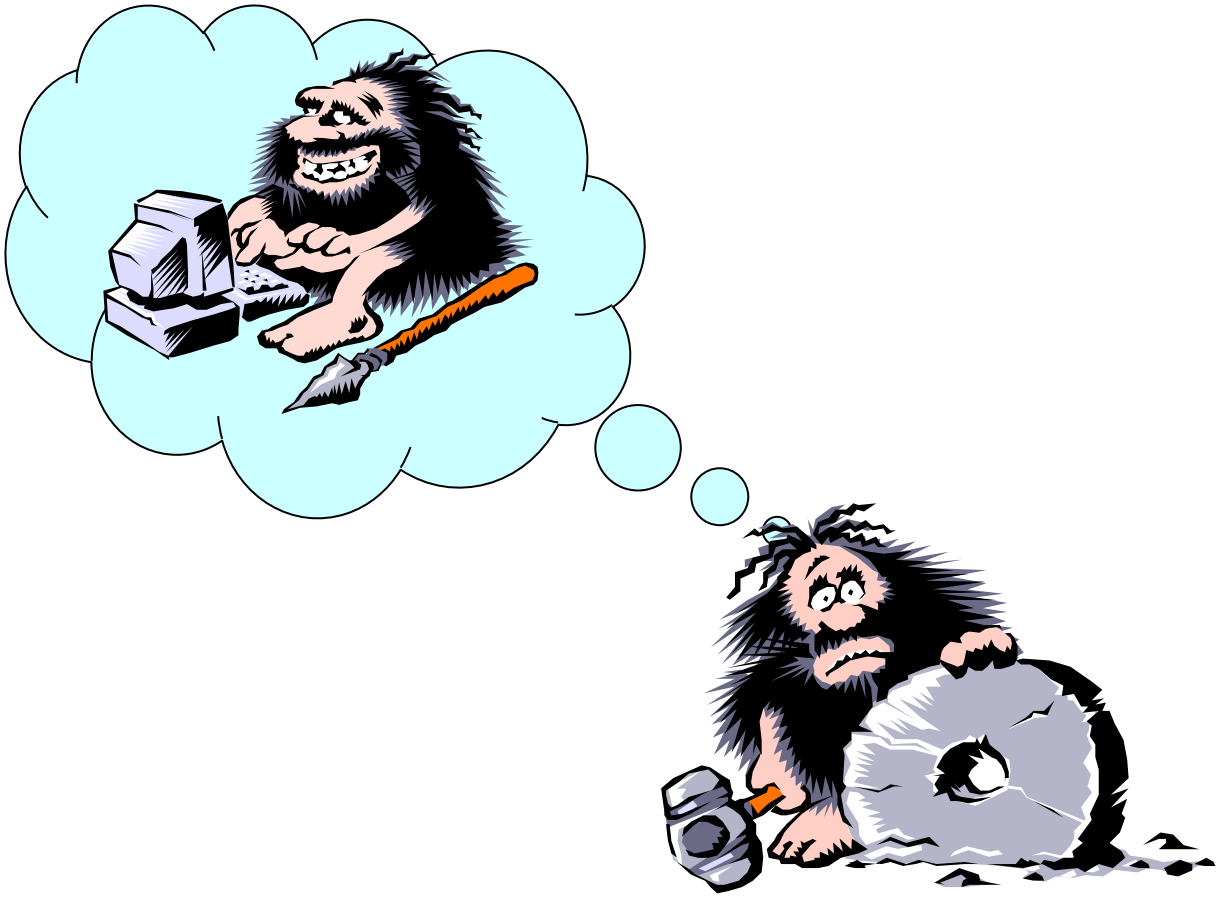
הנדסת מערכות עתירות תוכנה



מטרות הקורס

- יעוד
  - הקורס נועד להקנות ידע והבנה, רלוונטיים למהנדסי מערכת, לגבי התהליכים והשיטות שבפיתוח ותחזוקה של מערכות עתירות תוכנה, כבסיס לתכן מערכתי ולקבלת החלטות ניהוליות וטכניות.
- התועלות הצפויות
  - הבנת הייחודיות שבפיתוח מערכות עתירות תוכנה
  - הכרת מחזור החיים האופייני (פעילויות ותוצרים) של פיתוח מערכת עתירת תוכנה
  - הכרת שיטות וכלים לפיתוח מבוסס מודלים (Model-Based Development) של מערכות עתירות תוכנה
  - **שיפור האופן בו נערכים מפרטי המערכת לטובת מימוש התוכנה**
- מאפייני הקורס
  - גישה מערכתית
  - רלוונטיות ועדכניות
  - יישומיות

רוב המערכות המפותחות כיום בעולם הן עתירות תוכנה

More Software Code in Chevrolet Volt Car than Boeing 787

You might be surprised at the amount of software code that is in your modern car. The Chevrolet Volt has more lines of code than a Boeing 787. The Volt has 10 million lines of code. The Boeing only 8 million lines.

[1] Granted, the Chevrolet Volt is a plug-in hybrid, so it needs extra electronics for that system.

Text source: http://blissfulwriter.hubpages.com/hub/Software-Code-in-Your-Car
Picture source: http://www.ramanmedianetwork.com/ibm-software-drives-chevrolet-volt/

סוגים של מוצרי תוכנה

- Created for a specific Custom •
  - פיתוח עבור לקוח ספציפי (turn-key solution).
- Generic
  - מוצר תוכנה הנמכר בשוק החופשי.
  - נקרא גם Packaged, ready made
    או COTS (Commercial Off The Shelf)
- Embedded
  - חלק מגוף המערכת, משולבת עם חומרת המערכת. לרוב במערכת זמן אמת.

תוכנה

ע"פ ה-IEEE (זהה ל- ISO/IEC 9000-3)

Software is:

Computer programs, procedures, and possibly

associated documentation and data pertaining to the operation of a computer system.

הגבלת אחריות למוצרי תוכנה

מתוך הסכם רשיון למשתמש קצה עבור תוכנת *Microsoft*

7. **הגבלת אחריות.** האחריות המוגבלת המתוארת להלן היא האחריות המפורשת היחידה הניתנת לך, *והיא ניתנת במקום* כל אחריות מפורשת אחרת (אם קיימת) שנוצרה על-ידי תיעוד או אריזה כלשהם. למעט האחריות המוגבלת ועד למידה המירבית המותרת בחוק החל, חברת *Microsoft* *וספקיה מספקים לך את מוצר התוכנה ושירותי תמיכה (אם ניתנים)* במצב כפי שהוא (is as) ועם כל הפגמים, ומסירים מעצמם בזאת כל אחריות ותנאים אחרים, בין אם מפורשים, משתמעים או סטטוטוריים, כולל אך לא מוגבל לכל אחריות, חובות או תנאים משתמעים (אם קיימים) לגבי סחירות, **התאמה לתכלית מסוימת, שלמות או מידת הדיוק של תגובות, תוצאות, מקצועיות הביצוע, היעדר וירוסים והיעדר רשלנות, כל זאת ביחס *למוצר התוכנה*** *ולמתן או אי-מתן שירותי תמיכה. כמו כן, אין כל אחריות או תנאים לגבי בעלות,* הנאה ללא הפרעה, החזקה ללא הפרעה, התאמה לתיאור או אי-הפרת זכויות לגבי *מוצר התוכנה.*

כל יצרני תוכנה אינם מתחייבים שהתוכנה שהם מייצרים נקייה מפגמים, דבר רגיל עבור יצרני חומרת מחשבים.

ויחד עם זאת...

- "באג 2000" (Y2K) – האירוע המביך של האלף הקודם:
  - לקראת סוף המילניום "מגלה" העולם שכמויות אדירות של תוכנה נכתבו בהתבסס על ערך דו-ספרתי ל"שנה", והמעבר מ-99 ל-0 עלול להיות הרסני
  - תקציבי ענק מושקעים באיתור ותיקון התוכנות, תוך "גישוש באפילה"
  - התוצאה – מזערית!
    - האם עקב ההשקעה בתיקון או בגלל שהפאניקה היתה מוגזמת?

ארועים מביכים נוספים הקשורים בתוכנה

- יוני 1996 - הלווין "אריאן" מתפוצץ 40 שניות לאחר השיגור
  - אריאן 5 הוא משגר לווינים כבד פרי פיתוח של סוכנות החלל האירופאית. שיגור הבכורה של המשגר הסתיים בכשלון.
  - הסיבה: המרת ערך מספרי גבוה ממשתנה מסוג real (64 ביט) למשתנה מסוג integer (16 ביט) בחישובי המהירות
  - תקלה זו היוותה ציון דרך בפיתוחן של מערכות בדיקה לתוכנות זמן אמת.
  - http://www.around.com/ariane.html
- פברואר 1991 – טיל פטריוט אמריקאי מחטיא טיל סקאד עיראקי, הגורם ל-28 הרוגים
  - הסיבה: טעות מצטברת בחישוב עקב "קיצוץ" הערך 0.1 ל-24 ביט
  - http://www.ima.umn.edu/~arnold/disasters/patriot.html
- ספטמבר 1999 – חללית מחקר לאקלים המאדים אובדת בחלל
  - הסיבה: שימוש מעורב ביחידות משקל – pounds/KG
  - http://www.iki.rssi.ru/jplmirror/mars/msp98/orbiter
- נובמבר 2000 – 8 אנשים מתים מעודף קרינה ממכשיר הקרנה במכון הלאומי לסרטן בפנמה-סיטי
  - הסיבה: הרופאים סימנו, בחלק מהמקרים, את איזור הקרינה נגד כיוון השעון במקום בכיוון השעון. הסימון ההפוך גרם לתוכנה להכפיל את כמות הקרינה
  - http://www.scielosp.org/pdf/rpsp/v20n2-3/14.pdf
- ינואר 2010 – "באג 2010" בגרמניה – אלפי כרטיסי אשראי מפסיקים להגיב
  - הסיבה (לא ידועה לפרטיה): התוכנה הצרובה על הכרטיס לא מזהה נכון את שנת 2010
  - http://www.dw-world.de/dw/article/0,,5088075,00.html

- מקורות נוספים:
  - http://www.cs.tau.ac.il/~nachumd/verify/horror.html
  - http://www5.in.tum.de/~huckle/bugse.html
  - http://catless.ncl.ac.uk/Risks

ואצלנו ...

- דצמבר 2010 – רשת "סלקום" מושבתת כמעט לחלוטין למשך 12 שעות
  - הסיבה: סקריפט עדכון שהורץ בלילה הקודם השחית נתוני לקוחות במערכת הראשית. גם נתוני הלקוחות במערכת הגיבוי המקבילה הושחתו, כי מישהו "התעצל" לנטרל את סינכרון המערכות לפני הרצת הסקריפט.
  - http://it.themarker.com/tmit/article/13276
- ינואר 2011 – רשת "בזק" מושבתת כמעט לחלוטין למשך 4 שעות
  - הסיבה: "באג" בתוכנת הניהול של אחד המתגים יצר עומס מדומה על הרשת.
  - http://ladaat.net/forum/index.php?topic=53107.0

תקלות מופלאות

	בזק	סלקום
מועד התקלה	25 בינואר 2011	1 בדצמבר 2010
מספר לקוחות החברה, במיליונים	2.4	3.3
משך התקלה, בשעות	4	12
פיצוי כספי	לא נקבע	66 מיליון שקל

רק לאחרונה...

- הודעת הטקסט שגרמה ל-iOS לקריסה מוחלטת

- הודעת הטקסט שגרמה ל-skype לקריסה מוחלטת

http://:

ולא מדובר רק באמינות ובאיכות...

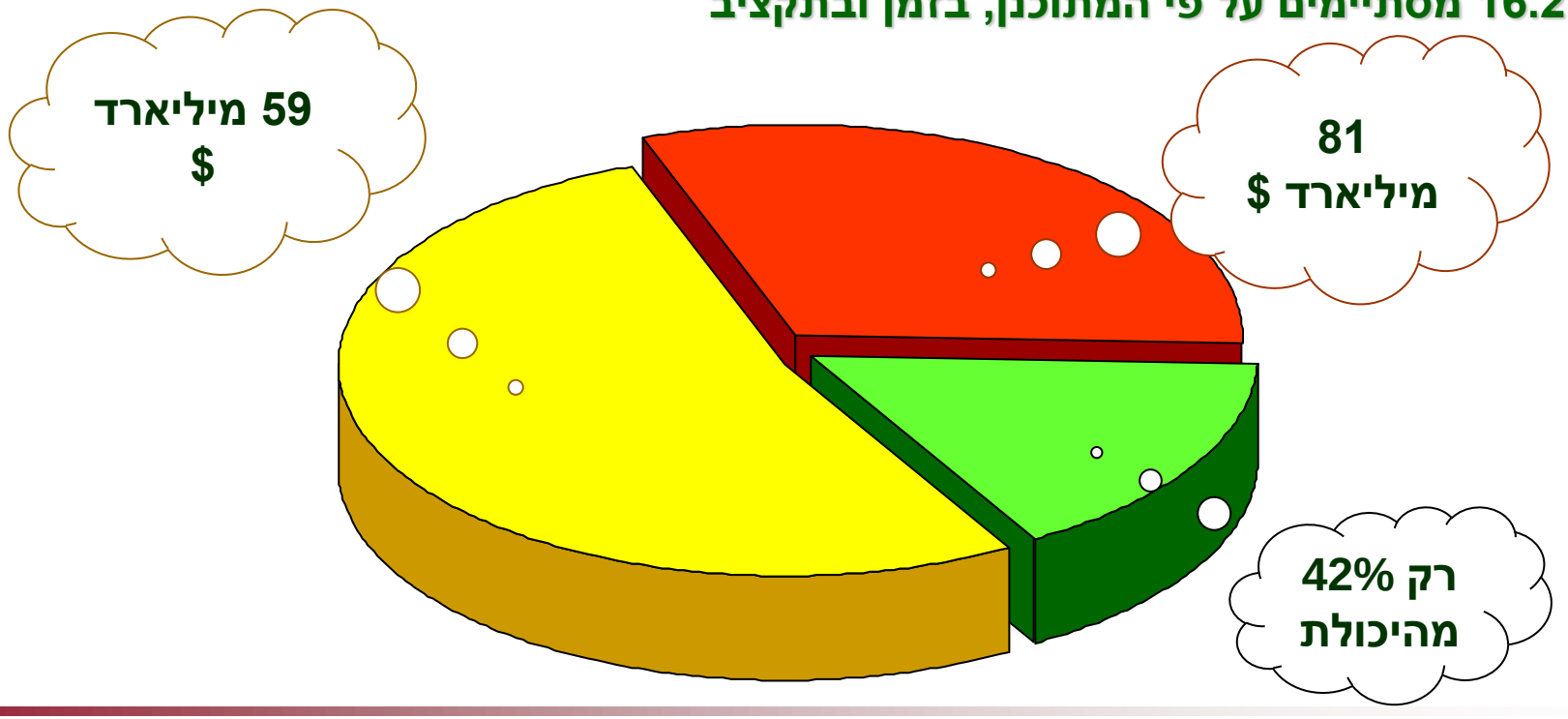
- The “Chaos” Report (Standish Group 1995):

מתוך כלל פרוייקטי התוכנה בחברות אמריקאיות גדולות, בינוניות וקטנות:

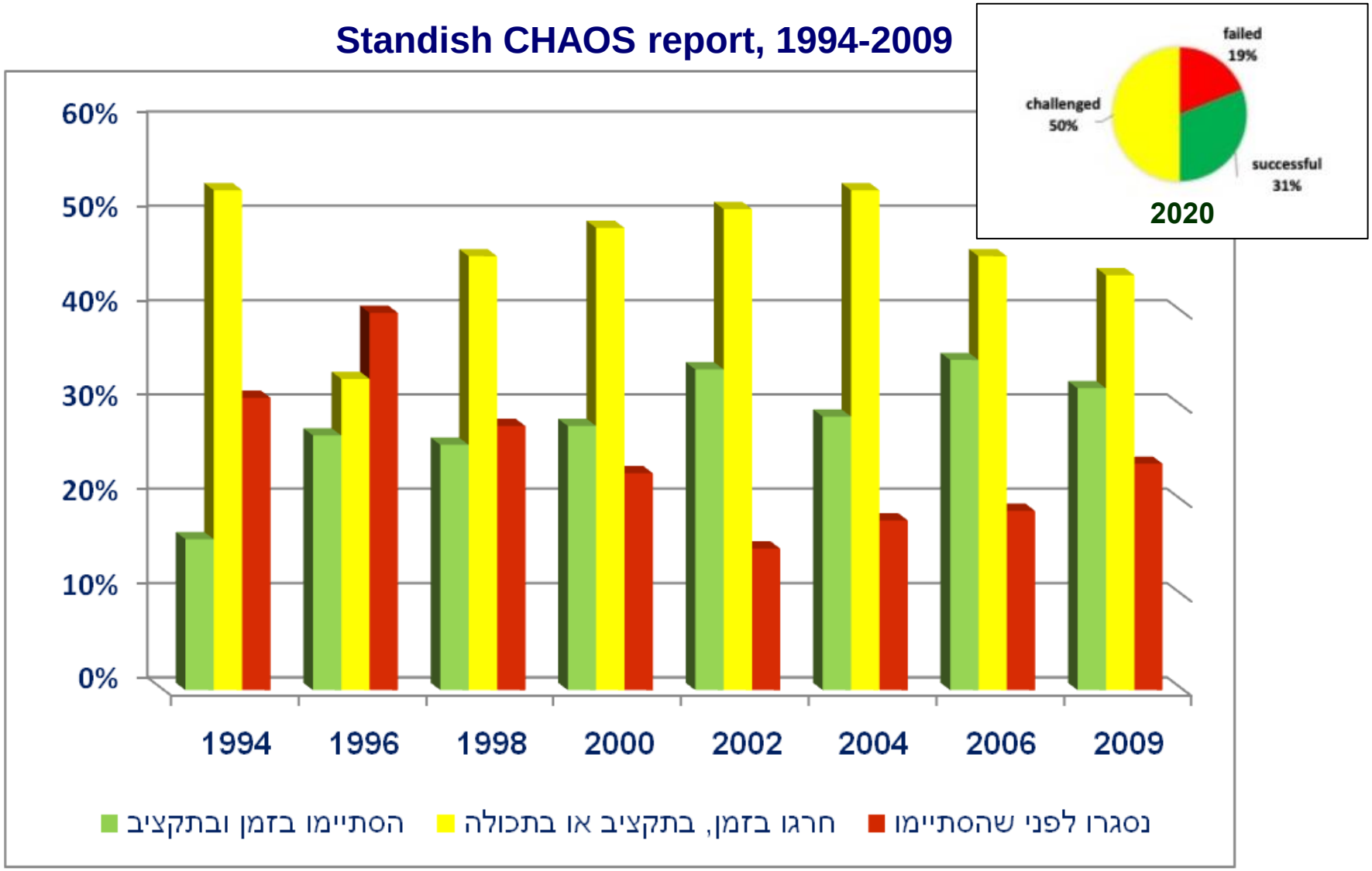
31.1% נסגרים לפני שהסתיימו

52.7% מסתיימים בחריגה של מעל 89% מההערכות המקוריות

16.2% מסתיימים על פי המתוכנן, בזמן ובתקציב



האם העולם משתפר?



שאלות רלוונטיות

- האם תוכנה היא בלתי אמינה מיסודה?
- האם האָשְמָה היא אכן בתוכנה ובמפתחיה?
- האם ניתן היה לחזות את התקלות מראש?
  - באיזה שלב?
  - באילו אמצעים?

- מדד אמינות מקובל לתוכנה
  - **מספר "באגים" לאלף שורות קוד (defects / KLOC)**
    - תוכנה "סבירה" – 25
    - תוכנה "טובה" – 2
    - תוכנת מעבורת חלל – 0.1>

כמה מאפיינים ייחודיים של תוכנה

- סיבוכיות (complexity)
  - מספר המצבים / התרחישים גדול מאד
    - קשה להבין את המכלול
    - קשה לבדוק בדיקה ממצה
- תאימות (conformity)
  - לתוכנה אין אילוצים פיזיקליים
    - נדרשת להתאים את עצמה לכל השאר
    - נדרשת לפצות על מגבלות הדיסציפלינות האחרות
- נסתרות (invisibility)
  - אין מודל מוחשי (tangible)
- יכולת שינוי (changeability) ואחזקתיות (maintainability)
  - עלות השידרוג = עלות הפיתוח בלבד
    - אין חומרים
    - אין תהליכי ייצור (למעט התקנה מחדש)
    - אין אריזה ושינוע
  - אין "כשל רכיבים"
    - כל כשל הוא "כשל תכן"
    - התגלתה תקלה בעותק אחד = כל העותקים לא תקינים
    - בעיה בהגדרת "אמינות תוכנה"

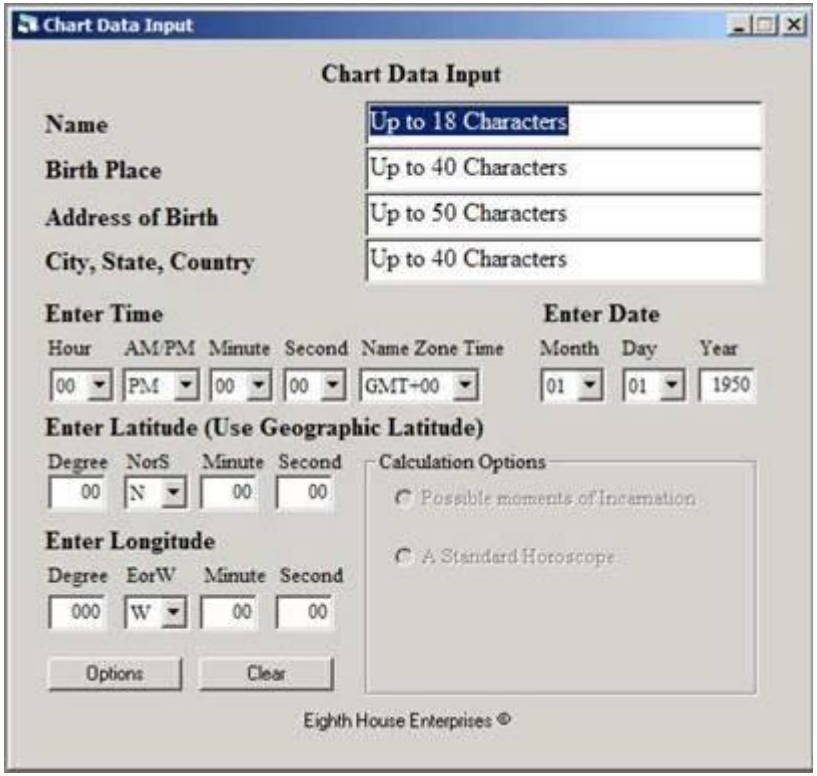
"…building software will always be hard. There is inherently no silver bullet."

F. Brooks (1987)*

*Brooks, F.P., *No Silver Bullet: Essence and Accidents of Software Engineering*, Computer, April. 1987

סיבוכיות (complexity)

- מה מספר המצבים האפשריים של מסך הנתונים הבא?



$\sim 30^{18}$
$\sim 30^{40}$
$\sim 30^{50}$
$\sim 30^{40}$

$\sim 10^{255}$

= 12 X 2 X 60 X 60 X 24 X
12 X 31 X 100 = 77,137,920,000

= 360 X 2 X 60 X 60 = 2,592,000

= 360 X 2 X 60 X 60 = 2,592,000

$\sim 10^{21}$

- האם ניתן לנתח את כל המצבים?
- האם ניתן לבדוק את התוכנה בכל המצבים?

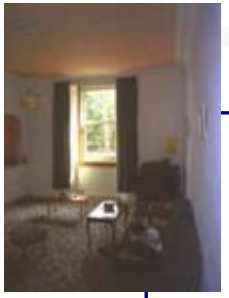
MSExcel 2007: 850 * 77.1 = ?

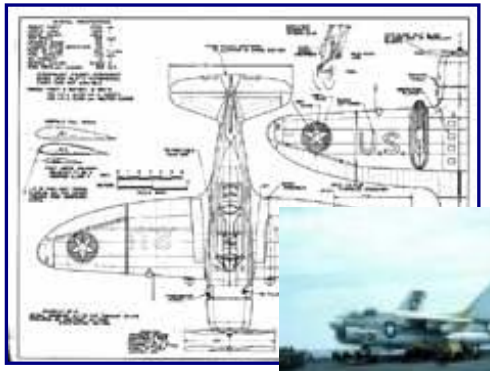
תאימות (conformity)

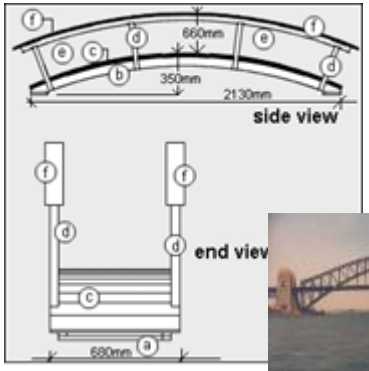
- הקניית יכולות מוספות לחומרה (מכניקה, אלקטרוניקה, ...)
  - התגברות על מגבלות פיזיקליות
  - תוספת דיוק, מהירות וכו'
  - הוספה/שינוי של תכונות
  - תיקון חריגות בדיעבד
- התוכנה נמצאת בקצה שרשרת הפיתוח!
  - אבל יש לקחת בחשבון את מאפייני התוכנה מתחילת הדרך.

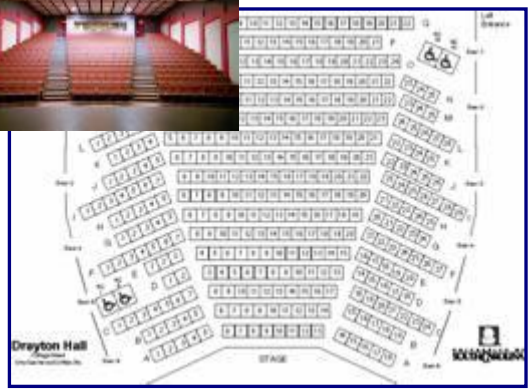
נסתרות (Invisibility)

- האם ניתן לתאר תוכנה באמצעות מודל מוחשי?









?

יכולת שינוי (changeability) ואחזקתיות (maintainability)

- מוצר תוכנה הינו מוצר "גמיש"
  - תוכנה אינה דורשת חומרי גלם
  - אמצעי הפיתוח זולים וזמינים
  - אין תהליך ייצור
  - "החלפת מודל" קלה ופשוטה
- ולכן
  - מוצר תוכנה צפוי לעבור שינויים ועדכונים תכופים

האחזקה* תופסת את חלק הארי במחזור חיי התוכנה!

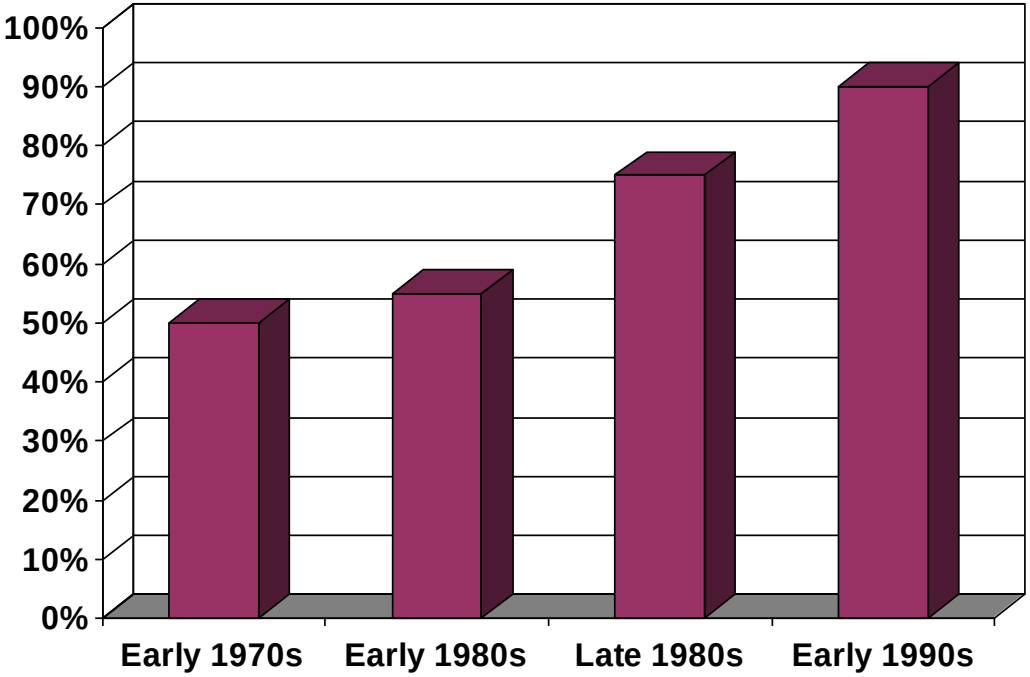
*** תיקונים, שיפורים, התאמות**

"The Legacy Crisis"

- תוכנות לא מתות, רק משודרגות...
- מהו החלק היחסי של ה"אחזקה" מסך העלות הכוללת של מחזור החיים ? = $\frac{\text{אחזקה}}{\text{פיתוח + אחזקה}}$
  - "אחזקה" = תיקונים, התאמות, שידרוגים.

מי רוצה להיות תכנת COBOL?

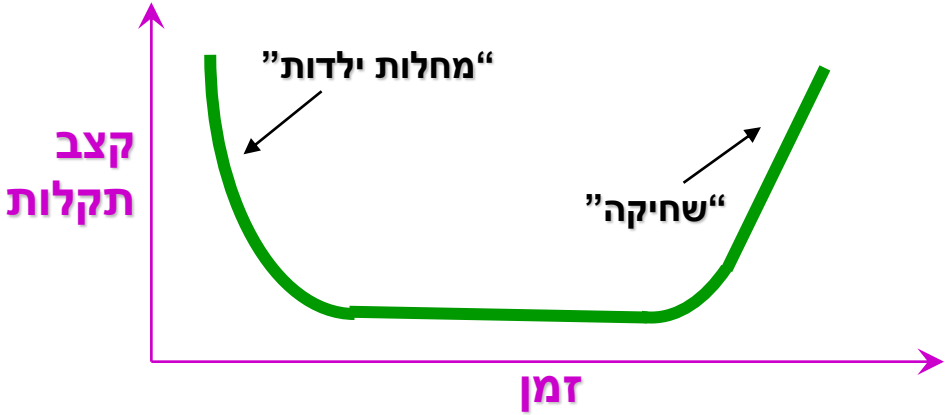
- **70% מהתוכנה העסקית הפעילה בעולם עדיין כתובה ב-COBOL!**
- **16,000 חברות גדולות עדיין משתמשות ב-COBOL!**
- **10,000 מחשבי MF בעולם מכילים 200 מיליארד שורות COBOL!**



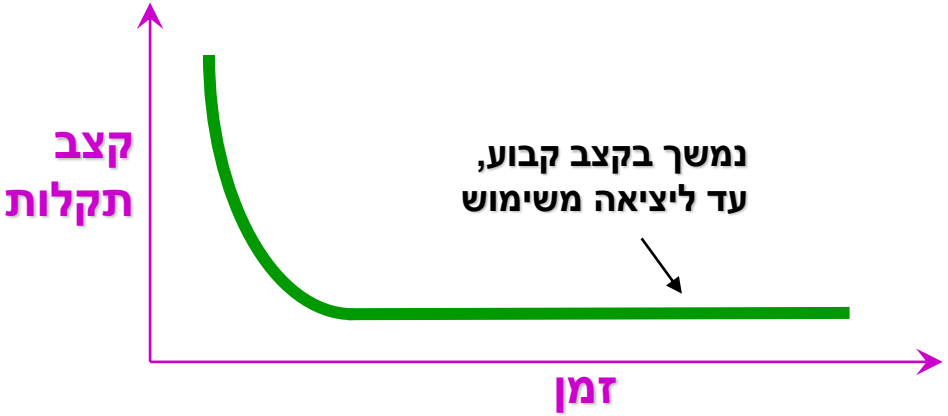
Source: R.C. Seacord, D.Plakosh, G.A. Lewis, *Modernizing Legacy Systems*, 2003

תקלות תוכנה לעומת תקלות חומרה

- עקומת כשל של חומרה

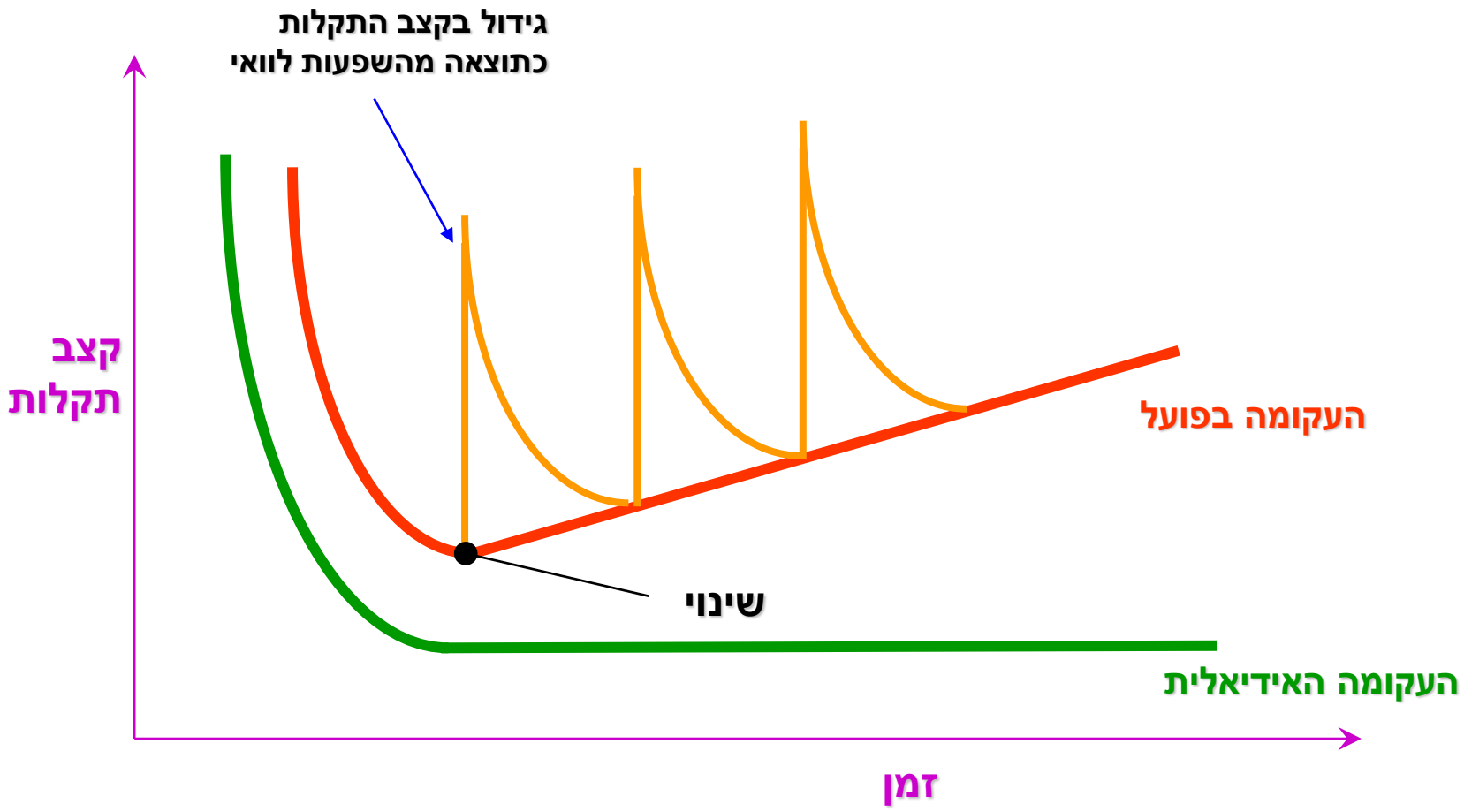


- עקומת כשל (אידיאלית) של תוכנה

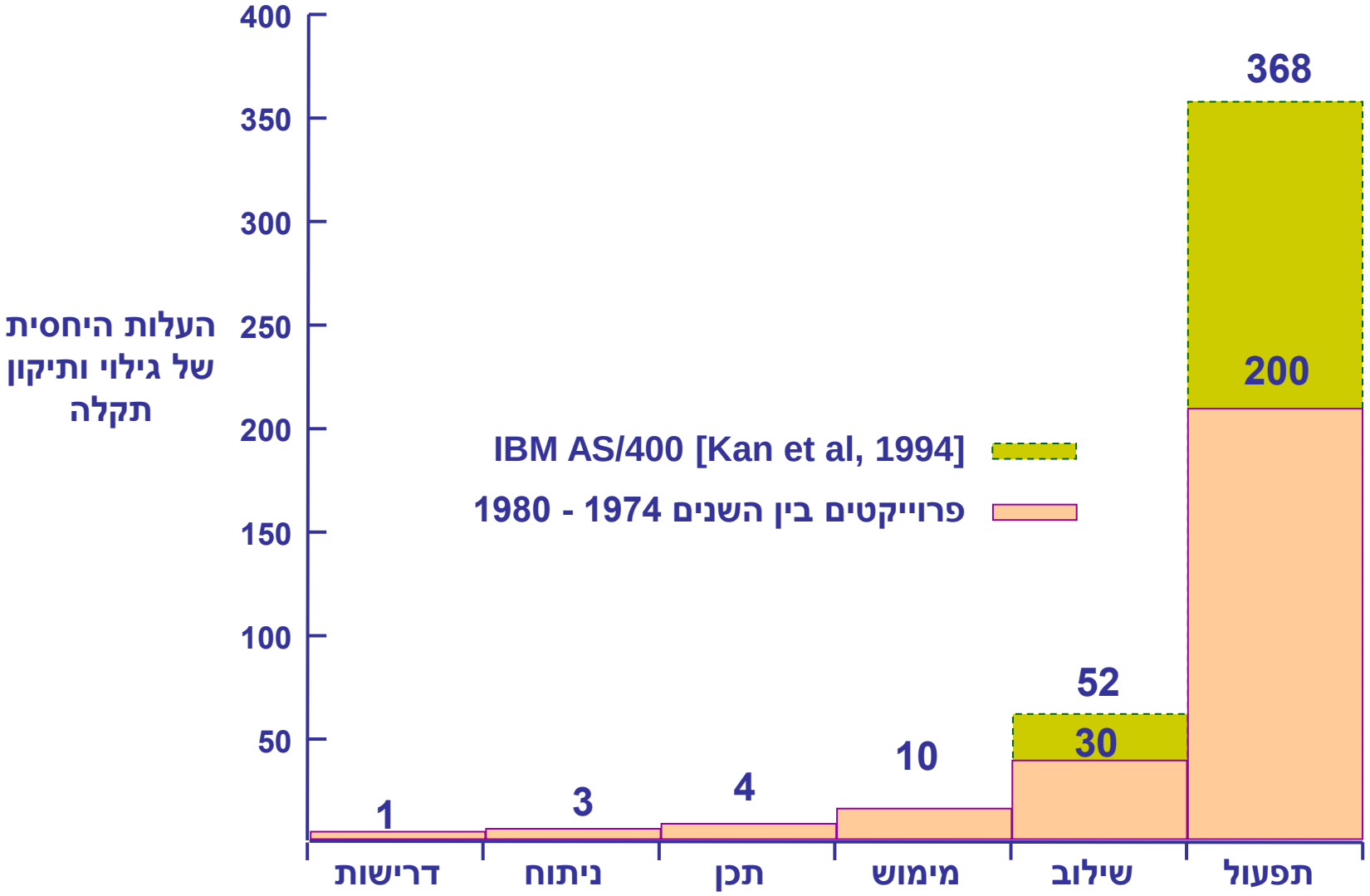


השפעת שינויים על עקומת התקלות

- עקומת כשל (בפועל) של תוכנה

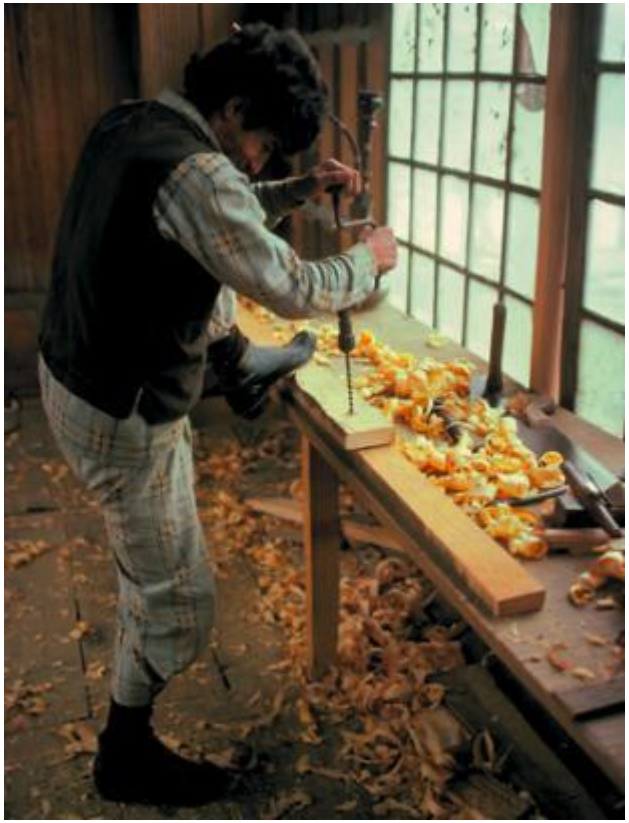


העלות היחסית של שינויים (כתלות בשלב הפיתוח)



תופעת "נדיפות הפסולת"

אילו תוכנה הייתה עשויה מעץ...



... בסוף הפרויקט היה כל תכנת "קבור" תחת ערימת קרשים!

מה עשוי לסייע לפיתוח אמין יותר של מע"ת?

- מחזור חיים ברור ומוגדר היטב
  - שלבים
  - פעילויות
  - תוצרים
- עקביות (consistency)
  - קשר ברור ונראה לעין בין תוצרי השלבים השונים
- שפה משותפת ברורה ואחידה
  - אפשרות להתייחס למאפיינים השונים של המוצר
  - גמישות לעריכת שינויים ולהבנת השפעתם

הקורס יעסוק בעיקרו בפיתוח מבוסס מודלים לאורך מחזור חיי הפיתוח של מערכות עתירות תוכנה

מה זה "Software Engineering"?

The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software, and the study of these approaches

The IEEE's Guide to the Software Engineering Body of Knowledge - 2004

- אפליקציה הנדסית לפיתוח תוכנה
  - שילוב של מדעי המחשב, מתמטיקה ושיטות הנדסיות.
  - גישה שיטתית לניתוח, תכן, הערכה, מימוש, בדיקות, תחזוקה של תוכנה.

מה זה "Software Engineering"?

- מענה לצרכי הלקוח
  - זוהי המטרה של הנדסת תוכנה.
  - מהנדסי תוכנה צריכים לזהות ולהבין את הבעיה שהמערכת אמורה לפתור.
  - הוספה של יכולות לא נחוצות לתוכנה אינה עוזרת לפתרון הבעיה.
  - לעיתים עדיף לקנות פתרון ולא לפתחו.
- פיתוח והתפתחות שיטתיים
  - פיתוח תוכנה הוא תהליך הנדסי המערב ישום שיטות בדרך מאורגנת.
  - שיטות שונות עברו תהליך של סטנדרטיזציה ע"י ארגוני תקינה כמו ISO .

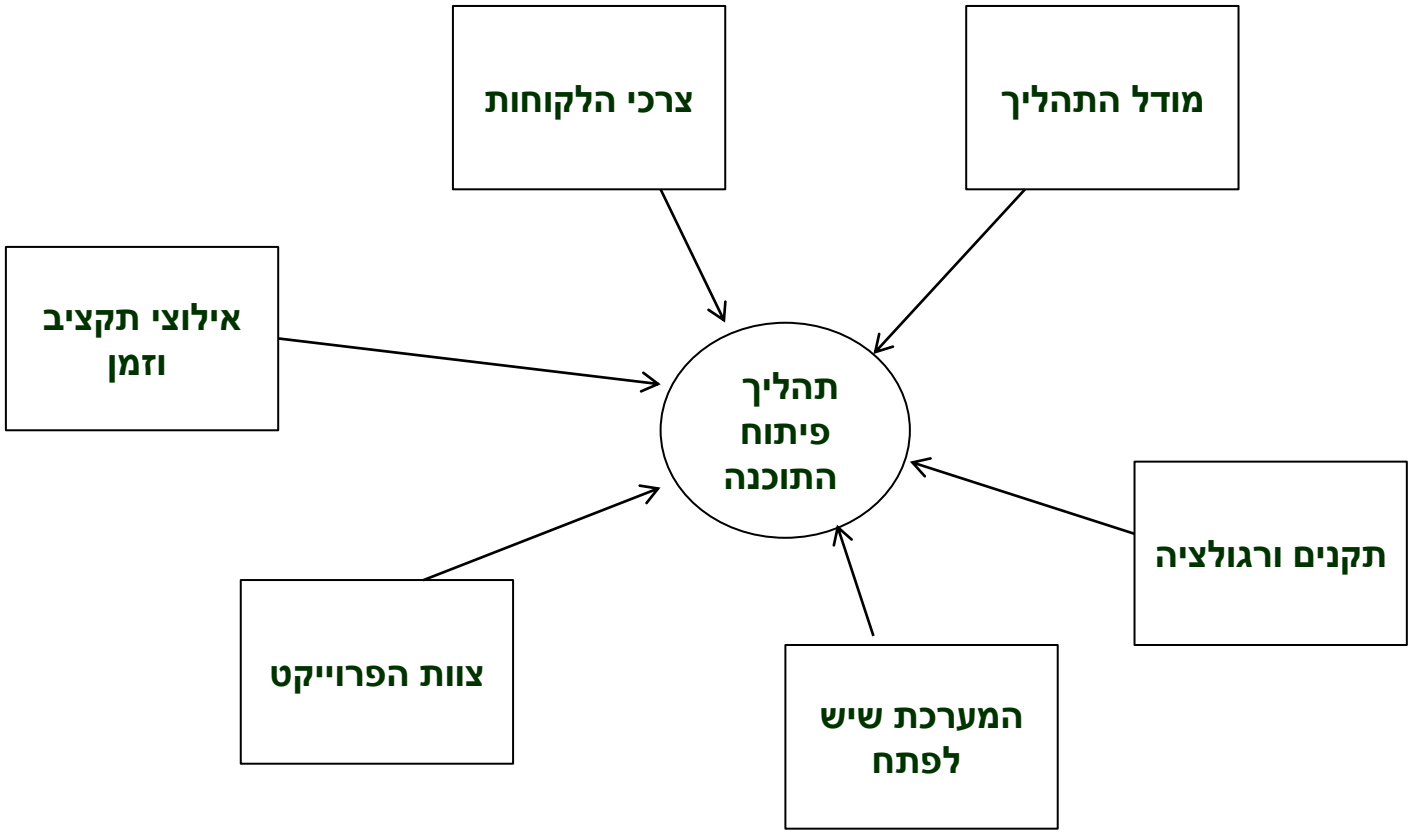
עוד על הנדסת תוכנה

- מערכות תוכנה גדולות ואיכותיות
  - שיטות של הנדסת תוכנה דרושות כי אדם אחד אינו יכול להבין בשלמות מערכות תוכנה גדולות.
  - פיתוח מערכות תוכנה גדולות מצריך עבודת צוות ותיאום.
  - האתגר: חלוקת העבודה ושילוב כל החלקים כך שיעבדו יחד נכון.
  - המוצר הסופי צריך להיות ברמת איכות נדרשת.
- עלות, זמן ואילוצים שונים אחרים
  - לרשות תהליך הפיתוח עומדים משאבים מוגבלים.
  - יש להבטיח רווח בסופו של התהליך.
  - בשוק מתחרים (זולים או מהירים יותר).
  - הערכה שגויה של עלויות ולוחות זמנים גרמה לכשלון פרויקטים רבים.

בעלי ענין בתהליך פיתוח

- משתמשים
  - המשתמשים בתוכנה.
- לקוחות
- המשלמים עבור התוכנה.
- מפתחי התוכנה
- מנהלי הפיתוח

הגורמים המשפיעים על תהליך הפיתוח



תהליך פיתוח תוכנה