NEW RESOURCES CONSULTING

CASE STUDY
CUSTOMIZATION REDUCTION

TECHNOLOGIES
PEOPLESOFT DEVELOPMENT

INDUSTRY
APPLICABLE TO ALL INDUSTRIES

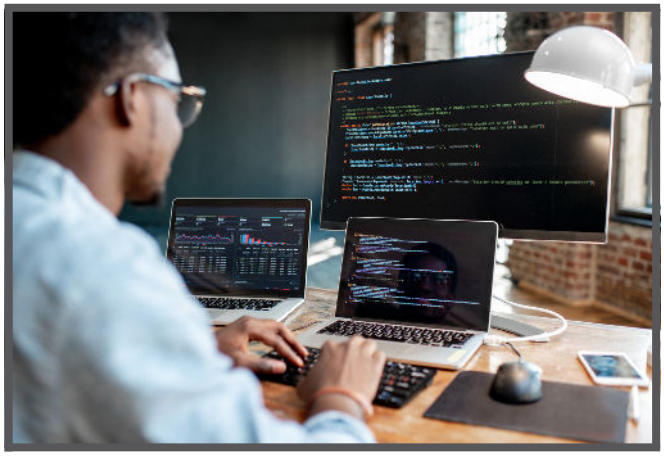
CHALLENGE

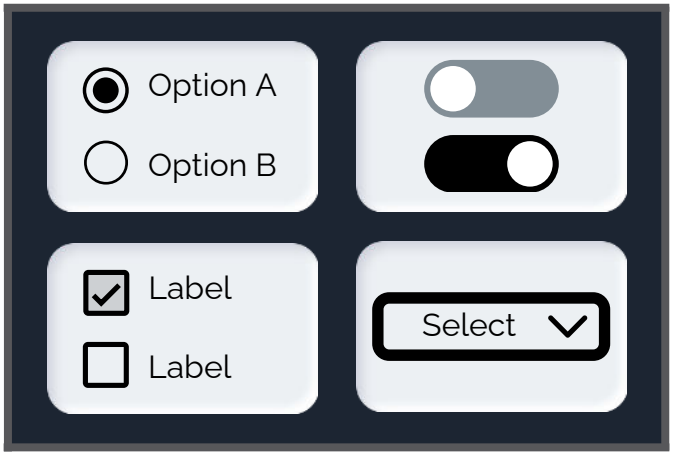
In this case study, the customer desired to reduce their PeopleSoft customizations to reduce the amount of retrofit required when applying PeopleSoft maintenance. Each cycle of PeopleSoft Update Manager, PeopleTools, and individual bug fixes introduced the possibility of overwriting or altering the functionality of all or part of a customization.

SOLUTIONS

Pure removal of a customization and reverting to the delivered code is not often possible unless the business requirement for the customization has changed or is no longer required. Fortunately, Oracle has been providing new features in the latest PeopleTools releases that allow us to "move" these customizations from being embedded in the code to something that sits on top of the delivered code in an additional metadata layer.

The primary methods utilized in this case study were Related Content, Application Engine Injection, and Page and Field composer.

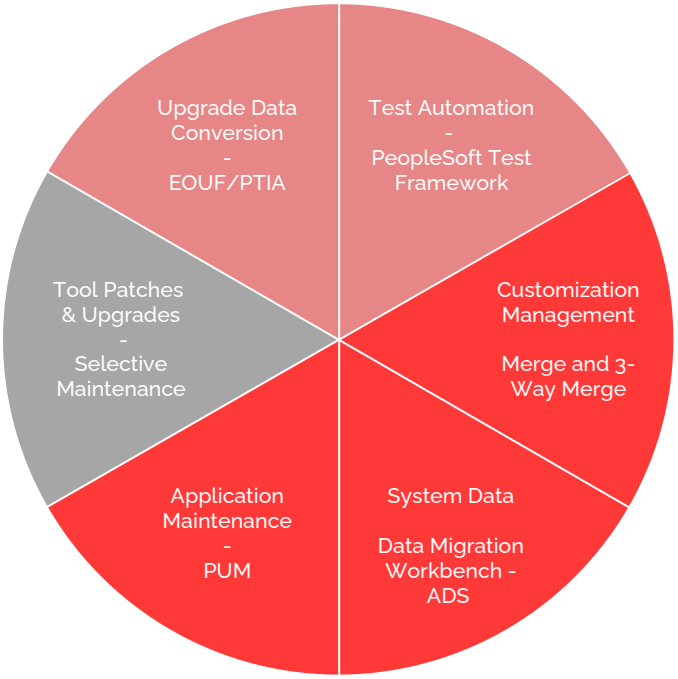




AN OVERWHELMING EFFORT BY ITSELF

When reviewing the compare reports for the system, it was discovered that thousands of customizations had been made over its 15+ years of operation. Taking a targeted effort to move these customizations to configurations as a stand-alone project provided little value to the organization. The application should function as it had done before, but there is limited new functionality, change management is required, and there is a risk of introducing bugs.

It was determined that the new Lifecycle Management tools would be incorporated into future maintenance applications instead of being migrated as a "big bang" of all customizations. It is important to note that each customization must be reviewed to determine if the LCM tools can be used. In some cases, customizing the delivered code is the only way to meet the business requirements.



Lifecycle Management PeopleTools

RESULTS

Over 4 years, with the new methodology of applying maintenance, the total number of actual system customizations has been reduced by approximately 25%. A more telling number is that of those 25% customizations, over half of the "objects" that make up those customizations have appeared in multiple update projects, meaning that not only has the time to apply the single customization been removed, but it also has eliminated the need to apply the same customization over and over repetitively. This is especially useful for customizations for core application components or objects that may see frequent updates. (e.g. Tax Updates)

The benefits of this model include:

- **Reduction of Rework:** The effort put into moving the customization to the configuration is "one-time," as opposed to the effort needing to be redone with each subsequent update.
- **Streamlined Testing:** Because the core application functionality is not being changed, the test cases can move from ensuring it was redone correctly to validating that the delivered code still aligns with the business needs.
- **System Stability:** Since the core application code is not being modified, it is easier to uptake minor bug fixes without the concern of needing to rework customization to the system.

While this approach does not address every single customization in the system, it allows you to approach this in smaller bits – choosing when and how much you wish to address based on the impact introduced during each maintenance cycle.