وزارة التعليم العالي والبحث العلمي
جامعة الموصل
كلية علوم الحاسوب والرياضيات
قسم البرمجيات

تصميم وتنفيذَ أداةِ اختبار تكامل البرمجيات القائم على المسارات الاساسية للوحدات البرمجية

رسالة مقدَّمة
الى مجلس كلية علوم الحاسوب والرياضيات في جامعة الموصل
كجزء من متطلبات نيل شهادة ماجستير علوم في
البرمجيات

من قبل

سكينة خليل عزَّت عباس

باشراف
أ.م.د. ندى نعمت سليم

١٤٤٥هـ ٢٠٢٣ م

الخلاصة

تعنى هندسة البرمجيات بتطوير وتصميم برمجيات عالية الجودة منذ المراحل الأُولى من دورة حياة النظام البرمجي مرورا بمراحل التحليل والتصميم ثم برمجة واختبار النظام. تعد عملية اختبار البرمجيات طريقة للتحقق فيما إذا كان المنتج الفعلي يطابق المتطلبات المتوقعة، وللتأكد من أن منتج البرمجيات خال من العيوب يتوجب تنفيذ واختبار مكونات البرنامج أو النظام باستعمال أَدوات يدوية أو الية لتحديد الأخطاء أو الثغرات أو المتطلبات المفقودة لضمان الموثوقية والأمان والأداء العالي مِمَّا يؤدي الى توفير الوقت والتكلفة وإرضاء العملاء. وبشكل عام يصنف اختبار البرمجيات الى فئتين رئيستين: الاختبار الوظيفي والاختبار غير الوظيفي. يُعَدُّ اختبار الوحدة واختبار التكامل من الاختبارات الوظيفية المهمة، إذ يساعد اختبار الوحدة في معرفة عمل الوحدة الوظيفية بشكل صحيح ام لا، أَمَّا اختبار التكامل فيركز على بناء وتصميم البرنامج واختبار وحدات النظام عند دمجها وربطها مع بعضها لضمان عملها بدون أخطاء.

في هذه الرسالة، تم تصميم ألاداة "تصميم وتنفيذ أَداة اختبار تكامل البرمجيات القائم على المسارات الاساسية للوحدات البرمجية" لاختبار التكامل بين الوحدات الوظيفية للنظام بشكل آلي باستخدام تقنية اختبار المسارات الأساسية بين الوحدات المتكاملة. يهدف الاختبار إلى التحقق من جودة البرمجيات وضمان تطابق المنتج الفعلي مع المتطلبات المتوقعة. تبنت الرسالة فئتي اختبار البرمجيات الوظيفي، إذ إن اختباري الوحدة والتكامل يُعدّان من الاختبارات الوظيفية الرئيسية، حيث يركز الأول على تحقق عمل الوحدة الوظيفية بشكل صحيح، بينما يركز الثاني على اختبار وحدات النظام عند دمجها وربطها.

بالأَداة المطورة تم تنفيذ اختبار التكامل بين الوحدات الوظيفية للنظام عبر مراحل عديدة؛ إذ يبدأ بمرحلة بناء محلل القواعد البرمجية (Parser) لتحليل البرامج والوحدات الوظيفية المراد اختبارها والمكتوبة بلغة بايثون وتكوين شجرة النحو التجريدية (AST: Abstract Syntax Tree) , وفي المرحلة الثانية تم تكوين مخططات تدفق السيطرة للوحدات الوظيفية (CFG:Control Flow Graph) لتوضيح العلاقة بين الوحدات المترابطة وسلوك النظام. و باستخدام مخطط (CFG) يتم في المرحلة التالية اشتقاق مسارات التنفيذ الأَساسيَّة بين النماذج والوحدات التي يتكون منها البرنامج ثم توليد حالات اختبار لاختبار المسارات الرئيسية التي تم استنباطها واختبار تدفق الرسائل بين هذه الوحدات.

أظهرت النتائج أن ألاداة نجحت في تقليل الجهد والوقت المطلوب لعملية الاختبار مقارنة بالاختبار اليدوي. حيث بلغت دقة الاختبار أكثر من ٩٥%، وتم تحقيق نسبة تغطية عالية وصلت إلى ١٠٠.% كما إضافة هذه الأداة قيمة لعمليات اختبار البرمجيات حيث ساعدت في تحسين الكفاءة وتقليل التكلفة، مع تحقيق دقة اختبار عالية ونسبة تغطية كاملة.

Ministry of Higher Education and
Scientific Research
University of Mosul
College of Computer Science and
Mathematics
Department of Software

Design and Implementation of Integration Testing Tool based on Basis Paths of Software Modules

**A Thesis Submitted to the Council of the College of
Computer Science and Mathematics
University of Mosul
as a Partial Fulfillment of Requirements
for the Degree of Master of Science
in
Software**

By
Sukaina Khalil Izzat Abaas

Supervised by
Assist.Prof. Dr. Nada N. Saleem

2023 A.D. **1445 A.H.**

Abstract

Due to the presence of engineering, there is no high-quality software design from the early stages of the life cycle of the weak software system through the stages of mental analysis and then programming and testing the system. It is the result of testing for a reason whether the actual product conforms to specific requirements and to ensure that the product is free from lack of complete registration and testing of the program or system using new or advanced tools for a reason or request or required, officials and security, especially which leads to saving time and cost to customer satisfaction.

Software testing is generally classified into two main categories: functional testing and non-functional testing. Unit testing and integration testing are considered professional functions, as unit testing helps in knowing whether the functional unit is correct or not, but integration focuses on building and designing the program and testing the system units when they are integrated and linked to each other and prevent errors from failing.

In this thesis, the MM-Tool was partially designed to test the integration between complete functional units using the technique of testing the basic paths between integrated units (Model Base Path Test) to test the interfaces and results between interconnected units.

Through the presented tool, the integration between the functional units is tested through several stages. It begins with the stage of analyzing the programs and functional units to be tested, which are written in the Python language, by building a code parser (Parser) and forming the programmed parsing tree (AST) in the second stage, then completing the flow charts. Functional units (CFG) to control the compatibility between interconnected units and the behavior system. By monitoring the next stage, the basic execution paths between the units that have been developed are derived and the flow of messages between these units is tested.

An integrated testing tool (MMT-Tool), which was developed to test the basic paths between integrated and integrated units, was implemented on several programs through the study (case study) and achieved successful test results exceeding 95%, and the percentage of paths test segments was 100%.