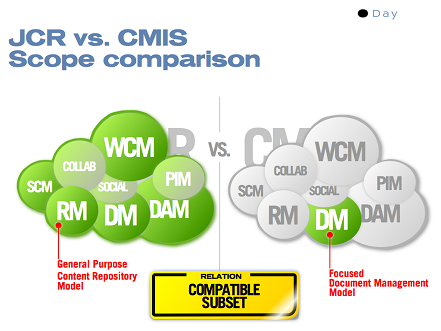
JCR specifies an API (Application Programming Interface) while CMIS specifies two protocol bindings. Much like the Servlet API in Java and the HTTP protocol are complementary this is also the case for JCR and CMIS. Similar arguments have been made for [Atom and WebDAV](http://dev.day.com/microsling/content/blogs/main/jcr-loves-atom.html). JCR and CMIS are complimetary in this aspect.

[](http://dev.day.com/content/ddc/blog/2009/05/jcrcmiscomparison/_jcr_content/images/jcrcmiscomparison/cmis-jcr.034.png)

**Focused Model vs. Generally Applicable Model**

JCR specifies a very general model based Node and Properties that lends itself to the implementation of specific domain models. On the hand, CMIS specifies such a specific domain model for document management.

The CMIS domain model can easily be implemented in a JCR model (see next point). Some people could say that if one were to implement CMIS from scratch a JCR repository would be the ideal starting point as it provides the perfect infrastructure to do that.

[](http://dev.day.com/content/ddc/blog/2009/05/jcrcmiscomparison/_jcr_content/images/jcrcmiscomparison/cmis-jcr.035.png)

I think it is one of the most important assets of CMIS that it exposes the domain model of document management. It will be helpful in further standard discussions to have an established consensus on the domain model amongst the document management vendors.

**Every JCR repository is a CMIS repository**

Based on this type of compatibility between CMIS and JCR [Apache Chemistry](http://incubator.apache.org/chemistry/)(currently in incubation) implements CMIS on top of JCR (amongst other things). This turns every JCR compliant repository into a CMIS compliant repository without any development effort. Even better news are that Chemistry gets bootstrapped with [numerous existing JCR repositories and connectors](http://dev.day.com/microsling/content/blogs/main/jcr-consolidation.html). These repositories can thus expose CMIS right from the start.

[](http://dev.day.com/content/ddc/blog/2009/05/jcrcmiscomparison/_jcr_content/images/jcrcmiscomparison/cmis-jcr.036.png)

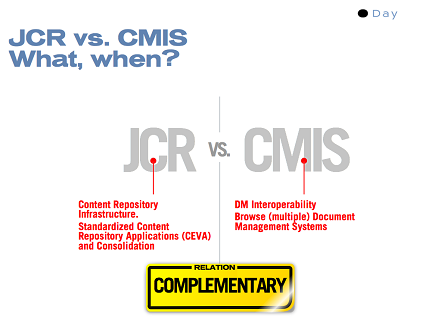
**Interop vs. Infrastructure**

Something that I learned during the years of specifying JCR: There is always a tension in a specification to address both "interop" and "infrastructure" needs. Interop enables different repositories to be compatible on some level, whereas infrastructure provides users with a platform to build upon.

There were tendencies in the JCR expert group to support users of the API in a way that they could build [real-life applications](http://dev.day.com/microsling/content/blogs/main/jcrisformore.html). Hence, the infrastructure aspect was always a very important aspect. Because of that, the least common denominator aspects that come with "interop" were only a part of the equation.

For CMIS, offering general purpose infrastructure is a stated non-goal. CMIS is only concerned with "interop".

Consequently, JCR is very successful with the number of users and applications built on top of JCR, while I believe that it is the goal of CMIS to be very successful with number of implementations.

[](http://dev.day.com/content/ddc/blog/2009/05/jcrcmiscomparison/_jcr_content/images/jcrcmiscomparison/cmis-jcr.037.png)

In terms of perception I think CMIS reduces the expectation on JCR to be a least common denominator interop spec. I welcome that because both specification efforts will be able to evolve in a more agile fashion when they focus on "interop" and "infrastructure" needs respectively.

In summary, I am very excited to have a specification both on the protocol and on the API level addressing the needs of a more open standards based content management landscape.