

Claude Code: loops, על Boris ROI והצוואר בקבוק הבא

שיחת במה של 40 דקות עם ראש Claude Code על איך הוא עובד היום, ואיך Anthropic מטפלת ב-review, ב-security ובעלויות כשכל הקוד כבר נכתב על ידי AI

Boris (ראש Anthropic, Claude Code) בשיחה עם מנחה מצוות DevInfra של Meta · כנס Scale@ (לפי הלוגו על הבמה, השם המלא לא זוהה) · הועלה מחדש ב-X על ידי 2026-06-19 · Movez

משך: כ-41 דקות · X

סיכום אוטומטי — AI Briefing (aibriefing.dev)

על המסמך הזה

זו שיחת במה (fireside chat) של כ-40 דקות בכנס Scale@. מראיין מנהל מוצר בצוות DevInfra של Meta, שעובד על פרודוקטיביות מפתחים עם AI. שמו לא נאמר בהקלטה. המראיין הוא Boris, ראש Claude Code ב-Anthropic. שם המשפחה Cherny מופיע רק בכיתוב של הפוסט ב-X, לא בהקלטה עצמה. השניים עבדו בעבר יחד על Threads. החלק הראשון מורכב משאלות של המנחה, והשני משאלות שהקהל שלח ודירג (לדברי המנחה הגיעו 107 שאלות).

הטענה המרכזית של Boris היא שכתיבת הקוד כבר לא צוואר הבקבוק. אצלו 100% מהקוד נכתב על ידי Claude Code מאז Opus 4.5. השלב הבא הוא loops, כלומר agents שרצים בלולאה ומפעילים agents אחרים, ומשם עוברים לצוואר הבקבוק הבא: CI, security, code review, ורעיונות. לגבי עלויות הוא אומר שצריך למדוד ROI ולא עלות, להשתמש במודל הכי חזק ולהתמקד בהגדלת התשואה.

איך לקרוא את המסמך

- **1. תקציר מנהלים:** הטענות המרכזיות עם חותמות זמן, ומה לא נאמר.
- **2. Loops:** ההגדרה של Boris ל-loops כמדרגת הפשטה, ואיך הוא משתמש בהם היום.
- **3. ROI ותקציבי tokens:** איך לפרוס, מתי לשלוט בעלויות ואיך למדוד תשואה.
- **4. Claude Code: Cework:** ללא מהנדסים, ושני מקרי שימוש של Boris.
- **5. Fable ובחירת מודל:** הקפיצה שהוא מתאר מעל Opus 4.8, ולמה הוא משתמש רק במודל הכי יקר.
- **6. צווארי בקבוק אחרי הקוד:** CI, Claude Security, Code Review ו-dynamic workflows.
- **7. תחזוקה, auto mode ולמידה:** loops לתחזוקה, עייפות מבקשות אישור, output styles.
- **8-9. פעולות, שאלות פתוחות ומילון מונחים.**

הערות מקור המסמך הופק אוטומטית מהתמלול של ההקלטה ומהפריימים שלה. בשיחה אין שקפים, וכל הפריימים הם צילומי במה. ה-ASR שיבש שמות ומונחים: Claude מופיע בתמלול כ-cloud, quad, ו-clod, ו-Cowork כ-co-work. שם המודל Fable מופיע כך באופן עקבי, אבל לא ניתן לאמת אותו מהמקור. הציטוטים מועתקים מילה במילה מהתמלול, כולל השיבושים, ונבדקו מול חותמת הזמן שלהם. לא נוסף שום מידע ממקורות חיצוניים. הכיתוב של הפוסט ב-X לא שימש כמקור לעובדות.

1. תקציר מנהלים

Boris מציג תמונה של צוות שכבר עבר את שלב השאלה אם AI כותב את הקוד. מבחינתו השאלה עכשיו היא מה חוסם את הערך העסקי אחרי שהקוד כבר נכתב. כמעט כל תשובה שלו חוזרת לאותו דפוס: מזהים את צוואר הבקבוק הבא, בונים עליו מוצר פנימי בעזרת Claude, ואחר כך משחררים אותו ללקוחות.

- **כל הקוד שלו נכתב על ידי Claude Code:** מאז Opus 4.5 (נובמבר בשנה הקודמת). השנה 1.7 אלף PRs, 400 אלף שורות שנוספו ו-250 אלף שנמחקו, ו-8 מיליארד tokens מאז מרץ. רוב הקידוד שלו נעשה היום מהטלפון [1:30].
- **Loops הם מדרגת ההפשטה הבאה:** קוד הוא agent, statement, function, ו-loop הוא higher-order function. כ-30% מהקוד שלו ביום ממוצע נכתב על ידי loops [9:07].
- **8x קוד למהנדס ב-Anthropic:** זה הנתון שנמסר מתחילת השנה. ב-Anthropic כולה 80 עד 90% מהקוד נכתב על ידי Claude Code, ובצוותים רבים 100% [6:06, 18:12].
- **מודדים ROI, לא עלויות:** לתת tokens לכולם, כולל PMs, מעצבים ואנשי data, ולשלוט בעלויות רק אחרי שמקרה שימוש מוכיח את עצמו, ובצד ה-back end [3:02–4:34].
- **המודל הכי יקר עדיף על חיסכון:** advisor model יכול לחסוך בערך 50% מההשקעה, אבל לדבריו פוטנציאל התשואה הוא 1,000% ויותר [19:43].
- **Code review אוטומטי:** Claude Code Review הוא המוצר שמשמש בכל PR ב-Anthropic. הוא יקר יותר ממתחרים, ולדבריו תופס 98–99% מהבאגים [25:56].
- **Claude Security תופס דברים שבדקי חדירות פספסו:** הוא רץ פעם בשבוע על כל ה-codebases ומתקן בעיות בעצמו [27:28].
- **Dynamic workflows:** הנחיה של משפט אחד האיצה את ה-CI. הריצה לקחה כמה שעות וכמה מיליוני tokens, ויצאו ממנה ארבעה PRs עם שיפור של כ-50% [29:00–27:28].
- **Auto mode במקום בקשות אישור:** מודל מאשר או דוחה כל פקודה. לדברי Boris זה בטוח יותר מבקשת אישור ידנית, כי אנשים מתעייפים ומאשרים בלי לקרוא, וזה מה שמאפשר ריצות של שעות וימים [38:10–36:37].

מה לא נאמר

- **אין שיטת מדידה ל-8x:** לא הוסבר מה זה "code per engineer" (שורות? PRs?), ואין נתונים על איכות או על תקלות בפרודקשן.
- **98–99% מהבאגים:** הטענה לא מגובה ב-benchmark או בהגדרה של מה נחשב באג. גם העלות של Claude Code Review לא צוינה.
- **איך loop רץ בפועל:** לא נאמר באיזו תשתית הוא רץ, עם אילו הרשאות, כמה הוא עולה, ומה קורה כשהוא פותח PRs גרועים.
- **Fable:** לא ברור מה סטטוס הזמינות שלו. המנחה אמר שהגישה ניתנה לשלושה ימים, ו-Boris קרא לזה "misunderstanding" שעובדים לתקן. אין מספרים על מחיר או latency.
- **Prompt injection:** הנתון "כ-1% ב-100 ניסיונות" מיוחס ל-system cards בלי פירוט של איזה מודל ואיזה מבחן.
- **שיתוף פעולה:** השאלה שקיבלה הכי הרבה הצבעות נענתה ב-"we have a bunch of stuff on the fire" ובהמלצה להשתמש ב-MCP. אין תוכנית קונקרטית.
- **כשלונות:** Boris לא סיפר על אף מקרה שבו loop או workflow שבר משהו.

2. Loops: מדרגת ההפשטה הבאה



המנחה מ-Meta (משמאל) ו-Boris על הבמה, ומאחוריהם הלוגו SCALE@ עם הכיתוב החלקי "DATA &". אין שקפים בשיחה.

השאלה הגיעה מ-Threads, מקולגה לשעבר של השניים: האם loops הם hype או משהו אמיתי? Boris מתאר שלוש מדרגות. לפני שנתיים כתבו קוד ביד, אחר כך agents כתבו את הקוד, ועכשיו agents מפעילים agents שכותבים את הקוד. במונחי תכנות, source code הוא statement, agent הוא function ו-loop הוא higher-order function.

Boris [9:07] So loops are sort of like as big as the step from source code to agents was, loops are — the step from agents to the next thing. It's just as important and as big a step

הדוגמאות שלו: code review ביד, אחר כך agent שמבצע review לפי בקשה, ואחר כך agent שרץ בלולאה ומבצע את כל ה-reviews. או agent שכל חמש-עשר דקות קורא פידבק ב-Threads ופותח PRs עם תיקונים. לדבריו loops נמצאים היום בערך במקום שבו agents היו לפני שנה וחצי. הוא מעריך שכ-30% מהקוד שלו ביום ממוצע נכתב על ידי [10:39] "If I try really hard, I can get to 100% for some days, but it doesn't totally click yet." loops. כשהוא שאל את הקהל מי ניסה loops ו-routines, מעט ידיים עלו, והוא קרא לזה [9:07] "Still pretty nascent".

3. ROI ותקציבי tokens

המנחה הזכיר שחברות כמו Uber מתחילות להגדיר תקציב חודשי למהנדס, למשל 1,500 דולר, בזמן שהמודלים מתייקרים. Boris מחלק את החברות לשתיים: חברות שחושבות על עלות וחברות שחושבות על ROI, ולדעתו השנייה היא המסגרת הנכונה. החברות המצליחות נותנות tokens לכולם, כי הרעיונות הטובים מגיעים לפעמים מ"an accountant somewhere in the corner of the org" [4:34] "accountant somewhere in the corner of the org". צריך גם לתת לאנשים ביטחון שלא ייענשו על ניסוי.

Boris [4:34] Once you find these internal use cases that kind of work, then you want to control the — costs. And you want to do that on the back end, not on the front end

- **כלי שליטה שהוזכרו:** בקרת עלות לכל seat, advisor models, החלפת מודל לכל הארגון, effort level ארגוני, ותקציבים לפי מחלקה או לפי RBAC [4:34].
- **מדידת תשואה:** פעם מדדו כמה אחוז מהקוד נכתב על ידי AI. כשמגיעים ל-100% המדד הזה כבר לא מועיל. בצוות DevInfra שיפור של 2–3% בשנה נחשב טוב, ועכשיו מדברים על מאות אחוזים [6:06].
- **סדר העבודה שהוא מציע:** להגיע ל-100% קוד מ-Claude, למדוד את ההאצה של קוד למהנדס, ואז לשחרר את צווארי הבקבוק הבאים: רעיונות (יותר PMs וחוקרי משתמשים) ו-go-to-market [6:06].

4. Cowork: Claude Code ללא מהנדסים

Cowork נמצא באפליקציית Claude ל־Mac ו־Windows, יחד עם chat ו־Boris Claude Code. אומר שמתחת זה אותו Claude Code ואותו Claude Agent SDK: "Co-work is quad code for non-engineers. It's just quad code under the hood." [12:10]. ההבדל הוא במעקות הבטיחות. יש VM שלם עם בידוד, חיבור למערכת ההפעלה שמנוע מחיקות בטעות, והגנות נגד prompt injection.

שני השימושים שלו: הראשון הוא ניהול פרויקט במקום Cowork. stand-up פותח spreadsheet של זרמי העבודה בדפדפן, שולח לכל מהנדס הודעה ב-Slack כדי לבקש סטטוס, וממלא את הטבלה. לפעמים מי שעונה הוא ה-Claude של המהנדס: "It's like the clods talking to the clods." [13:40]. זה עובד בלי הגדרות, רק Cowork ו-Claude Chrome extension. השני הוא הזמנת נסיעות. scheduled task יומי סורק את המייל ואת האירועים שאושר ב-Google Calendar. אם אירוע מתקיים מחוץ לטן פרנסיסקו, הוא מזמין טיסות ומלונות לפי ההעדפות של Boris, אחר אישור שלו. כך הוזמנו נסיעות רב-שלביות לטוקיו, ללונדון ולברלין לכנסי [15:11] Code with Claude.

5. Fable ובחירת מודל

לפי המנחה, הקהל קיבל גישה ל-Fable לכשלושה ימים בלבד. Boris ענה ש"it's a misunderstanding" ושעובדים להחזיר את הגישה [16:41]. הנסיבות לא הוסברו. הוא משווה את הקפיצה לרגע של Opus 4.5 בנובמבר, שבו הסיר את ה-IDE מהמחשב.

Boris [16:41] The leap from Opus 4.8 to Fable feels to me like a leap of at least that same size. — And I think it might be even a larger leap in model capability

לדבריו Fable מבין ניואנסים. הוא "שואל למה שלוש פעמים" בניתוח נתונים, ובודק השערות ב-I feel debugging. [18:12] "like I actually just ran out of hard problems to give it." הוא משתמש ב-Fable לכל המשימות. Anthropic כן סופרת tokens, כי כל token שהיא משתמשת בו לא נמכר ללקוח. ובכל זאת ההמלצה שלו: "just use the most expensive model and focus on how do I get more out of it?" [19:43]. advisor model Opus כברירת מחדל עם קריאה ל-Fable לפי הצורך, יכולים לחסוך בערך 50%. אבל כדי לשמור על חיסכון כזה צריך לכוון מחדש ולהריץ evals בכל דור מודלים. גם הוא מודה ש-advisors הם ברירת המחדל המומלצת.

איפה Fable עדיין חלש לדבריו: product sense ויצירת רעיונות, ותכנון של מערכות מבוזרות (שירותים, data flow, עומסים) [35:05]. מנגד, [35:05] "Its code is now better than the code I would have written" [35:05]. לגבי מתי הפערים ייסגרו: [35:05] "probably by the end of the year, it'll be quite good".

6. צווארי הבקבוק אחרי הקוד: review, security, CI

Boris מתאר את הדרך מקוד לפרודקשן ומשם למדד עסקי כשרשרת של צווארי בקבוק. בכל פעם שאחד נפתר, Anthropic בונה כלי פנימי לבא אחריו ומשחררת אותו כמוצר. בעזרת Code Review, כשמהנדס פותח PR הוא כבר לא מחפש בו באגים. הוא שואל אם ה-PR הזה צריך להתקיים בכלל. כלשונו: "I know there's no bugs, because Claude caught it and Claude fixed it." [25:56].

צוואר בקבוק	הכלי	מה נטען
כתיבת קוד	Claude Code	100% מהקוד של Claude Code ושל Cowork נכתב על ידי Claude Code מאז [18:12] Opus 4.5
Code review	Claude Code Review	רץ על כל PR ב-Anthropic. יקר יותר כי משתמש בהרבה tokens. תופס "99%, 98% of the bugs" [25:56]
Security	Claude Security	סריקה שבועית של כל ה-codebases ותיקון אוטומטי. תופס דברים שבודקי חדירות פספסו, מאז Opus 4.8 [27:28]
מהירות CI	Dynamic workflow	הנחיה של משפט אחד, כמה שעות, כמה מיליוני tokens, ארבעה PRs ושיפור של כ-50% (ה-ASR משובש בנקודה הזו) [29:00]

צוואר בקבוק הבא בתור	הכלי	מה נטען
	לא נבנה עדיין	יצירת רעיונות, ו-CI שיעמוד בסקייל [27:28]

use a workflow to look at my data, look at real CI timings from the data" במלואה: "Dynamic workflow [27:28]. set, and optimize CI to make it much faster." הוא פיצ'ר שהושק כמה שבועות לפני השיחה. Claude כותב תוכנית קטנה שרצה ב-VM ומתזמרת עשרות, מאות או אלפי sub-agents. מסביר שהבסיס הוא חוקי הסקיילינג: לשלושת הגורמים הקלאסיים (data, גודל רשת, compute באימון) נוסף רביעי, test time compute, שהוא "just a fancy way of saying how many tokens does the model generate" [33:34]. אפשר לכוון אותו דרך רמות effort (low עד max) או דרך Loop workflows. הוא משהו אחר: agent שרץ שוב ושוב על משימה חוזרת. Workflow מקצה יותר compute למשימה אחת.

7. תחזוקה, auto mode ולמידה

לשאלה על תחזוקה, Boris ענה שהוא משתמש ב-loops: שיפור ארכיטקטורה, איתור tests לא יציבים, מחיקת tests חסרי ערך ואיחוד של abstractions כפולות. כל אלה loops שרצים אצלו כבר עכשיו. הוא לא בודק מראש מה הם עומדים לעשות, רק את ה-PRs שהם פותחים. ואם התוצאה לא טובה, "use a", just add the magic words, workflow" [32:03].

לשאלה איך מונעים ממהנדסים להתעצל ולאשר הכל, יש לו שתי תשובות. הראשונה היא auto mode. צוות security-ב Anthropic שם לב שבקשות האישור פוגעות באבטחה, כי אנשים מאשרים בלי לקרוא.

— Boris [36:37] They just keep saying yes without actually reading the details

ב-auto mode כל בקשת אישור עוברת למודל, ש"ש" decides yes or no based on what you've said so far in" [36:37]. לדבריו זה בטוח יותר גם מ-dangerous mode וגם מבקשות אישור ידניות, ורוב המשתמשים כבר עובדים כך. ההגנה נשענת על עמידות המודל ל-prompt injection, "the success rate at 100-1% attempts is like around 1%" [38:10], ועל classifier שרץ על כל התעבורה. מה שזה מאפשר הוא ריצות של שעות וימים.

התשובה השנייה היא output styles. כל מהנדס חדש בצוות מקבל הנחיה להגדיר config output/style=exploratory, ש-Claude יסביר את הארכיטקטורה, את השפה ואת ה-codebase בכל שינוי. ל-learning-style, המיועד למי שלא מתכנת, Claude לא מבצע את השינוי בעצמו. הוא מלמד את המשתמש לבצע אותו צעד אחר צעד [39:40].

8. מה עושים מחר

- **לבדוק את מדד התשואה שלכם:** אם אתם עדיין מודדים כמה אחוז מהקוד נכתב על ידי AI, תעברו לקוד למהנדס ולזיהוי צוואר הבקבוק הבא [6:06].
- **להתחיל עם loop תחזוקה אחד:** tests לא יציבים או abstractions כפולות. ה-loop פותח PRs ואתם עושים להם review [32:03].
- **לנסות workflow על בעיה מבוססת data:** לדוגמה זמני CI, עם הנחיה קצרה שמפנה לנתונים אמיתיים [27:28].
- **לבחון auto mode** במקום בקשות אישור ידניות, במיוחד אם הצוות כבר מאשר אוטומטית בלי לקרוא [36:37].
- **Onboarding:** להגדיר exploratory output style למהנדסים חדשים [39:40].
- **שיתוף פעולה בינתיים:** לחבר את Claude Code ל-Slack, ל-Teams או ל-GChat דרך MCP [21:17].

שאלה פתוחה	למה זה חשוב
איך מודדים את ה-x8 ואת ה-98–99% מהבאגים?	בלי הגדרה אי אפשר להשוות לארגון שלכם
מה העלות של Code Review ושל workflow שרץ כמה שעות?	Boris ממליץ להתעלם מעלות, אבל לא נתן מספרים
מה קורה כש-loop פותח PRs גרועים בכמויות?	עומס ה-review עובר מכתובת קוד לסינון PRs
מה סטטוס הזמינות של Fable?	ההמלצה להשתמש ב-Fable לכל דבר תלויה בגישה אליו
האם auto mode מתאים לסביבות מוסדרות?	ההחלטה עוברת לשיקול דעת של מודל

9. מילון מונחים

מונח	משמעות
Loop	agent שרץ שוב ושוב על משימה חוזרת ומפעיל agents אחרים. אצל Boris זה higher-order function מעל agents
Dynamic workflow	תוכנית ש-Claude כותב ומריץ ב-VM כדי לתזמר עשרות עד אלפי sub-agents
Test time compute	כמות ה-tokens שהמודל מייצר בזמן ריצה כדי לשפר את התוצאה
Effort level	הגדרה (low עד max) שקובעת כמה tokens המודל מוציא
Advisor model	מודל זול כברירת מחדל שקורא למודל חזק רק כשצריך
Auto mode	מצב הרשאות שבו מודל מאשר או דוחה כל פקודה במקום המשתמש
Prompt fatigue	מצב שבו משתמשים מאשרים בקשות בלי לקרוא אותן
Output style	הגדרה ב-Claude Code שמשנה את אופן ההסבר. exploratory מסביר תוך כדי עבודה, learning מלמד במקום לבצע
Cowork	Claude Code ללא מהנדסים בתוך אפליקציית הדסקטופ, עם VM ובידוד
MCP	פרוטוקול לחיבור Claude לכלים חיצוניים כמו Slack
RBAC	הרשאות לפי תפקיד. כאן משמש להגדרת תקציבי tokens
Unhobble	להסיר מגבלה שמעכבת את המערכת. Boris משתמש במילה לשחרור צווארי בקבוק