Yale University

- Use case 1 - Fedora managing access conditions
- Use case 2 - Programmers use API for access condition support in external systems, i.e. Hydra
- Use case 3 - Applications use API for updating access conditions stored in Fedora

Use case 1 - Fedora managing access conditions

Title (goal)	**Fedora managing access conditions**
Primary Actor	Librarian/archivist/curator
Scope	
Level	User goal
Story	The producer of Fedora content wants to be able to set access conditions that would allow for the following scenarios: 1. Content can only be accessed by a specific IP address or list of addresses 2. Data streams in the object can have different access conditions. i.e. TIF restricted to a single user, JPG open to the campus, PDF open to the world. 3. Curator can set authorization in an external system which Fedora can access. a. EZProxy b. homegrown authorization systems 4. Curator can adjust access conditions per object or for thousands/millions of objects at a time 5. Curator can set restrictions on the application permitted to open an object. (focus on born digital for this) a. example: Disk image created for a Mac SE, the curator indicates in the access condition the emulation environment required to open the file. b. note 1: I'm not expecting Fedora to enforce the restriction, only to store it. c. note 2: I know emulation information can be stored in PREMIS, but information about the required emulation settings is different from requiring a specific software title. 6. Curators can set multiple access schemas to an object or data stream in an object. This means a curator could say that a set of IP addresses, an active directory group and a special group in our identity management software may access the materials. 7. Curators can set an access restriction flag for an external patron registration system or other complex authorization system such as Aeon (Atlas Systems). In this case, we only need Fedora to know that the restriction exists. We would apply the code that reacts to this requirement in Hydra or some other system which would cause the patron to go off and register/login to some patron tracking system and when they meet the requirements, Hydra/Blacklight would release the objects that meet the requirement. As for implementation, I can offer some examples of what we use now. We specify the file type/size, the authorization type and then any values associated. Some examples: TIF - Active Directory Group - ManuscriptCurators TIF - Aeon JPG 600px - IP - list of IP values or ranges PDF - external authentication JPG 1200px - Yale only JPG 150px - open access TIF - NetID - yale\mf438 (or a list of NetIDs) DSK - Active Directory Group - ManuscriptDirectors DSK - Emulation - AppleWin v1.1.8 Basically our need is for very granular levels of permission to be stored with the object in Fedora. Right now it is stored as XML as a data stream, it would be beneficial to have it stored differently so that we could make mass changes to materials for entire collections. Another note, we would only be storing a single JPG or possibly no JPG and only a JP2 and will derive the JPG on the fly. So the access condition setup may include conditions for resolutions of digital formats not contained in the data streams. The JPG examples above would indicate that a single JPG exists as a data stream and from that stream we will derive smaller images. But the access conditions are different for ranges of sizes. For Yale, we stick to these sizes, 150px or less (thumb), 151-600px (medium), 600+ (full resolution). For TIF images we use Full, Half page and Quarter page. Right now, all other sizes/resolutions are tied directly to the file type stored as a data stream. But being able to reference access for something that is dynamically generated would make this scale to future needs.

Use case 2 - Programmers use API for access condition support in external systems, i.e. Hydra

Title (goal)	**Programmers use API for access condition support in external systems, i.e. Hydra**
Primary Actor	IT/programming
Scope	
Level	User goal
Story	A programmer can use an API in Fedora that will provide access conditions for a requested object. In an ideal implementation I could make a request to Fedora with a PID and the data stream, the API would then return the requirements for access. For example, a URL might look like http://fedora/PID/TIF/access and then returns something sort of JSON or XML that has all the access information for the specific request. If the data stream type is not specified then all access information would be returned. Since we also would like to contain information related to born digital/emulation environments, this output would also list the restrictions for environments for accessing materials.

Use case 3 - Applications use API for updating access conditions stored in Fedora

Title (goal)	**Applications use API for updating access conditions stored in Fedora**
Primary Actor	IT/programming
Scope	
Level	User goal
Story	A programmer use the API to update/add access conditions to a PID, PID range or all PIDs in a namespace. This would essentially be API access to do everything in the Use Case 1 above. Since no curators at Yale will have direct access to Fedora and all interaction will be through a different software front end such as Archivematica or our homegrown solution Ladybird, we will need CRUD controls for Access Conditions that can be handled from the other software products after ingest takes place. Bulk updates are the most important part of this feature. Working with individual objects is time consuming and a wasteful operation. Ideally this is implemented so that a single update to a PID takes about the same amount of time as updating 10,000 PIDs.