OSA and Extensibility Q&A

Supporting all the technologies, their different versions, and their different variations is an impossible mission for UFT (QTP), and for cFactory which sits on top of UFT. For typical web applications UFT support is present and cFactory works well since the underlying UFT support for all objects on the application screens is provided. There may be one or two special things that the tester might want to do with an object where we might recommend a keyword to help facilitate that operation.

In other cases, object support is not completely available through UFT. The only way to know which objects are and are not supported by UFT is by doing an Object Support Analysis (OSA). Therefore, sometimes when TurnKey customers want to automate an application for which TurnKey has not built an Accelerator for, TurnKey will suggest that an Object Support Analysis (OSA) is performed.

What is the purpose of the OSA?

An Object Support Analysis is performed to evaluate UFT support for various object classes (WebEdit, WebList, WebTable, etc.) in the application. During the OSA we determine whether UFT has proper support for all of the operations and properties of the object classes presented on any given application screen. This analysis will help us evaluate whether all the objects are identified properly, and that they function as expected. For example, UFT might identify multiple objects of different classes on the same page as WebEdit boxes making it impossible to distinguish between them all. Other objects might be labeled 038184_13i31. This makes selecting the objects when creating components and identifying the objects within the TurnKey DataSheets extremely difficult. In some scenarios we might come across non-native base object classes such as Tab Strip or Menu Bar in a web application for which extensibility might be required.

What is expected of me during the OSA?

It is important to go through a very detailed end-to-end test case during the OSA so that we can see all the screens that might be used in your automated testing to ensure that the objects underneath are all represented. We will need machine setup with UFT and access to application. Also having the right application Subject Matter Experts on hand will go a long way towards ensuring that all test flows are properly identified.

How long does an OSA take?

An OSA can take anywhere between 1 and 2 hours.

What is Extensibility?

Extensibility is the process of extending UFT support for some or all of the objects in an application; HP admits that not every application can be supported, which is why it created the Extensibility Accelerator which is a Visual-BASIC-like IDE to facilitate the creation of an add-in to UFT that provides this support

What if my application requires Extensibility?

Because of the need to support objects for all applications, HP created an Extensibility Accelerator for UFT/QTP to allow for the creation of extensibility on the specific technology/control. Each of these might be utilized in the development of the extensibility add-in for your application:

1. Extensibility support for the .Net family (including WPF, Silverlight), JAVA, and Web, which allow for the addition of recognition methods for controls based on these technologies.
2. Test Extensibility API (TEA) enables the development of an add-in for a technology which isn't supported by QTP.
3. Extensibility Accelerator provides a Visual Studio-like IDE that accelerates and facilitates the design, development, and deployment of QuickTest Web Add in Extensibility support sets. These support sets extend the Web Add in so that one test Web controls that are not supported out of the box.

Who develops the extensibility?

TurnKey's Customer Success Team can develop the extensibility for a fee. A Statement of Work (SOW) will be developed for your application and the amount of work that will be required is estimated. Remember that you only pay for the actual amount of work done since TurnKey's CST bills on a time & material basis only.

What is involved in developing extensibility, or an application add-in, so that cFactory will work for me?

Extensibility, in the form of a custom UFT add-in, includes:

- Custom extensibility properties to assign logical names to both custom and native object classes as well as identification properties for all tab items
- Custom extensibility operations which can be used to select, select, click and manipulate tables within the application
- Custom UFT operations to handle Global Search, Table Search, and Table Entry

Rules Base so that the custom-add (extensibility) can be used within the cFactory framework.

Common Components may also be delivered for high-level navigation and other basic operations within the application.

Is Extensibility required if I'm just UFT-Scripting?

In short, yes. Extensibility is the best way to ensure that object recognition is achieved whether you are using cFactory or UFT-scripting. There are some rare instances where a UFT script doesn't need proper object identification properties – for example, logical object names and column names. UFT scripters typically capture only the objects which are required for their specific test scenario and can rename them manually to make sense within the script. Descriptive programming within the script can handle incorrect object parameterization and/or object operation. In that sense, scripting allows the implementer to fill in missing gaps without the need of extensibility. However, this scripting is extremely difficult to maintain since it basically needs to be rewritten when objects change.

How is cFactory different?

cFactory provides a scriptless framework that allows non-scripters the ability to create reusable components that cover the entire screen. In order to achieve that reusability and to facilitate the data-driven architecture, all objects on the screen must be identifiable and operate properly so that they can be referenced and used. Additionally, cFactory components are very easy to update, and those updates automatically carry through to the data management solution. In order to do this, up-front extensibility is required to develop the missing object support (custom Extensibility, custom rules base, and libraries and/or keywords). UFT scripters may also create extensibility before scripting, or program the missing object support as they create the scripts individually.