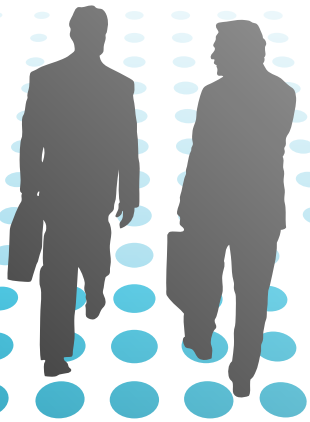
מגזין
עולם הבדיקות
רבעון שלישי 2015 | גיליון מס' 02
WWW.TESTINGWORLD.CO.IL

דבר העורכת | איריס קוטליצקי

קוראים יקרים, ברוכים הבאים לגיליון השני של מגזין עולם הבדיקות – מגזין המיועד לקהילת בודקי התוכנה בישראל.

גיליון הקיץ הגיע אליכם, ניתן להריח את הים, הרוח והחופשה.
בקיץ פעמים רבות הקצב קצת מאט וזה הזמן המתאים לחשוב על השלבים הבאים בהתפתחות וגדילה האישית והמקצועית ולהתחיל לתכנן... הרי התרגלנו מילדות שהלימודים מתחילים בסוף הקיץ.
במאמרים והכתבות המופיעים בגיליון זה תוכלו לראות חלק מהאפשרויות הרבות הקיימות להתפתח ולגדול, בצורה הפורמלית אך גם בצעדים קטנים בחיי היום יום. התפתחות ושיפורים משמעותיים דרך התבוננות אמיתית וכנה בטעויות. התפתחות בתחום העבודה ושמירה על עדכניות.
אנו שמחים לראות כיוון חיובי ביחס האקדמיה למקצוע הבדיקות, המתבטא בכך שיותר אוניברסיטאות מעודדות קורסי איכות ובדיקות במסגרת תואר ראשון ואף שני.
ITCB מקדמת תהליכים אלו בדרכים שונות, ומציעה מלגות לימוד לסטודנטים שיעסקו במחקרים אקדמיים בתחום.
גם אתם מוזמנים לתרום להתפתחות האישית של חבריכם הבודקים והתפתחות המקצוע ע"י הפצת הידע האישי לכלל דיון וחשיבה משותפת.
כתיבה שלכם הן בבלוג וגם לעיתון זה אם תרצו, מאפשרת לכם לגדול. עצם הצורך לעבד הנושא תוך כדי הכתיבה והן השיח שנוצר עם אנשים מהתחום שלכם אך לאו דווקא מסביבת העבודה שלכם – מאפשר גדילה משמעותית.
כמו כן תוכלו לקרוא בגיליון זה המלא בכל טוב מאמרים נוספים המרחיבים את הדעת: איך השימוש בהיגיון הפשוט ואינטואיציה נכון לתחום הבדיקות, מודל חדשני למבנה ארגוני ועוד.
אנו מודים על קבלת הפנים החמה למגזין החדש, ובמיוחד לחברות וחברים הרבים שנכנסו לקרוא את המגזין עשו Like ועזרו לקדמו.
וגם לאלו שהשתתפו בדיון המשך המעמיק על כתבתו של דורון בר בקבוצת הפייסבוק.

מאחלת לכם קיץ נעים מחשבות על גדילה והתפתחות אישית, צוותית וארגונית תוך כדי שימוש בהגיון ואינטואיציה.



מגזין עולם הבדיקות מתפרסם כל רבעון בגרסה אלקטרונית ומופץ חינם.
בכל רבעון תוכלו להתעדכן בטרנדים בעולם הבדיקות וכן במפגשים וכנסים המתקיימים בארץ ובעולם.
אתם מוזמנים להיכנס לאתר המגזין, אליו נעלה את הגיליונות החדשים – וכתבות בודדות בתקווה שיתפתח דיון פורה לגבי תוכנם.
www.testingworld.co.il
אם הנכם מעוניינים לקבל מייל ביציאת כל מגזין והדיונים המתפתחים בעקבותיו – הנכם מוזמנים לשלוח בקשה לכתובת register.testingworld@gmail.com
יש ביכולתכם להשפיע על התכנים שיופיעו במגזין.
לכל הערה, הארה, הצעה ורצון לכתוב אתם מוזמנים לפנות אליי במייל info.testingworld@gmail.com

בברכה,
איריס קוטליצקי

אם הבאגים קוראים לך!
יש לך רעיונות איך לבדוק טוב יותר?
אנחנו מחפשים אותך!

בוא/י לקחת חלק פעיל בקהילת הבודקים הישראלית – לכתוב על בדיקות.

בימים אלו צוות המגזין כבר שוקד על המהדורה השלישית. אנו מחפשים אנשי בדיקות נוספים שירצו לתרום ולפרסם מאמר מקצועי במגזין "עולם הבדיקות".
גם אם אין לך ניסיון בכתיבה או אם את/ה מתקשה לבטא הרעיון שלך בכתב, אנחנו כאן לעזור!
פנו אלינו לגבי הצטרפות לצוות הכותבים.

כותבים המעוניינים לקחת חלק בכתיבה עבור הגיליונות הבאים מוזמנים ליצור קשר: info.testingworld@gmail.com

מו"ל
Israeli Testing Certification Board
ITCB®

ניהול המגזין
יאן ברון, *iMDsoft*

ניהול התוכן
קובי הלפרין, *Ceragon*

עורכת
אירית קוטליצקי

עריכה
טל פאר, *Kongsberg Maritime*
איסי חזן
עמית וורטהיימר
משה ממיה

עיצוב גרפי
בית נלי מדיה
סטניסלב קולנקו
www.beitnelly.com

יצירת קשר
אימייל:
info.testingworld@gmail.com
הרשמה
register.testingworld@gmail.com
פקס:03-6176605
כתובת: ברוך הירש 14 בני ברק 51202

www.testingworld.co.il

www.itcb.org.il

מגזין עולם הבדיקות

עולם הבדיקות הינו מגזין רבעוני
כל הזכויות שמורות.
זכויות היוצרים על חומר שפורסם על-ידי המפרסם הינו רכושו של המחבר.
הדעות המובעות במאמרים והתוכן לא בהכרח מביעות את דעת המפרסם
המחברים הינם האחראים הבלעדיים על תוכן מאמרם.
אין לפרסם ו/או לשכפל בכל צורה שהיא את התוכן ללא אישור לקבלת אשור לשימוש בתוכן צור קשר במייל
info.testingworld@gmail.com

תוכן העניינים

דבר העורכת 2

לכתוב בלוג בדיקות | איסי חזן 4

בחן את עצמך | טל פאר..... 6

מחפש צרות | מיכאל שטאל 7

ריאיון עם מנהל בדיקות | עפר פרת, BMC 8

חדשות מעולם הבדיקות | קובי הלפרין..... 10

בדיקות תוכנה באקדמיה – מצב האומה | דני אלמוג ... 11

פינת הטיפים | קובי הלפרין 14

זוכה פרס מצויינות 2014 | עמיר ישראלי 15

האנציקלופדיה לבדיקות | אייל זילברמן 16

עזבו אתכם מיציבות בעבודה,
ביטוח קריירה כבר יש לכם?! | מאיה בוכניק..... 17

זיהוי והפעלה של פקדים גרפיים באוטומציה |
מאיר בר-טל 18

בדיקות מן העולם | אלון לינצקי 22

בחן את עצמך-תשובה | טל פאר 23

צוות הכותבים 25

לכתוב בלוג בדיקות | איסי חזן

מדוע ואיך להתחיל?

אתה קורא את המגזין הזה. על סמך עובדה זו הרשה לי להניח שאתה (או את...) לא אדיש, שאתה מוצא עניין במקצוע שלנו, אוהב ללמוד דברים חדשים ולקבל השראה מאחרים.

רוב הסיכויים הם שאתה קורא את המאמר למרות שהרעיון שאתה תכתוב בלוג[1] בדיקות נשמע לך הזוי לחלוטין. ואם לא הזוי –רעיון רחוק שאולי יהיה רלוונטי בעוד מספר שנים. גם אני, למרות שקראתי בלוגים של אחרים, חשבתי בדיוק כמוך. כל העניין היה נראה לי שמור למומחים ולא לבודקים "פשוטים" כמוני.
שתי עובדות או יותר נכון – שני בלוגים, שינו את דעתי. הראשון היה התייחסות (אוהדת) בבלוג של James Bach[2] לפוסט של בודקת "רגילה" ולא ידועית, בדיוק כמוני. העובדה השנייה הייתה הגילוי שבחור צעיר שעובד קרוב אלי באותה מחלקת בדיקות גם הוא כותב בלוג[3]. כך הבנתי שכדי לכתוב לא צריך להגיע קודם לאיזשהו מקום מיוחד. אפשר פשוט...לכתוב. ובאמת, לפני שנים התחלתי לכתוב את בלוג הבדיקות שלי[4].

> **"לכתיבה טובה יש כלל בסיסי אחד: אל תכתוב על דבר שלא אכפת לך ממנו"**

מה פתאום לכתוב בלוג?

סיבות טובות כדי לכתוב בלוג בדיקות.

- **לכתוב כדי שהעולם ידע**
כתיבת בלוג היא כלי למיתוג אישי שמשקף את הסגנון, המקצועיות והמומחיות שלך. קשר עם קוראים יכול להביא לקשרים חדשים ואולי גם הזדמנויות חדשות.
- **לכתוב כדי לזכור**
בלוג הוא יומן רשת. לרעיון שכתבת היום תוכל לחזור בעוד כמה שנים ולהשתמש בו כמו ברעיון שכתבת בעבר ביומן.
- **לכתוב כדי לפתח רעיון**
כתיבה היא דרך נהדרת לפיתוח רעיונות. היא עוזרת לנו לסדר את המחשבות שלנו בנושאים שונים וגם להעביר אותם לאחרים. כל מי שהתנסה בכתיבה או לימוד של נושא לאחרים יודע שזו הדרך הטובה ביותר ללמוד אותו בעצמך.
- הרשת מספקת לנו אפשרות להגיע לקוראים רבים ללא עלות ולקבל מהם פידבק מיידי על הרעיונות שלנו. העמדה של רעיון לביקורת של אחרים מאפשרת לנו לשפר ולשכלל אותו.
- **לכתוב במקום ללכת לפסיכולוג**
כתיבה מאפשרת להוציא החוצה ולפרוק חוויות שאנו עוברים בעבודה שלנו כבודקים.
לפני כמה שנים היה לי ויכוח מקצועי ש"לקחתי ללב" וגרם לי להרגיש צורך לכתוב. לא רציתי להביך את הצד השני לוויכוח, ולכן הפכתי את המעשה לסיפור פנטזיה של תכתובת דוא"ל בין מפלצת, גמד ושומר יערות. השינוי הפך את התוכן לראוי לפרסום, לא שינה את המסר ובסופו של דבר גם שדרג את הסיפור מבחינה ספרותית.

> **"כתיבת בלוג היא כלי למיתוג אישי שמשקף את הסגנון, המקצועיות והמומחיות שלך. היא יכולה להביא לקשרים חדשים ואולי גם הזדמנויות חדשות."**

איך באמת עושים את זה?

- לכתיבה טובה יש כלל בסיסי אחד: אל תכתוב על דבר שלא אכפת לך ממנו. כל השאר הן תוספות ועצות[5]. אם אתה בעצמך לא מתעניין במשהו, בוודאי שלא תצליח לעניין אחרים. היה עצמך. אחרת אתה מבזבז את הזמן.

איסי חזן

איסי (ישכשכר) חזן בודק תוכנה כבר יותר מ 15 שנה במרכז הפיתוח של אינטל בירושלים. הוא אוהב לחשוב ולדון בנושאים הקשורים לבדיקות ומשתתף מומחה בפורום הבדיקות של "תפוז". איסי ייסד את קבוצת המיטאפ Test.IL – המפגש הירושלמי בנושא בדיקות.
מידי פעם הוא כותב בבלוג שלו : http://testermindset.blogspot.com

1 על פי האקדמיה ללשון עברית המונח העברי לבלוג הוא יומן רשת.
2 www.satisfice.com/blog
3 http://gershon.info שמואל גרשון
4 http://testermindset.blogspot.com
5 הרעיון מופיע בספר Weinberg on Writing

לכתוב בלוג בדיקות | איסי חזן

- **על מה לכתוב?**

אין זה משנה אם אתה מתחיל או מתקדם, מקורי במיוחד או "הולך בתלם", לחוויה האותנטית שלך יש ערך בראש ובראשונה לעצמך.

כשאתה כותב, חשוב על עצמך ועל מה שתפיק מעצם הכתיבה – תהליך החשיבה והסיכום. את הקוראים הפוטנציאלים שים בצד כבונוס נוסף ולא כמטרה בפני עצמה.

אנו לומדים את המקצוע ב"בית הספר של החיים" במגוון צורות: מניסיונות טובים ורעים בעבודה, מחוויות אישיות מחוץ לעבודה, ומחשבות שבאות כתוצאה מקריאה ולימוד של עבודתם של אחרים. חשיבה ביקורתית על עצמנו ועל מוסכמות היא כלי הכרחי ללימוד והתפתחות. חשיפה שלה בבלוג הופכת אותו לרלוונטי עבורנו ומעניין עבור הקוראים הפוטנציאלים.

> **"כל מי שהתנסה בכתיבה או לימוד של נושא לאחרים יודע שזו הדרך הטובה ביותר ללמוד אותו בעצמך."**

- **אנגלית או עברית?**

זו החלטה אישית שתלויה בעיקר בשאלה באיזו שפה אתה מרגיש נוח יותר. כתיבה באנגלית תחשוף אותך לקהל רחב יותר. כתיבה בעברית תחשוף קהל יותר רחב בארץ לרעיונות שלך. אני בחרתי באנגלית, אבל שומר במגירה את הרעיון לכתוב גם בעברית.

- **ממה להיזהר?**

מכל דבר שהיית נזהר מלפרסם ברשת. שמור על זכויות יוצרים והישמר מצורת כתיבה לא הולמת. אל תכתוב דברים שהמעסיק שלך לא ישמח לקרוא או חומר חסוי, אבל אל תוותר על חשיפת חוויות ממקום העבודה באופן גורף.

> **להתראות ברשת! אל תשכחו להזמין אותי לקרוא את הפוסט הראשון. אני מבטיח לבוא ולפרגן.**

אני מקווה שהעברתי אליך את החשק לכתיבה. אני מציע לך לשוטט קצת ברשת ולקבל השראה מבלוגים אחרים. הבלוגרים הבאים מהווים לי השראה:

- **Pradeep Soundararajan** הוא מספר סיפורים נפלא. הבלוג שלו[1] מלא סיפורים מעניינים, ולא חוסך מעצמו ביקורת. קריאת הבלוג הפופולרי של האדם שמתפאר בכך שהוא "בודק התוכנה עם ניסיון של 7,400,000 שגיאות" נותנת אומץ לכתיבה שאינה חוששת מביקורת עצמית והופכת אותה לכלי ללימוד והתפתחות[2]. לאחרונה הבלוג שלו הפסיק להיות פעיל אבל הוא מאגר של פוסטים שאפשר לחזור ולהתענג עליהם כל פעם מחדש.
- **James Bach** הוא אחד האנשים המשפיעים בעולם הבדיקות. הוא אינו חושש מלהתעמת ולומר את דעתו בנושאים שונים כמו למשל הפוסטים שלו בנושא ISO 29119 או Certification. הוא משמש לי כמודל ליושרה.

להתראות ברשת! אל תשכחו להזמין אותי לקרוא את הפוסט הראשון. אני מבטיח לבוא ולפרגן.

1 http://testertested.blogspot.co.il

2 מומלץ לקרוא את הפוסט של פרדיפ שעוסק בדיוק בנושא של מאמר זה Why good software testers should come out of the well?

סקרים

מה הסקרים אומרים?...

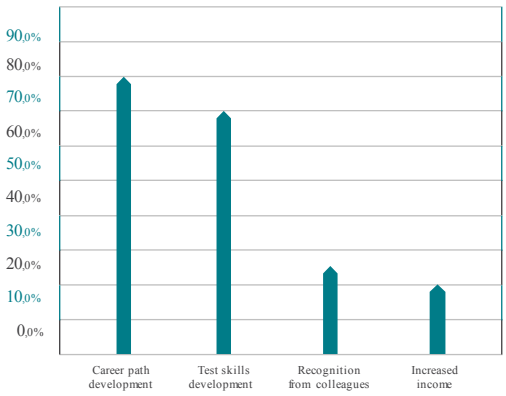
יאן ברון, iMDsoft

ISTQB International Software Testing Qualifications Board

טור זה מפוזר במספר בלוקים ברחבי המגזין ומציג ממצאי סקרים המציגים מגמות מרחבי העולם לעיונכם.
היום, אנו מציגים את דו"ח היעילות של ISTQB®,
ממצאי הסקרים 2013-2014.
בסקר זה השתתפו בודקים ומנהלי בדיקות מ-70 מדינות.
קהילת הבודקים מישראל הייתה מהפעילים ביותר שהשתתפו- 500 משתתפים.

ISTQB Effectiveness Survey 2013-14

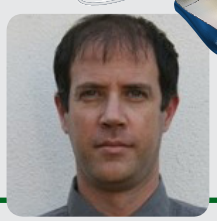
As a Test Engineer, what was your motivation in obtaining your ISTQB® CTFL certification?



מרבית מהנדסי הבדיקות עשו הסמכה בסיסית (ISTQB® CTFL) בכדי לקדם את מסלול הקריירה שלהם (80%) ולשפר את יכולות הבדיקה שלהם (70%), 25% מאמינים שההסמכה עוזרת להם לקבל יותר הכרה מחבריהם לעבודה ו 20% יגדיל את הכנסתם

Source: ISTQB® Effectiveness Survey Report

בחן את עצמך | טל פאר



השאלות במבחני ®ISTQB

מסווגות לפי דרגות שנקראות K-level. אלו רמות קוגניטיביות שמטרתן לסווג את מטרות הלמידה (learning objectives). השאלות לרמת Foundation מסווגות כך:

K1 (זיכרון) – על הנבחן לזכור או לזהות מושג או קונספט.

K2 (הבנה) – על הנבחן לבחור הסבר למשפט שקשור לנושא בשאלה.

K3 (יישום) – על הנבחן לבחור את היישום הנכון של קונספט או טכניקה וליישם אותה בפתרון השאלה.

K4 (ניתוח) – על הנבחן להפריד מידע הקשור לטכניקה או תהליך לצרכי הבנה טובה יותר של השאלה ולהבחין בין עובדות והפרעות.

בפעם הקודמת שאלנו שאלה מרמת K1. השאלה שנבחן היום תהיה מרמת K2, כלומר נצטרך לבחור הסבר מתאים למצב שמתואר בשאלה.

הערה קטנה: בעבודת היום יום נוהגים רבים מאיתנו להשתמש במונחים בשפה האנגלית והמונחים בעברית נשמעים לעיתים מוזרים ולא נכונים. בבחינות הנערכות בשפה העברית החליטו אנשי ITCB שכל המושגים המקצועיים יהיו בעברית אך בסוגריים יינתן המושג באנגלית.

השאלה של היום:

לפניך פרוייקט שמייצר גרסת תחזוקה (maintenance) למערכת בנקאית גדולה ומרובת סיכונים. בדיקות הרגרסיה (regression testing) מכילות 23525 מקרי בדיקה (test cases) שצריך להריץ על כל גרסה שמגיעה לבדיקות. איזה מהבאים מתאר את התועלת של בדיקות אוטומטיות עבור הפרוייקט הזה?

התשובות האפשריות:

1. דיווח תקלות בצורה קלה יותר
2. עלות נמוכה של מיכון (אוטומציה) של כל הבדיקות
3. הזדמנות ללמוד יכולת חדשה
4. צמצום עבודה ידנית חוזרת

***לצפייה בתשובה המפורטת – דפדפו לעמוד 24**

מרבית מנהלי הבדיקות מאמינים כי ההשתתפות בתכנית ההסמכה של ®ISTQB תשפיע באופן חיובי על איכות מוצריהם (75%) ותאפשר להם לספק נתיב קריירה חיובי לעובדיהם (55%).

חלק ממנהלי הבדיקות מאמינים כי תדמית החברה תשופר (30%) כתוצאה של השתתפות עובדיהם בהסמכת ®ISTQB.

סקרים

מחפש צרות | מיכאל שטאל

פחות זה יותר

לפני שנים רבות, כשעבדתי בבדיקות צ'יפים במפעל היצור של אינטל בירושלים (Fab8), הייתי שותף לוויכוח אופייני בין מפתחי מערכת אוטומציה. המפעל בירושלים לא עבד בשבת, והמטרה של מערכת האוטומציה הייתה לאפשר הרצת מכונות הבדיקה לתוך השבת בלי צורך במפעיל אנושי. הוויכוח היה על שפת התיכנות: C או Pascal? (כן... עברו כמה שנים מאז).

הטענות של שני הצדדים היו משמעותיות: מה קל יותר לעשות בכל שפה; איזו שפה יעילה יותר; באיזו שפה יש ספריות טובות יותר; וכו' וכו'. בנוסף, שני הצדדים טענו שאין ממש צורך להסכים על שפה אחת כי אפשר לכתוב מודולים בשפה אחת ולהשתמש בהם כספריות בשפה האחרת.

הוויכוחים נמשכו לאורך זמן, ואם זכרוני אינו מטעה אותי בסופו של דבר הקוד נכתב ברובו ב-C כי זאת הייתה השפה המועדפת על מי שכתב את רוב הקוד...

שנים אחר כך הנחיתי קבוצת בודקים בכתיבת מערכת לאוטומציה של בדיקות תוכנה – וחוץ מהעובדה שהוויכוח שם היה על Visual Basic מול #C, השאר היה מאוד דומה: הוויכוחים האין סופיים; פגישות טלפוניות ופנים-אל-פנים במטרה לשבת עד שיצא עשן לבן; הבאת הוכחות מכל צד למה הגישה שלו נכונה – והכל תוך המשך עבודה במקביל בשתי השפות כשכל צד מנסה לייצר מספיק קוד על מנת להבטיח שהשפה "שלו" תבחר.

מי צודק?

לדעתי – כל אחד ואף אחד. הטיעונים של כל צד נכונים, וברוב המקרים אי אפשר להכריע את הוויכוח על סמך טענות טכנולוגיות. השוואה של ה"בעד" של שפה אחת מול ה"בעד" של השפה השנייה היא בעצם וויכוח על למה שתי השפות קיימות, וברור שלכל אחת יש יתרונות מסוימים. כשמשווים את היתרונות מול החסרונות של שפה אחת קשה להעריך איזה פרמטר חשוב יותר. זאת גם הסיבה שהוויכוחים מתמשכים: אין טיעון מחץ שמטה את הכף. למעשה, אי אפשר להגיע להסכמה המבוססת על עובדות והגיון. כל מהנדס ימשיך לחפש את הטיעון האולטימטיבי לטובת השפה "שלו".

אכן, יש מקרים שבהם הפתרון קל כי יש יתרון ברור וחד משמעי לאחת השפות בהקשר של הפרוייקט הנדון. אבל מה עושים כשזה לא המצב, ולכל שפה יתרונות וחסרונות?

במצב כזה חשוב להבין שהשיקול העיקרי צריך להיות מספר השפות ולא

> אי אפשר להכריע ברוב המקרים את הוויכוח על סמך טענות טכנולוגיות. השוואה של ה"בעד" של שפה אחת מול ה"בעד" של השפה השנייה היא בעצם וויכוח על למה שתי השפות קיימות.

איזו שפה:

1. יש "אנרגיית אקטיבציה" לכל שפה נוספת. בכל מעבר משפה אחת לשנייה המהנדסים יבזבזו זמן להיזכר מחדש בפרטים טכניים. יתכן ויהיה צורך בשכפול כלים (כגון: IDE, Code Control) ומערכות תומכות כמו Nightly build ו-Continuous Integration. המהנדסים יצטרכו ללמוד לשלוט ביותר כלים. כל זה עולה כסף וזמן.
2. מהנדסים לא אוהבים במיוחד לעשות שימוש חוזר בקוד של מישהו אחר. יש תמיד הרגשה של "ייקח לי פחות זמן לכתוב את זה בעצמי מחדש". הסיכוי לשימוש חוזר בקוד שאינו בשפת התכנות שהמהנדס בקיא בה, הוא מזערי.
3. אם תצטרכו להעביר אחריות ממהנדס אחד לשני (עקב עזיבה, שינויי תפקיד וכו'), יתברר לכם שהשפה בה כתוב הקוד מצמצמת את מרחב התמרון שלכם מבחינת מועמדים מתאימים. גם בגיוס, תצטרכו לצמצם את החיפוש לאנשים ששולטים בצורה טובה בשתי שפות או לשלם את עקומת הלימוד של השפה שהמהנדס החדש לא בקיא בה.

> ומה אם אין אף תשובה שמכריעה את הכף? אז להישאר עם שתי שפות? במקרה כזה, גרירת ההחלטה לאורך זמן היא יקרה לא פחות מהעלות של בחירה קצת פחות נכונה.

מסקנה: ההחלטה הנדרשת היא ניהולית ולא הנדסית. על המנהל להקשיב לטיעונים משני הצדדים על מנת לוודא שאכן מדובר כאן במקרה של שוויון (אין שיקול ברור לאחד הכיוונים). ברגע שזה התברר, השיקולים הופכים ניהוליים ולא הנדסיים:

- באיזה כלי תמיכה הארגון משתמש?
- איזה שפה מוכרת היטב ליותר מהנדסים בקבוצה?
- מהי השפה המועדפת על האדם שעליו תוטל כתיבת רוב הקוד?
- מהי השפה שבה כתוב רוב הקוד שיש לכם כרגע?
- באיזה שפה כתובות מערכות שיתממשקו עם המערכת שלכם ויתכן שתצטרכו לעדכן אותן?
- וכו'

שימו לב שהשאלות לא דנות כלל בעדיפות הטכנולוגית של שפה אחת על השנייה! בנוסף, אלה שאלות אובייקטיביות שקל לענות עליהן. אם מי מהן לא עוזרת להחלטה (למשל: אין שפה אחת שבאופן בולט ידועה יותר למהנדסים), אפשר להוריד את הפן הזה מסך השיקולים.

ומה אם אין אף תשובה שמכריעה את הכף? אז להישאר עם שתי שפות? במקרה כזה, גרירת ההחלטה לאורך זמן היא יקרה לא פחות מהעלות של בחירה קצת פחות נכונה. אין הבדל ממשי אם ההחלטה נופלת לכאן או לכאן. אז מכניסים יד לכיס ו...

זורקים מטבע!

ריאיון עם מנהל בדיקות | עפר פרת, BMC

ספר לנו קצת על עצמך

שמי עפר פרת, אני בן 46, נשוי לתמי ואב למאיה, יעל ויונתן.
אני חי בקבוץ שמיר בגליל העליון ועובד מאז שנת 1999 בחברת BMC, חברה גלובלית שעוזרת ללקוחות לנהל בצורה אופטימלית את מרכזי המחשוב הגדולים שלהם. מאז 2003 אני עובד בסניף של BMC בפארק התעשייה תל-חי, שמצפון לקריית שמונה.
https://www.linkedin.com/in/oferprat

מתי בפעם הראשונה עבדת בפרוייקט אג'ילי?

לפני עשר שנים עבדתי בפרוייקט קשה. מה היה קשה? שהדרישות השתנו בקצב מסחרר. נכנסנו לכאוס של דרישות משתנות, דיזיין משתנה, בלאגן. ניסיתי לעבוד עם שותפי להנהלת הפרוייקט לייצב אותו, אבל לא הצלחתי. חיפשתי פתרון שיגן על צוות הבדיקות ויאפשר לנו להתייצב ולייצר בדיקות טובות בלי להתרוצץ כל הזמן אחרי שינויים. באותו זמן קראתי לראשונה על אג'ייל. קראתי את המניפסט האג'ילי, ואת הרעיונות הבסיסיים שמאחורי שיטות עבודה אג'יליות וחשבתי לעצמי: הנה שיטת עבודה שמקבלת שינויים, מצד אחד, אבל גם עושה שקט, לפחות זמני, מצד שני. החלטתי לנסות ליישם עקרונות אג'יליים בצוות הבדיקות, בלי לשתף את צוות הפיתוח. זה קונספט מוזר, אני חייב להודות. עבדנו בספרינטים של שבועיים. בתחילת כל ספרינט היינו מתכננים את תכולת הבדיקות לאותו ספרינט עם מנהלי המוצר והפיתוח, ומרגע שהסכמנו על התכולה, רצנו שבועיים ללא שינויים. לצוות הבדיקות זה עשה שקט ואיפשר לנו להתרכז ולגמור נושאים. יתכן שגם לצוות הפרוייקט זה עשה שקט, אבל אני קצת סקפטי בעניין הזה.

עבדת שנים רבות בשיטות שאינן אג'יליות. איך היה המעבר לאג'ייל?

רוב חיי המקצועיים עבדתי בפרוייקטים שנוהלו בשיטות Waterfall שונות. כאשר עברנו לעבוד בסקראם, ואני מתכוון לזה שעברנו לעבוד בסקראם באמת, ולא לעבוד על עצמנו שאנחנו אג'יליים, חוויתי משבר רציני בתפיסת התפקיד שלי כמנהל צוות בדיקות. קודם כל, שלבי התכנון השתנו לחלוטין. הייתי רגיל לעבודה על מסמכי דרישות מפורטים, לאחריהם מסמכי דיזיין מפורטים, ולבסוף תכנית בדיקות מפורטת. עכשיו הייתי צריך להתמודד עם תהליך שפירק את פירוט הדרישות למקטעי זמן שונים, ברזולוציות שונות, מסמכי דיזיין נכתבו רק עבור חלקים שבהם נדרשו במפורש, ותכנית הבדיקות, שתלויה מאוד בשני סוגי המסמכים הללו, נשארה קצת תלויה באוויר. איך מתמודדים עם מצב כזה? איך מבטיחים שנדע מראש שנוכל לבדוק את כל התכולה?

דבר שני, הניהול היומיומי של אנשי הבדיקות עבר לרמת צוות הסקראם. כלומר, איבדתי את נתח העבודה של מעקב יום יומי אחרי העבודה, קביעת קדימויות, פתרון בעיות סנכרון בין צוותים ועוד. הרגשתי לא נחוץ, וזאת תחושה לא נעימה במיוחד. מעבר לזה, הייתי צריך להתמודד עם שאלות של גבולות, כמו למשל, מהי הרזולוציה המתאימה לי למעקב אחרי הפרוייקט? האם להגיע לכל הפגישות היומיות? לבקש עדכון שבועי? מתי להתערב?

קושי שלישי היה כרוך ביכולת שלי להציג סטטוס בפני ההנהלה הבכירה, שמצד אחד הייתה מאוד מעוניינת בעבודה אג'ילית, מצד שני המשיכה לבקש סטטוס במונחים ומדדים של Waterfall. למשל, נדרשתי להציג התקדמות הבדיקות באחוזים של בדיקות שבוצעו מתוך בדיקות שתוכננו. המדד הזה משמעותי, אולי, בסיטואציה שבה יש כמות ידועה מראש של בדיקות, שנסמכת על דרישות מפורטות ודיזיין סגור, ואז ניתן אולי לראות התקדמות באחוזים, צפוי מול מצוי. היינו צריכים לעבוד על מדדים אחרים, שייצגו את המצב בצורה מתאימה יותר לאופי העבודה. מדדים כמו כמה user stories כוסו, ועמדו ב-Definition of Done.

לקח לי זמן למצוא את מקומי מחדש בסקראם, והיום אני מרגיש די בנוח. אני מבין שתפקידי הוא לעזור לצוותים כמירב יכולתי על-ידי עזרה באספקת כל מה שנחוץ לצוות (מידע, אנשים, מכונות, ידע, כלים ועוד), פתרון בעיות והסרת חסמים, הגנה על הצוות מפני רעשים והפרעות, עזרה בקבלת החלטות שלא מתקבלות ברמת הצוות, או שיש לגביהן מחלוקות. אני גם מתווך בין הצוות לבין ההנהלה הבכירה, מעביר את האסטרטגיה העסקית לצוות, ומעביר מידע מהצוות אל ההנהלה.

אני נהנה לראות את אנשי הצוות גדלים ומצליחים בתוך צוותי הסקראם. אני נהנה לראות אותם תופסים בעלות משותפת אמיתית על המוצרים שאנחנו מפתחים, תורמים להם רעיונות, מעצבים אותם תוך שותפות מלאה עם המפתחים, מנהלי המוצר, מעצבי חווית המשתמש, אנשי התיעוד וכל יתר השותפים לעשייה. אני נהנה לראות את אנשי הבדיקות לוקחים על עצמם את תפקיד ה-Scrum Master ומניעים את הצוות כולו להצלחה.

מה אתה עדיין לא מצליח לעשות? מה קשה לך?

אני לא מרגיש הצלחה רבה ביכולת שלנו לממש כמה שיותר בדיקות כבדיקות מקודדות, להבדיל מבדיקות ידניות. אנחנו תלויים יותר מידי בבדיקות ידניות וזה מקשה עלינו לשחרר גרסאות בתכיפות גבוהה יותר, תוך שמירה על רמת האמינות שאנחנו צריכים. השינויים התכופים במוצר, הפוקוס על הטווח הקצר והעומס עלינו, גורמים לנו להתמקד בבדיקות הידניות ולהקדיש פחות מידי זמן לקידוד. אנחנו עובדים עכשיו לשנות את זה.

ריאיון עם מנהל בדיקות | עפר פרת, BMC

למה אתה מתייחס לבדיקות מקודדות, ולא לאוטומציה, או לבדיקות אוטומטיות?

זה איזה ג'וק שנכנס לי בראש, אחרי ששמתי לב שמנהלים שאין להם ניסיון מעשי בבדיקות הולכים שבי אחרי המילה "אוטומציה". הם רואים בדמיונם תהליך פשוט שבו לוקחים בדיקות ידניות והופכים אותן לאוטומטיות, כמו צ'רלי צ'פלין על פס הייצור בסרט "חיים מודרניים". הם מפקששים לפחות שתי נקודות. קודם כל, את כמות ההשקעה הנדרשת לקודד ולשמר את הבדיקות כך שימשיכו לעבוד גם כאשר המוצר משתנה, אבל מעבר לזה, הם מפקששים את התובנה שבדיקות ידניות ובדיקות מקודדות הן שונות מאוד ביכולות שלהן. היכולת של בודק אנושי לקלוט בצורה אינטגרטיבית מסך שלם במבט יחיד, היכולת להעריך אם סדרת פעולות בתוכנה זורמת או לא, היכולת לסטות מהתסריט ולזהות כשלים נוספים, כל היכולות הקוגניטיביות הללו חסרות לתוכנה. לעומת זאת, היכולת של מחשב לבדוק אנליטית כמות גדולה של פרטים, לחזור שוב ושוב על אותן פעולות, לכסות API שלם, אלה יכולות שלאדם קשה מאוד ואולי בלתי אפשרי להתמודד איתן. מה שאני שואף לעשות הוא להסתכל על מכלול הבדיקות, מרמת ה-Unit Tests, דרך Component Tests, Integration Tests, Functional Tests ובדיקות מערכת מקצה לקצה, ולשאול איזה סוג בדיקות נחוץ לאיזו רמה, והאם ניתן לממש את הבדיקות הללו בקוד.

פספורט קבוצתי – BMC Performance and Availability , תל חי:

- **סוג המוצר הנבדק**

TrueSight Operations Management

מערכת Client-Server מורכבת המסוגלת לנטר מערכות IT גדולות, מרמת מערכת ההפעלה, הרשת, האפליקציות ועד לחוויית משתמש הקצה, ולהגיש את כל המידע הזה בצורה מרוכזת וברורה לאנשי האופרציה האמונים על שמירת על בריאות המערכת. יש לנו שם מגוון רחב של טכנולוגיות, מבסיסי נתונים ו-application servers ועד ממשקי משתמש בדפדפן ובמובייל.

- **גודל הקבוצה**

הצוות מונה כחמישה עשר איש ואנחנו מגייסים עוד. לכל אנשי הצוות הכשרה במדעי המחשב או הנדסת תוכנה. ארבעה אנשים מקודדים בדיקות על בסיס קבוע, פחות או יותר.

- **וותק הבודקים**

הכי צעיר נמצא בצוות שנתיים. זקן השבט עם עשרים שנות וותק בתעשייה. הממוצע הוא סביב 8-9 שנים בחברה.

- **מבנה הקבוצה**

רוב האנשים מחולקים לצוותי הסקראם השונים. יש לנו אדם אחד שאחראי על תשתית האוטומציה, והוא ממש צוות סקראם כשלעצמו...

- **סוגי בדיקות**

מכסים את כל קשת הבדיקות כולל אבטחה, ביצועים, עומסים, ופונקציונליות. בחלק מהמקרים אנחנו מקבלים תמיכה ושירותים מצוותים ייעודיים כמו למשל בדיקות גלובליזציה, אבטחה או ביצועים.

- **שיטת עבודה**

עובדים בסקראם.

- **כמות המוצרים הנבדקים וקצב שחרור גרסאות.**

חלק מהמוצרים משתחררים על בסיס רבעוני. הסרברים משתחררים כל חצי שנה ויותר.

חדשות מעולם הבדיקות | קובי הלפרין

כיצד למצוא מידע ב-Twitter

באופן די מפתיע, אנשי ההייטק הישראלים אשר לרוב מובילים בחדשנות, לא ממש התחברו לטוויטר (Twitter). אנו רואים כמות קטנה מאוד של בודקים פעילים בכלי תקשורת זה (ניתן לספור על 2 ידיים בערך), וכמות מעט יותר גדולה של מתכנתים (בעיקר יועצים למיניהם), בזמן שבשוק העולמי ישנה פעילות ענפה של עדכונים ודיונים טכניים.

היתרון הגדול בטוויטר – הנו שהוא מאפשר לכם גישה ישירה למומחי בדיקות בעלי שם עולמי, אינכם צריכים "להתחבר" אליהם ולחכות שיאשרו אתכם, אתם פשוט יכולים לקרוא את עדכוניהם ואף להגיב עליהם ולהשתתף בדיונים המתקיימים.
החיסרון של טוויטר – די בקלות ניתן להגיע להצפת מידע כך שקשה למצוא את ההודעות המעניינות מתוך ה"רעש" הסובב אותן.
סינון מידע בעזרת #Tags – כמעט לכל כנס, מפגשון, מגזין וארגון "טאג" המאפשר לנו כבודקים המשתתפים באירוע (אולא) לעקוב אחריו, לחלוק מידע ולפתח דיון עם משתתפים אחרים – לדוגמא למגזין עולם הבדיקות אנו משתמשים ב- #ttstwrld – ובעזרתו תוכלו לעקוב אחר עדכונים לגבי המגזין, ודיונים שמתקיימים בעקבות הכתבות המופיעות בו. וכמובן לדיונים כלליים לגבי בדיקות נשתמש ב #testing.

לא יודעים אחרי מי לעקוב – התחילו מרשימת הבודקים הישראליים:
https://twitter.com/ITCB_IL_Testers/lists/israeli-testers-qa
(פעילים ולא שם? – שילחו לנו הודעה ונוסיף)

איך אתם בוחרים כלי לניהול בדיקות / ALM?

אז באיזה כלי להשתמש לניהול בדיקות? – כל מספר שבועות אנו חוזרים לשאלה זו בפורומים ובקבוצות בפייסבוק, ונדמה כי שיטת הבחירה מתחלקת ל-2:

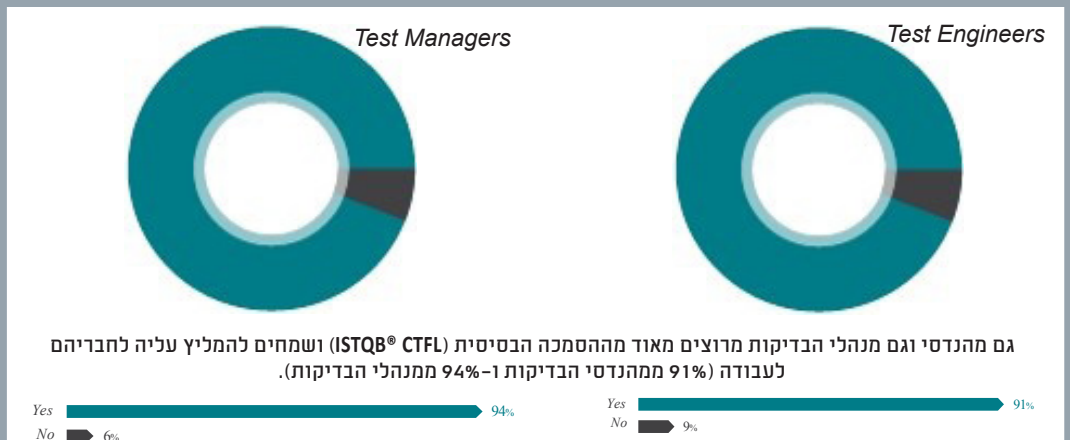
1. "שיטת האצבע" – מישהו בחברה עבד בעבר או שמע על כלי...
2. ניתוח צרכים – הגדרת הפרמטרים שחשובים לנו בכלי, הגדרת משקלות לכל פרמטר, בחינת מס' כלים ושקלול הציונים שקיבלו.

לצערנו עד היום לא קיימת רשימת פרמטרים מרכזית מסודרת לבחירת כל סוג של כלי, וכך אנו מפסידים זמן בבניית רשימות אלו מחדש בכל חברה, ומאבדים את היתרון של משוב ושיפור בעזרת חוכמת ההמונים.

השיקולים העיקריים לבחירה בין הכלים הנם המחיר: חינמי/פתוח, כ-שירות SaaS, מסחרי, התוכן: כלי משולב לכל שלבי הפיתוח (ALM) או בנפרד לניהול באגים, בדיקות, דרישות.
לא משנה איזה כלי תבחרו – רק אל תתחילו עם מסמכים ללא DB – אח"כ קשה מאוד לבצע את המעבר לכלי מסודר (ברגע ששמור במבנה נתונים – קל יחסית לעבור מכלי לכלי).

> **לא משנה איזה כלי תבחרו – רק אל תתחילו עם מסמכים ללא DB**

Would you recommend the ISTQB® Foundation Level (CTFL) certification to your colleagues?



גם מהנדסי וגם מנהלי הבדיקות מרוצים מאוד מההסמכה הבסיסית (ISTQB® CTFL) ושמחים להמליץ עליה לחבריהם לעבודה (91% ממהנדסי הבדיקות ו-94% ממנהלי הבדיקות).

Source: ISTQB® Effectiveness Survey Report

סקרים

בדיקות תוכנה באקדמיה – מצב האומה

דני אלמוג

דני אלמוג – חוקר ומרצה נלווה באוניברסיטת בן גוריון ובמכללת SCE, חבר במספר גופים ישראלים ובינלאומיים העוסקים באיכות ובדיקות תוכנה, משמש כאיש קשר בין התעשייה לבין האקדמיה בנושאים המקדמים איכות ובדיקות תוכנה. חבר בAB בארגון ITCB .
תחומי עניין מרכזיים ופעילות מחקרית: מיכון תהליכי בדיקה, הנדסת תוכנה, איכות ושיטות מתקדמות לפיתוח קוד וניהול פרויקטים בתוכנה. בעבר – 30 שנות ניסיון עשיר בכל תחומי הפיתוח והמחקר של תוכנה , פיתוח כלים ותשתיות למיכון תהליכי בדיקות. בתפקידו האחרון שימש כדירקטור ומוביל של תהליכי פיתוח וחידוש למיכון של בדיקות תוכנה בארגון פיתוח תוכנה גדול.
לינקדאין Almog.dani@gmail.com

לאן הולכת ההכשרה באקדמיה בנושאי איכות ובדיקות תוכנה

הקדמה:

בשנים האחרונות אנו חוזים שינויי הדרגתי במעמד והיחס אל בעלי המקצוע שעוסקים באיכות ובדיקות תוכנה. שנוי זה אולי גם נובע מהנזקים האדירים מהם סובל העולם בגין פיתוח תוכנה לא ראויה – שמעתי לא מכבר הערכה שרק בארה"ב מצטבר הנזק השנתי הזה לכדי 70 מיליארד דולר. מומחים מחלקים את הנזק לשנים:

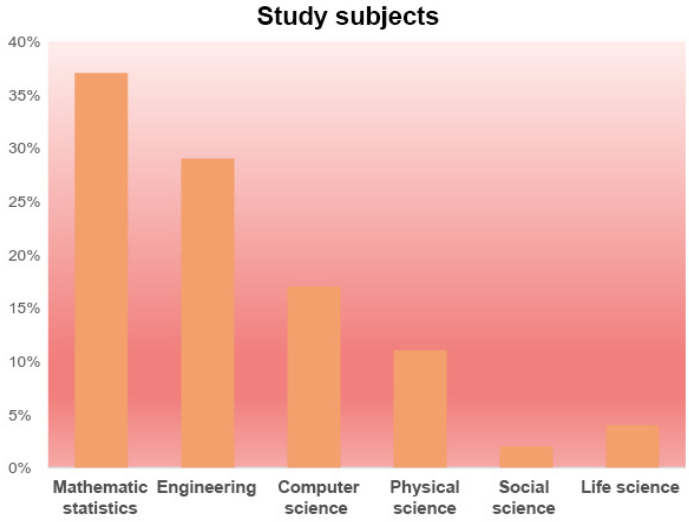
1. הנזק הישיר שנגרם בגין באג או תפקוד לא ראוי של התוכנה – למשל במכשור רפואי בו יכולים החולים למות, או יכולה (וקורית) הפסקת חשמל בגלל בעיית תוכנה במערכת הבקרה ואזור שלם מושבת מכל פעילות.
2. העלות שנגרמת בגלל הצורך לתקן את המעוות.

מזה שנים אני עוסק בצורה אינטנסיבית בקידום הכנסת מקצועות איכות ובדיקת תוכנה לתוך תכניות הלימודים בתוך המוסדות האקדמאים בארץ ובעולם. ניסיון זה מראה לי שקיים צמא בתעשייה לבוגרים המיומנים הרבה יותר בכל נושאי איכות ובדיקות התוכנה ושילובם בתהליכי הפיתוח. מצאתי שרבים מבוגרי המוסדות האקדמיים אותם אני מכיר, עוסקים בבדיקות תוכנה (לפחות בתחילת דרכם המקצועית) ניסיוני מהתעשייה היה שמי שהתחיל במסלול זה צייד עצמו היטב להיות מפתח ומוביל הרבה יותר טוב. גם אלו שלא פנו לפיתוח פיתחו לעצמם קרירה ומעמד מקצועי מכובד מאוד בתעשייה.
בהמשך דברי, אנסה להציג מקצת הממצאים המאפיינים שינויים אלו בתעשייה בעולם ובישראל – שהיא ללא צל של ספק אחת המדינות המובילות בעולם בתחומים אלו.

> **"מדען בונה על מנת ללמוד ואילו מהנדס לומד על מנת לבנות"** – פרד ברוקס

הנדסת תוכנה וכיצד היא רלוונטית לתהליכי איכות ובדיקות

ברצוני להבהיר אי הבנה בנושא לימודי הנדסת תוכנה – ראשית עלינו להבין שהתואר הוא תואר בהנדסה – לא במדעי המחשב. חלקו השני של שמו הוא "תוכנה" – לא מחשב. אז עם כל הכבוד ללימודי מדעי המחשב עלינו ללמד בעיקר תוכנה על כל משמעויותיה – לצערי יש מספר בתי ספר שלא הפנימו את ההבדל. התוכנה משתנה כל הזמן וזה מכתיב לנו דינמיות ושוני רב בנושאי ומקצועות הלימוד הנדרשים. למיטב הכרתי יש הבדל בין לימוד מדעים לבין הכשרת מהנדסים הבדל זה בא לידי ביטוי בעיקר במהות הדרישות מהתואר ואופי תהליכי הלמידה.
הדגשים בהכשרת תלמיד המדעים הם: ביכולות האישיות, הבנה עמוקה בחומרים, הפנמה ושימוש באותם הבסיסים התאורטיים לצורך גיבוש צורת חשיבה וביטוי הנדרש כחוקר. החוקר נדרש פחות להפגין את יכולת הביצוע שלו – נצפה יותר להדגשת אופני חשיבתו ויכולת הבעתם. אצטט: "מדען בונה על מנת ללמוד ואילו מהנדס לומד על מנת לבנות" – פרד ברוקס
מהנדס, לעומת זאת הוא מקצוע פרקטי יישומי שתפקידו ומבחנו הוא התוצאה. מהנדס הוא אדם תכליתי שתפקידו להגיע לתוצאות בשיטות יעילות, איכותיות וכנונות. הנדסת התוכנה של ימינו אינה מקצוע אינדיבידואליסיטי. חלק בלתי נפרד והכרחי של עבודת המהנדס הינה עבודת צוות ושיתוף פעולה. לכן מוכנסים לתוך תכניות הלימודים המקצועות ה "רכים" של יכולות תקשורת, ניהול, עבודת צוות ועוד. מי שאינו מכליל מקצועות אלו בתהליך הכשרת המהנדס עושה עוול אדיר למהנדס הטרי ולתעשייה. כמובן שחלק עיקרי בהכשרה הנו הקניית כלי חשיבה ויכולת התמודדות עצמית עם תהליכי למידה (בדיוק כנדרש מחוקר מדעי המחשב) מהנדס שאינו ממשיך ללמוד ולהשתפר בידע וביכולותיו מוצא עצמו טכנאי החוזר על אותן הפעולות במשך כל הקריירה שלו.
אחת הנקודות הלוהטות בדיונים האחרונים בפורומים ובכנסים העוסקים בהנדסת תוכנה היא מהם הנושאים המרכיבים את הדרישות המקצועיות הנדרשות ממהנדסי התוכנה. יש לציין שישנה תנועה רחבה של מרצים ומחנכים באקדמיה המרגישה בצורך להתאים את תכניות הלימוד לצרכים המעשיים של מהנדס התוכנה בשטח. למתעניינים אני ממליץ לעיין בשני ספרי ייסוד שחודשו לאחרונה (2014) על ידי IEEE שיזם חידושם של:SWBOK (Software) Engineering Body Of Knowledge זהו ריכוז המידע בנושאי הנדסת תוכנה[1] SWECOM (Software Engineering Competency Model) המתאר את היכולות והדרישות המקצועיות הנדרשות ממהנדס התוכנה. למותר לציין שכל נושא האיכות, בדיקות ועוד מאוד מפותח בשני מסמכים אלו. מעיון בשני מסמכים אלו ניתן להבחין על פניו שהקשר בינם לבין מה שמלמדים הוא קטן מאוד, כפי שמופיע באיור הבא[1] המתמצת את תכנית ונושאי הלימוד הקיימים היום במרב בתי הספר האקדמיים:



אחת הטענות אותן אנו שומעים יותר ויותר היא העובדה שמרב אנשי הסגל והמורים המלמדים הנדסת תוכנה מוצאם ממדעי המחשב. עדיין לא הצטברה מסה קריטית של אנשי סגל שקיבלו את התואר בהנדסת תוכנה.
ישנם רק שלושה מוסדות מוצהרים בעולם המעניקים תואר דוקטורט שההתמחות שלו הינה הנדסת תוכנה (אף לא אחד בישראל)
מסקנות המחקר ממנו הבאתי את התרשים* , הינן מאוד מובהקות – ישנו פער אדיר בין מה שלימדו את מהנדסי התוכנה בבית הספר לבין תחום עיסוקם בפועל. מחקר מעמיק זה מעלה נושאים רבים לעניין מהות התואר ותכניות הלימוד וכדאי לקרוא אותו, כדאי לטעמי לנסות להסביר מדוע קיים פער זה ומה ניתן לעשות על מנת להכין ולהכשיר את המהנדסים טוב יותר לתפקידיהם בתעשייה. כאמור יתכן שחלק מההסבר לפער זה נובע מהבלבול וחוסר הייחוד בין מקצוע הנדסת התוכנה לבין מה שאנו מכנים "מדעי המחשב" . בעיני פער בולט ביותר הוא חוסר הכשרה מתאימה למהנדסים באותן המקצועות הרכים – במחקר מעידים המהנדסים שעיקר הכשרתם כמפתחים ומהנדסי פיתוח ובדיקות נובעת מהליכי למידה עצמית, במידה רבה יותר מאשר החינוך הפורמלי אותו עברו. על פי מחקר זה, מרב מוסדות ההשכלה המכשירים מהנדסי תוכנה עדיין לכודים בתדמית המסורתית של המקצוע מתוך מקורם (מדעי המחשב) וממשיכים להכשיר את המהנדסים ללא קשב אמתי לצרכי השוק וכך פוגעים בכל התעשייה. עלינו להקנות ידע ויכולות מקצועיות עדכניות למהנדס הצעיר – מתקיים כאן מרוץ מטורף בין מציאות המשתנה בקצב בלתי אפשרי ומערכת הכשרה שבמקרה הטוב מפגרת מספר צעדים ובמקרה הרע קופאת על שמריה. הרלוונטיות של חומרי הלימוד משתנה כל יום. לפני 5 שנים לא ידעו תלמידי את פרוש המונח agile ואילו היום מי שאינו בקיא בשיטות פיתוח אג'יליות אינו מגיע מוכן דיו למקצוע. באופן זה אני יכול להצביע על תחומי הכשרה וידע אחרים שמלווים אותנו ומשתנים השכם והערב. ניקח לדוגמה: השנוי בממשקי המשתמש (השתלטות המובייל על עולמנו), שימוש בענן, ארכיטקטורת השירותים או הפלטפורמות עליהם מממשים את האפליקציות השונות. כמו כן ניתן לראות בברור את אי החשיבות (לפעמים עד כדי זלזול) הניתנת לנושאי הבדיקות ואבטחת האיכות. בוודאי מוסדות הלימוד האקדמאים בארה"ב המספקים יותר מקורס בסיסי אחד

1 **Self-Perceptions about Software Engineering**:
A Survey of Scientists and Engineers 2012 the University of Alabama
http://software.eng.ua.edu/reports/SERG-2011-04

בדיקות תוכנה באקדמיה – מצב האומה

דני אלמוג

של בדיקות תוכנה. יוצא מכך שמרב מהנדסי התוכנה אינם יודעים כיצד יש לבדוק את התוכנה אותה הם מפתחים. מהנדס תוכנה שלא למד את עקרונות בדיקת התוכנה (בייחוד ברמת בדיקת היחידה) מהווה בעיה אמתית לכל התעשייה ומעכב את ההתקדמות והגמישות הנדרשת היום בתהליכי הפיתוח של תוכנה.

> **התאמת תכניות הלימוד לצרכים המעשיים של מהנדס התוכנה בשטח.**

איפה משתלב בודק התוכנה בעולם זה

הנדסת התוכנה והבדיקות משולבים ללא יכולת הפרדה ברורה ביניהם: מחד אנו מצפים ממהנדס הפיתוח של היום שיבדוק היטב את עבודתו, המעבר לשיטות אג'יליות למעשה מכתיב התייחסות יותר מחייבת של מהנדס הפיתוח לתהליך בדיקת התוכנה שלו. מאידך חשוב לנו מאוד שמהנדס הבדיקות יהיה מסוגל להבין ולהתמודד עם הקוד ומשמעויותיו. אופן כתיבת הקוד משפיע בצורה ישירה על איכות התוכנה ועל יכולת הבדיקה שלה. מרכיב מאוד חשוב הוא אופי העובד. לאחרונה אנו מבחינים בדרישה גוברת והולכת לכישורים של עבודת צוות ויכולת שיתוף פעולה. בעברי הלא רחוק נדרשתי לגיוס נרחב של מהנדסים לביצוע בדיקות ואבטחת איכות של התוכנה – מצאתי עצמי מתלבט היכן ואיך לזהות את אותם המתאימים לתפקיד הקשה הזה. התלבטות זאת נובעת מכך שלמעשה לא מצאתי מוסד אקדמי מכובד "המייצר" מהנדסים אשר זאת הכשרתם והכוונתם. אין לי ספק באשר לרמת ההשכלה הנדרשת על מנת לעסוק בתחום (מהנדס), בתפקידי כמגייס נאלצתי לדלג בין מקצועות הנדסת התוכנה, הנדסת מערכות מידע והנדסת תעשיה וניהול (במגמת מערכות מידע) על מנת לבחור מועמדים לעסוק בתהליכי האיכות של התוכנה. די ברור לי היום שבוגרי מדעי המחשב הם מתאימים פחות לתפקידים אלו אולי משום האופי המדעי של הכשרתם – במידה רבה חוסר הדגש על הפרקטיקה של הביצוע והתוצאות – היו בעוכרי אותם מועמדים אשר נבחנו לתפקידים אלו.

עוד כמה השגות ואמיתות אותם אני לומד להפנים:

אישית, מאד מטריד אותי שאיננו מכלילים ומפנימים דיינו את הצורך והחובה לייצר תוצר איכותי. תהליכי האיכות והבדיקה בעולם התוכנה זוכים לגימוד ואי לגיטימיות (בעיני הסטודנטים) בגלל היסטוריה של תדמית וראיה לא נכונה ואמתית של מה באמת חשוב בתהליך ייצור התוכנה. אני מצפה שאת נושאי האיכות יתחילו ללמד כבר משנה א'. למשל איני מבין מדוע לא מלמדים כיצד לבדוק את היחידה אותה פיתחת (Unit Test) כחלק מתהליך לימוד שפת התוכנה הראשונה שלך. בכלל ישנם נושאים שהיו לטעמי חייבים להיות כחלק מההשכלה הבסיסית של מקצוע התכנות (כדוגמת איך לבדוק את הקוד). חשוב גם לזכור שכשליש מחיי כל מפתח תוכנה היא הבדיקה. השמירה על האיכות ובדיקת התוכנה הינה ההשקעה הגדולה ביותר במחזור הפיתוח של תוכנה – הרבה יותר מהפיתוח עצמו (ביל גייטס 2003). באם כותרת המקצוע שלנו זה הנדסת תוכנה הרי התחום הזה הוא בהחלט חלק חשוב מחיינו המקצועיים.

עבודה מעניינת ורלוונטית לשילוב ההתמחות בבדיקה תוך כדי לימוד שפות התוכנה ובכלל כחלק מתהליך לימוד הנדסת תוכנה נעשית בהובלת האוניברסיטה הבינלאומית פלורידה[1] פעילות מאורגנת בראשות Dr. Peter J Clarke מציגה כיצד מאמנים סטודנטים להנדסת תוכנה בשימוש בכלי בדיקה תוך כדי תהליכי הלימוד של השפה עצמה, יוזמה זאת מלווה בהקמת תשתיות ברשת להכשרה ולימוד: "WReSTT (Web-Based Repository of Software Testing Tools)". הטבלה הבאה מדגימה חלק מהכלים הנגישים לסטודנטים:

Name	Classification	Language
Cobertura	Coverage	Java
CppUnit	Test Execution, Unit	C++
EclEmma	Coverage, Plug-in, Java	Java
JDepend	Metrics	Java
JUnit	Plug-in, Test Execution, Unit	Java
SWAT	Test Execution, Web, System/UI	
Rational Functional Tester	Test Execution, Web, System/UI	Java, VB .NET

למעוניינים אני ממליץ להיכנס לאתרים שלהם ולבחון את האפשרויות המתפתחות.

> **עלינו להקנות ידע ויכולות מקצועיות עדכניות למהנדס הצעיר**

מה קורה בארץ

בארצנו הקטנה הדרישה למהנדסי בדיקות רחבה מאוד (מעל ל1000 עובדים חדשים כל שנה). נחלק את תהליכי ההכשרה של בודקים לשניים: מספר מוסדות (מכללות) לא אקדמיים המכשירים בצורה סדורה כל שנה מאות בוגרים (ממקורות שונים לא רק עם רקע של תוכנה) במרחב מקצועות והשתלמויות מקצועיות הקשורות בענף. ראוי לציין שמאז הקמת ISTQB® גדלה חלקה של ההכשרה הפורמלית במקצוע. חלק לא מועט מההכשרות הללו גם מתבצע בתהליכים פנים ארגוניים על ידי החברות המגייסות. גולת הכותרת של ענף הבדיקות היא מיכון בדיקות – test automation שתופס יותר ויותר משקל ויוקרה. צורות הפיתוח האג'יליות מכתיבות שילוב של מומחי אוטומציה כחלק מקבוצת העבודה הבסיסית ופיתוח התוצרים הללו כחלק בלתי נפרד מהנדרש בסוף כל מאוץ אג'ילי שכזה. מדינת ישראל מהווה ללא צל של ספק אבן שואבת ומרכז עולמי לפיתוח כלים וליישום שיטות עבודה ומימוש של אוטומציה. עלינו לטפח

1 Integrating Testing into Software Engineering Courses Supported by a Collaborative Learning Environment PETER J. CLARKE at al. ACM Transactions on Computing Education, Vol. 14, No. 3, Article 18, 2014

בדיקות תוכנה באקדמיה – מצב האומה

דני אלמוג

איכות ובדיקות במוסדות בארץ

המוסד האקדמי	קיום קורס	כותרות הקורסים	משקל אקדמי	הערות
אורט ברואדה	יש	אימות ובדיקות	3.5	
אוניברסיטה העברית	יש	בדיקות תוכנה	2	
אוניברסיטה פתוחה	אין			
אוניברסיטת בן גוריון	יש	הנדסת איכות בתוכנה, אימת ובדיקות, נושאים מתקדמים באיכות תוכנה	3.5 , 3	חובה + בחירה
אריאל	אין			
מכללת אפקה	אין			
JCE מכללת עזריאלי	יש	בדיקות תוכנה	3	בחירה
SCE מכללת סמי שמעון	יש	אימות, אבטחת איכות תוכנה	3.5 + 3	חובה
מכללת חולון	אין			
טכניון	יש	אימות , אבטחת איכות תוכנה	3.5 + 3	חובה
מרכז לב	אין			
מכללת כנרת	יש	אימות , אבטחת איכות תוכנה	3.5 + 3	
מכללת עמק יזרעאל	יש			
מכללת נתניה	אין			
מכללת ספיר	אין			
מכללת רופין	אין			
מכללת קריית אונו	יהיה			
מכללת תל-חי	אין			
המרכז הבין תחומי	אין			
אוניברסיטת בר אילן	חלקי	אימות	2	בחירה, חובה לתואר 3

ישנה דרישה גוברת והולכת לכישורים של עבודת צוות ויכולת שיתוף פעולה.

ולעודד את הכשרת המומחים בנושא. המשקל בארץ של גופי הסמכה למקצועות הבדיקה הוא רב. בארגון ITCB יש ייצוג מכובד לנציגי התעשייה ונראה שיש גב וכך נרחב ליוזמה ופעילות בנושא. ניתן לומר שעצם הקמת ארגון ISTQB® ודומיו עודדה ותרמה להכללת לימודי בדיקות כחלק מסדר היום האקדמי. חל שינוי מהותי – אם כי לא בקצב מספק. יותר ויותר מוסדות מכלילים את יכולת בדיקת בתוכנה כחלק בלתי נפרד מתהליך הכשרת מהנדס התוכנה.

ישנם בארץ מוסדות (BGU , SCE) המתייחסים להכשרה זאת כחלק מחייב מתהליך התואר להלן טבלה המפרטת את הקורסים במוסדות השונים (מקור- מפרסומי המוסדות). אני שמח לציין שהיום מלמדים בדיקות ואיכות תוכנה (לתואר ראשון) ב 9 מוסדות לימוד אקדמיים טכנולוגים נוספים. בחלקם אף שוקלים הענקת תעודות הסמכה כגון זו של ISTQB® כלימודי חוץ או הכשרה נוספת לסטודנטים השונים. לסבר את האוזן – בהודו הגדולה אין ולו מוסד אחד בו לימודים אלו הינם חובה ויש בערך אותו מספר של מוסדות אקדמיים שמכריזים שהם מציעים לבוגריהם לימודי בדיקות תוכנה כאשר השוק ההודי קולט יותר מ 10,000 מהנדסי בדיקות כל שנה. אל לנו לשכוח שחלק ניכר ממהנדסי הבדיקות בארץ מגיעים גם כמהנדסים שאינם מהנדסי תוכנה – בעיקר תעשיה וניהול ומערכות מידע.

עדיין לא ראינו הנחיה מפורשת של המל"ג שיש להכליל מקצוע זה כחובה לשם קבלת תואר בהנדסת תוכנה. עלינו כתעשייה ללחוץ על המל"ג להכליל את זה בתנאיהם לאישור להענקת תואר (וכמובן מימון). אסור לוותר על אחריות התעשייה לנעשה באקדמיה – בואו נעזור לאקדמיה לקבל את ההחלטות המתאימות באמצעות עידוד מחקרים (הצעות למחקר והתחייבות לממן אותו) והצבת דרישות מפורשות מהבוגרים הצעירים לגבי ידע נדרש. חשוב גם לשתף את הידע האדיר שנצבר בתעשייה עם האקדמיה וזאת באמצעות הצבת מומחים שיתמכו בתהליכי הלימוד ושיתוף מידע אפשרי לצורך אתחול מחקרים.

הדרישה למהנדסי בדיקות הינה מעל ל-1000 עובדים חדשים בכל שנה.

As a Test Engineer, to what extent do you think the ISTQB® CTFL certification will affect/improve your future?

מרבית המשיבים (82.7%) מרגישים כי ההסמכה הבסיסית (ISTQB® CTFL) משפרת את יכולות הבדיקה שלהם, מחזקת את סיכוייהם (80.1%) ומשפרת ההכרה לה הם זוכים מחבריהם לעבודה (61.5%). כמעט מחצית מהמשיבים חשים כי זה עשוי להגדיל את משכורתם. התוצאות משקפות את החשיבות שאנשים רואים בשיפור מתמיד של יכולותיהם.

Source: ISTQB® Effectiveness Survey Report

סקרים

בודק – הדרך אחרים בבדיקות

בודק "הדרך אחרים בבדיקות" ההבנה וההטמעה הטובה ביותר של כל ידע, מתבצעת בעת שאנו נאלצים להדריך אחרים בידע זה. אז עלינו לחשוב כיצד לעבד את המידע שבראשנו, לכדי הסבר שיהיה נהיר וברור גם לאחרים, עלינו להתמודד עם שאלות שעולות וכן הלאה.

ישנן הזדמנויות ודרכים רבות להדריך אחרים בבדיקות:

1. הדריכו חברים חדשים לעבודה, איננכם חייבים להיות וותיקים בכדי להשתתף בפעילות זו – להפך, לעתים קרובות דווקא המצטרפים הקודמים יכולים להרוויח המון אם יחנכו את המצטרפים אחריהם – זו הזדמנות פז לחדד את הידע ולוודא כי הנכם מכירים אותו כראוי. זכרו – כאשר אתם מדריכים – אסור לכם "לגעת במקלדת" – השאירו את התרגול המעשי למודרכים, תפקידכם מתמקד בהנחייתם ובמענה לשאלותיהם.
2. הדריכו אחרים בעזרת דיונים בפורומים השונים, חלקו ידע שלכם ובמקביל הרוויחו ידע ודעות של אחרים.
3. כתבו פוסטים בבלוג, ובהם חלקו התנסויותיכם והמסקנות אליהם הגעתם – זכרו איננכם חייבים להחזיק בלוג משלכם – ישנם בלוגים משותפים אליהם מגיעים אנשים רבים יותר, שם תקבלו חשיפה גבוהה. עצם המחשבה על המידע והכנתו לכדי חומר כתוב – מספקת לכם אמצעי מעולה להתמודדות ולהפנמת הידע.
4. הרצו במפגשים ובכנסים – כאן נדרשת קצת יותר הכנה מוקדמת והתמודדות עם פחד הקהל, כיום יש מגוון רב של מפגשים וכנסי בדיקות בארץ, ובכולם מחפשים מרצים מן הקהילה שיחלקו מניסיונם.
5. בקבוצות בדיקה רבות נהוג שהחברים חוקרים נושא ומרצים בפני חברי הקבוצה במפגש שבועי/חודשי – זה יכול להיות על מתודולוגיית בדיקות חדשה, על נושא מתחום העיסוק הטכני, ואף על חוויות מסבב הבדיקות האחרון.
6. ישנן קבוצות וצוותי בדיקה בהן אחת לשבוע דנים בנושא מקצועי במהלך ארוחת הצהריים, אחד מחברי הצוות מכין הצגה קצרה של הנושא, ולאחר מכן דנים בו.
7. לעתים קרובות לצוותי התוכנה חסרה הבנה בבדיקות – ושוב, זו הזדמנות נהדרת לקרב לבבות, ללמד ובכך להרוויח יותר מודעות לנושא אצל המפתחים, תוך כדי התמודדות עם השאלות העולות ומתוך כך לימוד עצמי.

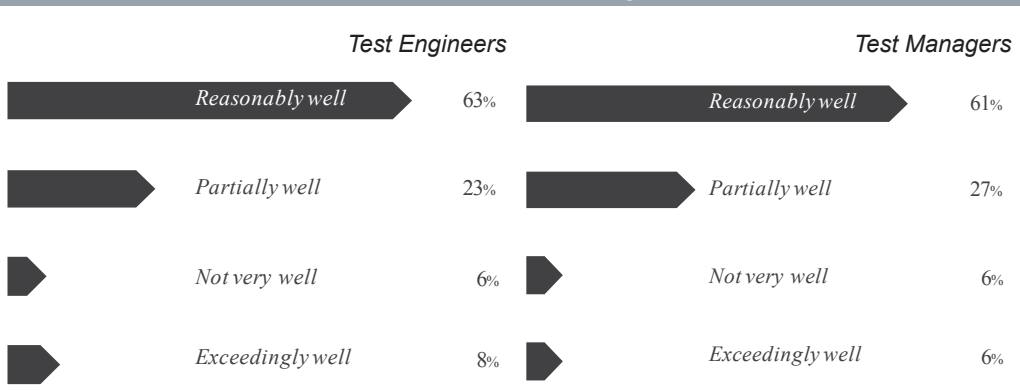
חומר קריאה נוסף:

http://www.mkltesthead.com/2013/08/teach-testing-99-ways-workshop-26.html

סדרת טיפים זו "כיצד להפוך לבודקים טובים יותר" מתבססת על דיון ב: Software Testing Club
99 Things Testers Can Do To Become Better Testers
ה-eBook החינמי שנוצר בעקבות דיון זה: 99ThingsEbook.pdf
וסדרת פוסטים מאת Michael Larsen בשם: 99Ways Workshop – בה מיכאל מרחיב על כל אייטם וגם מספק הנחיות כיצד לתרגל הנושא.

אתם מוזמנים לקרוא טיפים רבים נוספים ב: http://goo.gl/UcW42
ולהוסיף טיפים משלכם לרשימה משותפת זו.

To what extent do you feel the ISTQB® Foundation Level certification exam reflects the knowledge of the candidates?



מרבית המשיבים מתייחסים לבחינת ההסמכה הבסיסית (ISTQB® CTFL) כאמינה ותוצאות אלו אחידות ברחבי העולם.

Source: ISTQB® Effectiveness Survey Report

סקרים

QA Dashboard

עמיר ישראלי

נשוי + 3, גר בכפר סבא ומשרת פעיל במילואים כלוחם בצנחנים.
בוגר תואר ראשון בניהול מהאו"פ ומוסמך ISTQB®.
מנהל את קבוצת הבדיקות של מוצרי הפלטפורמה בנייס אקטימייז.
התחיל בתחום הבדיקות ב-2006 כבודק תוכנה בחברת PNMSOFT ובנייס אקטימייז.

"לא ניתן לשפר את מה שלא מודדים" זה מה ששמעתי מיגאל לוי, דירקטור הבדיקות של חטיבת אקטמייז בנייס, כאשר הצטרף לארגון. אז התחלנו למדוד "**Escaped Defects**" כלומר באגים שלא נמצאו בתהליך הבדיקות הפנימי אלא נמצאו אצל הלקוח. בהתחלה עשינו את זה באופן ידני בסוף השנה, זה היה תהליך ארוך וקשה שדרש מאמץ רב של ראשי הצוותים, לנתח כמות גדולה של באגים באופן מרוכז.

כחלק מתהליך השיפור של קבוצת הבדיקות בארגון הקמנו קבוצת עבודה שנועדה לשפר את איכות המוצרים שלנו ע"י שיפור תהליך המדידה של ה"Escaped Defects". אחרי הגדרת דרישות ואפיון בחרנו להשתמש במספר מוצרים של אקטמייז כדי ליישם את הפתרון שהצענו.

יצרנו תהליך אוטומטי שלוקח את הבאגים הרלוונטיים ממערכת דיווח הבאגים ומעתיק אותם למערכת שלנו. גיבשנו תהליך עבודה לניתוח ה"Escaped Defects" שבו כל צוות בישיבה שבועית עובר על כל הבאגים שנכנסו למוצר שלו, מנתח את הסיבה שבגללה הבאג לא נמצא במהלך הבדיקות, מעדכן אם זו בעיית איכות או דרישות וקובע לאיזו קטגוריה הבאג שייך. בהמשך התהליך ראש הצוות משייך את הפעילות המתקנת לאחד הבודקים בהתאם למה שהוחלט בישיבה.

המערכת מלווה את התהליך ב Workflow סגור, משלב קליטת התקלה ועד סגירתה ע"י ראש הצוות אחרי שווידא שהפעולה המתקנת הנדרשת הושלמה.

הדבר היפה שהשגנו מיצירת המערכת, הוא שיש לנו באופן שותף וללא מאמץ נוסף, דוחות בזמן אמת לגבי מצבו של כל מוצר מבחינת הבאגים שפספסנו בחתכים שונים כמו מגמה לאורך זמן, גרסת המוצר, חומרת הבעיה, הלקוח שדיווח ועוד מדדים נוספים. העובדה שהנתונים זמינים בלחיצת כפתור נותנת לנו יכולת תגובה מהירה במקרה של טיפול בתלונות הלקוחות.

אין ספק שתהליך העבודה שיצרנו גרם לשינוי תודעה בקרב הבודקים אצלנו בארגון, הצוותים מכירים את התקלות השונות עד אחרון הבודקים, חושבים ביחד על פתרונות איך למנוע מתקלות דומות לחזור ומיישמים לקחים באופן מידי על הרכיבים שנמצאים כרגע בתהליך הבדיקה. יתרון נוסף שהרווחנו נובע מהשימוש במוצרי אקטימייז, למעשה הרווחנו קבוצה שלמה של בודקים

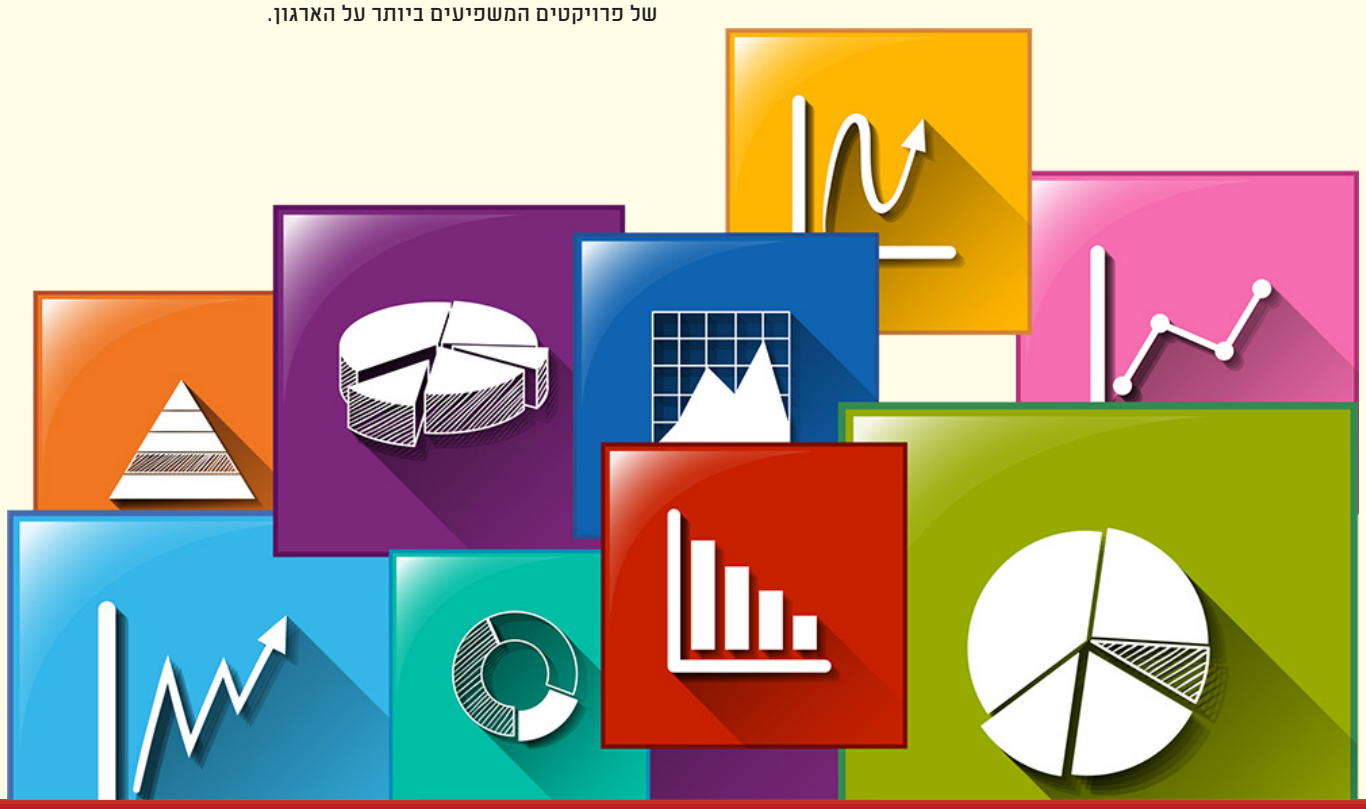
> **תהליך העבודה שיצרנו גרם לשינוי תודעה בקרב הבודקים אצלנו בארגון.**

> **הזכייה בפרס המצוינות נתנה חותמת רשמית למה שכבר הרגשנו.**

שעובדים באופן שוטף על מוצרי החברה, מצד אחד מוצאים תקלות אמת במוצר (אנחנו משדרגים את גרסת המוצר לפני השחרור הרשמי) ומצד שני הבודקים, בפעם הראשונה מקבלים נקודת מבט של משתמשי קצה במערכת מה שפותח את עיניהם לחוויית ממשק המשתמש.

באופן ברור אנחנו רואים אצלנו בארגון, שמאז שהתחלנו ליישם את התהליך חלה ירידה שנה אחרי שנה של כ 36% במספר הבאגים החוזרים מלקוחות. מדד שביעות הרצון של הלקוחות שנמדד שנה אחר שנה מצביע על שיפור באיכות המוצרים של החברה כמו שהם נתפסים בעיני לקוחותינו. ללא ספק אנו רואים שמידת המעורבות והמחויבות של צוות הבודקים שלנו עולה והארגון מרוויח גם בודקים יותר טובים וגם מוצרים יותר איכותיים.

הזכייה בפרס הראשון במצוינות בבדיקות בקטגוריה התהליכית של ארגון ITCB נתנה חותמת רשמית למה שכבר הרגשנו בעצמנו. הפרויקט אכן מהווה דוגמא כיצד ניתן להפגין תרבות של שיפור עצמי תוך הפגנת יוזמה ומקוריות בשיפור תהליך הבדיקות. כמו כן הפרויקט זכה במקום הראשון בנייס בתחרות פנימית של פרויקטים המשפיעים ביותר על הארגון.



Heuristic Evaluation

מונח קצת מוזר וקשה להגייה, אבל בעל משמעות אדירה בחיי היום-יום שלנו, הבודקים, בבואנו לבצע בדיקות שמישות (Usability) ונוחות משתמש (Ease of Use).
בניגוד לבדיקות פונקציונליות בהן ככל שלבודק יש היכרות טובה יותר עם המערכת או האפליקציה הנבדקת, בבדיקות שמישות היכרות עם המערכת מהווה חיסרון. קשה לנו להגדיר, לדוגמא, האם המערכת נוחה לשימוש למשתמש חדש. שימוש בטכניקה זו יאפשר לנו להתגבר על מגבלת "עודף הידע" ולעזור לארגון להוציא מערכות נוחות יותר למשתמש.

ההגדרה
היוריסטיקה (Heuristic) היא כלל חשיבה פשוט, מעין כלל אצבע המבוסס על הגיון פשוט או אינטואיציה, המציע דרך קלה ומהירה לקבלת החלטות ופתרון בעיות, ללא התעמקות ובמחיר דיוק נמוך.
חוקריהם החשובים של היוריסטיקות היו הפסיכולוגים הישראלים דניאל כהנמן ועמוס טברסקי, שהדגימו כיצד המח האנושי נוהג להשתמש בטכניקות היוריסטיות שונות, על מנת לקבל החלטות במהירות.
סקירה היוריסטית הינה טכניקה לסקירת שימושיות (Usability Review) המכוונת לאיתור בעיות שמישות בממשק המשתמש של אפליקציה או מערכת. הסוקרים בודקים את הממשק באמצעות טכניקה זו וקובעים את מידת התאמתה לעקרונות שימושיות מקובלים (ה"היוריסטיקה")[1].

מה זה שימושיות (Usability)?
שימושיות היא נוחות המשתמש ויכולת הלימוד העצמית של אובייקט או מערכת.
בעולם מרובה האפליקציות של היום, לשימושיות של מערכת השפעה אדירה על תפיסת האיכות של המערכת (Quality Perception). משתמשים רבים מחליטים תוך שניות בודדות האם להמשיך לגלוש באתר או באפליקציה ורמת השימושיות של האפליקציה תקבע במקרים רבים את מידת הצלחתה.
מכיוון שלשימושיות השפעה מהותית על האיכות, תפקיד הבדיקות הינו לוודא כי המערכת נוחה לשימוש עבור המשתמש הסופי.

> **שימושיות היא נוחות המשתמש ויכולת הלימוד העצמית של אובייקט או מערכת.**

איך מוודאים שימושיות?
השיטה המקובלת ביותר היא באמצעות סקירת שימושיות. באמצעות שיטה זו, מספר מצומצם של סוקרים (Evaluators) מבצעים בדיקה של ממשק המשתמש של המערכת ומחליטים האם המערכת עומדת בעקרונות שמישות מקובלים.
שיטה זו נועדה לכפר על העיוורון הבלתי רצוי (In attentional blindness). זו תופעה שבה בודק תוכנה, אשר לו כבר היכרות מעמיקה עם המערכת אותה הוא בודק, לא יכול למעשה "לראות" בעיות אשר יקשו על המשתמש לעבוד עם המערכת. הוא כבר מכיר את המערכת, את התהליכים ולכן שימוש בפונקציונליות מסוימת, לדוגמא, תיראה לו קלה מאוד בעוד שלמשתמש חדש אותה פונקציונליות תיראה קשה להבנה.
ביצוע סקירת שימושיות מבוצע באמצעות הכנה מראש של רשימה של עקרונות שימושיות (Usability Heuristics). לאחר מכן יש לעבור על מסכי המערכת ופונקציות עיקריות ולוודא שהמערכת עומדת באותו עיקרון.

> **בבדיקות שמישות היכרות עם המערכת מהווה חיסרון.**

דוגמאות לעקרונות השימושיות (Usability Heuristics)
קיימים מספר מודלים לעקרונות שימושיות. המוביל ביניהם הינו המודל של יעקב נילסן (Jacob Nielsen) כפי שפורסם בספרו "הנדסת שימוש"[2].
להלן מספר דוגמאות:

שיקוף סטטוס המערכת
המערכת אמורה לעדכן את המשתמש בכל נקודת זמן לגבי מה קורה, באמצעות פידבק רלוונטי ובזמני תגובה סבירים.

התאמת המערכת לעולם האמיתי
המערכת אמורה לדבר את שפת המשתמש באמצעות מילים, משפטים ותפיסות שהמשתמש מכיר ולא באמצעות מושגים של המערכת.

שליטה וחופש של המשתמש
משתמשים לעיתים יעשו שימוש בפונקציות מסוימות בטעות ויצטרכו "יציאת חירום" מבלי לעבור דרך דיאלוג ארוך. יש לעשות שימוש ככל האפשר ב-Undo ו-Redo.

אחידות (Consistency) וסטנדרט
משתמשים לא אמורים לשאול את עצמם האם למילים, סיטואציות או פעולות יש משמעות דומה.

מניעת תקלות
ההבדל בין חכם ופיקח... המערכת אמורה למנוע מהמשתמש מלהגיע לשגיאות במקום להתמקד בהודעות שגיאה יפות. הימנע מתנאים אשר יכולים להביא לשגיאה או הצג למשתמש הודעה לפני שהוא מבצע פעולה שעלולה להוביל אותו לשגיאה.

זיהוי ולא זיכרון
מנע מהמשתמש שימוש בזיכרון באמצעות יצירת אובייקטים, פעולות ואופציות שקופים. המשתמש לא אמור לזכור אינפורמציה משלב אחד בדיאלוג לשני. הוראות הפעלה אמורות להיות מוצגות ונגישות בכל מקום בו הם נדרשות.

עיצוב מינימלי
מנע פונקציות או מידע אשר לא רלוונטי או נעשה בו שימוש לעתים רחוקות.

נגישות (Accessibility)
האם המערכת מאפשרת גישה לבעלי מוגבלויות (לדוגמא: עיוורי צבעים)?

תרגול
על מנת לתרגל יישום של גישת Heuristic Evaluation מומלץ לקחת אחד מהעקרונות לעיל ולהבין אותו לעומק. לאחר מכן, גש למערכת אותה את/ה בודקים (או כל מערכת אחרת), בצע מעבר בין המסכים השונים ובדוק האם העיקרון מיושם בכלל המסכים והפונקציות של המערכת.

> **ביצוע סקירת שימושיות מבוצע באמצעות הכנה מראש של רשימה של עקרונות שימושיות**

1 ITCB-ISTQB_Gloss_2.2-MASTER-ENG-HEB-v0.80

2 Nielsen, Jakob (1994). Usability Engineering. San Diego: Academic Press. pp. 115–148. ISBN 0-12-518406-9.
http://www.nngroup.com/articles/ten-usability-heuristics/

עזבו אתכם מיציבות בעבודה, ביטוח קריירה כבר יש לכם?! | מאיה בוכניק

תגידו, מה הכי חשוב לכם כשאתם מחפשים עבודה? מהו המאפיין שהכי משפיע עליכם לבחור במקום עבודה מסוים ולא באחר?
לכל אחד מאיתנו חשובים דברים אחרים, אבל אחד הפרמטרים המשותפים לרבים הוא פרמטר היציבות בעבודה. "אני מחפש מקום יציב!", אומר לי מחפש עבודה, "כזה שאוכל לעבוד בו לטווח ארוך ולא לדאוג לענייני פרנסה כשאגיע לגילאים הבעייתיים של +45". אני שומעת את זה כל הזמן... רבים מחפשים חברות יציבות ובטוחות.

האם "יציבות בעבודה" בכלל קיימת?
בואו ונחשוב על זה לרגע ברצינות: האם יש דבר כזה בכלל? חברה שהיא בוודאות יציבה ובטוחה? כזו שבטוח, בטוח, בטוח (!) שנוכל לעבוד שם עד לפנסיה אם רק נרצה? כזו שאין בה שום סיכוי לרה-ארגון? לגל של קיצוצים? לירידה בביקושים? או חלילה לפשיטת רגל?
אם נהיה כנים עם עצמנו, נבין שאנחנו מחפשים חד-קרן או נכון יותר-דינוזאור... יצור שכבר מזמן לא קיים. אין תחום, ענף, חברה, מקצוע או תפקיד שעשוי להבטיח לנו ב 100% יציבות או ביטחון.
אז נכון... יש חברות שהן מסוכנות יותר מאחרות ויש ארגונים שהם מבוססים יותר מאחרים, גדולים יותר ולכן אולי בטוחים יותר, אבל הכל יחסי ופיטורים קורים גם בארגונים כאלה למרבה ההפתעה והצער.
למעשה, יציבות בעבודה זה מושג מאוד אמורפי... אתם יכולים לבחור לעצמכם תפקיד בטוח במוסד או ארגון ישנוני, שהריגוש הכי גדול בו במהלך היום קורה כשאתם מכינים תה וצריכים לבחור באיזה טעם, ולהיות בטוחים ששיחקתם אותה ושמצאתם לעצמכם מקום עבודה עד לגיל 65 ואז...? אז מגיע מנהל חדש, אחד ששם לכם רגל, כזה שאין ביניכם שום כימיה ושלאט לאט, אבל בטוח, דוחף אתכם החוצה לכיוון הדלת.
האם יכולתם לנבא מקרה כזה? ממש לא. האם זה עלול לקרות? ממש כן... מנהלים ישירים הם אחת הסיבות המובילות לעזיבת עבודה (מבחירה או שלא...).

זהירות, בקרוב אתם עלולים לגלות שלאף מעסיק אין צורך בניסיון שלכם
לפני מספר חודשים פורסמה בדה מרקר כתבה ושמה "זהירות: 900 אלף ישראלים עומדים לגלות שאין צורך במקצוע שלהם".
הנה מספר ציטוטים חשובים מתוך הכתבה:
"900 אלף ישראלים עומדים לאבד בעשורים הקרובים את העבודה שלהם. לא את מקום העבודה שלהם, אלא את העבודה שלהם ממש. מדובר ב-30% מהעובדים במדינת ישראל שבקרוב ימצאו את עצמם נזרקים משוק העבודה, כשאף מקום עבודה אחר אינו רוצה יותר להעסיק אותם. הסיבה: אבד הכלח על המקצוע שהם עסקו בו כל חייהם, ולכן לאיש אין יותר צורך בהם".
"המשרות שנפגעות הן בעיקר המשרות של מעמד הביניים... העובדים החלשים, עובדי הכפיים, מוגנים יחסית, כי אין מי שיחליף את מטאטא הרחובות או השומר בכניסה לבית הספר. העובדים החזקים מאוד מוגנים גם הם, מכיוון שהם נמצאים במשרות ניהוליות ויצירתיות, משרות המחייבות אינטליגנציה רגשית ותגובה לסביבה משתנה, ואלה הן תכונות שמכונות אינן יכולות להחליף אותן. מי שפגיע הם כל מי שמבצעים עבודות שגרתיות מסוגים שונים - ההגדרה היא עבודות שהן ידניות, חזרתיות (רוטיניות) ומבוססות נהלים. למשל סוכני ביטוח, קופאים, מתווכים, מוקדנים, עובדי מכירות פשוטים או פקידים."

> **אל תישארו עם הראש בחול- לכו ללמוד**

"ביטוח קריירה" כתחליף ליציבות בעבודה
אם הגעתם עד לכאן, אתם בטח תוהים לעצמכם "אם אי אפשר לבנות על יציבות במקום העבודה ואם העתיד התעסוקתי אף הוא איננו ברור, אז מה עושים?"
הדבר הנכון ביותר הוא לדאוג לעצמכם לביטוח קריירה: במקום להתלבט "איפה עדיף לעבוד" צריך להתעמק בשאלה "במה לעבוד?" ובמקום לבנות על גורמים חיצוניים, מתחילים לסמוך רק על עצמכם, אבל עושים זאת בחכמה.
"ביטוח קריירה" יעניק לכם ביטחון במקצועיות שלכם וביניסיון שלכם ללא קשר למקום העבודה שאתם עובדים בו באותה נקודת זמן. ממש לא ישנה לכם אם החברה שבה אתם עובדים נקנתה עכשיו על ידי תאגיד ענק, כי בכל מקרה שלא יבוא- גם אם תצטרכו לחפש עבודה, אתם תהיו בטוחים שתצליחו למצוא כזו ובזריזות! במקום להתפשר ולהישאר במקום עבודה כלשהו במשך שנים רק מהפחד לאבד את היציבות ותוך חשש מתמיד מפיטורים, תוכלו לבחור תפקיד על פי האתגר שבו ועל פי העניין ותוכלו גם לעזוב מרצונכם אם התפקיד לא יענה על המטרות שהצבתם לעצמכם. במקום לבחור בתפקיד במקום עבודה משעמם (אך "בטוח") תוכלו להעז לבחור בתפקיד החלומי ההוא בסטארט אפ המגניב- דבר שתמיד חלמתם לעשות. ולמה? כי טרחתם ובניתם לעצמכם מזרני ביטחון מכל הסוגים האפשריים.
אני יודעת שלחלקכם השורות הקודמות נשמעות כמו אוטופיה, אבל זה אפשרי להשגה (גם אם לא פשוט)! הנה מספר צעדים שאתם חייבים לבצע כדי ליצור לעצמכם ביטוח קריירה:
1. אף פעם אל תפסיקו ללמוד ולהרחיב ידע. ידע הוא כוח, הוא משאב. השקיעו ברכישת כישורים רוחביים, כאלה שעשויים להיות לכם לנכס בכל תפקיד ובכל מקום עבודה.
2. לא יודעים מה ללמוד? באיזה כיוון להשקיע? איזה כישורים כדאי לחדד? הכירו את שוק העבודה יותר לעומק! נתקעתם שנים בפיתוח קוד בשפה שהולכת ונעלמת מהעולם? אל תישארו עם הראש בחול- לכו ללמוד שפת פיתוח חדשה ויפה שעה אחת קודם. ובמילים אחרות: אל תגיעו מלכתחילה למצב שבו אתם עוסקים שנים באותו העיסוק בדיוק ואפילו לא מודעים לכך שבשוק הביקוש לזה יורד בהתמדה. אתם חייבים להיות עם היד על הדופק כל הזמן ולזהות שינויי מגמות ברגע שהן מתהווים.

ניקח לדוגמא (איך לא?!) את תחום הבדיקות. הנה 3 טרנדים עכשוויים וחמים למהנדסי בדיקות:

- אוטומציה ובדיקות מבוססות מודלים.
- ידע וניסיון בקידוד ובעבודה עם שפות תכנות.
- עבודה בסביבה אג'ילית (Agile).
-

ועבור מנהלי בדיקות:

- ניהול בסביבת Agile.
- אוטומציה ובדיקות מבוססות מודלים.

> **תמיד אבל תמיד השקיעו בהרחבה, בטיפוח ובשימור של קשרים מקצועיים.**

3. בנו לעצמכם פרופיל לינקדאין מלא ומרשים. לא! לא כשתתחילו לחפש עבודה- הרבה לפני כן! בדיוק כמו ביטוח, בלינקדאין משקיעים לפני התאונה ולא אחריה...
4. בנו את עצמכם כמותג, כמומחים, כטאלנטים ללא קשר לחברה בה אתם עובדים. השתתפות בפורומים, כתיבת בלוג, התנדבות, מתן יעוץ מקצועי באופן עצמאי הם דרך מצוינת לבנות לעצמכם שם שאינו תלוי בכלל במקום העבודה.
5. דמיינו לעצמכם שאתם עצמאים ושהעסק שלכם הוא... אתם. מה הייתם עושים בכדי לשמור על תזרים קבוע של לקוחות? בכדי לבנות מוניטין? בכדי להעלות הכנסות? עשו זאת! כל עובד הוא בשורה התחתונה עצמאי בעל עסק, רק שבמקרה במקום פרוייקטים זמניים וקצרים יש לו לקוח אחד בלבד ופרוייקט ארוך טווח...
בהצלחה! :)

http://www.careertips4geeks.blogspot.co.il/p/blog-page_15.html

מאיה בוכניק -
יועצת חיפוש עבודה וקריירה.

בעלת הבלוג Career Tips 4 Geeks שבו תמצאו מאות טיפים לתכנון קריירה ולחיפוש עבודה, שימוש נכון בלינקדאין וברשתות חברתיות, כתיבת קורות חיים, הצלחה בראיונות והכל מזווית מרעננת ומשעשעת (עד כמה שאפשר).

זיהוי והפעלה של פקדים גרפיים באוטומציה
מאיר בר-טל

מאיר בר-טל

מאיר בר-טל הוא ארכיטקט אוטומציה עם ניסיון של מעל 15 שנה בתחום התוכנה, מתוכן כ-12 שנה בתחום האוטומציה. מאיר עובד כפרילנסר מאז 2008 וביוני 2014 התמנה לתפקיד מנהל תחום אוטומציה בחברת UGenTech, שמתמחה באוטומציה פונקציונלית ובבדיקות ביצועים ועומסים. מאיר מתמחה באפיון ופיתוח של פתרונות אוטומציה כולל בניית Frameworks, תוספים והרחבות לכלים (לזיהוי פקדים לא סטנדרטיים), אינטגרציה בין כלים, בפרט עם כלי HP כגון QTP/UFT ו-ALM/QC, ו-Test Complete של Smartbear. למאיר ניסיון עם מגוון טכנולוגיות (Java, Web, .Net, WPF, PB וכו') ותחומי תעשיה (בילינג, CRM, פיננסים, אחסון, מכשור רפואי, תקשורת וכו') כמו כן מאיר עוסק בהדרכה (הן העברת קורסים על UFT והן הכשרת אנשי אוטומציה בארגונים). כשכיר מאיר עבד ב: נס איי.אס.איי. (כיום dbMotion), טייפרידר, אמדוקס (מפתח ואיש אוטומציה במגוון פרוייקטים), מטריקס (אינטרווייז - איי.טי. אנד טי., לאומי), וסופר דריבטיב (ראש צוות אוטומציה). בין החברות שלהן מאיר סיפק בעבר או מספק בהווה משירותיו: HP, IBM XIV, Ginger Software, Omnisys, Experitest, dbMotion, Mazor Robotics, Stratasys, Carestream, סופרפארם, בנק לאומי, האוניברסיטה הפתוחה, הבנק הבינלאומי, Biosense Webster, Kodak, Hermes Logistics, YIT. מאיר הוא הבעלים של האתר הפופולארי www.advancedqtp.com (מקושר לדף הפייסבוק באותו שם), שהוקם ב-2007 ובו מתפרסמים חומרים לימודיים ומאמרים מקצועיים והאתר גם מארח פורום להעלאת שאלות. לאחרונה מאיר גם פרסם ספר הדרכה בשם Advanced UFT 12 for Test Engineers Cookbook במשותף עם Jonathon Lee Wright, עם דגש מאוד חזק על נושא השימוש בטכניקות קידוד מתקדמות להשגת פתרונות יעילים באוטומציה. הספר מכיל מתכונים מעשיים לטיפול במגוון נושאים באוטומציה עם UFT 12 ויצא לאור בהוצאת Packt Publishing. הספר זמין גם בכריכה רכה וגם כ-eBook בכתובת: https://www.packtpub.com/application-development/advanced-qtp-12-test-engineers-cookbook.

המייל ליצירת קשר עם מאיר: meir.bar-tal@advancedqtp.com

הקדמה: הטמעת כלי אוטומציה בתהליך הבדיקות

כלי אוטומציה לבדיקות פונקציונליות, כגון UFT[1] של חברת HP, שתומכים בבדיקת תקינות הפעולה של פקדים גרפיים (GUI controls) ודרכם את הקוד בשכבות הלוגיות של האפליקציה, קיימים כבר שנים רבות. מיכון תסריטי בדיקות הוא למעשה תרגום לקוד של פעולות המשתמש, כך שבעת הרצה כלי האוטומציה מבצע חיקוי (emulation) של פעולותיו של משתמש בן-אנוש.

הרעיון המרכזי באוטומציה הוא שבאמצעותה ניתן למכן בדיקות רגרסיה, כאלה שמבוצעות באופן חוזר ונשנה לפני שחרור כל גרסה. בנוסף, אוטומציה פותחת אפשרויות למיכון בדיקות שקשה מאוד, אם לא בלתי אפשרי, לביצוע ידנית. אוטומציה מאפשרת קבלת תוצאות יותר מדויקות, מהימנות וניתנות לשחזור בהשוואה לבדיקות ידניות. זאת כמובן, בהנחה שהן מפותחות בצורה נכונה. כפי שמצויין בהמשך, אוטומציה דורשת משאבים לא מבוטלים כדי להשיג תוצאות. אוטומציה היא רובד נוסף שתומך בתהליך הבדיקות, אך אין ביכולתה להחליף את התבונה האנושית ולכן הכנסת אוטומציה לארגון בדרך כלל לא רק שלא תצמצם את המשאבים המושקעים בבדיקות - היא תגדיל אותם.

הטמעת אוטומציה לתהליך בדיקות האפליקציה אין פירושה פתרון קסמים. כמו שעמיתי Jim Hazen ציין בקבוצת דיון באינטרנט:

"It's automation, not auto-magic!". כך למשל, יכולות ההקלטה והניגון של תסריטים אוטומטיים באמצעות הכלים השונים מטעים לפעמים אנשים לחשוב שניתן להפיק תועלת מאוטומציה ללא מאמץ ניכר. אולם, המציאות מוכיחה שהדברים מורכבים הרבה יותר, כיוון שהקלטת סדרת פעולות יכולה להוות בסיס לתסריט, אבל אין היא יכולה להחליף את הקידוד והלוגיקה של בדיקה. בשונה מבן-אנוש, התסריט הממוכן אינו גמיש ויכול לבצע רק את הפקודות שניתנו לו, כך שלא יוכל להפעיל שיקול דעת בעת תקלה לא צפויה או התנהגות משתנה שתלויה בהרשאות משתמש או בקונפיגורציה ולקבל את ההחלטות המתבקשות. כמו כן, לא תמיד כלי האוטומציה תומך בטכנולוגיה הספציפית ששימשה לבניית האפליקציה, ואז נדרש איש האוטומציה לפתח הרחבות שיאפשרו לכלי לזהות את הפקדים ולבצע פעולות באמצעותם.

הטמעת אוטומציה דורשת השקעה ראשונית גבוהה למדי ברישיונות (אם לא בוחרים בפתרונות קוד-פתוח כגון Selenium, שמוגבל בכל מקרה רק לאפליקציות Web אך דורש ידע תכנותי רב בשפות כמו Java או C#) ובפיתוח התשתיות הנחוצות (כולל Framework שישרת את אנשי האוטומציה והבדיקות כאחד). לאחר שלב ההקמה, כדי שאוטומציה תניב החזר על ההשקעה היא דורשת השקעה עקבית בתחזוקת התשתיות, התסריטים והרחבת כיסוי הבדיקות שהם מאפשרים. לסיכום, פיתוח והטמעת אוטומציה דורשים יכולת לאפיין ולתכנת, וכן ידע מעמיק בטכנולוגיות שונות וראוי לבצע אותו באופן מושכל, עם כוח אדם מיומן, באופן שיטתי ולפי הסטנדרטים המקובלים בתעשייה, עם משאבים ראויים וקביעת מטרות בנות השגה.

בהמשך, המאמר יתמקד בהיבטים של זיהוי והפעלת פקדים בעזרת UFT של חברת HP ויסקור בקצרה שתי גישות מרכזיות לאוטומציה, מה ניתן לעשות עם פקדים גרפיים וסוגיות בזיהויים על-ידי הכלי, זיהוי שבלעדיו לא ניתן לממש תסריטים אוטומטיים.

סוגי כלים: אוטומציה מכוונת אובייקט או תמונה

ככלל, כלי אוטומציה פונקציונלית ל-GUI מתחלקים לשני סוגים עיקריים: כלים מכווני אובייקט (למשל: Test Complete, UFT, Rational Robot, Ranorex), שפועלים ברמת הקוד באמצעות hooks, וכלים מכווני תמונה שמזהים אובייקטים באופן שמדמה משתמש אנושי (למשל: SeeTest, eggPlant, T-Plan Robot). יחד עם זאת, לפחות לחלק מהכלים מכווני האובייקט יש גם יכולות מכוונות תמונה, כגון טכנולוגיית ה-Insight שנוספה ל-UFT (בנוסף ליכולת שהייתה קיימת ב-QTP לבצע נקודת בדיקה על bitmap). לשתי השיטות יתרונות וחסרונות. ככלל, כלים מכווני אובייקט נוחים יותר לתחזוקה ומתפקדים טוב יותר, אבל בגלל שהם תלויים במערכת ההפעלה, לפעמים יהיה עדיף לעבוד עם כלים מכווני תמונה, כמו למשל במקרה של אפליקציה שפועלת תחת מערכת הפעלה כגון לינוקס או Mac OS X, ואף כדי למכן בדיקות על מערכות שפועלות בסביבת מובייל (לדוגמא: SeeTest של חברת Experitest).

הטבלה הבאה מסכמת (בהכללה או בהערכה ממוצעת) השוואה לגבי עשרה היבטים רלוונטיים.

---

1 UFT הוא בעצם איחוד של שני כלי אוטומציה של חברת HP: QTP (לבדיקות GUI) ו-ST (לבדיקות SOA/API) לכלי אחד ועל כן שמו Unified Functional Testing, החל מגרסה 11.5.

זיהוי והפעלה של פקדים גרפיים באוטומציה

מאיר בר-טל

קריטריון	מוכוון אובייקט	מוכוון תמונה	הערות*
תלות ברזולוציה, גודל, מיקום, עיצוב	לא	כן	
גישה למאפיינים ופונקציות	כן	לא	
תלות במערכת הפעלה	כן (Windows בלבד*)	לא	קיימים תוספים/כלים נלווים שמאפשרים גם עבודה על mobile devices וכן דרך terminal emulator על מערכות Mainframe וכו'
תלות בחומרה	בינונית	בינונית-גבוהה*	תלוי כלי לדוגמה: רזולוציית מסך
תלות בטכנולוגיה	כן	לא	לדוגמה: סביבת הפיתוח: *Java, .Net, Web* מערכת הפעלה: חלונות/לינוקס
רמת קושי בפיתוח (*development*)	בינונית*	בינונית	תלוי באופן היישום
רמת קושי בתחזוקה (*maintainability*)	נמוכה-בינונית*	בינונית-גבוהה	תלוי באופן היישום
ביצועים	בינוניים-גבוהים*	נמוכים-בינוניים*	תלוי כלי וחומרה
רמת חוסן (*robustness*)	גבוהה*	נמוכה-בינונית	תלוי באופן היישום
רמת כישורים טכנולוגיים נדרשת	גבוהה	בינונית	

מהטבלה אנחנו למדים ששקלול הפרמטרים - לפחות עבור אפליקציות שתומכות ב-Windows - נוטה לטובת אוטומציה מוכוונת אובייקט, כיוון שחוסן, רמת קושי בתחזוקה, אי-תלות בפרמטרים של תצוגה וגישה לקוד הפנימי של האפליקציה הם קריטיים. לדוגמא, כדי לבדוק את הערך של מאפיין מסויים (לדוגמא: טקסט או מצב הפקד - זמין (enabled) או לא זמין (disabled), וכיו"ב), נדרשת גישה לקוד הפנימי - למעשה גישה לכתובות ב-RAM שבהן הקוד של האפליקציה או המערכת הנבדקת נטען מרגע הפעלתה ועד סגירתה. השחרור מתלות בטכנולוגיה על-ידי כלים מוכווני תמונה אמנם מפתה, אך קיום אפשרויות הרחבה בלתי מוגבלות, ובהשקעה סבירה, לכלים מוכווני אובייקט - מפחיתה משמעותית את הנטייה לבחור בהם. בצד אלה, הרגישות שיש לטכנולוגיות עיבוד תמונה לאוטומציה במאפייני החומרה (רזולוציה, גודל וסוג מסך משתנים) אף היא צריכה להילקח בחשבון. בנוסף, היכולות הנוספות לזיהוי אובייקטים לפי תמונה, כפי שהוכנסו ל-UFT, מאפשרות שילוב בין שתי הגישות במקומות שבהם אין מספיק משאבים לפיתוח הרחבה שכזאת. מאמר זה, על כן, יתמקד באוטומציה מוכוונת אובייקטים תוך שימוש ב-UFT.

הערה: פרט לסעיף על זיהוי פקדים באמצעות תמונה (*Insight*), סקירה זאת נכונה גם לגרסאות קודמות של *QTP*. בנוסף, כפי שהכותרת מעידה, לא נעסוק כאן במיכון בדיקות *API*, ש-*UFT* תומך גם בהן.

כיצד UFT עובד עם פקדים גרפיים

UFT עושה שימוש במחסן אובייקטים (Object Repository) כדי לשמור את הנתונים הדרושים לו לזיהוי הפקדים בזמן ביצוע פעולות עליהם, כשמריצים תסריט אוטומטי. מחסן זה הוא למעשה בסיס נתונים, ופרט לייצוג של פקדים גרפיים הוא גם משמש לאחסון נקודות בדיקה (checkpoints) שבעזרתן ניתן לבדוק ערכים של מאפיינים שונים של פקדים ועוד. ניתן להוסיף פקדים למחסן האובייקטים בשני אופנים:

1. אוטומטית: ל-UFT יש יכולת הקלטת פעולות משתמש על הממשק הגרפי. במצב הקלטה, הפעולות הללו מתורגמות לקוד (VBScript) שהוא למעשה תסריט אוטומטי בסיסי, וכן להוספה של אותם פקדים שעליהם בוצעו הפעולות.
2. ידנית: באמצעות חקירת אובייקט על ידי שימוש ב-Spy והוספתו, או באמצעות שימוש במסך של מחסן האובייקטים עצמו, שמאפשר למעשה חקירה והוספה.

בעת הרצת התסריט UFT מאחזר את תיאור האובייקט המצויין בקוד ממחסן האובייקטים, ודרך הממשקים שלו משיג גישה לפקד עצמו, על הפונקציות והמאפיינים הציבוריים שלו (כמובן שאלה שמוגדרים באפליקציה כפרטיים לא יהיו נגישים ל-UFT), ומבצע את הפקודה הכתובה בשורת הקוד עם הפרמטרים שלה.

פעולות אלה נכתבות אוטומטית על-ידי UFT לדו"ח, בדרך כלל כ-Done. במידה ויש כישלון הוא יסומן ככזה (Fail), לפעמים רק כאזהרה (Warning), כמו במקרה של נקודת סנכרון. נקודת בדיקה (Checkpoint) תסומן כ-Pass/Fail.

ביצוע פעולות על פקדים גרפיים

UFT מספק לנו ממשק מאוד נוח להפעלת פונקציות נפוצות של לחיצה על כפתורי העכבר (click), גרירה (drag, drop), הקלדת נתונים (type, set), בחירה מרשימה (select) ועוד. כך למשל, נוכל לנווט בין הדפים של אתר אינטרנט בעזרת הקלטת פקודות כגון:

```
Browser("חדשות תוכן ועדכונים ynet").Page("חדשות תוכן ועדכונים ynet").
Image("מחשבים").Click
```

כאשר "חדשות תוכן ועדכונים ynet" ו-"מחשבים" הם שמות דף האינטרנט והתמונה שהוא מכיל במחסן האובייקטים. שמות אלה הם בעצם כמו מפתח ראשי (primary key) - מזהה ייחודי בבסיס נתונים (זכור כי זו מהותו של מחסן האובייקטים). בזמן ריצה, UFT פונה למחסן האובייקטים ושולף את תיאור האובייקט המאוחסן בו, ובעזרתו משיג גישה לאובייקט שטעון בזיכרון המחשב.

ניתן כמובן גם לכתוב פקודות באופן ידני, שלא דרך הקלטה, בעזרת טכניקה שמכונה על-ידי HP בשם Descriptive Programming (או בקיצור, ר"ת DP). המשמעות של DP היא שתיאור האובייקט (description) נכתב ישירות בתוך הקוד, במקום שישמר במחסן האובייקטים ויישלף משם בזמן ריצה. מסיבה זאת, אני מעדיף לכנות טכניקה זו שימוש ב-inline description (כלומר, תיאור שנכתב בשורת הקוד).

לדוגמא, נוכל לכתוב את הפקודה הקודמת באופן ידני (לא דרך הקלטה) כך:

זיהוי והפעלה של פקדים גרפיים באוטומציה

מאיר בר-טל

```
Browser("CreationTime:=0").Page("title:=ynet.*").Image("image type:=Image Link", "alt:=מחשבים").Click
```

וכך אנחנו פונים לדפדפן הראשון שנפתח, לדף שהכותרת שלו מתחילה במחרוזת "ynet" (תוך שימוש בביטוי תבניתי – regular expression) ולתמונת קישור שבין מאפייניה alt שערכו "מחשבים" ומבצעים פעולת לחיצה (כפתור שמאלי הוא ברירת המחדל של פונקציית ה-Click). הפעלת הפקד באופן הזה נעשית בדרך כלל על-ידי שליחת Browser Event, ולאו דווקא Mouse Event. אולם, באפליקציות Web כמו גם באפליקציות אחרות, ישנם מקרים רבים שבהם הפעלת פקד באופן הזה לא תעלה יפה, כיוון שמצב הפקד משתנה לעיבוד event של onclick אך ורק אחרי שהעכבר נכנס לתחומי הפקד במסך (onmouseover). על כן בעת בניית תשתיות אוטומציה, מקובל לפתח גרסה אחרת לפונקציית ה-Click (תוך שימוש במנגנון דרישת הפונקציות של UFT עם RegisterUserFunc) או שיטות מתקדמות אחרות), שתבצע תחילה הזזה של סמן העכבר לתחומי הפקד, ורק לאחר מכן לחיצה, תוך שימוש במנגנון ה-DeviceReplay של UFT.

נקודות בדיקה וסינכרוניזציה

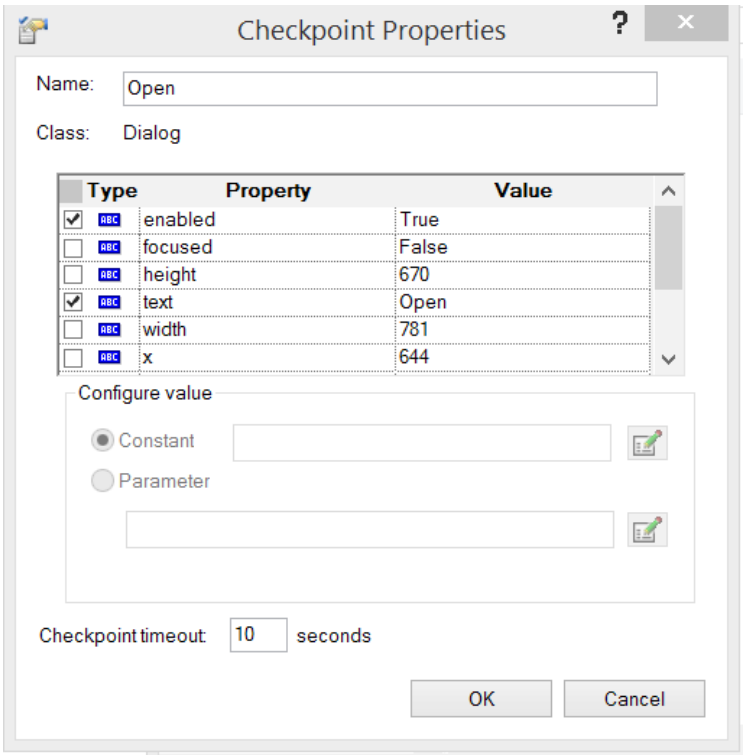
בעת הרצת תסריט בדיקה ממוכן, ודאי נרצה לאסוף מידע על מצב האפליקציה לשם דיווח על סטיות מההתנהגות הצפויה וכן לבקרת הזרימה של התסריט. כך למשל נרצה לוודא שכפתור ה-OK הופך לנגיש בטופס רק לאחר שמולאו כל שדות החובה (mandatory fields), שהכותרת של החלון מכילה את פרטי חשבון הלקוח הנוכחי, שהצבע של תמונה משתנה בהתאם לבחירה, שחלון נפתח אחרי בחירה מתפריט, ועוד. UFT מאפשר לנו לוודא את מצבו של כל פקד (להזכירכם, ב-Windows כל פקד הוא בעצם חלון). למשל, אחרי שבחרנו קובץ->פתח מהתפריט של Notepad על-ידי הפקודה:

```
Window("Notepad").WinMenu("Menu").Select "File;Open...  Ctrl+O"
```

נוכל לוודא שאכן נפתח חלון שמאפשר לנו את בחירת הקובץ על ידי הפקודה:

```
Window("Notepad").Dialog("Open").Check CheckPoint("Open")
```

שמשתמשת בפרמטר באובייקט מסוג CheckPoint ששמור במחסן האובייקטים. בעת ההקלטה נבקש שימתין עד 10 שניות תוך בדיקה שהחלון נגיש עם הכותרת המתאימה Open , וכך נראה חלון ההגדרה שלו:



כפי שכבר נאמר, ברור שנצטרך לשלב את הפקודה במבנה החלטה כלשהו כגון if...else...end if שיאפשר לנו לשלוט בזרימת התסריט בהתאם להצלחה או הכישלון של הפעולה הקודמת. בנוסף לשימוש בנקודות בדיקה, UFT מספק פונקציות נוספות (CheckProperty, WaitProperty) כדי לבדוק מאפיינים בודדים ולבצע סינכרוניזציה של התסריט עם תגובות האפליקציה, ואפילו בפשטות לבדוק הימצאות של החלון או הפקד עם הפקודה Exist, בדיוק כפי שנוהג משתמש בן-אנוש. ניתן גם לבדוק באמצעות CheckPoints נתונים שמוצגים בטבלאות, קבצי XML וטבלאות בבסיסי נתונים, עליהם לא נפרט כאן מקוצר הירעה.

זיהוי פקדים

ביצוע הפעולות המוזכרות לעיל עם הפקדים מתאפשרת הודות למנגנון זיהוי האובייקטים של UFT, שתומך במגוון רחב מאוד של סביבות טכנולוגיות לפיתוח אפליקציות ובהן Web, Java, .Net, WPF ועוד. כאמור לעיל, UFT מאפשר פיתוח הרחבות עבור אובייקטים לא סטנדרטיים, בין אם הם מסחריים כגון Infragistics, DevExpress ואחרים, ובין אם הם מפותחים בתוך הארגון. ניתן לחקור את האובייקטים של האפליקציה על ידי שימוש ב-Spy ש-UFT מספק. מבלי להיכנס לעומק הטכנולוגיה שבבסיס זיהוי אובייקטים, בזמן הקלטה או לימוד אובייקטים, UFT אוסף נתונים על הפקדים בהם אנחנו "נוגעים". נתונים אלה נלקחים מתוך המאפיינים הפנימיים של האובייקט וכוללים את סוג הפקד (כפתור, כפתור רדיו, רשימת בחירה, חלון וכדומה) וסט מינימלי של מאפיינים שמאפשר זיהוי חד-חד ערכי של הפקד באותו מצב אפליקציה (application state) או קונטקסט). כדי להשיג אוטומציה חסונה יותר לשינויים ועל-ידי כך להקל על התחזוקה רצוי להשתמש במאפיין בודד שהוא תג הזיהוי הייחודי של הפקד (למשל: html id ב-Web, devname ב-WPF וכיו"ב). כדי להבין את התהליך שמתרחש בזמן ריצה, UFT מבצע סוג של תשאול דרך ה-hooks ששתל באפליקציה כדי לקבל מצביע לכתובת בזיכרון (reference) של פקד שעונה על הקריטריון של מאפיין = ערך (או סט של מאפיינים כאלה וערכיהם), כפי שמאוחסן במחסן האובייקטים או כפי שהראינו לעיל כפי שהוגדר באמצעות inline description. מרגע שיש ל-UFT את ה-reference, הדרך פתוחה להפעלת מתודות ציבוריות פנימיות ואחזור נתונים מתוך מאפייניו.

> **הערה: אם באפליקציה הנבדקת אין לפקדים תגי זיהוי כאלה (שהם דבר סטנדרטי בפיתוח), רצוי לברר עם צוות הפיתוח אם יש סיבה מיוחדת לכך ואם לא, לבקש שיוסיפו אותם למען יעילות העבודה.**

זיהוי פקדים דינמי בזמן ריצה תוך שימוש במאפייני אובייקט (*description*)

לעתים קרובות אין אפשרות לזהות אובייקטים מראש על ידי שימוש ב-Spy ואחסון מאפיינים במחסן אובייקטים, מפני שהם נבנים באופן דינמי. דוגמא טיפוסית לכך הם פקדים כגון קישורים (Links) אשר מבוססים על נתונים שמאוחזרים בזמן אמת מבסיס הנתונים. במקרה כזה ניתן להשתמש בטכניקה מתקדמת ש-UFT מאפשר לנו: שליפת אוסף האובייקטים, שנמצאים בזמן נתון בחלון האפליקציה הרלוונטי, ועונים על תיאור מסוים (לדוגמא: מסוג Link ושהטקסט שלהם (innertext) תואם לתבנית מסויימת). כלומר, אנחנו בעצם יכולים להפעיל סינון על אוסף האובייקטים ש-UFT "רואה", ולקבל רק את אלה שרלוונטיים למשימה הנוכחית. כדי לעשות זאת, נשתמש בפונקציית ChildObjects אשר משותפת לכלל האובייקטים ש-UFT מספק בטכנולוגיות השונות (Web, .Net, Java וכו'). לדוגמא, כדי לשלוף את כל הקישורים מדף אינטרנט שהטקסט שלהם מתחיל במחרוזת "account", נכתוב קוד כדלהלן:

```
Dim LinksDescription, LinksCollection
Set LinksDescription = Description.Create()
LinksDescription("html tag").Value = "a|A"
LinksDescription("html tag").RegularExpression = True
LinksDescription("innertext").Value = "account.+"
LinksDescription("innertext").RegularExpression = True
Set LinksCollection = Browser("Account Management").Page("Search Accounts").ChildObjects(DescriptionObject)
```

זיהוי והפעלה של פקדים גרפיים באוטומציה

מאיר בר-טל

אוטומציה מאפשרת קבלת תוצאות יותר מדויקות, מהימנות וניתנות לשחזור בהשוואה אך אין ביכולתה להחליף את התבונה האנושית ולכן הכנסת אוטומציה לארגון בדרך כלל לא רק שלא תצמצם את המשאבים המושקעים בבדיקות - היא תגדיל אותם.

כעת, לאחר שיש לנו אוסף קישורים במשתנה LinksCollection, אותו קיבלנו בעזרת התיאור ששימש כמסנן (LinksDescription), ניתן לבצע פעולות ובדיקות עליהם, למשל, נוכל לעבור אחד אחד וללחוץ על הקישור כדי לבצע תסריט מסויים עבור כל אחד (במקרה כזה, עדכון מידע עבור כל חשבון בעזרת AccountUpdateInfo, פונקציה שכמובן יש לממש ומוצגת כאן רק לצורך המחשה):

```
For i = 0 To LinksCollection.Count-1
        Set objLink = LinksCollection(i)
        objLink.Click()
        Call AccountUpdateInfo()
Browser("Account Management").Back()
Next
```

זיהוי פקדים דינמי בזמן ריצה תוך שימוש בעוגנים (*anchors*)

טכניקה נוספת לזיהוי אובייקטים, שלא ניתן לזהות לפי מאפיינים רגילים, היא שימוש בעוגנים. UFT מאפשר לעשות את זה באופן מובנה על-ידי הגדרת Visual Relations במחסן האובייקטים לפקדים כאלה, באופן שהזיהוי של האובייקט יהיה ביחס למיקומו ביחס לאובייקט אחר (העוגן), שאותו כן ניתן לזהות בשיטה הרגילה. השימוש בעוגנים כמובן בעייתי עקב הרגישות למיקום הפקדים על המסך, ולכן לא שכיח.

UFT עושה שימוש במחסן אובייקטים (Object Repository) כדי לשמור את הנתונים הדרושים לו לזיהוי האובייקטים בזמן ביצוע פעולות על פקדים

זיהוי פקדים באמצעות תמונה (*Insight*)

UFT הכניס לשימוש מנגנון שמאפשר לזהות אזור במסך (מלבן) לפי תמונה, בלי תלות במיקום. בשימוש סטנדרטי האובייקט מאוחסן במחסן האובייקטים והוא מסוג InsightObject, וניתן להגדיר את מידת הדמיון הנדרשת מהאזור במסך לתמונה (ברירת המחדל היא 80%). התכונה הזאת של UFT עדיין רגישה לגודל ולרזולוציה, אבל ניתן להתגבר על זה עם קידוד מנגנון חכם שמאפשר זיהוי דינמי בזמן ריצה תוך התאמה ליותר מתמונה אחת. כך ניתן להחזיק סט של תמונות בגדלים שונים ולזהות את אובייקט המטרה בלי תלות בגודל ובמיקום על המסך.

כדי להשיג אוטומציה חסונה יותר לשינויים ועל-ידי כך להקל התחזוקה, רצוי להשתמש במאפיין בודד שהוא תג הזיהוי הייחודי של הפקד

חדש ומתחדש בעולם ה- Agile
Tribes, Chapters, Squads & Guilds

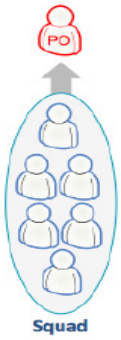
אלון לינצקי, מנכ"ל Best – Testing, מייסד פורום סיגיסט ישראל, ומייסד-שותף של ISTQB®

[בהשראת שיחות עם מיישמים של המודל כתבות ועדכונים בנושא ובהשראת המאמר Scaling Agile Spotify, שנכתב על ידי Henrik Kniberg & Anders Ivarsson, בשנת 2012. כל הזכויות שמורות להם]

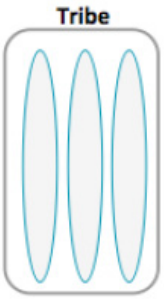
לא מכבר, חידשה לנו חברת Spotify והביאה לנו מודל יישום חדשני בעולם האג'ייל (החל מ- 2012 עבורם). ארגון קבוצות הפיתוח ב- Squads, Tribes, Guilds. הנושא תופס תאוצה כיום ומתפרסם יותר ויותר כמודל לחיקוי, בחברות סטרטאפ שמתרחבות מהר ומאמצות תרבות ארגונית ושיטת ארגון פיתוח מוצרים זו. בכנס האחרון בו הרצתי, TechHub בפורטוגל, הרצה מנכ"ל Fiverr, שי ווינינגר, והציג איך הם מיישמים את המודל בהצלחה.

השינוי העיקרי הינו בתרבות הארגונית, ואופן יישום העצמאות והמוטיב של - Self-Organizing Teams בשיטה שמאפשרת התפתחות כמעט עצמאית של כל קבוצה ועבודה מהירה בתחומה כולל שחרור גרסאות לייצור.

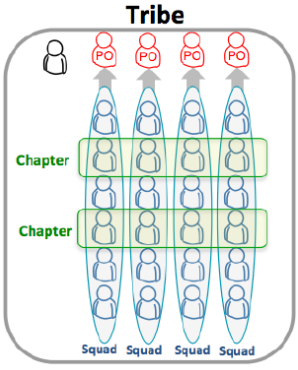
הרעיון מאחורי שיטה זו הינו ארגון קבוצות פיתוח באופן עצמאי (כל Squad היא קבוצה עצמאית) לשחרר גרסאות של אזור הקוד שלהם לייצור (Production) ללא שתצטרך לחכות לקבוצות אחרות. משמעות העניין הוא שיש לארגן כל קבוצה באופן שיש בה את כל התפקידים הנדרשים לפיתוח, עיצוב, בדיקה ושחרור לייצור.



SQUAD: למעשה מעיין "מיני סטרטאפ"
אנשי ה Squad יושבים יחד באותו מיקום (Co-location), ממוקדים על מטרה ארוכת טווח, והינה **Self-Organizing Team** (קבוצה המנהלת את עצמה). לקבוצה יש גם Product Owner שתפקידו לתעדף את העבודה ונמצא בקשר עם קבוצות Squad אחרות באותו Tribe (ראה בהמשך).
ה-Squad הינו "הבית העיקרי" של הקבוצה (או השיוך העיקרי שלה).

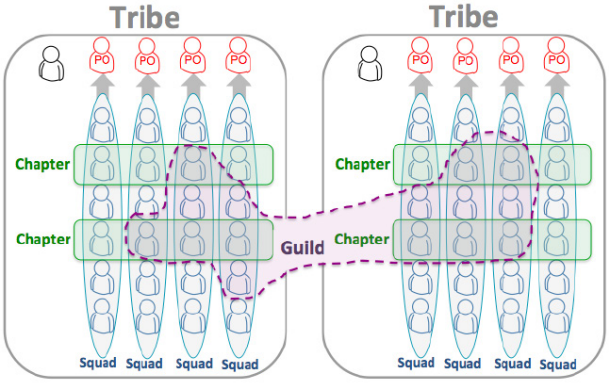


TRIBE: מקום "גידול" עבור ה-"מיני סטרטאפים"
מספר קבוצות Squad בעלות קשר מקצועי משותף, השייכות לאותו אזור עיסקי (אותו מוצר למשל) ישתתפו ב- Tribe. הקבוצות יושבות באותו מקום גיאוגרפי (עד 100 איש ב Tribe). ה- Tribe מקדם מטרות המשותפות לקבוצות ה- Squad שלו ומקיים פגישות לא-פורמליות, שיתוף מידע ומצב עבודה בקבוצות השונות, מקיים Demos משותפים, ומסנכרן את העבודה של ה- Squads (לדוגמא: ה- Mobile Tribe, כולל את ה- Squads : iPad, iPhone, Android).
ל- Tribe יש מוביל Tribe, שאחראי לספק לקבוצות יחד סביבת "צמיחה וגידול" מתאימה.



CHAPTER: לאנשים שעוסקים בתפקידים דומים
נשוא ה- Chapter, למעשה מיישם פורומים באופן קבוע אשר מסנכרנים בין המקצועות השונים והידע הרוחבי של אותם בעלי תפקידים לרוחב קבוצות ה- Squad השונות באותו ה- Tribe. אנשים אלו נפגשים תדירות לשוחח על אתגרים, ופתרונות אפשריים. ה- Chapter הינו "הבית השני" של ה- Squad. לדוגמא: Testing Chapter, Web Development Chapter וכו'.
ל- Chapter יש מוביל Chapter, אשר משמש כמנהל line לכל דבר ואחראי על פיתוח מקצועי של האנשים ב- Chapter, קביעת משכורות וכו', אך עדיין הוא שייך ל- Squad שלו, ועוסק בה ביום יום.

GUILD: קהילות עניין
קבוצות אנשים, אשר מעוניינים לחלוק ולשתף מידע, כלים, תהליכים, שיטות, קוד, וטכניקות בארגון לרוחב ה- Tribes. דוגמאות: Web Technology Guild, או Test Automation Guild. שלא בדומה ל- Chapter, ה-Guild כוללת אנשים לרוחב הארגון. למעשה ה- Guild כוללת את כלל האנשים בעלי אותו תפקיד ב- Chapters השונים. כך ה- Test Automation Guild תכלול את כל ה- Test Automation Chapters לרוחב ה- Tribes. למרות זאת, כל מי שמעוניין יכול להצטרף.



לכל Guild יש "Guild Coordinator" שמבצע רק את התפקיד הזה בארגון.

רוצים לדעת יותר? בבקשה!

- Scaling Agile at Spotify – with Tribes, Squads, Chapters and Guilds
- Here's How Spotify Scales Up and Stays Agile
- Spotify Labs
- Squads, Chapters and Guilds
- Scaling Agile at Spotify - Slide share

פתרון השאלה

שאלה זו בודקת את הידע של הנבחן לגבי הצורך והתועלת של בדיקות אוטומטיות שמופיע בפרק 6 בסילבוס.
פרק זה מדבר על שימוש בכלים שונים לצרכי בדיקות. השאלה הזו מדברת על הכלים המוכרים לכולם ככלים להרצת בדיקות.
הבה נבחן את התשובות השונות:

הפתרון

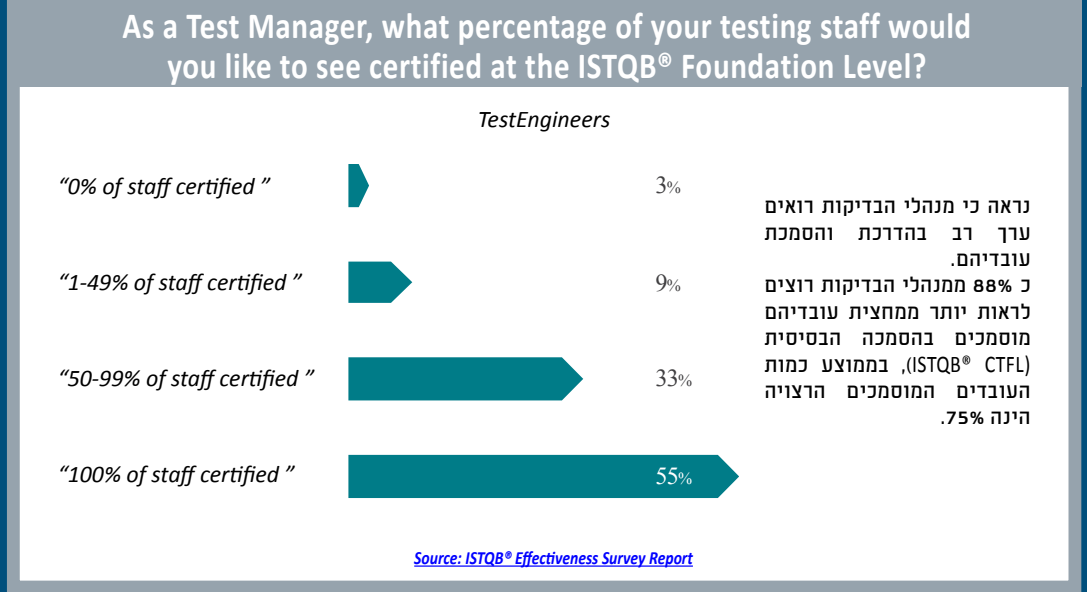
השאלה מדברת על ההבדלים בדרך החשיבה בין הבודקים והמפתחים, כמתואר בפרק 1.5 בסילבוס, שאומר שנקודת מבט של בודקים עצמאיים תהיה יעילה יותר מזו של בודק שבודק את התוכנה שלו. הפרק מוסיף ומתאר רמות שונות של עצמאות, החל מבדיקות שנעשות על ידי האדם שכתב את התוכנה, וכלה בבדיקות שנעשות על ידי חברת מיקור חוץ.

I. ☒ שימוש בכלים להרצת בדיקות יכול לאפשר לספק מידע נוסף לצרכי ניפוי באגים (debugging) אבל אין בזה כדי להקל על דיווח תקלות בכל צורה שהיא. תשובה זו אינה נכונה.

II. ☒ אחת האמונות הרווחות היא שמיכון של בדיקות תוביל להפחתה משמעותית בעלות הבדיקות. סברה זו היא שגויה מאוד. מיכון של בדיקות לא רק שלא יוזיל את עלות הבדיקות, אלא שאף עשוי לייקר אותן. עלות הכלים, תחזוקה של הכלים, כתיבת הקוד הדרוש לצורכי הבדיקות ותחזוקה של קוד זה עולים כסף. החיסכון יבוא בכך שמיכון הבדיקות יאפשר להריץ יותר בדיקות וכך לאפשר כיסוי יותר טוב של המערכת הנבדקת ולגלות יותר פגמים (באגים). תשובה זו אינה נכונה.

III. ☒ הזדמנות ללמוד יכולת חדשה ולחזק את אנשי הבדיקות היא אכן תועלת שתושג משימוש בכלים אוטומטיים. אך זו לא תהיה סיבה להשתמש בכלים אוטומטיים וגם לא בטוח שהיא תביא תועלת לפרוייקט המדובר, בייחוד אם זהו הפרוייקט הראשון שנעשה בו שימוש בכלים אוטומטיים. תהליך הלימוד של אנשי הצוות לוקח זמן. מיכון כל הבדיקות לוקח זמן. שני אלה עשויים להימשך הרבה יותר זמן מהפרוייקט. ולכן הזדמנות זו לא תייצר תועלת עבור הפרוייקט. תשובה זו אינה נכונה.

IV. ☑ בדיקות רגרסיה הן בדיקות שחוזרות על עצמן שוב ושוב במהלך הפרוייקט ולעיתים בין פרוייקטים שונים, בייחוד כשמדובר על גרסאות תחזוקה של מערכת. הרצה של בדיקות אלו דורש מהבודקים לחזור ולהריץ את אותן הבדיקות שוב ושוב, במקביל להרצה של בדיקות חדשות שבודקות שינויים שנעשו במערכת ותיקוני באגים. בפרוייקט המדובר יש 23525 מקרי בדיקות רגרסיה שצריך להריץ. הרצה ידנית של בדיקות אלו תימשך זמן רב וכך צוות הבדיקות יתקשה לעמוד בלוחות הזמנים ויצטרך לוותר על חלק מהבדיקות. מיכון של בדיקות הרגרסיה יאפשר הרצה אוטומטית של בדיקות אלו ובכך יימנע עבודה ידנית חוזרת של הבודקים שיתפנו לבצע מטלות בדיקה אחרות.
תשובה זו נכונה.



נראה כי מנהלי הבדיקות רואים ערך רב בהדרכת והסמכת עובדיהם.
כ 88% ממנהלי הבדיקות רוצים לראות יותר ממחצית עובדיהם מוסמכים בהסמכה הבסיסית (ISTQB® CTFL), בממוצע כמות העובדים המוסמכים הרצויה הינה 75%.

סקרים

התוכנית לקידום המחקר האקדמי בתחום איכות ובדיקות תוכנה

ITCB® הינה עמותה מקצועית, שקמה בשנת 2004 כחלק מהפעילות העולמית של ארגון ISTQB®, כדי לקדם את כל פעילות בדיקות התוכנה בישראל. החברים בארגון הינם אנשי מקצוע בתחום הבדיקות, שפועלים במסגרת הארגון בהתנדבות ושלא למטרות רווח.
חברי הארגון הישראלי לוקחים חלק פעיל ומשפיע בוועדות ההיגוי והעשייה השונות בארגון הבינלאומי.
מתוך כוונה להשפיע ולעודד שיתוף פעולה בין האקדמיה לתעשייה בתחום חשוב זה, הארגון יוצא בקול קורא לעידוד מחקרים אקדמיים בתחום בדיקות תוכנה ואיכותה.
לאחרונה אנו חשים התעוררות בקרב הקהילה האקדמית והגברת המודעות לענף איכות ובדיקות התוכנה. ארגון ITCB® מברך על כיוון חיובי זה ומעודד מעורבות נוספת ואינטראקציה בין התעשייה ובין האקדמיה בכל הקשור לאיכות ובדיקות תוכנה.
על מנת לעודד ולהחיש את המחקר האקדמי בנושאים הקשורים לכך החליט ITCB® לתמוך בתהליכי המחקר באמצעות הענקת מלגת לימוד שנתית למחקרים אקדמיים בתחום.
המלגה תבטא את הערכת הקהילה המקצועית לתרומתו הסגולית ולרמתו של המחקר שיתבצע. המלגות יתנו בשלוש רמות של מחקרים:

- מלגה בסה"כ 3000₪ לעבודת מחקר במסגרת תואר ראשון (פרויקט גמר)
- מלגה שנתית בסה"כ 5000₪ לעבודת מחקר ברמת תואר שני (Master)
- מלגה שנתית בסה"כ 10000₪ לעבודת מחקר ברמת תואר שלישי (PHD)

ייבחנו עבודות מחקר שייעשו במהלך שנות 2015 - 2016 ואשר יגיעו לידי סיום לפני סוף שנת 2016

תחומי מחקר (רשימה חלקית כל תוספת תיבחן)

- מדידה ובקרה של תהליכי איכות בתוכנה
- מיכון תהליכי בדיקה
- שילוב תהליכי איכות בפרויקט תוכנה
- מחקרים אמפיריים בתעשייה בתחום בדיקות התוכנה
- שיטות לניתוח קוד ואיכותו
- בדיקות התוכנה במציאות הטכנולוגית החדשה
- ALM ומחזור האיכות של התוכנה
- בדיקות תוכנה וניהול סיכונים בארגון המפתח
- בדיקות בעולם המוביל
- בדיקות משולבות תוכנה/חומרה
- בדיקות שירותי תוכנה בענן
- SOA ובדיקות תוכנה
- כלים ותשתיות לתהליכי איכות ובדיקות תוכנה
- תהליכי שיפור לבדיקות התוכנה
- שיטות לימוד והקניית ידע בנושא איכות ובדיקות

תנאי ההגשה

- עבודת המחקר צריכה להיות עבודה מקורית המתבצעת במסגרת מחקר אקדמי בליווי ופיקוח של איש סגל אקדמי מטעם המחלקה בה למד הסטודנט
- תינתן עדיפות למחקרים המתבצעים בשיתוף עם התעשייה
- המחקר מתקיים ומסתיים באותן שנים בהם מוענקת המלגה
- תקציר להצעת מחקר הכולל את תיאור נושא המחקר, תהליך העבודה עיקרי לביצועו ושמות המנחים האקדמיים יש להגיש לוועדה המקצועית אשר תוקם מטעם הארגון, לפני חודש אוקטובר 2015 (לשנה הקרובה).
- אין הארגון מתחייב להעניק מלגה כלשהי למחקר באם לא יוגשו הצעות ברמה אקדמית ומקצועית מספקת.

תאריך אחרון להגשת בקשות למלגה: 18.10.2015
תהליך בחינת הזכאות למלגה יתבצע באמצעות צוות המורכב מפרופ' מכובדים מהמוסדות האקדמיים בארץ וממומחים מהתעשיה.
הענקת המלגה תתבצע תחת פיקוח ובקרת הגופים המשפטיים המלווים את ITCB®.

ליצירת קשר והגשת העבודות – דני אלמוג 050-6316423 dani.almog@itcb.org.il
ובאתר ITCB® – www.itcb.org.il

ITCB® Israeli Testing Certification Board
ISTQB® International Software Testing Qualifications Board

צוות הכותבים

יאן ברון – M.Sc מדעי המחשב
הסמכת ISTQB® FL מארגוני הסמכה אמריקאי וישראלי
35 שנות ניסיון בהנדסת תוכנה וניהול בדיקות.
תפקיד נוכחי: מנהל בדיקות, **iMDsoft** ישראל
תחומי עניין – מתודולוגיה וכלי פיתוח לאזורים שונים בבדיקות. בדיקות אוטומציה, ניהול מחזור חיי מוצר: מתודולוגיה וכלים.
התנדבות – **ISTQB®**, ראש ועדת סקרים. **ITCB®**, ועדה המנהלים. **SIGIST** ישראל, יו"ר תכנית הכנסים.

קובי הלפרין – עוסק בבדיקות מזה מעל ל-20 שנה בעיקר בתחום התקשורת טלקום/דאטהקום (ECI, Alvarion, Axerra, Ceragon), בעל מעורבות גבוהה בקהילות בעולם ובארץ ומוביל מספר פורומים אינטרנטיים, מחפש להרחיב את הדיון בתחום הבדיקות, שיפור כלים והרחבת הידע של הבודקים.
פעיל בוועד המייעץ של ITCB® – בעיקר בשיפור האתר www.itcb.org.il והקשרים ברשתות החברתיות.

אייל זילברמן – מומחה בדיקות בכיר ונשיא קבוצת קווליטסט, חברת הבדיקות המובילה בישראל והשנייה בגודלה בעולם. פעיל בתחום בדיקות התוכנה משנת 1995, עבד כבודק תוכנה, מנהל ומוביל בדיקות במספר ארגונים כמו בנק ירושלים, Teleknowledge, T-Mobile ועוד. פרסם עשרות מאמרים מקצועיים ונואם פעיל בכנסים ישראליים ובינלאומיים. אייל הוא בין המקימים של "חושבים בדיקות" מגזין בדיקות התוכנה הישראלי הראשון וחבר בצוות ה-AB של ITCB® – ארגון ההסמכה לבדיקות הישראלי.
לינק: www.qualitestgroup.com

אלון לינצקי – בשלושים השנים האחרונות התמקצע, הדריך ואימן קבוצות וחברות בפיתוח, בדיקות והבטחת איכות.
תחומי המומחיות העיקריים של אלון הינם: עיצוב בדיקות, תכנון ואופטימיזציה של בדיקות, ניהול יעיל של בדיקות, שיפור ואופטימיזציה של תהליכי בדיקות ופיתוח, בדיקות אוטומטיות בפרוייקטים, ניהול בדיקות מבוסס סיכונים, מדדים ומטריקות לניהול איכות ובדיקות, בניית צוותים יעילים, שיפור תהליכי פיתוח ואיכות, אג'ייל ומעבר לאג'ייל.
אלון מרצה בינ"ל פופולארי מאז 1995, מוביל תוכנית שותפים של ISTQB® Partner Program,
מייסד The Israeli Testing Forum – SIGiST Israel, מייסד שותף של Israel Testing Certification Board – ITCB®. היה אחד מכותבי הסילבוס של ISTQB® Agile Tester – Foundation Level Extension, סגן נשיא ודירקטור שיווק של ITCB®.

טל פאר – בעל ניסיון של כ-20 שנים כבודק ומנהל בדיקות במגוון חברות וטכנולוגיות במודלי פיתוח שונים, וכמו כן משמש גם כיועץ ומדריך בדיקות. כיום טל מנהל בדיקות בחברת Kongsberg Maritime.
טל חבר ב-ITCB® ומוביל את קבוצת התהליכים ב-ISTQB®.

מיכאל שטאל – הוא ארכיטקט בדיקות תוכנה באינטל, ישראל. במשך 15 השנים האחרונות מיכאל בדק בתחום מודם כבלים, Wi-Fi, טלוויזיות חכמות, כרטיסים גרפיים, ולאחרונה הוא עובד בקבוצה שבודקת את התוכנה שמאחורי טכנולוגיית RealSense (מצלמות תלת-מימד) של אינטל.
מיכאל חבר ב – ITCB ומוביל את פעילות ה – Advisory Board.
במסגרת תפקידו, מיכאל מגדיר שיטות בדיקה ומתודולוגיות עבודה, עוסק הרבה בהדרכה ולפעמים אפילו מרשים לו לבדוק משהו (שזה הכי כייף).
מיכאל מציג תכופות בכנסים בארץ ובחו"ל והציג בין היתר ב – SIGiST Israel, STAR conferences, QA&Test ובכנסים אחרים. מיכאל מלמד בדיקות תוכנה בפקולטה למדעי המחשב באוניברסיטה העברית. ניתן לראות חלק מהמצגות והמאמרים שלו באתר www.testprincipia.com.

הסמכה חדשה מ ISTQB®

הסמכת בודק אג'ייל!

הסמכת בודק אג'ייל (Agile) נועדה להקנות ידע ל עובדים בסביבת אג'ייל או לאלו שמתכוונים להתחיל ביישום אג'ייל בעתיד הקרוב.

ההסמכה מעניקה יתרון לכל מי שמבקש להתמקצע בשיטות העבודה, תפקידים, פעולות נדרשות ומתודולוגיה של סביבת האג'ייל ספציפית לתפקידם במערכת, כולל בודקי תוכנה, אנליסט בדיקות, מהנדסי בדיקות, יועצי בדיקות, מנהלי בדיקות, בודקי קבלה ומפתחי תוכנה.

ההסמכה הנה תוספת (Extension) לרמה הבסיסית של ISTQB, המהווה תנאי לקבלתה .

דרישות להסמכה

- תעודת בודק תוכנה ברמה בסיסית של ISTQB® (CTFL)
- ציון עובר בבחינת אג'ייל של ISTQB®

סילבוס אג'ייל ומילון מונחים

הסילבוס זמין וחינם להורדה באתר ISTQB® ואתר ITCB®, כמו כן שאלות דוגמה לבחינה ומילון מונחים מעודכן לסביבת האג'ייל.

קורסי הדרכה לאג'ייל

מרכזי ההדרכה שיעבירו קורסי אג'ייל מפורסמים באתר הבית של ITCB®

לפרטים נוספים:

אתרים: www.istqb.org , www.itcb.org.il

מייל: info@itcb.org.il

טלפון: 03-6176640

ISTQB® Certified Tester | ITCB® Israeli Testing Certification Board