The de novo check is already implemented in the latest version. I think I sent to you an explanation on that. It is based on the conditional filtering of two rules depending on the results of one rule. That is configured in the “ruleFilter_semantic.xml” file:

```
<ruleFilter xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://proteo.cnb.csic.es/miape-api/schemas/ruleFilter_v1.4.xsd">
    <ruleConditions>
    <ruleCondition valid="true" id="DenovoSearchType_rule">
        <ruleToSkip id="SpectrumIdentificationItemPeptideEvidenceRefObjectRule"/>
    </ruleCondition>
    <ruleCondition valid="false" id="DenovoSearchType_rule">
        <ruleToSkip id="SpectrumIdentificationItemNullPeptideEvidenceRefObjectRule"/>
        <ruleToSkip id="DenovoSearchType_rule"/> <!-- discard its error messages -->
    </ruleCondition>
    </ruleConditions>
</ruleFilter>
```

One thing that I would like to make you sure about is that the latest mapping files are located at: https://psi-pi.googlecode.com/svn/trunk/validator/trunk/src/main/resources/mzIdentML-mapping_1.2.0.xml so the mapping file located at: https://psi-pi.googlecode.com/svn/trunk/cv/mzIdentML-mapping_1.2.0.xml is not up to date, so the first one is the one you should work with.

By the way, I have just updated the google code repository with one junit test that I wrote that checks the issue found to get the cvParams from a PDL in a 1.2 mzIdentML file (that is what the validator is using a custom built jmzIdentML that is modified following the instructions from Florian to circumvent that problem).

Let’s say, as you said, that we want to check that some CVs must be present (let’s call them CV terms B) just when other certain CV term (let’s call it CV term A) is present in the protocol.

There are two files that we should modify: the mapping file and the ruleFilter file:
In the **mapping file** we should write a rule for checking for the presence of CV term A, let’s say, rule A, and another rule that checks the presence of CV terms B, let’s say, rule B. At this stage, each of the rules (A and B rules) will generate an error message when their corresponding CV terms are not present.

Now we want to say to the validator that the error messages from rule B should only be showed when rule A is valid. Additionally, we don’t want to see any error from rule A, since it is only used for the condition. So then, we want to say to the validator that errors coming from rule A always been discarded. How to do that? Editing the **ruleFilter_semantic.xml file** as follows:

```
<ruleFilter xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://proteo.cnb.csic.es/miape-api/schemas/ruleFilter_v1.4.xsd">
        <ruleConditions>
                <ruleCondition valid="true" id="rule A">
                        <ruleToSkip id="rule A">
                </ruleCondition>
                <ruleCondition valid="false" id="rule A">
                        <ruleToSkip id="rule B"/>
                        <ruleToSkip id="rule A">
                </ruleCondition>
        </ruleConditions>
</ruleFilter>
```

Just some comments about this approach:
Maybe you can notice that in the resulting mapping file both rule A and rule B will be present. This can be confusing for the community, since you can think that both rules must be valid in order to get a semantically valid mzId file, which is not true, since some rules are only applied when some other CV terms are present (rules like rule B in this

example), and other rules are just running as conditions and they never will show any error message (rules like rule A in the example). So that is something we should aware to the people somehow, apart from writing the appropriate comments in the mapping file (See DenovoSearchType_rule comments on https://code.google.com/p/psi-pi/source/browse/trunk/validator/trunk/src/main/resources/mzIdentML-mapping_1.2.0.xml?spec=svn752&r=752).

- The rules involved in the conditionally checking can be either cvMapping or object rules. As soon as they have a unique id, it doesn't matter which rule type they are.
- MIAPE validation is using other different mapping and ruleFiler files (all located at: https://code.google.com/p/psi-pi/source/browse/trunk/validator/trunk/src/main/#main%2Fresources):
    - mzIdentML-mapping_1.2.0.xml, ObjectRules.1.2.0.xml and ruleFilter_semantic.xml for semantic validation, and
    - miape-msi-rules.1.2.0.xml, ObjectRulesMIAPE.1.2.0.xml and ruleFilter_MIAPEMSI.xml for the MIAPE MSI validation.

So any addition to the semantic validation files should also be incorporated to the MIAPE validation files.