Rapport Calidos

UID’s

To be able to use a control in a test, the control has to be added to a UIMap. This is a document which contains a hiërarchy of all controls of a page. When a control is added to this map, you can generate it in code by writing the hiërarchical path from the top parent down to the control you need.
Example:
XamlButton ValidateOverlayButton = UIMapWorkbookConfig.UICalidosMaatWindowsWindow.UICommandBarMainActionCustom.UIValiderenButton;

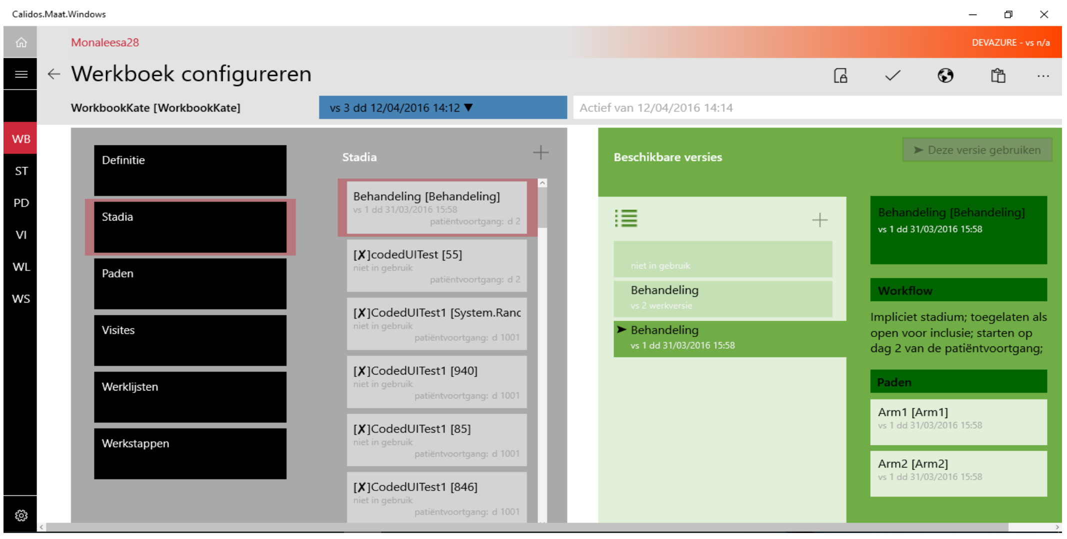
Because some controls do not contain a UID (unique identifier) to search on, the UIMapping sometimes goes wrong. For example, when there are several identical controls displayed, all on the same hiërarchical level, but they do not contain a UID, they are mapped by instance. The elements inside these controls will all be mapped to the first control, even if they exist inside the second control. There are other possible scenarios in which the UIMapping goes wrong, but this one is the most common. And all scenarios have something to do with UID’s.

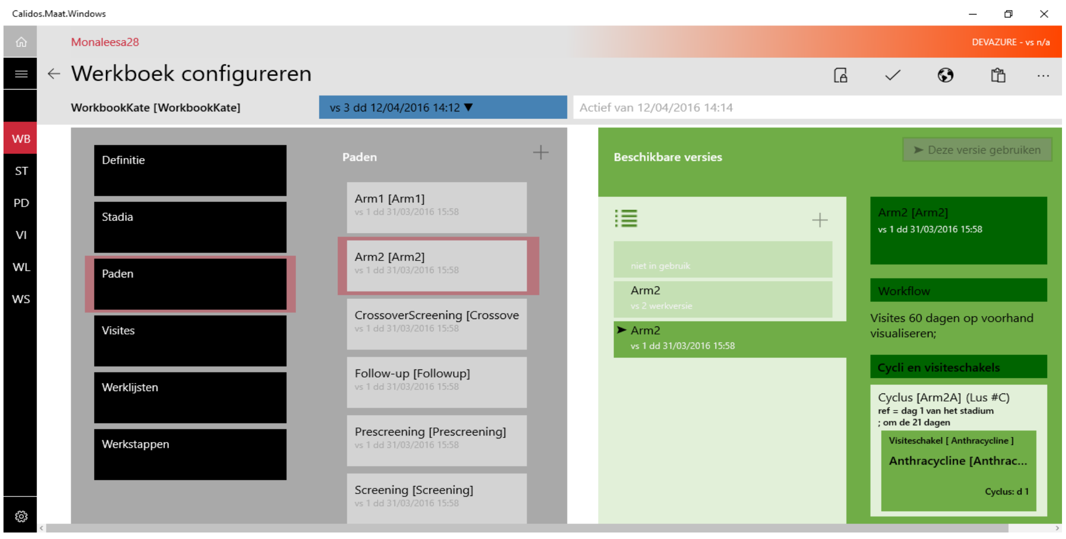
Because of this bad UIMapping, the tests often require impractical workarounds to test some controls. A solution would be to make sure every single element is given a unique UID. When this is correctly done, the hiërarchical structure should be correctly mapped.

Xaml paradigms

We found that sometimes, elements that look the same and act the same visually, still have a different hiërarchical structure or different algorithms. To make the application easier to test, it would be useful to create some sort of guideline for every paradigm, explaining a set of coding rules to which the code should conform for that paradigm.

example:





Look at the controls on the far right, starting from the first control containing the main info of the version, with below it the workflow and below it a list of some other elements. Although this set of controls look the same on 2 different tabs, and have the same functionality, they are differently mapped. On the top screen (stages), every control I just described (interpreting the list on the bottom as one control) exists separately as a child-control of the complete hubsection. However on the bottom screen (paths), the complete set of controls just described, exists inside a list, which is a child of the hubsection.

Another good example is the semantic-zoom functionality. On some pages, the hiërarchical structure of the elements inside the semanticzoom is different from other screens, even though the paradigm stays the same.

Random failing tests

We noticed that even when tests are written completely correct, there are still some tests failing when executing them all together (in a sequence). One possible explanation for this problem could be that the test is trying to find a control before the page has been refreshed (due to appStartUp), and has already found the parent-control but then, the page refreshes and the originally found parent control has therefore disappeared.
After testing, we came to the conclusion that putting in a delay before searching the first control has a positive effect on the amount of tests that fail. However we cannot guarantee a 100% that all tests will pass.
Note that this delay-time is defined in our BasClassCodedUI, so when you want to lengthen or shorten the delay-time, you can do this by changing it only once.

Future work (What was planned)

TODO:

TestResultParser

- Revise ‘Failed Tests’ div in HTML file
  - Only shows tests that have ran and got an outcome but have no cell to map their data towards
- 'Done' labels don’t add up as they should
  - Needs a fix at MatrixCell’s CalculatedValue
- Adding floating headers to HTML file
- ‘info’ tag in CategoryDefs and ObjectDefs is redundant
  - Check out ProcessCategories and/or ProcessObjects method(s))
  - ‘Info’ tag is used, but the defined value is overwritten/ignored
- ‘level’ tag in Definition XML is redundant
  - Gets automatically calculated by the system and should not be defined empty

Testing Maät

- Workbookconfig
  - +-40% tested
  - Up to date according to latest rules (baseclass per page etc.)
- All other tests:
  - Outdated according to latest rules
  - Outdated tests
  - → Re-analyse and adapt where necessary
  - Analyse all other pages & start writing tests
- Manual checklist:
  - WorkbookConfigPage-checklist = up to date
  - Paradigm checklist = up to date
  - All other checklists: Re-analyse & adapt where necessary