חושבים בדיקות
www.thinktesting.co.il
יולי 2011 גליון מס' 04

3 סיפורים מהחיים
על בדיקות אוטומטיות

הפוך למנוי
חינם
הרשם באתר

- סוף סוף רואים פנים
- כיצד לבחור כלי אוטומטי מתאים לארגון?
- ניהול צוותי בדיקה בינ"ל במודל Scrum
- מודל חישוב חומרת הבאג
- בדיקות Installation - לא פחות חשוב...

מגזין הבדיקות הראשון בעברית

חושבים בדיקות
www.thinktesting.co.il
אפריל 2011 גליון מס' 03

טורים, בלוגים, ספרים, מידע מקצועי, מאמרים....

כל מה שחשוב לדעת בתחום הבדיקות

הפוך למנוי וקבל את העיתון עד המשרד - בחינם:

www.ThinkTesting.co.il

תוכן העניינים

גליון מס' 04 | יולי 2011

4 **דבר העורך**

6 **3 סיפורים מהחיים על בדיקות אוטומטיות** מאמר מערכת

10 **ראיון עם מנהל בדיקות** צחי מלכי

12 **מודל לחישוב חומרת באג** אפרת וינברג

16 **המלצה על בלוג**

17 **המלצה על ספר**

18 **כיצד לבחון כלי אוטומטי לארגון?** רונית שמאי

22 **שילוב יוצאי אתיופיה בתחום הבדיקות**

24 **התנהלות קבוצות בדיקה בינ"ל מבוזרות בסקראם** נרי לביא

28 **בחן את עצמך – אוטומציה**

30 **סוף סוף רואים פנים – השימוש בפרסונות ככלי לתכנון ובדיקות שמישות** שמואל גרשון

34 **רשמים מכנס QA Extreme 2011**

36 **בדיקת ההתקנה והוראות ההתקנה** דורון בר

38 **מגמות בבדיקות תוכנה**

חושבים בדיקות

www.thinktesting.co.il

יולי 2011 גליון מס' 04

Editor: Joel Montvelisky
Professional Committee: Ayal Zylberman, Ram Yonish and Joel Montvelisky
Production coordinator: Ami Sterling
Design: Red Designer Studio
Website and free subscription: www.ThinkTesting.co.il
Contact: info@ThinkTesting.co.il
Printing: Papirus
Join "חושבים בדיקות" group on LinkedIn

עורך ראשי: יואל מונטבליסקי
וועדה מקצועית: איל זילברמן, רם יוניש ויואל מונטבליסקי
רכז מערכת: עמיחי סטרלינג
עיצוב: סטודיו Red Designer
אתר והרשמה למנוי חינם: www.ThinkTesting.co.il
יצירת קשר: info@ThinkTesting.co.il
הדפסה: פפירוס
הצטרף לקבוצת המגזין "חושבים בדיקות" ב-LinkedIn

דבר העורך יואל מונטבליסקי

שלום וברוכים הבאים לגליון החדש של חושבים בדיקות, העיתון העברי לבדיקות תכנה.

כפי שבטח שמתם לב (הרי אנחנו אנשי הבדיקות תמיד שמים לב לפרטים הקטנים) הגליון הנוכחי מוקדש לנושא בדיקות האוטומציה וראיתי לנכון להתחיל את "דבר העורך" עם האתגר הגדול ביותר שעומד בפנינו בנושא הזה: להוציא את האוטמציה מהאפלה. לפני מספר שבועות התחלתי דיון באחד מקבוצות ה- LinkedIn בתחום הבדיקות. הדיון היה סביב האתגר הגדול של מנהלי ה-QA בתחום האוטומציה ואחת התשובות המעניינות ביותר היתה של מנהל שכתב שבקרב הצוות שלו, אוטומציה היא עדיין בגדר "אמונה תפלה" (Dark Art). האתגר העומד לפניו הוא להוציא את האוטומציה לאור ולעזור למנהדסים שלו לעבוד על החששות המונעים מהם לקפוץ למים ולהתחיל לאמץ כלים אוטומטים כחלק מהעבודה היומיומית שלהם.

לערכתי, אותו מנהל QA ב- LinkedIn פגע בול באחת הבעיות המרכזיות שלנו כבודקים. מדובר בבעיה כפולה: קודם כל עלינו כבודקים להבין שאוטומציה הוא עסק טכני הדורש מאיתנו להתעסק (קצת) עם תיכנות, סקריפטים ואפילו עם שאילתות ובסיסי נתונים. אי אפשר לחשוב שנוכל לעשות שימוש באוטומציה בלי "ללכלך את הידיים" בכלל. החלק השני של הבעיה נעוץ בעובדה שאחוז גדול מהבודקים נמנעים מלהתעסק עם תכנות מכיוון שהם חושבים, באופן שגוי לחלוטין, שהם לא מסוגלים לכתוב או אפילו לקרוא שורת קוד אחת. אז נכון, אנחנו בודקים ולא מפתחים, אך חשוב לזכור שלרובנו יכולות טכניות לא מבוטלות וכתיבת קוד וסקריפטים בסיסים הם דברים פשוטים למדי. בדיוק כמו שחייה או רכיבה על אופניים, אתה קודם כל חייב לעבוד על הפחד מטביעה בבריכה או להבין שגם אם תיפול מהאופניים פעם אחת, שתיים או מאה, בסוף גם אתה תדע לרכב. חוץ מזה, גם אם תיפול, תמיד תוכל לקום ולהמשיך לרכב שוב - לא פלא שילדים היום לומדים לתכנת בעצמם בגיל צעיר ביותר.

אבל מספיק עם הקיטורים! אוטומציה כיום היא הרבה יותר ממה שהיכרנו לפני 5 או 10 שנים. בעבר כלים אוטומטיים התמקדו בשתי חזיתות מאוד מוגדרות - בדיקות רגרסיית GUI ובדיקות עומסים. בנוסף, מספר הכלים היה אז מצומצם ויקר למדי (אפילו אם היו כמה כלים חינמיים, הם היו מאוד מסובכים ועבדו רק בסביבות וטכנולוגיות מאוד ספציפיות). היום אוטומציה היא בעצם שם קוד למגוון רחב של כלים וטכנולוגיות שבאות לעזור לנו במספר תחומים: ביצוע פעולות מסוימות שחוזרות על עצמן הרבה (כמו בדיקות רגרסיה או Continuos Integration), פעולות בסיסיות שקשה לבדוק באופן ידני (כמו בדיקות API) וכן פעילויות מורכבות שגוזלות זמן רב (כמו בניית סביבת בדיקות, כולל התקנות וקונפיגורצות של כל הרכיבים). כיום, אוטומציה מתבססת בין היתר על סט מגוון מאוד של כלים, שחלק גדול מהם בחינם! המשמעות היא שניתן ליהנות מיתרונות האוטומציה ביותר פעילויות ותחומים וביותר ארגונים. אותם ארגונים שלפני מספר שנים נמנעו מאוטומציה מכיוון שלא היה להם את התקציב לאותם הכלים. כיום זו באמת שאלה של יוזמה אישית והחלטה "לקפוץ למים".

מבלי להתחרות עם מספר כתבות סביב הנושא המופיעות בעמודים הבאים, אני רוצה לתת עצה אישית אחת לכל מי שחושב להתחיל פרויקט אוטומציה או כבר עובד על זה ורוצה לשפר את תוצאות עבודתו. אוטומציה היא עבודת פיתוח תוכנה לכל דבר. עליכם לגשת לאוטומציה תוך כדי שאתם מקפידים על אותם כללי פיתוח של החברה שלכם. כולל ביצוע תכנון נכון, עבודה עם כלי בקרת תצורה, תהליך נכון של בדיקות האוטומציה עצמן וניהול מסודר של התקלות שעולות. בנוסף, עליכם להתייחס לקשר ההדוק שבין האוטומציה למוצר הנבדק ולערב את צוות הפיתוח בתהליך ההגדרה וכתיבת האוטומציה. המפתחים, להם מטבע הדברים יש יותר נסיון בכתיבת קוד, יוכלו לעזור לכם ולתת עצות על איך לגשת, לתכנון ולכתוב את הפיתרון. בנוסף, עירוב המפתחים יוביל ליתרון נוסף כאשר אותם מפתחים גם יקחו בחשבון את האוטומציה לפני ותוך כדי שהם משנים שורת קוד במוצר. כך הם ימנעו מלשבור את האוטומציה עם כל גרסה חדשה של המוצר (או לפחות ידעו לכוון את השינוים שלהם כך הנזק יהיה מינימלי).

חוץ מזה, (וללא קשר לאוטומציה) יש בגליון זה מספר כתבות נוספות מאוד מעניינות כגון המאמר של שמואל גרשון על שימוש ב"פרסונות" או פרופילים לצורך תכנון והרצת הבדיקות, המאמר של אפרת וינברג על קביעת חומרת הבגים וכן הכתבה על פרויקט טק-קריירה המשלב יוצאי אתיופיה בעבודה בהיי-טק, בעיקר בבדיקות תוכנה.

בנוסף, רציתי לספר לכם על יוזמה המתגלגלת במסדרונות חושבים בדיקות בזמן האחרון. מדובר באתר חדש המיועד לבודקים המחפשים עבודה ולחברות המחפשות בודקים. האתר החדש, ששמו QAJobs.co.il, יעלה לאוויר בזמן הקרוב והוא מיועד לתת מענה יעודי לשוק בדיקות התכנה בישראל. מטרתו לספק את הפלטפורמה הטובה ביותר, הן לבודקים חדשים והן לבודקים וותיקים, במציאת האתגר המקצועי הבא שלהם.

אני מסיים בהזמנה לכולכם ליצור איתנו קשר דרך האתר עם רעיונות, שאלות או אפילו מאמרים שכתבתם או שאתם רוצים לכתוב. אנו נשמח לעזור לכם להפוך לחלק פעיל ותורם מקהילת הבודקים בישראל.

יואל

More to come
in the near future
QAJobs.co.il

קרוב יותר
טוב יותר
זול יותר
לא צריך לטוס לחו"ל
כדי למצוא מציקות
ב- Low cost
פרויקט מרגליות של
קוולישטס מעסיק מעל 100
בודקות ומציע את המקצועיות
הגבוהה של קוולישטס במודל
LOW COST מקומי
QUALITEST
software testing experts
למידע נוסף ותיאום פגישת היכרות:
Margaliot@QualiTest.co.il

3 סיפורים מהחיים על בדיקות אוטומטיות

להלן 3 סיפורים המבוססים כולם על ניסיון אמיתי של חברי המערכת מיישום בדיקות אוטומטיות בארגונים שונים. כל שמות הארגונים בדויים כמו גם שמות האנשים.

אנו משוכנעים שתוכלו להתחבר לפחות לסיפור אחד מהשלושה (אם לא לכולם) ומקווים שתוכלו ליישם את העקרונות המוצגים בסיפורים ולסיכום בסוף המאמר ובכך תוכלו לחסוך לכם ולארגון הרבה כאב ראש בהמשך.

11:55 AM

סיפור ה"סטארט-אפ"

בחברת סטארט-אפ המפתחת מערכות לחיזוי מזג האוויר עובדים כ-50 מפתחים ו-10 אנשי בדיקות. גרסאות יוצאות אחת לרבעון ותיקונים (Patches) יוצאים אחת לשבוע שבועיים.

מנהל הפיתוח, יוסי, החליט שצריך לשדרג את מערך הבדיקות בארגון. הוא קרא אליו לשיחה את אבי, מנהל ה-QA. "החלטתי שאנו צריכים לשדרג את הבדיקות אצלנו. בוא נכניס כלים אוטומטיים לבדיקות" אמר יוסי. אבי כמובן התלהב ושבוע לאחר מכן נקבעה פגישה עם איש המכירות של חברה מובילה בתחום הכלים האוטומטיים. איש המכירות הדגים בפגישה (בנוכחות יוסי ואבי) את נפלאות הכלי. "שימו לב איך אני לוחץ על כפתור ההקלטה, מבצע הבדיקה, מסיים ההקלטה והופ - יש לנו סקריפט אוטומטי". אבי ויוסי שיפשפו עיניהם בתדהמה. "איך לא חשבנו על זה קודם" הם שאלו. מייד הזמינו אבי ויוסי 10 רשיונות ויצאה הנחיה של אבי ש"תוך 3 חודשים אני רוצה שכל הבדיקות ירוצו בצורה אוטומטית".

הרשיונות הותקנות במחשבים של אנשי הצוות וההקלטות התחילו. כבר בסוף השבוע הראשון הצטברו עשרות סקריפטים מוקלטים. אחרי חודשיים נכנס אבי אחוז התרגשות ליוסי ולחדר ודיווח "סיימנו את כל ההקלטות, אנחנו 100% אוטומטיים לקראת הגירסה בשבוע הבא".

הבדיקות האוטומטיות נכתבו ע"י מומחה האוטומציה אשר עשה שימוש נרחב בפונקציות הבסיס.

נמצא כי ברוב הפעמים בהן נכשלה ההרצה התקלה היתה בפונקצית הבסיס כך שתיקון הבעיה ארך דקות בודדות ומייד אפשר היה להמשיך בהרצת הבדיקות.

שבוע לאחר מכן הגיעה הגרסה. כל הבודקים קיבלו הנחיה להריץ את הסקריפטים שהוקלטו. אבי ויוסי (שפינה את כל היום על מנת לראות את הנס בהתהוותו) צפו בהתרחשויות. הבודק הראשון (שמגיע תמיד ראשון לעבודה) התקין את הגרסה החדשה ולחץ על כפתור ההרצה. הסקריפט אכן רץ והצליח אפילו לפתוח את חלון הכניסה למערכת, אבל אז התחיל הבלאגן. שום פעולה לא הצליחה. יוסי תפס את אבי בצד ואמר לו "הבודק הזה לא מכיר את העבודה. בוא נשקול לפטר אותו ונחכה לשאר הצוות שיגיע". אחרי חצי שעה הגיעו שאר אנשי הצוות וגרמו ליוסי ואבי לאכזבה אדירה כשראו שאף לא סקריפט אחד שהוקלט לא רץ בצורה תקינה.

אבי ויוסי החליטו לקרוא למומחה חיצוני. המומחה עבר על כל התרחישים והגיע לשורה של מסקנות, ביניהם:

- יש להגדיר מטרות האוטומציה מראש
- אוטומציה אמורה להיות מבוססת על תשתית נכונה
- יש לקחת בחשבון את אלמנט התחזוקה של הסקריפטים האוטומטיים

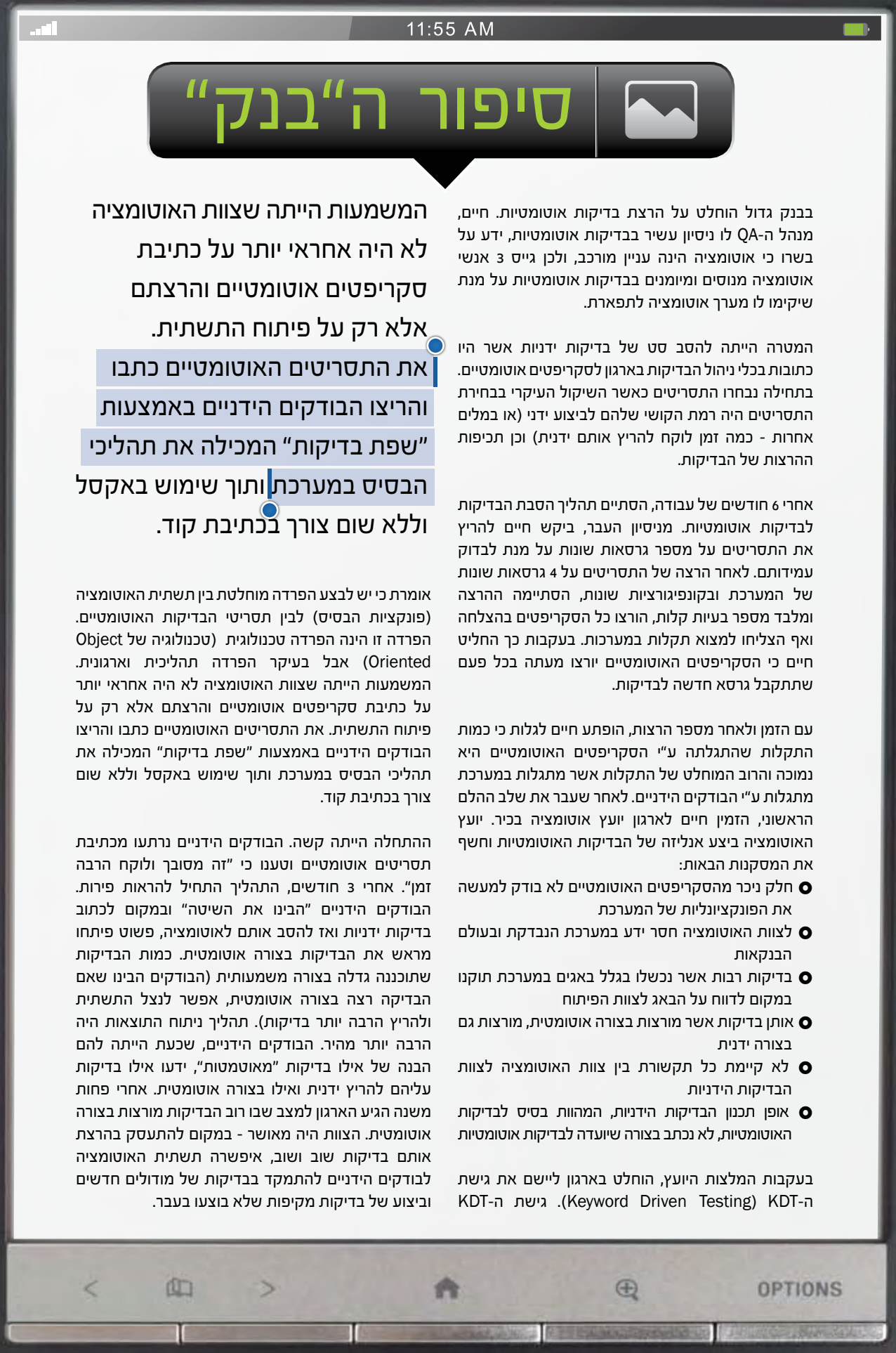
מיד לאחר עבודת האנליזה, גויס לצוות מומחה אוטומטציה אשר לו ניסיון של מספר הטמעות מוצלחות. בשלב ראשון הוגדרה מטרת האוטומציה ל-"הרצה אוטומטית של בדיקות Sanity כל לילה מייד אחרי סיום ה-Build האחרון".

האוטומציה במבנה החדש שלה התבססה על גישת ה-Functional Decomposition. נבנו סט של פונקציות בסיס (כגון כניסה למערכת, יצירת שאילתא לפי מיקום, יציאת שאילתא לפי תאריך וכו'). סה"כ הוגדרו מעל 150 פונקציות בסיס.

הבדיקות האוטומטיות נכתבו ע"י מומחה האוטומציה אשר עשה שימוש נרחב בפונקציות הבסיס. נמצא כי ברוב הפעמים בהן נכשלה ההרצה התקלה היתה בפונקצית הבסיס כך שתיקון הבעיה ארך דקות בודדות ומייד אפשר היה להמשיך בהרצת הבדיקות.

הבדיקות רצו ברמה יומית. בסיום ההרצה הופק דו"ח שנשלח באימייל ליוסי ואבי. תקלות במערכת נמצאו בשלב מוקדם של התהליך והתרומה של הכלי הייתה אדירה לארגון.

OPTIONS



סיפור ה"בנק"

בבנק גדול הוחלט על הרצת בדיקות אוטומטיות. חיים, מנהל ה-QA לו ניסיון עשיר בבדיקות אוטומטיות, ידע על בשרו כי אוטומציה הינה עניין מורכב, ולכן גייס 3 אנשי אוטומציה מנוסים ומיומנים בבדיקות אוטומטיות על מנת שיקימו לו מערך אוטומציה לתפארת.

המטרה הייתה להסב סט של בדיקות ידניות אשר היו כתובות בכלי ניהול הבדיקות בארגון לסקריפטים אוטומטיים. בתחילה נבחרו התסריטים כאשר השיקול העיקרי בבחירת התסריטים היה רמת הקושי שלהם לביצוע ידני (או במלים אחרות - כמה זמן לוקח להריץ אותם ידנית) וכן תכיפות ההרצות של הבדיקות.

אחרי 6 חודשים של עבודה, הסתיים תהליך הסבת הבדיקות לבדיקות אוטומטיות. מניסיון העבר, ביקש חיים להריץ את התסריטים על מספר גרסאות שונות על מנת לבדוק עמידותם. לאחר הרצה של התסריטים על 4 גרסאות שונות של המערכת ובקונפיגורציות שונות, הסתיימה ההרצה ומלבד מספר בעיות קלות, הורצו כל הסקריפטים בהצלחה ואף הצליחו למצוא תקלות במערכות. בעקבות כך החליט חיים כי הסקריפטים האוטומטיים יורצו מעתה בכל פעם שתתקבל גרסא חדשה לבדיקות.

עם הזמן ולאחר מספר הרצות, הופתע חיים לגלות כי כמות התקלות שהתגלתה ע"י הסקריפטים האוטומטיים היא נמוכה והרוב המוחלט של התקלות אשר מתגלות במערכת מתגלות ע"י הבודקים הידניים. לאחר שעבר את שלב ההלם הראשוני, הזמין חיים לארגון יועץ אוטומציה בכיר. יועץ האוטומציה ביצע אנליזה של הבדיקות האוטומטיות וחשף את המסקנות הבאות:

- חלק ניכר מהסקריפטים האוטומטיים לא בודק למעשה את הפונקציונליות של המערכת
- לצוות האוטומציה חסר ידע במערכת הנבדקת ובעולם הבנקאות
- בדיקות רבות אשר נכשלו בגלל באגים במערכת תוקנו במקום לדווח על הבאג לצוות הפיתוח
- אותן בדיקות אשר מורצות בצורה אוטומטית, מורצות גם בצורה ידנית
- לא קיימת כל תקשורת בין צוות האוטומציה לצוות הבדיקות הידניות
- אופן תכנון הבדיקות הידניות, המהוות בסיס לבדיקות האוטומטיות, לא נכתב בצורה שיועדה לבדיקות אוטומטיות

בעקבות המלצות היועץ, הוחלט בארגון ליישם את גישת ה-KDT (Keyword Driven Testing). גישת ה-KDT

המשמעות הייתה שצוות האוטומציה לא היה אחראי יותר על כתיבת סקריפטים אוטומטיים והרצתם אלא רק על פיתוח התשתית.

את התסריטים האוטומטיים כתבו והריצו הבודקים הידניים באמצעות "שפת בדיקות" המכילה את תהליכי הבסיס במערכת ותוך שימוש באקסל וללא שום צורך בכתיבת קוד.

אומרת כי יש לבצע הפרדה מוחלטת בין תשתית האוטומציה (פונקציות הבסיס) לבין תסריטי הבדיקות האוטומטיים. הפרדה זו הינה הפרדה טכנולוגית (טכנולוגיה של Object Oriented) אבל בעיקר הפרדה תהליכית וארגונית. המשמעות הייתה שצוות האוטומציה לא היה אחראי יותר על כתיבת סקריפטים אוטומטיים והרצתם אלא רק על פיתוח התשתית. את התסריטים האוטומטיים כתבו והריצו הבודקים הידניים באמצעות "שפת בדיקות" המכילה את תהליכי הבסיס במערכת ותוך שימוש באקסל וללא שום צורך בכתיבת קוד.

ההתחלה הייתה קשה. הבודקים הידניים נרתעו מכתיבת תסריטים אוטומטיים וטענו כי "זה מסובך ולוקח הרבה זמן". אחרי 3 חודשים, התהליך התחיל להראות פירות. הבודקים הידניים "הבינו את השיטה" ובמקום לכתוב בדיקות ידניות ואז להסב אותם לאוטומציה, פשוט פיתחו מראש את הבדיקות בצורה אוטומטית. כמות הבדיקות שתוכננה גדלה בצורה משמעותית (הבודקים הבינו שאם הבדיקה רצה בצורה אוטומטית, אפשר לנצל התשתית ולהריץ הרבה יותר בדיקות). תהליך ניתוח התוצאות היה הרבה יותר מהיר. הבודקים הידניים, שכעת הייתה להם הבנה של אילו בדיקות "מאוטמטות", ידעו אילו בדיקות עליהם להריץ ידנית ואילו בצורה אוטומטית. אחרי פחות משנה הגיע הארגון למצב שבו רוב הבדיקות מורצות בצורה אוטומטית. הצוות היה מאושר - במקום להתעסק בהרצת אותם בדיקות שוב ושוב, איפשרה תשתית האוטומציה לבודקים הידניים להתמקד בבדיקות של מודולים חדשים וביצוע של בדיקות מקיפות שלא בוצעו בעבר.

11:55 AM

סיפור "היי-טק"

בחברת היי-טק גדולה ניסו במשך שנים להטמיע תהליך של בדיקות אוטומטיות וללא הצלחה.

הסיפור התחיל כששלומית, מנהלת ה- QA של האירגון ראתה באחד מכנסי ה- QA שני סיפורי הצלחה של בדיקות אוטומטיות, מכאן ועד להחלטה להטמיע בעצמה כלים אוטומטיים, הדרך היתה קצרה, בכדי להכנס למשהו חדש התייעצה עם המנהל הישיר שלה שכמובן הראה נכונות להכנס לתהליך מפני שאישתו שעובדת בחברת תוכנה גדולה סיפרה לו ניסים ונפלאות על הכלים האוטומטיים באירגון, "מה ייצא לי מההשקעה הנ"ל?" שאל המנהל את שלומית, וכמובן התשובה הייתה "חיסכון גדול בכוח אדם, הרצות ליליות, הכפלת כמות הבדיקות ע"י בדיקות אוטומטיות" ו- "מה זה ייעלה לי?" "קניה של כלי והכשרת אנשי הצוות לכתיבת בדיקות אוטומטיות..."

וכך היה, רכשו כלים, שלחו 2 מהנדסי בדיקות לקורס אוטומציה והתחילו בתהליך ההטמעה בארגון... מייד לאחר הרכישה מצאו בחברה שכבר בשלב ההתקנה הם נתקלים בבעיות ולא מצליחים לפתור אותן לבד או בייעוץ וסיוע טלפוני...

היו מספר בעיות בסיסיות שפגעו בתהליך ההטמעה:

- הגדרת ציפיות עם המנהלים לדעת מה החזר ההשקעה (ROI) הנכון מבדיקות אוטומטיות
- הכנת צוות מקצועי לטיפול בהטמעה
- רכישת כלים רק לאחר בדיקה מדוקדקת של המערכת

לאחר שנתיים שפרויקט האוטומציה לא התקדם לשום מקום, החליטו להביא ייעוץ מקצועי שייבחן את כל התהליך מההתחלה ועד הסוף. וכך נראה הפרויקט החדש ותוצאותיו:

- אנליזה של:
- לאיזה מערכות כדאי או לא כדאי לפתח בדיקות אוטומטיות, עבור מערכות שהוגדרו כ"מערכות ירוקות" הוחלט בשלב ראשון שלא "לאטמט" (מה שבנסיונות קודמים הכשיל את הפרוייקט כי ניסו לעבוד על כל המערכות)
- חלוקת התסריטים לאוטומציה לפי Priorit בהתאם לקריטריון של החזר השקעה
- הועבר קורס של חמישה ימים לכל העובדים הרלוונטיים שבעתיד תהיה להם יכולת להתעסק בבדיקות אוטומטיות
- בוצע תהליך התקנות ויישור כל נושא הרשיונות מול חברת הכלים
- הוטמעה מתודולוגיית פיתוח מסודרת בשיטת השכבות
- בוצעה חפיפה מסודרת לאחת העובדות לאחר פיתוח סט שלם של בדיקות Sanity

כל תהליך העבודה לווה ע"י תוכנית עבודה וטבלאות מעקב מסודרות. בוצעה הערכת משאבים בפועל ובוצע ניתוח החזר השקעה לאוטומציה שהתבסס על המודל הבסיסי של "כמה אנחנו משקיעים באוטומציה לעומת כמה זמן היה לוקח לנו להריץ בצורה ידנית את כל הסקריפטים שהורצו בצורה אוטומטית".

התוצאה הראתה כי סט ראשון עובד מוכן אצל הלקוח וכבר מההרצה הראשונה מחזיר ROI. הלקוח מאורגן ומסוגל לקחת שליטה על האוטומציה. האוטומציה פותחה בצורה מקצועית ומתודולוגית, בוצעה אנליזה והופקו דו"חות למקבלי ההחלטות ברמה חודשית, רבעונית ושנתית.

10 המסקנות העיקריות לנושא אוטומציה:

01. הגדר את מטרות האוטומציה ותאם ציפיות מול מקבלי ההחלטות
02. בצע ניתוח ROI ותעדף בדיקות להם החזר השקעה מהיר
03. בצע ניתוח הוכחת יכולת מדוקדק לפני כל מעבר שלב
04. אוטומציה מודולרית אשר נכתבה במספר שכבות (פחות קוד = פחות תחזוקה = אוטמציה עובדת)
05. בחן שימוש ב-KDT - תן לבודקים הידניים לתכנן את הסריפטים האוטומטיים ולאנשי האוטומציה לייצר תשתית
06. סמן את הבדיקות שהוסבו לאוטומציה. אל תריץ את אותן בדיקות גם ידנית
07. בחן טוב את כלי האוטומציה לפני קבלת החלטה. זכור שעלות הכלים הינה חלק קטן בתקציב, אך חלק גדול בהצלחה
08. אוטומציה דורשת מומחיות רבה - שלב בצוות אנשים להם ניסיון של מספר הטמעות אוטומציה
09. השקע בהדרכות לאוטומציה לכל הצוות
10. מדוד את יעילות האוטומציה באופן רציף

ראיון עם מנהל בדיקות

צחי מלכי

שם: צחי מלכי

מצב משפחתי: נשוי +3

מגורים: גבעתיים

תפקיד: דיירקטור

קרירה: בשנת 98 התחלתי לעבוד כאיש בדיקות בחברת וולקטק, החברה הראשונה בארץ שהתעסקה בנושא VoIP.

כעבור 4 שנים, עברתי לעבוד בחברת מרקורי, שם ניהלתי את צוות בדיקות QC. בתקופה זו נחשפתי לעולם המתודולוגי לפיו מתנהלות הבדיקות ברוב הארגונים בארץ כיום.

לאחר 3.5 שנים במרקורי חוויתי נסיון קצר בחברת סטארטאפ בתחום מערכות ההפעלה לטלפונים סלולארים.

ב-6 שנים האחרונות אני נמצא בחברת ורינט. התחלתי כמנהל קבוצת הבדיקות בחטיבה האזרחית ובשנה האחרונה עברתי לנהל את הבדיקות בחטיבה הבטחונית.

1. איך הגעת לתחום הבדיקות?

הגעתי לתחום הבדיקות ממש במקרה. בתקופה ההיא, הייתי פרילנסר והתעסקתי בפיתוח על פי דרישה למכרים וחברים. מכרה שלי שעבדה בחברת וולקטק הציעה לי תפקיד שם בתחום הבדיקות וכך נכנסתי לעולם אבטחת האיכות לראשונה.

2. אילו נושאים נוספים בארגון תחת אחריותך?

תחת תחום אחריותי נמצאים תקציבים, בקרת פרויקטים, הגדרות יעדים ומדדים, אבטחת איכות, תקנים ו-ISO, כח אדם, מתודולוגיות ותהליכים חוצי ארגון.

3. באילו בעיות ואתגרים נתקלת במהלך עבודתך?

כמו בחברות רבות, אנו עובדים במטווה פרוייקטלי. האתגר הגדול מולו אנו ניצבים הוא הנסיון לספק את התוכן הרצוי, באיכות הגבוהה ביותר ובהתאם לדרישות הלקוח, וכל זאת תוך עמידה בלו"ז ועם תקציב ומשאבים מוגדרים.

בנוסף, אני מאמין שקבוצת בדיקות נמדדת על ארבה בסיסים: מקצוענות, יעילות, חדשנות ושירות. השאיפה לטווח ארוך היא להעלות את הסטנדרטים בארבעת הבסיסים האלו. המאמץ לעמוד בשאיפה זו, תוך עמידה ביעדים השוטפים – מהווה אתגר לא פשוט.

4. מהם הדברים שמבחינה מקצועית הכי מתסכלים אותך במהלך עבודתך?

כמו שכולנו מכירים, בתחום ישנם הרבה דברים מתסכלים. מתוכם, יש שניים עיקריים שמפריעים לי במיוחד. הראשון הוא ההתנגשות בין השאיפה לדחוף קדימה בעולמות החדשנות והאיכות ובין עולמות הלו"ז והמציאות. לפעמים אנו לא מספיקים לממש פוטנציאל או רעיונות חדשים וזאת בגלל הצורך להספיק ולעמוד בקצב של העולם המהיר בו אנו חיים והמשאבים המוגבלים שלרשותנו.

השני הוא כאשר מתגלים דברים בשטח שיכולנו לגלות לפני. כאשר זה קורה, אנו משקיעים המון זמן בלנתח מדוע זה קרה. בין אם זה סביבות עבודה שונות, מקרי בדיקה שלא היו כתובים נכון, חוסר הבנה של המוצר או פערי ידע של הבודקים. התסכול נובע מכך שיכולנו לעצור את זה לפני אבל לא השכלנו לעשות את זה.

5. מה דעתך על תחום הבדיקות בארץ יחסית למה שאתה מכיר שקורה בתחום בחו"ל?

בארץ, ה-QA עדיין אינו נתפס כמקצוע לחיים אלא כמשהו זמני, כמקפצה בדרך לדבר אחר. בארה"ב ובמקומות רבים

האני מאמין שלי – מקצוענות, חדשנות, התיעלות ושירות הינם ארבעת הערכים שבונים קבוצה איכותית. כמנהל, אני אשאף כל הזמן לאתגר את המנהלים שלי בשיפור מתמיד של עבודתם בעזרת ערכים אלו.

נוספים בעולם, ה-QA נתפס כמקצוע לכל דבר. ישנם אנשי בדיקות שזה תפקידם לאורך כל הקריירה. לרוב האנשים בארץ עדיין חסרה ההבנה ש-QA הוא אחד מהתחומים החשובים והרחבים מבין מקצועות ה-R&D. מלבד אולי ארכיטקטים ותחום השירותים המקצועיים, אנשי ה- QA הינם היחידים שבאמת מכירים ורואים את המערכת מקצה לקצה ולכן החשיבות שלהם צריכה להיות גבוהה משמעותית ממה שמקובל היום בארץ.

שוני נוסף קשור למהירות התגובה. אחד הייתרונות בתחום בארץ הוא הדגש והחשיבות הניתנים למהירות התגובה לשינויים. לפיכך, היכולת להסתגל למצבים חדשים טובה יותר בארץ.

6. איך נראה יום טיפוסי שלך?

היום שלי מתחיל ב-8:30 בבדיקת מיילים, תוך כדי קריאה, אני מנסה לזהות את הנקודות בהן עלי להיות מעורב באופן מיידי. בהמשך היום אני עורך סיבוב עם כל ראשי הצוותים על כל הפרויקטים, מזהה בעיות ולומד היכן הפרוייקטים עומדים.

לקראת סוף היום מתקיימות פגישות עם ממשקים (מנהלי גרסאות, מנהלי פיתוח וכו'), ביצוע סבב נוסף בפרוייקטים והכנת יום המחרת.

7. איך לדעתך יראה תחום הבדיקות בעוד 5-10 שנים? במה הוא יהיה שונה מהיום?

אני מאמין שלעולם הוירטואליזציה המתפתח תיהיה השפעה משמעותית על היכולת להרים סביבות בדיקה.

בנוסף, כיום עולם האוטומציה עדיין לא הצליח להמריא במלואו. אני מאמין שעם הזמן הוא יצבור תאוצה ואנו נלמד להכיר ולהשתמש בטכנולוגיות חדשות בתחום. תחום נוסף שאני מאמין שישתנה הוא מיצוב תחום ה-QA. התחום ימשיך להתעצם יחסית לשאר הממשקים ויתבסס כגורם חזק ומשפיע לאורך חיי המוצר / הפרויקט.

8. האם יש שיטות בדיקה, מתודולוגיות, רעיונות מקוריים או טיפים בתחום הבדיקות עליהם היית ממליץ?

אנו עובדים בהתאם למודל ניהול הפרוייקטים T.O.C. זהו מודל המראה איך צריך להתנהל פרויקט, איך מתחילים פרויקטים רק בידיעה שניתן לסיים בזמן המוקצה, ניהול תיעדוף וסדר יום.

אנו עובדים עם QC בתצורה מלאה, ניהול דרך QC ממגוון ממשקים כגון דרישות לקוח, עולם הבדיקות עולם הריצות וכמובן עולם הבאגים.

בלחיצת כפתור ניתן לראות את השפעת פעילויות שונות על הפרויקט. כך ניתן לנהל בייתר קלות את הפרויקט כמכלול התלוי בגורמים שונים המשפעים זה על זה.

נושא נוסף שאני מאמין בו הוא חדשנות. ישנם כיום לא מעט כלי בדיקה חדשים העוסקים במגוון אספקטים של תחום הבדיקות (תעבורות רשת, סטטיסטיקה, ניהול באגים וכו'). אני מאמין שיש להשקיע את הזמן הדרוש כדי לבחון אותם, להכיר אותם וללמוד כיצד לנצל אותם. אומנם חלקם יתגלו כלא שימושיים או כלא מתאימים, אבל חלקם עשויי לתרום רבות לתהליך בארגון.

רק דרך חקירה ולימוד של שיטות חדשות וכלים חדשים ניתן להתפתח.

9. אילו תכונות אתה מחפש כשאתה מראיין בודק תוכנה?

כשאני מראיין הדבר החשוב ביותר הוא הפוטנציאל, היכולת לתת תשובות לא טריוויליות, היכולת לפרק בעיה לגורמים גם בלי להבין מקצועית את הבעיה וגם ללא נסיון ב-QA. אני מחפש את היכולת להסתכל על הבעיה בחשיבה לוגית. בנוסף, חשובה לי העמידה מול לקוחות, היכולת לרוץ עצמאית ולקחת אחריות, ללוות את הלקוח לאורך כל הדרך עד לאישור המוצר מול הלקוח בסיום התהליך.

10. מה היית ממליץ לבודק תוכנה שנמצא בתחילת הקריירה שלו?

אני מציע להתחיל מחברות גדולות ולאו דווקא חברות ענק. חברות בהן תוכל ללמוד מתודולוגיה מסודרת ועל תהליכי העבודה המקובלים בעולם. סטארטאפ קטן המבוסס על תפקיד בו אתה עושה הכל (פיתוח, אינגרציה וכו') - לא תורם מספיק להתמקצעות שלך בתור בודק. כדי להיות איש מקצוע - יש להשקיע בתחום אחד וללמוד אותו לעומק. משם תוכל להמשיך לכל כיוון שתרצה. חשוב גם לא לבחור מקום עבודה רק על פי השכר אלא לפי מה מקום העבודה יתן לך מבחינה מקצועית. צריך לחשוב איפה אהיה עוד שנתיים אם אקח את התפקיד הזה.

11. מאיפה אתה לומד על חידושים בתחום הבדיקות? קורא חומרים מקצועיים?

אני לוקח חלק בכנסים, בפורומים באינטרנט בארץ ובעולם, מפגשים עם מנהלי בדיקות בחברות אחרות שנועדו ללמוד ולשתף רעינות, באמצעות רשתות חברתיות הקשורות לבדיקות ודרך חברים קרובים העוסקים בתחום.

12. אם לא הייתי בודק תוכנה היום, באיזה תחום היית?

אם לא הייתי בתחום הבדיקות, אני מאמין שהייתי עוסק בחינוך. הייתי לוקח חלק בהכשרה של דור העתיד.

מודל חישוב חומרה של באג

אפרת וינברג

קונפליקט מוכר ולא חביב בארגונים רבים הינו הוויכוח בין קבוצת הפיתוח לקבוצת הבדיקות לגבי חומרת הבאגים המדווחים. קבוצת הפיתוח תטען שאנשי קבוצת הבדיקות "מחמירים מדי - ידם קלה על הדק ה-critical", קבוצת הבדיקות תטען שאנשי קבוצת הפיתוח "לא נותנים את ההתייחסות ההולמת לבעיה". אנו פיתחנו בצוות הבדיקות מודל קביעת חומרתו של באג, המבוסס על הערכות של מספר ממדים וחישוב סופי של חומרה. גילינו שהשימוש במודל זה דרש מהבודקים חשיבה והבנה עמוקה של המערכת, וכן הביא לדיווחים מדויקים יותר ולהפחתה בקונפליקטים בתוך הארגון.

אפרת וינברג
אפרת בעלת ניסיון של כ-14 שנה בבדיקות תוכנה. החלה את דרכה המקצועית כבודקת ובהמשך עברה לתפקידים ניהוליים כראש צוות בדיקות, מנהלת מחלקת בדיקות וכתיבה טכנית ומנהלת פרויקטים בבדיקות.

מפתחת ומרצה קורסים מגוונים בנושאי בדיקות תוכנה, ניהול פרויקטים ותכנות במכללות שונות.

בוגרת הטכניון BSc בכלכלה וניהול ומוסמכת ISTQB ברמת Foundation וברמת Advanced-Management Test.

efratwein@gmail.com

להלן דיאלוג דמיוני, מתוך ישיבת צוות דמיונית:

(לרוב, נציג הפיתוח) "לא ייתכן שאחוז גבוה כל כך של באגים הוא קריטי בנקודת הזמן הזו. באג שדורש תיקון קל ומהיר אינו יכול להיחשב כבאג קריטי. אנשי הבדיקות צריכים לעדכן את החומרה לבאגים"

(לרוב, נציג הבדיקות) "חומרת הבאג גבוהה משתי סיבות: חלק מהבאגים יגרום לדחיית המערכת על ידי המשתמשים. חלק מהבאגים חוסמים את הפונקציונליות ומעכבים את התקדמות הבדיקות. החומרה גבוהה כי כך או כך חשוב לנו לקבל מאנשי הפיתוח תיקון מהיר לבאגים האלה"

מה בעצם הבעיה? גם לפיתוח וגם לבדיקות אותו אינטרס - להוציא מערכת מוצלחת, איכותית ובזמן המתוכנן. בתיאור הדמיוני להלן, גם אנשי קבוצת הפיתוח וגם אנשי קבוצת הבדיקות מחמיצים אבחנה חשובה: אבחנה בין חומרת הבאג, severity , לבין עדיפות הבאג, priority.

מהם חומרה ועדיפות?

חומרה ועדיפות הינם שני ממדים הקשורים לתיאור הבאג, וביחד מבטאים את מידת הסיכון של באג זה על פרויקט פיתוח התוכנה כולו.

חומרה הינה ההערכה על מידת הפגיעה בתפקוד הפונקציונלי של המערכת עקב קיום הבאג המדווח, ותקבע בדרך כלל על ידי הבודקים.

עדיפות הינה ההערכה על מידת הפגיעה בהצלחת פרויקט פיתוח המערכת מבחינה עסקית- ניהולית, ותקבע בדרך כלל על ידי נציגי הנהלת הפרויקט או על ידי התייעצות מספר בעלי עניין בפרויקט פיתוח המערכת. לעיתים יופעל נושא ההתייחסות לעדיפות רק בשלביו האחרונים של פרויקט הפיתוח, לקראת סוף תקופת הבדיקות.

לרוב חומרה ועדיפות יהיו בעלי ערכים דומים, אך לא תמיד. להלן דוגמא להבדל בין חומרה לעדיפות: שערו בנפשכם באג (מופרך למדי...) על כך שמסכי התיעוד של המערכת שלכם מכילים טעות באיות כתובת אתר הבית של הארגון...

מבחינה טכנית - אין כל השפעה על תפקודה הפונקציונלי של המערכת. מבחינה עסקית, החברה עלולה להחמיץ לקוחות פוטנציאליים שינסו לאתר את הארגון לפי כתובת לא נכונה, שלא לדבר על הנזק התדמיתי שנגרם... ולכן באג זה יהיה בחומרה נמוכה, אך בעדיפות גבוהה.

מדוע אנו מדרגים את הבאגים גם מבחינת חומרה וגם מבחינת עדיפות?

הסיבה לכך הינה סיבה ניהולית.

בעולם מושלם, בדיקות המערכת מסתיימות בזמן וללא באגים פתוחים... בעולם האמיתי, אילוצי זמן, תקציב, זמינות משאבים, דרישות שונות של לקוחות ואפילו סיבות פנים ארגוניות ופוליטיות יאלצו את מנהלי הפרויקט לתעדף את המשימות - להחליט מה יטופל ומה לא, ומתוך מה שיטופל - מה יטופל קודם.

חומרה גבוהה לבדה אינה ערובה לכך שהבאג יתוקן מידית; הנהלת הפרויקט יכולה, למשל, להחליט על אי שחרור חלק מסוים של המוצר הנבדק או לשנות את אפיון החלק הנבדק לחלוטין. לעיתים הנהלת הפרויקט תתעדף תיקון באג בחומרה נמוכה שנמצא ביישום שעתיד להגיע למשתמשים רבים על פני תיקון באג בחומרה גבוהה יותר שנמצא ביישום נדיר יותר לשימוש.

קביעת חומרה של באג

בארגונים רבים, קביעת חומרת הבאג נעשית לפי שיקול דעתו של הבודק, לפי ניסיונו בבדיקות המוצר והידע האפליקטיבי שצבר.

מצב זה עלול לטמון בחובו מספר נקודות פתוחות.

- עמימות - מהו ההבדל בין showstopper, critical, high? מהו הגבול בין חומרה high לחומרה critical? או בין high ל- medium?
- סולם מדידה - האמנם ניתן לכמת ולקבוע מהי מערכת יציבה יותר - זו שיש בה הרבה בגים בחומרת medium או זו שיש בה מעט בגים בחומרת high?

למשל, מהו ההבדל בין מערכת בה 80% מהבאגים הם בחומרה medium ו- 20% בחומרה high לבין מערכת בה 70% מהבאגים הם בחומרה medium ו- 30% בחומרה high? האם ניתן להגדיר כמה בגים מחומרת medium שקולים לבג אחד ברמת high (או כל צירוף חומרות אחר, לצורך הדיון)?

- אובייקטיביות הדיווח - האמנם שני בודקים שונים, בעלי ידע שונה, רקע שונה וניסיון שונה, יעריכו חומרת באגים בצורה זהה? ממה מושפע אופן הדיווח?

ומכאן, האם תמיד נוכל להסתמך על דיווח החומרה כחד משמעי ומוחלט? שאלות רבות, ולא תמיד יש תשובה אחת נכונה.

ההשלכות של דיווח חומרה שגוי

דיווח שגוי - עלול להביא לקבלת החלטות שגויה.

חומרה שדווחה נמוכה מהחומרה האמתית - עלולה לגרום למצב בו באג שהוחלט לא לטפל בו מתגלה כמשמעותי ביותר לאחר סיום הבדיקות - נזק לפרויקט ולמשתמשים.

חומרה שדווחה גבוהה מהחומרה האמתית - עלולה לגרום למצב בזבוז משאבי פרויקט יקרים על חשבון נושאים חשובים יותר. נוסף לכך, ריבוי אזעקות שווא עלול לגרום לאי אמון בארגון ופגיעה בתהליכי העבודה השוטפים בהמשך.

מכאן ניתן להסיק, שאנו שואפים לסולם אחיד, ברור שיבטא נכונה את חומרתו של הבאג ויהווה בסיס להערכת סיכונים כוללת על מצב המערכת הנבדקת ומכאן קבלת החלטות על המשך ניהול הפרויקט.

שיטות שונות להתמודדות עם הערכות שגויות

ארגונים רבים מכירים בבעייתיות דיווח החומרה שנידונה קודם, ונוקטים שיטות שונות להתמודדות עם הבעיה. להלן מספר דוגמאות:

- חפיפה: בודק חדש מדווח באגים ובפרט חומרה תוך הנחיית בודק ותיק. יתרונות שיטה זו - אחידות רבה יותר בדיווחים וכן חפיפה טובה יותר לבודק לא מנוסה. החיסרון, כמובן, הינו צריכת הזמן והעבודה הכפולה על כל דיווח באג.
- אישור נוסף: הארכת מחזור חיי הבאג על ידי הוספת גורם נוסף אשר יבחן ויאשר את החומרה טרם העברת הבאג לטיפול. היתרון דומה לשיטה הקודמת, החיסרון הוא היווצרות "צוואר בקבוק" ועיכוב במחזור חיי הבאג.
- מתודולוגיה: הגדרת "כללי אצבע" לדיווח חומרת באגים, כגון: "כל תעופה במערכת הינה באג קריטי, כל 'שגיעט קטיו' הינה באג מינורי". היתרון הינו דיווח אחיד, שאינו תלוי בגורמים מעכבים כלשהם. החיסרון הינו שיש הרבה מאוד "יוצאים מהכלל" שעלולים ליפול בקטגוריות השונות. חיסרון נוסף הינו העובדה שבודק שנצמד לכללים פשטניים מדי עלול לא להתעמק במערכת ולהחמיץ אספקטים חשובים והשלכות שונות של גילוי הבאג.

מודל קביעת חומרה לבאג - האתגרים והמטרות

טרם פיתוח המודל, הגדרנו מהם האתגרים העומדים בפנינו, להם אנו מחפשים מענה:

- לפתח מודל אשר מחד ייתן סולם אחיד, פשוט ושלא יעמיס את המערכת בצווארי בקבוק שונים, ומאידך לא ייקבע בודקים לקבלת החלטות בצורה שבלונית וללא מחשבה מצדם.
- דרך נוספת לגרום לבודקים להכיר את המערכת בצורה רחבה, גם מעבר לתחום עבודתם, ביחד עם מגבלות הזמן והעבודה הרבה שלפניהם.
- מודל גנרי אשר יתאים למגוון פרויקטים בארגון ולא רק לפרויקט אחד, בכדי למנוע צורך לעבוד בכפל מתודולוגיות או לפתח לכל פרויקט מתודולוגיה ספציפית.

לפיכך, מטרות מודל קביעת החומרה הוגדרו כדלהלן:

- הגדרת סולם ערכים אחד ואחיד אשר יהיה בסיס יציב ואמין לשם קבלת החלטות על ידי הנהלת הפרויקט.
- יצירת שפה משותפת בין צדדים שונים בארגון, ומתוך כך הפחתת קונפליקטים ביניהם.

מודל חישוב חומרה של באג

- נטרול הטיות סובייקטיביות של הבודק בעת הדיווח.
- גירוי הבודק לעבודה מעמיקה וצבירת ידע אפליקטיבי תוך כדי עבודה.

מודל קביעת חומרת הבאג - היישום

המודל נדרש לספק אחידות וגמישות כאחד. לשם כך, המודל מחולק לשני חלקים - חלק קבוע (שאלות) וחלק גמיש (תשובות וסולם קביעת החומרה הכוללת של הבאג).

בעת קביעת חומרתו של הבאג המדווח, הבודק ישאל עצמו את השאלות הבאות, יבחר את אחת התשובות המוצעות ויסכם את סך הציון שהתקבל. טווח הערכים בו נופל ציון זה בסולם החומרה יקבע למעשה מהי חומרתו של הבאג.

השאלות הן קבועות ברמת כלל הארגון. התשובות לשאלות ומשקלותיהן, וכן סולם החומרה הסופית עשויים להשתנות בין פרויקטי הבדיקות השונים באותו ארגון לפי שקולי הנהלת הפרויקט.

יישום המודל

יישום המודל כולל מספר צעדים שיש לבצע .הגישה המומלצת ליישום הינה בצוע pilot - עבודה בקנה מידה קטן (מדגמי - מייצג) ולאחריו הפקת לקחים והתווית תכנית יישום לכלל הארגון.

- שלב ההגדרות: יש להחליט מהו סולם החומרה שבו ישתמש הארגון. עד כמה פרטנית תהייה החלוקה לחומרה? יש להחליט על סולם תשובות אפשרי לכל שאלה, וניקוד לכל שאלה. יש להגדיר בבירור כל תשובה אפשרית ומהו משקלה בציון החומרה הסופי.
- שלב הגדרת הידע: המודל מחייב את הבודקים ליידע אפליקטיבי מסוים - בייחוד במענה לשאלות 2 ו - 4. יש להגדיר מהי רמת ידע זו, למפות את צרכי הלימוד הנדרשים לצורך כך, ולבצע השלמות ידע לבודקים במידת הצורך.
- שלב קביעת אופן הביצוע אל מול כלי ניהול הבאגים: יש להחליט האם החישוב יתבצע בצורה ידנית על ידי הבודקים, ורק החומרה הסופית תוזן לכלי ניהול הבאגים , או שמשימת החישוב תתבצע אוטומטית בכלי ניהול הבאגים, והבודקים ידינו את כלי ניהול הבאגים בערכי התשובות לשאלות המודל.
- שלב הגדרת ה pilot: מיהם הבודקים שישתתפו ב- pilot ועל איזו פעילות בדיקות יבוצע ה pilot. יש להגדיר מהי תקופת הזמן בה יבוצע ה- pilot.
- שלב הגדרת הבקרה ובחינת ההצלחה: יש להחליט מהי ההגדרה להצלחה בשימוש במודל זה לעומת עבודה בדרכים אחרות, ולהגדיר מדדים ודרכי איסוף הנתונים כדי לברר האם הושגה הצלחה בצורת העבודה עם המודל. יש לבחון את יחס העלות - תועלת המופק כתוצאה משימוש במודל. העלות הינה מאמץ הבודקים ותוספת הזמן הנדרשת מהם בעת קביעת החומרה. התועלת הינה רמת הדיוק בחומרה המושגת, וכן שיפור הידע האפליקטיבי של הבודקים. תועלת נוספת עשויה להיות שחרור מסבבים רבים של בקרה ועדכונים בשדות שונים של טופס דיווח הבאג, לאחר דיווחו.

בסיום ה- pilot התוצאות מנותחות כדי להסיק מסקנות ולשפר את המודל לקראת סבב הבדיקות הבא והיישום בארגון כולו - בין אם לשנות את משקלות התשובות, לשנות את התשובות, את סולם סיכום ציון החומרה ואולי אפילו להוסיף, לעדכן ולהרחיב את החלק הקבוע של השאלות.

סיכום

כשם שאין פרויקט בדיקות אחד דומה למשנהו, וגם אין ארגון אחד דומה למשנהו, אין מודל אחד שיתאים לכל ארגון ולכל פרויקט.

מודל זה מספק שיטה מובנית לקביעת חומרה מחד, ועשוי לספק אפשרויות לאיסוף וניתוח נתונים בצורה מדויקת, ומאידך עלול להתגלות כצורך משאבים, זמן ושינוי בנהלי עבודה מוכרים ברמה זו או אחרת.

טרם ההחלטה על אימוץ מודל זו או אחר יש לזהות את צרכי הארגון ולהחליט האם המודל אכן ייתן מענה לאותם צרכים, וכן להעריך האם המאמץ שיידרש בעת ההקמה וההטמעה עתיד להשתלם בעת ההפעלה.

מודל קביעת חומרת הבאג - היישום

שאלה	תשובות אפשריות	ציון לתשובה	התשובה הרלבנטית לבאג שנתגלה
1. מה קורה?	כשל תפעולי: קריסת מערכת/סביבה/מחשב	5	
	כשל אלגוריתמי: המערכת מנפקת תוצאה לא נכונה או סותרת את הדרישות	5	
	כשל טיפול בנתונים: נתונים מושחתים, אבדו או לא נשמרו	4	
	חוסר בתיעוד או חיווי	3	
	תיעוד או חיווי שגוי	3	
	אזהרות גלויות	2	
	אזהרות סמויות (כגון אזהרות ב- log)	1	
2. איפה זה קורה?	במסלול הפעילות הפונקציונלי הראשי של המערכת	5	
	במסלול פעילות תומך פונקציונליות ראשית.	4	
	באחת האופציות של מודול פונקציונלי מסוים	3	
	במסך, טופס או באנר	2	
	אחר	1	
3. מתי זה קורה?	תמיד	5	
	עבור setting ספציפי (וכאן ניתן לדרג את ספציפיות אותו setting)	4	
	עבור data מסוים. (וכאן ניתן לדרג את ספציפיות אותו data)	4	
	עבור הרשאות מסוימות	3	
	עבור תנאי סביבה ספציפיים	2	
	מקרה נקודתי יותר	1	
4. האם ניתן לעקוף את הבעיה workaround?	לא ניתן לעקוף את הבעיה בשום דרך	5	
	פעולה הדורשת אתחול או כניסה מחדש	4	
	פעולה הדורשת חזרה על הפעולות הקודמות	3	
	פעולה שאינה אינטואיטיבית, הדורשת הנחייה	2	
	בצורה מידית/אינטואיטיבית	1	
סך הכל - סיכום כל ציוני התשובות			

סולם חומרה אפשרי:

- ציונים 15-20: חומרה **high**
- ציונים 9-14: חומרה **medium**
- ציונים 4-8: חומרה **low**

מה ה-FUNTASY שלך?

FUNTASY
Functional Test Automation System

כלי בדיקות אוטומטיות ייחודי שיגשים לך את כל הפנטזיות*

*בתחום בדיקות האוטומציה בלבד...

יושב מעל רוב כלי האוטומציה כגון QTP, Test Complete ועוד

מאפשר לבודקים ידניים לתכנן ולהריץ בדיקות אוטומטיות

מאפשר כתיבת בדיקות אוטומטיות לפני שהמערכת מוכנה

משלב בדיקות Non-GUI כגון הרצת API

QUALITEST
software testing experts

למידע נוסף ותיאום פגישת היכרות:
www.QualiTestGroup.com/FUNTASY FUNTASY@QualiTestGoup.com

המלצה על בלוג -

עולם בדיקות התוכנה וה-QA של דורון בר

http://www.qapro.org

דורון בר פעיל בתחום הבדיקות מאז 1998 ויש לו נסיון רב בתחום ממספר חברות. באתר שלו ניתן למצוא מגוון רחב של חומרים בתחום הבדיקות: החל מטקסטים מקצועיים ושימושיים, דרך המלצות על אתרי ופורומים בתחום ועד לפוסטים מעניינים. אפילו הפניה לאתר המאגד שאלות ותשובות לראיונות בתחום הבדיקות. החומרים כוללים מידע רב ושימושי כגון ראשי תיבות ומושגים, סוגי בדיקות, הגדרת בדיקת תוכנה, מסמכי בדיקות, צ'ק ליסט לכתיבת מסמך בדיקות ועוד. הפוסטים כוללים סיפורים אישים מקצועיים, מקרים בהם נתקל ותובנות מעבודתו ומניסיונו. הכל זמין, נגיש ובאופן די נדיר בתחום שלנו – בעברית.

להלן 3 פוסטים מעניינים במיוחד מהבלוג:

מה קורה כשלא ברור לאף אחד מאין לעזאזל נובע הבאג?

(http://www.qapro.org/2010/05/blog-post.html)

פוסט המדבר על סביבה של מערכת מורכבת בעלת שרתים רבים ובה נופל שרת אחד או קומפוננטה ספציפית ולא ניתן לשחזר את התקלה. הפוסט מתאר מצב קיצוני בו בודקים טובים ומקצועיים פשוט עומדים ומסתכלים אחד על השני בחוסר אונים. הפוסט מציע מספר פעולות גנריות ושלבים שכדאי לנסות לבצע כדי למצוא את הבאג.

הבאג ה-32, או: איך לבודד באג

(http://www.qapro.org/2010/07/32.html)

פוסט הנותן "סיפור מהחיים" אודות מקרה בו נוצר באג אחרי שימוש ממושך במערכת הנבדקת אך היה מאוד קשה לבודד אותו ולהצביע על הגורם שלו. לאחר מספר נסיונות, נמצאה דרך לבצע את הבדיקות באמצעות כלי פשוט ביותר של מאקרו. הוקלטה פעולה אחת והורצה באופן אוטומאטי שוב ושוב. לאחר כמה סבבים כאלו הסתבר שהמערכת מסוגלת לבצע 31 פעם את הפעולה כראוי, אבל מרסקת את המערכת בפעם ה-32.

חשיבותן של מדידות

(http://www.qapro.org/2010/10/blog-post_18.html)

סיפור נוסף מהחיים המספר על תהליך בדיקות של מערכת מורכבת במיוחד בה כל Feature נוגע ב- Feature אחר, לכן כל תיקון באג באזור מסוים הביא לבעיה ב-Feature אחר. כל זה תרם כמובן לאחוז מאוד גבוה של Reopens (אחוז הפתיחה מחדש של באגים). הפתרון עליו מספר הפוסט קשור להעברת האחריות על אחוז ה- Reopens למפתחים שנמדדו ותוגמלו בהתאם לאחוז ה-Reopens הנמוך אותו הם הצליחו להעמיד. הירידה בממוצע ה- Reopens בחברה לא איחר להגיע.

TMap Next, for result-driven testing

by Tim Koomen; Leo van der Aalst; Bart Broekman; Michiel Vroon

TMap הינה מתודולוגיית בדיקות שפותחה לראשונה בשנות ה-90 ע"י חברת Sogeti ההולנדית. המתודולוגיה משלבת תובנות לגבי מה לבדוק ואיך לנהל את הבדיקות עם טכניקות מעשיות לבודק האינדיבידואלי. גרסה זו, שיצאה בשנות ה-90, התמקדה בעיקר בהצדקת הצורך הכללי בבדיקות, קביעת הבדיקות כתהליך נפרד מהפיתוח והדגשת הצורך בתהליך בדיקות מובנה. מאז עברו כמה שנים והזמנים השתנו. חברות התחילו לקחת את תחום הבדיקות ברצינות גבוהה יותר והצורך בבדיקות כדיציפלינה בפני עצמה מושרש עמוק ביותר ויותר חברות. בעקבות זאת, ב-2006 פורסמה גרסה חדשה שיצאה בספר TMap Next, for result-driven testing. הגרסה החדשה, הנקראית TMap Next (מפתיע...), מתייחסת לחשיבות הבדיקות כמובנת מאילה ולוקחת את הבדיקות צעד אחד קדימה. מטרתו העיקרית של הספר היא תיאור הבדיקות תוך מתן דגש על תהליך הבדיקות עצמו וכן התמקדות במטרות העסקיות כגורם המוביל את התהליך. הספר מספק כלים למומחי בדיקות לאזן בצורה טובה יותר בין עלויות הבדיקה לבין התועלות שניתן להפיק מביצוע בדיקות ולכן מקל על מקבלי החלטות בכירים לבחור איך וכמה בדיקות לבצע.

לגישת ה TMap ארבעה יסודות עקריים:

1. TMap הינה גישת בדיקות המונחית ע"י התהליך העסקי
2. TMap מתארת תהליכי בדיקה מובנים
3. TMap מכילה ארגז כלים כולל
4. TMap הינה שיטה בדיקות אדפטיבית

באמצעות מעל 400 דוגמאות וטיפים, הספר מתאר גישה מובנית ומוסדרת לבנייה, ביצוע ותחזוק תהליך בדיקות. בנוסף, הספר מכסה פיתוחים חדשים בעולם הבדיקות כגון ארגוני בדיקות יעודיים, מיקור חוץ של תהליך הבדיקות, סביבות בדיקה וכלי בדיקות. פעילויות הבדיקה מתוארות בפירוט כזה, שיכול לספק לבודקים כלים פרקטיים לשימוש יומיומי. בנוסף, הספר מציע דרכים לבעלי עניין להשפיע על תהליך הבדיקות וזאת בשונה מרוב הספרים בתחום.

הסופרים רתמו את נסיונם הרב לתוך מדריך שיכול לשמש הן צוותי בדיקות לא מנוסים באימוץ מתודולוגיה שימושית והן צוותים מנוסים הרוצים להבין כיצד לנצל גישת בדיקות העוזרת לתעדף משימות ולקבוע כיצד להתשמש במשאבים כאשר הדד ליין לסיום הפרוייקט דוחק (בסביבות ה-100% מהפרויקטים...).

למה כדאי לקרוא את הספר?

- הספר הוא אחד מאבני היסוד של תחום הבדיקות בעולם
- קל לקריאה
- מספק פתרונות פרקטיים לבעיות נפוצות בתחום הבדיקות
- מכיל מתודולוגיה מובנית לבדיקות אשר מוטמעת בקרב אלפי ארגונים ברחבי העולם
- נכתב ע"י מומחים בעלי עשרות שנות נסיון

ניתן להשיג את הספר כאן:
http://www.amazon.com/exec/obidos/ISBN=9072194802

כיצד לבחון כלי אוטומטי לארגון?

רונית שמאי
רונית שמאי, מנהלת תחום בדיקות אוטומציה ועומסים בחברת טסקום ומאמנת אישית.

שלום, שמי רונית שמאי ואני למעלה מעשור בתחום הבדיקות האוטומטיות וראיתי לא מעט אירגונים שנכוו ממה שאנו מכנים פרוייקט אוטומציה. אני משוכנעת שחלקכם לפחות מבינים למה אני מתכוונת...

אכן זה פרוייקט והוא יכול לחולל דברים מופלאים כמו לקצר זמנים למצוא באגים בנקודות אפלות, להוריד מאנשי הבדיקות הידניות את התהליכים הסיזיפיים והשיגרתיים לבדיקת עומק איכותית יותר ברכיבים חדשים, יחד עם זאת בצעדים שגויים זה גם יכול להיות פיל לבן.

אם כך מהם הצעדים לבחינת כלי אוטומציה?
פרוייקט אוטומציה מכיל שלבים רבים, החל מ-כיצד ניתן לקבל תמיכה מההנהלה למהלך זה, כיצד ומה יש לתכנן בפרוייקט האוטומציה, בחינת כדאיות (אופן החישוב), כיצד לבחור את הכלי המתאים לארגון? בניית תוכנית פרוייקט אוטומציה, מי הם הגורמים שלוקחים חלק בתהליך, מהו פרופיל מפתח בדיקות אוטומציה, הבנת התהליכים העיסקיים הנכונים למיכון בניית מסמך תיאום ציפיות, הטמעת מתודולוגיית בדיקות אוטומציה, תיעוד פרוייקט אוטומציה, Code Review, ועד לשלב התחזוקה והבקרה השוטפת.

בכתבה זו אני רוצה לשתף אתכם בתהליך בחינת כלי אוטומטי חדש לארגון.

אם כך כיצד עלי לבחון כלי אוטומטי חדש?

כלי בדיקות אוטומציה רבים צצים חדשות לבקרים, האם האומדן היחידי הינו עלות הכלי?

מנהלים לרוב חותרים להוזלה בהוצאות, כיצד אני יכול לשכנע מנהל לרכוש כלי לבדיקות אוטומציה גם אם הוא מעבר לתקציב? אילו שאלות אני צריך לשאול בהכרעת בחירת הכלי?

שלב א': בדק בית
חפשו כלים שקיימים בארגון. נתקלתי לא אחת בארגונים שכלל לא ידעו שקיימים כלים בחטיבות/אגפים שונים בארגון, שיכולים לסייע בבדיקות אוטומציה.

שלב שני: כמה?
שרטטו את כלל המערכות בארגון מפו מתוכם את מערכות הליבה, כמתו את מספר תהליכי הריגרסייה, הבינו מהי טכנולוגיית הפיתוח עבור כל אחת מהמערכות הללו.

שלב שלישי: פרמטרים
בבואכם לבדוק התאמת כלי אוטומטי כזה או אחר הבינו איזה מענה הוא נותן עבור הפרמטרים הבאים:

1. תמיכה וסביבת עבודה
   a. תמיכה טכנולוגית
   b. פלטפורה
   c. זיהוי אובייקטים
   d. סביבת עבודה
   e. שפות פיתוח בכלים
   f. קלות הבנת הקוד
   g. התקנה
   h. עד כמה הכלי קל ללמידה עצמית
2. פונקציונליות
   a. ממשק עם כלי ניהול בדיקות
   b. אופן חקירת אובייקט
   c. עבודה עם DB
   d. כיצד בנוי ה- ObjectRepository
   e. Debug
   f. Recovery
   g. האם הכלי תומך ב- DataDrivenTesting
   h. עד כמה מפורט הדוח תוצאות ריצה והאם ניתן לקנפג אותו
   i. האם ה- Syntax Error ברור
   j. האם ה- Help מכיל מענה עשיר הכולל דוגמאות

שלב רביעי: עלויות
הבינו את עלויות הכלי (הכולל את מספר הרשיונות הנדרשים) וברו עם אכן עומד בתקציב שלכם.

שלב חמישי: בדיקת היתכנות
בצעו בדיקת היתכנות עם הכלי שהחלטתם לגביו.

בתום בדיקת ההיתכנות ציינו את: מסקנותיכם וכן מהם הסיכוניים צפויים בשימוש עם הכלי.

שלב שישי: חוות דעת נוספת
התייעצו עם חברות המתמחות בתחום, ייתכן שישנן בעיות או פתרונות שאינכם מודעים אליהם.

זכרו בדיקות אוטומציה נועדו להקל ולא להקשות עליכם. זיכרו שברוב המקרים הן אינן מחליפות כח עבודה ידני אלא מספקות כיסוי רב יותר ויכולת לבצע בדיקות ידניות מעמיקות לפונקציונאליות חדשה. זו אחת השגיאות וההתנגדויות הקיימות לפרויקטי אוטומציה מצד הנהלות. נושא שחשוב להכיר ולהתכונן גם אליו מבעוד מועד.

Support and Workspace

CRITERIA	DESCRIPTION	RFT	QTP	TESTPARTNER	ORACLE ATS
SUPPORTED TECHNOLOGIES	What kind of application the tool can test? Like .Net Mainframe/ Java/J2ee/32bit applications	JAVA, HTML, VB.NET and Windows App (.NET Framework 2.0). In March - April support .NET Framework 3.0	Support application Based on Win32 API ו-MFC. In Addition support: JAVA, .NET, Windows and web forms, sap, oracle, siebel, peoplesofe, visualage smalltall, stingray, web services and terminal emulator applications	Supports applications written in VB, C++, .Web, .NET, Java, SAP, Oracle and many other distributed Windows-based applications. Test Partner has support for SAP 6.20 GUI for Windows that includes object recognition, capture and replay for SAP 6.20 Win GUI controls.	Supports Web, Oracle EBS/Forms, Oracle Fusion/ADF, Adobe Flex, Web Services and Siebel. Both functional and load testing.
PLATFORM	What kind of platform the tool can test on?	Windows XP PRO,Windows 2000 Pro, Windows 2003 Server, Windows Vista, Linux (In Linux you can only program, you can't record). Advantage: support Linux	Windows XP PRO, Windows 2000 Pro, Windows 2003 Server, Windows Vista	Windows XP PRO, Windows 2000 Pro, Windows 2003 Server, Windows Vista	Windows XP, Windows Vista, Windows 2003, Windows 7, Windows 2008 (32-bit and 64-bit versions of these operating systems)
OBJECT RECOGNITION	How well the tool recognize & test objects	It's can be choose for an object -what properties will recognized for it. In additon there is weight to each property. You can choose which proprty will be the most important. (you get Warning in the report if there is no match in all proprties.)	It's can be choose for an object -what properties will recognized for it. In additon there is an option to set the recognition level	Java Active Object Recognition (AOR) Utility configures Java objects for testing. It requires knowledge of Java development. The utility includes core Java class collections, and lets you add custom Helper classes and metadata based on your application and requirements.	No such tool like Object Spy in QTP
WORKSPACE	What workspace has the tool	Advantage: it can be work on 2 workspaces: 1. Eclipse - Open source environment that IBM has adopted. 2. visual studio environment another advantage: you can open two or more tests simultaneously (In QTP you can open only with one test each time)	Workspace that Mercury built.	Have to maintain a file server. Versioning needs to be taken care of. Object repository, tests, function libraries recovery scenarios have to be maintained separately in file directories.	Eclipse-based OpenScript IDE. In addition offers Test Manager (analog QC) and Load Testing interfaces.
COMPUTER LANGUAGE	Which Programming languages use on the tools?	On the Eclipse workspace - you can use JAVA. On the Visual workspace - you can use VB.NET those languages are advantage because those languages are more powerful than VBScript (QTP Programming language.)	You can work only with VBScript. Advantage: VBScript is easy learning language. There isn't to be experience developer to program with VBScript.	Test partner's scripting language is Microsoft Visual Basic for Applications (VBA) complete with intelligence, auto-indent, etc	Tree View together with the Java Code view.
EASILY CODE UNDERSTANDING	How easy to people without Programming background, understand the code	Very hard. Need a lot of comments	Advantage: there is exist a Tree View. In this view people without Programming background, can build a test or understand exist test	TestPartner (6.1) is a Great tool especially if you have a flair for Pragramming.	Tree View together with the Java Code view. With accent on the Tree View usage. Java code is properly formatted and organized.
INSTALLATION	How easy to install the tool?	Easy installation. first installing" "installation manager" and from the "installation manager" - install the RFT (Eclipse or visual mode or both). It's seems to be very complex, but it's easy installation	Easy installation	Easy installation	Installation is easy enough, though it is strongly adviced to read the installation instructions first.
SELF LEARNING	How easy, self-learning the tool?	Multiplicity menus - it's take some time to know well the functionality of this tool	Advantage: The menu is very simple to understand	Record-and-playback scripts a bit easier to read	There is a good tutorial, but as it is with the QTP, it teaches only some basic steps to use the program. A lot has to be learned by experimenting or by reading the User guides. From the first glance, the program is easy to use.

Functionality

CRITERIA	DESCRIPTION	RFT	QTP	TESTPARTNER	ORACLE ATS
RUNNING AUTOMATIC TESTS FROM TEST MANAGER SOFTWARE	Are the Automatic tests can run from test manager software?	They can running from "Test manager" (Old software) or from "Clear Quest" (version 7 and up)	They can running from TD (Old software) or from QC	TestPartner's value is further enhanced through its tight integration with Testing & ASQ solution, which improves application delivery by providing IT with the ability to more effectively assess and balance risk, cost, and schedule. TestPartner is also an integral component within Quality Management, which provides IT with the ability to control, measure, manage and continuously improve the delivery of higher quality applications.	With Oracle Test Manager (part of the Oracle Applications Testing Suite) you can individually run manual tests, Oracle OpenScript functional tests, third party, or even Test Groups.
COMMENTS	Can comments be added while recording?	This option is exist	This option is exist	This option is exist	No
RECOVERY	How easy to deal with Recovery	There is no Wizard. Dealing only with code	Advantage: there is a Wizard	Recovery system with TestPartner is a huge disadvantage since it is a time consuming manual process.	A dialog with OpenScript Error Recovery categories very thoroughly developed.
WORKING WITH DDT		You can work indirect with excel. You have import the data from the excel to the Datapool	Advantage: you can work direct with excel	TestPartner scripts uses Active Data with Excel documents as the underlying data source and to determine if a test script passed or failed we save the test status to the active data used by the testpartner test script	Data Input Parameterization through either CSV (comma delimited), or TXT files.
DP	Ability working with Descriptive Programming	Exist	Exist: example: http://www.advancedqtp.com/knowladge-base/code-techniques/dp-101/		Uses Descriptive Programming to describe objects in the script.
OBJECT ANALYSIS		Exist: There is an option to analyze an object with object inspector ,In addtion the tester can "see" internal object using Find function and other useful function in RFT Help: com.rational.test.ft.object.interfaces.TestObject Advantage: after clicking "."" you can ""see"" internal object (In QTP you can ""see"" only one level of internal object and in RFT you can ""see"" a lot of levels)	Exist: there is an option to analyze an object with object spy ,In addtion the tester can "see" internal object using .object.	TestPartner provides full object recognition, recording, and playback support for testing 32-bit applications running on Windows 7 (32-bit and 64-bit editions) and Windows Server 2008	No possibility for preliminary object analysis.

CRITERIA	DESCRIPTION	RFT	QTP	TESTPARTNER	ORACLE ATS
OR	Is the OR local or Shared?	There is an option to choose for a test from this options(but not both): 1. Local OR 2. Shared OR disadvantage: When the OR is shared - all tester can change it	There is an option to choose for a test that the OR will be Local, Shared Or combination of both advantage 1: there is an option that the OR will be combination of local OR with Shared OR advantage 2: Shared OR is disable	TestPartner scripts, Objects, Forms, variables, modules (assets) are stored in Database that can be shared	There is OR.
REPORT		Advantage: There are 3 king of report types that you can choose: 1. HTML 2. Text file 3. TPTP	There is only one type of report	Apart from the test execution report generated by the TestPartner tool, testers can customize reports in an external spreadsheet format. This provides report details for which test scripts have failed or passed while running a test suite. The reports detail exception cases handled, defects found, and errors logged. These functions are customized in the driver script. This helps in performing effective analysis on the execution report.	Results and Error Log views, that are generated after the script run.
HELP		Advantage: Very plentiful help	Simple help	Simple help	Good. Offers dynamic context help (when pausing the mouse over a method) No programming examples yet though.
DEBUG		Advantage: You can change variable's value during running(in debug mode)	You can not change variable's value	TestPartner provides some of the features of the Microsoft VBA environment, including assistance with debugging and information on runtime variables.	You can use features of the Eclipse IDE to debug scripts.
SYNTAX	How easy to use during finding a syntax error	Advantage: the error messages are written very clearly. In addition there is "Quick Fix" option. The tool show some suggestions to fix the error	Sometime it's very hard to understand from the error message - what is the problem and what need to do to slove it.	VBA	Very useful. 1. Error annotations (red boxes) appear in the overview ruler positioned on the right hand side of the editor, 2. Error icons appear in the vertical ruler positioned on the left of the editor, 3. An error indicator appears in the top right corner of the editor, 4. Errors are marked in the text.
WORKING WITH DB		This option is exist using jdbc programming. For example: http://www.jdbc-tutorial.com/	This option is exist	Test Partner supports multiple users on the following databases: Microsoft SQL Server v7 and SQL Server v 2000, Oracle v8.1.6 or 8.1.7.	Query from any database for which you have anJDBC: ODBC Bridge driver.
AUTO COMPLETE	Is auto complete option exist?	Advantage: This option exist	Exist but very poor	Exist, but not good	No

טק-קריירה
Tech-Career
מרכז הכשרה טכנולוגית ליוצאי אתיופיה בישראל
Technology Training Center for Ethiopian Israelis

טק-קריירה

מרכז הכשרה טכנולוגית ליוצאי אתיופיה בישראל

אודות טק-קריירה

טק-קריירה מציעה לצעירים וצעירות יוצאי אתיופיה תכנית משולבת ייחודית וחדשנית, הכוללת הכשרה טכנולוגית איכותית, סדנאות העשרה ופיתוח אישי ושירותי השמה בעולם ההיי-טק. תכנית זו מעניקה למשתתפים הזדמנות אמיתית לרכוש ידע, ניסיון וכישורים הדרושים על מנת להשתלב בתעשיית ההיי-טק וה-IT בישראל, לפתח קריירה איכותית ולהתקדם לעצמאות כלכלית. טק-קריירה הינה ארגון חברתי מיומן ומקצועי ללא כוונות רווח. הארגון מוכר כ- IT Academy של Microsoft והכשרותיה מוכרות על-ידי משרד התמ"ת. הסטודנטים בטק-קריירה נהנים מתנאי למידה אופטימאליים, הכוללים מלגות מחייה לתקופת הלימודים, מגורים בתנאי פנימייה בקיבוץ נחשון וגישה למעבדות מחשבים חדישות 24 שעות ביממה. מאז הקמתה, הוכשרו בטק-קריירה כ-200 צעירים וצעירות בתחומי תוכנה ותקשורת נתונים. אחוזי ההשמה של הבוגרים בתחומי הכשרתם עומדים על כ-70%.

קורס QA

מסלול QA הינו מסלול מבוקש ומשמעותי בטק-קריירה, העונה לדרישות שוק העבודה ומאפשר לבוגרים השתלבות הולמת בעולם ה-IT. מאז 2008 התקיימו 5 מחזורי לימוד בתחום זה שבוגריהם השתלבו בארגונים מובילים במשק: בנק דיסקונט, טבע, מטריקס, ג'ון ברייס הדרכה, נס-גילון, קוואליטסט, טלדור, ריל קומרס, DSPG, ישראכרט, מנורה, בזק, בנק הפועלים ועוד.

תומר אברה, בוגר קורס QA מחזור 11, טק-קריירה

תומר נולד בשנת 1982 בעיירה קולה שבצפון אתיופיה. בשנת 1988 החלה משפחתו מסע בן שלוש שנים לכיוון ישראל, שהסתיים בעלייתם ארצה במבצע שלמה בשנת 1991. תומר, אז בן 9, למד עברית באולפן בנתניה במשך שנה וחצי, ורק לאחר מכן הצטרף ליתר בני גילו ללימודים בבית הספר. את לימודי התיכון השלים בפנימייה בחיפה, וסיים בציונים גבוהים. בשנת 2000 התגייס לצה"ל ולאחר שחרורו רכש את השכלתו כהנדסאי אלקטרוניקה. במשך תקופה ארוכה חיפש עבודה בתחום אך ללא הצלחה. בשנת 2010 התקבל תומר ללימודי בדיקות תוכנה בטק-קריירה, סיים את לימודיו בהצטיינות והתקבל לעבודה בחברת טאקט במרכז הבדיקות של טבע. על ימיו הראשונים בחברת טבע אמר תומר: "לא היה קל, הייתי מאוד נבוך כשפגשתי את הצוות והמנהלים, ותהיתי איך אשתלב. אולם, כולם נתנו לי תחושה טובה ועזרו לי להבין את הסביבה החדשה. למדתי את התפקיד ואת המערכת לניהול מחסן מהר מאוד והתחלתי לתת תפוקות מעבר לציפיות. הפידבקים מהצוות ומהמנהלים לא אחרו להגיע ועד מהרה התחלתי להרגיש בבית".

נכון לשנת 2011, רק כ-200 בני ובנות הקהילה האתיופית שמונה כ-120,000 איש בישראל, השתלבו בתעשיית ההיי-טק וה-IT. תומר הוא אחד מהם. "טק-קריירה העניקה לי הזדמנות אמיתית להשתלב בחברות כמו מטריקס וטבע, ואני עשיתי הכל כדי להפיק ממנה את המקסימום. אני מקווה ומאמין שעוד רבים יזכו להזדמנות שכזו, יפתחו קריירה מספקת ויממשו את יכולותיהם."

טק-קריירה מגייסת מתנדבים לפעילות ייחודית

מרכז המתנדבים המתפתח של טק-קריירה כולל תחומי התנדבות מגוונים, ביניהם: מתרגלים ומרצים מקצועיים, מרצים אורחים בתחומי העשרה, ארגונים מארחים ועוד. בימים אלו אנו פועלים להרחבת מרכז המתנדבים, תוך דגש מיוחד על תוכנית מנטורים חדשה וייחודית.

מטרת התוכנית הינה לסייע לבוגרי טק-קריירה בהשתלבותם בארגונים עסקיים בתחום ההכשרה הטכנולוגית שרכשו, תוך חיזוק ביטחונם העצמי והעמקת היכרותם עם התעשייה. זאת, באמצעות ליווי אחד-על-אחד למשך 6-10 חודשים, מפגש אחת לשבועיים במקום העבודה של המנטור, לאורך תהליך מציאת עבודה ובשלבי הקליטה בעבודה.

אם את/ה מעוניינים בעשייה משמעותית ומספקת, הצטרפו למאגר המתנדבים שלנו!

מי הם המתנדבים שלנו?

- בעלי רקע, ניסיון, היכרות מעמיקה וקשרים בארגונים עסקיים
- בעלי ניסיון בחניכה, הדרכה וליווי של עובדים, סטודנטים ו/או חניכים
- מזדהים עם מטרות חברתיות
- סובלניים, סבלניים בעלי יכולות הקשבה, הכלה והובלה

ליצירת קשר ולקבלת פרטים נוספים:
איריס אגם כוכבי, מנהלת תחום מתנדבים,
iris@tech-career.org ;08-9278700

התנהלות קבוצות בדיקה בינ"ל מבוזרות בסקראם

קבוצות בדיקה רבות נאלצות להתמודד עם מגוון אתגרים כאשר הן מקבלות על עצמן את שיטות ה-Agile השונות, ובכללן שיטת ה-SCRUM. מספר טקסים לא ברורים, מפגש יומיומי, ספרינטים ועוד אוסף שלם של מושגים שנופל על הבודק הממוצע, ועל מנהלי הבדיקות בדרגים השונים.

אז נניח לרגע שאנחנו כבר מבינים את היתרון (חרף הכאב ראש) שבכל טקס וטקס, שהבנו מה המנהלים מחפשים, ומה מטריד את ה-Scrum Master, וגם הטסטרים, ראשי הצוותים ומנהלי הבדיקות קלטו למה השיטה טובה לנו (אם יש איזורים לא מובנים בנושא הזה, זה כבר מאמר שונה לגמרי). כעת תעמודנה בפנינו קבוצת בעיות מסוג שונה לגמרי:

כיצד נבנים הצוותים וכיצד מתנהלת תקשורת בתוך צוות מבוזר?

כיצד מתנהלים כשחלקים מהקבוצה (הן מהפיתוח והן מהבדיקות) נמצאים במקומות שונים בחו"ל?

מה עושים כשמספר קבוצות בדיקה בודקות כמה מוצרים במקביל?

ועוד ועוד...

נרי לביא

נרי פעיל בתחום התוכנה מזה כ-8 שנים, ובכללן בתחומי תמיכת לקוחות תוכנה, פיתוח ובדיקות. בתפקידו הנוכחי כמנהל בדיקות בקו מוצרים בפרונטליין, נרי מנהל בדיקות הנערכות בארץ ובעולם. לפני כ-3 שנים ניהל את מעבר הבדיקות משיטות עבודה ותיקות לסקראם ואת הקמת צוותי הסקראם המבוזרים במספר יבשות. צוותים אלו עוסקים בבדיקות ידניות ואוטומטיות לאורך כל אחד מהמוצרים בקו.

בוגר מדעי המחשב (BSC) ומנהל עסקים (MBA) במסגרת התכנית למצטיינים באוניברסיטה העברית וסקראם מאסטר מוסמך.

המאמר הבא ייתן דוגמאות כיצד אנחנו (כקבוצת בדיקה) התמודדנו עם בעיות אלו, טעויות שביצענו ושיעורים שלמדנו. כמובן שיש עוד מגוון דרכים להתמודדות עם כל בעיה. היכן שרלוונטי, אשתף אתכם בלבטים שליוו אותנו.

להלן מעט רקע הכרחי על מנת להבין את המציאות אותה אתאר בהמשך:

1. מחלקת הבדיקות מורכבת מכמה צוותים בארץ ובסין, כאשר הפיתוח מחולק בין ישראל, סין ואוקראינה.

2. קו המוצרים עליו מחלקת הבדיקות אחראית כולל 4 מוצרי מדף שונים, כאשר לכל אחד מחזור חיים משלו, יעדים משלו, גרסאות משלו וכן הלאה.

3. בין המוצרים השונים קיימים איזורי חפיפה לא מעטים, כלומר שלא מעט מהנדסים מסוגלים לבחון ביעילות יותר ממוצר אחד.

4. הבדיקות מחולקות ל-3 חלקים:
- בדיקות ידניות
- פיתוח אוטומציה עבור פונקציונליות חדשה
- סימולציות סביבות לקוח במעבדה המיועדת ספציפית לכך (כולל גישה ללקוחות בחו"ל, השגת DATA עדכני וכיו"ב)

5. כיום כלל המהנדסים בקו המוצרים המדובר, מחולקים ל- 5 צוותי scrum. ההיסטוריה מראה שמתקיים Team formation (חלוקה מחדש של המהנדסים לצוותים על מנת להשיג את דרישות ה- backlog ביעילות המירבית) אחת לשנה בממוצע.

6. כל צוות scrum כולל לפחות מהנדס QA אחד מהמחלקה, לרוב אף יותר. (מגבלה פנימית - מהנדס QA לא יכול להשתתף ביותר מ- 2 צוותים בעת ובעונה אחת).

איך נבנים הצוותים וכיצד מתנהלת תקשורת בתוך צוות מבוזר?

כשנפגשים ליצור את צוותי הסקראם עפ"י הצרכים העתידיים, אפשר לבחור להתעלם מהמיקום הגיאוגרפי וליצור צוותים על סמך יכולות בלבד, או לחילופין להציב את המיקום הגיאוגרפי כמגבלה, אשר תכתיב את מבנה הצוותים. החלופה השניה כמובן אומרת שלא יהיו צוותי סקראם שחלקם בארץ וחלקם בחו"ל.

אנחנו בחרנו באופציה הראשונה וכך נוצרו מספר צוותי סקראם אשר מורכבים הן ממהנדסים בארץ והן ממהנדסים בחו"ל (פיתוח ובדיקות כאחד). כמובן שהחסרון הבולט בשיטה זו היא התקשורת היומיומית בין חברי הצוות, והבנת תמונת המצב לאורך הספרינט.

על כך התגברנו באמצעות כמה עזרים (רובם פשוטים לשימוש):
- פגישות צוות יומיות: בתחילה היה הצוות הישראלי מנהל פגישה יומית לבדו כאשר החלק הסיני / אוקראיני שולח עדכונים יומיים במייל ומקבל תגובה בהתאם, אך ככל שעבר הזמן הובן שעל אף הנוחות שבצורת תקשורת זו (offline), המעבר לשיתוף כל חברי הצוות בפגישה היומית מונע סחבת בנושאים שוטפים ומשפר את התקשורת ומערכות היחסים בתוך הצוות.

כך, בהיעדר מנגנון שיחות וידאו מפותח, נעשות הפגישות היומיות בטלפון, וחלקן ב- skype + מצלמת אינטרנט. הצוותים חוו מספר קשיים בפגישות הראשונות, אך לאחר מספר שבועות בודדים הפגישות נערכות כאילו היו במקום גיאוגרפי אחד (בחלק מהצוותים יותר, ובחלק פחות).

- חברי הצוות משתדלים לתקשר יותר פנים מול פנים ע"י שימוש בשיחות וידאו, צ'אט וטלפון, ופחות בעזרת מסמכים ומייל – מקל על העברת המסרים בייחוד בין חברי הצוות בישראל ואלו שבחו"ל.

> כשנפגשים ליצור את צוותי הסקראם עפ"י הצרכים העתידיים, אפשר לבחור להתעלם מהמיקום הגיאוגרפי וליצור צוותים על סמך יכולות בלבד, או לחילופין להציב את המיקום הגיאוגרפי כמגבלה, אשר תכתיב את מבנה הצוותים.

מעבר לכך קיימים מספר קשיים שיש לקחת בחשבון:

1. הבדלי שעות - במידה והחלקים השונים נמצאים במקום בו הבדל השעות רב (במקרה שלנו - 6 שעות לסין, לאוקראינה שעה), צריך לקחת זאת בחשבון בכל צעד שעל צוות ה- scrum לעשות.

2. הבדלי תרבות - באופן פתאומי (בעת יצירת הצוות) נדרשים גם אנשי הבדיקות וגם אנשי הפיתוח, לשתף פעולה עם מקביליהם השייכים לתרבות שונה בתכלית. השוני בא לידי ביטוי בכל: בצורת הדיבור (אפילו אם כל הצדדים דוברי אנגלית, המבטא שונים ולעיתים קרובות לא מובן), בצורת החשיבה ובהרבה אנקדוטות שונות.

חלקנו (כולל עבדכם הנאמן) עבד במשותף עם אנשי הצוות עוד לפני המעבר לסקראם, דבר שהקל במעט. אך המעבר כאמור דרש באחת שלא רק המנהלים ידברו ביניהם, אלא כל חבר צוות עם כל חבר צוות.

במקרה כזה, סדנה לצורך היכרות עם הבדלי התרבות עשויה לתרום רבות.

3. התרגלות - באופן טבעי, בהתחלה התקשורת בין אנשי הצוות בעייתית. מסר פשוט שאמור לעבור בדיבור או בכתיבה, דורש זמן רב יחסית. פועל יוצא מכך (ובתופעה כזו נתקלתי רבות בשנים האחרונות), הינו שחלק מאנשי הצוות בארץ נרתעים מלתקשר ישירות. כאן שם המשחק הוא לתרגל כמה שיותר העברת מסרים בדיבור חרף הקושי, ובצ'אט לכל הפחות (בנושאים רבים זה עדיף על תקשורת offline כמו מייל).

כיצד מתנהלים כשחלקים מהקבוצה (הן מהפיתוח והן מהבדיקות) נמצאים במקומות שונים מחו"ל?

מעבר לתקשורת השוטפת, על צוותי הסקראם לתחזק מספר עזרים ולנהל מידע בצורה משותפת.

- Scrum Board - בתחילה מולא ע"י הצוותים על גבי Excel שהושם במקום משותף ונגיש לכלל חלקי הצוות. כיום מנוהל ע"י תוכנת ניהול Scrum (יפורט בהמשך), באמצעותה הצוותים רואים את פתקי המשימה בצורה אלקטרונית ללא צורך לנהל מספר לוחות במקביל עקב הריחוק הגיאוגרפי.

- שימוש בתכנת ניהול Scrum - דוגמאות למוצרים שכאלה הם Version1, Green Hopper, Rally וכיו"ב. באמצעות התוכנה הצוותים נגישים לכל המידע הרלוונטי לניהול המשימות - product backlog, scrum boards, STP's, STD's, Impediments, sprint backlog, וכן הלאה, וכל פרט המתעדכן ע"י ה- product owner או אחד מחברי הצוות נראים מיידית. חלק מהמפגשים היומיים מבוצעים תוך כדי התבוננות בלוחות שבתוכנה.

אחד היתרונות המהותיים בשימוש בתוכנה הוא שכל החומר אודות המשימות השונות, ממסמכי הגדרה, דיונים בנוגע ל- User story כזה או אחר, חומר נלווה, וכמובן מצב הפיתוח המדובר נמצאים במקום אחד נוח לשימוש.

תחילת כל ספרינט מתחיל בישיבת ה- Planning, בסיומה צוותי הסקראם הרלוונטיים למוצר מתחייבים לספק פעילויות מסוימות. במקרה שלנו קיימים אנשי בדיקות רבים שלנוכח היכולות שלהם ולנוכח אופי המוצרים של החברה, הם נדרשים לקחת חלק ממעל צוות סקראם אחד בספרינט.

כיצד נראות הבדיקות בצורה כזו?

כאשר עברנו את ה- "מהפך" ידענו שברצוננו להצמיד את הבדיקות הפונקציונליות לפיתוח כמה שיותר ולספק מענה רגרסיה כבר תוך כדי סיבוב הבדיקות הראשון. עד לנקודה זו, היו מקרים רבים של טווח זמן מסוים בין הפיתוח עצמו לבדיקתו בפעם הראשונה...

מתוך ראיה שיש לנו פער לגשר עליו בספרינטים הראשונים (עלינו לבדוק גם את מה שפותח בחודשים טרם המעבר, ובנוסף את מה שמפותח בספרינטים הראשונים), גייסנו כ"א יוצא דופן בגודלו (כולל מפתחים), שנרתם לצמצם את הפער, ולאחר פחות מ- 2 ספרינטים המטרה הושגה.

כיום הבודקים בצוותי הסקראם בחברה מספקים מענה בדיקות לכ-80% בממוצע מהפונקציונליות עוד באותו הספרינט, ובו בזמן מפתחים אוטומציה עבור הפונקציות החדשות (כ-50% מפותח on the spot).

כמובן שבמידה ופונקציה מסוימת אמורה להיות מוכנה לשחרור ללקוח (Potentially shippable), היא מפותחת, נבדקת ומתוקנת עוד באותו ספרינט (במציאות שלנו הצורך בכך נדיר).

בחודשים הראשונים שלאחר אימוץ הסקראם כשיטה מובילה, היה מאד קשה לכל אנשי הצוות (ולבודקים כחלק מהם), להתנתק מהשלבים הנוקשים המלווים את פיתוח התוכנה - מפתחים > בודקים > מתקנים תקלות > בודקים וכן הלאה. ברור שאפילו בשיטות אחרות ישנם תהליכים מקבילים בין 2 החלקים (לדוגמא - אפיון STD בזמן ה-design), ובכל זאת ניכר היה שאפילו שאנו מתאמצים למקבל את המשימות על מנת להביא את התוצר במהירות המירבית, היינו עדיין שרויים תחת החשיבה הזו. שלב שלב, החשיבה השתנתה, לדוגמא, design מוקדם של פונקציה שנעשה ע"י אנשי הפיתוח הפך למונחה בדיקה. כלומר נעשתה חשיבה לגבי איך לפתח כך שניתן יהיה לבדוק חלקים של הפונקציה במקביל להמשך הפיתוח, וכמובן מבלי לפגום ביעילות הפיתוח.

המגמה המשיכה להעמיק עד שלאחר שנה של עבודה הופיע פרויקט מורכב יחסית, שעבורו הורכב צוות סקראם במיוחד. לפרויקט נדרשה עבודת תשתית רחבה, GUI מורכב, אלגוריתמים חדשים ובלחץ זמן לא מבוטל, כלומר מציאות שרובנו מכירים מדי פעם.

הצוות הורכב חלקו ממהנדסים בארץ ומהנדסים בסין. רק לאחד מחברי הצוות היה מעט ידע אפליקטיבי רלוונטי, כך שנדרשה תקופת לימוד הן לבודקים והן למפתחים. הצוות ניגש לעבודה והכין רשימת משימות פיתוח ובדיקה, כאשר כל פונקציה בפרויקט הינה בדיקה ברמת המוצר (כלומר דרך ה- GUI הייעודי לה). מרגע שהסתיים התכנון ועד לסיום הפרויקט, ניכר היה כי בכל מפגש יומי, עלה מישהו מאנשי הצוות עם השאלה הגלובלית - "סיימתי עם מה שעשיתי, מה הכי יעיל לעשות כרגע?". ההחלטה על החלק הבא לביצוע היתה משותפת לכלל הצוות, כאשר החלק הנבחר לא בהכרח היה באיזור מוכר לאיש הצוות (כלומר לימוד נוסף היה נדרש כאן), אבל היה ברור שעל מנת לא ליצור צווארי בקבוק בהתקדמות, זה מה שנדרש לעשות. במקרים רבים עזרו אנשי הפיתוח בבדיקות (תחת תשומת לב ,בקרה ותמיכה מאנשי הבדיקות), חלקם על בסיס STD שנכתב מראש וחלקם Exploratory, בהתאם ליכולות ולידע של אותו מפתח. מאידך, התקבלו החלטות בצוות במסגרתן יתמוך בודק ב- unit test של המפתח היכן שזה נמצא לנכון, כלומר כמעט ולא היו חסמים באשר למי מסוגל לבצע מה. סיום את הפרוייקט בזמן המוקצב לו, הגעה לאבני הדרך בזמן בסוף כל ספרינט ובסופו של דבר לפתח מוצר ראוי, היו המטרות שהובילו את האנשים בראש ובראשונה.

כחוויה אישית אוסיף, שכאחראי על צד הבודקים, כצופה וכמייעץ, זה היה הפרויקט שהכי נהניתי לראותו מפותח.

התנהלות קבוצות בדיקה בינ"ל מבוזרות בסקראם

כיצד מתנהלים כאשר מספר קבוצות בדיקה בודקות כמה מוצרים במקביל?

תחילת כל ספרינט מתחיל בישיבת ה-Planning, בסיומה צוותי הסקראם הרלוונטיים למוצר מתחייבים לספק פ' מסוימות. במקרה שלנו קיימים אנשי בדיקות רבים שלנוכח היכולות שלהם ולנוכח אופי המוצרים של החברה, הם נדרשים לקחת חלק ממעל צוות סקראם אחד בספרינט. דוגמא לכך היא במקרה ויש לנו 2 מוצרים שונים שפותחו עבורם פונציונליות מאוד דומות. למעשה כאשר ניגשים הבודקים וראשי הצוותים לתכנון הספרינט, עומדות בפניהם מספר דילמות:

- בהינתן 2 מוצרים שונים ו-2 baclogs שונים, כיצד מתעדפים בין כל משימה באחד לבין כל משימה בשני?
- מהי חלוקת ה- capacity הנדרשת לכל מוצר? איך אני (כבודק, כר"צ) מאפיין אותה?
- איזה צוות (בהינתן המשימות הצפויות להיות תחת אחריותו) זקוק לי (כבודק, כר"צ) יותר בספרינט הבא?
- כאשר יתחיל הספרינט, איך מוודאים שהבודק לא יעבור בין הצוותים מספר רב של פעמים? (unnecessary context switches).
- ועוד ועוד...

הדרך שלנו לפתרון הדילמות הסבוכות האלו נשענת על שלושת השלבים שלהלן המבוצעים אחד אחרי השני:

1. הבנת האילוצים וצפי הפיתוח קדימה - אני (מנהל ה-QA) מפרסם את האילוצים, כלומר אילו גרסאות עבור אלו מוצרים עומדות להשתחרר ומהם סדרי העדיפויות בהיבט המוצרים. (לדוגמא, גרסא X במוצר הראשון הינה בעדיפות ראשונה, גרסא Y במוצר השני הינה בעדיפות שניה וכן הלאה). במקביל, כל בודק יושב עם צוות הסקראם שלו על מנת להבין מה הוא צפי הפיתוח, כלומר על מה הצוות מתכוון לעבוד, הבנת הסיכונים (אילו פיתוחים מסוכנים יותר ואילו פחות, משמעויות וכיו"ב), וכמובן מציף פערים מספרינט קודם לצוות הסקראם, לר"צ שלו ולי.

2. הכנת תכנית פעולה אינדיבידואלית ע"י הבודקים / ראשי צוותים – כל בודק מכין תכנית (בסיס לשינויים כיוון שב- Planning עצמו הדברים עשויים להשתנות). תכנית זו משקללת את הפערים מספרינט קודם (אם יש), התחשבות בפיתוחים מסוכנים יותר ומסוכנים פחות, ה- backlogs השונים אותם עליו לשקלל וכמובן האילוצים שניתנו בסעיף 1.

3. סנכרון כלל חוות הדעת, פידבק והתאמת המשימות לבודקים – היכן שמשוייך בודק רק לצוות אחד, המשימה פשוטה כיוון שרק אלוצי המוצר הרלוונטי ותכנון צוות הסקראם שלו משפיעים על ההכנה שלו לספרינט. כאשר המצב שונה, עלינו לשקלל את כל חוות הדעת ולהבין:

- האם קיים מוצר אשר יווצרו בו פערים שאנחנו מסוגלים לחזות כרגע, כלומר האם עקב בעיית משאבים הוא לא יקבל את המענה הנדרש בספרינט הקרוב. מענה נדרש משמעו בדיקת כל פונקציה שמפותחת, שחרור גרסא במידת הצורך ופיתוח אוטומציה על מנת להקל על בדיקות עתידיות.
- האם נדרשת הקצאת משאבים שונה מהמלצות הבודקים על מנת לחפות על חוסרים כאלו. לאחר מתן פידבק וביצוע התאמות תוך שיתוף הבודקים בביצוע ההחלטה, לכל בודק יש כעת תמונה כללית לגבי הבדיקות אותן יבצע בספרינט הקרוב. כמובן שזו אינה תכנית סופית אלא בסיס כיוון שבפגישת התכנון התמונה עלולה להשתנות.

המטרה של השלבים שביצענו עד כה הינה לוודא שחלוקת המשאבים בין המוצרים מספקת מענה לכל האילוצים והמשימות וכן ביצוע לימוד ע"י כלל אנשי הצוות את המשימות העתידיות שהם עשויים לקחת (sniffing).

4. תזמון המשימות ברגע שנלקחו בתוך הספרינט עצמו – יש לשים לב במהלך ה- Planning לתזמון המשימות בתוך הספרינט. אנו מנסים לשים דגש ולתת עדיפות לבדיקת המשימות המסוכנות יותר וכן למשימות תלויות (כלומר משימה ב' לא יכולה להתחיל לפני שמשימה א' נבדקת). בנוסף, אנו מנסים לאחד משימות הדורשות פעולות זהות וזאת על מנת לייעל את התהליך.

בנימה אישית אני חייב להוסיף כי לא תמיד התהליך נראה כך. לקח לנו לא מעט זמן להכניס את אנשינו בסין לתהליך הזה, וגם היום יש בו עדיין מקום לשיפור.

בחן את עצמך | בדיקות אוטומטיות

1. מי מהכלים הבאים אינו כלי בדיקות אוטומטיות?
א. Oracle ATS
ב. JVC
ג. QTP
ד. RFT

2. על אילו רכיבים במערכת ניתן לתכנן ולהריץ בדיקות אוטומטיות?
א. על ממשק המערכת בלבד (GUI)
ב. רק על הקוד (HTTP, API וכו') מכיוון שה- GUI משתנה כל הזמן
ג. הן על GUI והן על Non-GUI
ד. רק על הרכיבים שמשתנים תכופות שכן שם יש את הסיכון הגבוה ביותר למערכת

3. מה נדרש על מנת לבצע פרויקט אוטומציה בצורה מוצלחת?
א. כלי אוטומציה טוב
ב. מתודולוגיית בדיקות אוטומציה מבוססת
ג. מומחה אוטומציה
ד. כל התשובות נכונות

4. אילו סוגי בדיקות לא ניתן ל"אטמט"?
א. בדיקות רגרסיה
ב. בדיקות שימושיות
ג. בדיקות משעממות
ד. בדיקות פונקציונאליות

5. כאשר בוחרים כלי אוטומציה, איזה קרטריון פחות משמעותי?
א. על הכלי לתמוך במערכת אותה אני מעוניין לבדוק
ב. קלות השימוש בכלי ויכולת ההטמעה שלו בארגון
ג. הספק צריך להגיע ולהציג את יתרונות הכלי למול אלטרנטיבות אחרות בשוק
ד. עלות הכלי צריכה להיות במסגרת התקציב שלי

6. כלי אוטומציה אינם תומכים ב:
א. הרצה של בדיקות לילות (nightly builds)
ב. ביצוע בדיקות חוזרות כמו התקנה וקונפיגורציה
ג. בחירה בסקריפט הבדיקות המתאים ביותר לאוטומציה
ד. שליחת דו"ח סיכום בדיקות האוטומציה לאחר סיום הבדיקות

7. אילו פרמטרים לא ישפיעו על קביעת עדיפות לאיטמוט של מקרה בדיקה?
א. כמות הרצות של הבדיקה
ב. מורכבות איטמוט הבדיקה
ג. יכולת חיזוי תוצאה צפויה של הבדיקה
ד. רמת הבדיקה (יחידה/אינטגרציה/מערכת)

8. סמן את הנכון: בשיטת KDT
א. מומחי האטומציה מפתחים תשתית והבודקים מתכננים את מקרי הבדיקה ומבצעים אותם
ב. אנשי האוטומציה מפתחים תשתית ומקרי בדיקה והבודקים מריצים הבדיקה
ג. אנשי האוטומציה אחראים על פיתוח תשתית, תסריטים והרצות ואילו הבודקים מתכננים את הבדיקות ומפקחים על התצאות
ד. הבודקים אחראים על כלל פעילויות האוטומציה אין צורך במומחי אוטומציה

9. אילו מהנושאים הבאים הינו הפחות קריטי להצלחת אוטומציה?
א. היקף האוטומציה מכלל בדיקות המערכת
ב. הגדרת מטרות ויצירת תכנית עבודה
ג. סינכרון בין אנשי אוטומציה לבודקים הידניים בארגון
ד. תכנון תשתית בדיקות נכונה

10. אילו מהבאים אינו יכול להוות מטרה לאוטומציה?
א. חסכון בכח אדם
ב. זירוז הבדיקות והקדמת הוצאת גרסאות
ג. ניתוח מעמיק של תוצאות הבדיקה
ד. כיסוי רחב של יותר המערכת

גלה האם אתה כבר יכול לבצע בדיקות על אוטומט או שתאלץ להמשיך לעבוד באופן ידני...
(הציון בהתאם לכמות התשובות הנכונות שענית).

0-3 לא רק שאתה לא מוכן לאוטומציה, גם בדיקות ידניות לא היינו נותנים לך לבצע
4-6 תפתח את עמוד 6 בגליון שאתה מחזיק. יש לך עוד לא מעט ללמוד בתחום

התשובות הנכונות בעמוד 35

7-8 לא רע! עוד קצת ידע ונסיון וגם אתה תוכל בלחיצת כפתור לתת לכלי אוטומציה לעשות עבורך את כל העבודה
9-10 אתה מומחה אוטומציה בכיר שמריץ אלפי בדיקות בכל רגע נתון. אתה בטח מתחיל להזיע כשאתה חושב על לבצע בדיקות (או כל דבר אחר) באופן ידני. כל הכבוד!

טאקט בדיקות

matrix company

חדשנות, מובילות, יצירתיות

לעבוד בחברת בדיקות מתמחה שמסתכלת קדימה ומרחיבה את שרותיה לעולמות חדשים ומרתקים

עולם הבדיקות למובייל

- מגוון מכשירים ניידים במקום אחד
- שימוש בסימולטורים ואוטומציה

עולם בדיקות אבטחת מידע

- Penetration Testing
- Security Auditing & Scanning

עולם בדיקות BI ו- DWH

- Quality Gates - כלי ייחודי לבדיקות נתונים
- בדיקות אוטומטיות וידניות למגוון מערכות

עולם הבדיקות ב-cloud

- בדיקות ביצועים ועומסים במודל SAAS
- בדיקות אוטומציה במודל SAAS

היכנסו לדף ה- facebook של **טאקט** בדיקות, לראות משרות פתוחות לפני כולם להנות ממגוון הפתעות והטבות או לשמוע עוד על מגוון השירותים החדשים שלנו

www.tact.co.il | 08-9146000 | www.facebook.com/TactQA

סוף סוף
רואים פנים
השימוש בפרסונות ככלי לתכנון ובדיקות שמישות

אנו רגילים להסתכל על תוכנה כאוסף הפוקציות שהיא חושפת. יותר מזה, נפוץ להגדיר תוכנה כ"סידרת הוראות המובילות מחשב בביצוע משימה". הבנת יישום בצורה זו, בדגש על מאפייניו, גורמת לנו להגדיר תוכנה בצורה מוגבלת לפי מה שהיא יודעת לעשות. כך יוצא שאנו – בודקי התוכנה, המנהלים, הארכיטקטים, והמתכנתים – מאמינים ש-features חדשים הם התשובה לבעיות איכות וחוסר שביעות רצון.

שיטת הפרסונות נמצאת בשימוש בעולם המרקטינג כבר הרבה שנים, אנשי שיווק יכולים להשתמש בה בכדי לזהות איזה מוצר או פתרון חסר למשתמשים מסויימים.

שמואל גרשון
שמואל גרשון בודק תוכנה בחברת אינטל, מדי פעם מדריך טסטרים וקבוצות ומרצה בכנסים מקצועים. הוא מתעניין בשאלות פילוסופיות על בדיקת תוכנה, איכות וערך, ומנסה ללמוד כדי להבין אותן (אם כי לא תמיד לענות עליהם). ניתן לקרוא חומרים נוספים ולהוריד את Rapid Reporter, כלי לרישום בדיקות אקספלורטוריות באתר http://testing.gershon.info

במשתמשי התוכנה סובלים מאותה הראייה, וכחלק מכוונתם הטובה לגרום לתוכנה להיות ידידותית למשתמש, מבקשים פונקציות נוספות... כולם מחפשים את הפקודה הבאה שתהפוך את התוכנה לשימושית.

הבה ננתח את תהליך יצירת תוכנה טיפוסית. אם התחלנו את התהליך עם מסמך דרישות, במסמך תהיה רשימה של כל מה שהתוכנה אמורה לעשות - רשימה של פונקציות, מפורטות יותר או פחות. שלב התכנון והארכיטקטורה יתרגם את הפונקציות לאלגוריתמים ואלו יתורגמו לקוד תוכנה. בדיקת התוכנה גם היא תשקף את הגישה הזו ותתבסס על פונקציות בלבד. אף אחד מהשלבים אינו נותן מענה לניתוח חווית המשתמש ושמישות התוכנה.

לדוגמה, נשתמש באתר אינטרנט להזמנת כרטיסי טיסה (זהו מוצר נפוץ כך שקל לדון עליו). גם כשכותבים את מסמך הדרישות בהתמקדות על המשתמש, הדרישות יתבטאו כפונקציות. לדוגמה, יהיו לנו דרישות מהסוג הבא:

- האתר יציג את כל האופציות השונות לטיסה בין יעדים נבחרים
- האתר יגיש את המבצעים החמים ודילים של הדקה האחרונה במקום בולט
- האתר ייתן אופציה לרכישת כרטיס אחד או יותר
- משתמש יוכל להזמין כרטיסים לכיוון אחד או הלוך ושוב וכן הלאה...

מתכנת יוכל לכתוב תוכנה (אתר אינטרנט) בהתבסס על דרישות אלו, ובודק יוכל לאשר שפונקציות אלו קיימות ואפילו לנסות "לשבור" אותן עם נשקי הבדיקה שלו. המוצר יהיה מוצר חזק שמציג את כל הפונקציות הדרושות. המוצר עונה על כל אחד מהדרישות... אבל האם הוא מוצר מוצלח? חוויות שונות בהזמנת כרטיסים באתרים שונים מוכיחות שיש הרבה מעבר לכמות הפעולות שכל אתר מאפשר (במובן זה הרבה מהם דומים), אבל הדרישות בקושי יכולות להכליל תיאור החוויות, ובכמה מקרים כל מה שמסמך הדרישות יכול להגיד לנו הוא ש-"התוכנה תהיה ידידותית למשתמש" או "חווית ההזמנה תיהיה נעימה למשתמש".

לפרסונות יתרונות נוספים. ראשית, ההיכרות העמוקה הזאת עם משתמש פיקטיבי מאפשרת לנו לענות על שאלות כגון "האם הפעולה הזאת ממוקמת בתפריט הנכון?"

Personas

פרסונות היא שיטת חשיבה שיכולה לעזור בפיתרון בעיות אלו. ראשית כל, נסקור מה השיטה ואיך היא עובדת.

שיטת הפרסונות נמצאת בשימוש בעולם המרקטינג כבר הרבה שנים, אנשי שיווק יכולים להשתמש בה בכדי לזהות איזה מוצר או פתרון חסר למשתמשים מסויימים.

בעולם המחשבים וכתיבת תוכנה, השימוש בפרסונות קיבל תאוצה אחרי שהוצג בספר "The inmates are running the asylum" מאת אלן קופר בשנת 1999. ספר זה טוען שאנשי תוכנה לדורותיהם לא מצליחים לייצר תוכנות המתאימות למשתמשים מפאת חוסר היכולת להתנתק מהידע שהם רכשו ומהצרכים של עצמם.

בעוד שבאופן אינטואיטיבי נהוג לאפיין את המשתמשים של התוכנה בקבוצות כלליות או בעלי תפקידים (כגון 'משתמשים מתקדמים', 'אמהות חד-הוריות', 'סטודנטים'),שיטת הפרסונות דוגלת בהכנת משתמשים ספציפיים בדויים המייצגים את ה-users. השיטה דורשת דיוק בפרטים, גם אם הם לאו דווקא נאמנים למציאות. בין המשתמשים הבדויים שאנו ממציאים כ-פרסונות יהיו נציגים ללקוחות החשובים ביותר של המוצר. רצוי שהדמויות יהיו מוכרות ומהימנות לכמה שיותר צוותים בפרוייקט.

"דיוק בפרטים" משמעותו היא שבכדי ליצור פרסונה, אנו ניתן לה זהות די מורכבת שתהפוך אותה למוכרת לנו באופן אינטימי. לכל פרסונה נגדיר שם, תמונה, גיל, מצב משפחתי, רמת הידע במחשבים, רמת היכרות עם התוכנה שלנו (ועם אלו של המתחרים), רמת שביעות רצון ממקום העבודה, שעות העבודה (והאם יוצא לה לעשות שעות נוספות מדי פעם), דרכי הגעה לעבודה (עד לרמה של איזו מכונית וצבעה, או כמה אוטובוסים הנסיעה מצריכה) וכן הלאה. נקודה נוספת חשובה מאוד היא מטרת הפרסונה בעת שימושה בתוכנה שלנו, הרי נרצה לפעול כך שהדרך למטרה תהיה נטולת מכשולים.

לדמיין את שגרת חיי המשתמשים עוזרת לנו לזהות צרכים. למשל, אם התוכנה שאתם בונים היא תוכנת תקשורת בין-אישית, לדעת על הקשרים האישיים של אחד המשתמשים מעלה כמה שאלות מעניינות לגבי פרטיות ושליטה על המידע.

לפרסונות יתרונות נוספים. ראשית, ההיכרות העמוקה הזאת עם משתמש פיקטיבי מאפשרת לנו לענות על שאלות כגון "האם הפעולה הזאת ממוקמת בתפריט הנכון?", "האם סביר לדרוש את פרטי המידע האלו לפני ביצוע פעולה זו?", "האים בחרנו במינוח מובן לאורך התוכנה?" וכן הלאה. שנית, קריאה למשתמש בשם יוצרת מחוייבות כלפיו ובכך פותרת לנו שלוש בעיות נפוצות:

- בעיית המשתמש האלסטי - כשהמשתמשים מוגדרים בקבוצות כלליות קל להלביש על קבוצת משתמשים תכונות מנוגדות או העדפות סותרות
- בעיית הצדקת התכנון - כאשר אנו מצדיקים תכנון לא מושלם לאחר מעשה
- בעיית התכנון המסתמך על המתכנן - כאשר משקפים על המשתמשים את ההבנה והעדפות שלנו לגבי התוכנה או הטכנולוגיה

השימוש בפרסונות עשוי להיות מהנה מאד, בעיקר כשעוברים את שלב הסקפטיות (תמיד יש סקפטים בהתחלה, במיוחד כשמדובר בפתרון שהוא לא 'טכני' בכלל). בכמה שעות ישיבה משותפת יכול צוות להמציא פרסונות שמוסכמים על כולם כקהל יעד, והצעד בו מדביקים תמונה ושם לדמות הוא מהנה במיוחד.

כדי לבנות דמות פיקטיבית יעילה, כדאי לבסס אותה על מידע אמיתי אודות לקוחות ומתוך סטטיסטיקות, סקרים והתבוננות במשתמש. אך גם כשאלה חסרים, המצאת משתמש בדיוני על סמך ניסיון ושיחות צוות עדיין עדיפה על עבודה עם משתמש אלסטי. לא מומלץ להעתיק פרסונות בין פרוייקטים (מעצם ההגדרה של פרסונה כמשתמש יחודי של פרוייקט), אבל לפעמים קבוצות היעד מספיק קרובות כדי להתבסס על פרסונה קיימת.

משתמשים סופיים של מוצר לא תמיד כשירים להמציא פתרון לבעיותיהם. כמו כן, פרסונות אינם תמיד עוזרים כשהם נשאלים "איך לתכנן מערכת". השיטה עובדת יותר טוב עם שאלות שיפוטיות מאשר עם שאלות יצירתיות. כך למשל שאלות שנותנות תוצאות טובות יותר הם בסגנון "בתכנון מסויים, איזה צעד הכי מפריע בביצוע משימה" או "איזה מבין שני תכנונים מאפשר להגיע עד המטרה בצורה פשוטה יותר". זו עוד נקודה בה השיטה מראה את חוזקה בבדיקות תוכנה - היא חושפת נקודות חולשה בשימושות של תכנון או תוכנה נתונה בצורה פשוטה ומובנת. בנוסף, שאר הצוות (מנהלים ומתכנתים) מבינים בקלות את נקודות החולשה האלו כשהם חשופים לפרסונה שלה זה מפריע. סוף עידן הויכוחים על באגים של שימושות?

מקומות עבודה מסויימים מדפיסים את כרטיס הפרסונות בפורמט גדול ותולים בחדר הישיבות כדי שיהיו זמינים בכל שיחה על איכות, באגים או תכנון. חברות אחרות מושיבות את הכרטיסים על כסאות כך ש-'משה', 'עובדיה' ו-'יוליה' הפרסונות, יתפסו מקום של כבוד בכל החלטה. אם הדמויות מוכרות לכולם ונראות לעיין, יש סיכוי לשיחות יותר ממוקדות והבנה קולקטיבית אודות באגים ותקלות.

הלכה למעשה

ראינו ששיטת הפרסונות יכולה לעזור לנו לשפר את תהליך התכנון ובדיקות תוכנה ולכוון את פתרון התוכנה טוב יותר לקראת הבעיה שהיא באה לפטור. השיטה גורמת לנו להבין את הדרישות (מתוך מסמך, בעל פה או מרומזות) בצורה שונה - לא רק ה-'מה' (פונקציות) אלא גם ה-'מי', ה-'איך' (משתמשים) וה-'למה' (מטרות המשתמש והעסק).

אם נחזור, לתרגיל קצר, לדוגמת האתר להזמנת כרטיסים, נוכל לדמיין פרסונות שונים. חשוב להבין שחברות שונות להזמנת כרטיסים יכולות לעבוד מול פרסונות שונים לפי המטרות העיסקיות של כל אחת, והדוגמאות בכתבה זו הם פרי דימיוני בלבד.
מי יכולים להיות ה-users הטיפוסיים לאתר זה? ננסה:

- בועז, 42, מדימונה הוא נוסע מתמיד המבלה כחצי מזמנו מחוץ לארץ. הוא נציג senior בחברת חלקי מכונות. הנסיעות שלו הן לכל העולם, אך לעיתים תכופות ליעדים חוזרים בארה"ב וסין.
- רקפת, 30, ממבשרת ציון, נוסעת לעיתים רחוקות מאוד, רק כשיש שמחות אצל משפחת בעלה בלונדון.
- חגית, 38, מתל אביב היא סוכנת נסיעות מקצועית. היא משתמשת באתר כתחליף מזדמן למערכת הרשמית לאיתור טיסות, כדי לבדוק אופציות וקומבינציות שונות להגיע ליעד אחד.
- שרה בת 35 היא אמא לארבעה מרעננה. בעלה נוסע לאירופה לעיתים קרובות, וכשמסתדר להם היא מצטרפת אליו עם הילדיםאו בלעדיהם.

מבלי להכנס לדיון לגבי נכונות או איכות הבחירות הנ"ל, עם הפרסונאס האלו בתודעה נוכל לחזור ולקרוא את הדרישות שבתחילת הכתבה בעיניים חדשות.

כך למשל, כשמביימים ביקור באתר של כל הפרסונאס העיקריות, נראה לעיין שלרוב הם מגיעים עם יעד מוגדר, הדילים והמבצעים לא שימושיים כלל. האם כדאי להסתיר את המבצעים לטובת דף בית נקי יותר? לחלופין, האם כדאי להבליט אותם כך שאולי בכל זאת נשכנע מישהו מהמשתמשים?

כמו כן, 'חגית', כסכנת נסיעות מוצאת ערך באופציה של תמחור מספר רב של כרטיסים. טסטר בתפקיד 'שרה' יבחין שבנוסף ישנו צורך להציג באופן ברור כיצד לשייך חלק מהכרטיסים לנוסעים צעירים.

חשוב לציין שבחירה ברשימה שונה של פרסונאס היתה יכולה להביא לתוצאות שונות ולמוצר שנראה אחרת לגמרי... פרסונאס מעניקים יחודיות וכושר הסתגלות למוצר תוכנה.

בניגוד לדעה הרווחת, פרסונאס אינם מתאימים רק ליישומים בעלי ממשק גרפי (GUI), אלא לכל מערכת בעלת ממשק. למשל, במערכת טלפונית לניתוב אוטומטי, פרסונאס יוכלו לעזור לנו לחשוב על שאלות כגון "אילו הן הפעולות הנפוצות יותר אצל 'אריאל'?" או "איזה מידע אישי זמין נגיש ל'אהרון' (72) בעת השיחה?". בשביל ספריות תכנותיות (כ- APIs או SDKs), עם שימוש בפרסונאס אולי יעלו השאלות "על איזה מהפרמטרים 'לאונרד' רוצה לשלוט?", "איזו מהפעולות הוא יעדיף שהספרייה תעשה עבורו בלי אינטראקציה?"

סיכום

המילון הממוחשב שלי מתרגם "לא ידידותי", בין השאר, כ-"שאינו מתייחס לאדם או חפץ מסוימים".

לפי הנאמר לעיל, ניתן לראות שיחס לא-אישי יכול להקשות על בניית תוכנה ידידותית ושימושית, ואילו היכרות אינטימית עם לקוח דמיוני בכוחה להפוך משימה זו לקלה ומהנה יותר. הצעד הבא הוא לנסות זאת בעצמכם, בפרויקטים בהם אתם עובדים. פרסונה בצורתה הפשוטה ביותר תצריך רק מספר שעות עבודת צוות.

חמש נקודות בבניית פרסונות:

1. אל תוותרו על שם ותמונה בבניית הפרסונות שלכם

2. השתמשו בשם הפרסונה בשיחות היום יום, במקום לדבר על ה-"פרסונות" בצורה כללית (אחרת זה לא יעיל יותר מלדבר על משתמשים כללים)

3. כל משתמש מעוניין בתוכנה שלכם בשביל משימות מסויימות. התמקדו במטרות שלהם

4. אפשר להשתמש במגוון מקורות מידע על המשתמשים כדי לגלות מי הם. חפשו מחוץ למחלקת הפיתוח, כגון מחלקות שיווק ושירות לקוחות

5. כדי שבניית הפרסונות תהייה יעילה וממוקדת יותר, ניתן להשתמש בשירות Personas Profiling כמו של http://www.usersbox.com

רשמים מכנס QA EXTREME 2011

בתאריכים 12-14/6 התקיים לראשונה בישראל, כנס בדיקות מקצועי שכלל מעל ל- 20 סמינרים מקצועיים שונים ומגוונים.

למעלה מ-400 משתתפים לקחו חלק בכנס הבדיקות QA Extreme שערכו ג'ון ברייס הדרכה וטאקט בדיקות במלון דניאל שבהרצליה. הכנס נמשך 3 ימים ובמסגרתו עברו המשתתפים סמינרים מקצועיים בתחומים המובילים כיום את השוק כמו Agile, Cloud, Mobile ועוד.

כנס הבדיקות היה הכנס הראשון מסוגו שנערך בישראל שהוקדש כולו לחשיפה ולמידה על החידושים והטכנולוגיות המתקדמות בתחומי הבדיקות. את הסמינרים העבירו מרצים בעלי שם וניסיון רב בתעשיית הבדיקות, כגון: ד"ר רונן בר-נהור - סמינר בדיקות Agile, טלמון בן כנען, מנהל אגף איכות מוצרים באמדוקס - סמינר מדדים אפקטיביים לפרויקטי בדיקות, ליאור כץ, מנהלת תחום אוטומציה ובדיקות עומסים בטאקט בדיקות - סמינר בנושא אוטומציה ובנושא עומסים, אבנר אלגום, יו"ר איגוד הגריד הישראלי (IGT) - סמינר בנושא טכנולוגיות ב- cloud, רם יוניש, סמנכ"ל בטאקט בדיקות תוכנה - סמינר ניהול קבוצות QA בהווה ובעתיד, אריאל גור אריה, מנכ"ל קפיטו - סמינר בנושא אוטומציה ב- VSTS, יואל מונטליבסקי, מייסד PractiTest - סמינר בנושא בדיקות ב- cloud ועוד.

המשתתפים בכנס הביעו שביעות רצון גדולה נוכח קיומו של כנס מקיף בתחום הבדיקות שהדגש בו הוא שיח מקצועי וממוקד. אחד המשתתפים המרוצים הוא גדי פלד, מנהל בדיקות בחברת אלתא, שאמר בסיום הכנס: "הכנס היה מעניין וענה על כל ציפיותיי ואף מעבר. נהנתי מאוד גם מהשיח עם העמיתים למקצוע ובכלל, הסדנא נתנה לי דגשים על נושאים שאני רוצה לקדם אצלי בקבוצה כגון: Agile, ROI ונושאים חשובים נוספים".

אלון אנגלר, מנהל QA בחברת סאפיאנס סיפר בתום הכנס כי "נדיר למצוא כנסים טכנולוגיים חדשניים עם אורינטציה לתחום הבדיקות. סוף סוף מישהו הרים את הכפפה ונתן לנו משהו שיכול לשרת בצורה טובה את כל אנשי ה- QA באשר הם. הסדנאות היו מאוד מקצועיות וברמה מעולה, העשירו את הידע ופתחו אופקים מקצועיים חדשים".

בין ההרצאות המעניינות שהיו בכנס בחרתי להציג את הנושאים הבאים:

סדנה מעשית לניהול בעולם בדיקות התוכנה: ROI, practical metrics, Estimation model

טלמון בן כנען (אמדוקס)

לטלמון ניסיון מעשי רב בתוקף תפקידו בתחום האיכות באמדוקס והוא בחר להציג בפני המשתתפים מודלים ומדדים מעשיים שבהם הוא משתמש בחיי היום יום באמדוקס. במסגרת הסמינר שלו התייחס טלמון לשלושה נושאים מרכזיים:

- מדדים - כיצד להתמקד על מה שחשוב, שימוש במדדים כמנוע לשיפור, ניתוח נתונים ויצירת תכנית התייעלות מבוססת מספרים
- מודל הערכה - כמה עולה לבדוק תכנה? מה הם הפרמטרים המשפיעים על מחיר הבדיקות?
- חישובי ROI וכיצד להשתמש בהם למינוף תקציב הבדיקות

בנוסף לניסיונו הרב, שיתף טלמון גם הרצאת אורח של עמוס אוזן, לשעבר QA director בחברת BMC שהציג אף הוא היבטים יישומיים למדידה של פרויקט.

כיצד לנהל פרויקטי בדיקות ביצועים

ליאור כץ (מנהל תחום אוטומציה, עומסים ו- ALM בטאקט בדיקות תוכנה)

לליאור ניסיון רב מאוד בתחום האוטומציה, עומסים וביצועים. ליאור חלק עם המשתתפים מניסיונו הרב בניהול פרויקטים של עומסים וביצועים והתייחס להיבטים הבאים:

- כיצד לנהל פרוייקטי עומסים בצורה הנכונה
- איך לבחור את כלי העומסים שמתאים לך...
- סוגים של בדיקות עומסים
- Configuration management
- Monitor tools

כמו כן הציג ליאור את מתודולוגיית ה- TTM (Testing Tools Methodology) לניהול מסודר של פרוקיטי אוטומציה ועומסים.

מגמות ודגשים לבדיקת אפליקציות מובייל וסקירה על עולם המובייל

ערן קינסברונר (מוביל תחום בדיקות מובייל בטאקט בדיקות תוכנה)

לערן ניסיון רב מאוד בתחום המובייל בכלל ובבדיקות מובייל בפרט. בסמינר של ערן למדו המשתתפים כיצד נראה שוק המובייל העולמי והמקומי היום ולאן הוא צפוי ללכת בשנתיים-שלוש הקרובות. בנוסף לסקירת שוק המובייל בכלל פירט ערן והציג דוגמאות רבות וטיפים לביצוע בדיקות על פלטפורמות ניידות. להלן הנושאים המרכזיים שהתייחס אליהן ערן בסמינר:

- קשיים ואתגרים בבדיקות מובייל
- סקירת מערכות ההפעלה הבולטות בעולם המובייל כיום ועד לא מזמן
- מתודולוגיות בדיקה בעולם המובייל
- Porting - מה משמעות המושג במובייל ואיך אפשר לעשות זאת חכם ויעיל.
- סקירת מספר כלים שימושיים בעולם המובייל עפ"י מערכת הפעלה
- מה צפוי לנו בעתיד הקרוב בעולם המובייל
- הרצאת אורח - ערן אריה, מנהל בדיקות PS בטלמפ שהציג היבטים מעשיים של ניהול בדיקות בחברה שמקבלת 7-8 מכשירים חדשים כל חודש ויכצד מתבצע תהליך התאמת המוצר למכשירים חדשים בשוק
- הרצאת אורח - עומר בונפיל, קומיוניטייק - סקירה קצרה על סביבת הפיתוח במק לאייפון כולל הדגמה שלתהליך הרצת אפליקציה לדוגמא מסביבת הפיתוח על סימולטור של אייפון

כמו כן הציג ערן דוגמא מעשית לביצוע אוטומציה על מכשיר אנדרואיד ואייפון באמצעות SeeTest של חברת Experitest.

שימוש בכלי האוטומציה והביצועים של Microsoft - VSTS 2010

הסמינר הזה חולק לשניים, כשאר בחלק הראשון הציג דניאל בראל כיצד ניתן להשתמש בצורה נבונה ביכולות מדידת הביצועים והעומסים הקיימות ב- VSTS 201.

בחלקו השני הציג אריאל גור אריה, מנכ"ל קפיטו את היכולות והדגשים לפיתוח תסריטי אוטומציה באמצעות Coded UI.

הסמינר בוצע במעבדת מחשבם שהוקמה לצורך כך באחד מהחדרים בבית המלון וכלל תרגול מעשי של המשתתפים

סמינר בדיקות אבטחת מידע - penetration testing, חיליק טמיר (Appsec) חיליק טמיר מחברת Appsec העביר סמינר שלם בנוגע לאיומי אבטחת המידע הקיימים כיום וכיצד אנו כבודקים/מפתחים יכולים לזהות איומים או פרצות במערכות שלנו כחלק מתהליך הפיתוח. בין היתר הציג חיליק דוגמאות לשיטות שכיחות יותר ופחות של האקרים לפריצה למערכות כגון: XSS, SQL Injection, log injection, code injection וכדומה.

מן הסמינר הזה ניתן היה לקבל הבנה טובה כמה חשוב מחד לבצע את הבדיקות האלו בתהליך הפיתוח ומאידך שלא חייבים בהכרח מומחי אבטחת מידע מדופלמים לכל סוגי ההתקפות.

סדנה מעשית לבדיקות WEB לירון בידרמן (ג'ון ברייס הדרכה)

לירון בידרמן, מוביל תחום הבדיקות בג'ון ברייס הדרכה, העביר סדנה שהתמקדה בהיבטי בדיקות ב- web כולל התייחסות לצד השרת ולצד ה- client.

בסדנה הוצגו טכניקות ודגשים לבדיקות בעולם ה- UI וכן הוצגו כלים מעשיים לייעול בדיקות אלו. גם בסדנה הזו הוקמה מעבדת מחשבים לתרגול מעשי.

לסיכום:

בשורה תחתונה, הכנס היה מקצועי מאוד וכלל הרבה תכנים מעשיים ואפקטיביים במגוון תחומים. בשונה מאוד מכנסים אחרים בכנס הזה לא היו הרצאות מליאה של מאות אנשים אלא מפגשים מקצועיים ואינטימיים מאוד שאיפשרו תקשורת ישירה בין המרצה למשתתפים ובינם לבין עצמם. בנוסף, היה שימוש נרחב במרצים "מהשטח". מנהלים ואנשי בדיקות שהביאו מהניסיון האמיתי שלהם בחברות בהם עבדו וחלקו את הניסיון הזה עם עמיתים למקצוע.

מעבר לפן המקצועי, האירוח של הכנס במלון דניאל היה מדהים וכלל ארוחת צהריים מאוד מפנקת והפסקות אמצע מתוקות ומשובחות.

לאור הצלחת הכנס ג'ון ברייס הדרכה כבר קבעה מועד לכנס הבא שיתקיים בחודש מרץ- אפריל בשנה הבאה.

נתראה שם,
רם

תשובות נכונות מתוך:
בחן את עצמך - בדיקות אוטומטיות (עמ' 28):

1(ב) 2(ג) 3(ד) 4(ב) 5(ג) 6(ג) 7(ג) 8(א) 9(א) 10(ג)

בדיקת ההתקנה והוראות ההתקנה

דורון בר

פעמים רבות אנו הבודקים מזניחים את בדיקות ההתקנה ובדיקות הוראות/מסמכי ההתקנה, כשלמעשה זהו אחד החלקים החשובים ביותר בהעמדת המערכת ובשיווקה.

ברור שאנו מתקינים את המערכת במהלך הבדיקות, וברור שאנו עורכים בדיקות קיט (אני מקווה), וכנראה בסביבות שונות. כמו כן ברור שבמידה שלא ניתן לסיים את ההתקנה אנו נתריע על כך כדבר קריטי. אבל זהו מסלול אחד של בדיקות, ופעמים רבות - זה אינו המסלול היחידי.

Next

דורון בר

דורון בר נמצא בתחום הבדיקות משנת 1998. בין יתר תפקידיו הוא בדק וניהל בדיקות של מערכות ERP בחברה שנרכשה ע"י אורקל, בדיקות של מסרים בסביבת טלפוניה, ובדיקות של אתרי אינטרנט בינלאומיים. בנוסף דורון ניהל בקומברס פרוייקטי בדיקות מורכבים של מוצר הדגל של החברה, כולל ניהול של ארבעה ראשי צוותים.

הבדיקות שנוהלו על-ידו כללו את כל סוגי הבדיקות, כולל בדיקות עומסים, בדיקות אוטומאטיות, ידניות וכו'.

ייתכן שמצב זה קורה כיוון שהרבה פעמים יש לנו בעיות שנראות קריטיות יותר (פונקציונליות שאינה עובדת למשל), ובואו נודה על האמת - סקסיות יותר. יותר קל לנו לדווח על באג הקשור ביוזביליות בקליינט מאשר על זו שבתהליך ההתקנה. ובדיוק נגד גישה זו אני רוצה לצאת.

בעיקרון יש לציין שאני מדבר בעיקר על התקנות מורכבות יותר שלא כוללות רק next next next, אך מן הראוי גם לתת תשומת לב גם לאלו הפשוטות יותר.

להתקנה יש להתייחס ככרטיס הביקור הטכני של המוצר שלנו. היא צריכה להיות אלגנטית, ברורה וקלה ליישום, וכן על המסמכים שלה לצפות לבעיות שעלולות להתעורר ולהציע דרכים לפתור אותן, כמו גם דרך ליצירת קשר למתן פידבק או במידה של בעיה. צריך להיות ברור לנו שמי שיאבד את האמון בהתקנה יאבד את האמון במוצר. ולא משנה העובדה שבמקרים מסויימים מי שמתקין את המוצר הם אנשים מהחברה שלנו, או אנשים בחברה אחרת שאינם אותם האנשים שחתמו לנו על הצ'קים. אם האנשים שלנו יאבדו את אמונם, ובוודאי שאם המתקינים בצד של הרוכשים ייאבדו את אמונם, זה עלול לחלחל הלאה וליצור תסכולים ובעיות בהמשך.

כמובן שגם אם ההתקנה הושלמה כביכול בהצלחה, אבל שלא נעשתה מתוך מסמכים ברורים, זה עלול ליצור בעיות פונקציונליות אח"כ. אלו בעיות שבדרך-כלל קשה מאוד לעלות על הסיבות שלהן ולכן דורשות שעות תמיכה מרובה וגורמות שוב לתסכול אצל כולם.

ראיתי מסמכי התקנה עם תמונות שאינן רלוונטיות יותר, עם שורות הרצה "דוסיות" ארוכות שיש להעתיק אות-אחרי-אות, מונחים לא ברורים וכד'. פעם אפילו עם משפט כזה: "במידה ונתקלתם בבעייה תשאלו את יוסי כהן (שם בדוי), בטל' 054-1111111", כשאותו "יוסי" הוא הבודק היחידי שמכיר את העניין ושבהחלט אינו אמור ואינו יודע לענות לבעיות של לקוחות - ואני לא מגזים!

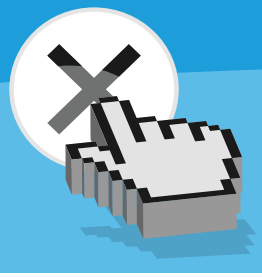
באחד מתפקידי בחברת "קומברס", עבדנו על המערכת המורכבת ביותר שהייתה שם שכללה שרתים רבים ומסוגים שונים, בעלי רידנדנסי ועוד. את המערכת היינו מתקינים בדרך כלל בארץ בעצמנו (מחלקת הייצור), אלא אם היא הייתה נשלחת לארה"ב. בארה"ב היתה לנו מחלקה שהייתה ממוקמת שם והם היו מתקינים את המוצר בעצמם.

ואכן האמריקאים התלוננו רבות על ההתקנה. חשבנו שכאמריקאים פדנטים בטח חסר להם איזה פסיק איפשהו במסמך ההתקנה וזו כל הבעיה שלהם. ברור שזו היתה גישה מתנשאת ושאינה במקום. ואכן כאשר בדקנו לעומק התוצאות היו חד-משמעיות: קהל היעד - שהיה אדם שהוא איש טכני ברמה המתאימה אך שלא התקין את המערכת בעבר (גם לא נעזר באדם כזה) - לא היה יכול להתקין את המערכת! ההליכה לפי ההוראות פשוט לא הייתה מביאה אותו לסיום מוצלח של ההתקנה.

באותה נקודה בעצם הבנו שאנחנו פשוט מכירים את המערכת לעומק כך שאנו בכלל איננו עוברים צעד-צעד אחרי ההתקנה, אלא נעזרים בה פה ושם בכדי להיזכר בפקודות וכד'.

בנקודה זו כבר היה ברור לנו שעלינו כבודקים (למרות שהבעיה היא רוחבית) לערוך מהפכה בנושא: קודם כל מהפכה תפיסתית, אח"כ מעשית.

ואם לאפיין את המהפכה במילה: ההתקנה היא חלק אינטגרלי של המוצר ולכן יש להתייחס אליה כמו לכל חלק אחר של המוצר



יש לאפיין את ההתקנה כולל דרישות שליליות כמו שלא יצטרכו להקליד שורות פקודה ארוכות אלא שיהיו קבצי אצווה בעלי שמות פשוטים שיעשו את העבודה. יש לכתוב מסמכי בדיקות ולבצע אותם, יש לדווח על באגים בהתקנה וזה כולל באגים של יוזביליות, וכמובן לדאוג שהם ייסגרו.

מהאיפיון עד לבדיקות. ובשתי מילים: יש לאפיין את ההתקנה כולל דרישות שליליות כמו שלא יצטרכו להקליד שורות פקודה ארוכות אלא שיהיו קבצי אצווה בעלי שמות פשוטים שיעשו את העבודה. יש לכתוב מסמכי בדיקות ולבצע אותם, יש לדווח על באגים בהתקנה וזה כולל באגים של יוזביליות, וכמובן לדאוג שהם ייסגרו.

בתחילת דרכנו כבודקים, מתן דגש להוראות ההתקנה עורר אנטיגוניזם בחלקים מסויימים בפיתוח, כנראה מאותן סיבות שכתבתי למעלה אשר בגינן לא שמנו דגש על ההתקנה בתחילה. אני זוכר מיילים שבאחד מהם אני, כמנהל הפרוייקט בצד של הבדיקות, הייתי הנמען. זה הלך משהו בסגנון של: מה, נגמרה לכם העבודה? אין באגים יותר במערכת? יש לי דברים חשובים יותר לעסוק בהם ואני לא מתכוון להתייחס לבאגים של התקנה.

הנושא הזה טופל נקודתית, אבל חשוב לציין שעלינו להסביר "לעולם" (כמו שאכן עשינו אגב גם במקרה שלמעלה), לפני שאנו פותחים שורה של דיווחי באגים, מה היא המוטיבציה שלנו, ולרתום את כל הגורמים לעניין.

בכל אופן, בסוף זה השתלם ומסמכי ההתקנה וההתקנה עצמה שופרו ללא הכר.

וכעת לעצות מעשיות יותר:

דבר ראשון: בכל מקום שאפשר - יש לארגן מה שנקרא "One click installations" שכשמו כן הוא: לוחצים על כפתור אחד (גם שלושה זה בסדר) וכל השאר הוא אוטומאטי (לפחות פר שרת). זהו תהליך פיתוח רגיל של אפיון, פיתוח ובדיקות.

דבר שני, וכשכבר אנו חושבים שההתקנה טובה, יש לזכור את קהל היעד. לכן מומלץ מידי פעם להביא לבדיקות אלה בודק (או מישהו אחר) בעל רקע מתאים שלא התקין את המערכת הזו מעולם ולתת לו לעשות זאת בעצמו (ובהשגחה שקטה - מעיין מבחן יוזביליות).

בנוסף, וזה עניין שצריך להתחיל בדרישות ואם אין התייחסות אזי עלינו לדרוש את זה בתהליך ההתקנה: עלינו לוודא שניתנה הדעת לשני הנושאים הבאים:

- Fallback - אם לא הצלחנו בדרך א' יש לנסות דרך ב'.
- Rollback - במידה וההתקנה פשוט לא מצליחה - אפשר לחזור בבטחה למצב טרום ההתקנה.

ברור שאנו לא רוצים להגיע לאף אחד מהמצבים הללו, אך עלינו להיערך להם.

מבחינת הבדיקות זה "סיוט" לבדוק התקנה טובה (דווקא!), כלומר בנוייה היטב, של מערכת גדולה, כי כמות ה- Fallbacks וה- Rollbacks היא אדירה. כל שרת בתת-מערכת לחוד, ואולי גם ביחד של כל אחד מהמצבים האלה. יש לבצע כמובן אופטימיזציה וניהול סיכונים.

ולבסוף: יש שני מצבים של הרצת מסמך בדיקות ושנייהם חשובים: Dry run וכמובן הרצה בפועל. מטרת ה Dry run היא למנוע מאיתנו הרצה של שלושת-רבעי המסמך (ויום בדיקות) בכדי להבין שמשהו חסר, משהו שהיה אפשר להבין מתוך דפדוף במסמך בזמן של עשרים דקות.

להלן צ'ק ליסט לעקרונות בדיקת התקנות:

דיוק:

כמו בבכל TD - ההתקנה עובדת בהתאם לדרישות.

בהירות:

1. המונחים וראשי התיבות שמשתמשים בהם בתהליך ההתקנה מוסברים
2. שפה ברורה ותמציתית
3. הפעולות והתוצאות ברורות וחד משמעיות
4. ישנן תמונות להמחשה כולל סימנים בתמונות שיבהירו בדיוק מה יש לעשות
5. אם ישנן כמה אפשרויות - להסביר כל אפשרות ובמה היא שונה מהאחרות

שלמות:

1. שום צעד אינו חסר
2. הצעדים הם לפי הסדר (חשוב מאוד!)
3. ציון דרישות הקדם
4. שגיאות אפשריות ופרוצדורות של recovery מצויינות במסמך
5. ציון של כמה זמן אמור לקחת כל תהליך
6. הסבר של שיטות ההתקנה (ידניות, הרצות של exe וכד')

קונסיסטנטיות:

1. פנה את קהל היעד שלך כיאות (אם אלו אינם אנשים טכניים אל תצפה שיבינו מונחים כמו batch file לדוגמא).
2. הפעולות והתוצאות כתובות בצורה עקבית עם הפירוט המתאים, ובאותו פורמט
3. התהליך והמעבר בין הנושאים הם לוגיים, אין קפיצות מנושא לנושא

תצורה גראפית:

1. Toolbars, תפריטים, פקודות ואפשרויות פועלים בדיוק כמו במסמך
2. ברירות המחדל במסמך וב GUI זהות.
3. דוגמאות מתועדות נכון

ארגון:

1. יש אזכור של מסמכים קשורים
2. מטרות כל תהליך מוסברות
3. יש overview של ההתקנה
4. לא יותר מ-10 עד 15 צעדים בכל פרוצדורה
5. הצעדים ממוספרים
6. מינימום של פעולות ידניות
7. יש מקום להערות וחתימה של הבודק
8. להכניס הסברים מפורטים על כל צעד זה מבורך, אבל לא במסמך עצמו - יש מי שמעוניין פשוט להתקין! לכן אפשר להשאיר קישור לנספח ולרשום שם את הפירוט - מי שמעוניין - יחפש שם
9. המשך של 8#: אפשר ורצוי גם לתת טיפים. את זה אפשר להשאיר ליד המקום הרלוונטי, אבל בקצרה ובפורמט וצבע שונים עם תיוג ברור

סטייל:

1. נראה מקצועי
2. משתמש במונחים מקצועיים

סקירת כלי קוד פתוח/חינמיים לשימוש בבדיקות אוטומטיות

אז הטור שלי עוסק במגמות בבדיקות תוכנה והמגזין של יולי עוסק באוטומציה אז ישבתי לי וחשבתי, איך לשלב בין השניים ולהביא ערך אמיתי לקוראים שלנו (כן, לכם). ואז החלטתי פשוט להתקשר לידיד טוב, שמבין דבר או שניים באוטומציה וגם עושה ניסים בשימוש בכלי open source ליצירת סביבת אוטומציה ייחודית שכולנו יכולים רק לחלום עליה ולבקש ממנו להסביר לנו איך הוא עושה את זה. אז הנה הסיפור האמיתי של ליאור קינסברונר, מנהל אוטומציה בחברת RSA, לקרוא וללמוד מהטובים ביותר.

עורכי מגזין יולי

ליאור קינסברונר, בן 36, נשוי +1, גר בחיפה. מנהל האוטומציה בחברת RSA ואח תאום של ערן קינסברונר (מי שמכיר מכיר).

אזהרה והמלצה: ליאור כמו ערן אחיו אנשים מאוד טכניים. מי שמגדיר עצמו כטכנופוב ראוי שלא ימשיך לקרוא את המאמר. מי שרואה עצמו כמומחה אוטומציה ראוי שלא יוותר על פסיק.

בימים של משברים כלכליים, תקציבים מוגבלים והחיפוש המייגע עבור חבילת כלים שבעלות נמוכה או אפסית יביאו לתוצאות הרצויות נושא השימוש בכלי קוד פתוח או כלים חינמיים הופך להיות מרכזי ביותר. בשנים האחרונות נכנסו לתחום הבדיקות האוטומטיות כמה שחקנים חדשים בתחום הזה ואני אנסה לתת סקירה מהירה על יעילותם והערך המוסף שבשימוש בהם.

אתחיל ואומר שכל הנאמר במאמר זה הינו בגדר המלצה על סמך נסיון אישי וביקורות של עמיתים לתחום – ישנם ארגונים או מוצרים מסויימים שהכלים המוזכרים לאו דווקא יביאו את התועלת המקסימלית.

פלטפורמה לניהול, הרצה, דיווח וניטור הבדיקות האוטומטיות JSYSTEM מדובר במערכת קוד פתוח מבוססת ג'אווה שנותנת אפשרות לכתיבה מתודולוגית של תשתית האוטומציה (בשפת ג'אווה כמובן בכל IDE שמועדף על המשתמש (ECLIPSE / INTELIJ) ולממשק אותה לאפליקציה חלונאית (JSYSTEM RUNNER או JRUNNER) השולטת בכל הקשור בניהול חבילת הבדיקות שפותחה: בניית תסריטי הבדיקות בצורה ידידותית למשתמש, קינפוג של המשתנים בטסטים המורצים, קינפוג של הסביבה מולה רצים (SUT) וניתוח קל של תוצאות ההרצה. המערכת מאפשרת הרצה מבוזרת על כמה סביבות בו זמנית, כמו כן מאפשרת הרצה טורית של תסריטי בדיקות מול כמה סביבות. אמנה כמה יכולות נוספות ושימושיות של המערכת כגון: אפשרות הרצה ושימוש בכלי גם בסביבת WINDOWS וגם בסביבות UNIX. כתיבת לולאות עבור הרצה מרובה עם פרמטרים שונים עבור כל איטרציה ללא צורך ביכולות פיתוח, עריכת מאות טסטים אוטמטיים וקינפוג הפרמטרים בצורה קלה וידידותית, התממשקות לכלי תצורה וכלי עזר כמו QC, HUDSON, MAVEN ואחרים שיוזכרו בהמשך המאמר.

כתובת האתר הרשמי:
http://www.jsystemtest.org

BadBoy – כלי בדיקות לאפליקציות WEB המבוסס על הקלטה-הרצה עם יכולות נוספות כגון הטמאת קוד JavaScript בתוך הטסטים. הכלי מאפשר הקלטה, בניית תסריטים (עם יכולת לשלב לוגיקה, לולאות וולידציות) והרצתם בקלות יתרה תוך שילוב יצירת נקודות ביקורת, לקיחת צילומי מסך תוך כדי הרצה וכו'.

כתובת האתר הרשמי:
http://www.badboy.com.au

חסרונות בולטים שגיליתי בכלי:
- הכלי דורש התקנתו בכל סביבה שמריצה אותו!
- הכלי אינו תומך בכל גרסאות הדפדפנים האחרונות (לפחות לא בגירסא היציבה האחרונה שהשתמשתי 2.1).
- יכולות הדיווח וניטור התקלות לא משביעות רצון ודורשות מאמץ ניכר.

כלי לבדיקות אוטומטיות של אפליקציות WEB הנקרא SELENIUM – הכלי מאפשר הרצה של מגוון תסריטי בדיקות על מבחר רחב של דפדפנים (דוגמת: אינטרנט אקספלורר, פיירפוקס וגוגל כרום). מערכת selenium-rc היא למעשה שרת הכתוב בשפת התכנות java, מהרגע שטוענים את המערכת לזכרון ניתן להעביר לה הוראות לביצוע בדיקות. כאשר המערכת מקבלת הוראות כאלו היא טוענת את הדפדפן הרלוונטי ומעבירה לו רשימת פקודות ב- JavaScript. פקודות אלו הן למעשה פעולות המשתמש המבוצעות על הדפדפן. ניתן להעביר לדפדפן מגוון רחב של פקודות וכן ניתן לבדוק קיומו של טקסט ספציפי וכן אלמנטים שונים. ניתן לכתוב בצורה פשוטה וקלה אפליקצית java שיוצרת קשר עם השרת selenium-rc ומעבירה לו פקודות בעזרת API של java. לצורך עבודה יותר יעילה עם הכלי הזה קיימים כמה תוספים שמותאמים (כרגע) רק עבור פיירפוקס כגון:

SeleniumIDE – כלי עזר המאפשר כתיבה והרצה של סקריפטים של סלניום דרך הדפדפן בשלבי הפיתוח הראשונים (לפני שעושים אינטגרציה ושיפורים במסגרת פלטפורמת האוטומציה). כלי זה גם מאפשר לסמלץ קוד של סקריפט שנכתב בכמה שפות פיתוח שונות כגון: JAVA, PHP, PERL, GROOVY וכו'

FIREBUG – כלי מוביל לכל מי שעוסק בפיתוח WEB. הכלי מאפשר בחינת אובייקטים בדפי HTML ומתן מאפיינים עבורם, משמש ככלי ל DEBUG של קוד JAVASCRIP.

XPATH checker – כלי אינטראקטיבי שמאפשר בדיקת ביטויי XPATH ווידוא תקינותם. הכלי מאפשר עריכה ואיבחון של הביטויים עד להבאתם למצב שבו זיהוי האלמנט הוא חד-ערכי וגנרי ככל האפשר.

כתובות האתרים הרשמיים:
http://seleniumhq.org
http://getfirebug.com
https://addons.mozilla.org/en-us/firefox/addon/xpath-checker

כלי חינמי נוסף הנקרא AUTOIT – זהו כלי סקריפטים המאפשר אוטומציה עבור WINDOWS GUI וכן יכולות סקריפטים גנריות נוספות (כגון מניפולציות על קבצים, הרצת אפליקציות אחרות, יצירת GUI עבור כל מיני צרכים גנריים וכו'). הכלי מסמלץ פעולות מקלדת-עכבר כדי לבצע פעולות על אפליקציות חלוניות (לדוגמא: Windows Explorer, CALCULATOR, NOTEPAD וכו'). הכלי תומך בכל גרסאות מערכת ההפעלה WINDOWS ולא דורש קינפוג מיוחד. לכלי שפת סקריפטים קלה ללמידה המאפשרת יכולת שימוש ללא הכשרה מיוחדת. הכלי מגיע עם עזרה מאוד ידידותית וישנה גם חשיפה די גדולה ברשת לכלי הזה כך שאפשר למצוא פתרונות לבעיות בעזרת חיפוש קצר.

כתובת האתר הרשמי: http://www.autoitscript.com/site

פלטפורמה לפיתוח אוטומציה והרצתה הנקראת FIT או FITNESSE (שזה גרסאת FIT (המנוע) עם התוספת של ה WIKI/WEB SERVER) –
הפלטפורמה מאפשרת למפתחים ולבודקים לפתח בדיקות פונקציונאליות, Sanity, ATP וכדומה בצורה קלה וללא צורך בניסיון פיתוחי עשיר. הפלטפורמה מאפשרת כתיבת טסטים, ניתוחם בקלות בעזרת ממשק WIKI ידידותי וקל לעריכה (במידה ועובדים עם FITNESSE).

כתובות האתרים הרשמיים:
http://fit.c2.com/wiki.cgi?IntroductionToFit
http://fitnesse.org/FitNesse

כלי קוד פתוח נוסף לבדיקת אפליקציות SWING ו JFC הנקרא JEMMY –
הכלי מאפשר זיהוי אובייקטים וביצוע פעולות עליהם כגון: זיהוי חלונות, לחיצה על כפתורים, הקלדת טקסט בתיבות טקסט וסימולציה עבור כל פעולת משתמש אחרת על האפליקציה החלונאית. האתר הרשמי כבר לא פעיל משום מה...

ניהול תצורה ובניית גרסאות ליליות - HUDSON ו- MAVEN – השילוב בין 2 כלי ניהול התצורה מאפשר בניית גרסאות והרצת טסטים אוטמטיים בתיזמון יומי/לילי (דבר שאנו מכנים אותו Continuous Integration):

HUDSON מאפשר לנו שירותי: בנייה מתוזמנת של פרוייקטים, התקנה אוטומטית של הגירסא והרצת טסטים אוטומטיים לאחר ההתקנה. HUDSON קל להתקנה וקינפוג ומאפשר דיווח על תוצאות באופן ידידותי למשתמש כמו כן שליחת תוצאות באימייל / RSS ע"פ דרישה. אתר הבית: http://hudson-ci.org

MAVEN - כלי לניהול ובניית פרוייקטי תוכנה. הכלי מאפשר ניהול "חכם" ומתודולוגי של משאבי הפרוייקט וניהול התלויות של הפרוייקטים במשאבים אלה (הניהול נעשה בעזרת קבצי POM (Project object model) ומאגרי משאבים (Maven repositories). MAVEN מאפשר התממשקות לכל מיני התקנים כמו כן מאפשר התממשקות אליו כפי ש HUDSON עושה זאת להרצה של משימות הכתובות ב- MAVEN. אתר הבית: http://maven.apache.org/index.html

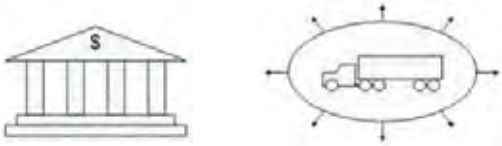
האתגרים בהם נתקלנו (כמו שכנראה כל גוף תוכנה סטנדרטי ייתקל בהם) בבחירת הכלים לצורכי בדיקות אוטומציה:
- פלטרפורמת אוטומציה שאינה תלויית מערכת הפעלה.
  - שימוש מרובה בפלטפורמה על כל הפרמוטציות הנדרשות (הפלטפורמה שנבחרה היא חינמית ואינה מוגבלת במספר המשתמשים וכמו כן תומכת בהרצה על כל מערכות ההפעלה בעצם היותה מבוססת ג'אווה).
  - התממשקות לכלים חיצוניים (כגון: BadBoy, AutoIT, Selenium, SoapUI וכו').
- עלות - כפי שהוזכר הפלטפורמה היא חינמית.
- דוקומנטציה – הפלטפורמה באה עם תיעוד מלא ומגוון דוגמאות להתחלת עבודה ללא קושי (כמו כן, עם ההתקנה מגיע פרויקט עזר לדוגמא שמאפשר הבנת מבנה הפרויקט הנכון והמתבקש).
- עזרה ברשת – פורום פעיל ועשיר בשאלות ותשובות - http://sourceforge.net/projects/jsystemtest/forums
- יכולות דיווח על תוצאות ריצה – הפלטפורמה מגיעה עם אפליקציית דיווח שרצה בשרת ומאפשרת צפייה וסינון תוצאות ריצה בקלות ויצירת דוחות להשוואה בין ריצות קודמות.
- קלות ניטור הרצות וניתוח תקלות – הכלי נותן יכולות מובנות לנטר ריצה ולעצור בכל נקודה שנדרשת על מנת להבין תקלות ולגלות שגיאות.
- התממשקות לכלי תצורה – עצם היותו של הכלי מבוסס ג'אווה והעבודה איתו נעשית בסביבת פיתוח סטנדרטית (ECLIPSE וכדומה) ההתמשקות לכלי ניהול תצורה היא טריוויאלית (אצלינו אנחנו מתממשקים לכלי דרך, HUDON MAVEN ומנהלים גירסאות דרך PERFORCE).

חסרונות שגילינו עד כה בשימוש בפלטפורמה:
- כתיבת התשתיות והמתודות הראשיות בכל פרויקט אוטומציה (כמו כן כתיבת הטסטים) דורשות יכולות פיתוח בג'אווה של 3-1 שנות ניסיון.
- התממשקות של הפלטפורמה לכלי ניהול הבדיקות הפופולריים (QC וכדומה) אינם טריביאליות ודורשות פיתוח מיוחד.
- תוספות פונקציונאליות לפלטפורמה דורשות פיתוח פנימי שכן זהו כלי קוד-פתוח והדרישות הנוספות הם על אחריות המשתמש.

איך מתבצע החיבור בין כל הכלים המוזכרים במאמר אצלינו בארגון?
כדי לתאר את התהליך הקורה החל מבניית המוצר, התקנתו, כתיבת תרחישי הבדיקה, הרצת הבדיקות האוטומטיות ודיווח התקלות אראה תרשים זרימה שבו בכל שלב מתוארת הפעולה והכלי האחראי לביצוע של אותה פעולה. בכללות כל הקשור להרצת הבדיקות ודיווח התוצאות נעשה ע"י הפלטפורמה JSYSTEM – כתיבת הבדיקות נעשית דרך QC (בנינו כלי פנימי המאפשר קישוריות בין JSYSTEM ל QC ובעזרתו אנחנו מעדכנים את תוצאות ההרצות ב QC), דיווח התקלות נעשה דרך JIRA וניהול הגרסאות דרך PERFORCE. ההרצות הליליות ובניית הגרסאות נעשות דרך HUDSON ו MAVEN. בעזרת התרשים אפשר לראות את כל הכלים בשימוש לפי סדר התרחשותם בתהליך האוטומציה:

QA to provide test plan - CC
Automation to design and implement - ECLIPSE
Automation to provide final package to QA

QA to run automation test and track results - JSYSTEM
QA to validate automation package, report bugs - JIRA


Version Control - PERFORCE
Continuous integration and hightly builds - Hudson & MAVEN

QAjobs.co.il

פורטל משרות
ה-QA של ישראל

לוח דרושים ייחודי

המתמקד בשוק הבדיקות
של ישראל ומציע פרסום
וחיפוש משרות באופן
ממוקד, יעיל ומהיר.

כל משרות הבדיקות
וה-QA באתר אחד.

בודק תוכנה, כאן תמצא
את התפקיד הבא שלך!

ההשקה בקרוב!