Aureos: Software Validation

**Software Quality Testing**

*B. Ruiz Sabido Bryan F.,  B. Lizarraga Franco Mauro J., B. De la Cruz Ramos Carlos J. & B. Martinez Contreras Yeshua Javier.  Universidad Politécnica de Yucatán, Tablaje Catastral 4448, Carretera Mérida-Tetiz. Km.4.5, 97357 Ucú, Yuc.* *Embedded Systems Engineering. Advanced Programming. November, 2018.*

All rights reserved. Universidad Politécnica de Yucatán. 2018.

**Abstract**

Before we can even start testing Aureos, it is quite important to consider the methods that are behind it. However, this report exists for getting conscious of how to proceed during the software validation stage, and even more important, to get efficient conclusions at the end of the software cycle itself. Not only this, but also as a requirement for the market commercialization.

**First Method: Black-Box**

Focusing primordially in the external behavior, ‘Black-Box’ allows to test any AR application, slightly efficient. In spite of focusing in how users get in touch with the functionality, the perspective should be, in how the environment interferes in such of convenient situations where Aureos take approach. Black-Box attempts to find errors in the following categories (Sommerville I.):

1. Incorrect or missing functions
2. Interface errors
3. Errors in data structures or external database access
4. Behavior or performance errors
5. Initialization and termination errors

Taking this into account, Black-Box take a new vision in how the user interacts with his normal paradigm of seeing the world that surrounds him; this translates to Aureos, for having an actual concentration in the camera movements, targets recognition and the database link that is being stimulated during the running of the app.

The regular sets of inputs and test cases that were mention before, staged on the Black-Box technique, making the delimitation of objectives, almost digestive; nonetheless, Black-Box only takes the relationship between the user and the external factors, which means that other scenarios are not taking into account. To solve this situation, another technique of validation is taken.

**Second Method: Monkey Testing**

As a matter of fact, XR apps use the graphic engine of mobile devices. This demand of resources of the system, can set up tons of events during the day life of the user. Monkey Testing will allow to test Aureos in almost every single scenario where the user and the app, get in touch. It performs the regular routines of an application; how many crashes can occur through different platforms and/or devices.

For allowing the use of Monkey Testing, terms are defined below:

* A number of testers will be defined
* Random devices should be reported
* The support to iOS and Android is mentioned in the system requirements, which means that any device with a lack of those, will not enter into the test.
* The routines that every user make of their apps will not contrast with the test cases.