**SISTEMA GESTOR SAMOC**

**PRODUCTO DE TRABAJO: TEST DE DESARROLLO**

**PROPOSITO**

Este Artefacto se utiliza para evaluar si un elemento de aplicación se lleva a cabo.

**RELACION**

|  |  |  |  |
| --- | --- | --- | --- |
| **Roles** | Responsable:   * Desarrollador | Modificado por:   * Desarrollador | |
| **Entradas para** | Obligatorio   * Desarrollar Casos de uso de prueba | Opcional:   * Implementar solución | Externo:   * Ninguna |
| **Salidas para** | * [Implement Developer Tests](http://epf.eclipse.org/wikis/openup/process.openup.base/capabilitypatterns/implement_developer_tests_6442995D.html?proc=_SuWj4dOPEdyqlogshP8l4g&path=_SuWj4dOPEdyqlogshP8l4g,_ieYHIVQwEd2i9JHp7xurvw,_51ewYdOPEdyqlogshP8l4g,_aSrtcdOOEdyqlogshP8l4g,_TV_k8dOKEdyqlogshP8l4g,_gXo2UNOKEdyqlogshP8l4g) | | |

**DESCRIPCION**

|  |  |
| --- | --- |
| **Main Description** | Este artefacto cubre todos los pasos para validar un aspecto específico de un elemento de aplicación . Por ejemplo, una prueba podría garantizar que los parámetros de un método aceptan correctamente los valores superiores y más inferior requeridos. Una prueba de desarrollador especifica entradas de prueba, condiciones de ejecución y resultados esperados. Estos detalles se identifican para evaluar un aspecto particular de un escenario .  Cuando recoja pruebas de desarrollador para un elemento específico de la aplicación , se puede validar que el elemento realiza como se especifica.  Las pruebas de ser auto-documentado de modo que es evidente tras la finalización de la prueba si el elemento de aplicación se ha ejecutadocorrectamente |
| **Brief Outline** | Although there is no predefined template for this work product, and testing tools affect how the work product is handled, you should address the following issues:   * Setup * Inputs * Script * Expected Results * Evaluation Criteria * Clean-Up |

**TAILORING**

|  |  |
| --- | --- |
| **Impact of not having** | If you do not run developer tests, you cannot ensure that elements that you modify over time are working. This can inhibit iterative development and maintenance. |
| **Reasons for not needing** | If you can embed the tests into the production code, you might not need a separate work product. Nonetheless, some level of support for developer testing is always necessary when you develop application software. |
| **Representation Options** | Suggestions and options for representing this work product:  **Suggestion: Automated code unit**  The most appropriate technique for running these tests is to use code that tests the implementation element scenarios and that you can run regularly as you update the system during development.  When code is the sole form of the tests, ensure that the code is self-documenting. The code should document the specifications of the conditions you are testing and the setup or clean-up that is required for the test to run properly.  **Option: Manual instructions**  In some cases, you can use manual instructions. For example, when testing a user interface, a developer might follow a script, explaining the implementation element. In this case, it is still valuable to create a test harness that goes straight to the user interface. That way, the developer can follow the script without having to follow a complicated set of instructions to find a particular screen or page.  **Option: Embedded code**  You can use certain technologies (such as Java(TM)5 Test Annotation) to embed tests in the implementation. In these cases, there will be a logical work product, but it will be assimilated into the code that you are testing. When you use this option, ensure that the code is self-documenting. |