Kepler - Bug #3948

Create new configuration system supporting modules

04/06/2009 12:01 PM - Chad Berkley

Status:	Resolved	**Start date:**	04/06/2009
Priority:	Immediate	**Due date:**	
Assignee:	Chad Berkley	**% Done:**	0%
Category:	core	**Estimated time:**	0.00 hour
Target version:	2.0.0	**Spent time:**	0.00 hour
Bugzilla-Id:	3948		

Description

Kepler currently has at least 4 different configuration systems each using their own configuration file. This makes it really difficult to figure out where a specific configuration item is stored. These systems should be merged into, ideally, one system. Realistically, however, two systems will probably need to exist. Currently all UI related configurations are done in .properties files with standard java localization. Workflow and core configuration is done with the ptolemy configuration system. Various other actors and subsystems use their own configuration systems.

Since kepler 2.0 will support modular additions to kepler, the configuration system needs to be flexible enough to allow the modules to add configuration items into the system at runtime. The ptolemy configuration system is flexible enough to do this and since it is used by the kernel, I propose keeping it as the main config system. I also propose keeping the .properties system for UI specific options.

Kepler, possibly through the module manager or possibly through a new configuration manager, should provide an API for allowing modules to merge their configuration options. It should provide error handling for any conflicts.

An extensive search will need to be done on the standard modules to find any other configuration subsystems being used and to merge them into the appropriate new system.

Related issues:

Is duplicate of Kepler - Bug #4088: Create new configuration system	**Resolved**	**05/20/2009**
Blocked by Kepler - Bug #4009: kepler 2.0 must provide everything needed for ...	**Resolved**	**04/21/2009**
Blocked by Kepler - Bug #4336: Merge the old configuration systems	**Resolved**	**08/26/2009**
Blocks Kepler - Bug #4330: common/resources/configurations/config.xml gets mo...	**Resolved**	**08/25/2009**

History

#1 - 04/06/2009 12:27 PM - Matt Jones

See notes on wiki about existing and proposed systems.

https://kepler-project.org/developers/teams/framework/kepler-configuration

#2 - 04/20/2009 05:19 PM - Christopher Brooks

Last week, I spoke with Edward about runtime configuration and he
believes that it would be fairly easy to create an editor to develop
configurations.

For example, if you run
cd $PTII
$PTII/bin/vergil ptolemy/configs/full/configuration.xml

Then you can view the configuration. Using Graph -> Automatic layout
helps clean it up. However, currently, changes are not preserved
by saving. If we provide the user with an environment to create the
configuration, then the question of xml or not xml becomes moot.

We have funding to work on configurations, in particular to provide
an easy way for users to create custom configurations.

A few issues:
- There are plenty of things to configure:
- menus
- the right hand actor pane
- A large problem seems to be in overriding and removing features.

For example, removing a menu choice is difficult
- The Vergil graph viewer should not necessarily expect to be a toplevel window. The issue is: what happens when we look inside or when we plot data? Right now, the position of the plots is encoded in the model. This would be an issue in an Eclipse View, were the plots should be inside an Eclipse view.

About internationalization, we should stick with the standards, see
http://java.sun.com/javase/technologies/core/basic/intl/
and
http://java.sun.com/docs/books/tutorial/i18n/index.html

The i18n tutorial suggests using ResourceBundles.

#3 - 07/13/2009 04:08 PM - Chad Berkley

- Bug 4088 has been marked as a duplicate of this bug. ***

#4 - 08/26/2009 12:50 PM - Matt Jones

This bug now focuses exclusively on creating a new configuration system for managing configuration properties. It does not include changing the portions of kepler that are configurable, but rather only on the creation of a system that allows modules to set system-wide and module-specific properties.

Bug 4336 was created for merging the existing (old) configuration files into the new system.

#5 - 09/11/2009 10:27 AM - Chad Berkley

After looking at Matt's UML diagram (https://code.kepler-project.org/code/kepler-docs/trunk/teams-and-wg/2-infr-teams/framework/design/configuration/configuration-architecture.png), I've been thinking about a couple ideas for serialization. The config system will need to serialize/deserialize to multiple different locations. I think this should use the build system's moduleTree functionality. The workflow would look something like this:

```
deserialize:
for each module in moduleTree {
get file module.config
read module.config into common.config
}

serialize:
for each module.namespace in common.config {
find module.namespace in common.config
write module.namespace to module.config
}
```

The moduleTree functionality is in the build system jar that is accessible from the kepler runtime.

Instead of using a String to represent a module in the ConfigurationProperty, we should probably use a Module object like the build system does. That way, the configuration system has all of the module's metadata that is read in at startup. This could be a useful way for other modules or the core to get information on modules at runtime.

#6 - 10/05/2009 11:55 AM - Chad Berkley

Configuration System Requirements agreed upon on 2009.10.02:

1) A module is able to add configuration properties to the configuration

2) A module is able to read its own configuration properties

3) A module is able to add or override configuration properties added by other modules.

4) A module is able to overwrite another module's configuration properties

5) A module can indicate whether a property is mutable (i.e., if property changes are noticed and incorporated without restarting the application). At runtime, only mutable properties can be overwritten. It is the responsibility of all modules to utilize mutable properties to monitor those properties and respond appropriately to changes.

6) A configuration property can have either or both of a single scalar value or a list of subordinate configuration properties. Subordinate configuration properties must have unique names within their parent property.

7) The configuration system is able to notify modules or other registered listeners that a configuration change has taken place

8) The configuration system is able to serialize each module's configuration properties as text

9) The configuration serialization should be able to store different language versions of each property file (internationalizable)

10) Each module can have its own configuration and can organize its configuration into any number of namespaces, allowing the module to organize its properties into groups. One use case for this is to allow a module to separate its UI strings from other configuration properties.

11) The configuration system is able to tell which module owns a configuration property

12) The configuration system is able to store default configuration properties separately from user-modified configuration properties. Consider whether user preference files, including loading and sharing of such files, is in scope or not.

13) The configuration system should be loaded under all variants of the Kepler application and should have minimal dependencies.

14) The configuration system should be able to store and make accessible documentation about configuration properties (name, description(s), etc.)

15) The configuration system stores strings only. It is up to a module to cast strings to implied value types.
-- desire to include data types as as part of the model (rather than only accept strings)
-- desire to have utility methods for doing casting (e.g., getValueAsInt)
-- change this requirement to: The configuration system will support basic types. If no type is specified, the type will default to String.
-- basic types TBD.

16) The config system should define the set of allowable characters for use in names (e.g., no spaces, or should be java identifiers)

17) The configuration system should specifically consider if and how configuration values or sets of values from the command-line, environment variables, module.txt and other sources should be merged or provided with values from configuration files

#7 - 10/06/2009 12:26 PM - Chad Berkley

Current configuration files in kepler:

Directory: common/configs/ptolemy/configs/kepler
authServicesBundle.properties
caseTableauFactory.xml
ConfigGUIAndCache.xml
ConfigGUINoCache.xml
ConfigNoGUIWithCache.xml
configuration.xml
graphTableauFactory.xml
jobLauncher.properties
repositoryBundle.properties
uiContextMenuMappings_en_US.properties
uiDisplayText_en_US.properties
uiMenuMappings_en_US.properties
uiSettings.properties
uiSVGIconMappingsByClass.properties
uiSVGIconMappingsByLSID.properties

Directory: common/resources/configurations

config.xml

Not sure if some of these are still used. Will have to determine that when converting to the new format (whatever that ends up being). This is just a list so we know what's out there. There are also other configuration files in various modules, but these are the main kepler ones.

#8 - 10/19/2009 01:56 PM - Chad Berkley

I've been working on prototyping and setting up unit tests for the configuration system. There are a few issues I've been running into. I'd like to get a general discussion of these issues and try to come to some conclusions.

The issue we left off with on the last call is what serialization format to use. The conclusion I've come to is that we really do not **need** to come up with one standard configuration format to use. If we think this is very important, we should, but the system I've been working on uses interfaces for serialization and deserialization making it very easy to plug different readers and writers in.

I've been experimenting with YAML and Apache Commons Configuration. Commons is much more flexible. It can read/write 9 different types of files. It supports highly nested structures very well. XML is the easiest format to use with Commons.

While the formatting of the YAML configuration file is very nice in its simplicity, nested structures are somewhat difficult to handle in the API. Since most of our current configuration is highly nested, it's my opinion that commons should be used for porting these configuration files to the new system. YAML could be used for new, or more simply structured files. The use of soft tabs to indicate nesting is somewhat fragile, imho, since accidentally deleting a couple spaces can give you a vastly different structure that what you intended.

Another issue is when a serialization should take place. There are a couple different choices. 1) Serialize the configuration file whenever a change is made. This is the Mac OSX style of doing things. 2) Wait for a calling class to initialize a serialization (basically calling save()). Number 1 makes it easier to keep a serialized config file in sync with the more abstract ConfigurationProperty class. Number 2 is simpler and will allow many properties to change then allow the entire file to be overwritten which is a simpler operation that doing a diff to just update the "dirty" properties.

One more issue that I've come up with is that requirement #6 should be changed. It states that all subproperties should have a unique name. This should not be a requirement since we have many instances of subproperties in our current config files that use the same name multiple times. This is not a requirement of XML or any other nested language that I know of. Anyone object to me changing this requirement?

I'll leave this open to discussion here on bugzilla and on kepler-dev. If we think we need a phone call to figure out these issues (or if anyone has any other issues) let me know and we can arrange that.

#9 - 10/29/2009 01:12 PM - Chad Berkley

The configuration system is now 99% done with only minor tweaks needed. I am still finding and adding utility methods to make accessing properties easier for programmers. I am now moving on to bug 4336. I'm starting with getting rid of the configxml system and replacing it with this. Then I'll move on to the configuration.xml file and the .properties files.

This system conforms to the requirements set out here:
https://kepler-project.org/developers/teams/framework/kepler-configuration/proposed-future-kepler-configuration-system

I'm closing this bug, as I believe this is effectively done. We can reopen new bugs later if issues crop up.

#10 - 03/27/2013 02:25 PM - Redmine Admin

Original Bugzilla ID was 3948