



המכללה האקדמית להנדסה ירושלים

"הזמן קצר והמלאכה מרובה..." – מסכת אבות
Eisenhower: "plans are useless, but
planning is everything"



הנדסת תוכנה

5. הערכה תכנון ובקרת גרסאות

כלים: | github: issues+git

Pragmatic Programmer Tip :

Iterate the Schedule with the Code

Use experience you gain as you implement to refine the project time scales.

הלקוחות רוצים תוכנה (דרישות)
בזמן מסוים שבו הם צריכים
אותה
?

מה היום?

- הערכה
- תכנון (מערכת ניהול פרויקט)
- הזנת משימות, יצירת Backlog (ננסה עבור ההרצאה?)
- בקרת גרסאות קוד – Version Control
 - התנסות git
- סקר SDS – (מהנדס עמית)
- פרויקט שלב 5 – איטרציה 0 – ZFR
- תרגיל – עבודה ב- github (git+issues)



הערכה - מקורות

- Cohn, Agile Estimating and Planning
- Agile Product Management with Scrum, chap. 3: [Working with the Product Backlog](#)
- Fundamentals of Software Project Estimation
<http://www.spc.ca/downloads/resources/estimate/fullestbasics.pdf>
- Pressman ch. 27
- [Evidence Based Scheduling](#) (Joel S.)
- The Art of Project Management: How to Make Things Happen
<http://msdn.microsoft.com/en-us/library/aa480154.aspx>
- COCOMO II <http://www.softstarsystems.com/overview.htm>
http://sunset.usc.edu/csse/research/COCOMOII/cocomo_main.html
- Pilone&Miles, Head First Software Development
- Kniberg, [Scrum and XP from the Trenches](#)
- Simula Research, <http://simula.no/news/simula-and-joergensen-top-ranked-within-software-engineering>

Estimation & Planning Links

- [Learn to Estimate](#)
- Steve McConnell, [video on](#) 10 deadly sins of estimation
- Planning Poker Discussions [1](#), [2](#)
- Motley says: “Planning Poker? I bet my estimates are better than yours!”, [blog](#), 2011
- [Estimation Toolkit](#), Article 2010
- [Working with the Product Backlog](#) (from Agile Product Management with Scrum, chap. 3)
- [Estimating](#) (from Applied Software Project Management, chap. 3)
- [Agile Sticky Board](#) video of low-tech means

Kanban Links

- infoq mini-book: [Kanban and Scrum - making the most of both](#)
- <http://jamesshore.com/Blog/Kanban-Systems.html>

בקרת גרסאות קוד - מקורות

- Sink, Version Control by Example
- Google Tech Talk: [Linus Torvalds on git](#)
- Spolsky, Hg Init: a Mercurial tutorial
<http://hginit.com>
- Sink, Source Control HOWTO
http://www.ericssink.com/scm/source_control.html
- Intro to Distributed Version Control
<http://betterexplained.com/articles/intro-to-distributed-version-control-illustrated/>

VCS Links

- Eric Raymond on [vcs](#)
- [The 10 commandments of good source control management](#), blog 2011
- FogBugz and Kiln [video](#), 2011
 - DVCS University Slides ([*](#))
- [Spolsky, Hg Init: a Mercurial tutorial](#)
- Azad, [A Visual Guide to Version Control](#)
 - [Intro to Distributed Version Control \(Illustrated\)](#)
- [Eric Sink, Source Control HOWTO](#)

Git Links

- <http://git-scm.com/>
- Set Up Git (Win), [ssh issues](http://help.github.com/win-set-up-git/)
<http://help.github.com/win-set-up-git/>
- Info: <http://git-scm.com>, gitref.org,
progit.org (book, [basic](#)),
- [Windows client list](#), [Egit](#) (Eclipse), [VS](#),
[power-shell](#), [github for windows](#) ([slds](#))
- [git internals](#) video

More Git / Github Links

- Videos: [Git For Ages 4 And Up](#), [Git Going](#)(oredev'12), [Think like a](#)
- [learn.github.com/](#), [try.github.com/](#)
[Gitflow](#)
- saastv: [Using Branches with Git](#)
- [no branches at flickr](#)
- [Insider Guide to GitHub](#) (video)
- Articles: [article](#) (including .gitignore), [post](#),
[difficulties](#),



איפה אנחנו בפרויקט (בקורס)?

- למה?
בעיה (פלט: הצעת פרויקט\חזון\SOW)
- מי?
צוות (Inception, אתחול\תכנון פרויקט)
- מה?
דרישות (SRS)
- איך?
תיכון (ארכיטקטורה) (SDS)
- מתי?
תכנון וניהול – (ZFR)
- הלאה
(איטרציות, Code)



















הערכה

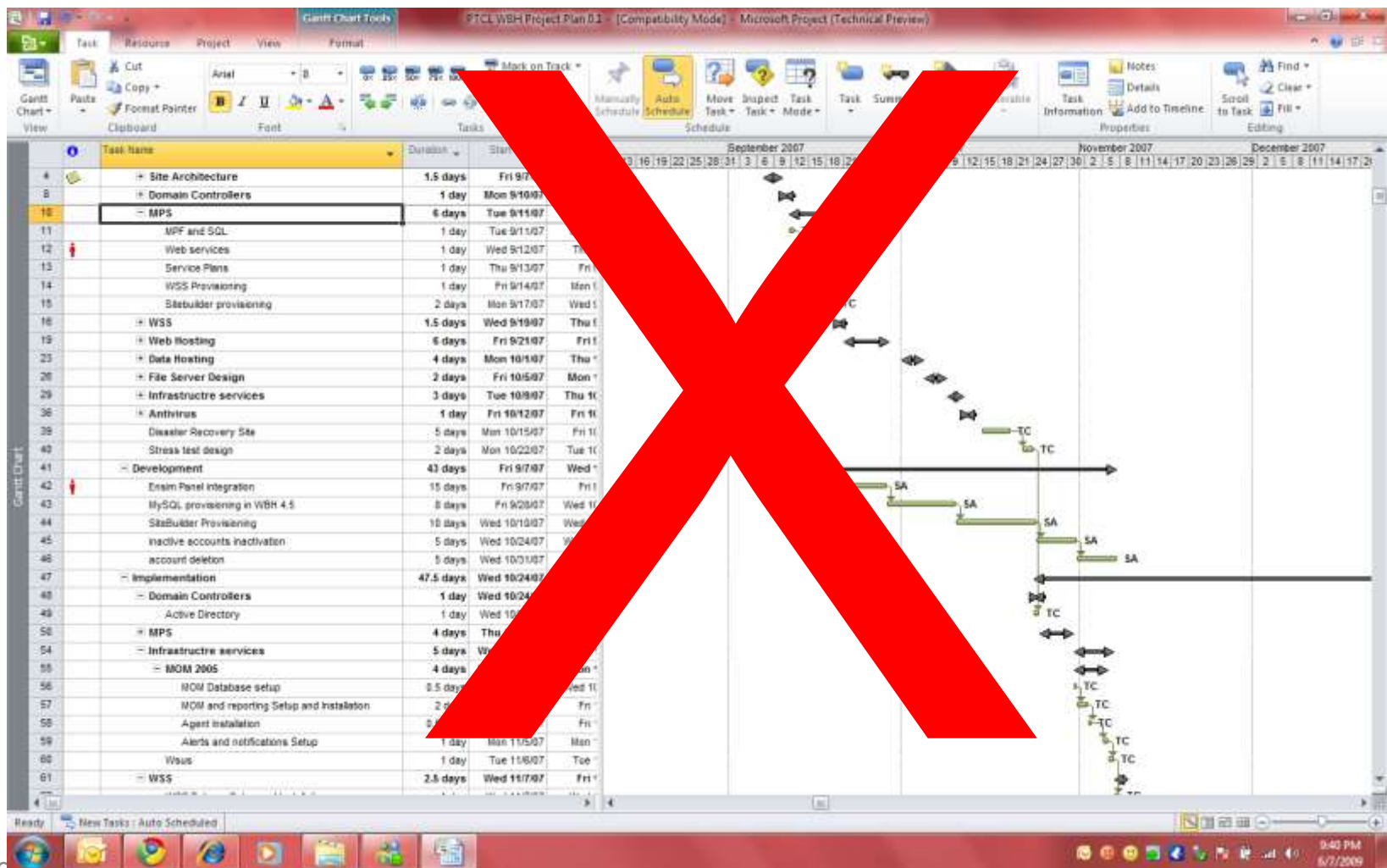
- מהי הערכה?
- למה זה חשוב? ולמה עכשיו?
- איך יוצרים הערכה?
- מה ההערכה שלכם?

מהי הערכה?

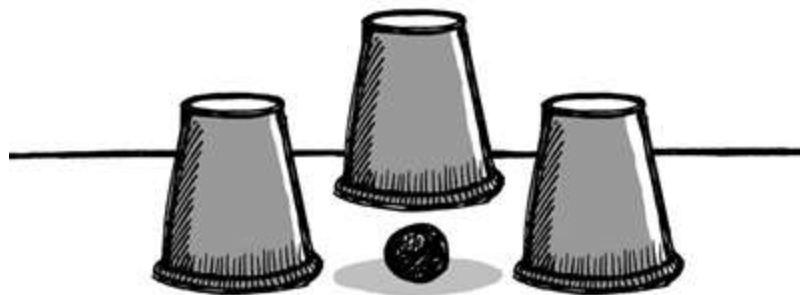
- פרדיקציה
- לא מטרה, לא התחייבות וגם לא...



וגם לא תכנון



PLANNING IS GUESSING





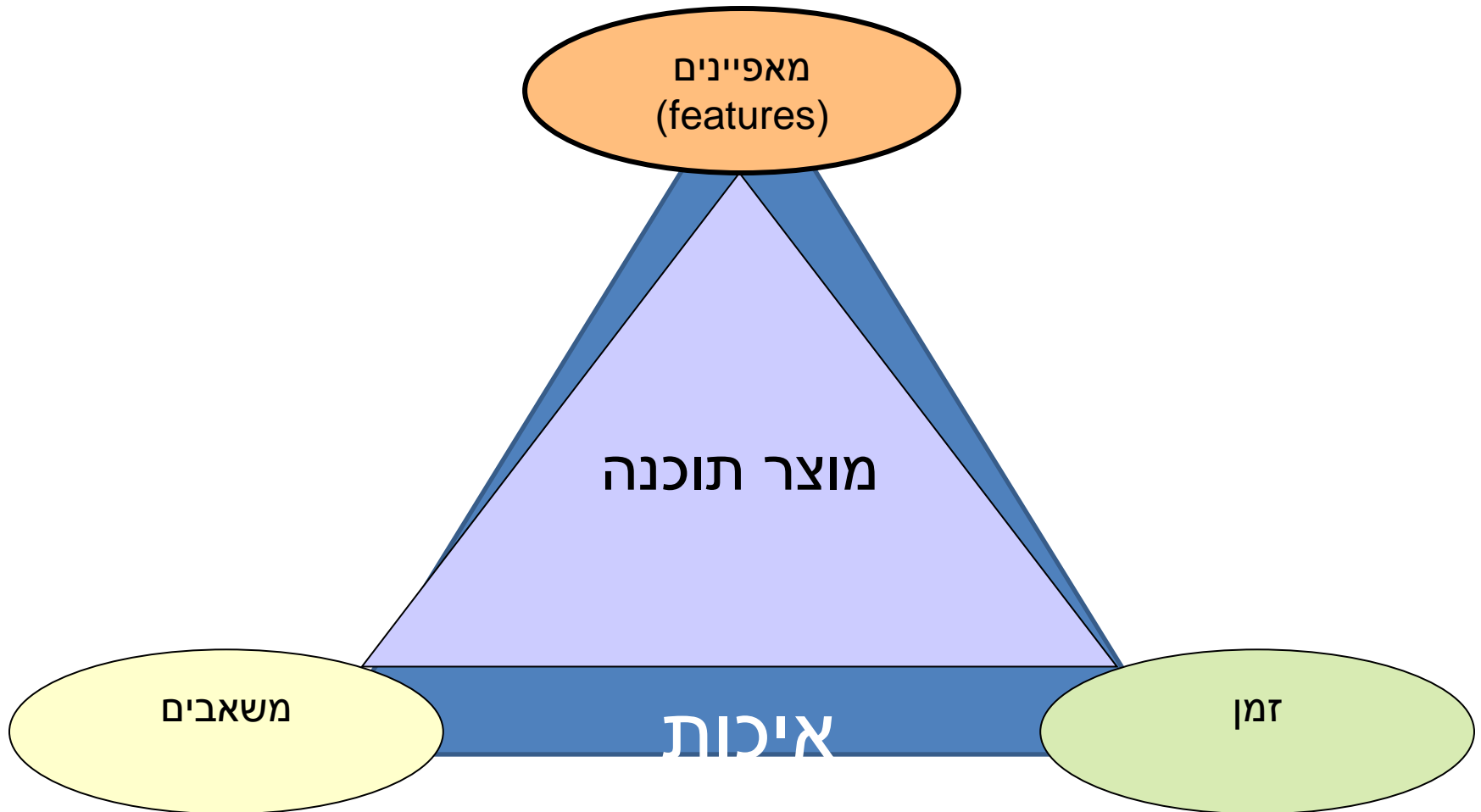
שאלות הערכה

- תן הערכה ברמת סמך של 90% לשאלות הבאות:
 - כמה סטודנטים בכיתה?
 - בכמה שעות הליכה אפשר להגיע לכנרת?
 - כמה עצים יש בכרמל?
 - כמה באגים יהיו בפרויקט שלך בסיומו?
 - מה המרחק בין מאדים לשמש?
- לפי מה הערכת ב-90%?
- האם ה-90% הלחיץ? האם טווח עוזר בכך?

למה חשוב להעריך?

- לפי הניסיון, **מאד קשה** להעריך פרויקטי תוכנה
- מצד שני **תתבקשו** לתת הערכות ("זה יהיה מוכן כשאסיים...")
- הערכות **גרועות** יכולות להיות גרועות לכם, לצוות, לחברה, ללקוח
- בואו נראה מה אפשר לעשות

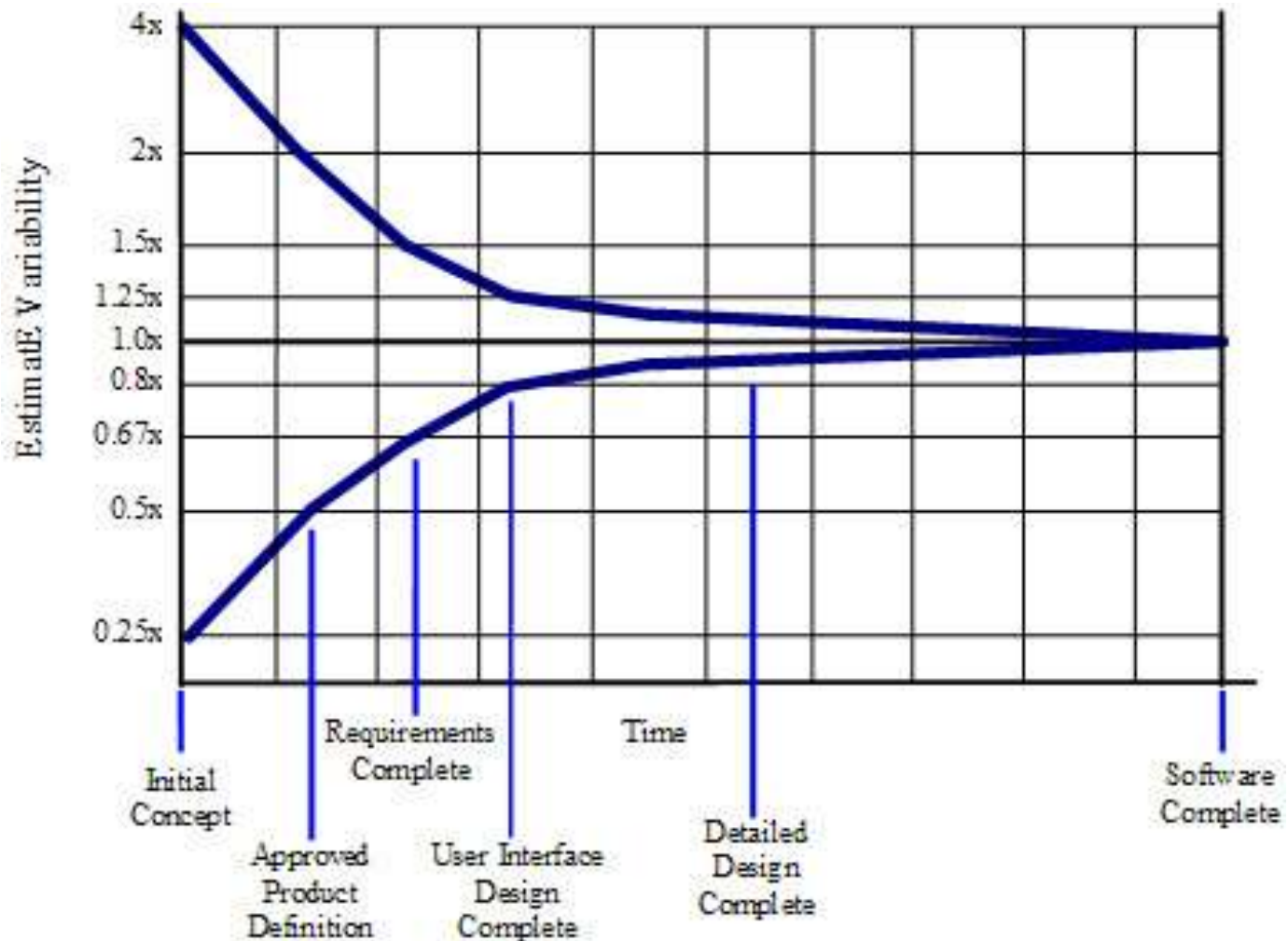
השפעת הערכה



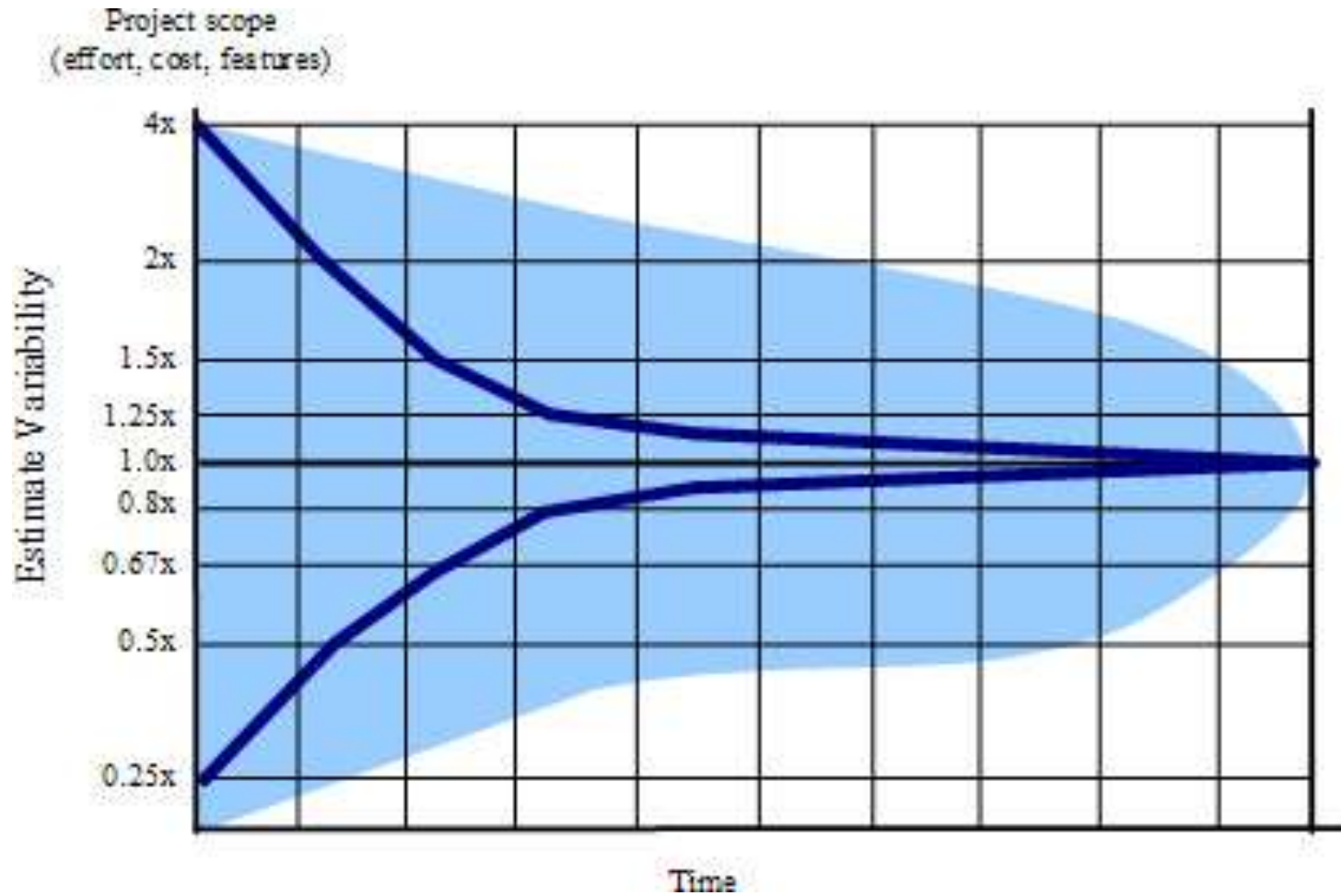
חוקי "יסוד"

- [חוק פרקינסון](#): משימה לוקחת את כל הזמן שניתן לה
- [סינדרום הסטודנט](#): כאשר ניתן זמן ארוך למשימה, מתחילים לעבוד עליה ברגע האחרון האפשרי... ומאחרים
- Planning Fallacy (D. Kahneman, [wikipedia](#)):
“a tendency for people and organizations to **underestimate** how long they will need to complete a task, even when they have **experience** of similar tasks over-running.”

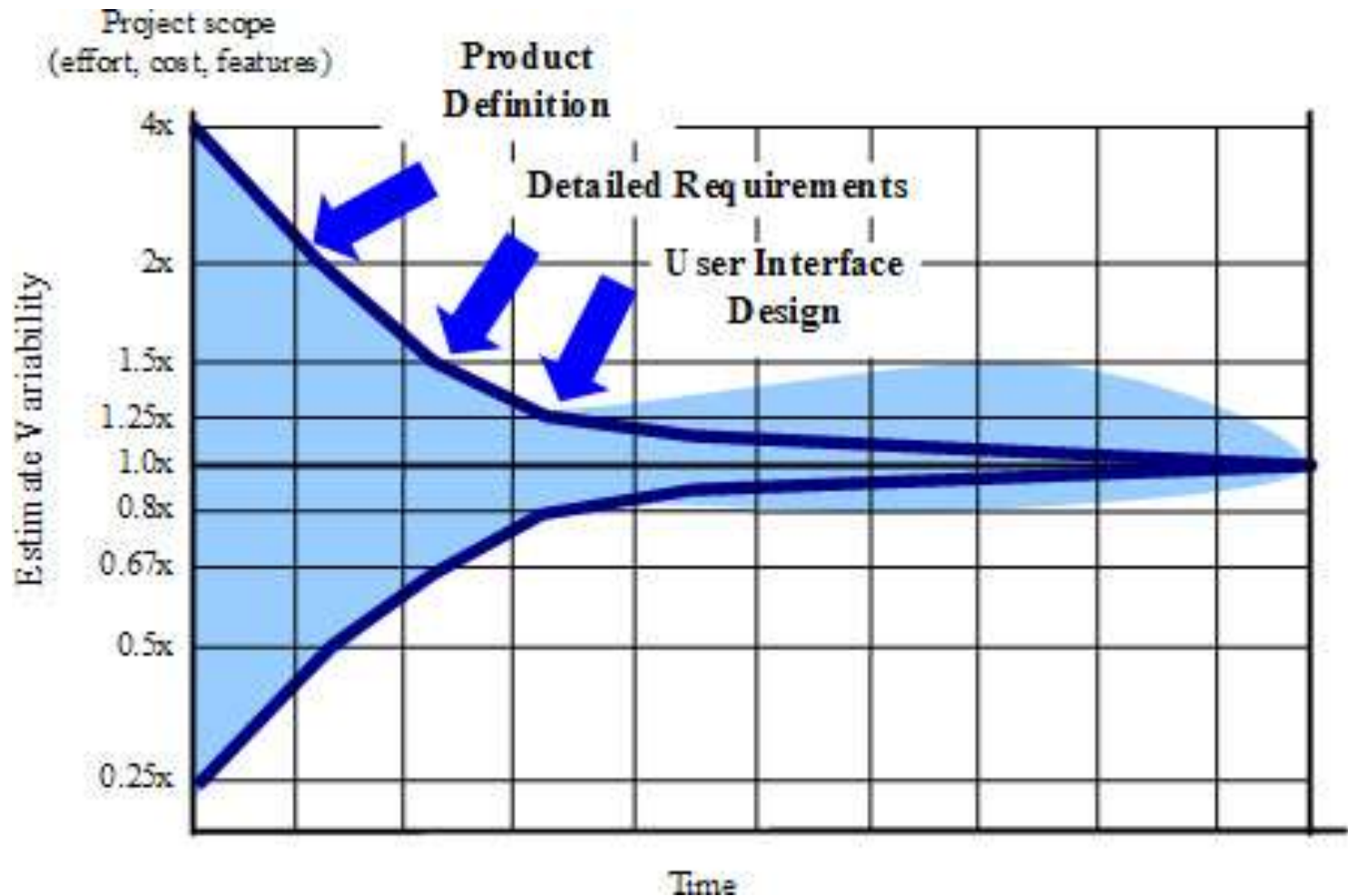
משפך אי-הוודאות



משפך אי-הוודאות

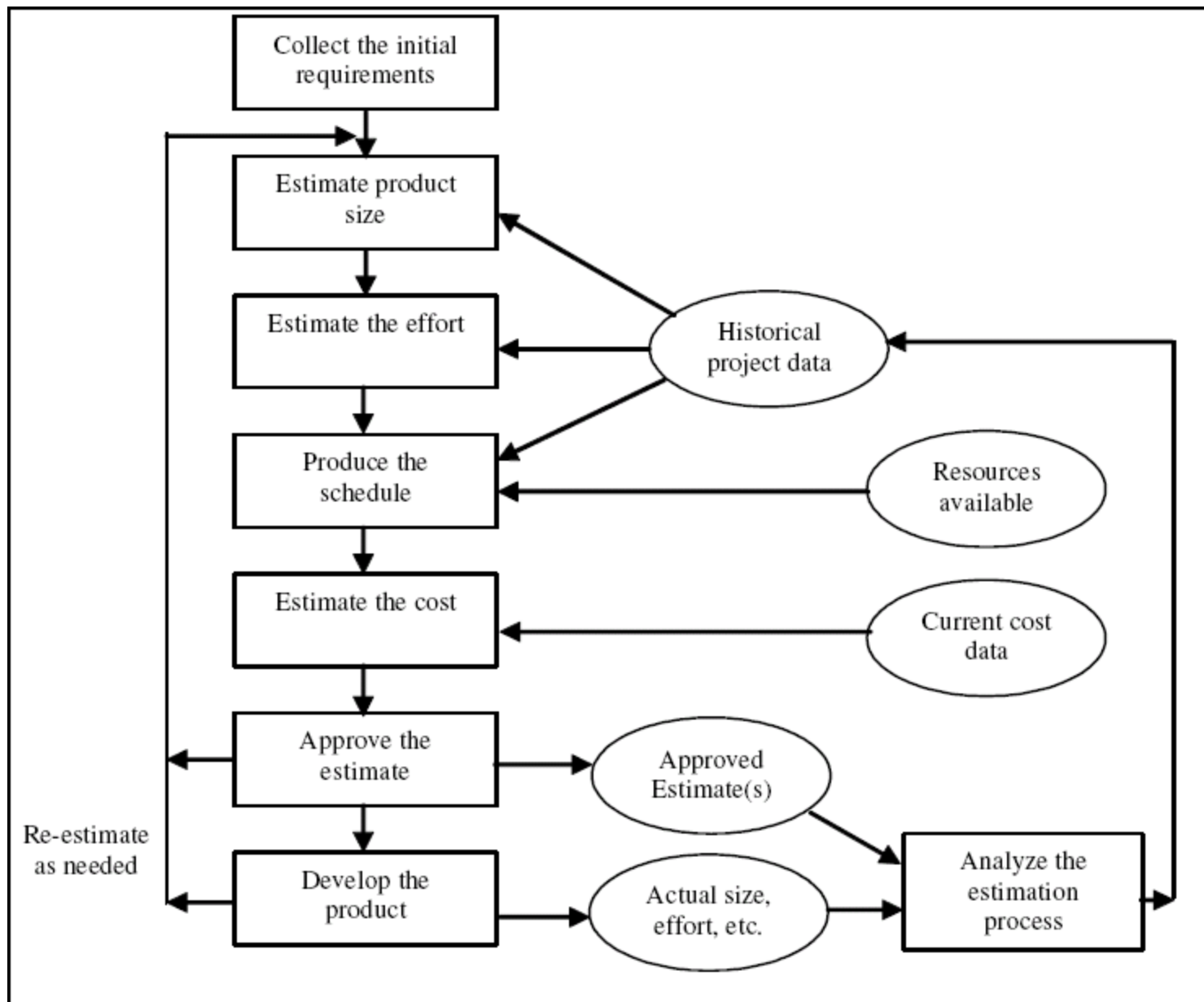


משפך אי-הוודאות



שלבי יצירת הערכה

- הערכת **גודל** המוצר
- הערכת **המאמץ** הנדרש (חודשי\שעות אדם)
- הערכת **לוח הזמנים** בחודשי יומן
- הערכת **עלות** הפרויקט בש"ח
- עדכון ההערכות לפי התקדמות



1. הערכת גודל המוצר

- השוואה לפרוייקט בעבר
- הערכת מספר שורות קוד (SLOC)
- הערכת מספר פונקציות (Function Points)
- Use-case \ Story Points (RUP, Agile)
- התייעצות עם מומחים

האם SLOC מדד טוב?

- יש מתאם אך שונות גדולה
- תלוי בסוג הפרויקט (אינטרנט\זמן אמת)
- תלוי בשפת תוכנה*
- גודל (למשל שימוש חוזר)
- צוות

Language	Statements relative to C
C	1:1
C#	1:2.5
Fortran95	1:2
Java	1:2.5
Macro assembly	2:1
Perl	1:6
SQL	1:10

Source: Microsoft

2. הערכת המאמץ הנדרש

- תלוי בגורמים רבים ולא רק בגודל הפרויקט
- אפשר להשתמש בנתונים היסטוריים
- אם אין כאלו אז בגישה אלגוריתמית, למשל:
- COCOMO (Constructive Cost Model)
 - פותח ע"י Boehm בשנות ה-80
 - התפתח לגרסה 2 (agile +)
 - פשוט, כלים זמינים, מבוסס רגרסיה
 - הנחה: עלות הפרויקט תלויה בגודל הקבצים
 - לוקח בחשבון מאפיינים שונים



נוסחה בסיסית של COCOMO II

- $\text{Effort (person/mon.)} = (2.94 \times a) \times K \text{SLOC}^b$
- a - “effort factor”
- b - “scaling factor”
- דוגמאות להתאמת מאמץ (מתוך 17)
 - גודל DB
 - שמישות נדרשת
 - נסיון בשפה ובכלים
- התאמת סדר הגודל (מתוך 5)
 - בשלות תהליכים
 - רוח הצוות

בואו ננסה

- Basic:
 - http://sunset.usc.edu/research/COCOMOII/cocomo81_pgm/cocomo81.html
 - <http://cost.jsc.nasa.gov/COCOMO.html>
- Real:
http://sunset.usc.edu/research/COCOMOII/expert_cocomo/

שיטות נוספות

- PERT
- [Wideband Delphi](#)
- Monte Carlo Simulation...

3. הערכת לוח הזמנים בחודשי יומן

- האם לא מספיק לחשב תאריך התחלה + מאמץ \ מספר מפתחים = תאריך סיום?
- Cocomo:
$$\text{Duration} = 3.67 * (\text{Effort})^{\text{SE}}$$

– Effort מהנוסחה

– SE מקדם סדר הגודל (וגודל קבוצה)

Agile Estimation

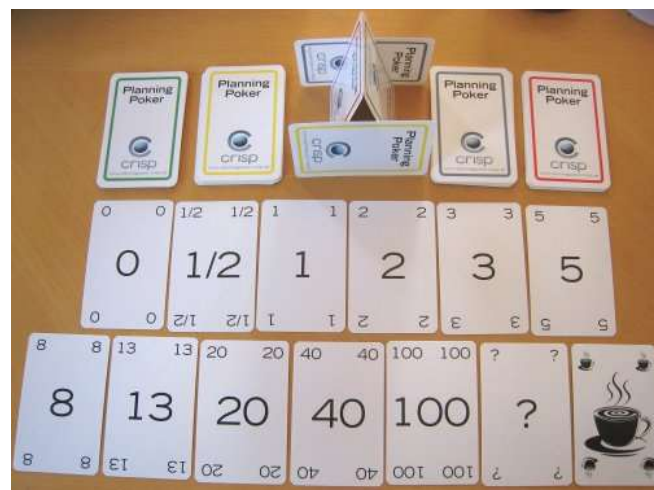
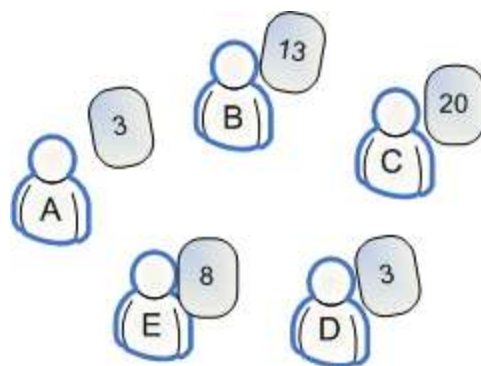
- הערכה לפני מימוש כל סיפור

- כל הצוות משתתף

- איך? למשל: פוקר התכנון

planningpoker.com

old.crisp.se/planningpoker



הערכה אג'ילית

- מי שעושה את העבודה הוא זה שמעריך
- עבודת צוות
- שאיפה להסכמה (כיצד בד"כ יעריכו מתחילים?)
- “Less is more”
- הערכה בנקודות
- סדרות פשוטות (חזקות 2, פיבונאצ'י)
- משימה שחורגת שוברים לתת-משימות
- שיפור ההערכות תוך כדי - velocity

עוד: חטאי הערכה

- Steve McConnell

www.construx.com/Page.aspx?hid=2617



גיליון הערכה באקסל לדוגמא

- <http://cid-9c28ebc63295af67.office.live.com/view.aspx/Public/Estimation%20Madness/Estimation%20and%20Prioritization.xlsx> (*)

סיכום הערכה

- קשה להגיע להערכה טובה
- אך יש צורך למניעת הפתעות ותיאום עם אחרים
- לקחת בחשבון את סוג הפרויקט, הסביבה ועוד
- לנסות להימנע מטעויות..
- עוד

– תכנון הלו"ז

– ניהול סיכונים

Brooks' Law, "adding people to a late project only makes it later"

What to say when asked for an estimate?

"I'll get back to you"

Pragmatic Programmer, p. 68

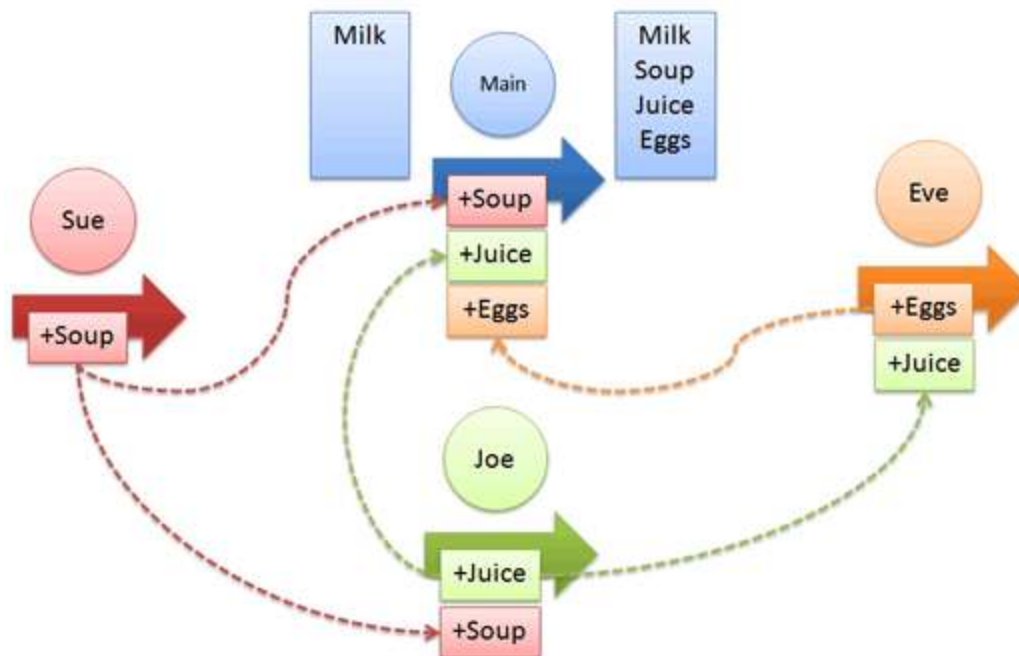
I will not ship SH%T

I will learn to say NO

When I give an estimate I will not LIE

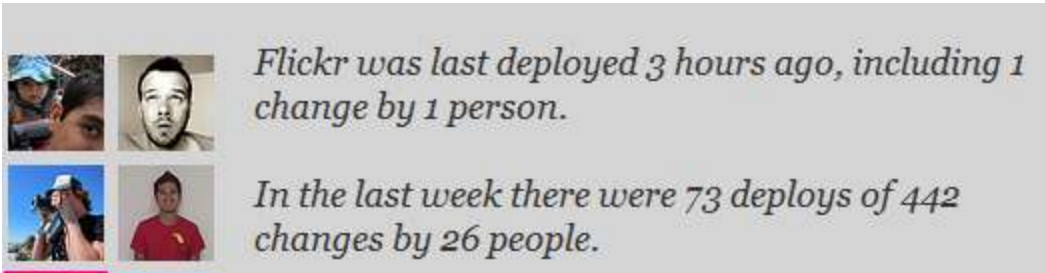
Robert Martin (SCNA'12):

2. תכנון ובקרת גרסאות קוד



כיצד לחלק פרויקט?

- אבן דרך – milestone
- שחרור – release ([flickr 10/d](#), chrome, [facebook](#))
- איטרציה \ ספרינט
- יום \ שעה
- ראו גם:
http://en.wikipedia.org/wiki/Software_release_life_cycle
- אצלינו?



סבב 0 - ZFR

• Zero Feature Release

– רשימת נושאים \ סיפורים לעבודה – Backlog

• איטרציה 1 מפורטת - Sprint Backlog

– מערכת בקרת גרסאות קוד (ומסמכים)

• תשתיות כלליות נצרכות להמשך

– תחילת תיעוד

• האם צריך ZFR? עוד דין

סבב מס' 1 - MVP

- **הצגת תרחיש עיקרי מוכן**
- **איך קובעים על מה לעבוד?** – Minimum Valuable Product/Main UC
- **באחריות מי לתעדף?**
- **מה עושים אם לא מספיקים?**
- **משימות משנה לפי חומרי ההרצאה**
 - סבב 2 – בדיקות יחידה
 - סבב 3 – Refactoring / Usability
 - סבב 4 – Stable/Final Release
- **בסיום כל סבב:**
 - רטרוספקטיבה (מה היה לנו?) ותכנון (מה הלאה?)

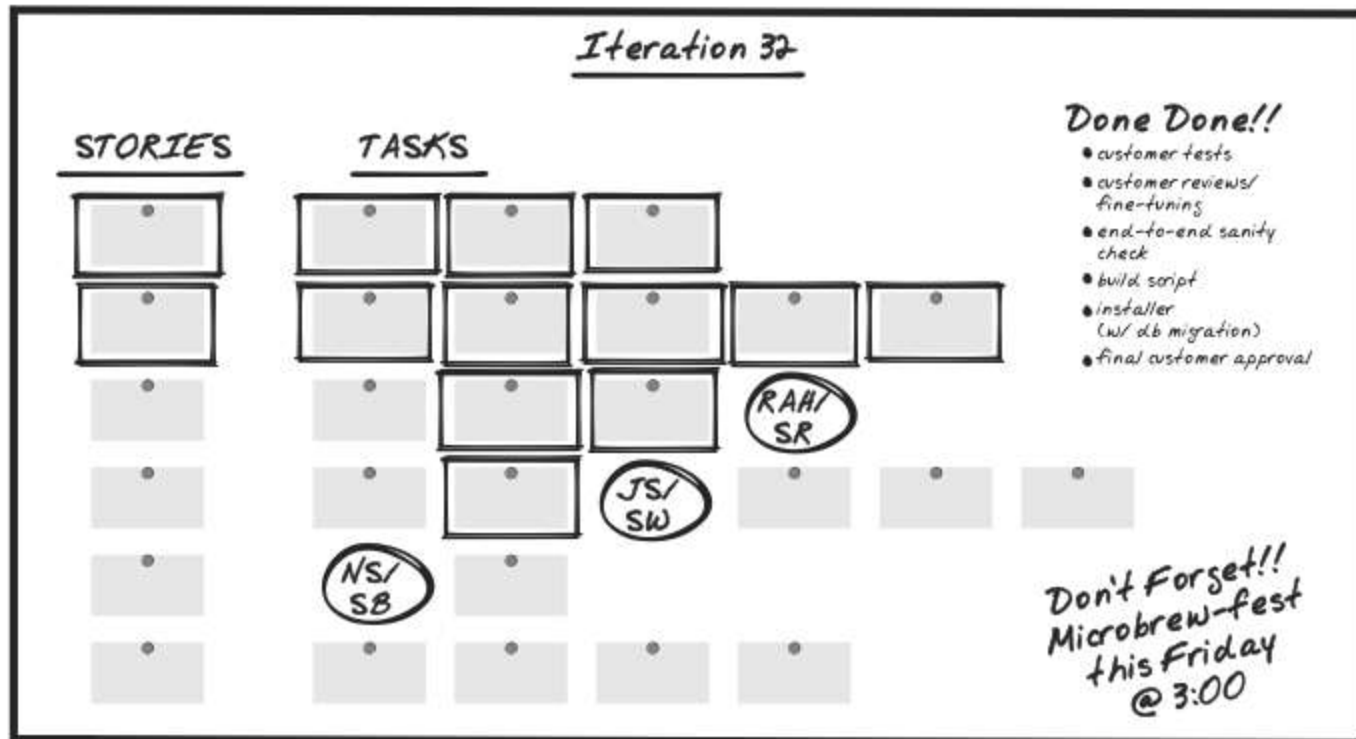
כלי עזר להנדסת תוכנה

- CASE: Computer Aided Software Engineering
- מערכת ניהול פרויקט ברשת, למשל GitHub
- אחרים: google-code, CodePlex, BitBucket
- <http://trac.edgewall.org/>
- <http://www.redmine.org/>
- <http://gforge.org/gf>
- <http://www.pivotaltracker.com/>

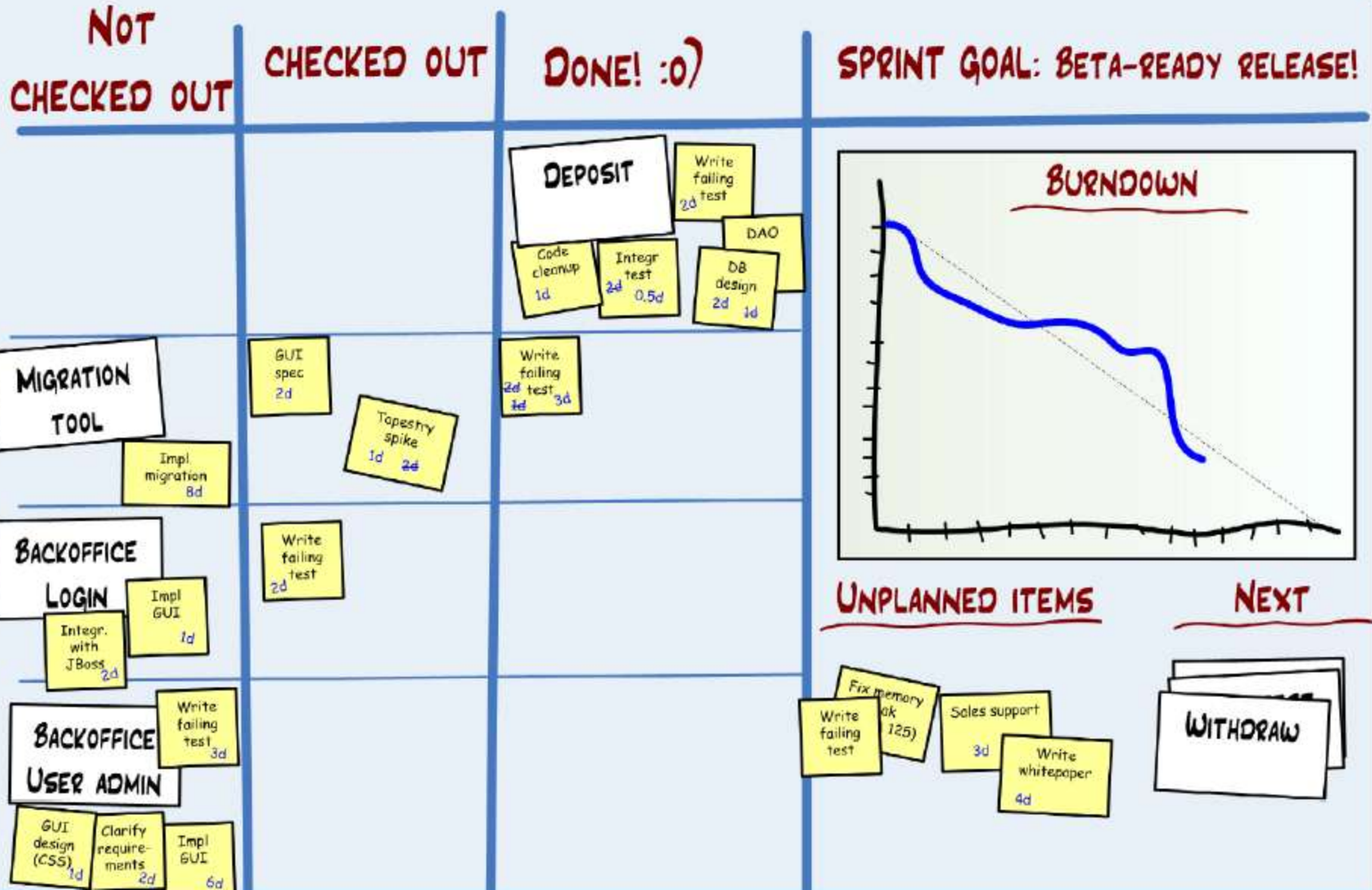
Planning with a Backlog

	Item #	Description	Est	By
Very High				
	1	Finish database versioning	16	KH
	2	Get rid of unneeded shared Java in database	8	KH
		- Add licensing	-	-
	3	Concurrent user licensing	16	TG
	4	Demo / Eval licensing	16	TG
		Analysis Manager		
	5	File formats we support are out of date	160	TG
	6	Round-trip Analyses	250	MC
High				
		- Enforce unique names	-	-
	7	In main application	24	KH
	8	In import	24	AM
		- Admin Program	-	-
	9	Delete users	4	JM
		- Analysis Manager	-	-
	10	When items are removed from an analysis, they should show up again in the pick list in lower 1/2 of the analysis tab	8	TG
		- Query	-	-
	11	Support for wildcards when searching	16	T&A
	12	Sorting of number attributes to handle negative numbers	16	T&A
	13	Horizontal scrolling	12	T&A
		- Population Genetics	-	-
	14	Frequency Manager	400	T&M
	15	Query Tool	400	T&M
	16	Additional Editors (which ones)	240	T&M
	17	Study Variable Manager	240	T&M
	18	Haplotypes	320	T&M
	19	Add icons for v1.1 or 2.0	-	-
		- Pedigree Manager	-	-
	20	Validate Derived kindred	4	KH
Medium				
		- Explorer	-	-
	21	Launch tab synchronization (only show queries/analyses for logged in users)	8	T&A
	22	Delete settings (?)	4	T&A

Iteration Planning Board

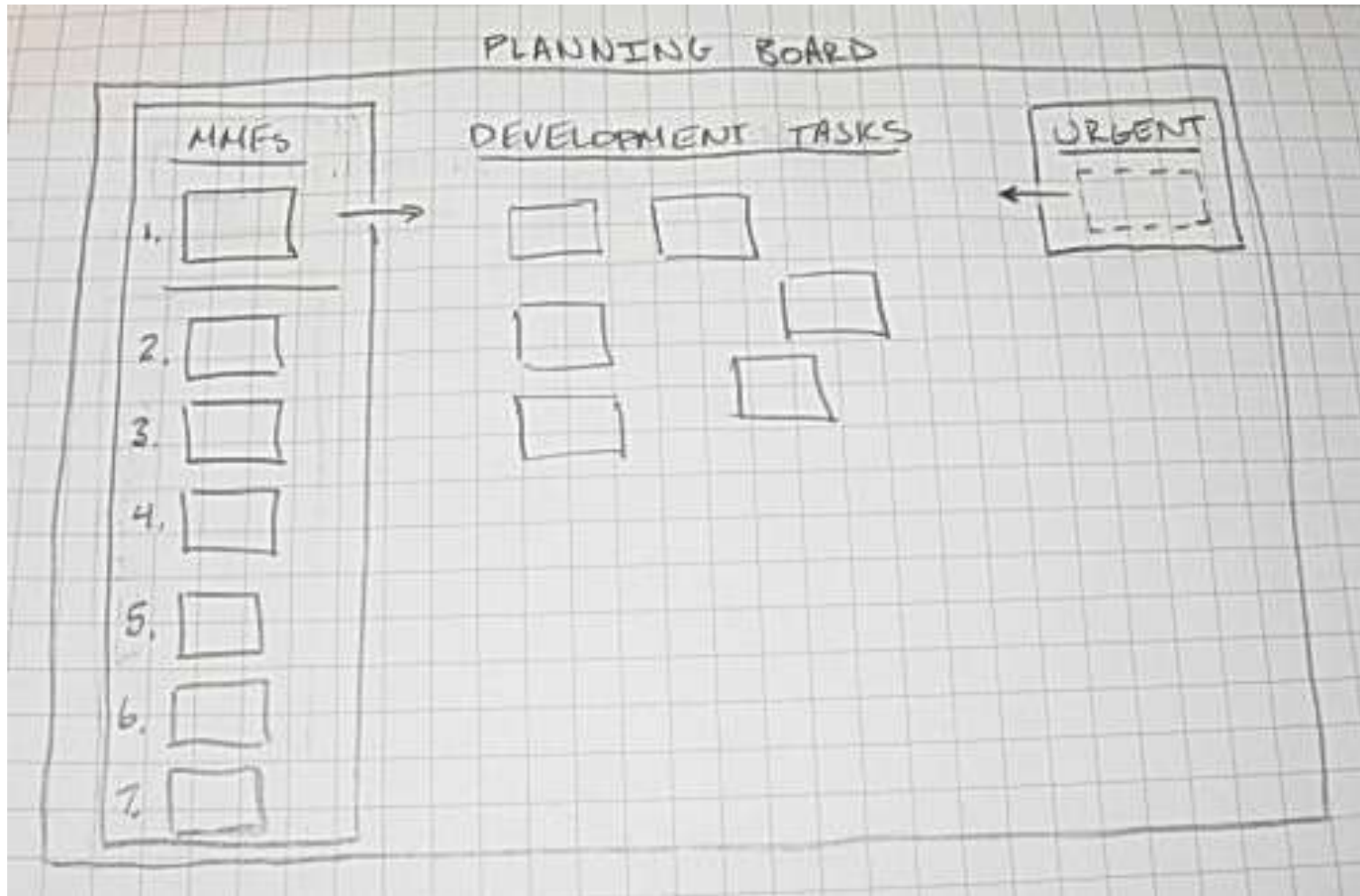


Burn Down Chart





Kanban Boards



EBoard

Teams Board

Kanban Board, Prioritizing and Assigning Team Members all on one Board

- www.infoq.com/articles/agile-kanban
- Fogbugz, CodePlex, PivotalTracker, <http://asana.com/>, leankitkanban.com, kickstarter.com, agilezen.com/, jetbrains.com/youtrack/, leankitkanban.com
- Simple g-spreadsheet board kanban.ws/, trello.com (used e.g. by uservoice, no sprints)



• בפרויקט שלכם:

• דף איטרציה + <http://gissues.com/>

Github

- ויקי
- הורדות
- חברתי \ גרופים
- ✓ ניהול משימות
- בקרת גרסאות
- תרומות
- סקרי קוד (בהמשך)
- קישור לאתרי רזומה
- חינוך לקוד פתוח

בקרת גרסאות – Version Control

- איך (האם?) אתם שומרים גרסאות של עבודה שלכם?
- האם אפשר לשפר?
- האם יש הבדל בין מפתח בודד לחברה גדולה?
- שמות שונים:
 - בקרת תצורה
 - Revision Control
 - Software Configuration Management
 - Source-Code/**Version Control System**

Joel Test (~2000)

1. Do you use source control?

...גם היום...

"You've just spent twenty minutes doing a presentation for your teammates on **adopting source control**. Yeah, they don't do source control at all. Yep, **not at all—it's as if the last 20 years of computer science never happened**. But better late than never, and frankly any source control is better than none, because disaster is one errant delete away."

- "Driving Technical Change: Why People on Your Team Don't Act on Good Ideas, and How to Convince Them They Should", chap. [The Cynic](#)

בקרת גרסאות – בשביל מה? יעדים

- שיתוף מספר מפתחים (מרוחקים) בו זמנית
- לדאוג שלא יהיו סתירות בין המפתחים
- איסוף כל הגרסאות ומעקב אחרי שינויים

– מאפשר מחיקת קוד

- ניהול מספר גרסאות במקביל
- גיבוי

- מאגר מעודכן של תוצרי הפרויקט

– Continuous Integration\daily build

בפרויקט תדרשו להדגים את בקרת
התצורה שלכם



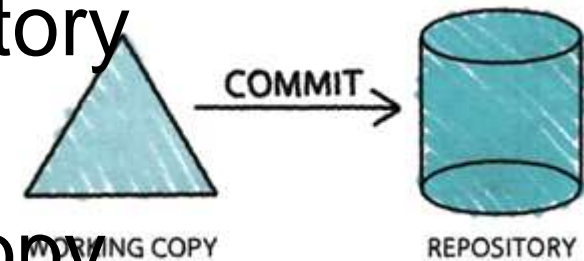
היסטוריה (השוואה)



Generation	Networking	Operations	Concurrency	Examples
1	None	One file at a time	Locks	RCS, SCCS
2	Centralized	Multi-file	Merge before commit	CVS, SourceSafe, Subversion, Team Foundation Server, IBM Rational ClearCase
3	Distributed	Changesets	Commit before merge	Bazaar, Git, Mercurial

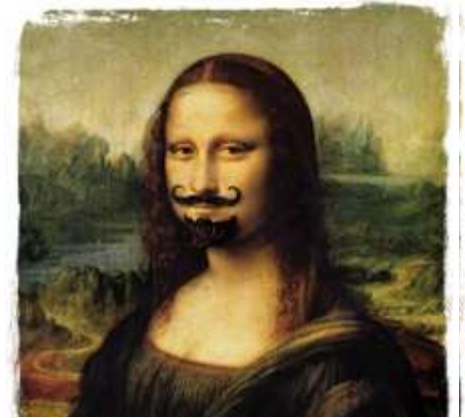
CVCS - פעולות עיקריות

- **Create:**
repository = filesystem * time
- **Checkout:** Create a working copy
- **Commit:** Apply the modifications in the working copy to the repository as a new changeset
- **Update:** Update the working copy with respect to the repository



18 פעולות עיקריות

- **Add**: a file or directory
- (**Edit**): modify a file
- **Delete**: a file or directory
- **Rename**: a file or directory
- **Move** (""): a file or directory



18 פעולות עיקריות

- **Status**: list the modifications that have been made to the working copy
- **Diff**: show the details of the modifications that have been made to the working copy
- **Revert**: undo modifications that have been made to the working copy
- **Log**: show the history of changes to the repository



18 פעולות עיקריות

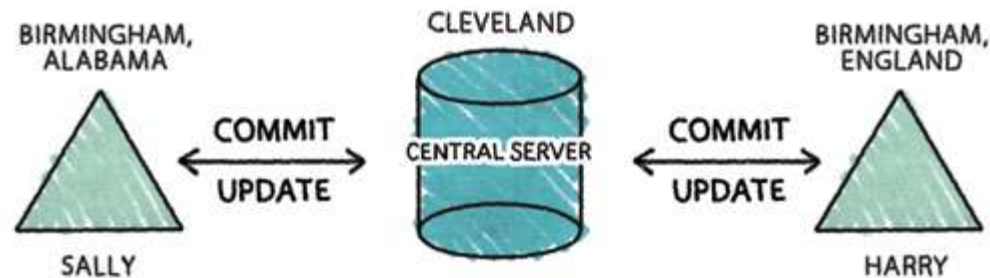
- **Tag**: associate a meaningful name with a specific version in the repository
- **Branch**: create another line of development
- **Merge**: apply changes from one branch to another
- **Resolve**: handle conflicts resulting from a merge
- **Lock**: prevent other people from modifying a file



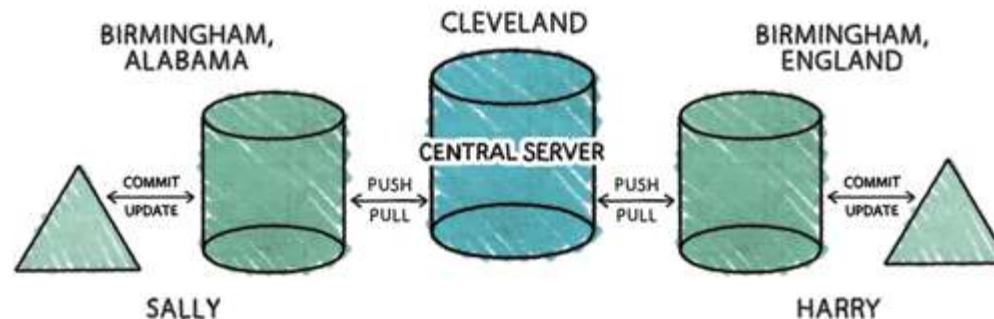
DVCS – עוד 3 פעולות

- **Clone**: create a new repository instance that is a copy of another

Centralized:

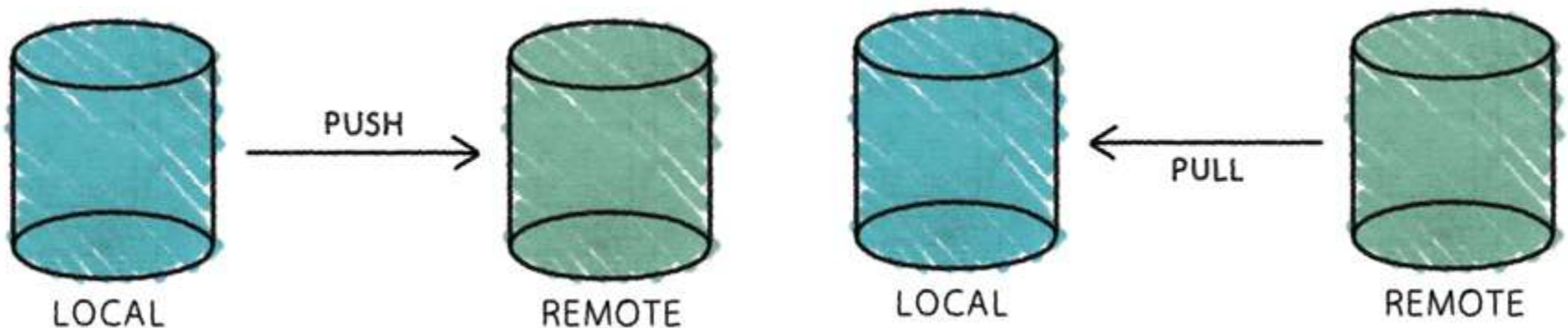


Distributed:

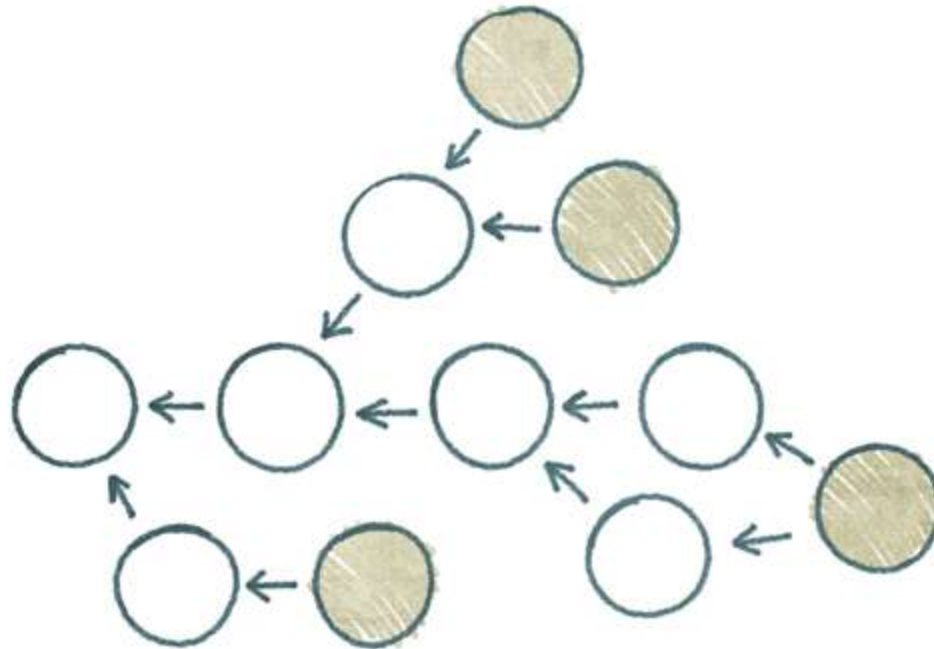


DVCS

- **Push**: copy changesets from a local repository instance to a remote one
- **Pull**: copy changesets from a remote repository instance to a local one

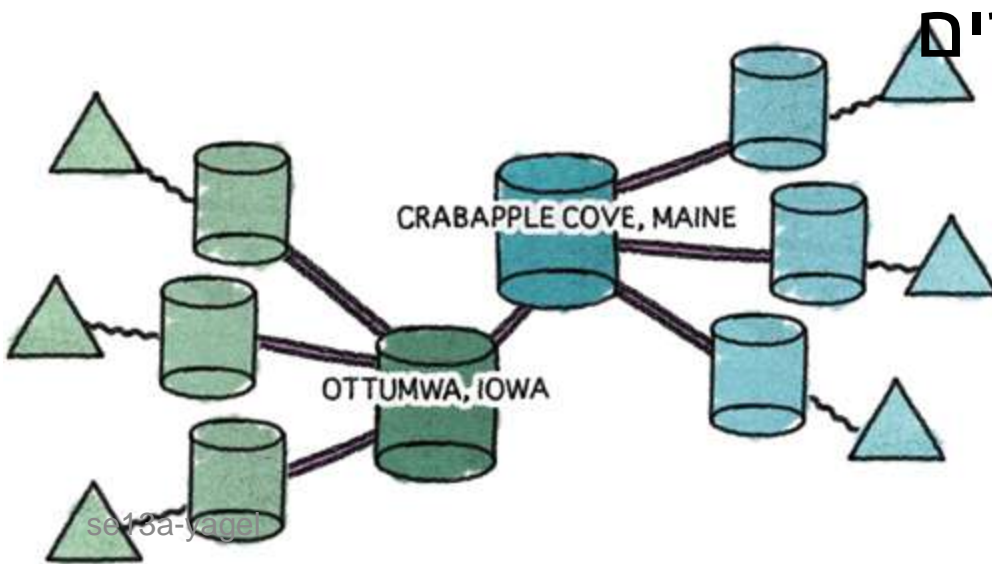


DAG!



DVCS יתרונות

- עותק פרטי של מאגר לכל מפתח (שינויים, גיבוי, מדרגיות השרת)
- מהירות ([בדיקה](#))
- Offline
- תמיכה בצוותים מבוזרים
- גמישות בגרסאות
- מיזוג קל יותר



DVCS - חסרונות

- חדש ובפיתוח (GUI)
- עקומת למידה
- ללא נעילות (כשצריך)
- גודל מאגרים
- ניהול \ אבטחה \ הרשאות
- תיאום לכלים קיימים
- קשה להעלים קבצים (כשצריך)
- ובכל זאת ~~כנראה העתיד~~ ההווה...

Git makes more sense when you understand [...]



@KentBeck

Kent Beck

finally figuring out that git commands are
strangely named graph manipulation
commands--creati
moving pointers at



@tabqwerty

chi wai lau

1 Mar via TweetDeck ☆ Unfavorite ↻

"git gets easier once you get the basic idea
that branches are homeomorphic
endofunctors mapping submanifolds of a
Hilbert space."

9 Mar via web ☆ Unfavorite ↻ Undo Retweet ↻ Reply

Mapping to git

<u>Operation</u>	<u>Git Command</u>
Create	git init
Checkout	^[a]
Commit	git commit -a ^[b]
Update	git checkout ^[c]
Add	git add ^[d]
Edit	git add ^[e]
Delete	git rm
Rename	git mv
Move	git mv
Status	git status
Diff	git diff
Revert	git checkout ^[f]
Log	git log
Tag	git tag
Branch	git branch
Merge	git merge
Resolve	^[g]
Lock	^[h]
Clone	git clone
Push	git push
Pull	git fetch ^[i]

Git clients (Windows)

- CLI Shell: Git Bash
- Windows Shell
- Github got Windows
- IDE Integration

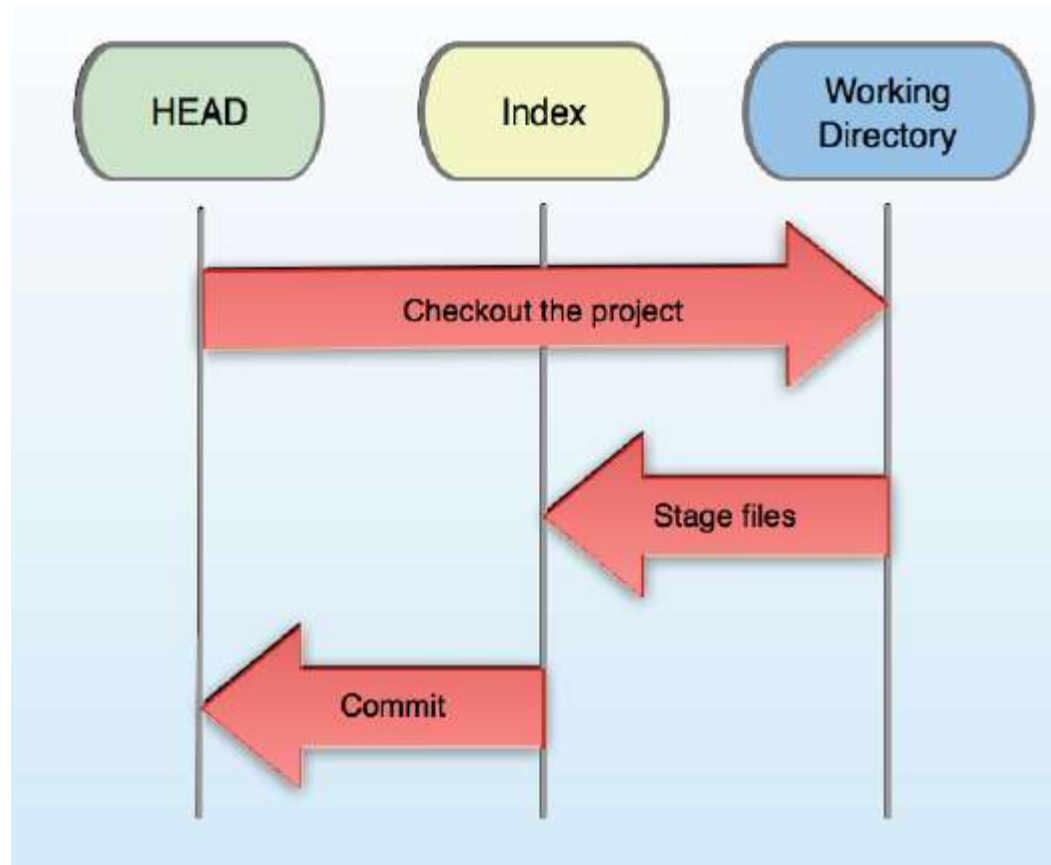
הדגמה

- <http://learn.github.com/p/intro.html>

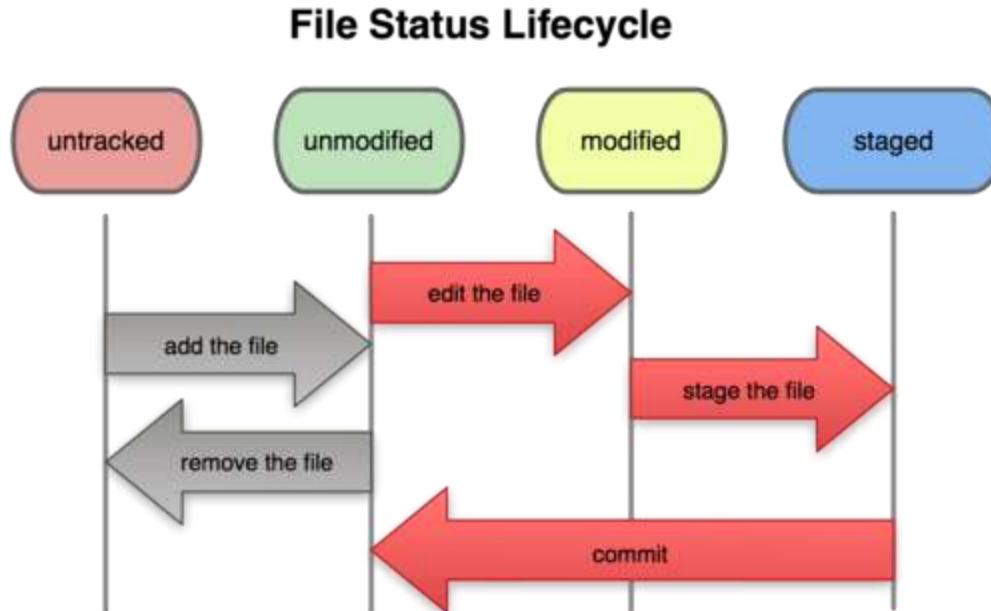
הדגמה - טיפים

- <http://help.github.com/create-a-repo/>
Local user settings:
git config user.name <user>
git config user.email user@example.com
- git pull == “pull (fetch)”+”update (checkout)”
- git add: add / stage
- git commit -a == add+commit

Git – 3 Areas



File Status Lifecycle



DVCS for OSS

- Git hub flow
- למשל תהליך העבודה בפרויקט Nuget:
[Contributing a Bug Fix or Feature](#)

בפעם הבאה

- עוד על אג'ייל וסקראם
 - מעבר לפיתוח איטרטיבי
- בקרת גרסאות II – תרחישים נוספים עם git
- פרויקט: סבב 0 - ZFR
 - תכנון המשך הפרויקט
 - העלאת תשתית קוד ראשונה
 - תכנון סבב 1
 - סקר ZFR בשבוע הבא (הרצאה + תרגיל, יש להירשם)
- משימה אישית מס' 4:
 - קורס מקוון קצר <http://try.github.com>
 - להגשה במייל: קישור לתגית סיום הקורס עם שם המשתמש שלכם ב-github (דוגמא)

לסיכום

- הערכה
- תכנון
- בקרת גרסאות
- כלים – האם תורמים? האם יתרמו?
- המשך הפרויקט: 4 סבבי פיתוח
 - רטרוספקטיבה ותכנון
 - משימות משנה
- בהרצאה:
 - בדיקות, חווית משתמש
 - עקרונות עיצוב מונחה עצמים
 - כלים ושיטות נוספים...