

# הנדסת תוכנה

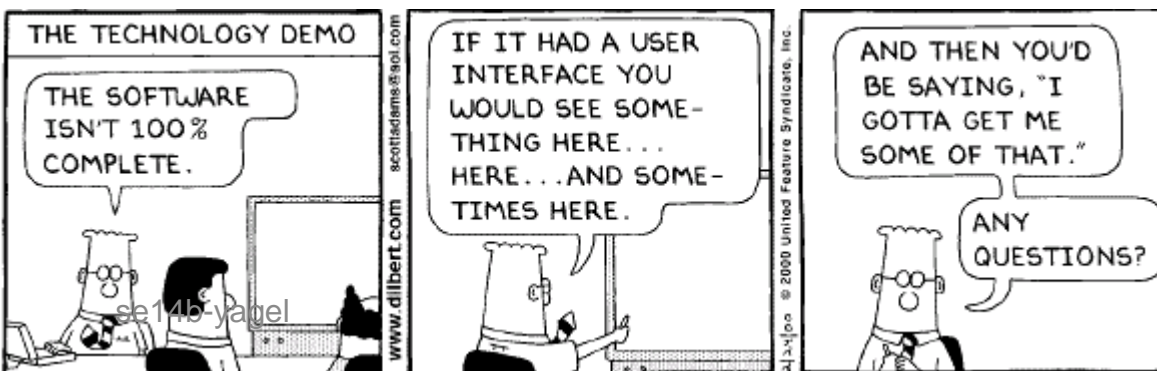
## 14. סיכום הקורס ומצגות סיום



[Pragmatic Programmer Tip](#) :  
**learn a new language every year**

# מה היום?

- סיכום הקורס
  - מה הלאה?
  - על המבחן
- "מסיבת שחרור" – מצגות (CD+) ושיקוף
- ~~תרגיל: השלמות ובירורים~~



# לסיכום הקורס

- האם בכלל צריך כזה קורס? האם אפקטיבי
- מיקום בתואר וקשר לקורסים אחרים
- האם וכמה תרם הפרויקט? ההרצאות?
- גודל קבוצות? סוג משימה?
- מה מיותר\מעמיס? מה חסר?
- התנסות מול "less is more"
- סקר...
- מי שבכל זאת אהב (או רוצה לשנות) מוזמן להצטרף לצוות הקורס (או לבקר בהמשך) ...

# ACM/IEEE Computer Science Curricula 2013, start with:

- “In general, **students learn best** at the application level much of the material defined in the **software engineering** knowledge area **by participating in a project**. Such projects should require students to work on a **team** to develop a software system through as much of its **lifecycle** as is possible. Much of software engineering is devoted to effective **communication** among team members and **stakeholders**. Utilizing project teams, projects can be sufficiently challenging to require the use of **effective software engineering techniques** and that students develop and practice their communication skills. While organizing and running effective projects within the academic framework **can be challenging**, the best way to learn to apply software engineering theory and knowledge is in the **practical environment of a project**.”

# כמה שקפים מהרצאת המבוא

# מהי הנדסת תוכנה?

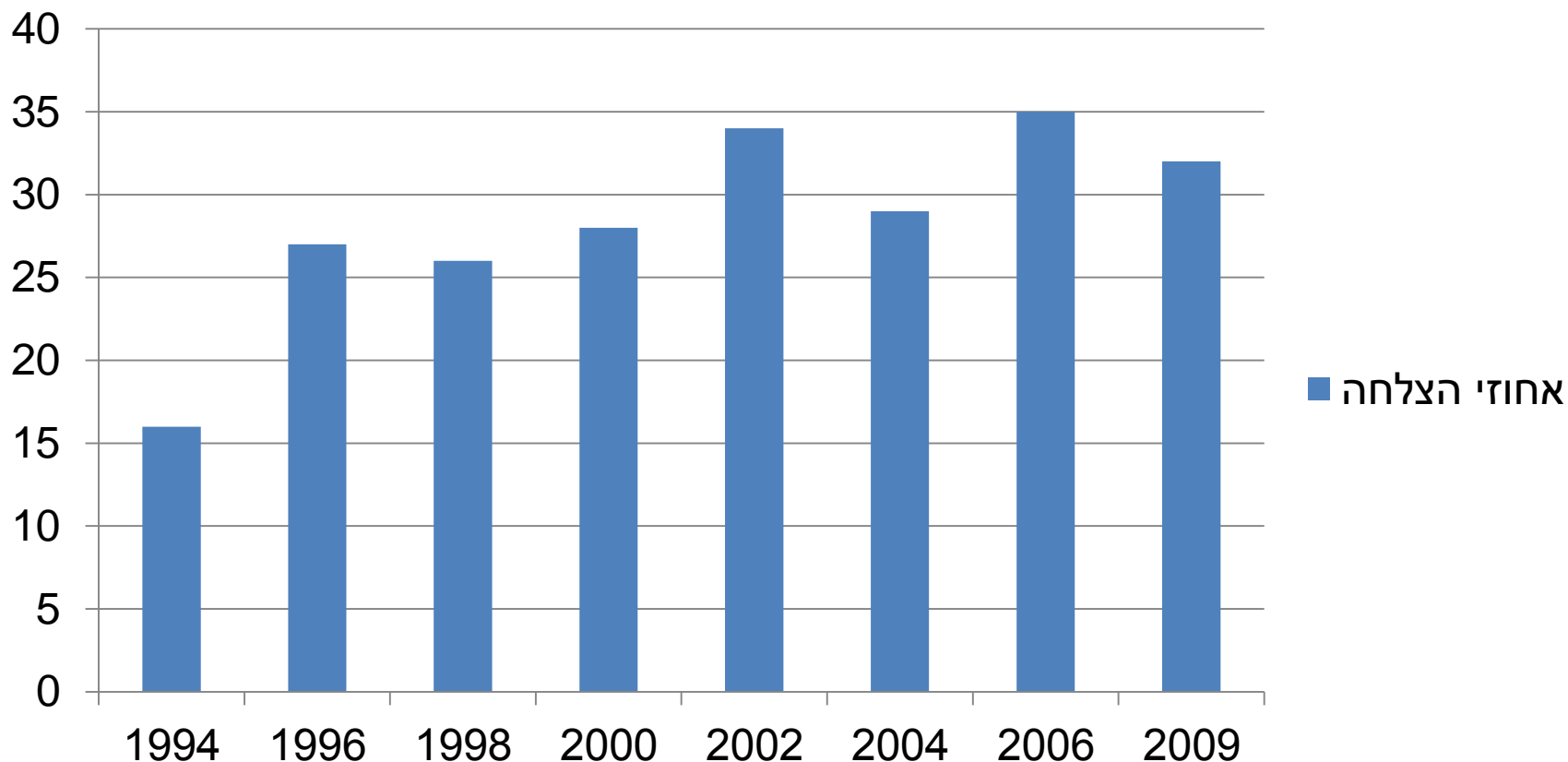
**“Software Engineering** is the study and application of [engineering](#) to the design, development, and maintenance of software”  
[IEEE SWEBOK](#)’04’13, IEEE Glossary, [Wikipedia](#)

אוסף תהליכים, שיטות וכלים לפיתוח מוצר תוכנה  
בעל ערך ללקוח ויכולת התאמה למצבים שונים תוך  
שימוש מיטבי ואיכותי במשאבים, משלב הרעיון ועד  
לשלב הפרישה

Software engineering has accepted as its charter,  
"How to program if you cannot." -- *E. Dijkstra*

# Chaos Report (debatable...)

## אחוזי הצלחה של פרוייקטי תוכנה



# מה מיוחד\קשה בפרויקט תוכנה?

- Brooks: סיבוכיות מובנית מול אקראית  
“No Silver Bullet”

– סיבוכיות

– תאימות (לכל הדרישות)

– גמישות לשינויים

– חוסר נראות (סינדרום 90% לסיום)

- בד"כ: עבודה אינטלקטואלית,  
עבודת צוות, רב תחומי

- רכישת ידע

Why software projects are not routine work





# מדוע פרויקטי תוכנה נכשלים לעיתים כל כך קרובות? \*

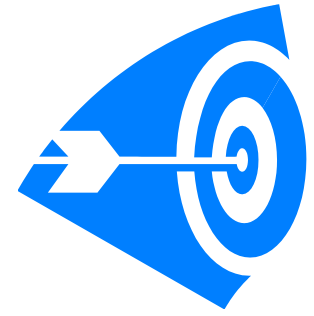


- יעדי הפרויקט לא מציאותיים או לא ברורים
- אומדנים לא מדויקים של המשאבים הנדרשים
- דרישות מערכת אינן מוגדרות היטב
- דיווח לקוי לגבי מצב הפרויקט
- סיכונים לא מנוהלים
- תקשורת לקויה בין הלקוח, המפתחים והמשתמשים
- שימוש בטכנולוגיה לא בשלה
- אי-יכולת לנהל את מורכבות הפרויקט
- פרקטיקות פיתוח מרושלות
- ניהול לקוי של הפרויקט
- פוליטיקה של בעלי עניין
- לחצים מסחריים

בסיום הקורס נחזור לרשימה  
זו ונבדוק האם רכשנו כלים  
מתאימים להתמודד עם  
בעיות אלה.

\* [Charette, R. N., Why Software Fails?, IEEE Spectrum, Vol. 42, Issue 9, Sept. 2005](#)

# מטרות הקורס (סילבוס)



- הבנת הבעיות והפתרונות המרכזיים של הנדסת תוכנה בפיתוח מוצרי תוכנה.
- פיתוח ראיה מערכתית והיכרות עם תהליכים, שיטות עבודה וכלים רלוונטיים בשימוש התעשייה.
- לימוד והתנסות בבניית פרויקט תוכנה משמעותי תוך כדי עבודת צוות וכישורים רכים נוספים.
- הכנה לפרויקט הגמר.

# הפרויקט - מטרות

- נסיון ישיר עם חומר הקורס
- אתגרים טכניים בשל גודל הפרויקט
- אתגריים חברתיים במסגרת מאמץ קבוצתי
- הזדמנות להתנסות בסביבה חדשה, למשל: התקנים ניידים, **Web**, .net, Ruby, APIs, קוד פתוח, רשתות חברתיות, דרייברים, תוסף דפדפן, [html5](http://html5.com) ....  
(בכל זאת מומלץ RAD)
- הזדמנות עסקית (זכויות יוצרים!)
- **המלצה: לא לפתוח הרבה חזיתות!**
- **מה בכל זאת שונה מהתעשייה?**



# לסיכום

- דיון בהנדסת תוכנה, תוכנה היום, אתגרים
- מטרת הקורס: מתכנת  $\leq$  מהנדס תוכנה
- המיוחד בקורס
  - רב תחומי
  - הזדמנות לעבוד על רעיון שלכם
  - בד"כ אין תשובה אחת נכונה, מותר לטעות
  - כישורים "רכים" (יצירתיות, שיתוף פעולה)
  - לא קשה, אבל עבודה די רבה (ומהתחלה!)
- תוכן הקורס: תהליכים, דרישות, תיכון, בדיקות, מימוש, כלים ועוד.....
- שאלות \ הבהרות \ הצעות ?

## בהצלחה ובהנאה



# מה למדנו: תהליכים, שיטות ו

כלים SE

שיטות

מודל \ תה

במוקד: א

- מפל המים, פיתוח איטרטיבי, אג'יל, אתחול פרויקט, ניהול, עבודת צוות, דרישות, תיעוד, ניתוח, הערכה, תכנון, תיעדוף, רטרוספקטיבה, תיכון (מונחה עצמים), ארכיטקטורה, בדיקות, חווית משתמש (שמישות), איכות, סקרים (עמידה לפני קהל)

<http://scottberkun.com/2011/scrum-for-weddings/>

- Version Control, Unit Testing \ **TDD**, OODP, Refactoring/Reuse/Legacy Code

# Computer Aided Software Eng.

- Collaboration / Project: Wiki, Issue/Bug Tracking, Planning, **VCS (git / github)**, Code Review, etc.



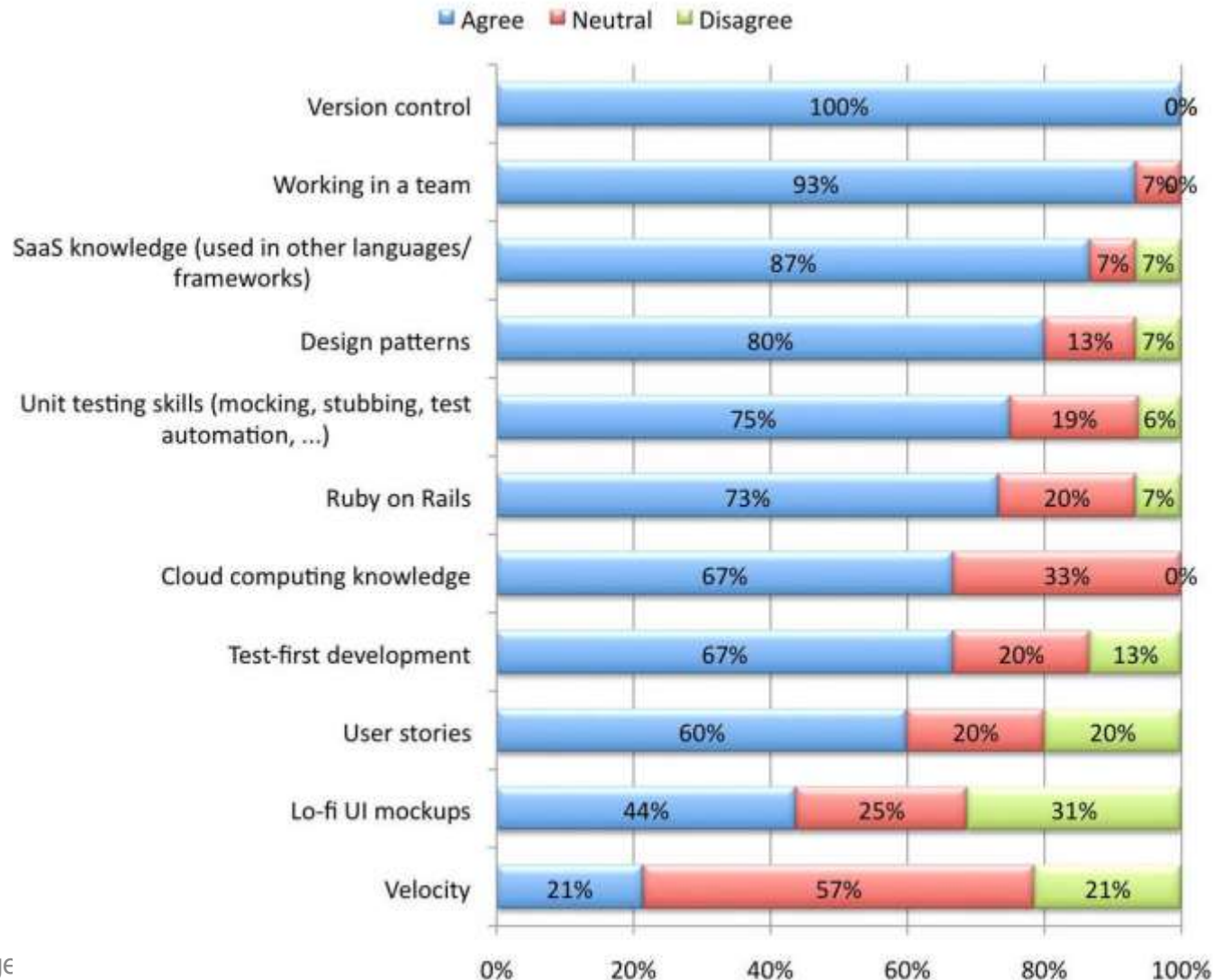
SCISR9\_(mp3cut.net34-36).mp3

[resume](#) ([StartupEng. ~1:40](#), [Oredev Conf. ~21](#))

- Test & Design Tools: Unit Testing, Mocking Framework, Coverage, CI, Refactoring, UML TOOLS, Measurements
- Standard Development & Productivity Tools



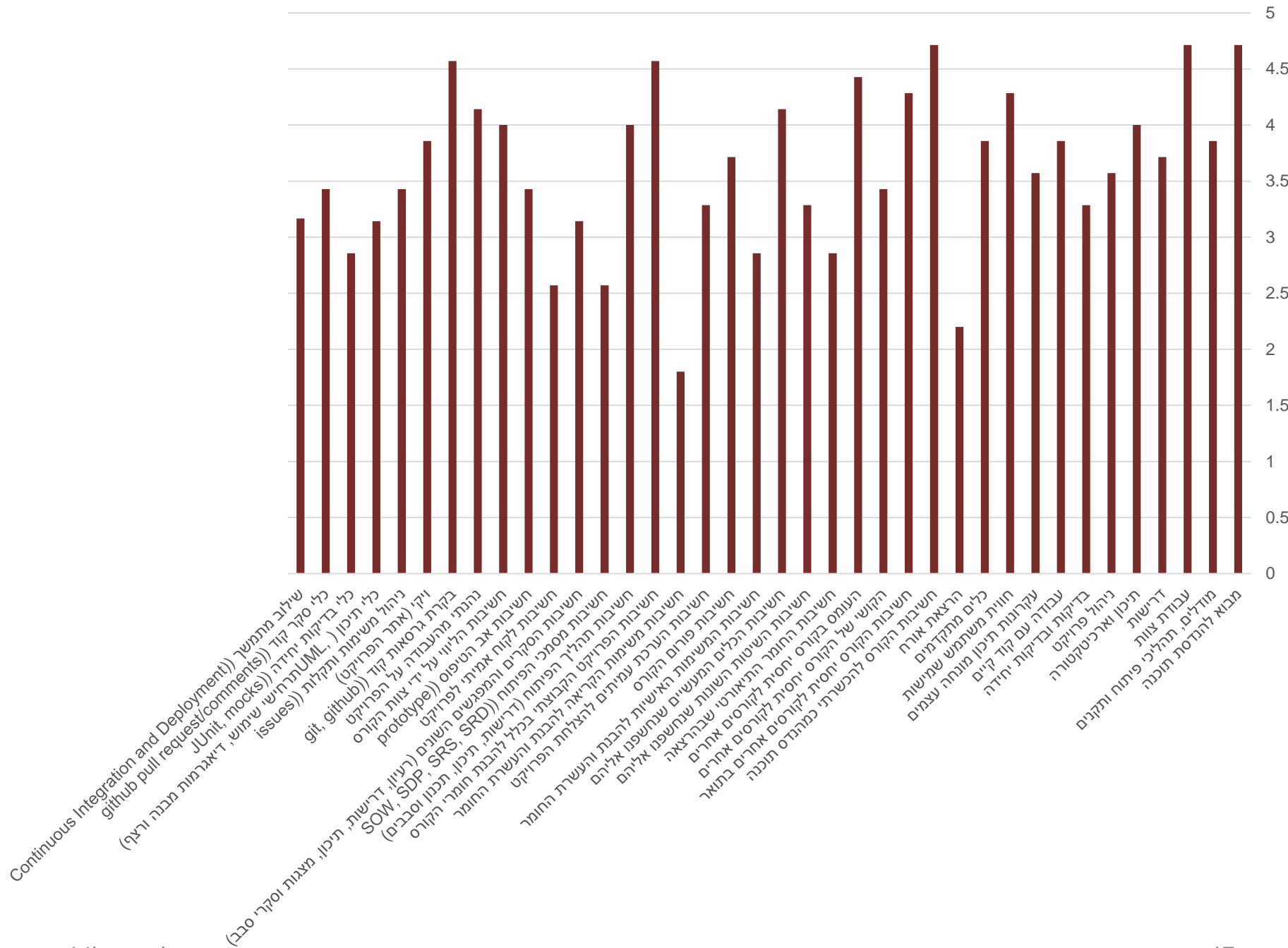
# סקר בוגרים בברקלי, אלו מנושאי קורס דומה הכי חשובים



# סקר

- 7 משיבים - תודה
- נושאים עיקריים בהערות:
  - הקדמת תכנות רשת
  - העומס בהכרת סביבות וכלים חדשים
  - יותר דגש על עבודה קבוצתית כולל מעקב
  - יותר עזרה טכנית (בפעם הבאה – שעתיים מעבדה)
  - קורס חשוב, קישורים להמשך



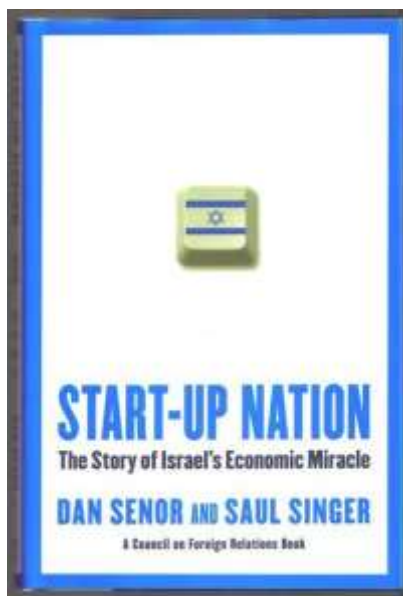


# מה אולי חסר

- מידת תיאום הפרויקט לקורס
- כלים ייעודיים (תיעוד, ביצועים, דיבאג, ניהול משימות, למשל, Slack, [PivotalTracker](#))
- עוד תהליכים ושיטות (סקרים\לקוחות, הפחתת סיכונים, מדידת התקדמות\velocity, הערכת עמיתים, תחזוקה)
- גיבוש צוות
- שיווק, פטנטים, סטארט-אפ
- המשך בקורס בחירה \ תואר שני

# מקומה של התוכנה

	IN THE OLD DAYS	NOW
LISTENING TO MUSIC		
WATCHING FILMS		
CHATTING WITH FRIENDS		
READING THE NEWS		
PLAYING MUSIC		



# איך להישאר מהנדס תוכנה רלוונטי (או פיתוח הפרופיל המקצועי שלך)

- ניסיון מעשי
- חונך טוב
- לימוד מתמיד: קריאה, כנסים, קורסים מקוונים  
([Startup Eng.](#), [Master Prgrm.](#)), שפת תכנות  
חדשה, קהילות\פורומים\רשתות חברתיות, כתיבת  
בלוג, ...
- תרגול ([Kata](#), [Dojo](#), [Koans](#), [exercism.io](#))
- קוד פתוח \ תרומה לקהילה
- הוראה
- משהו אחר בחיים... \ ניהול זמן

# (Self) Improvement Links

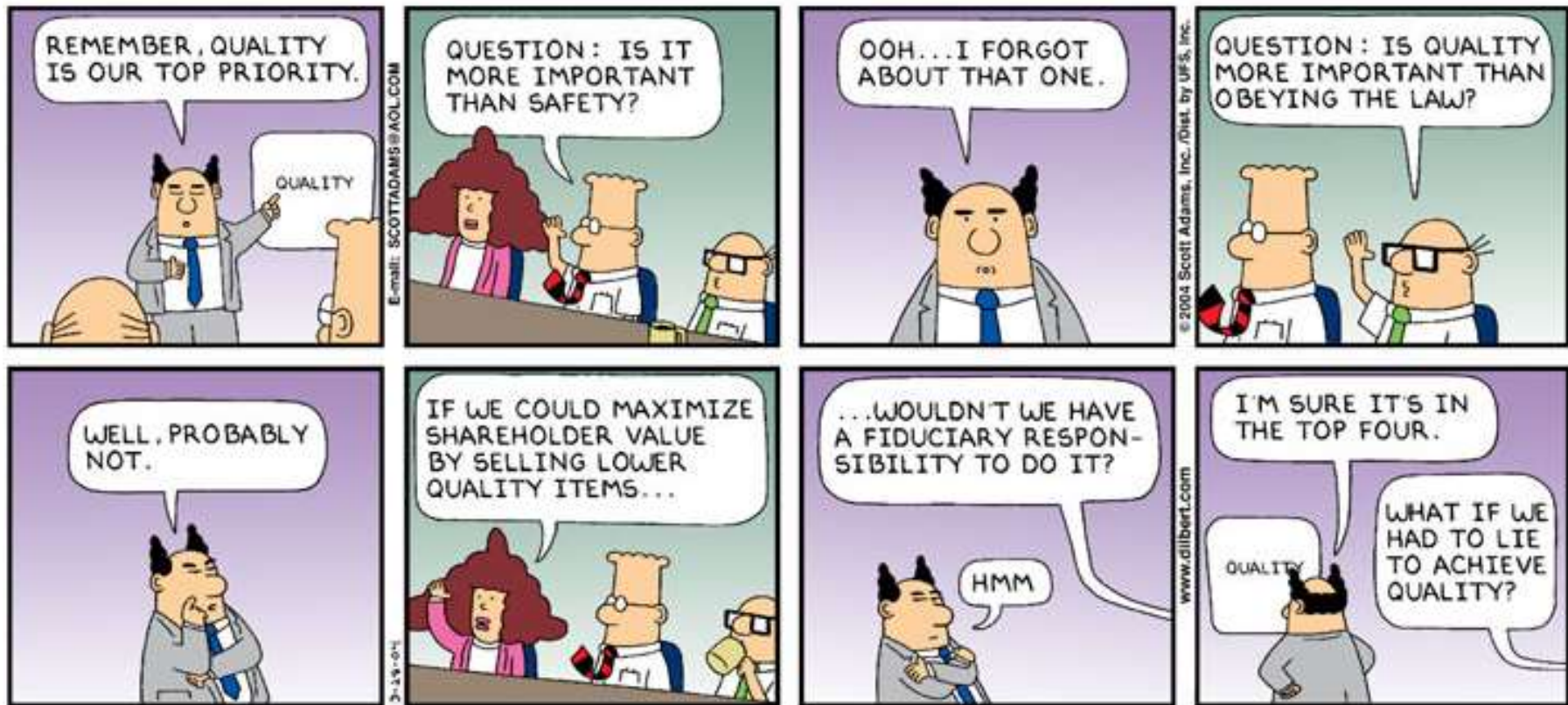
- Learning and self-improvement, Software Craftmanship [thread](#), 2010
- [The single most important thing you must do to improve your programming career](#), blog, 2008
- [10 טיפים שיהפכו אותך ממהנדס תוכנה למותג!](#), 2011
- [The Effective Workplace](#), 2011
- [www.kalzumeus.com/2011/10/28/dont-call-yourself-a-programmer](http://www.kalzumeus.com/2011/10/28/dont-call-yourself-a-programmer)
- [Why In-Person Socializing Is A Mandatory To-Do Item](#)
- More at course wiki links

# מתי פרויקט בעצם נגמר?

• תשובות מבלוג [Artima](#):

- When the deadline arrives or shortly thereafter
- "When I'm sick of working on it any longer"
- When the project is canceled and all the finger pointing stops
- [Stackoverflow](#): When the client/user signs off on the UAT (User Acceptance Test) after testing it ([SCRUM](#))
- [Hofstadter's Law](#): It always takes longer than you expect, even when you take into account Hofstadter's Law.

# איכות



© UFS, Inc.

# עוד נושאים לסיכום

- פיתוח (כמעט) ללא קוד
- פיתוח (כמעט) ללא פרויקט
- פיתוח (כמעט) ללא עבודת צוות
- פיתוח (למרות) כישלונות \ מכשילים



# מבחן

## • מאגר שאלות (שיפורסם מראש)

- נושאים עיקריים מחומרי ההרצאה התאורטיים
- דיון והדגמה על הפרויקט שלכם

## • מקורות:

- חומרי ההרצאה (מצגות+מקורות+סיכומים)
- הפרויקט שלכם

## • פורום הקורס – פתוח לדיון

## • שיעור חזרה

# הצגת\שחרור גרסה סופית

