Repository Reporting API

This is a rough sketch of the API. For the first few reports, they will all go in one JAR: maven-repository-reports-standard. No integration with the discoverer is necessary at this point.

ArtifactReporter interface

This interface is used by the single artifact processor.

The initial implementation of this will just need to be a mock implementation in `src/test/java`, used to track the failures and successes for checking assertions. Later, implementations will be made to present reports on the web interface, send them via mail, and so on.

```
public interface ArtifactReporter
{
   String ROLE = ArtifactReporter.class.getName();

   void addFailure( Artifact artifact, String reason );

   void addSuccess( Artifact artifact );

   void addWarning( Artifact artifact, String message );
}
```

ArtifactReportProcessor interface

This interface will be called by the main system for each artifact as it is discovered. This is how each of the different types of reports are implemented, so there will be implementations such as: ChecksumArtifactReport MRM-17 (see repoclean's digestor for some related code), and TransitiveClosureArtifactReport MRM-2.

```
public interface ArtifactReportProcessor
{
   String ROLE = ArtifactReportProcessor.class.getName();

   void processArtifact( Model model, Artifact artifact, ArtifactReporter reporter );
}
```

MetadataReportProcessor interface

This interface is called by the main system for each piece of metadata as it is discovered, similarly to above. ChecksumArtifactReport will also implement this (as metadata has checksums), and there will be a MetadataValidationReport MRM-16

```
public interface MetadataReportProcessor
{
   String ROLE = MetadataReportProcessor.class.getName();

   void processMetadata( RepositoryMetadata metadata, ArtifactReporter reporter );
}
```

Repository interface

The transitive and metadata validation reports will need to query the repository for artifacts. There is no implementation of this at present - instead, a mock implementation should be created that acts as a repository of test artifacts.

```
public interface RepositoryQueryLayer
{
   String ROLE = RepositoryQueryLayer.class.getName();

   void containsArtifact( Artifact artifact );
}
```