Scrum Manager®
Uncovering better ways

Guide

OKRs and product strategy

In-depth development of the State of the art topics

Updated November 2026

Scrum Manager® · Skill Arena

About this document

This document is an in-depth development of the topics in the reference map (State of the art) for connecting product strategy with the work of teams through objectives and key results (OKRs), which is available at: Skill Arena.

Use it as reference material to keep your practice aligned with, and checked against, current professional knowledge.

It is complemented by the training and assessment platform in Skill Arena. In the OKRs and product strategy area you can take training tests to check and improve your level of knowledge and, if you wish, you can also obtain a diploma that provides curricular accreditation of professional competence and currency in this area.

State of knowledge

Knowledge about OKRs and product strategy is a settled field: its conceptual core has been consolidated for years and evolves slowly, more through the refinement of practice than through novelties. What sets a professional apart today is applying what is known well, and what is moving is the use of artificial intelligence to write and track objectives. Throughout the document, the labels (ESTABLISHED, CONSOLIDATING, EMERGING) help identify the maturity of each concept:

ESTABLISHED settled consensus; knowledge taken as required.
CONSOLIDATING gaining adoption fast; not yet universal but already relevant.
EMERGING recent frontier; high relevance and high volatility.

Contents

Chapters of the guide, in the order of the State of the art.

Block A · The paradigm: outcome over output

1. Outcome versus output: the cornerstone
2. The OKR as a tool for team empowerment
3. The feature factory and the due date epidemic

Block B · Anatomy of the OKR: objectives, key results, and metrics

4. The objective: qualitative, inspiring, and number-free
5. The key result: measurable, with a baseline and a formula
6. Leading versus lagging metrics
7. Types of OKR: committed, aspirational, and learning

Block C · From strategy to OKR: alignment, roadmap, and prioritization

8. Product strategy as the source of OKRs
9. Bidirectional alignment, not a cascade
10. The OKR as a roadmap prioritization filter
11. Not everything should be an OKR: the two-bucket technique

Block D · Operation: check-ins, confidence, and learning

12. The check-in: a decision session, not a reporting session
13. The confidence system: green, amber, red, and the assumptions
14. Validation, learning, and closing the cycle
15. AI in writing and tracking OKRs

BLOCK A · THE PARADIGM: OUTCOME OVER OUTPUT

Chapter 1. Outcome versus output: the cornerstone · ESTABLISHED

An OKR measures the change that the work brings about in the world, the outcome, and not the work delivered, the output.

Introduction

The distinction between output and outcome is the cornerstone of OKRs applied to product. Without it, everything else (the wording, the metrics, the rituals) turns into an exercise in managing tasks with percentages.

An output is what the team delivers: a launched feature, a deployment, a completed project. An outcome is the change that output brings about, such as a different user behavior, a business metric that moves, or a problem that stops being one. In this chapter we will see why the OKR measures the latter, how to recognize it, and why it is so hard to sustain.

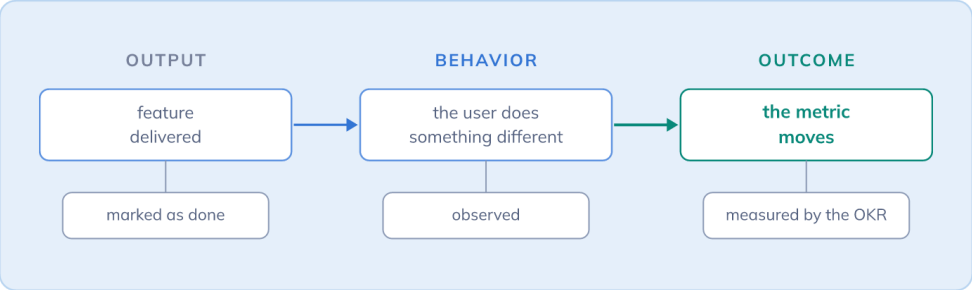

Figure 1. *From output to outcome: what the team delivers only counts if it changes someone's behavior, and the OKR measures that change.*

1.1 The difference and why it matters

Defining success as “launch feature X” creates a perverse incentive, because the team optimizes for delivering and stops asking whether what it delivered creates value. It can happen, and it often does, that the feature is deployed, the team celebrates, and no user changes their behavior and no business metric moves. The work was done and the value did not appear.

Central idea. Output answers what has been done; outcome, what has changed because it was done. A well-worded key result always describes the latter.

1.2 The observable behavior test

A practical way to tell whether something is an outcome is to check whether it describes an observable change in a user's behavior or in a business metric. “Publish the new sign-up screen” describes work. “Increase the share of completed sign-ups from 60% to 75%” describes a behavior that changes and that can be measured.

The outcome lives outside the team, in the user or in the business, while the output lives inside, in what the team produces. The Scrum Manager BoK page on OKRs adds a complementary test, the multiple paths test: a key result measures an outcome when several different solutions could achieve it, and it probably measures a task when it can only be achieved in one way.

1.3 Why it is so hard

Measuring outputs is easy, because a feature is either delivered or it is not. Measuring outcomes requires instrumenting the product, knowing the starting point, and accepting that sometimes the desired change does not come. That difficulty explains why so many organizations, even those that believe they have mastered the framework, end up sliding toward project lists disguised as OKRs.

In October 2026 Marty Cagan proposed a deeper reading of the problem. In his view, the focus on output is a symptom, and its cause is the desire for predictability: committing to dates and deliverables gives a sense of control that outcomes cannot offer, because many, if not most, product ideas will not work. Framing the choice as predictability versus outcomes brings us closer, in his opinion, to the underlying decision that every company has to make.

Artificial intelligence has sharpened the tension. By making building cheaper, it multiplies the available output without multiplying outcomes, and Cagan observes that the problem of abundant output with scarce outcomes has never been so visible. Recognizing and resisting that slide is the first competence of the topic.

What you should be able to do

Having mastered this chapter, a professional is able to:

- Define output and outcome and explain why an OKR measures the latter.
- Classify a statement as output or outcome by applying the observable behavior test.
- Apply the multiple paths test to decide whether a key result measures an outcome or a task.
- Explain the perverse incentive created by measuring success by deliveries instead of by impact.
- Relate the focus on output to the desire for predictability and recognize how the abundance of output brought by AI aggravates it.
- Reword an objective expressed as a delivery of work in terms of the change sought in the user or in the business.

Common mistakes and antipatterns

- **Confusing activity with results.** Counting it a success to have launched something without checking whether it produced the desired change is the source of almost every other mistake in the topic.

- **Calling a disguised output an outcome.** “Implement the redesign” is still work even if an achievement verb is added; the outcome describes the effect.
- **Assuming impact instead of measuring it.** Taking for granted that a feature improved the experience without instrumenting or observing the user's real behavior.
- **Celebrating building speed.** Producing faster, with or without AI, does not bring the outcome closer if nobody checks what changes with what is produced.

Further reading

· Marty Cagan (SVPG) — Predictability vs Outcomes

· Scrum.org — Transforming an output-oriented mindset to an outcome-oriented mindset using OKRs

· Felipe Castro — An OKR should measure the outcome, not the work

· Scrum Manager BoK — Outcome over output (in Spanish)

BLOCK A · THE PARADIGM: OUTCOME OVER OUTPUT

Chapter 2. The OKR as a tool for team empowerment ·

ESTABLISHED

The ultimate purpose of the OKR is to give the team a problem to solve and the context to solve it, rather than to control its activity.

Introduction

It is easy to read OKRs as a control mechanism, in which management sets targets and measures compliance. That reading dooms them to failure. In the product management tradition that Marty Cagan represents, their most important benefit is the opposite: empowering the team.

In this chapter we will see what empowering with OKRs means, why the technique only pays off with a certain team model, and why empowering does not mean washing your hands of the team either.

2.1 A problem to solve, not a list of features

Empowering a team with OKRs means two things: giving it a problem to solve instead of a list of features to build, and making sure it has the strategic context to understand the why and make good decisions. The objective describes the problem and the key results define how we will know it has been solved. What to build is left to the team, which is the one closest to the user and to the solution.

Why it matters. If the team receives the solution already decided, OKRs only serve to measure whether it executed it. The value appears when it has real room to discover the best solution to the problem it has been entrusted with.

2.2 The cultural mismatch of feature teams

Cagan distinguishes feature teams, which receive the list of what they must build, from empowered product teams, which are assigned problems to solve. In the overview of his series on team objectives (2020) he warns that, in a company that still works with feature teams, the OKR technique is a cultural mismatch and almost certainly a waste of time and effort.

His explanation is historical: the technique was born in companies that had empowered product teams in their DNA. And it is consistent with the previous chapter, because if success is already defined as delivering a specific feature, there is no outcome left to pursue and no decision left to delegate. OKRs only pay off where the team model leaves them room.

2.3 Empowering with direction

Empowering the team does not mean leaving it alone and hoping for the best. There is still active leadership work to do: providing strategic context, making sure the team has the necessary skills, and supporting the tracking. Autonomy is granted over the how, within a clear strategic framework, and it is a different way of leading.

Alex Ballarín describes on the Scrum.org blog (2024) what changes in a team that works with Scrum when it adopts OKRs:

- **The Product Owner gains context.** They receive the business problems to solve and the strategy that motivates them, instead of isolated feature requests.
- **The whole team takes part.** Defining the objectives and key results, and planning the work to achieve them, stop being the task of a single person.
- **A criterion for questioning appears.** The team can distinguish strategic requests from those that are not, without rejecting them automatically.

What you should be able to do

Having mastered this chapter, a professional is able to:

- Explain the empowerment purpose of OKRs: a problem to solve and the strategic context, as opposed to a list of features.
- Distinguish an empowered product team from a feature team and reason why OKRs only pay off in the former.
- Identify when an organization is not culturally ready for OKRs to add value.
- Differentiate empowering (autonomy over the how, with context) from abandoning (absence of direction).

- Describe what changes for the Product Owner and for the rest of the team when they work with OKRs.

Common mistakes and antipatterns

- **Using OKRs as a control panel.** Turning them into an instrument for monitoring compliance empties them of their empowerment purpose and reproduces the command model.
- **Assigning the solution and measuring its delivery.** If the team receives the feature already decided, the OKR only times the execution.
- **Introducing OKRs in a culture of feature teams.** Without teams that have room to decide the how, the technique consumes effort without returning value.
- **Confusing autonomy with absence of direction.** A team without strategic context or support sets OKRs that connect with nothing.

Further reading

· Marty Cagan (SVPG) — Team Objectives: Overview

· Scrum.org — How do OKRs empower Scrum Teams? (in Spanish)

· Scrum Manager BoK — OKR (in Spanish)

BLOCK A · THE PARADIGM: OUTCOME OVER OUTPUT

Chapter 3. The feature factory and the due date epidemic · ESTABLISHED

The dominant antipattern consists of using the OKR format to list projects and dates instead of measurable results.

Introduction

The most widespread mistake in the practice of OKRs is applying them as a list of projects with dates. The document looks like a set of OKRs, with numbered objectives and key results, but on closer inspection it does not contain a single number that measures a change in the user or in the business.

In this chapter we will see where the expression feature factory comes from, how the antipattern shows up inside OKRs, and how to convert an unhealthy OKR. Knowing how to detect it is the maturity test of the topic.

3.1 The feature factory

In 2016 John Cutler popularized the expression feature factory in an article with twelve signs for recognizing it. He himself says he took it from a developer friend who complained that he spent his days in the factory, producing features and sending them down the line. The first sign on his list is the absence of measurement: nobody checks whether what was delivered worked.

An organization like this produces nonstop, celebrates every delivery, and rarely asks whether anything changed for the user. Poorly applied OKRs reinforce that dynamic instead of fighting it, because the format gives the appearance of management by results to what is still management by activity.

Warning sign. If a key result can be marked as done on a specific date (“implement X in June”), it is a task. A key result is measured on a scale that moves, and not with a done-or-not-done switch.

3.2 The due date epidemic

The most common symptom inside OKRs is the key result that is really a project with a date, such as “implement the new process in the CRM by June.” Felipe Castro calls it the due date epidemic (2024) and warns that it is so widespread that it reaches even those who believe they have mastered the framework.

Castro tells the case of the CEO of a fast-growing company who considered himself advanced in OKRs. In a first review, 8 of the company's 19 key results were projects with dates, and in its two product lines 5 of 13 and 8 of 15 were. The OKR format did not protect against the error: it disguised it.

3.3 How to convert an unhealthy OKR

The cure starts with a question: what change is this delivery meant to produce. “Implement the new sign-up process” hides an outcome, perhaps “reduce drop-off during sign-up.” Rewording the key result in terms of that change, with its metric and its starting point, restores the OKR to its function.

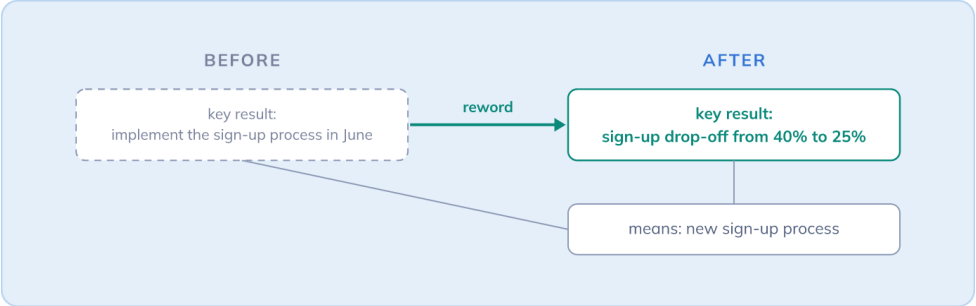

Figure 2. *An unhealthy OKR and its conversion: the project with a date becomes the means, and the key result measures the change that project was meant to bring about.*

The project still exists, but it becomes the means and is no longer what is measured. And when there is no way to measure the change, perhaps that work should not have been in the OKRs: chapter 11 presents the criterion for deciding.

What you should be able to do

Having mastered this chapter, a professional is able to:

- Recognize a set of OKRs that is really a list of projects with dates.
- Apply the done-or-not-done warning sign to distinguish a key result from a task.
- Explain what the feature factory is, who popularized the expression, and how poorly applied OKRs reinforce it.
- Convert a key result expressed as a project with a date into an outcome key result, with its metric and its starting point.

Common mistakes and antipatterns

- **Confusing format with content.** Having numbered objectives and key results does not guarantee that outcomes are being measured; the format can disguise a task list.
- **Key results with a delivery date.** “Launch X by June” is a project commitment and belongs somewhere else (chapter 11).
- **Believing you have already mastered the framework.** The due date epidemic mainly affects those who take the topic for granted and do not critically review their own OKRs.

Further reading

· John Cutler — 12 signs you're working in a feature factory

· Felipe Castro — An OKR should measure the outcome, not the work

· Scrum Manager — 5 key aspects for using OKRs effectively (in Spanish)

· Scrum Manager BoK — Feature factory (in Spanish)

BLOCK B · ANATOMY OF THE OKR: OBJECTIVES, KEY RESULTS, AND METRICS

Chapter 4. The objective: qualitative, inspiring, and number-free · ESTABLISHED

The objective gives direction and meaning: it is qualitative, memorable, and contains no figures.

Introduction

The half of the framework that is most neglected is the first letter, the objective. In the rush to have metrics, many teams jump straight to the key results and leave the objective as an empty headline. Yet the objective is what gives meaning to the rest, because it answers what is to be achieved and why it matters.

In this chapter we will see what a good objective looks like, what qualities it must have, and what happens when it is missing.

Figure 3. *Anatomy of an OKR: the objective gives the direction without figures, and each key result measures it from a baseline to a target.*

4.1 Qualitative and number-free

The objective describes the aspirational direction. It must be qualitative, inspiring, and understandable to anyone on the team, and it contains no figures, because its function is to give meaning and direction, while measurement belongs to the key results. An objective with numbers in it is usually a misplaced key result.

The Scrum Manager BoK page on OKRs adds one more quality: the objective is not prescriptive; it describes the destination and leaves the path open. “Become the benchmark for agile training in the Spanish-speaking market” is an objective, while “launch three new courses” describes activities and not the impact sought.

Rule of thumb. If an objective contains a percentage or a quantity, it is probably a key result written in its place. The objective says where to; the key result, how much.

4.2 Memorable and understandable

A good objective can be remembered without looking it up and is understood by anyone on the team, technical or not. That readability has practical consequences: if the team cannot repeat from memory where it is heading, it will hardly align its daily decisions with it.

Inspiration is not an ornament either. The objective must give a reason to make an effort, and a statement that only gives an instruction loses half of its function.

4.3 The soulless objective and its consequences

An empty objective, such as “improve the product” or “grow,” produces key results whose purpose nobody understands, because there is no why to justify them. The objective is the context that makes the numbers mean something.

That is why it is worth investing time in writing it well before moving on to the key results, instead of treating it as a preliminary formality. And it is worth knowing where it comes from: a good objective translates a decision of the product strategy, as we will see in chapter 8.

What you should be able to do

Having mastered this chapter, a professional is able to:

- Write a qualitative, inspiring objective without figures.
- Detect and correct an objective that actually hides a key result because it contains numbers.
- Distinguish an objective that describes the destination from one that prescribes the path.
- Explain the function of the objective, giving direction and meaning, as opposed to measuring, which belongs to the key results.
- Assess whether an objective is memorable and understandable to any team member.

Common mistakes and antipatterns

- **Putting numbers in the objective.** Figures belong in the key results; a quantified objective loses its function of giving direction and meaning.
- **Treating the objective as a formality.** Jumping to the how much without a clear why produces key results orphaned of purpose.
- **Generic, soulless objectives.** “Improve the product” directs nothing; a good objective points to a specific change worth the effort.

- **Prescribing the path.** An objective that lists what has to be built leaves the team no room to find the best solution.

Further reading

· Google re:Work — Set goals with OKRs

· Scrum Manager BoK — OKR (in Spanish)

· Scrum Manager BoK — Impact map (in Spanish)

BLOCK B · ANATOMY OF THE OKR: OBJECTIVES, KEY RESULTS, AND METRICS

Chapter 5. The key result: measurable, with a baseline and a formula · ESTABLISHED

A key result measures an observable change, and to mean anything it requires knowing the starting point and defining how it is calculated.

Introduction

If the objective gives direction, the key results make it measurable. A key result measures an observable change in users or in the business, and for it to mean anything it needs two things that are often forgotten: a starting point, the baseline, and a calculation formula.

In this chapter we will see the craft of a well-formed key result, what to do when there is no starting data, and how many key results it is wise to set.

5.1 The need for a baseline

A key result such as “reach 80%” means nothing if you do not know where you start, because starting from 79% is not the same as starting from 30%. The effort, the ambition, and the very feasibility depend on the starting point, so the baseline is established before committing to the figure.

That is why the Scrum Manager BoK recommends the format “from X to Y by date Z,” which expresses a measurable movement between two points in time: “increase the course completion rate from 42% to 60% before June 30.”

No baseline, no key result. Setting a target without knowing where you start leads to discovering halfway through the quarter that progress cannot be measured. If there is no data, the first problem to solve is to start measuring.

5.2 When there is no data: guess and learn

In January 2026 Christina Wodtke answered a question that comes up in almost every team: how to set a key result when there is no baseline. Her answer was as simple as it was uncomfortable: you guess. You can look for figures from competitors or from similar products, but in the end you have to set an informed number, launch, measure, and learn.

Her argument is that guessing works as a learning mechanism. When a team predicts how much a number will move and then checks whether it was right, it corrects its model of how the product works, and over the cycles experienced teams learn to anticipate what will move the metrics. Waiting for the perfect data, by contrast, delays the learning that would produce it.

5.3 The calculation formula

Defining the formula means deciding exactly how it will be measured: what is counted, in which population, how often, and with which data source. Two people who measure "customer satisfaction" without an agreed formula will get different numbers and will argue about progress instead of about decisions.

The formula removes that ambiguity before it appears, and it also makes clear what instrumentation the product needs in order to measure.

5.4 How many and how ambitious

A few key results per objective are enough. Google's OKR guide on re:Work recommends around three, and the Scrum Manager BoK page puts the range between two and four: more than four usually indicates that the objective is too broad or that tasks are slipping in.

Each key result must be measurable and ambitious at the same time, demanding enough to push the team and realistic enough to be credible. The balance between ambition and realism depends on the type of OKR, as we will see in chapter 7.

What you should be able to do

Having mastered this chapter, a professional is able to:

- Word a key result that measures an observable change in users or in the business, with the format "from X to Y by date Z."
- Establish the baseline of a key result and explain why a target without a starting point cannot be interpreted.
- Set an estimated baseline when there is no data and plan how it will be measured and corrected in the next cycle.
- Define the calculation formula of a key result: what is measured, in which population, with which source, and how often.
- Limit the number of key results per objective and reason about the balance between ambition and realism.

Common mistakes and antipatterns

- **Setting the target without a baseline.** "Reach 80%" cannot be interpreted without the starting point; progress cannot be assessed and the ambition is arbitrary.
- **Getting stuck waiting for the perfect data.** Delaying key results until there is a history prevents the learning that would generate it; it is better to guess and measure from day one.
- **Not agreeing on the formula.** Without a precise definition of how it is measured, the team argues about the numbers instead of about the decisions.
- **Too many key results.** A long list dilutes focus and usually hides tasks; a few well-measured ones pay off more than many vague ones.

Further reading

· Google re:Work — Set goals with OKRs

· Christina Wodtke — In praise of guessing

· Scrum Manager BoK — OKR (in Spanish)

BLOCK B · ANATOMY OF THE OKR: OBJECTIVES, KEY RESULTS, AND METRICS

Chapter 6. Leading versus lagging metrics · CONSOLIDATING

Lagging metrics confirm the result once it has happened; leading metrics anticipate it and allow adjusting in time.

Introduction

A key result can be measured with two types of indicator, and choosing well between them is where the product community is still fine-tuning. Lagging metrics confirm the result once it has happened; leading metrics predict it and can move within the quarter.

In this chapter we will see the difference between the two, how the metrics tree helps relate them, and why a good set of key results combines both.

6.1 Lagging: the confirmed result

Lagging metrics measure the final effect sought, such as revenue, ninety-day retention, or customers' willingness to recommend the product. They are the ones that really matter to the business, but they have an operational problem: they change slowly and are known late.

If a team is guided by them alone, it reaches the end of the quarter without having been able to correct course, because the signal that something was going wrong appeared too late to act.

6.2 Leading: the signal that allows action

Leading metrics measure behaviors that anticipate the final result and that the team can influence within the cycle: activations, the use of a specific feature, return frequency. They make it possible to measure progress before it is too late to adjust.

Tim Herbig proposes finding them by walking the customer journey backward from the desired result and identifying which behaviors predict it.

The relationship between the two. The same metric can be leading at one level and lagging at another. Quarterly upgrade revenue is a leading indicator for the company and, at the same time, the final result for the team that pursues it. The level determines the role.

6.3 The metrics tree and the North Star Metric

One way to organize the relationship between metrics is the metrics tree. At the center is a metric that captures the value the product delivers to its customers, the North Star Metric; above it, the business results that follow from that value, such as revenue or retention; and below it, the input metrics that drive it and that teams can move.

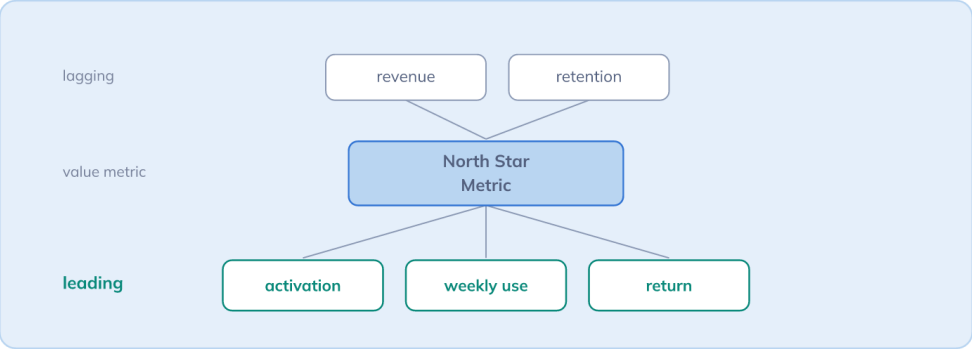

Figure 4. *A metrics tree: the input metrics, which the team moves within the quarter, anticipate the value metric and the business results that come later.*

Amplitude's *North Star Playbook*, co-authored by John Cutler, has spread this breakdown: the North Star Metric guides the organization as a whole, and each team works on the input metrics it can influence. Those input metrics are, by construction, good candidates for leading indicators in the teams' key results.

The tree is a hypothesis about how the metrics relate, and like every hypothesis it is tested against data. An input metric moving does not guarantee that the value metric will follow, and the team that takes it for granted ends up optimizing a signal disconnected from impact.

6.4 The balance, which is not yet settled

Getting the set right is more art than recipe. With too many lagging metrics the OKR is not actionable in time, and with too many leading ones the connection with real impact is lost, with the risk of optimizing an indicator that never ends up moving the business result.

That is why this knowledge is still consolidating: there is consensus on the distinction and on the usefulness of the tree, but not on the ideal combination, which depends on the product and the team.

What you should be able to do

Having mastered this chapter, a professional is able to:

- Distinguish a lagging metric from a leading one and give examples of each.
- Explain the operational limitation of being guided only by lagging indicators.
- Derive leading indicators by walking the customer journey backward from the desired result.

- Build a simple metrics tree, with the North Star Metric, its business results, and its input metrics, and identify which input metrics can serve as a team's key result.
- Reason why a set of key results combines both types of metric and what risks each excess carries.

Common mistakes and antipatterns

- **Measuring only with lagging indicators.** The team finds out too late that things are going badly, with no margin to adjust within the cycle.
- **Choosing an arbitrary leading metric.** A leading indicator is only useful if it truly predicts the result; it should be derived from the customer journey or the metrics tree, not picked for convenience.
- **Taking the metrics tree as a truth.** The relationships in the tree are hypotheses; if they are not checked against data, the team may move an input metric that does not pull the value metric along.
- **Optimizing the signal and losing the impact.** Maximizing a leading metric that never ends up moving the business result is a sophisticated way of measuring output.

Further reading

· Tim Herbig — Leading and lagging indicators in OKRs

· Amplitude — North Star Playbook

· Scrum Manager BoK — North Star Metric (in Spanish)

BLOCK B · ANATOMY OF THE OKR: OBJECTIVES, KEY RESULTS, AND METRICS

Chapter 7. Types of OKR: committed, aspirational, and learning · CONSOLIDATING

Not all OKRs are read the same way: committed ones are fully achieved, aspirational ones hover around 70%, and learning ones seek knowledge.

Introduction

Measuring all OKRs with the same yardstick distorts the reading of progress. Google distinguishes two types with opposite logics, committed and aspirational, and What Matters, the organization founded by John Doerr to spread OKRs, has added a third, the learning OKR.

In this chapter we will see the three types, who proposes each one, and why it is worth labeling them.

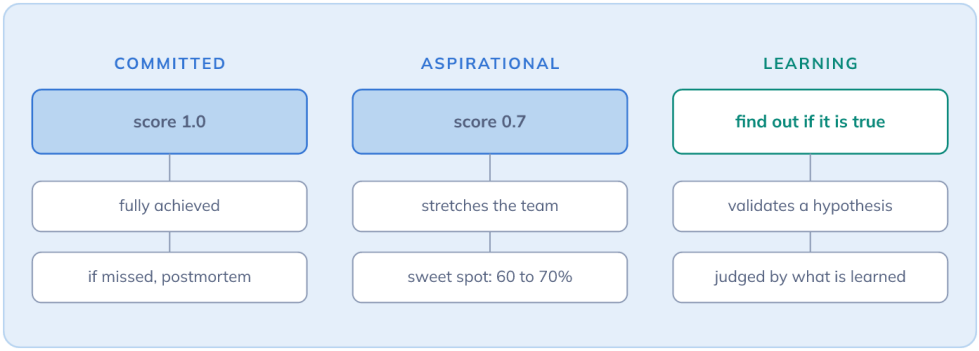

Figure 5. *Three types of OKR: each type is read with its own yardstick: the committed one is fully achieved, the aspirational one hovers around 0.7, and the learning one is judged by what it discovers.*

7.1 Committed

Google's OKR playbook, which Doerr's *Measure What Matters* reproduces and What Matters spreads, defines committed OKRs as OKRs that are agreed to be achieved, adjusting schedules and resources if necessary. Their expected score is 1.0, and falling short requires explaining the miss, because it reveals planning or execution errors.

When a committed OKR is not achieved by its date, the playbook calls for a postmortem, in order to understand what went wrong in planning or execution and improve the team's ability to deliver, without any intent to punish. With committed OKRs the point is not to stretch, but to deliver.

7.2 Aspirational

Aspirational OKRs express how one would like the world to be even if it is not clear how to get there and not all the resources are available. According to Google's playbook, their expected average score is 0.7, with high variance. Google's guide on re:Work places the sweet spot in the 60 to 70% range: scoring lower may indicate that not enough is being achieved, and scoring higher, that the aspirational goals were set too low.

Why label the type. A 70% can be excellent in an aspirational OKR and a failure in a committed one. Without knowing what type the OKR is, the same number is interpreted in two opposite ways, and Google's playbook counts failing to distinguish them among the classic OKR traps.

7.3 Learning

What Matters proposes the third type based on its work with teams venturing into something new. In those cases, learning is a valuable outcome in its own right, and the learning OKR prioritizes it over reaching a figure: validating a hypothesis, exploring a new market, understanding a behavior.

The key result shifts from “increase X” to “find out whether Y is true.” These OKRs are especially useful in early stages or in the face of high uncertainty, where committing to an impact metric would be premature.

The pair of committed and aspirational is settled. What is still consolidating is the learning type and the habit of labeling each OKR with its type, which is not universal: many implementations still measure all OKRs with the same yardstick.

What you should be able to do

Having mastered this chapter, a professional is able to:

- Distinguish the three types of OKR, committed, aspirational, and learning, by their logic and their expected level of achievement.
- Attribute each type to its origin: committed and aspirational to Google, learning to What Matters.
- Assign the right type to an OKR according to its nature and the degree of uncertainty.
- Interpret a score correctly according to the type of OKR.
- Explain why demanding 100% of an aspirational OKR or tolerating misses without analysis in a committed one is a mistake.

Common mistakes and antipatterns

- **Treating all OKRs the same.** Applying the same yardstick to a committed and an aspirational OKR makes progress unreadable and demotivates the team.
- **Demanding 100% of an aspirational OKR.** Penalizing a miss pushes teams to set timid targets and destroys its function of stretching the team.
- **Turning the analysis of a committed OKR into a punishment.** The postmortem serves to improve planning; if it is experienced as a trial, teams stop committing.
- **Forcing an impact metric onto learning territory.** When uncertainty is high, committing to a premature figure replaces real learning with a made-up number.

Further reading

· What Matters — Google's OKR playbook

· What Matters — What is a learning OKR?

· Google re:Work — Set goals with OKRs

BLOCK C · FROM STRATEGY TO OKR: ALIGNMENT, ROADMAP, AND PRIORITIZATION

Chapter 8. Product strategy as the source of OKRs · ESTABLISHED

OKRs translate product strategy into objectives for the teams, and without a prior strategy they remain a wish list.

Introduction

The previous chapters have dealt with how an OKR is built. A prior question remains that the framework does not answer on its own: where the objectives come from. The answer is product strategy, and it is the most frequently skipped link.

In this chapter we will see what product strategy is according to two references of the craft, Marty Cagan and Roman Pichler, how it turns into objectives for the teams, and what happens when it is missing. Scrum Manager's Agile product operating model skill covers the place of strategy in the operating model; here we focus on its relationship with OKRs.

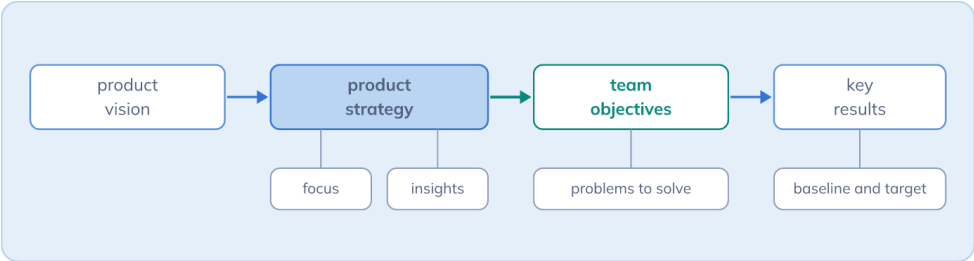

Figure 6. *From strategy to OKR: the strategy chooses the few problems that matter, the team objectives turn them into work, and the key results measure whether they are progressing.*

8.1 What product strategy is

Cagan describes product strategy in four parts:

1. **Focus.** Deciding the few problems that really matter and, therefore, everything that will not be done.
2. **Insights.** The knowledge that comes from analyzing the data and learning from customers, which can concern the dynamics of the business, one's own capabilities, new technologies, or the competition.
3. **Action.** Turning those insights into work for the product teams.
4. **Management.** Actively supporting that work, without falling into micromanagement.

Focus is the hardest part. Cagan says he has seen companies with a list of fifty major objectives or initiatives on a wall, and teams that complain they have no time for their own work because those obligations take up everything. Strategy begins by accepting that not everything is equally important.

8.2 From strategy to team objectives

The action part is where OKRs come in. Cagan explains that insights point to problems worth solving, and that the most widespread way of assigning them to teams is the OKR system: the objective is the customer or business problem to be solved, and the key results measure progress. To move to action, each product team receives its own objectives, the team objectives.

Central idea. OKRs are the operational translation of the strategy. A team objective is a problem that the strategy has decided is worth solving, and its key result measures whether the bet is progressing.

This reading connects with chapter 2. Assigning problems, instead of features, only makes sense if someone has first decided which problems matter, and that decision is the strategy.

8.3 The strategy stack

Roman Pichler orders strategic decisions in a stack (*The Strategy Stack*, revised in September 2026). The business strategy guides the product portfolio strategy, which directs the strategy of each product; this, in turn, guides the product roadmap, which steers the backlog. The technology strategy sits at the same level as the portfolio and product strategies, influences them, and is influenced by them.

The consequence for OKRs is direct. A well-chosen objective inherits its meaning from the levels above, and an OKR that cannot be traced back to a product strategy decision is disconnected: it may be well written and still serve nothing that the organization has decided to pursue.

8.4 Without strategy, a wish list

When there is no prior strategy, OKRs fill up with what each area wants to achieve or with the usual requests written in objective format. The result is easy to recognize:

- **Too many objectives.** Without focus, everything seems a priority and teams receive more than they can take on.
- **Objectives that mirror the org chart.** Each department has its own, even if none responds to a customer problem.
- **Objectives that nobody can justify.** Asked why this objective and not another, the only answer is that someone asked for it.

The product vision provides the horizon; the strategy decides the path to get closer to it with the available resources, and the OKRs measure whether that path works.

What you should be able to do

Having mastered this chapter, a professional is able to:

- Describe the four parts of product strategy according to Cagan (focus, insights, action, and management).
- Explain how product strategy translates into team objectives and what role OKRs play in it.
- Place OKRs in Pichler's strategy stack, between the product strategy, the roadmap, and the backlog.
- Trace an OKR back to the strategic decision it comes from, or recognize that it has none.
- Identify the symptoms of OKRs set without a prior strategy.

Common mistakes and antipatterns

- **Starting with the OKRs.** Writing objectives without a strategy that decides which problems matter produces wish lists in OKR format.
- **Confusing strategy with a roadmap.** A list of initiatives with dates does not say which problems have been chosen or why.
- **Trying to cover everything.** Without focus, teams receive dozens of objectives and make no real progress on any of them.
- **Distributing OKRs by department.** If the objectives mirror the org chart instead of customer problems, the strategy is left out.

Further reading

· Marty Cagan (SVPG) — Product Strategy: Overview

· Marty Cagan (SVPG) — Product Strategy: Actions

· Roman Pichler — The strategy stack

· Skill Arena — Agile product operating model

· Scrum Manager BoK — Product vision (in Spanish)

BLOCK C · FROM STRATEGY TO OKR: ALIGNMENT, ROADMAP, AND PRIORITIZATION

Chapter 9. Bidirectional alignment, not a cascade · CONSOLIDATING

Setting OKRs in parallel, with influence in both directions, instead of breaking them down in a top-down cascade.

Introduction

The intuitive alignment model is the cascade: the company OKR is broken down into OKRs for each area, and these into those of each team, level by level. It is slow and turns outcomes into outputs as soon as it moves below the executive level.

In this chapter we will see why the cascade fails, what the alternative gaining ground, bidirectional alignment, consists of, and why it is still consolidating.

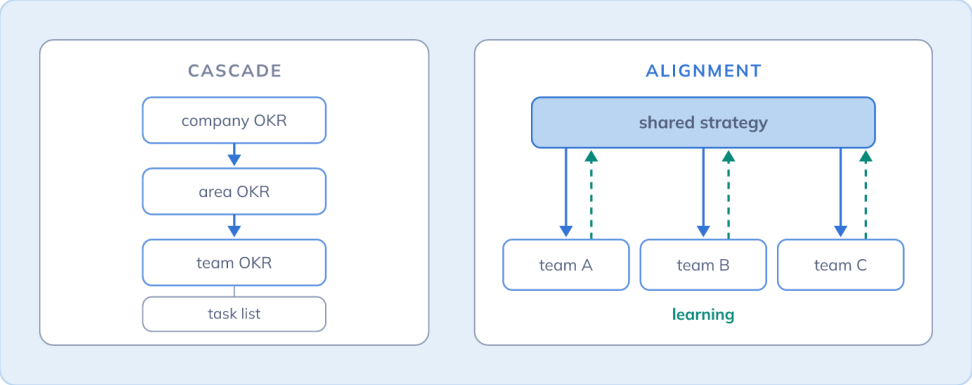

Figure 7. *Cascade versus alignment: in the cascade the objective flows down and turns into tasks, while in alignment the teams propose and what they learn flows up.*

9.1 Why the cascade fails

Yuval Yeret explains on Scrum.org (2021) that many organizations define their OKRs and then distribute them among the teams and functions they already have. In doing so, the OKR moves further and further away from the desired outcome, because an isolated function cannot deliver an outcome, only its share of output. It is the old work breakdown structure of project management, carried over to OKRs.

It is also slow. Each level waits for the one above to finalize its OKRs before defining its own, and by the time the frontline team has its OKRs, the quarter has already begun.

The symptom. If the teams' OKRs are the list of projects needed to meet the OKR above, the cascade has turned results into tasks. It is the due date epidemic of chapter 3, now by organizational design.

John Cutler (2022) adds other flaws: the cascade tends to reproduce the boundaries of the org chart, imposes a tree structure on relationships between variables that are rarely that simple, leaves out ongoing work that does not fit into the few objectives at the top,

and encourages putting on the roadmap anything that can be connected, however forcibly, to those objectives.

9.2 Bidirectional alignment

The alternative is for teams to set their OKRs in parallel from a shared strategy, with influence in both directions. Leadership provides the strategic context, the teams propose how to contribute from their knowledge of the user, and that feeds back into the strategy. It takes much less time because it is no longer sequential.

The specific proposals vary:

- **A shared model instead of the org chart.** Cutler proposes decoupling goals from the organization's structure and anchoring them in a model of the business's drivers, with its metrics. On that model the leverage points are prioritized, and each team looks for where it can have influence and with whom it must collaborate.
- **Teams organized around OKRs.** Yeret suggests forming cross-functional teams around the most complex OKRs, so that each one has that OKR as its Product Goal, and limiting to three per quarter the OKRs assigned to any one team.
- **A mix of top-down and bottom-up.** Google's guide on re:Work observes that the OKRs that work usually combine proposals from leadership and from the teams.

9.3 Why it is not yet a consensus

Parallel alignment is faster and keeps the focus on outcomes, but it demands an organizational maturity that not every company has: trust to let the teams propose, transparency of the strategy, and willingness to let the base influence the top.

Where that maturity does not exist, the cascade reappears through inertia, because it is the model that fits annual planning and the hierarchical structure. That is why bidirectional alignment is recognized but is not yet the dominant practice.

What you should be able to do

Having mastered this chapter, a professional is able to:

- Explain why cascading breakdown turns outcomes into outputs and why it is slow.
- Describe bidirectional alignment: OKRs set in parallel from a shared strategy, with influence in both directions.
- Compare the alternative proposals to the cascade (a shared model of the business's drivers, teams organized around OKRs, a combination of top-down and bottom-up proposals).
- Recognize the symptom of a cascade: team OKRs that are the list of projects of the level above.
- Identify the organizational conditions that bidirectional alignment needs in order to work.

Common mistakes and antipatterns

- **Breaking the higher OKR down into mechanical OKRs.** It replicates the work breakdown structure and degrades results into tasks as soon as it moves down a level.
- **Aligning sequentially.** Waiting for each level to finish before the next one starts uses up the quarter before it gets going.
- **Copying the org chart.** Assigning objectives by department reinforces silos and leaves the outcomes that cut across areas without an owner.
- **Adopting the parallel model without the necessary maturity.** Without strategic transparency or the trust to let the base have influence, the cascade reappears in practice even if the opposite is preached.

Further reading

· John Cutler — Why goal cascades are harmful (and what to do instead)

· Scrum.org — Organizing around outcomes with OKRs and Scrum

· Google re:Work — Set goals with OKRs

· Scrum Manager BoK — OKR (in Spanish)

BLOCK C · FROM STRATEGY TO OKR: ALIGNMENT, ROADMAP, AND PRIORITIZATION

Chapter 10. The OKR as a roadmap prioritization filter ·

ESTABLISHED

With OKRs on the table, prioritizing stops being a matter of opinion and becomes a question of measurable contribution.

Introduction

One of the most practical contributions of OKRs to product management is that they turn prioritization into a question that can be answered. Without them, prioritizing is a negotiation of opinions and of who speaks loudest; with them, the question is which initiative moves this key result the most.

In this chapter we will see the OKR as a roadmap filter, as a tool for negotiating with stakeholders, and as an argument for not committing with dates to what has not yet been tested.

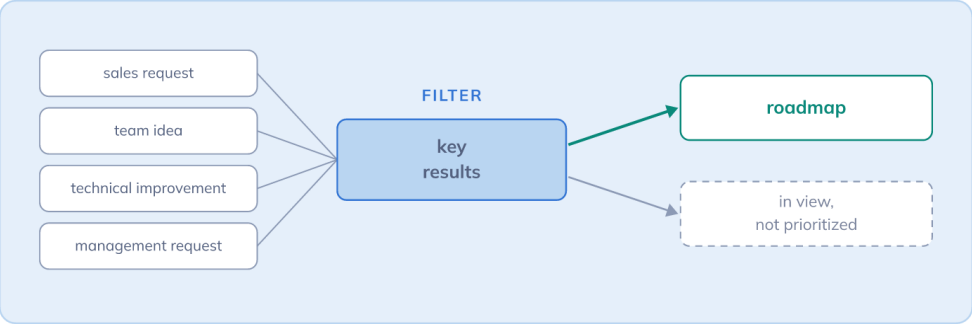

Figure 8. *The OKR as a filter: what moves a key result goes onto the roadmap, and the rest stays in view for what it is.*

10.1 From opinion to contribution

When the key results are defined, each candidate initiative is evaluated by its expected contribution to them, and the roadmap is ordered by the expected impact on the OKRs and not by who asks the loudest. Requests that contribute to no key result become visible for what they are: work that may be legitimate, but that does not advance the period's objectives and must be justified separately.

The OKR as a criterion for saying no. Its greatest value in prioritization is giving whoever prioritizes a defensible argument for turning down work. The answer stops being a personal preference and becomes a fact: this request does not move any key result of the quarter.

10.2 The OKR as a shield in negotiation

Faced with pressure from a stakeholder who wants to slip in a request, the OKR offers an answer based on data instead of on hierarchy: “if we prioritize this, key result X stops

progressing this quarter; shall we go ahead?" The conversation stops being about who is in charge and becomes about what one is willing to sacrifice.

That protects the team and makes the cost of each change of priority explicit.

10.3 Ideas that have not yet been tested

Cagan points out (October 2026) that, if you prioritize by outcomes, you have to accept that many, if not most, of the product ideas on a roadmap simply will not work. Putting untested ideas on the roadmap, with dates or at least with timeframes attached, is choosing predictability over outcomes, and Cagan notes that many product leaders contribute to it every day.

A format that avoids that trap is the horizon-based roadmap, for which the Scrum Manager BoK describes the Now-Next-Later format, usually attributed to Janna Bastow and ProdPad. It organizes priorities into what is being worked on now, what will probably come next, and future opportunities, so that detail and commitment decrease with distance and no distant initiative becomes a rigid promise.

When the roadmap initiatives are expressed as problems tied to key results, the team can test several solutions for each one. Product discovery, covered by Scrum Manager's Product discovery and customer validation skill, is the work of finding out which ones work before building them.

10.4 The connection with the product dashboard

Alex Ballarín proposes on Scrum.org (2024) bringing OKRs onto the product dashboard, next to the roadmap, the metrics, and the risks. This way prioritization is visibly connected with the strategy: you can see which initiatives contribute to which key results, what dependencies exist between teams with shared OKRs, and where there is effort that contributes to nothing.

The impact mapping technique helps trace that causal relationship between initiatives and business results.

What you should be able to do

Having mastered this chapter, a professional is able to:

- Prioritize a set of initiatives by their expected contribution to the key results.
- Use OKRs as a defensible criterion for rejecting or postponing work that contributes to no key result.
- Frame a negotiation with a stakeholder in terms of the impact on a specific key result.
- Explain why committing untested ideas to dates is choosing predictability over outcomes, and how a horizon-based roadmap avoids it.
- Explain the value of displaying OKRs on the product dashboard to connect prioritization and strategy.

Common mistakes and antipatterns

- **Prioritizing by who speaks loudest.** Without the OKR as a criterion, whoever pushes hardest wins, not whatever has the most impact.
- **Not making visible the work that does not contribute.** If requests unrelated to the OKRs are not labeled as such, they erode focus without anyone consciously deciding it.
- **Committing to dates for what has not been tested.** Turning unvalidated ideas into dated deliveries closes the door to discarding them when they do not work.
- **Accepting priority changes without making their cost explicit.** Bringing in work without naming which key result suffers turns prioritization into an invisible trickle of concessions.

Further reading

· Scrum.org — Add OKRs to your product dashboard in Scrum (in Spanish)

· Marty Cagan (SVPG) — Predictability vs Outcomes

· Scrum Manager BoK — Now-Next-Later roadmap (in Spanish)

· Scrum Manager BoK — Impact map (in Spanish)

· Skill Arena — Product discovery and customer validation

BLOCK C · FROM STRATEGY TO OKR: ALIGNMENT, ROADMAP, AND PRIORITIZATION

Chapter 11. Not everything should be an OKR: the two-bucket technique · CONSOLIDATING

Separating investments into two buckets, the outcomes managed with OKRs and the dated commitments managed as projects, avoids forcing everything into the OKR format.

Introduction

Knowing what not to put in an OKR is as much a part of mastering the topic as knowing how to write one. Part of product work consists of outcomes to pursue and another part of dated commitments that simply have to be met, and forcing both into the same format is one cause of the due date epidemic.

In this chapter we will see Felipe Castro's two-bucket technique, a practical question for applying it, and why drawing the line requires judgment.

11.1 The two buckets according to Castro

Castro (2024) proposes dividing product investments into two buckets according to a single criterion: whether their outcome can be measured. Investments whose outcome can be measured go into the outcomes bucket, which is managed with OKRs. Those that cannot be measured, at least for now, go into the due dates bucket, which is managed as a project, with what he calls the due dates approach or PMO approach.

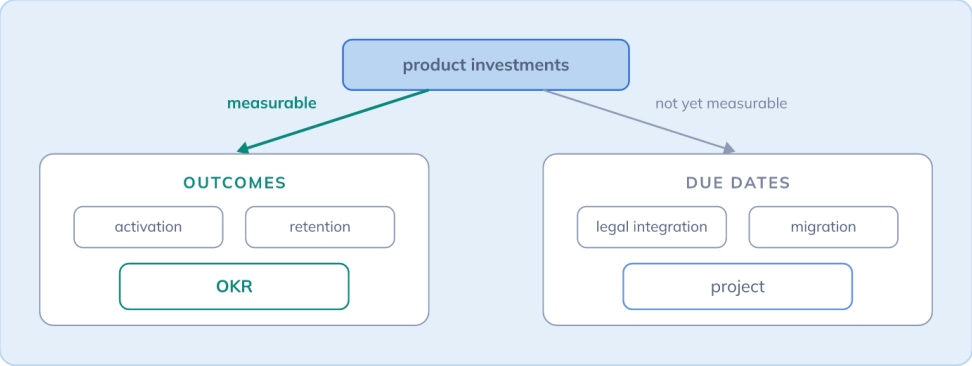

Figure 9. *The two buckets: what has a measurable outcome is managed with OKRs, and what does not have one yet is managed by dates, like a project.*

The two buckets coexist in the same organization. The due dates bucket is not a second-class drawer: it holds an integration required by law, an infrastructure migration, or a delivery agreed by contract, necessary work whose success consists of being done on time. What the criterion avoids is dressing that work up as a key result.

11.2 A practical question for separating the buckets

In practice, the measurability criterion is well supported by a simple question worth asking about every investment.

The question that separates the buckets. Is the success of this work a change you want to bring about, or simply having it done? The former goes into the OKR bucket if the change can be measured; the latter, into the due dates bucket.

The question adds something to Castro's criterion: it forces you to ask whether behind a dated commitment there is actually an outcome that nobody has worded. Often there is, and the work moves to the other bucket as soon as a way to measure it is found.

11.3 Why mixing them makes OKRs unhealthy

When a dated commitment is forced to dress up as a key result, the “implement X by June” of chapter 3 appears. The obsession with turning everything into an OKR, with having all the work appear in the OKRs, is precisely what fills the framework with disguised tasks.

Recognizing that part of the legitimate work has no measurable outcome frees the OKR from having to pretend it does.

11.4 Drawing the line requires judgment

The technique is recognized, but where to draw the line between the two buckets requires judgment, and that is where the community is still talking. The same work can be a dated commitment in one context and an outcome in another.

There is also the temptation to send to the due dates bucket what really should pursue a change, because managing by dates is more comfortable. That is why this knowledge is consolidating: the principle is clear and its fine-grained application is not settled.

What you should be able to do

Having mastered this chapter, a professional is able to:

- Explain Castro's two-bucket technique and its separation criterion, the measurability of the outcome.
- Classify a work item into the outcomes bucket or the due dates bucket, applying the criterion and the practical question.
- Explain why forcing dated commitments into the key result format makes OKRs unhealthy.
- Detect the unworded outcome that may lie behind a dated commitment.
- Reason where to draw the line between the two buckets in a specific case and why it requires judgment.

Common mistakes and antipatterns

- **Putting everything in the OKRs.** Wanting all the work to appear as OKRs fills the framework with tasks disguised as results.

- **Disguising a dated commitment as a key result.** "Launch X by June" as a key result is a misplaced project commitment; it belongs in the due dates bucket.
- **Taking refuge in the due dates bucket for what should pursue a change.** The reverse mistake: parking as a mere deliverable a piece of work whose purpose was to move a metric.
- **Treating the due dates bucket as second-class work.** Dated commitments also need tracking; taking them out of the OKRs does not mean neglecting them.

Further reading

· Felipe Castro — Not everything needs to be an outcome, or an OKR

· Felipe Castro — Is your company using OKR only on paper?

· Scrum Manager BoK — OKR (in Spanish)

BLOCK D · OPERATION: CHECK-INS, CONFIDENCE, AND LEARNING

Chapter 12. The check-in: a decision session, not a reporting session · ESTABLISHED

The check-in is a management session in which the team decides what to do differently based on what the data says.

Introduction

OKRs do not live in the document written at the start of the quarter, but in the rhythm with which the team reviews them. The point where most OKRs die is the check-in degraded into a status report, a meeting in which each person reads out their numbers and nobody decides anything.

In this chapter we will see which question organizes a good check-in, which cadence works best, and how it fits into the events of a team that works with Scrum.

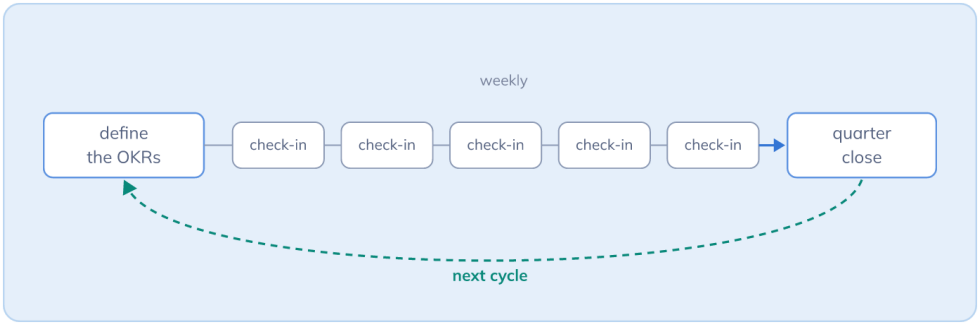

Figure 10. *The rhythm of an OKR cycle: the weekly check-ins decide what to change, and the close of the quarter turns what was learned into the next OKRs.*

12.1 The central question

The question that organizes a check-in is what will be done differently from now on, rather than how much progress has been made. Going over the numbers is only the input; the output of the meeting is the changes of plan that those numbers justify. A check-in that ends without any decision has failed, however thorough the review was.

Reporting versus deciding. Reporting is easier than deciding, and that is why the check-in tends to degrade into reading out figures. Restoring its character as a decision forum is what keeps it useful.

12.2 The short cadence

Reviewing OKRs only once, at the end of the quarter, amounts to being unable to correct anything. That is why a short, regular cadence, weekly or every two weeks, is advisable, giving the team room to maneuver within the cycle.

Christina Wodtke anchored the practice in a weekly rhythm in *Radical Focus*, with two moments she named Monday commitments and Friday wins. On Monday, the team reviews the progress of and confidence in its key results and commits to the tasks that can best help achieve them. On Friday it shows what it achieved, so the week ends by celebrating progress and not just measuring it.

12.3 Fit with the Scrum events

In a team that works with Scrum, tracking OKRs fits naturally alongside the Sprint Review. Alex Ballarín presents it on itnove (2024) as a formula for aligning teams and strategy:

- connect the Product Goal and the Sprint Goal with the OKRs;
- go over the progress of the key results with data and metrics;
- decide from there the changes to the roadmap and the priorities for the next sprints.

This way the check-in stops being one more meeting and becomes the strategic layer of an event the team already holds.

What you should be able to do

Having mastered this chapter, a professional is able to:

- Explain the difference between a check-in as a decision session and as a reporting meeting.
- Word the central question of the check-in and steer it toward producing decisions.
- Justify a short, regular review cadence as opposed to a single end-of-cycle review.
- Describe Wodtke's weekly cadence, with Monday commitments and Friday wins.
- Fit OKR tracking into the Sprint Review, connecting it with prioritization.

Common mistakes and antipatterns

- **The check-in as a round of numbers.** Each person reads out their figures and nobody decides anything; reviewing without deciding is the most common way to kill OKRs.
- **Reviewing only at the end of the quarter.** Without a short cadence there is no margin to correct, and tracking becomes an autopsy.
- **Treating the check-in as an isolated meeting.** Disconnecting it from the Sprint Review and from prioritization turns it into one more burden.

Further reading

· Christina Wodtke — Monday commitments and Friday wins

· itnove — Sprint Review + OKR check-in: the formula for aligning teams and strategy (in Spanish)

· Scrum Manager BoK — Sprint Review (in Spanish)

BLOCK D · OPERATION: CHECK-INS, CONFIDENCE, AND LEARNING

Chapter 13. The confidence system: green, amber, red, and the assumptions · CONSOLIDATING

The status of a key result is the team's confidence in achieving it, and the assumption that supports it, rather than its percentage of progress.

Introduction

A subtle mistake in tracking is reading the status of a key result as its percentage of progress. The confidence system proposes looking at something else, the probability the team assigns to achieving it, and explaining what it is based on.

In this chapter we will see the difference between progress and confidence, Christina Wodtke's original formulation and its traffic-light variant, the role of assumptions, and the culture the system needs.

13.1 Progress is not confidence

A key result at 30% halfway through the quarter may be going well if the momentum is favorable and the team is confident it will get there; another at 60% may be going badly if it has stalled and there are no clear levers to move it. The percentage looks to the past, at what has been covered, and confidence looks to the future, at the probability of getting there.

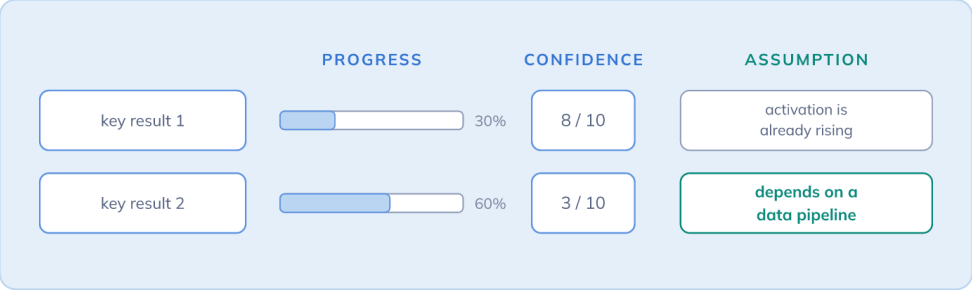

Figure 11. *Progress versus confidence: the status of a key result is the team's confidence in achieving it, and the assumption that supports it, not the percentage covered.*

Central idea. The percentage says where the team is; confidence, whether it believes it will get there. A 60% with low confidence is a more valuable alarm than a 30% with high confidence, because it points to where a decision is needed.

13.2 Wodtke's scale and the traffic light

The original formulation is Christina Wodtke's in *Radical Focus*. Each key result carries a confidence on a scale from 1 to 10, and a demanding result starts at 5, that is, with a 50% probability of achieving it. In the Monday meeting each person reports whether their confidence has gone up or down and why, so it is immediately clear what is getting closer and what is slipping away.

Wodtke presents confidence as one of the two usual systems for managing OKRs; the other is scoring at the end of the cycle, covered in chapter 14. Over the weeks, confidence fluctuates as progress and setbacks appear.

The green, amber, and red traffic light is the usual variant in teams and tools: it translates the scale into three states that are easy to read at a glance. It loses nuance, but keeps the essentials as long as the color expresses the confidence in getting there and not the percentage covered.

13.3 Making the assumptions explicit

Confidence is only useful if it is explained. Declaring a key result at amber with nothing more is opaque; declaring it at amber and adding the assumption makes it actionable: "we are at amber because we are counting on the data team delivering the data pipeline we need this week; if it does not arrive, we move to red."

The assumption turns the color into a conversation about what has to happen to get there and about what to do if it does not.

13.4 The culture it requires

The system is gaining adoption, but it demands a culture in which declaring a risk in time is rewarded instead of punished. If putting a key result at red is experienced as confessing a failure, teams will touch up the color and the system will lose its value as an early warning.

That is why it is consolidating: the mechanics are simple, but they depend on a psychological safety that not every organization has.

What you should be able to do

Having mastered this chapter, a professional is able to:

- Distinguish the percentage of progress from the confidence status of a key result.
- Assign a confidence to a key result, on the 1-to-10 scale or as a traffic light, reasoning about the probability of achieving it and not about what has been covered.
- Explain the origin of the confidence scale in *Radical Focus* and its relationship with the traffic-light variant.
- Accompany a confidence status with the assumption that supports it in order to make it actionable.
- Explain what cultural conditions the confidence system needs in order to work as an early warning.

Common mistakes and antipatterns

- **Confusing progress with confidence.** Reading the color as the percentage covered hides the risk: high but stalled progress looks healthy when it is not.

- **Declaring a color without the assumption.** An amber without explanation does not allow any decision; the value lies in knowing what has to happen to get there.
- **Punishing red.** If declaring a risk is penalized, teams touch up the colors and the system stops giving warning in time.

Further reading

· Christina Wodtke — Tracking and evaluating OKRs

· Christina Wodtke — Monday commitments and Friday wins

· One Knight in Product — Maintaining Radical Focus and Staying on Strategy with OKRs

BLOCK D · OPERATION: CHECK-INS, CONFIDENCE, AND LEARNING

Chapter 14. Validation, learning, and closing the cycle ·

ESTABLISHED

Closing a cycle means checking the outcome and extracting learning for the next one, in addition to scoring the key results.

Introduction

The part of the framework most often skipped in a hurry is also the one that gives it meaning: closing the cycle. Closing a quarter requires more than giving each key result a score, because learning has to be extracted and it has to be checked that the desired change really happened. Without this phase, the next cycle repeats the same mistakes.

In this chapter we will see why reaching the number is not enough, which questions the close should ask, how experimentation helps to attribute an outcome, and how all of this connects with agile empiricism.

14.1 Reaching the number is not having arrived

Completing a key result does not automatically mean having achieved the outcome. The figure can be reached without the underlying change taking place, because the metric was an imperfect proxy or because it moved for an unrelated reason. John Gillespie sums it up on Scrum.org: you should not assume you have arrived just because you have completed a key result.

The closing question. It is not enough to ask whether the key result was met. You have to add whether the outcome happened and which assumptions turned out to be false, because that is where the learning that improves the next cycle comes from.

14.2 The closing questions

A useful close reviews, for each objective:

- whether the outcome was achieved or only the number;

- which initial assumptions turned out to be false;
- what was learned about the user or the business;
- what changes in the strategy or in the next OKRs as a result.

These questions turn the end of the cycle into the informed start of the next one.

14.3 Experimenting to attribute

Knowing whether an outcome is due to what the team did is not always obvious. Cagan points out (October 2026) that the product model revolves around outcomes and that the way to attribute them is experimentation with real data. With a product in use and customers generating data, the team can check what effect a change had instead of assuming it.

Cagan also stresses that often the purpose of an experiment is to understand why the results are what they are, and not just to prove something. That understanding is the raw material of the close: a key result achieved without knowing why teaches little for the next cycle.

14.4 Connection with agile empiricism

The close is the OKR expression of agile empiricism: transparency (the data in view), inspection (the honest review of what happened), and adaptation (the adjustment of objectives and results). Yeret insists that OKRs reflect the strategic direction at a given moment, and that what is learned can lead to changing a key result or even the objective.

Ballarín observes that, with two-week sprints and quarterly OKR cycles, a team holds six Sprint Retrospectives for each retrospective of the OKR cycle. The former can focus on technical or process improvements; the latter focuses on how to improve the results of the work. Without this closing phase, OKRs lose their empirical character and become just another quarterly plan.

What you should be able to do

Having mastered this chapter, a professional is able to:

- Explain why reaching the number of a key result is not the same as having achieved the outcome.
- Lead the close of a cycle based on the learning questions, beyond the score.
- Explain how experimentation with real data makes it possible to attribute an outcome to what the team did.
- Identify the failed assumptions of a cycle and translate the learning into adjustments for the next one.
- Relate the close of the OKR cycle to agile empiricism and to the team's retrospectives.

Common mistakes and antipatterns

- **Closing by scoring and nothing else.** Reducing the close to a score per key result wastes the learning and condemns the next cycle to repeat mistakes.
- **Taking the outcome as achieved because the number was reached.** The number may move for unrelated reasons or be an imperfect indicator; the real change has to be checked.
- **Claiming the result without checking it.** Without experiments or data to back it up, celebrating an outcome as your own is an assumption.
- **Treating OKRs as fixed planning.** Without inspection and adaptation at the close, OKRs become a rigid quarterly plan.

Further reading

· Marty Cagan (SVPG) — Experimentation, AI and the Product Model

· Scrum.org — Transforming an output-oriented mindset to an outcome-oriented mindset using OKRs

· Scrum.org — Organizing around outcomes with OKRs and Scrum

· Scrum.org — How do OKRs empower Scrum Teams? (in Spanish)

· Scrum Manager BoK — Retrospective (in Spanish)

BLOCK D · OPERATION: CHECK-INS, CONFIDENCE, AND LEARNING

Chapter 15. AI in writing and tracking OKRs · EMERGING

Using AI assistants to propose, critique, and track OKRs, with the team responsible for every decision.

Introduction

The previous version of this guide pointed to artificial intelligence as the novelty to watch in the operation of OKRs. In 2026 the signal is visible, but still weak: assistants have been built into the tools and some reference authors are starting to address the topic, although nobody has yet written a settled doctrine.

In this chapter we will see what AI can be useful for in the OKR cycle, what the little evidence available says, what risks it carries, and what principle of use follows from the rest of the guide. It is an emerging topic and is best read as a provisional state.

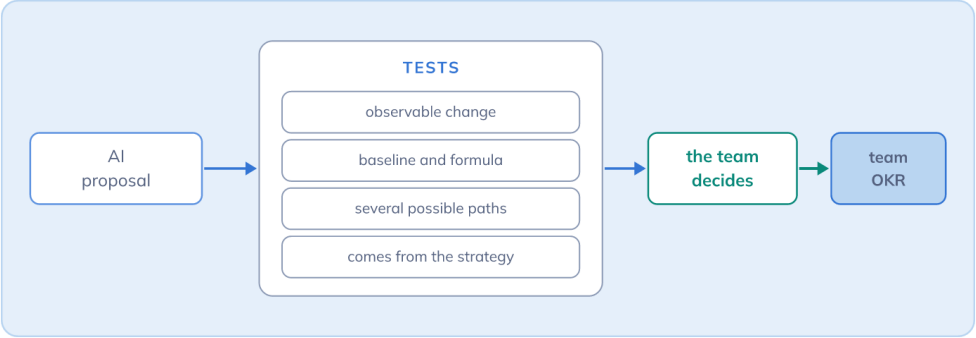

Figure 12. *The AI proposes, the team decides: the drafts and critiques the AI proposes go through the same tests as any OKR before the team makes them its own.*

15.1 What it can be useful for

The uses that make sense are those that apply, more quickly, the criteria we already know:

- **Writing and critiquing drafts.** Proposing objectives and key results based on the strategy, and above all reviewing the team's own: flagging those that measure output, those without a baseline, or those that can only be achieved in one way.
- **Detecting misalignments.** Comparing the OKRs of several teams and flagging overlaps, gaps, or contradictions.
- **Preparing check-ins.** Summarizing the progress of the key results and the week's data so that the meeting is devoted to deciding.
- **Synthesizing the close.** Organizing what was learned in the cycle, the assumptions that failed, and the open questions.

As in other uses of AI in agile work, it performs better reviewing than writing: asking it to critique a key result written by the team is more reliable than asking it to write one.

15.2 What the evidence says

In January 2026 Christina Wodtke devoted a Lean Product Meetup session to writing better OKRs with AI. It is the clearest signal that the reference authors are starting to address the topic, although of that session only the public announcement is on record, not a text that sets out criteria.

The only usage data available comes from the *OKR Intelligence Report 2026* by OKRs Tool, a vendor of OKR software, covering 222 tech companies with 51 to 200 employees. According to the report, 83% use AI in their OKR process:

- 47% take it as a good starting point that always needs human refinement;
- 20% use it only as inspiration and thoroughly rewrite what it proposes;
- 13% accept its suggestions as they are, which the report itself considers a risk pattern;
- 11% review what it produces with skepticism and rarely adopt it.

Caution with the data. It is a vendor report, with no published collection method and with a sample of mid-sized tech companies. It serves as an indication that use is widespread, not as a measure of how it is used in general.

15.3 Risks

The risks follow from the previous chapters, rather than from specific studies:

- **Generic OKRs.** An assistant produces correct, plausible statements that could serve any company, and that for that very reason do not translate the company's own strategy (chapter 8).
- **Measuring what is easy.** The proposals tend toward what can be counted easily, which is often output (chapter 1).
- **Loss of ownership.** An OKR the team has not thought through is not its own, and the empowerment effect that justifies the technique is lost (chapter 2).
- **Exposed strategic information.** A company's objectives and metrics are sensitive information, and sharing them with a tool requires knowing where the data goes.

15.4 A principle of use

The AI proposes and the team decides. Every objective or key result an assistant suggests goes through the same tests as any other: whether it describes an observable change, whether it has a baseline and a formula, whether it allows several paths, and whether it comes from a strategy decision.

Which tools are used, with what data, and for what tasks is a team decision that is worth making explicit. Scrum Manager's AI applied to agile work skill develops that usage policy, the verification of what AI produces, and the protection of information.

What you should be able to do

Having mastered this chapter, a professional is able to:

- Identify at which moments of the OKR cycle an AI assistant can help (drafting, critique, alignment, check-in, and close).
- Use AI to critique an existing key result by applying the criteria of the outcome, the baseline, and the multiple paths.
- Assess the strength of the available evidence on the use of AI in OKRs and distinguish a vendor figure from a general measure.
- Recognize the risks of accepting without review the OKRs an assistant proposes.
- Apply the principle that the AI proposes and the team decides, and explain why it protects the team's empowerment.

Common mistakes and antipatterns

- **Accepting the suggestions as they are.** An OKR nobody has thought through tends to be generic, to measure what is easy, and to belong to no one.
- **Asking the AI to set the objectives.** Deciding which problems matter is the strategy; delegating it empties everything that follows of meaning.
- **Taking a vendor figure as general evidence.** The available usage figures are a partial indication, useful for orientation and little more.
- **Sharing the strategy without thinking about the data.** Objectives, metrics, and plans are sensitive information, and their use with external tools must be agreed.

Further reading

· Lean Product Meetup — How to craft better OKRs with AI, by Christina Wodtke

· OKRs Tool — OKR Intelligence Report 2026

· Skill Arena — AI applied to agile work

© 2026 Scrum Manager®. This work is published under a Creative Commons Attribution-NonCommercial 4.0 International licence (CC BY-NC 4.0). Official Scrum Manager trainers and centres are licensed under the terms of CC BY 4.0 for their training activity.