ניהול תצורה בפרויקטים קטנים

(או פרויקטים גדולים בראשית דרכם)

שלומית מורד

© כל הזכויות שמורות לשלומית מורד

הגדרה

- תצורה של מערכת תוכנה היא גירסא של המערכת (קוד מקור ותוצרים סופיים) + כל המידע הרלוונטי לגירסא זו:
  - כל תוצרי הביניים (מסמכים, מודלים ואבי-טיפוס)
  - כל תוצרי הבדיקות ואבטחת האיכות
  - תוכנות עזר (סימולטורים למשל)
  - כל כלי הפיתוח
  - נתונים שהם חלק מהמערכת
  - תעוד ו- manuals
- ניהול תצורה הוא שם כולל לשיטות המשמשות לשליטה, בקרה ומעקב אחר התפתחות המערכת

© כל הזכויות שמורות לשלומית מורד

מטרות

- ניהול תצורה מורכב משני תחומים בעלי קשר חזק ביניהם:
  - ניהול גירסאות
  - ניהול שינויים
- ניהול גירסאות נועד ל:
  - ניהול מספר מהדורות הנתמכות בו זמנית
  - ניהול גירסאות של תוכנה המשרתת מוצרים שונים תחת product-line
  - ניהול גירסאות תוכנה שעובדות תחת תשתיות/טכנולוגיות חליפיות
  - ניהול גירסאות ספציפיות ללקוח/ות
  - ניהול גירסאות ביניים בזמן פיתוח
- ניהול שינויים נועד ל:
  - אבטחה שכל השינויים במוצר מתבצעים באופן מבוקר ומתואם
  - ניהול בעיות (באגים)

© כל הזכויות שמורות לשלומית מורד

אינטגרציה

- גירסא של המוצר (גם אם נכלול את כל מוצרי הביניים) אינה תצורה
- תצורה כוללת גם את כל המידע על השינויים המתבצעים במערכת
- לכן רק האינטגרציה של שני התחומים מספקת ניהול של תצורה אמיתית
- האינטגרציה מספקת גם כלים מתקדמים לניהול הפיתוח: לתכנון עבודה ובקרה על התקדמותה
- האינטגרציה מספקת נירואות (Visibility) טובה יותר של סטטוס המוצר

© כל הזכויות שמורות לשלומית מורד

סטטוס כללי

- חלק משמעותי מכלי ה-SCM עונים רק על ניהול הגירסאות
- רוב היישומים בתוכנה (בלי קשר לכלי) הם רק של החלק הזה
- ברוב החברות יש ניהול באגים של תוכנה (ולפעמים בקשות פיתוח) בכלי נפרד שאינו מחובר ומשולב בכלי ניהול הגירסאות
- בפרויקים קטנים או בפרויקטים גדולים בראשית דרכם (כשהצוות קטן) יש נטיה להשקיע רק את המינימום ההכרחי: להקים מערכת בסיסית באמצעות כלים בסיסיים
- בפיתוח קושחה (Firmware) ניהול תצורה אינו נפוץ; במקרים שקיים ניהול גירסאות, אין ניהול משותף עם מערכת התוכנה (למרות הממשקים ההדוקים ביניהם)

© כל הזכויות שמורות לשלומית מורד

סטטוס בפרויקטים קטנים

- ניהול תצורה בסיסי המתקיים בגופי הפיתוח הקטנים מוגבל:
  - שימוש בכלי לניהול גירסאות (ולא ניהול תצורה) - OSS בדרך כלל
  - ה-PLC של הפיתוח אינו משוקף בתהליכי ניהול תצורה
  - אין מדיניות ניהול תצורה (עבודת צוות, branch policy, baseline policy, version numbering, וכו')
  - אין הגדרת תפקידים בניהול תצורה
  - פיקוח על הכנסת שינויים נעשית מחוץ לכלי/מתודולוגית ה-CM
  - תהליך שחרור גירסא הוא תהליך ידני שלא משתמש בכלי ה-CM
  - מערכת ה-build מינימלית ואינה אוטומטית

© כל הזכויות שמורות לשלומית מורד

סיבות למצב הקיים

- חסכון בזמן וכסף
  - הגדרת נהלים ומדיניות CM לוקחת זמן
  - פעילויות ניהול ובקרה גוזלות מזמן הפיתוח
  - קנית כלי מתקדם עולה כסף רב ודורשת זמן רב להטמעה
- חוסר מודעות
  - מנהלי תוכנה לא מנוסים שלא מבינים לא את הצורך המיידי ולא את ההשלכות לטווח הארוך
  - אנשי חומרה שמנהלים צוותי תוכנה
  - אנשי חומרה שכותבים תוכנה

© כל הזכויות שמורות לשלומית מורד

מה הבעיה (במצב הנתון) ?

- יש הסטוריה של הקבצים אבל לא תמיד של גירסאות שלמות
- שילוב תוצרים של מספר מפתחים (אינטגרציה) נמשכת זמן רב
- ייצוב גירסא נמשך זמן רב מדי
- אין תעוד (בתוך המערכת) של השינויים שבוצעו בתוכנה
- יתכנו שינויים לא מבוקרים בקוד (שיפגעו באיכות התוכנה ו/או לו"ז)
- נפגעת המודולריות של הקוד (כי אין ב-CM תמיכה בארכיטקטורה)
- העברת תוכנה ל-QA אינה מסודרת או עקבית
- תשוחרר גירסא לא נכונה, לא שלמה או לא קוהרנטית
- בנית התוכנה בשלמותה (ידנית למחצה) גוזלת זמן רב

© כל הזכויות שמורות לשלומית מורד

מה הבעיה (בהמשך) ?

- כאשר הצוות גדל:
  - סנכרון עבודה של מפתחים יהיה מסורבל וארוך
  - אינטגרציות ארוכות וזמן ממושך להתייצבות גירסא
  - פיתוח לטווח ארוך יפריע למשימות לטווח קצר
  - חוסר עיקביות בבית גירסא
  - "העלמות" של פיתוח מגרסאות קודמות
- החל מגירסא שניה ואילך:
  - קושי באיתור (מדויק) של גירסאות ישנות
  - מסירה של גירסא לא נכונה ללקוח
  - שחרור גירסא לא נכונה (לא אחרונה או לא מיטבית)
  - חוסר יכולת לדעת במדויק מה תכולתה של גירסא

© כל הזכויות שמורות לשלומית מורד

כמה זה יעלה ?

- הגדרת מדיניות ופרוצדורות עבודה חדשות
  - הדרכת המפתחים
  - העמסת תפקידי CM על המפתחים
  - שינוי הרגלי עבודה קיימים
  - התפשרות על פרוצדורות נחותות כדי להקל על השינוי
- החלפת כלי CM
  - העברת כל התוכנה מכלי לכלי (בדר"כ מלווה באובדן היסטוריה)
  - העברת דווחי באגים (ובקשות שינוי) לכלי החדש או יצירת אינטגרציה בין כלי הגירסאות החדש וכלי ניהול השינויים הקיים
  - הדרכת מפתחים
  - התפשרות על כלים נחותים כדי להקל על השינוי

© כל הזכויות שמורות לשלומית מורד

שביל הזהב - Scalability

- "סוף מעשה במחשב תחילה" - להשקיע מחשבה בראשית הפרויקט על מנת להקטין את העלויות של השינוי העתידי
- כמו שאנו חושבים על scalability בתוכנה שאנו מפתחים, צריך לחשוב על scalability במערכת ה-CM
  - להגדיר שיטות עבודה רצויות ו-policies בטווח קצר וארוך
  - לבחור את הכלים המתאימים לטווח הארוך
  - אם לא מעשי לרכוש את הכלים בשלב זה, יש לבחור את הכלים הבסיסיים לטווח הקצר כך שיאפשרו מעבר עתידי במאמץ סביר

© כל הזכויות שמורות לשלומית מורד

שאלות ?

© כל הזכויות שמורות לשלומית מורד