مجلة جامعة تشرين للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (40) العدد (2) 2018
Tishreen University Journal for Research and Scientific Studies - Engineering Sciences Series Vol. (40) No. (2) 2018

Evaluating web application quality using Model Based Testing

Dr. Yousef Ibrahim Dalla [*]
Sally Mahmoud Jarkas[**]

(Received 29 / 5 / 2017. Accepted 20 / 3 / 2018)

□ ABSTRACT □

Software shifts more and more to the online world. Testing web applications is important to decrease cost and get high quality applications. Testing web applications requires documents that specify what their correct behavior is, and this makes testing difficult.

In this study, we will test web application based on a model-based testing approach. We use Model Based Testing (MBT) for detection failures in web applications, and to apply functional testing so we overcome the drawbacks of traditional testing approaches, then we compare MBT with automatic testing according to time, number of errors and coverage criteria. We have chosen for the model-based testing approach because it automates the testing process, adapts quickly to changes, it is time saving, and less error prone if the system is modelled correctly.

Keywords: Model based testing; web application, Finite state machine, automatic testing, functional testing, and random algorithm.

[*] **Associate professor, Department of Software and Information Systems, Faculty of Information Engineering, Tishreen University, Lattakia, Syria.**

[**] **Postgraduate student(Master) ,Department of Software and Information Systems, Faculty of Information Engineering, Tishreen University, Lattakia, Syria.**

مجلة جامعة تشرين للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (40) العدد (2) 2018
Tishreen University Journal for Research and Scientific Studies - Engineering Sciences Series Vol. (40) No. (2) 2018

تقويم جودة تطبيقات الويب باستخدام الاختبار المعتمد على النّماذج

الدكتور يوسف إبراهيم دلا *

سالي محمود جركس**

(تاريخ الإيداع 29 / 5 / 2017. قُبِل للنشر في 20/ 3 / 2018)

□ ملخّص □

تُركِّز معظم الشّركات البرمجية توجهاتها على نشر منتجاتها باستخدام تطبيقات الويب، ولهذا فإن عملية تحديد الأخطاء مهمة جداً لتقليل التّكلفة والحصول على تطبيقات بجودة عالية. تتطلب عملية اختبار تطبيقات الويب تحديد السلوك الصحيح من خلال التوصيفات التي غالباً ما تكون غير موجودة. وهذا ما يشكل عائقاً في تحديد الأخطاء الممكن ظهورها في التطبيقات.

سيتمّ في هذا البحث استخدام الاختبار المعتمد على النماذج Model based testing (MBT) لتحديد الأخطاء التي تحدث في تطبيقات الويب ولإجراء الاختبارات الوظيفية على النظام، وهذا يؤدي إلى تجاوز المشاكل التي تحصل في اختبارات تطبيقات الويب، ثم سيتم مقارنة أداء هذه الطريقة مع طريقة الاختبار الآلي وفقا لمعايير الزمن وعدد الأخطاء المكتشفة. تكمُن فعالية الاختبار المعتمد على النماذج في تطبيقات الويب بسهولة استخدامه للعمل الآلي والتكيُّف مع التغيرات السريعة إضافة إلى تخفيض الكلفة الزمنية والكشف المبكر عن الأخطاء في حالة النمذجة الصحيحة للتطبيق.

الكلمات المفتاحية: الاختبار المعتمد على النّماذج، تطبيقات الويب، آلة الحالة المنتهية، الاختبار الآلي، الاختبارات الوظيفية، الخوارزمية العشوائية.

* مدرس –قسم البرمجيات ونظم المعلومات–كلية الهندسة المعلوماتية–جامعة تشرين–اللاذقية–سورية

**طالبة دراسات عليا(ماجستير)–قسم البرمجيات ونظم المعلومات–كلية الهندسة المعلوماتية–جامعة تشرين–اللاذقية–سورية

مقدمة:

تتزايد تعقيد تطبيقات الويب نظرا للحاجة الملحَّة لذلك لزيادة طلبات المستخدمين من عدة نواحي كالأداء والأمان والتفاعل مع المستخدم، إضافة إلى الطبيعة الديناميكية والموزعة وغير المتجانسة التي تتمتع بها، وتوظيفها لعدد من لغات البرمجة. طرحت مجموعة Business Internet في San Francisco في عام 2003 دراسة أوضحت أن 70% من تطبيقات الويب تحتوي على أخطاء غير مكتشفة [1]. ومن هنا جاءت أهمية الاختبار باعتباره أفضل معيار لتحديد وثوقيّة تطبيق ويب. تأتي صعوبة تطبيق الاختبارات التقليدية على تطبيقات الويب نظراً لنموها الهائل، وعدم وجود تغطية جيدة لمتطلبات الاختبار الوظيفية، ومن هنا أصبح لا بد من توفير تقنيات اختبار برمجيات أكثر وثوقيّة وفاعلية.

تركّزت الجهود على دراسة الاختبار المعتمد على النّماذج[1] ، وهو تقنية تستخدم سلوك النِّظام الخاضع للاختبار لاستنتاج حالات اختبار، لفحص خصائص النّظام الوظيفيّة وغير الوظيفيّة بهدف الكشف المبكّر عن الأخطاء مما يؤدّي إلى تخفيض تكلفة تصميم الاختبار والحصول على منتج بجودة عالية، زمن أسرع وتكلفة أقل.

أهمية البحث وأهدافه:

تعاني تقنيات اختبار تطبيقات الويب التّقليدية من التَّكلفة العالية في مراحل التّصميم والتّنفيذ نظراً للتكرار وغياب وجود تغطية مناسبة. وهنا تكمن أهمية الاختبار المعتمد على النّماذج في حل هذه المشاكل عن طريق تأمين طرق تغطية مختلفة لتوليد عدد أكبر من حالات الاختبار بطريقة منظّمة، إضافة إلى أتمتة عملية تصميم وتنفيذ حالات الاختبار.

أشارت عدد من الأبحاث المنشورة في مجال الاختبار المعتمد على النماذج إلى عدم فعالية المعايير المستخدمة عند المقارنة [2]، وعانت من ضعف الأدوات المستخدمة[3] [2] ، إضافة إلى عدم اعتماد أسلوب المقارنة بين الخوارزميات المستخدمة في توليد حالات الاختبار، أو المقارنة مع طريقة الاختبار الآلي [7][6][5][4]. لذلك توجه البحث إلى عرض الطرق التقليدية لاختبار تطبيقات الويب، ثمَّ تطبيق أسلوب الاختبار المعتمد على النّماذج وفق معايير مختلفة ودراسة تأثيرها على تقييم نتائج البحث بشكل أفضل. تمّ اعتماد نموذج آلة الحالة المنتهية Finite State Machine في تصميم نموذج تطبيق الويب الخاضع للاختبار، ومن ثمّ توليد حالات الاختبار بتطبيق نوعين من الخوارزميات وتحليل الأداء الأفضل وفق معايير الزّمن، ونسبة تغطية العقد والوصلات. في المرحلة الآتية قمنا باقتراح دمج بين الخوارزميتين السّابقتين لنتحقق من إمكانية استخدامها في تحسين نتائج توليد حالات الاختبار وفق المعايير المختارة. اقترح البحث أفضل الخوارزميات من بينهم لتوليد حالات الاختبار باستخدام الاختبار المعتمد على النماذج، ومنه يتم تقييم الأداة المستخدمة. عَمِدَ البحث في المرحلة الآتية على مقارنة نتائج الاختبار المعتمد على النماذج مع النتائج المستخلصة من تنفيذ طرق الاختبار الآلي من حيث الزّمن والتغطية وعدد الأخطاء. وبهذا فالمقارنة الحاصلة توضح قدرة الاختبار المعتمد على النّماذج في تلافي الأخطاء المحتملة ومدى فعاليتها من ناحية الزمن والتغطية. نلخص أهداف البحث بالنقاط الآتية:

[1] **النموذج هو الشّكل المبسَّط والمجرَّد من سلوك وبيئة النّظام المرغوب.**

- مقارنة الخوارزميات المستخدمة في توليد حالات اختبار وذلك لمعرفة أثر هذه الخوارزميات على فعالية النتائج من حيث الحصول على حالات الاختبار المثلى، وتقييمها وفق معايير التغطية المناسبة والزمن.
- مقارنة بين بيئة الاختبار المختارة في البحث وبين الاختبار الآلي، لتوضيح أهمية الاختبار المعتمد على النماذج في تطبيقات الويب.

طرائق البحث ومواده:

يبدأ البحث بتعريف الاختبار بشكل عام ومن ثم ينتقل لشرح مبسط عن أنواعه والتّحديات عند استخدامه في تطبيقات الويب ومن ثم ينتقل للتّعريف بالاختبار المعتمد على النّماذج ومراحل عمله، ليتم الانتقال بعدها إلى الدّراسة التجريبية التي عملت على بناء نظام الاختبار يستهدف وظائف النظام كاملاً. ويتلخص بالمراحل الآتية:

- مرحلة تصميم النّموذج باستخدام الأداة . yEd
- توليد الاختبارات عن طريق خوارزميتي *A والخوارزمية العشوائية مستخدمين الأداة Graphwalker ومن ثم المقارنة بين نتائج هاتين الخوارزميتين، ومن ثم اقتراح استخدام خوارزمية دمج بين الخوارزميتين السابقيتين وتحليل نتائجها.
- تنفيذ الاختبار باستخدام الأداة .Selenium
- تحليل نتائج الاختبار.

ثم ستتم مقارنة نتائج الاختبار المعتمد على النّماذج مع الاختبار الآلي لتوضيح أهمية الاختبار المعتمد على النّماذج في تطبيقات الويب.

1- الاختبار

الاختبار هو مفهوم أساسي في تطوير البرمجيات، هدفه تقليل كمية الأخطاء التي تظهر في المنتج البرمجي. لا يمكن الكشف عن كافة الأخطاء وذلك نتيجة ضخامة البرمجيات التي تزداد احتمالات الخطأ فيها بشكل كبير، لذلك يتم تحديد مدى وثوقيّة وجودة المنتج البرمجي بناء على نتائج الاختبار. والاختبار هو عملية تقنية تتمثّل بتنفيذ وتجريب النتائج في بيئة مجهَّزة تتبع إجرائية معينة بهدف قياس واحد أو أكثر من خصائص الجودة التي توضح انحراف وضع المنتج الحالي عن المنتج المرغوب به من حيث المواصفات، وظيفة النظام، الوثوقيّة، قابلية الاستخدام، الكفاءة، السلوك وإمكانية الصيانة .[2] وفيما يلي سنذكر الطرق الأساسية لاختبار البرمجيات[3] ، وسيتم التطرّق للنوع الأخير لاحقاً بالتّفصيل لأنه يشكل موضوع البحث.

1- الاختبار اليدوي [8] : يقوم المختبر بتصميم وكتابة حالات الاختبار المُمكنة بناء على المتطلبات ويطبقها يدوياً على النظام الخاضع للاختبار (Software Under Test (SUT). هذا النوع مرن كما أنّه سهل ورخيص في حال كان النّظام صغيراً ولكنّه يحتاج إلى عمل كبير، إضافة إلى عدم القدرة على تحديث عمليات الاختبار في حال تغير النظام وإنّما يجب إعادة كتابتها من البداية. لا يفضَّل استخدام هذا النوع في اختبار تطبيقات الويب نظراً لحجمها الكبير، والكلفة العالية من أجل كل عملية تنفيذ.

2- اختبار شبه الآلي (الالتقاط والإعادة [8](capture and replay: هذه الطريقة مخصَّصة لاختبار التطبيقات التي تملك واجهات رسومية، حيث يقوم المختبر بتشغيل التطبيق الخاضع للاختبار وتسجيل التفاعلات التي

تحصل بين المستخدم والتطبيق، وبعدها يمكن أن يتم إعادة استخدام هذه التسجيلات في اختبار التطبيق مرة أخرى عند الحاجة. هذه الطريقة بسيطة ومرنة، لكنها مكلفة ولا يمكن التعديل على الاختبارات الناتجة.

3- الاختبار الآلي[8] : يتم فيها كتابة برامج صغيرة منشأة إما باستخدام لغة معروفة أو باستخدام أدوات ذات واجهات رسومية تقوم بتوليد الكود تلقائياً، تأتي أهمية الاختبار الآلي من خلال إمكانية تنفيذها باستمرار وبشكل متكرر دون الحاجة للتدخل البشري. ولكنّها تعاني من أنها لا تستطيع اختبار إلا ما يتم برمجتها عليه وهنا يمكن غياب الاختبار لأجزاء مهمة من النظام البرمجي.

4- اختبار البرمجيّات المعتمد على النّماذج. وسيتم شرحه في الفقرات الآتية.

2- الاختبار المعتمد على النماذج **Model Based Testing (MBT)**

عرّف[8] Utting & Legread الاختبار المعتمد على النّماذج على أنه "التوليد الآلي لحالات الاختبار انطلاقاً من نماذج وتوصيفات النظام"، يساعد تطبيق الاختبار المعتمد على النماذج في الكشف المبكر عن الأخطاء مقارنة مع الاختبار التقليدي. والحصول على أعلى درجة من العمل الآلي المؤتمت وذلك لتوليد أكبر عدد ممكن من حالات الاختبار، وتقليل التكلفة الزمنية والجهد اليدوي المبذول.

2-1 أنواع الاختبار المعتمد على النماذج

هناك نوعان من الاختبار المعتمد على النّماذج [7] هما Online MBT، Offline MBT.

- الاختبار ذو التنفيذ الآني Online يتم توليد وتنفيذ حالات الاختبار في نفس الوقت. هذا النوع من الاختبارات عشوائي ولكنّه يحقق أكبر تغطية لحالات الاختبار، وبهذا النوع يمكن أن نختبر الأنظمة البرمجية المتغيرة.
- الاختبار ذو التنفيذ المؤجل Offline يتم فيها توليد حالات الاختبار وتخزينها في ملف وعند الانتهاء يتم تنفيذها يدوياً، ممّا يساعد على بناء سلسلة من حالات الاختبار وإنشاء أدوات تدعم توليد حالات الاختبار آليّاً.

2-2 مراحل الاختبار المعتمد على النماذج

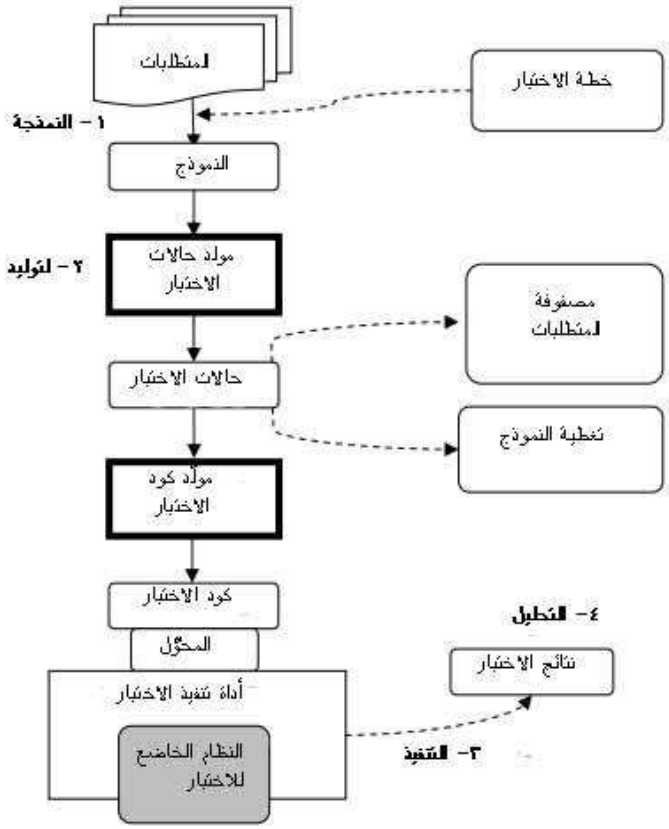
يمر الاختبار المعتمد على النماذج بالمراحل الآتية:[9]

1- إنشاء النموذج الخاص بعملية الاختبار اعتماداً على تحليل النظام المرغوب اختباره.

2- توليد حالات الاختبار المجرَّدة اعتماداً على تغطية النموذج ومصفوفة المتطلبات.

3- تنفيذ الاختبار وتتضمن هذه المرحلة تحويل سلاسل الاختبارات المجرَّدة إلى سلاسل اختبار بقيم فعلية.

4- تحليل نتائج الاختبار والتي تتضمن عملية تحديد الحاجة لتوليد اختبارات أخرى أو إيقاف عملية الاختبار ومن ثمّ تقييم وثوقية وجودة الاختبار.



الشكل (1) المراحل الأساسيّة للاختبار المعتمد على النّماذج[9]

2-2-1 النَّمذجة (إنشاء النموذج الخاص بعملية الاختبار)

يتم بناء نموذج الاختبار اعتماداً على التّوصيفات الخاصة بالنّظام بحيث يعكس السّلوك الصّحيح للنّظام وذلك بهدف توليد حالات اختبار وظيفيّة وعلى مستويات مختلفة من التّجريد[10]. حيث أنّ التّجريد يشير إلى عملية ربط الكيان مع النموذج دون الأخذ بالحسبان التّفاصيل غير المفيدة. تستخدم كلّ أداة من أدوات الاختبار المعتمد على النّماذج نموذجاً مختلفاً لتوليد حالات الاختبار. في الفقرة الآتية سيتمّ تسليط الضوء على اثنتين من لغات النَّمذجة المتداولة والمستخدمة في الاختبار المعتمد على النماذج.

2-2-1-1 آلة الحالة المنتهية **Finite State Machine (FSM)**

هي آلة مجردة تتكوَّن من عدد منتهٍ من الحالات وعدد منتهٍ من الانتقالات بين الحالات، ومن أفعال دخل وأفعال خرج. لابدّ من وجود حالة بداية وحيدة وممكن وجود مجموعة من حالات النهاية. آلية الانتقال في هذا المخطط تتم بناء على تابع يقوم بأخذ حالة النّظام الحالية والدّخل المُعطى ويعيد أفعال خرج جديدة وحالة جديدة. يمكن أن نعتبر آلة الحالة المنتهية كربط لتتالٍ مرتب من أفعال الدخل مع أفعال الخرج الموافقة لها. ونقصد بالفعل هو النشاط الذي يتم تنفيذه في لحظة معينة. [11]

2-2-1-2 لغة النّمذجة الموحَّدة :**Unified modeling language(UML)**

هي لغة قياسية تستخدم لتعريف، تصوير، إنشاء وتوثيق سلوك النظام. وقد تم تصميمها لتتوافق مع الطّرق غرضية التوجه، وتستخدم لوصف سلوك الأنظمة المعقّدة. تستخدم لغة النّمذجة الموحّدة المخططات لتمثيل النموذج ويوجد ثلاثة أنواع أساسية من المخططات وهي :[11]

●المخططات البنيويَّة structure diagram: تصف العناصر الأساسية المكوِّنة للنظام وتتضمَّن مخططات الصفوف و الأغراض .class , objects

●المخططات السلوكيَّة behavior diagram: تصف كيفية عمل النظام وتتضمَّن مخططات التدفق workflow ومخططات حالات الاستخدام. usecase diagram

●مخططات التفاعل interaction diagram: تصف البيانات وكيفية التحكم بتدفق البيانات في النظام.

تصف مخططات لغة النّمذجة الموحّدة النّظام بالتفصيل على كافة مستويات التجريد وهذا يؤدي إلى زيادة تعقيد النماذج الناتجة عنها، لذلك تُعدّ مخططات حالات الاستخدام كافية وحدها لنمذجة المهام الصغيرة بسلوك النظام.

2-2-2توليد حالات اختبار مجرَّدة

يتمّ توليد حالات الاختبار باستخدام أدوات تعتمد على المواصفات وعلى النّموذج المُختار من المرحلة السّابقة وتعتمد هذه الأدوات في توليد حالات الاختبار (سلاسل الاختبار) على خوارزميات متعددة منها المرور على البيان[8] ، حيث يتمّ محاولة المرور على كامل الوصلات أو كامل العقد مرة واحدة على الأقل وعملية المرور هذه نطلق عليها اسم التّغطية.

تساعد حالات الاختبار الجيّدة في تحديد الأخطاء بكلفة معقولة، لذلك في هذه المرحلة يتم اختيار معيار اختبار مناسب يملك القدرة على تحديد حالات الاختبار الجيّدة للنّظام المدروس. تأتي أهمية تحديد معيار تغطية الاختبار المناسب من أجل تقليل فضاء حالات الاختبار غير المنتهي. حيث أنّ ناتج هذه المرحلة هو عدد كبير جداً من حالات الاختبار المجردة في حين أنّ الهدف الأساسي هو الحصول على أقل قدر ممكن من حالات الاختبار بأعلى تغطية ممكنة للنموذج. سيتم مناقشة بعض معايير اختيار حالات الاختبار.[2]

●معيار تغطية بنية النموذج structural model coverage criteria: يتم استغلال بنية النّموذج في الوصول لأعلى تغطية، فعند تمثيل النظام بعقد وانتقالات في نموذج آلة الحالة المنتهية مثلاً، يتم قياس نسبة المرور بالمقارنة مع كامل العقد أو كامل الوصلات.

●المعيار المعتمد على الخطأ Fault based criteria: بما أنّ غرض الاختبار هو إيجاد الأخطاء في النظام الخاضع للاختبار، فإن هذا المعيار هو الأكثر استخداماً في الاختبار المعتمد على النّماذج. حيث تجري مقارنة بين النّموذج الأصلي والخرج المتوقع للنظام وبالتّالي يتمّ قياس عدد الأخطاء المكتشفة.

●معيار تغطية البيانات data coverage: يتمّ في هذا المعيار تقسيم مجموعة بيانات الدخل إلى صفوف متساوية ومن ثمّ يتمّ اختيار قيمة واحدة تمثل الصف كاملاً مع الأخذ بالحسبان أنّ كل قيمة من القيم المُمثلة للصفوف لها نفس القدرة على تحديد الخطأ. يمكن استخدام طرق مثل تحليل الحدود boundary analysis وتحليل المجال domain analysis [12] لتغطية البيانات وتوليد الاختبارات.

من المهم التنويه إلى استحالة تحديد أفضل معيار، وتكمُن هنا مهمة المختبر في تحديد جودة حالات الاختبار واختيار المعيار الأنسب. فعملية اختيار المعيار الأفضل هو موضوع لازال قيد الدراسة والبحث.

2-2-3 تنفيذ الاختبار

يتم تنفيذ حالات الاختبار الفعلية على النظام الخاضع للاختبار بهدف الكشف عن الأخطاء. وهذا يتطلب آلية ربط لإعطاء قيم حالات الاستخدام المطبقة على النظام الخاضع للاختبار قيماً فعليّة.

كما ذكرنا فإنّ النّموذج يكون مكتوباً بصيغة مجردة، ولكن عند التنفيذ فإن هذه الصيغة تحتاج لأن تتناسب مع صيغة النظام الخاضع للاختبار، لذلك نحتاج قبل عملية التنفيذ إلى تجهيز آلية ربط بين النّموذج والنّظام الخاضع للاختبار. يتم ذلك من خلال تحويل حالات الاختبار المجردة إلى حالات قابلة للتنفيذ وتقوم أيضاً بتجريد قيم الخرج الناتجة من النظام الفعلي لتكون قابلة للفهم من قبل النموذج باستخدام أدوات تحويل (محوّلات .(adapters

يقوم المحوِّل بترجمة حالات الاختبار المولَّدة من أدوات الاختبار المعتمد على النّماذج إلى دخل مناسب للنّظام الخاضع للاختبار، وكما أنّه يقوم بترجمة الخرج النّاتج عن النظام الخاضع للاختبار إلى دخل مفهوم من قبل أدوات الاختبار المعتمد على النماذج. هذا المحول يقوم داخليّاً بما يلي:

- التّنصيب :Setupتهيئة وإعداد النّظام الخاضع للاختبار ومن ثمّ تشغيله.
- ترجمة الدخل المولَّد من النّماذج إلى دخل قابل للاستخدام على النظام الخاضع للاختبار من خلال استدعاء الطرائق.
- التّجريد(Abstraction): إعادة ترجمة النتائج الفعلية الناتجة عن النظام الفعلي إلى قيم مجردة خاصة بالنموذج.
- إطفاء (Teardown): إغلاق النّظام الفعليّ عند نهاية كل حزمة اختبار، وذلك للإعداد لحزمة الاختبار الآتية.

4-2-2 الحكم **verdict**

هي عملية تحليل نتائج الاختبار وذلك بمقارنة خرج النظام الخاضع للاختبار مع الخرج المتوقع وناتج الحكم ممكن أن يأخذ إحدى القيم الآتية: نجاح، فشل، الناتج غير محدد ومن ثمّ تحديد إيقاف الاختبار أو إعادته.

الدّراسة التجريبية

فيما يلي سنقوم بوصف بيئة الاختبار التي تم إنشاؤُها لتحقيق الهدف المرجوّ من البحث وذلك لكل مرحلة بشكل منفصل.

1- **تطبيق الويب:** عملنا على إنشاء تطبيق ويب خاص بنظام متجر الكتروني باستخدام لغة البرمجة php لكتابة الجزء الخاص بالمُخدّم ولغات HTML5, CSS3, javascript لكتابة الجزء من التطبيق الخاص بالواجهات من جهة العميل. ومكوَّن من الصّفحات الآتية: الصّفحة الرئيسية، صفحة تسجيل وتعديل سجل البائع، صفحة خاصة بالمسؤول عن الموقع ليتمكن من البحث عن بائع ما، صفحة الزبائن، صفحة تسجيل وتعديل البضائع المعروضة إضافة إلى إضافة الطّلبات المرغوبة على هذه البضائع وصفحة الخطأ.

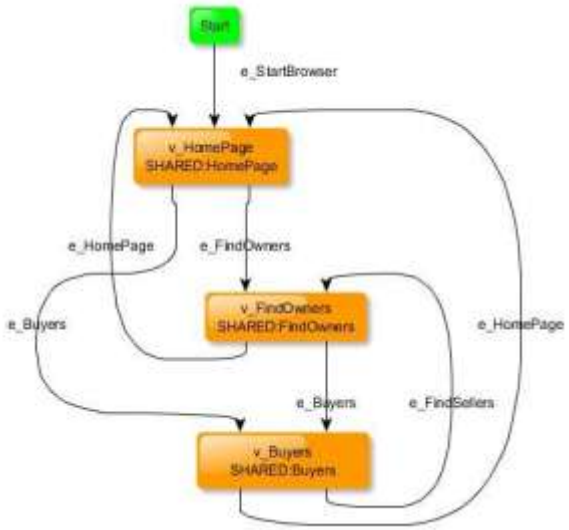
الشكل(2) الصفحة الرئيسية الخاصة بتطبيق الويب الخاص بالمتجر الالكتروني

2- **اختيار وبناء النموذج:** من بين العديد من النّماذج التي من المُمكن أن تصف حالة النظام تمَّ اختيار نموذج آلة الحالة المنتهية المذكور في الفقرة 2-2-1-1 لتصميم نموذج لتطبيق الويب الخاص بالمتجر الالكتروني، نظراً لقدرة هذا النموذج على محاكاة طبيعة تطبيق الويب الخاضع للاختبار المتمثلة بالّتناسق.

تمَّ تمثيل كلّ صفحة ويب بنموذج مكوَّن من عقد، كلّ عقدة تمثّل مكوِّناً[21] في صفحة الويب، ومثَّلنا كلّ انتقال بين الصفحات بوصلة، مع إمكانية استخدام الشّروط التي تخضع لها عملية الانتقال بين الصفحات. تم بناء ستة نماذج آلة الحالة المنتهية نموذج لكل صفحة ويب باستخدام الأداة yEd، حيث تمّ ذكر الحرفv قبل اسم كل عقدة دالاً على كلمة vertex(عقدة) والحرف e قبل كل وصلة دالاً على كلمة edge (حافّة أو وصلة).

تساعد هذه الأداة في بناء النموذج الخاص بالنّظام من خلال تسهيلها لتمثيل العقد والوصلات بطريقة نموذج آلة الحالة المنتهية، وأيضاً إمكانية حفظ النماذج بصيغةgraphml حيث يمكن لأداة الاختبار المعتمد على النّماذج قراءة هذه الصيغة من أجل إتمام عملية الاختبار. وأيضاً هناك ميزّة العقدة التّشاركية التي تقدمها هذه الأداة، حيث تعمل العقدة التّشاركية على الرّبط بين النماذج المشكِّلة لتطبيق الويب. وهذا مهم جداً عند اختبار الانتقال بين صفحات تطبيق الويب.

[21] **المكوّن هو أي عنصر من عناصر صفحة الويب ممكن أن يعطي حدثاً ما عند تفعيله كالأزرار أو الروابط**



الشكل(3) النموذج الممثل للصفحة الرئيسية باستخدام طريقة آلة الحالة المنتهية

3- **توليد حالات الاختبار**: تمَّ اختيار الأداة Graphwalker لتوليد صفوف مجرّدة من أجل كل نموذجٍ مصمَّم لصفحة تطبيق الويب، ويحوي هذا الصف طرائق مجرّدة من أجل كل عقدة وكل وصلة موجودة في النّموذج، ثمّ عملنا على تحقيق الطَّرائق المجردة لكي نتمكَّن من فحص هذه العقد والوصلات بالطريقة التي نحتاجها.

قُمنا في البحث باختبار السلوك الوظيفي للنّظام، ونقصد بالسلوك الوظيفي كيفيّة تصرّف النّظام عندما يخضع لأحداث معيّنة، ففي حال وجود نماذج خاصة بإرسال بيانات المستخدمين (Forms) ومن أجل عملية الاختبار نقوم بإدخال قيم للحقول تقع ضمن المجال المسموح، وأيضاً ندخل قيم لا تقع ضمن المجال المسموح لنتحقق من فعالية تطبيق الويب تجاه الأفعال غير الصحيحة من المستخدم (مثلا عند طلب رقم الهاتف ندخل أحرف أو نتركه فارغاً). وأيضاً نعمل في الطّرائق على تجربة الانتقالات بين الصّفحات، فمثلاً نختبر إمكانية الانتقال من الصفحة الرئيسية إلى صفحة البائع وهذا انتقال مسموح، أما لو حاولنا الانتقال من الصفحة الرئيسية إلى صفحة إضافة الطلبات المرغوبة على البضائع فهذا غير ممكن ولابد للنظام أن يعطي رسالة خطأ، وإلا فإنّ السلوك الوظيفي الخاطئ لتطبيق الويب غير مُكتشف بعد. وإذا حققنا هذا من أجل كل الحالات الممكنة نكون قد فحصنا وظائف صفحات تطبيق الويب والانتقالات بينها ضمنيًّا.

معايير التَّقييم المستخدمة:

نظراً لأن طريقة الاختبار المعتمد على النماذج حديثة، فإنّ معايير الاختبار المستخدمة لا تزال موضوع بحث فتكلفة إيجاد وإصلاح الأخطاء هي مرحلة منفصلة من مراحل تطوير المنتج البرمجي، لذلك سيتم اختيار معايير تقييم خوارزميات العبور على نموذج آلة الحالة المنتهية بهدف اختيار الخوارزمية الأفضل وبهذا نكون قد ضمِّنا تغطية بنية النّموذج وبالتالي تغطية المتطلبات .[12].

في بيئة الاختبار التي نعمل عليها اعتمدنا على نموذج آلة الحالة المنتهية، لذا تمثل نسبة تغطية العقد مقدار اختبار الأفعال في صفحة التطبيق الخاضع للاختبار، في حين تغطية الوصلات تمثل مقدار الانتقالات الممكنة بين هذه الصفحات، وهكذا نتمكن من تغطية المتطلبات وعملية الانتقال بينها. معايير التغطية المستخدمة هي:

a- نسبة تغطية العقد :[12] مقدار اختبار أفعال التطبيق وتُحسب $\frac{\text{عدد العقد التي تم زيارتها}}{\text{عدد العقد الكلي للبيان}} \times 100$

b- نسبة تغطية الوصلات :[12] مقدار اختبار الانتقال بين الصفحات وتُحسب $\frac{\text{عدد الوصلات التي تم زيارتها}}{\text{عدد الوصلات الكلي للبيان}}$ x 100.

c- زمن تنفيذ الاختبار [12]: هو الزّمن المستغرق من بداية تنفيذ الاختبار إلى نهايته، بعد حذف الزَّمن اللازم لتشغيل واجهة التطبيق وزمن تشغيل المتصفح. عملنا في هذا البحث على تقدير الزّمن بالدّقائق نظراً للوقت الكبير الذي تأخذه الخوارزميات في إيجاد المسارات التي تحوي عقداً أو وصلات جديدة.

قمنا باختيار خوارزمية المرور العشوائية وخوارزمية A* [31]، كطريقة لتوليد سلاسل الاختبار المكوَّنة من حالات الاختبار المجرَّدة من النّموذج المُصمَّم.

4- **تنفيذ الاختبارات:** يتمّ تنفيذ الاختبارات الفعلية الناتجة باستخدام الأداة Selenium المختصَّة بالتَّوليد الديناميكي لحالات الاختبار الفعليّة انطلاقاً من حالات الاختبار المجردة، كما أنه يقوم بتأمين سهولة التواصل بين تطبيق الويب وبيئة الاختبار ذات لغة البرمجة المختلفة عن اللُّغة التي تم تطوير تطبيق الويب بها. حيث يعمل المحوِّل الموجود ضمنيّاً في الأداة على تحويل حالات الاختبار المجردة المكتوبة بلغة الجافا إلى حالات اختبار مناسبة للغة تطبيق الويب [13]. جرت عملية تنفيذ الاختبار آنيّاً نظراً للسرعة التي تحققها هذه الطريقة عن نظيرتها في أسلوب الاختبار المؤجَّل. ثمَّ حصلنا على النتائج الخاصة بالاختبار وفق معايير الاختبار المختارة وبعدها تمَّ استخلاص وتحليل النتائج.

النتائج والمناقشة

تمَّ اعتماد خوارزمية المرور العشوائي في مرحلة توليد حالات الاختبار وخوارزمية A* وفيما يلي نستعرض نتائج تطبيق الخوارزميات بعد تنفيذها على النّظام الخاضع للاختبار.

1- الخوارزمية العشوائية المستخدمة في توليد سلاسل الاختبار تتبع أحد أسلوبين الأول زيارة أول عقدة لم يتم زيارتها بعد، مع مراعاة تغطية أكبر عدد ممكن من العقد، أو زيارة أول وصلة لم يتم زيارتها بعد، مع مراعاة تغطية أكبر عدد ممكن من الوصلات. تنفَّذ الخوارزمية العشوائية على كل نموذج بشكل منفصل وعند وصول الخوارزمية لعقدة تشاركية فإنها تدخل إلى النموذج الآخر لتتمّ زيارة عقده ووصلاته بطريقة عشوائية أيضاً.

عند استخدام الخوارزمية العشوائية في مرحلة توليد سلاسل الاختبار لكافة النّماذج المعبِّرة عن صفحات تطبيق الويب وبعد تنفيذ الاختبار حصلنا على النتائج المبينة في الجدول(1) حيث ظهرت فعالية استخدام الخوارزمية العشوائية في إعطاءها نسب تغطية عقد ووصلات عالية في معظم حالات الاختبار المطبقة.

الجدول(1) نتائج الاختبار بعد تطبيق خوارزمية المرور العشوائي على نموذج آلة الحالة المنتهية لتطبيق الويب

حالة الاختبار	الزمن(دقائق(	نسبة تغطية العقد(%)	نسبة تغطية الوصلات(%)
1	9	87	80
2	4	100	92
3	13	100	96
4	2	100	92

[31] **خوارزمية A*: هي خوارزمية عبور على عقد البيان بحيث يتم تحديد المسار الأقصر من العقدة المصدر إلى العقدة الهدف.**

5	9	100	92
6	2	100	95
7	6	93	88

2- عند استخدام خوارزمية *A من أجل توليد حالات الاختبار انطلاقاً من نماذج آلة الحالة المنتهية لتطبيق الويب فإننا في كل نموذج من النماذج، قمنا بتحديد عقدة البداية وتحديد العقدة المرغوب الوصول إليها، بعدها تعمل الخوارزمية على البحث عن أقصر طريق يصل بين هاتين العقدتين.

بتطبيق خوارزمية *A على كل نموذج على حده حصلنا على النتائج المُمثَّلة بحالات الاختبار من 1 إلى 5 المبينة في الجدول، أما حالة الاختبار 6 فهي تكافئ تطبيق خوارزمية *A على كامل تطبيق الويب والنتائج ممثلة بالجدول الآتي:

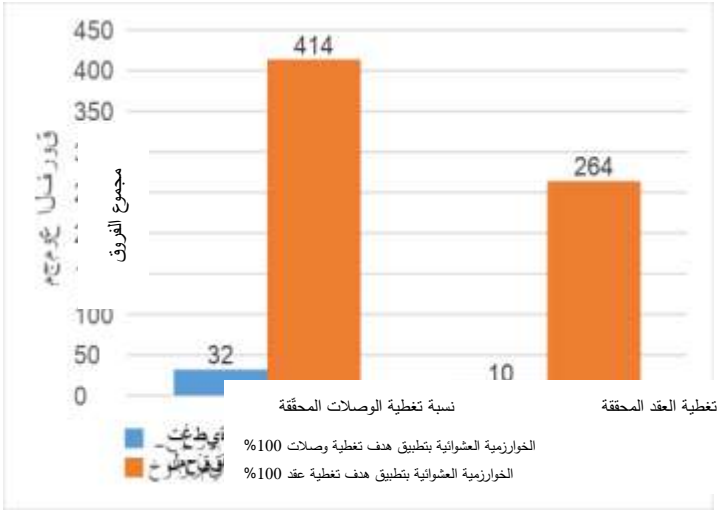
الجدول (2) نتائج الاختبار بعد تطبيق خوارزمية *A على تطبيق الويب

حالة الاختبار	الزمن(دقائق(	نسبة تغطية العقد(%)	نسبة تغطية الوصلات (%)
1	1	66	25
2	1	66	33
3	4	40	11
4	3	100	50
5	1	100	50
6	6	28	62

الزمن المحسوب والمذكور في الجدول(2) هو زمن الوصول إلى عقدة الهدف. ونلاحظ أن نسبة تغطية تطبيق الويب كاملاً هي نسبة ثابتة نظراً للمسار المحدد الذي تسلكه الخوارزمية في البحث عن أقصر طريق في كل نموذج.

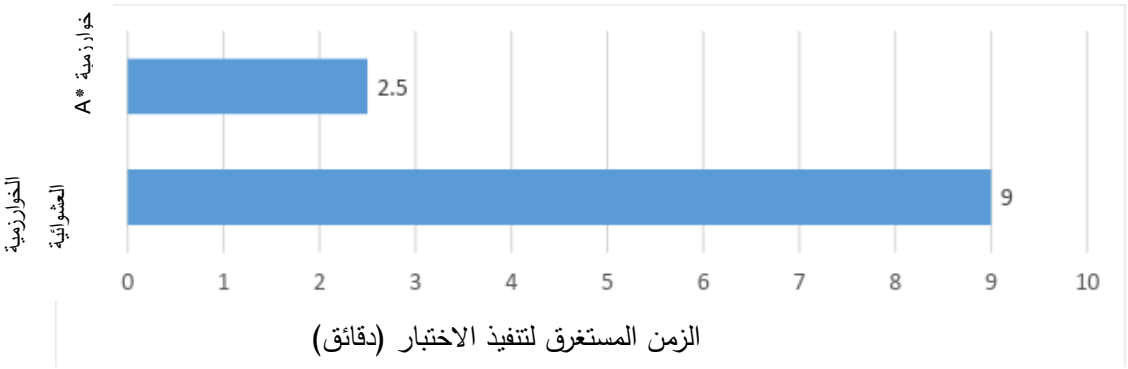
بيَّنت النتائج ضعف الخوارزمية عند تقييمها وفق معايير التغطية (عقد ووصلات) وتفوقها وفق معيار الزمن، ولا يمكن أن تغطي قوة الخوارزمية من ناحية الزمن على ضعفها من حيث التغطية، فضعف التغطية يعني عدم اختبار أحداث أو انتقالات ممكنة الحدوث فيه، وبهذا نكون قد غفلنا عن اختبار أجزاء مهمة من التطبيق ممكن أن تحوي على نسبة من الأخطاء

عند إجراء مقارنة بين الخوارزمية العشوائية وخوارزمية *A وفقاً للبيانات الموجودة في الجداول 1 و2، استنتجنا أنّ القيام بمقارنة حالات الاختبار بشكل مباشر لن يوضح الفروق في البيانات، لذلك احتجنا إلى إيجاد علاقة رياضية تعطي تمثيل بياني لتوضيح المقارنة بين البيانات وقد اخترنا مقياس مجموع فروق التّجارب السّابقة في كل خوارزمية من الخوارزميات المستخدمة حيث أنه كلما كانت الفروق أكبر كلّما كانت نسب التّغطية المحقَّقة أقل.



الشكل (4) – مقارنة بين نتائج الاختبار الناتجة من الخوارزمية العشوائية (العمود اليساري) وخوارزمية *A (العمود اليميني) من حيث نسبة تغطية العقد والوصلات المحققة

أظهرت النتائج ضعف الخوارزمية *A عند مقارنتها مع الخوارزمية العشوائية الموجهة بالوصلات وفق معايير التغطية (عقد ووصلات) حيث أنّ نسبة التغطية التي تعطيها الخوارزمية منخفضة جداً، أي يتم تجاهل الكثير من الصفحات والانتقالات دون اختبارها مما يقلل إمكانية كشف الأخطاء فيها. وفيما يلي نوضح مقارنة بين الخوارزميتين من ناحية الزمن.



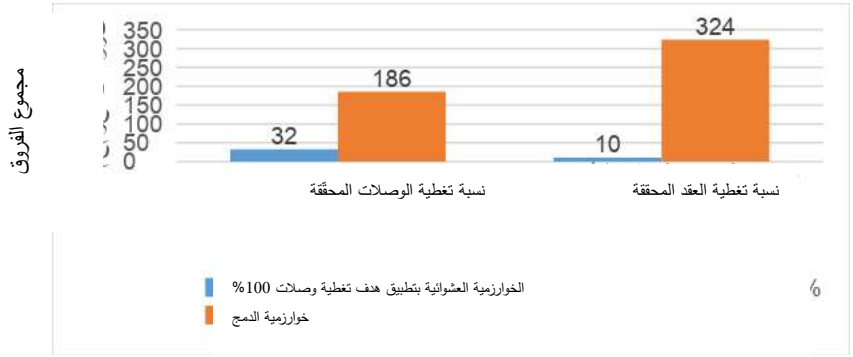
الشكل (5) مقارنة بين الزمن المستغرق لتنفيذ الاختبار عند تطبيق الخوارزمية العشوائية و بين خوارزمية *A

يتبين من النتائج المبيّنة في الشكل 5 تفوق الخوارزمية *A على الخوارزمية العشوائية من حيث متوسط الزمن المستخدم. استناداً للنتائج السَّابقة، عملنا في بحثنا على اقتراح تطبيق دمج بين الخوارزميتين السابقتين (الخوارزمية العشوائية و خوارزمية *A)، هذه الخوارزمية غير مقترحة في الأبحاث السابقة، هدف هذا المقترح الاستفادة من الزّمن القليل الناتج عن خوارزمية *A ومن التغطية العالية التي تنتجها خوارزمية المرور العشوائي، بهدف إيجاد كفاءة عالية للنتائج التي تعطيها عملية توليد حالات الاختبار، وبعد التطبيق تمَّ الحصول على النَّتائج الموضحة في الجدول الآتي:

الجدول (3) نتائج الاختبار بعد تطبيق دمج خوارزمية A* والخوارزمية العشوائية على نماذج تطبيق الويب

حالة الاختبار	نسبة تغطية العقد(%)	نسبة تغطية الوصلات(%)
1	28	56
2	36	68
3	28	62
4	76	87
5	68	81
6	20	50

عند إجراء مقارنة بين خوارزميتي الدمج والخوارزمية العشوائية وفق معايير التغطية يتوضّح تفوُّق الخوارزميّة العشوائية على خوارزمية الدمج ومنه تفوقها ضمنيّاً على خوارزمية A* في الحصول على تغطية أكبر لعقد البيان أي تغطية كامل مكوّنات صفحات تطبيق الويب المختبَر وأيضاً اختبار الانتقالات الممكنة بينها.



الشكل (6) مقارنة الخوارزمية العشوائية (العمود الأيسر) مع خوارزمية الدّمج (العمود الأيمن) وفق معايير تغطية العقد والوصلات

لم نتمكن من تحديد الزَّمن عند تطبيق خوارزمية الدمج، نظراً للمرور بحلقات غير منتهية تمّ إيقافها يدوياً. هذا الزمن غير المنتهي ناتج عن تطبيق الخوارزمية العشوائية على النموذج الأول مثلاً والذي يتوقف عند تغطية كامل الوصلات فعند الدخول في عقدة تشاركية يتنقل التنفيذ إلى نموذج آخر يطبق خوارزمية A* التي تنهي التنفيذ عند عقدة ما وتتوقف، مما سبب الدخول في حلقة لانهائية ناتجة عن انتظار الخوارزمية العشوائية في النموذج الأول لنتائج خوارزمية A* في النموذج الثاني حيث أن الناتج يتمثل بالوصول إلى عقدة تعيدنا للنموذج الأول أو الانتقال إلى نموذج آخر حتى الوصول إلى عقدة تعيدنا للنموذج الأول، وهذا ما لم يتحقق. الشكل الآتي يوضح مقارنة بين الخوارزمية العشوائية وخوارزمية الدمج من حيث الزمن وقد مثلنا الزمن في خوارزمية الدمج بعدد كبير نسبيّاً لتوضيح الزمن غير المنتهي.

الشكل (7) مقارنة بين خوارزمية الدمج والخوارزمية العشوائية من حيث متوسط الزمن اللازم لتنفيذ الاختبار

في المرحلة الآتية من الدّراسة عملنا على تصميم اختبار يدوي لكل صفحة من صفحات تطبيق الويب المنشَئ وتنفيذه آليّاً باستخدام الأداة Selenium. حيث قمنا بتنفيذ اختبار لكل عنصر من عناصر صفحات تطبيق الويب وفق تصميم الاختبار المنشَئ. وحصلنا من التجربة على نسبة انتقالات بين الصفحات تبلغ 92%. في الجدول الآتي نبيّن مقارنة بين الاختبار الآلي والاختبار المعتمد على النّماذج في حال استخدام خوارزمية توليد حالات الاختبار العشوائية، تمّت هذه المقارنة وفق معايير متوسط الزمن المُستخدم، عدد العقد والوصلات التي تمَّ زيارتها وعدد الأخطاء المُكتشفة في الصّفحات.

الجدول(4) مقارنة بين الاختبار المعتمد على النماذج والاختبار الآلي

المعيار المُستخدم	الاختبار المُعتمد على النّماذج	الاختبار الآلي
عدد الصفحات التي تم زيارتها	16	16
عدد الانتقالات التي تم تجربتها	25	23
عدد الأخطاء المكتشفة في الصفحات	5	1

من تحليل نتائج الجدول تبيّن تفوّق الاختبار المعتمد على النّماذج على نظيره الاختبار الآلي وفق معايير عدد الانتقالات والصفحات وعدد الأخطاء المكتشفة. تتضمن التكلفة الزّمنية الكليّة للاختبار زمن تصميم النموذج، الزمن اللازم لتوليد حالات الاختبار المجردة الناتجة عن عملية تصميم النموذج، إضافة إلى الزمن الخاص بتنفيذ حالات الاختبار الفعلية. تم تصميم وتنفيذ 7 سلاسل اختبار مختلفة في الاختبار الآلي، واعتماد الاختبارات التي تمّ تطبيقها في الاختبار المعتمد على النماذج التي تستخدم الخوارزمية العشوائية. نوضح في الجدول الآتي مقارنة تفصيليّة للزّمن اللازم للقيام بعملية الاختبار الآلي مقارنة مع الاختبار المعتمد على النماذج ويتوضح من النتائج تفوّق الاختبار المعتمد على النماذج على نظيره الاختبار الآلي من ناحية الزمن أيضاً.

جدول(5) مقارنة تفصيلية للزمن المستغرق في الاختبار المعتمد على النماذج مع نظيره الاختبار الآلي

	الاختبار الآلي	الاختبار المعتمد على النماذج
زمن تصميم سلاسل الاختبار مقابل تصميم النموذج(ساعة)	3	45 دقيقة
زمن كتابة السكريبت الخاص بالاختبار(ساعة)	2	1
متوسط زمن تنفيذ الاختبار (دقائق)	8	9
المجموع (ساعة)	5:8	1:54

الاستنتاجات والتوصيات

تمّ في هذا البحث دراسة الاختبار المُعتمد على النّماذج الذي تعتمد فكرته على توليد حالات اختبار مجرّدة من النموذج المُمَثِّل للنظام الخاضع للاختبار، وتطبيق مراحله المُختلفة وتمَ العمل على وضع دراسة لأفضل الخوارزميات المُمكنة لتنفيذ هذا النوع من الاختبار بدءاً من مرحلة النّمذجة باستخدام نموذج آلة الحالة المنتهية وانتهاءً بتنفيذ الاختبارات. وقد بيَّنا في هذا البحث أهمية استخدام الخوارزمية العشوائية في توليد سلاسل حالات الاختبار نظراً لتفوقها على خوارزمية .A* وللتأكد من فعالية الخوارزمية طبّقنا دمجاً للخوارزميتين السّابقتين وقارناها مع الخوارزمية العشوائية. وهنا أثبتت الدّراسة تفوُّق الخوارزمية العشوائية على الخوارزميتين الأُخريين وِفق المعايير المستخدمة في هذه المرحلة وهي الزّمن، وتغطية المتطلبات التي تتضمَّن تغطية العقد وتغطية الوصلات.

كما بيّنت النتائج أن الزّمن اللازم لتنفيذ الاختبار العشوائي الموجه بتغطية الوصلات أكبر من الزّمن اللازم لتنفيذ الخوارزميات الأخرى. هذا الاختلاف الزّمني يكون بنسبة دقائق، وهنا يمكن التغاضي عن هذه الدقائق على حساب تغطية أكبر للموقع الذي يتم اختباره لذا احتمال أكبر لكشف الأخطاء، وتعتبر المعايير المستخدمة في المقارنات مفيدة في تقييم النتائج بالشكل الصحيح.

من ناحية أخرى وبعد الوصول للنتائج السابقة تمّ توضيح تفوق الاختبار المعتمد على النماذج على نظيره الآلي وتحسينه لعملية الاختبار، حيث أنّه بالاعتماد على الشكل 2 والجدول 3 تبيَّن ما يلي:

-يساعد الاختبار المعتمد على النماذج في تقليل التكلفة الزمنية مقارنة مع الاختبار الآلي.

-في الاختبار الآلي تمّ توليد سلاسل الاختبار بناء على تصميم يدوي لسلاسل الاختبار، بينما في الاختبار المعتمد على النماذج تمَّ اختبار سلاسل اختبار مختلفة مولّدة آليّاً باستخدام خوارزميّات عبور مختلفة مثل الخوارزمية العشوائية، ممّا يعطي إمكانية تجربة سلاسل أكبر على النّظام المُختَبر، ومنه قدرة أكبر للكشف عن الأخطاء الوظيفية في النظام.

-تؤثِّر عملية التّصميم اليّدوية في الاختبار الآلي على الاختبار في حين تتمتَّع طريقة الاختبار المعتمد على النماذج بمستوى أعلى من الأتمتة.

-يؤدي تعديل على الاختبارات لاحقاً في الاختبار الآلي إلى تعديل كافة المسارات اللاحقة له وبالتالي تعديل كامل سلسلة الاختبار. أما في الاختبار المعتمد على النماذج ففي حال طلب أي تعديل على الاختبارات يكفي تعديل الطريقة المسؤولة عن هذا الاختبار في الصف الخاص بالنموذج المخصَّص، مع المحافظة على باقي النماذج كما هي وبدون تعديل على أي جزء آخر من الكود.

المراجع

1. ACHKAR, H. "Model Based Testing Of Web Applications" In Proceedings of 9th annual STANZ, Sydney, Australia 2010

2. MALIK, Y. M. "Model Based Testing: An Evaluation" , Blekinge Institute of Technology, ACM Computing Surveys May 2010,p.3-8.

3. SABHARWAL, S. BANSAL, P. AGGARWAL M."Modeling the Navigation Behavior of Dynamic Web Applications "International Journal of Computer Applications (0975 – 8887) Volume 65, March 2013, No.13.

4. LEGEARD B, PEUREUX F, and VERNOTTE A. "Model-Based Vulnerability Testing for Web Applications" FEMTO-ST Institute FRANCE, IEEE Sixth International Conference on Software Testing,. Verification and Validation Workshops (ICSTW) (pp. 445-452) 2012.

5. SABHARWAL, S. BANSAL, P. AGGARWAL M.” Modeling the Navigation Behavior of Dynamic Web Applications “International Journal of Computer Applications (0975 – 8887) Volume 65, March 2013, No.13.

6. ALEXANDER, A. OFFUTT, M. Testing Web applications by modeling with FSMs. Software & Systems Modeling, Springer Berlin Heidelberg, Volume 4, Issue 3, pp 326–345 2005.

7. VADLAMUD, L. N. Automated Model Based Testing for an Web Applications, international Journal of Science, Engineering and Technology Research (IJSETR) Volume 5, Issue 5, May 2016

8. UTTING, M. PRETSCHNER, A. and LEGEARD, B. A Taxonomy Of Model Based Testing Approaches, Jornal of Software Testing, Verification and Reliability, Volume 22, Issue 5, pp 297–212 2012.

9. Utting M. and Legeard B., Practical Model-Based Testing: A Tools Approach 2005: Morgan Kaufman Publishers Inc. San Francisco, CA, USA ©2007 ISBN:0123725011 9780080466484 p20, 22.

10. RAFIQUE, N.RASHID, N. AWAN, S. NAYYAR, Z. Model Based Testing in Web Applications, International Journal of Scientific Engineering and Research (IJSER) Volume 2 Issue 1, January 2014,57-58.

11. KULVINDER, S. and SEMMI, D.” Finite State Machine based Testing of Web Applications”, International Journal of Software and Web Sciences, 2013.

12. HANG, Z. Patrick .Hall and John H. R. May, Software Unit test coverage and adequacy, ACM Computing Surveys Vol 29, , December 1997 ,No 4 Pages 366-42.

13. www.seleniumhq.org/