איך מייסד אחד מנהל 20 coding agents במקביל — Ryan Carson, Untangle

ב-Untangle פיתוח התוכנה רץ בענן: manager agent אחד מקבל הוראות בעיקר בקול ומנהל צי של builders, ובשלב הגיוס מועמדים מצלמים את עצמם עובדים עם agent

Ryan Carson · AI Engineer · 9 בנובמבר 2026

משך: כ-15 דקות · YouTube

סיכום אוטומטי — AI Briefing (aibriefing.dev)

על המסמך הזה

זו הרצאה קצרה של Ryan Carson, מייסד ומנכ"ל Untangle, מכנס AI Engineer. לפי הבאנר בווידאו מדובר ב-breakout session ב-New York מ-14 באוקטובר 2026. Untangle בונה AI paralegal לדיני משפחה, ובעיקר לגירושין. החברה התחילה כאפליקציה לצרכנים ועברה מהר ל-B2B, והיום היא מוכרת למשרדי עורכי דין לדיני משפחה. Carson הקים ומכר ארבע חברות, והאחרונה שבהן גדלה לכ-110 עובדים.

הטענה המרכזית: ניהול agents דומה לניהול אנשים. Carson מציג את המבנה הארגוני שהם מריצים ב-production: מעל כל צוות builders יושב session של engineering manager שלא כותב קוד, עם automation שבודקת כל builder כל 15 דקות, תקרה של שלוש לולאות review, QA agent, merge queue manager וצעד אנושי אחד שאסור לדלג עליו, צפייה בסרטון של כל פיצ'ר. בחלק השני הוא טוען שגיוס מהנדסים צריך להשתנות מהיסוד: בודקים מועמדים לפי היכולת שלהם להפעיל agent, וזה נעשה דרך הקלטות מסך ולא דרך שיחות.

איך לקרוא את המסמך

- **1. תקציר מנהלים:** הטענות המרכזיות עם חותמות זמן, ומה לא נאמר.
- **2. הרקע:** Untangle, והחיבור בין ניהול אנשים לניהול agents.
- **3. ה-org chart:** manager agent, ticket board ועד 20 builders לכל manager.
- **4. Review, QA וסרטון:** תקרת שלוש לולאות, QA agent, והצעד האנושי שאין לו תחליף.
- **5. בקנה מידה של צוות:** עבודה בענן בלבד, sessions שמדברים זה עם זה, ו-merge queue manager.
- **6. גיוס:** סרטון של 45 דקות, דירוג אוטומטי ופרויקט ניסיון בתשלום.
- **7. מה עושים עכשיו:** פעולות מומלצות ושאלות פתוחות.
- **8. מילון מונחים**.

הערת מקור המסמך נוצר אוטומטית מהתמלול ומהשקפים של ההקלטה. ייתכן ששמות ומונחים שובשו בזיהוי הדיבור (ASR). למשל, "Devon" בתמלול הוא כמעט בוודאות Devin של Cognition. הציטוטים מועתקים מילה במילה ונבדקו מול חותמת הזמן שלהם. לא נוסף מידע ממקורות חיצוניים. בהקלטה נקלט שקף אחד בלבד (כתובות האתר וה-X של הדובר), ולכן כמעט כל התוכן כאן מבוסס על האודיו.

1. תקציר מנהלים

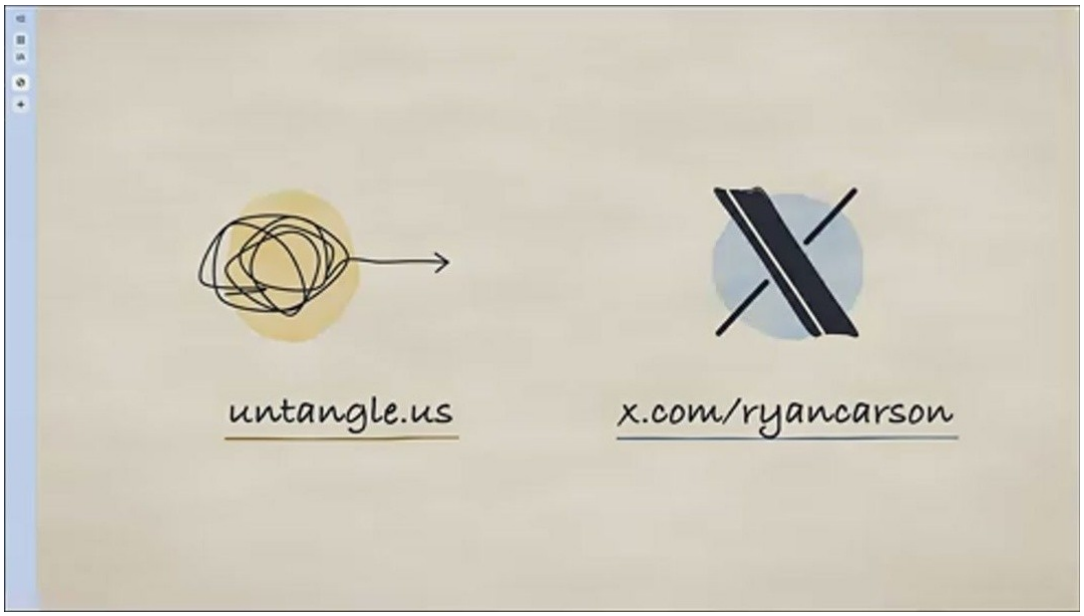
Carson מציג תהליך עבודה קונקרטי שהוא עצמו מגדיר "לא rocket science", אבל הוא רץ אצלם ב-production ועבר לדבריו מאות איטרציות. רוב הרעיונות פשוטים. החוכמה היא בפרטים: מה עושים בצורה דטרמיניסטית, איפה שמים תקרה ואיפה האדם לא זז מהלולאה.

- **ה-manager agent לא כותב קוד:** זה session רגיל ב-Devin שנקרא engineer manager. Carson מדבר בעיקר איתו, ובעיקר בקול, והוא מנהל את ה-builders.
- **עד כ-20 builders לכל manager:** הטווח הוא בין 5 ל-25 בערך. צוואר הבקבוק הוא היכולת של האדם להבין מה קורה, ולא המודל.
- **מעקב דטרמיניסטי ולא בתקווה:** automation מכריחה את ה-manager לקרוא את כל ה-sessions של הילדים כל 15 דקות, כי "It's not deterministic."
- **תקרה של שלוש לולאות review:** אחרי שלוש לולאות, באגים חשובים שנשארו עולים ל-manager, ואם צריך, ממנו לאדם.
- **חובה לצפות בסרטון לפני ship:** כל builder מקליט את הפיצ'ר עובר את הבדיקות, ואדם חייב לצפות ולאשר. זה המקום שבו מתחשק לקצר דרך, ואסור.
- **לא לפתח מקומית:** מי שמפתח ב-local dev "you are screwing up", כי המבנה הזה פשוט לא אפשרי בלי cloud agents.
- **sessions צריכים לדבר זה עם זה:** ובכל זאת צריך merge queue manager שיתאם, למשל, את המספור של database migrations.
- **גיוס בלי שיחות:** סרטון מסך של כ-45 דקות, agent שמדרג את המועמדים, חמשת הראשונים עושים פרויקט ניסיון בתשלום של 16 שעות, ושיחה מתקיימת רק עם הפיינליסטים.

מה לא נאמר

- **אין מדדים:** אין מספרים על תפוקת PRs, שיעור באגים, זמן ל-merge או עלות חודשית של צי ה-agents. הטענה "we ship a lot of PRs at high quality" לא מגובה בנתונים.
- **ה-demo לא הוצג:** בתחילת ההרצאה Carson הבטיח live demo, ובסוף סיים שלוש דקות מוקדם בלי להציג אותו.
- **ה-QA agent וה-merge queue manager נשארו ברמת הכותרת:** לא נאמר מה ה-QA בודק, איזה prompt מפעיל את ה-manager או איך בדיוק מתאמים migrations.
- **פרטי הגיוס חסרים:** לא נמסר הסכום הקבוע לפרויקט הניסיון, כמה מועמדים הגישו, באיזה מודל או prompt משתמש ה-agent המדרג, ואיך מטפלים בהטיות של המודל עצמו.
- **אי-התאמה קטנה:** לדבריו השקף אומר 10 דקות, אבל בפועל המעקב רץ כל 15 דקות.

2. הרקע: Untangle, ולמה ניהול agents הוא ניהול אנשים



השקף היחיד שנקלט בהקלטה: לוגו Untangle עם הכתובת untangle.us והכתובת x.com/ryancarson. Ryan Carson על הבמה של AI Engineer New York

Carson פותח בסקר קהל על גירושין. כולם הרימו יד כשנשאלו אם הם מכירים מישהו שהתגרש. כך הוא מסביר למה בחר בבעיה: שתיים מאחיותיו התגרשו, וגם הוריו. Untangle התחילה כאפליקציה לצרכנים ועברה ל-B2B מול משרדי עורכי דין לדיני משפחה. לדבריו יש product-market fit והחברה גדלה מהר.

החיבור לנושא ההרצאה הוא הניסיון שלו: בחברה הקודמת הוא ניהל כ-110 עובדים, ולדבריו מה שלמד שם תקף כמעט אחד לאחד לניהול agents. ההרצאה מחולקת לשניים: איך הם מנהלים agents, ואיך הם מגייסים.

> — **Ryan Carson** And it turns out managing people is like managing agents.

3. ה-org chart: manager agent, ticket board ו-20 builders

המבנה פשוט. בראש נמצאים בני אדם: הם קובעים סדרי עדיפויות, מדברים עם לקוחות ועושים את "כל הדברים האנושיים". מתחתם יש manager agents, שהם בסך הכול sessions ב-Devin. בתמלול השם מופיע בדרך כלל כ-"Devon", וכנראה מדובר ב-Devin של Cognition. כל הפיתוח רץ בענן. Carson מספר שעבר מכתיבת קוד ידנית, דרך Cursor, ועד Devin.

ה-manager עובד מתוך ticket board ב-Linear, שהוא ה-source of truth למה שצריך לבנות. הכלל הוא שכל ticket הוא PR אחד, וה-manager בוחר את ה-ticket הבא לפי dependencies. Carson מדבר עם ה-manager בעיקר בקול, כי לדבריו הוא קולט אודיו טוב יותר מטקסט.

הנקודה המעניינת היא ההיקף: כל manager מנהל במקביל עד כ-20 builders, ולפעמים בין 5 ל-25. אף אדם לא נכנס ל-sessions של ה-builders. כל התקשורת עוברת דרך ה-manager, שבעיקר עונה על שאלות טכניות ופותר חסימות קטנות. בנוסף יש automation שמכריחה את ה-manager לקרוא את כל ה-sessions של הילדים כל 15 דקות, "Push things along. Keep things moving", ולהחזיר אליו את השאלות שנפתחו.

> — **Ryan Carson** I find we can concurrently manage about 25 builders. Now the key here is that the manager is talking to these builders. You never go into these sessions.

התובנה ההנדסית הדבר החשוב כאן הוא ההבחנה בין מה שה-agent מחליט לבין מה שהמערכת מכריחה אותו לעשות. ה-manager חכם, אבל אי אפשר לסמוך עליו שייזכור לבדוק את כל ה-builders. לכן הבדיקה התקופתית היא סקריפט, ובלשונו של Carson: "You have to build an automation. You can't trust your manager agent to do this, right?". גם הגבול העליון של 20-25 builders לא נובע מהמודל אלא מרוחב הקשב של האדם: "if you push too hard on this, you quickly spin out of control".

4. Review, QA, והסרטון שחייבים לראות לפני ship

לכל builder יש review agent שהוא קורא לו. Carson מציין ש-Devin, וכנראה כל harness, כולל reviewers מובנים. הכלל אצלם הוא עד שלוש לולאות review. אם אחרי שלוש לולאות נשארו באגים חשובים, הם עולים ל-manager, ואם צריך, ממנו ל-Carson.

מעבר לקוד ולבדיקות הרגילות הם מריצים תהליך QA נוסף: ה-builder מרים session-ילד שמריץ QA. את התהליך הקים מהנדס בשם Dale שהצטרף לאחרונה. העיקרון הכלכלי: agents זולים, אז כדאי להרים כמה שיותר, כל עוד ה-merge queue לא נתקע. מנגד הוא מזהיר מהסכנה ההפוכה: "There's this fine balance of just pecking your PRs to death versus just getting them merged".

הצעד האנושי היחיד שהוא מסרב לוותר עליו הוא הסרטון. ב-Devin יש פיצ'ר מובנה בשם Test, שבונה test cases, מקליט סרטון של הפיצ'ר בשימוש ומחזיר ירוק או אדום. כשהסרטון מוכן, ה-manager מודיע ל-Carson. הוא קורא סיכום של מה ה-PR אמור לעשות, צופה בסרטון ורק אז מאשר ship. לדבריו, מי שעובד ב-harness בלי פיצ'ר כזה צריך לבנות אותו בעצמו, "It's hard".

> **Ryan Carson** — And I would encourage you, this is the part where you want to cut corners. You got to watch the video. Like there is no, this is the part of the bottleneck you have to engage in as a product owner.

הדוגמה שלו מאותו בוקר: לפני ההרצאה הוא דיבר עם ה-manager מהטלפון, קיבל הודעה שסרטון מוכן, ולא הצליח לראות אותו טוב על המסך הקטן. "I was so tempted to be like, it's okay, ship it". הוא התאפק, חזר למחשב וצפה. הוא גם מסייג את כל המתודולוגיה: "Now, this will change tomorrow and probably will be useless a week from now, but it works well today".

5. בקנה מידה של צוות: ענן, sessions שמדברים, ו-merge queue manager

הנחת הבסיס של כל המבנה היא עבודה עם cloud agent. Carson מנסח את זה בחריפות: מי שמפתח מקומית מעכב את עצמו ואת הצוות, "because you can't do this".

בצוות של עשרה אנשים, לכל אחד יש session של engineering manager משלו, ולפעמים שניים. ל-Carson עצמו רצים כרגע שניים. ה-sessions מכירים זה את זה ומעבירים הודעות זה לזה, ולדבריו זה עובד מצוין: "it's not chit chat or stupid stuff", אלא מידע שהם צריכים כדי לשחרר חסימות. הוא מציג את זה כתנאי הכרחי: מי שלא מאפשר ל-sessions לדבר זה עם זה "probably shipping worse software".

בכל זאת, גל ה-PRs מחייב תיאום מרכזי. בשביל זה יש merge queue manager, גם הוא session רגיל ב-Devin עם שם שמסביר את התפקיד שלו. הדוגמה שהוא נותן היא מספור database migrations: אם כל ה-agents מנסים לתפוס מספר בעצמם, הם מתנגשים כל הזמן. מתחת לכל זה רץ GitHub merge queue רגיל. Carson מקשר את זה להרצאה של מישהו מ-OpenAI על "un-hobble" של המודל.

> **Ryan Carson** — So set up the roles and then un-hobble the model. And I think just having a merge manager say, hey, make all this better really works well.

6. גיוס: מודדים הפעלה של agent ולא כימיה

החלק השני של ההרצאה קובע ש"Everything you used to do about hiring employees is over". אצלם אין pre-screen ואין שיחות וידאו ברוב התהליך, כי שיחה מכניסה הטיה: "You'll like their hobbies. You'll like the

השאלה ".way they talk to you. You'll think they're interesting. And it's all noise. None of it is signal היחידה שמעניינת אותם היא אם המועמד יודע להפעיל agent.

שלב	מה קורה	פרטים מההרצאה
1. סרטון כניסה (ללא תשלום)	הקלטת מסך מלאה של כ-45 דקות שבה המועמד מעלה פיצ'ר, בדרך כלל PR, באפליקציה שהוא כבר מתחזק	בלי הנחיות ובלי הכוונה, כדי לראות "how good of an agent operator is this person"
2. דירוג אוטומטי	ranking agent צופה בסרטונים ובתמלולים ומדרג את המועמדים	מועמדים שממלמלים או לא חושבים בקול קשים להערכה
3. פרויקט ניסיון בתשלום	חמשת הראשונים מקבלים סכום קבוע על 16 שעות, seat ב-Devin, NDA ומשימה: להביא PR אמיתי למצב שמוכן ל-merge	הכול מוקלט ב-Loom, והסרטונים עוברים ל-agent שממליץ את מי לבחור
4. שיחה עם הפיינליסטים	רק בשלב הזה מתקשרים: "we think we'd like to hire you"	קריאה אנושית אחרונה ומעט חיבור אישי

הוא מספר בכנות על הפעם הראשונה: הוא לא סמך על הדירוג של ה-agent, ובדיעבד ה-agent צדק. בסבב האחרון הוא צפה בעצמו בכל 16 השעות של אחד המועמדים, "It was brutal", אבל זה נתן לו תמונה טובה של האנשים. הסיכום שלו: "I've hired hundreds and hundreds and hundreds of people in my career. This is just way easier and way better".

> **Ryan Carson** — Now, what's interesting is I did this and I didn't trust it the first time. I didn't like what it said. And what's ironic is it actually was correct. Like the person that was Number one, I should have hired them. But I didn't trust it because I was biased.

7. מה עושים עכשיו

- **להעביר את הפיתוח לענן:** בלי cloud agents אין sessions מקבילים ואין manager שמנהל אותם. זה התנאי לכל השאר.
- **להגדיר תפקידים כ-sessions עם שם:** engineering manager שלא כותב קוד, builders ו-merge queue manager. כל אחד הוא session רגיל, והשם מבהיר מה התפקיד שלו.
- **לבנות מעקב תקופתי כסקריפט:** automation שמכריחה את ה-manager לעבור על כל ה-builders בפרק זמן קבוע. 15 דקות עובד אצלם.
- **לשים תקרה על לולאות review:** שלוש לולאות ואז escalation. זה מאזן בין איכות לבין PRs שנתקעים בתור.
- **לחייב סרטון לכל פיצ'ר UI:** ולצפות בו על מסך מחשב לפני ship, גם כשמתחשק לאשר מהטלפון.
- **לתאם משאבים משותפים במקום אחד:** מספור migrations וסדר merges עוברים דרך session אחד ייעודי.
- **לבחון גיוס מבוסס סרטון:** להחליף את ה-pre-screen בהקלטת מסך של עבודה עם agent, ולדחות שיחות אישיות לסוף התהליך.

שאלה פתוחה	למה זה חשוב
מה העלות החודשית של 20-25 builders לכל manager, כפול מספר המהנדסים?	הטענה "this is cheap" לא כומתה
אילו מדדי איכות מוכיחים שהתהליך עובד (באגים ב-production, rollbacks)?	"high quality" נאמר בלי נתונים
מה בדיוק בודק ה-QA agent, ובמה הוא שונה מה-review agent?	התהליך הוזכר רק ברמת הכותרת
איך מתמודדים עם פיצ'רים שאינם UI, שאין להם סרטון משמעותי?	הצעד האנושי היחיד בתהליך תלוי בסרטון
האם דירוג מועמדים על ידי מודל מכניס הטיות משלו, למשל לטובת מי שמדבר הרבה?	Carson עצמו ציין שמועמדים שממלמלים קשים להערכה
כמה זמן אנושי באמת נחסך כשצריך לצפות בכל סרטון?	Carson צפה בעצמו ב-16 שעות של מועמד אחד

8. מילון מונחים

מונח	משמעות
manager agent / engineer manager	session של agent שמנהל builders, עונה על שאלות שלהם ומשחרר חסימות, ולא כותב קוד בעצמו
builder agent	agent שמממש ticket אחד, כלומר PR אחד
session	מופע ריצה של agent ב-harness בענן (אצלם Devin). כל התפקידים בארגון הם sessions
review agent	agent שבודק את הקוד של builder. אצלם עד שלוש לולאות
QA agent	session-ילד של ה-builder שמריץ בדיקות איכות נוספות
merge queue	תור שמסדר את מיזוג ה-PRs לענף הראשי. אצלם GitHub merge queue
merge queue manager	session שמתאם את גל ה-PRs ומשאבים משותפים כמו מספור migrations
database migration	שינוי סכמה ממוספר. מספור כפול גורם להתנגשויות בין agents
ticket board	לוח משימות (אצלם Linear) שמשמש source of truth למה שצריך לבנות
dependencies	תלויות בין tickets, שלפיהן ה-manager בוחר מה לפתח קודם
automation	משימה מתוזמנת ודטרמיניסטית, כמו מעבר על כל ה-builders כל 15 דקות
harness	סביבת ההרצה של ה-coding agent (Devin, Cursor וכדומה)
un-hobble	להסיר מגבלות מלאכותיות מהמודל ולתת לו לעבוד במלוא היכולת
product owner	מי שאחראי על המוצר ומאשר ship. כאן הוא זה שצופה בסרטון
agent operator	מי שיודע להפעיל agents ביעילות. זו התכונה שנבדקת בגיוס
paid trial	פרויקט ניסיון בתשלום קבוע (16 שעות) עם PR אמיתי