Problems

1. The CPI logging module is tightly coupled with UIDocument class, therefore if a document class doesn’t have “First Name” and “Last Name” properties, it won’t be able to get the appropriate “Personal ID” for CPI logging system.
2. It is not designed to accommodate events other than “GetContent” event.
3. The application is not expandable, input and output is rigidly hardcoded, no point to grow.

Goal to achieve with this new design

1. Divide module at least into two parts; input (extractor) and output(exporter) so that it can be expandable by adding different types of extractor or exporter.
2. Extractor and Exporter is defined as Interface, so it is easily expandable by adding a new implementation if necessary.
3. Factory design pattern is used to avoid binding to the concrete instances.

Remaining Issues

1. Almost all the modules depend on the core class, CPILog. If this changes, the whole application will need to be updated. This not only doesn’t sound good, but also very difficult to implement without dropping a module.
2. It is not simple to configure the each extractor and exporter and factory classes in the hierarchy. For example, to implement a FileNet-CSV extractor, at least three configuration file is required; Extractor, AppInfo, and Exporter.   
   Also, if we can use the application for the same type of target applications, for example Special Claims and Trade in UIDMS, it would be cumbersome to configure the sub modules.