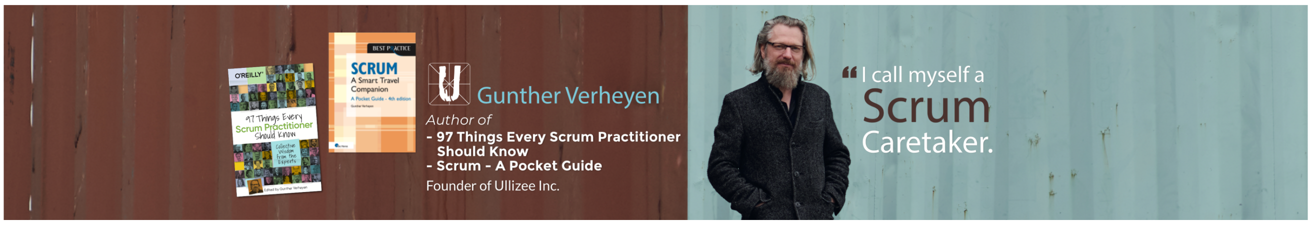
Whitepaper

Scrum. Not Scrum.

Redefining the Future Through the Roots

by Gunther Verheyen
independent Scrum Caretaker

March 2026


Gunther Verheyen
Author of
- 97 Things Every Scrum Practitioner Should Know
- Scrum - A Pocket Guide
Founder of Ullizee Inc.
I call myself a Scrum Caretaker.

Technology and cognitive, creative work have become critical drivers of organizations and of society at large. At the same time, the playfield of development is going through fundamental changes. Is it not time to redefine how to organize development? Revisiting the roots of what is called "Scrum" shows the path to shaping the future of development. In 1986, Hirotaka Takeuchi and Ikujiro Nonaka shared their research on successful development organization in the field of new product development. They called it "the rugby approach" because it revolved around self-organizing, multidisciplinary teams. In 1995, Jeff Sutherland and Ken Schwaber used "Scrum" as the name for their new software development process. The Agile Manifesto was published in 2001. Is it not time to move on and emerge a new development paradigm?

The Rugby Approach and Scrum

In their 1986 paper *'The New New Product Development Game'* Takeuchi and Nonaka described cases of companies developing non-software products, like Fuji-Xerox (a medium-sized copier), Honda (a City car), Canon (a personal-use copier and two cameras) and NEC (a personal computer)[1]. They described the six characteristics of the new "rugby approach" development method of self-organizing, multidisciplinary teams.

The potential value of 'self-organization' in the field of software development was pointed out by Peter DeGrace and Leslie Hulet Stahl in 1990. They briefly described "Scrum" as one "righteous" approach in their *'Catalogue of Modern Software Engineering Paradigms'*[2]:

> *"Suppose you have a software development project to do. (…) Rather than have several designers do the design phase and have several coders do the construction phase, etc., you form a team by carefully selecting one person from each pool. (…) You further unsettle the team by saying that their job is to produce the system in, say, half the time and money and it must have twice the performance of other systems. Next, you say that how they do it is their business. Your business is to support them in getting resources. Then, you leave them alone."*

In his essay *'The Origins of Scrum Might Not Be What You Think They Are'*, Rafael Sabbagh shared more background on this forgotten stage in the history of Scrum[3].

In 1995, Jeff Sutherland and Ken Schwaber used "Scrum" as the name for their new software development process. Scrum as we know it today was mainly shaped through their subsequent work with a group of patterns people. It lead to the introduction in 1998 of patterns like the (daily) "Scrum Meeting", "Sprint", "Backlog", "Scrum Team" and "Scrum Master"[4]. Mike Beedle of that group was the third representative of the early Scrum movement to become a signatory of the Agile Manifesto in 2001[5]. In 2010 Schwaber and Sutherland published the first Scrum Guide with the latest update released in November 2020[6].

My whitepaper *'Scrum: A Brief History of a Long-Lived Hype'* documents these and other important evolutions of the definition of Scrum since 1995[7].

The key for outstanding performance in the new new development rugby approach is self-organization: "The process of people forming organized groups around problems or challenges without external work plans or instructions being imposed on them". Multidisciplinary teams are not given tasks but figure out themselves how to develop a product upon challenging objectives.

Scrum defines that Scrum Teams are "self-managing, meaning they internally decide who does what, when, and how" and that "No one else tells the Developers how to turn Product Backlog items into Increments of value"[6].

Although "self-organization" was officially replaced by "self-management", I still consider the former to be the actual overarching management principle.

What Else?

Takeuchi and Nonaka observed how in the new new development organization that they called the rugby approach, the actual process emerged from the interactions between the team members.

Scrum explicitly implements an empirical process: "The process control type in which decisions are based on observed results, experience and experimentation". People use the Scrum artifacts to inspect their work and their process in order to adapt—minimally at the time of the Scrum events. It replaced the staged, predictive ("fully defined") process that dominated software development.

I call the combination of self-organization and empiricism "Scrum's DNA" (see Exhibit 1):

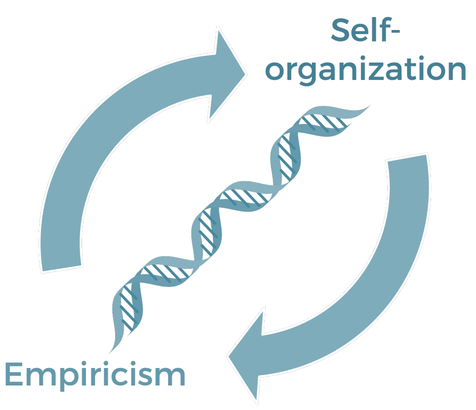


Exhibit 1: Scrum's DNA

Essentially, the rules of Scrum serve to establish a frame of boundaries within which teams devise their own way of working. The deeper value of Scrum is therefore not in the process but in the *behavior* of the people and the stance they take.

[1] "The New New Product Development Game", Hirotaka Takeuchi & Ikujiro Nonaka, Harvard Business Review, 1986.

[2] "Wicked Problems, Righteous Solutions (A Catalogue of Modern Software Engineering Paradigms)", Peter DeGrace & Leslie Hulet Stahl, Prentice Hall, 1990.

[3] "97 Things Every Scrum Practitioner Should Know", Gunther Verheyen, O'Reilly Media, 2020.

[4] "SCRUM: An extension pattern language for hyperproductive software development", Mike Beedle, Martine Devos, Yonat Sharon, Ken Schwaber & Jeff Sutherland, 1998.

[5] agilemanifesto.org, agilemanifesto.org/principles.html, Beck, K., Beedle, M., v. Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., Thomas, D., 2001.

[6] "The Scrum Guide", Jeff Sutherland & Ken Schwaber, 2020.

[7] "Scrum: A Brief History of a Long-Lived Hype", Gunther Verheyen, Ullizee-Inc, 2020.

Overlapping Phases

In the paper '*The New New Product Development Game*' the authors distinguish three types of development organization (see Exhibit 2):

- Type A is the traditional, linear approach. The product development process advances by sequentially going from phase to phase. In each separate phase specific functional work is performed before "passing the baton" to the next group of specialists for the next phase consisting of their work. The authors used the "Phased Program Planning" (PPP) of the National Aeronautics and Space Administration (NASA) to illustrate the Type A development organization. The PPP consisted of the six (sequentially organized) phases of *Concept Development*, *Feasibility Testing*, *Product Design*, *Development Process*, *Pilot Production* and *Final Production*.
- Type B is the development approach that has some overlap at the border of adjacent phases. A phase starts before the preceding phase is fully completed. An example is the phase of *Product Design* starting before all results of *Feasibility Testing* are in. The authors share how Fuji-Xerox achieved spectacular improvements with the development of the FX-3500 medium sized copier by moving from a Type A to a Type B development organization. They do stress the importance of the "constant interaction" of team members working together "from start to finish" –"also with suppliers"–and engaging in iterative experimentation "in even the latest phases".
- Type C is the development organization that has overlap that extends across several phases. A ("hand-picked") multidisciplinary team does all the work that used to be organized in separate phases, concurrently. The authors used Honda and Canon as examples for their observations of Type C development. They stress how the actual product development process "is born out of the team members' interplay"–rather than consisting of "defined, highly structured stages". The shift, they say, "stimulates new kinds of learning and thinking within the organization at different levels and functions".

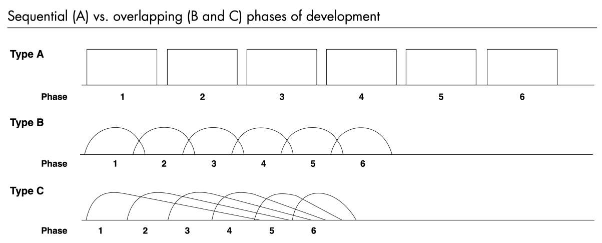

Sequential (A) vs. overlapping (B and C) phases of development
Type A
Phase
1
2
3
4
5
6
Type B
Phase
1
2
3
4
5
6
Type C
Phase
1
2
3
4
5
6

Exhibit 2: Sequential vs. Overlapping Phases of Development

Takeuchi and Nonaka initially refer to both Type B and Type C development as "the rugby approach" as opposed to the "relay-like" approach of Type A development: "as in rugby, the ball gets passed within the team as it moves as a unit up the field". They do further typify Type B as the "Sashimi" system–with the overlapping phases resembling slices of raw fish arranged on a plate–and Type C as the more holistic rugby approach.

While there is no mandatory process in the new new development rugby approach, 'development' does encompass *all* the stages in the lifecycle of a product (see Exhibit 3). The Type C approach is described as most appropriate *for that purpose*.

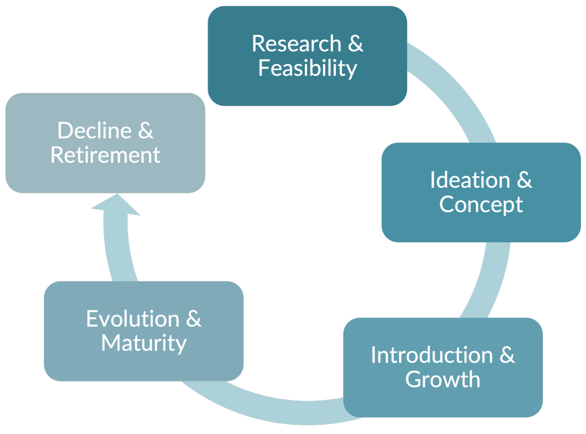


Exhibit 3: The Product Lifecycle

One of the cases described in '*The New New Product Development Game*' is that of the development of the FX-3500 medium-sized copier at Fuji-Xerox. Its development team consisted of "members from the planning, design, production, sales, distribution, and evaluation departments". The team could thus cover even the earliest stages in the product's lifecycle. A team member is quoted on the rationale for locating the complete multifunctional team together in one large room:

"When all the team members are located in one large room, someone's information becomes yours, without even trying. You then start thinking in terms of what's best or second best for the group at large and not only about where you stand. If everyone

understands the other person's position, then each of us is more willing to give in, or at least to try to talk to each other. Initiatives emerge as a result."

Scrum officially defines a strict Type C approach where *all* work is done within *each* Sprint. However, this does *not* extend to the complete lifecycle of a product. It is typically limited to the implementation work only and thus the product lifecycle stages of *Introduction & Growth* and *Evolution & Maturity*. In the PPP methodology the *Development Process* phase would be the point where Scrum starts.

The Purpose of Development

Even if the belief is true that pressuring people to deliver more or 'work harder' works in a manufacturing environment or other forms of mechanized work, it certainly does not make sense for cognitive, creative work with technology.

Generating value is the shared purpose for knowledge work. 'Product' (defined broadly and outside-in) is the vehicle to generate value. Although output is needed, only outcome is the measure of success. And that is what 'value' is about: the impact and the outcome of work. Value is in the appreciation of the people receiving and using the output and the broader effect of that on an organization and its many ecosystems.

The official definition of Scrum does nominate 'value' as its purpose: "Scrum is a lightweight framework that helps people, teams and organizations generate value through adaptive solutions for complex problems."[6] This is repeatedly emphasized throughout the Scrum Guide. The first principle of the Agile Manifesto also states that "Our highest priority is to satisfy the customer through early and continuous delivery of valuable software"[5].

Yet, the focus of time-boxed iterations is typically on the internal operations and the produced output. In practice, the focus is hardly on the impact and outcome of the work. Assessing the value generated through the product is out of the scope of the development organization.

As 'value' is the purpose of cognitive, creative work, it is *the* angle to look at development organization (see Exhibit 4):

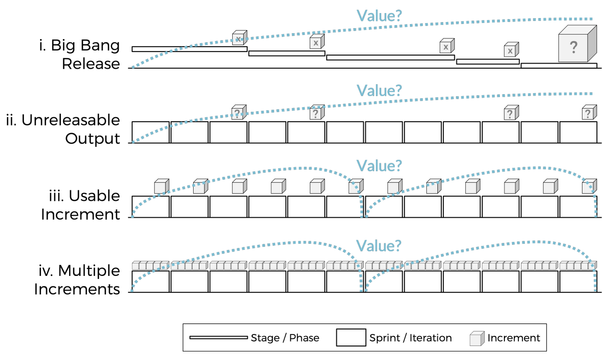


Exhibit 4: Output vs. Outcome

- Situation i is the traditional development approach that is often referred to as "Waterfall". *This is much comparable to Type A development with phases for specific types of work being sequentially organized.* Each stage or phase must be completed and signed off before the next one can start. Typical phases are *Requirements*, *Analysis*, *Design*, *Coding*, *Testing* and *Integration*. The work is often organized as a fixed price project aiming at one ('big bang') release. All value is expected to be generated through that release. Practice shows how such large endeavours are typically late, over budget and never 'complete', despite stubbornly fixing these three major variables. And during the project, expectations and conditions invariably change so that the final result is no longer what is expected or needed at the time of release. The initially envisioned value has dissipated.
- Situation ii represents a vast majority of 'Agile' implementations around the world. Work is organized in time-boxed iterations, but they do not result in output that can be released in production. There is an amount of open (post-iteration) work remaining to reach that state: work to be performed by the same team in subsequent iterations, work by other teams (after which the work is bounced back), missing activities due to a lack of skills or expertise, work to integrate with the work of other teams or work to be performed by other departments. It is unclear when a releasable "Increment", as it is called in Scrum, will become available, let alone when it is released and how value will be generated. *This situation is actually quite comparable to waterfall development.*
- Situation iii is the uncommonly sophisticated implementation of 'Agile' of a small minority of implementations around the world. Every time-boxed iteration results in an Increment as releasable output. It is a product version that could be released in production. There is no open (post-iteration) work that still needs to be performed to achieve that state: no missing activities and no work to be performed in subsequent iterations, by other teams, for integration purposes or by other departments. There is the potential of actually generating value because there is potentially releasable output. In practice, it does remain unclear when the output is released and even if it is, the actual impact is rarely measured and acted upon. Ideally, the impact would be known by the end of the next iteration. In practice, it is not the focus because it is out of scope.
- Situation iv is an alternative of the uncommonly sophisticated implementation of 'Agile'. Multiple releasable Increments are produced throughout an iteration, potentially even on a daily base. These product versions might be ready for easy manual or for automatic deployment in production. The output is minimally ready to be released or is even actually released. Ideally, the

actual impact would be known by the end of the current or the next iteration. In practice, it is ignored for being outside of the scope of the development process—as in situation iii.

Although time-boxed iterations are not mentioned or prescribed in the Agile Manifesto, they are in Scrum and eXtreme Programming. Whether called "Sprints" or not, organizing work in time-boxed iterations is a generally applied 'Agile' practice.

A New Development Paradigm

A scrum in the game of rugby serves to "Restart play" or to "Get the ball back in the game". These metaphors capture what we have been trying to achieve with Scrum: getting the ball back in the game of software development because the game had derailed; restart play because the ball had gotten unplayable. Despite the many challenges, Scrum *has* replaced the waterfall process.

A major challenge is to keep the ball in the game. Unfortunately, Agile and Scrum are subsumed, recuperated and neutralised by organizations and instances that pretend to shift to the Agile paradigm while they actually keep operating within the industrial paradigm. They twist Scrum to fit their existing structures rather than adapt the structures to maximize the benefits of Scrum. The result is that they create an illusion of agility, rather than upgrading to a more agile organization.

In 2019 I shared my observations regarding this effect in my whitepaper *'The Illusion of Agility (What Most 'Agile Transformations' Deliver)'*[8].

In my pocket guide to Scrum I shared how in the fourth Scrum Wave more is required than merely adopting Scrum as the new development process[9]. Scrum is used only effectively by organizations that re-organize around it and emerge new structures doing so. *Yet, that is exactly what most organizations resist.*

In many of today's organizations the reigning beliefs are rooted in the past, industrial age. Many executives, managers and even teams are still unable and even unwilling to move beyond the industrial recipes that they are familiar with. The reality of many of today's organizations is that they are organized for volume and not for value, even after adopting Agile and Scrum.

Given the reality of today's business and the acceleration of technology and its impact, the more fundamental question is whether 'agility' still is the right goal—if it even ever was? There actually is a bigger problem underlying that of "the illusion of agility". It is the illusion of striving for agility.

Getting the ball back in the game is not enough. Keeping the ball in the game is not enough. We need to take the game to the next level(s). Because technology and cognitive, creative work are critical drivers of organizations and society at large, while the playfield of development is going through drastic changes. Because we need to improve our ability to navigate complexity.

We must look beyond our past achievements to redefine the future. Going to the next level of work organization is going beyond ad-hoc and waterfall, beyond Type B or Type C development (1986), beyond Scrum (1995-1998) and Agile (2001) (see Exhibit 5). A new development paradigm cannot build on ideas and concepts from manufacturing—directly nor indirectly. *'Productivity' is about value and is aligned with 'humanity'.* Its adoption will start with a small group of pioneers and visionaries before entering and crossing the chasm. The existing approaches will remain to linger on.

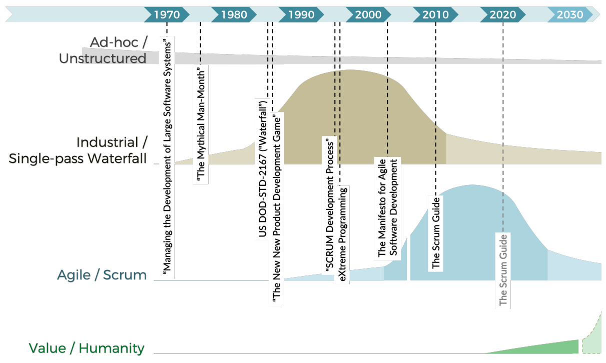


Exhibit 5: Development Paradigms at Large

Leaders and proponents of Agile or Scrum have done little or nothing to inspire or challenge organizations to implement Type C development as described by Takeuchi and Nonaka, with 'development' covering the complete product lifecycle. The most probable reason is that Agile and Scrum have their roots in software development. The ultimate Type C approach covering the whole product lifecycle is hardly known, let alone effectively applied.

And even that wouldn't suffice to redefine product development organization today. Taking the game to the next level means no longer relying on approaches where 'productivity' is completely about volume and the focus is only on what yields the highest 'productivity' gains.

A traditional definition of productivity is "The rate at which a worker, a company or a country produces goods" or "An amount produced compared to how much time, work and money is needed to produce it". It is high time to go beyond the productivity obsession because it does not fit cognitive, creative work and the steep rise of technology. A definition for 'productivity' in the context of creative, cognitive work is "The ability to produce valuable outcomes."

That even includes letting go of managing for 'performance' because that ultimately is just synonymous to old-school 'productivity' and the illusionary way to 'success': producing higher volumes of work. In the interaction game of navigating complexity 'performance' and 'productivity' are side effects of collaboration. *The focus should be on the latter to get the former as part of a very different management paradigm.*

Given the need to shift from volume to value, it is time for deep, systemic change to optimize how work is organized, the applied processes as well as how organizations are set up. It is time to redefine 'productivity' and add 'humanity' to the equation as we optimize our organizations for value.

"This strategy for product development can act as an agent of change for the larger organization." [1]

In the spirit of '*The New New Product Development Game*', we could picture an evolved version of Type C development. Except, the goal of such 'Type D' development should go beyond faster production or more output. The focus has to be on value as the measure of progress and success. The "Sashimi" metaphor still serves, but to describe the ability to serve 'Sashimi' slices of product to the product's consumers. But 'development' must include actively collecting feedback from them about the tastiness and act upon the findings.

Expanding human accountability to development domains not covered by technology is how to benefit from the improved opportunities to create technology products—like with the advent of AI. *The future is not about adding or removing people but about acquiring new skills and on-going professional development.*

What Else?

The new development paradigm requires different, new organizational structures.

That starts with forming '**Development Hubs**' as organizational entities that have everything—the mandate, resources, skills and support—to develop and maintain products autonomously.

In short, to form and grow a Development Hub:

1. Identify the product.
2. Create tasty Product Slices.
3. Release and measure to learn and to adapt.

With every product moved to a newly formed Development Hub the silos of the traditional development organization dissipate a little bit.

Combined, all Development Hubs form a '**Development Studio**'—an alternative, networked structure residing within the larger organization.

A Development Hub becomes an **ARMADA** of value by rigorously ADApting upon the insights gained from ARMing for value.

In short, the players in a Development Hub repeatedly:

1. **A**ssume Value (Development).
2. **R**elease Output (Product Slices).
3. **M**easure Value (Outcome, Impact).
4. **ADA**pt (to optimize for value).

A Development Hub is autonomous in product development but semi-autonomous for being dependent on the larger organization.

While existing Development Hubs grow and new ones spin up, the skills and mandate to address *all* stages in the product lifecycle are added to evolve to a '**Product Hub**'. The traditional silos and layers dissolve further to ultimately completely disappear. The organization is now a networked structure of Product Hubs (see Exhibit 6).

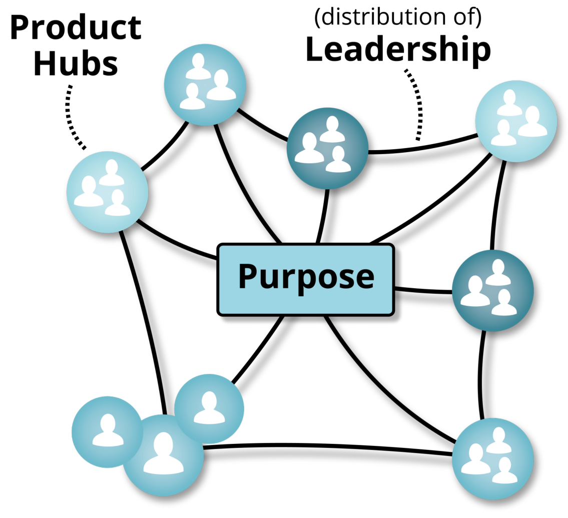


Exhibit 6: A Networked Structure of Product Hubs

[8] "The Illusion of Agility (What Most 'Agile Transformations' Deliver)", Gunther Verheyen, Ullizee-Inc, 2019.
[9] "Scrum - A Pocket Guide (A Smart Travel Companion)", Gunther Verheyen, Van Haren Publishing, 2024.

*I am describing the new development paradigm as part of a cohesive vision for systemic organizational change towards more humane organizations in **a new book**.*

Many ideas have been lingering in my head for a long time. I shared them in whole or in part in my consulting work, in my classes and workshop and at various events, meet-ups and gatherings. In the end, it has taken me a considerable time to distill and fine-tune them. Ultimately, I couldn't have written a book any sooner than until now.

It will also help those that haven't adopted Scrum and have no plans to do so. It takes the reader beyond what most understand as Agile and Scrum anyhow. It goes beyond anything we know today all together.

It is for those courageous pioneers and leaders who want to overcome the fear to touch, update and re-think organizational structures, procedures and constructs. Together we will challenge the status quo at large to help shape a better world to live and work in.

About the Author

Gunther Verheyen calls himself an independent (Scrum) Caretaker and Workplace Humanitarian on a journey of humanizing the workplace. He is a long-time Scrum practitioner who started applying Scrum in 2003. Besides having engaged with many teams and organizations, he has published two acclaimed books about Scrum and was the partner of Ken Schwaber (co-creator of Scrum) as Director of the "Professional Scrum" series at Scrum.org. Although his professional life has mostly revolved around Scrum, the aspiration that drives him in the first place is: to create a better world to live and work in.

Gunther ventured into IT and software development after graduating as Engineer in Electronics in 1992. His Agile adventures started with eXtreme Programming wrapped in Scrum in 2003. Upon his broad practical experience with Scrum, he became the inspiring force behind some large-scale enterprise transformations in 2010. In 2011 he acquired his license as a Professional Scrum Trainer for Scrum.org.

Gunther left consulting in 2013 to establish Ullizee-Inc and partner with Ken Schwaber, co-creator of Scrum. He managed the "Professional Scrum" series of Scrum.org and stewarded its global network of Professional Scrum Trainers. He co-created Agility Path, EBM (Evidence-Based Management) and the Nexus framework for Scaled Professional Scrum.

Since 2016 Gunther is continuing his journey to humanize the workplace as an independent Scrum Caretaker; a connector, teacher, writer, speaker. As such, he goes beyond merely applying Scrum as a process by actively nurturing and upholding the values, behaviors and people-centric aspects that make Scrum more effective.

Gunther created his acclaimed book *Scrum – A Pocket Guide* in 2013, with updates published in 2019, 2021 and 2024. In 2020 he published the book *97 Things Every Scrum Practitioner Should Know*, a collection of essays from field experts across the world. Gunther is currently (2026) writing a new book about a new development paradigm and organizational design to redefine 'productivity' and align it with 'humanity'.

When not travelling to humanize the workplace, Gunther lives and works in Antwerp (Belgium). More at guntherverheyen.com.