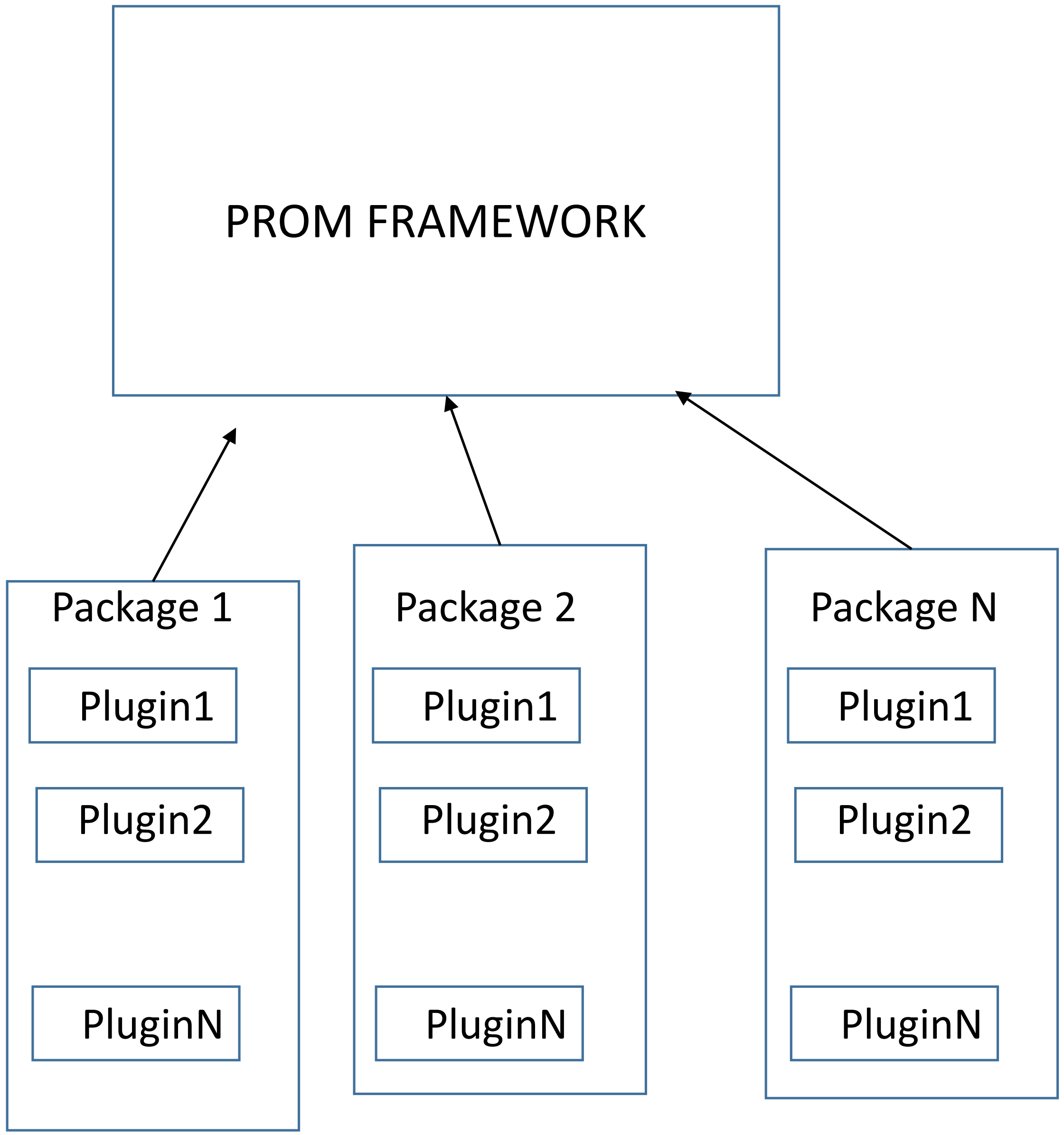
PROM Architecture Overview

Prom important components

- Prom Framework
- Plugin Context
- IVY
- Package Manager

Prom Framework

- Main tasks are
  - Load Plugins
  - Register Plugins
  - Have interfaces for PluginContext
  - Also has implementations for
    - PluginContext, UIPluginContext
  - Includes the Prom annotations
  - *Import*
  - *Export*
  - *Visualization*
  - *Regular Plugins*



Prom Framework - continued

- During boot Prom framework does the following
  - Reads prom.ini file for details like prom version, packages and other internal directory structures like lib, images folder
  - From the package URL obtained get the best suiting repository.
  - Gets instance of Package Manager (Detailed description in next slide)
  - Sets the plugin context in package manager
  - Gets instance of the PluginManager

Package Manager

- Scans for all the packages. Creates a package directory in local .PROM directory and creates all package folders
- URLClassLoaders loads all the plugins from the packages. Scans through every file for plugin annotations and registers accordingly (might not do)
- Checks for dependencies and installs the required ones too

Plugin Context

- Provides interactions for framework with the plugin.
- Based on the annotations defined Plugin context is aware of the inputs and outputs of plugins
- Some examples of extended Plugin Contexts are
  - UIPluginContext
  - CLIPluginContext

- Example :
  - Consider alpha Miner package which contains the implementation of alpha algorithm. The Plugin context is aware of the variants of the different implementations and its corresponding input and output based on the annotations specified.

IVY

- Used for Dependency management in Prom in development

Example – Most commonly all plugins use Xlog object which is present in Log Package.

Hence, we will need to specify this in the ivy.xml

```
<dependency org="prom" name="Log" rev="latest" changing="true" transitive="true" />
```

Mention about conf="lib->default" inclusion to have the jars onto the prom releases

Note – ivy transitively installs the dependencies present on the specified dependencies too. We can turn it off by setting the transitive to FALSE if required especially during development phase

BEST PRACTICES

- Follow the packaging convention as used in
  - https://svn.win.tue.nl/repos/prom/Packages/NewPackageIvy/ (include log package link)
- Always try to use PluginContext implementation. Unless your code requires visualization and needs the UI variant or CLI variant
- One java file will and must have only one Plugin annotation
- It can have multiple PluginVariants annotations based on the arguments it takes
- Have class/es which has algorithm implements. Then have a wrapper class for plugin annotations to be defined.
- Always have the algorithm classes separated from Plugin Definition class. Just have a wrapper class for the algorithm to include the plugin annotations

Slides to be added/modified based on Eric’s comments

- Mention about Build server
- Ivy – importance of conf="lib->default" in the ivy.xml for new dependencies and the resolve in the build.xml performs based on this

PROM creating plugins

Initial setup steps

- In Eclipse, Checkout the trunk folder of NewPackageIvy from the svn repository. This is the template for directory structure
- Code the plugins.
- Ensure you use the right PluginContexts(as explained before).
- Include all the related ivy dependencies in the ivy.xml file

Build and launch

- Once the code is in test phase
- Right click on “ProM with UITopia (NewPackageIvy).launch” and select run as “Prom withUITopia”to launch Prom