Conversion Tools - Feature #1770

multi-threaded conversion driver

10/30/2012 08:34 AM - Greg Shah

Status:	New	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	40.00 hours
Target version:			
billable:	No	**vendor_id:**	GCD

Description

Related issues:

Blocked by Conversion Tools - Feature #1769: implement full locking/synchroni...	**New**
Blocked by Conversion Tools - Feature #1768: eliminate statics/singleton patt...	**New**

History

#1 - 10/30/2012 08:34 AM - Greg Shah

Implement a fully multi-threaded conversion driver including all phases of conversion (some phases may have to wait until all files are processed before continuing, but within any given phase it should be highly parallel).

#2 - 10/31/2012 01:12 PM - Greg Shah

- Target version set to Code Improvements

#3 - 11/16/2016 12:46 PM - Greg Shah

- Target version deleted (Code Improvements)

#4 - 12/27/2017 12:43 PM - Greg Shah

This is an obvious improvement to make that would cut conversion times to a fraction of the current single-threaded time (when run on a multi-processor system). We have tried some simple approaches to make TRPL work in a multi-threaded environment, but it is much more complicated than one might think.

- There are many data structures that are shared and would need to be synchronized. See #1769. This includes data structures that are being directly used by conversion rules as well as much shared state inside the workers, pattern engine, symbol resolver etc...
- There are many uses of a singleton pattern and other static usage for workers which complicate the threading issue. See #1768.
- Properly handle the OO class dependencies such that recursive parsing is done in the correct order and no deadlocks occur.

If it was simple, it would have been done by now. We do plan to fix this, but we will probably not do it before we shift to a cleaner TRPL 2.0 syntax and implementation. This implementation would naturally be multi-threaded and it will eliminate most of the current limitations of the language. See https://proj.goldencode.com/projects/trpl/issues.