



מיכאל שטאל

הוא ארכיטקט בדיקות תוכנה באינטל, ישראל. במשך 17 השנים האחרונות מיכאל בדק מוצרים בתחום Wi-Fi, טלויזיות חכמות, כרטיסים גרפיים, אפליקציות ראייה ממוחשבת מבוססות על מצלמות תלת-מימד, ולאחרונה הוא עובד בקבוצה שבודקת את התוכנה שמאחורי טכנולוגיית ניהול אקטיבי ומנוע האבטחה (CSME) של אינטל.

במסגרת תפקידו, מיכאל מגדיר שיטות בדיקה ומתודולוגיות עבודה, עוסק הרבה בהדרכה ולפעמים אפילו מרשים לו לבדוק משהו (שזה הכי כיף).

מיכאל מציג תכופות בכנסים בארץ ובח"ל והציג בין היתר

ב- STAR conferences, QA&Test, SIGIST Israel ובכנסים אחרים. מיכאל מלמד בדיקות תוכנה בפקולטה למדעי המחשב באוניברסיטה העברית. ניתן לראות חלק מהמצגות והמאמרים שלו באתר www.testprincipia.com.

כל פעם אותו סיפור

לפני שעברתי לבדוק תוכנה, עבדתי במפעל היצור של אינטל בירושלים. גם שם עסקתי בבדיקות – אבל כחלק מעולם הייצור. העולם השני של אינטל הוא עולם הפיתוח. "R&D" למי שמעדיף את השם היותר-יוקרתי. הרבה מההילה של פיתוח נובע מזה שבהגדרה עושים שם משהו חדש. משהו שלא היה קיים קודם. ממציאים.

האמת היא שגם בעולם הייצור המצאנו לא מעט דברים – אבל זה לא הנושא היום. אני רוצה להתמקד יותר במיתוס שבפיתוח עוסקים בחדשנות.

אז האמת היא שכן – אכן עוסקים בחדשנות. אבל השאלה היא באיזה מינון. אם נחפור קצת פנימה נראה שהחלק של ההמצאה והחידוש הוא יחסית נמוך. מרגע שיש רעיון חדש והוא נכנס לתוכנית העבודה – מכאן והלאה רוב המאמץ הוא בביצוע פעולות שכבר עשינו הרבה פעמים בעבר.

למשל: עדכון הקושחה (firmware) במערכת משובצת מחשב (embedded system). כל פעם אותו הדבר. כל האתגרים, כל הבעיות וכמובן כל הבאגים חוזרים על עצמם. כנ"ל תוכנת ההתקנה. כנ"ל התממשקות לבסיס נתונים, וכו' וכו'. אם לא אנחנו פיתחנו משהו דומה בעבר, אז מישהו אחר עשה זאת וכבר כתב מסמכי בדיקות, וכבר פיתח מקרי בדיקה. ובכל זאת, הנה אנחנו משקיעים מאמץ בהמצאה חוזרת של הגלגל.

זה נראה די מיותר. סביר יותר למצוא דרך להעביר את הידע הנצבר מדור לדור בין קבוצות או אפילו בתוך הקבוצה – בין בעלי הניסיון למהנדסים החדשים.

אחת השיטות להעברת ידע על בדיקות נקראת "טקסונומיה של באגים" (בעברית זה יהיה משהו כמו "רשימות מיון של פגמים"). הרעיון מופיע מזמן בספרות – ראו רשימת מקורות לדוגמה בסוף הטור. הרעיון הוא לייצר רשימות של קטגוריות של באגים שנמצאו בעבר, ולהמשיך לפרט באגים אפשריים בכל קטגוריה. ישנן מספר אפשרויות לשימוש ברשימות אלה:

- גירוי החשיבה של בודקים מנוסים עם עוד רעיונות של "מה צריך לבדוק"
- לתת לבודקים חסרי ניסיון רשימה של דברים שצריך לבדוק מבחינה פונקציונאלית ולהציף אספקטים של המוצר שאולי לא הבינו שצריך לבדוק (כמו עומס; שימושיות)
- הערכת הכיסוי של תוכנית הבדיקות – בחירה מתוך הרשימה וחיפוש האם תוכנית הבדיקות מכסה מקרה זה הנה למשל קטע קצר שתרגמתי מתוך הרשימה שמופיעה בספר Testing Computer Software – ספר קלאסי ומומלץ:

ביצועים

התוכנה איטית יש עיכוב בין הקלדת אות ועד שהיא מופיעה על המסך תגובה איטית אי אפשר לתקתק יותר מהר ממה שהמחשב מספיק להציג אין התראה שפעולה מסוימת תיקח זמן רב אין חייוו על התקדמות של פעולה ארוכה בעיות של time-out

כל פעם אותו הדבר...

כל האתגרים, כל הבעיות וכמובן כל הבאגים חוזרים על עצמם

חלק ממה שכתוב כאן נראה מובן מאליו ("התוכנה איטית") – ואילו דברים אחרים בהחלט מעלים רעיונות לבדיקות שאולי לא עשינו. למשל "התרעה שפעולה מסוימת תיקח הרבה זמן": זה מעלה את המחשבה איזו מהפעולות הנתמכות בתוכנה הובדקת עלולות לקחת זמן רב, באיזה מצבים זה יקרה – ומשם למקרה בדיקה שמייצר את המצב, ובדוק את תגובת המערכת.

עוד דבר מעניין הוא שבהכרח לחלק מהרעיונות שכתובים ברשימות כאלה יש "זמן תפוגה". למשל, הנקודה על תקתוק במהירות שמעל קצב ההצגה למסך כבר לא כל כך רלוונטית להיום. במחשבים ישנים זה היה מצב רגיל מאוד ומי שידע להקליד מהר – אפילו לא מהר מאוד – הצליח "לעבור" את קצב ההצגה למסך.

אישית, למרות שאני מכיר מספר רשימות כאלה, מעולם לא השתמשתי בהם. ניסיתי פה ושם בחצי חשק אבל אני מודה שלא התאמצתי. אז מצד אחד אני חושב שהרעיון נכון, ומצד שני...



לא נוח להשתמש ברשימות האלה. ה"ראש" של מי שכתב את הרשימה לא בהכרח (יותר נכון: בהכרח לא) תואם את צורת המחשבה שלכם והתוצאה היא שדברים שאתם הייתם מצפים לראות תחת סעיף מסוים, מופיע תחת סעיף שלא מתקשר לכם לעניין. למשל, בקטע שהבאתי, המחברים בחרו לשים את "חיווי על התקדמות" תחת "ביצועים". אני הייתי מצפה לראות אותו תחת "חוויות המשתמש". גם הניסוח, שמנסה להיות קצר ומתמצת, לפעמים לא מובן. התוצאה היא שכדי להשיג תועלת מרשימת טקסונומיה עבור תחום מסוים, צריך לעבור על חלקים גדולים מהרשימה – גם אלה שתחת כותרות שלא נראות לכם קשורות, לסגן המון הדברים שלא רלוונטיים עבורכם ולקבל את זה שחלק ממה שכתוב שם לא יהיה מובן. חוויה מפוקפקת.

אז יש לנו רעיון טוב, אבל לא ממומש בצורה נוחה עבורנו. האם לוותר עליו?

אחת האפשרויות שמניסו אני יודע שעובדת היא לייצר רשימה כזו בעצמכם, עבור דברים שאתם בודקים באופן חוזר. כיוון שאתם כותבתם את הרשימה, היא ודאי הגיונית עבורכם, ועל כן יש סיכוי גבוה שתשתמשו בה.

לי יש רשימה אחת שעשיתי לפני זמן רב, המתארת מה צריך לבדוק על API. פה ושם אני לומד משהו חדש שצריך לכסות ומעדכן את הרשימה. לא – אני לא אחלוק אותה אתכם, מסיבה פשוטה: היא כתובה בפורמט שנהיר לי, אבל לא יהיה מיידית ברור לאדם אחר. עבורי הרשימה בהחלט בעלת ערך ויצא לי להשתמש בה פעמים רבות. חלקתי אותה עם אחרים כשיכולתי לעבור עליה איתם לוודא שהיא מובנת להם. אני גם מכיר מספר רשימות כאלה שיוצרו על ידי אחרים בחברה שלי, לצרכים שלהם.

זה לא בדיוק מדגם סטטיסטי, אבל שתי האנקדוטות האלה רומזות על כיוון אפשרי: הרעיון של רשימות באגים הוא הגיוני ושימושי – אם (אולי אפילו "ורק אם") אלה רשימות שפיתחתם בעצמכם.

אפשר לפתח רשימה על ידי איסוף אישי של פרטים. אפשר לעשות עבודה בצוות. ואפשר גם לקחת נשימה עמוקה ולעבור על רשימות קיימות ולשלוף מהן דברים שנראים לכם מתאימים.

אגב: אם מתארגנים בצוות – וזה לא צריך להיות מאמץ של יותר מפגישה אחת או שתיים – מאוד כדאי לנסות "לגייס" גם מפתחים ומנהלי פרויקט להשתתף במאמץ. נוכחות כזו תביא נקודת מבט אחרת לדיון וגם תעלה את המודעות של המפתחים והמנהלים לבעיות שמופיעות תדיר, דבר שיכול לעזור במניעת בעיות כאלה בעתיד.

אז בהצלחה, ואם מצאתם משהו יעיל – או שפיתחתם משהו שנראה לכם שכן יהיה מובן לקהל רחב – זה יהיה נחמד אם תפרסמו אותו.



דוגמאות ומקורות

קישור לדוגמה שחשבתי שהיא ממש טובה, התפרסם לאחרונה בדף הפייסבוק של [פורום בדיקות ואיכות תוכנה](http://www.flylib.com/books/en/2.156.1.108/1/). <http://www.flylib.com/books/en/2.156.1.108/1/> זו גם דוגמה לשימוש ב-mind map ליצירת רשימת רעיונות לבדיקה. bettertesting.co.uk/content/wp-content/uploads/2011/09/login-test-ideas.png (תודה לחיים טל).

רשימות של באגים אפשר למצוא בספרים הבאים

- (Kaner at al.) Testing Computer Software
- (Beizer) Software Test Techniques
- (Whittaker) How to break software
- Binder's Object-Oriented Taxonomy

מאמר מקיף על הנושא

(Giri Vijayaraghavan , Cem Kaner) Use Them to Generate Better Tests :Bug Taxonomies

קישור:

http://www.testingeducation.org/articles/bug_taxonomies_use_them_to_generate_better_tests_star_east_2003_paper.pdf

רשימות יותר מודרניות ובמידה מסוימת גם משעשעות:

<http://www.kalzumeus.com/2010/06/17/falsehoods-programmers-believe-about-names/>

<https://github.com/minimaxir/big-list-of-naughty-strings>

