Scrum Master: Teamführung neu definiert und agil gestaltet

Category: Online-Marketing
geschrieben von Tobias Hager | 17. August 2025



Scrum Master: Teamführung neu definiert und agil gestaltet

Du hast Post-its, ein Jira-Board und jeden zweiten Dienstag eine Retro – und trotzdem brennt dein Sprint-Backlog lichterloh? Willkommen in der Realität moderner Teamführung, in der ein Scrum Master nicht der Meeting-Moderator mit freundlichem Lächeln ist, sondern der Architekt für Flow, Fokus und Vorhersagbarkeit. In diesem Artikel zerlegen wir den Mythos der agilen

Kuschelecke, definieren die Rolle Scrum Master neu, und zeigen, wie echte Führung im agilen Kontext funktioniert: datengetrieben, systemisch, gnadenlos transparent – und messbar erfolgreich.

- Was ein Scrum Master wirklich tut – und warum das mit "Meeting organisieren" nur am Rand zu tun hat
- Agile Führung im Detail: Servant Leadership, Systemdenken und Organizational Design
- Die wichtigsten Metriken, die ein Scrum Master beherrschen muss: Flow, Throughput, Cycle Time, WIP
- Werkzeuge, die wirken: Jira, Azure DevOps, CFD, Monte-Carlo-Forecasting, Value Stream Mapping
- Die häufigsten Anti-Pattern – und wie du sie gnadenlos eliminierst
- Scrum Master in skalierten Umgebungen: SAFe, LeSS, Nexus und die harte Wahrheit über Abhängigkeiten
- Ein Schritt-für-Schritt-Playbook für nachhaltigen Impact in Sprints, Quartalen und Roadmaps
- Zertifizierungen, Skills und Karrierepfade – was wirklich zählt und was nur Logo-Sammeln ist
- Am Ende: Warum ein guter Scrum Master eine Führungskraft mit Produktinstinkt und technischer Neugier sein muss

Scrum Master ist die Rolle, die Teams trägt, wenn Methodenhype und Folienglanz verpuffen. Ein Scrum Master formt die Teamführung, schafft Bedingungen für Fokus und Qualität und hält eine Organisation auf Kurs, die sonst jede Woche neue Prioritäten erfindet. Der Scrum Master ist kein Projektmanager im Tarnanzug, sondern eine Führungskraft, die Systeme baut, Bottlenecks findet und Blockaden beseitigt. Der Scrum Master ist die Person, die Messbarkeit fordert und Raffinesse liefert, wenn alle anderen noch über Story Points philosophieren. Der Scrum Master ist derjenige, der Empirie ernst nimmt und Hypothesen testet statt Meinungen zu verhandeln. Kurz: Ohne Scrum Master bleibt Agilität ein Poster an der Wand.

Der Scrum Master sorgt nicht für Wohlfühl-Meetings, sondern für einen belastbaren Wertstrom. Wer Teamführung neu denken will, setzt auf Evidenz und Flow, nicht auf Agenda-Feenstaub. Deshalb reicht es nicht, Daily, Planning und Review zu "hosten". Der Scrum Master muss den End-to-End-Flow verstehen, vom Refinement über Trunk-Based Development bis zur produktiven Auslieferung. Er muss die Sprache der Entwickler, Designer, Tester und Stakeholder sprechen – und die Logik von Produkt, Architektur und Betrieb verbinden. Er muss in Metriken denken und in Beziehungen handeln. Und ja, er muss die Organisation konfrontieren, wenn Policies, Silos und Bürokratie das System verstopfen.

Das mag den Charme einer kalten Dusche haben, aber es ist wirksam. Ein starker Scrum Master etabliert DoR und DoD, reduziert WIP, moderiert Konflikte professionell und entfernt systemische Hindernisse. Ein starker Scrum Master setzt auf Transparenz durch Kanban-Praktiken, nutzt Cumulative Flow Diagrams und steuert Erwartungen mit probabilistischen Forecasts statt Astrologie. Ein starker Scrum Master verhindert Heldenkult und baut verlässliche Arbeitsrhythmen, die Produktteams langfristig tragen. Und ja, der starke Scrum Master kennt Tools, versteht Deploy-Pipelines und weiß, dass ohne CI/CD und Feature-Toggles kein Sprint "done" wird. So sieht Teamführung

in agil wirklich aus – unromantisch, aber effektiv.

Scrum Master Rolle verstehen: agile Führung, Servant Leadership und harte Systemarbeit

Die Rolle Scrum Master ist in der Theorie simpel und in der Praxis anspruchsvoll. Auf dem Papier dient der Scrum Master dem Team, dem Product Owner und der Organisation – Servant Leadership eben. In der Realität bedeutet das, unpopuläre Wahrheiten auszusprechen und strukturelle Hindernisse zu beseitigen, die niemand "besitzt", aber alle behindern. Es bedeutet, Meetings so zu gestalten, dass sie Entscheidungen forcieren, nicht Zeit vernichten. Es bedeutet, die Regeln des Systems zu kennen und zu verändern, statt Menschen zu mehr "Engagement" zu motivieren. Kurz: Der Scrum Master ist weniger Moderator und mehr Systemdesigner.

Servant Leadership ist kein Kuschelkurs, sondern ein Führungsstil, der Klarheit, Verbindlichkeit und Schutz für produktive Arbeit priorisiert. Der Scrum Master schützt das Team vor chaotischem Scope, vor hüpfenden Zielen und vor unlimitiertem Parallelisieren. Er schafft psychologische Sicherheit – nicht durch warme Worte, sondern durch verlässliche Arbeitsweisen, klare Definitionen und transparente Metriken. Er befähigt das Team, die eigene Arbeitsweise zu reflektieren, Hypothesen zu testen und kontinuierlich zu verbessern. Servant Leadership ist damit die Praxis, Macht für das System und nicht über Menschen zu nutzen.

Teamführung im agilen Kontext heißt, den Wertstrom als Ganzes zu sehen. Ein Scrum Master pflegt nicht nur ein Backlog – er kuratiert einen Fluss von Ideen zu wertstiftender Software. Er erkennt, wo Arbeit warten muss, wo Qualität leidet, und wo Eingriffe in Architektur, Deployment oder Teststrategie nötig sind. Er bringt Product Owner dazu, Entscheidungen zu treffen, die die Durchlaufzeit respektieren. Und er coacht das Management, Ziele als Hypothesen zu formulieren und Risiken messbar zu machen. So entsteht Führung ohne Mikromanagement, aber mit maximaler Klarheit.

Scrum Master sind Brückenbauer zwischen Produktstrategie und operativer Realität. Sie kennen Impact Mapping, Opportunity Solution Trees und die Logik von OKR, aber ebenso Taktiken wie Pairing, Mob Programming und Exploratory Testing. Sie wissen, dass Work-in-Progress-Limits kein Dogma sind, sondern eine Absicherung gegen Kontextsprüngen und versteckte Multitasking-Kosten. Und sie passen das Framework an die Domäne an, statt Scrum als Religion zu predigen. Wer Rolle und Verantwortung so versteht, liefert echte Ergebnisse und baut belastbare Teams.

Scrum Master Verantwortlichkeiten: Facilitation, Empirie, Artefakte und Flow-Mechanik

Ein Scrum Master ist der Garant dafür, dass Empirie nicht nur auf Folien existiert. Er stellt sicher, dass Sprint-Ziele präzise sind, Definition of Done mehr ist als eine Floskel und dass Backlog-Items in einer granulären Tiefe vorliegen, die Umsetzung erlaubt. Er coacht das Team im Refinement, damit Akzeptanzkriterien testbar, Abhängigkeiten sichtbar und Risiken benannt sind. Er sorgt dafür, dass das Sprint Planning Kapazität, historische Daten und Risikopuffer berücksichtigt. Und er verhindert, dass das Daily zur Statusrunde verkommt, indem er die Aufmerksamkeit auf Flow, Blocker und Re-Priorisierung lenkt.

Facilitation ist hier Präzision und nicht Smalltalk. Ein guter Scrum Master nutzt Liberating Structures, Lean Coffee, Fishbowl oder 1-2-4-All, um kollektive Intelligenz zu aktivieren und echten Fortschritt in Workshops zu erzeugen. Er achtet auf Timeboxing, klare Decision-Logics und das Festhalten von Commitments. In der Review werden nicht nur Features vorgeführt, sondern Hypothesen gegen Outcomes geprüft. In der Retro werden keine Klagemauern aufgestellt, sondern Ursachenanalysen durchgeführt – Five Whys, Ishikawa-Diagramme, Fault Tree Analysis, je nach Problem. So entsteht Lernfähigkeit.

Artefakte sind kein Deko-Regal, sondern Steuerinstrumente. Product Backlog, Sprint Backlog und Increment sind nur die Spitze; die eigentliche Arbeit liegt in Definitionen, Policies und Visualisierungen. Ein Scrum Master etabliert Service-Level-Erwartungen für Tickets, klare Kriterien für “Ready” und “Done” und eine workflow-orientierte Board-Struktur, die reale Zustände abbildet. Er führt WIP-Limits ein, verfolgt Aging WIP und macht Blocker sichtbar. Er nutzt Cumulative Flow Diagrams, um Engpässe messbar zu identifizieren, statt über “Gefühl” zu diskutieren. Wer Artefakte so denkt, lenkt nicht Tasks, sondern Fluss.

Empirie ohne Metriken ist Wunschdenken. Deshalb bringt der Scrum Master Kennzahlen wie Throughput, Cycle Time und Lead Time in den Alltag. Er erklärt Little’s Law, wenn jemand wieder 20 Tickets parallel startet. Er nutzt Monte-Carlo-Forecasting auf Basis historischer Durchsatzdaten, um Planung als Wahrscheinlichkeiten zu kommunizieren statt als Illusion von Sicherheit. Er arbeitet mit Run-Charts, Control Charts und Percentiles, damit Fortschritt ohne Schönfärberei sichtbar wird. Und er baut eine Kultur, in der Daten nicht drohen, sondern helfen.

Technik trifft Führung: Metriken, Tools und Daten, die ein Scrum Master beherrschen muss

Ein Scrum Master ohne Daten ist blind, einer ohne Tool-Verständnis ist lahm. Jira oder Azure DevOps sind nicht “nur Tools”, sondern das operationale Gedächtnis des Teams. Der Scrum Master sorgt für saubere Workflows, eindeutige Statusdefinitionen und sinnvolle Felder, die echte Analysen ermöglichen. Er führt Policies für Pull statt Push ein, damit Arbeit vom System gezogen wird, wenn Kapazität frei ist. Er pflegt Boards so, dass sie die Realität abbilden: keine Fantasie-Spalten, keine “sonstiges”-Eimer, keine Status-Inflation. So entstehen Daten, die vertrauenswürdig sind.

Auf der Metrik-Seite braucht es mehr als Burn-Down-Romantik. Cumulative Flow Diagram zeigt Verteilung, Engpässe und Schlangenbildung. Cycle-Time-Scatterplots offenbaren Varianz, Ausreißer und Stabilität. Throughput-Histogramme liefern die Basis für Monte-Carlo-Forecasts, die dem Management endlich Wahrscheinlichkeiten statt Fixdaten geben. Aging-WIP-Tabellen decken schleichende Blockaden auf, bevor sie den Sprint zerfressen. Und ein einfaches Flow Efficiency Measurement macht sichtbar, wie viel Zeit Arbeit ruht statt fließt. Das ist die Sprache, in der Verbesserungen argumentierbar werden.

Technische Praxis ist kein optionaler Bonus. Ein Scrum Master muss CI/CD-Pipelines kennen, Feature-Toggles verstehen und die Auswirkungen von Branching-Strategien auf Integrationskosten erklären können. Trunk-Based Development ist kein Dogma, sondern ein Hebel, um Durchlaufzeiten zu verkürzen und Merge-Hölle zu vermeiden. Testautomatisierung reduziert manuelle Prüfkosten, Shift-Left senkt Defektkosten früh. Zusammen mit DevOps-Metriken wie Deployment Frequency, Lead Time for Changes, Change Failure Rate und MTTR entsteht ein Gesamtbild von Liefertauglichkeit. Wer das ignoriert, moderiert am Symptom.

Planung wird mit Daten erwachsen. Der Scrum Master ersetzt Schätz-Orakel durch empirische Forecasts, die historische Realität respektieren. Statt Story Points als Währung zu vergöttern, nutzt er Throughput und Cycle Time, um Lieferzeitfenster zu bestimmen. Er arbeitet mit Service-Klassen, um Dringlichkeit und Risiko bewusst zu steuern. Und er koppelt Ziele an Evidence-Based Management: Current Value, Unrealized Value, Time-to-Market und Ability to Innovate. So wird Führung messbar, ohne Kreativität zu ersticken.

Anti-Pattern entlarven: Was ein Scrum Master nicht ist – und wie man Schäden behebt

Ein Scrum Master ist kein PM mit Post-its, kein Task-Zuteiler und schon gar nicht der Hüter der Geschwindigkeit. Wer Teams mikromanagt, vernichtet Ownership und erzeugt Eskalationskultur. Ein häufiges Anti-Pattern ist das Status-Daily, in dem jeder dem Scrum Master berichtet, statt gemeinsam Flow-Probleme zu lösen. Ein anderes ist die Retro ohne Konsequenz, die sich im Kreis dreht, weil Entscheidungen und Experimente fehlen. Ebenfalls beliebt: Sprint Planning als Wünsch-dir-was, völlig entkoppelt von historischer Kapazität und Risiken. All das kostet Vertrauen, Fokus und Planbarkeit.

Auch die Überfrachtung mit Meetings ist ein Klassiker. Wenn ein Scrum Master in jedem Termin dabei ist, aber niemand Verantwortung übernimmt, ist nichts gewonnen. Stattdessen braucht es klare Delegation, explizite Verantwortlichkeiten und Entscheidungsprozesse. Ein weiteres Anti-Pattern: Scrum als Religion ohne Kontext. Nicht jedes Problem ist mit einer puristischen Event-Routine lösbar, und nicht jedes Team braucht identische Rituale. Ein professioneller Scrum Master passt Praktiken an Produkt, Domäne und Teamreife an, ohne das Fundament zu verlieren.

Die toxischste Variante: Velocity-Fetischismus. Velocity ist ein lokales Maß und leicht manipulierbar, daher als Zielzahl untauglich. Wer Velocity jagt, lädt das System mit Scheinproduktivität auf, während Lead Times explodieren und Qualität leidet. Besser ist es, auf Durchsatz, Stabilität und Vorhersagbarkeit zu steuern. Ebenso gefährlich: versteckte WIP-Erweiterungen durch “kleine Nebenaufgaben”, die niemand trackt. Ein Scrum Master muss diese Muster sichtbar machen und konsequent abstellen. Transparenz ist keine Option, sondern Pflicht.

Schadensbegrenzung gelingt mit Klartext und Systemwechsel. Raus mit Status-Dailys, rein mit Flow-Inspektion. Raus mit Wunschplanung, rein mit probabilistischen Commitments. Raus mit Scheintransparenz, rein mit echten Metriken. Dazu braucht es Konfliktfähigkeit, denn echte Veränderung kratzt an Komfortzonen. Doch genau hier trennt sich die Moderation von der Führung: Der Scrum Master schützt Prinzipien, nicht Befindlichkeiten. Und er beharrt auf Ergebnissen, nicht auf Ritualen.

Skalierung ohne Illusionen: Scrum Master in SAFe, LeSS,

Nexus und realen Abhängigkeiten

Skalierung ist dort nötig, wo ein Produkt mehr Teams braucht, nicht weil das Organigramm es hübsch findet. Ein Scrum Master in skalierten Umgebungen kämpft nicht gegen Menschen, sondern gegen Abhängigkeiten, Schnittstellen und inkonsistente Prioritäten. SAFe, LeSS oder Nexus liefern Muster, aber kein Wunder. Sie helfen, wenn Wertströme klar geschnitten, Architekturgrenzen sinnvoll und Team-Topologien bewusst gewählt sind. Ohne produktgeschnittene Teams und klare Ownership entstehen lediglich Meeting-Orchester und Reporting-Feuerwerke.

Der Scrum Master adressiert Abhängigkeiten systematisch. Dependency-Mapping, Integration-Cadence, Contract-Tests und Enabler-Work sind Pflicht, keine Kür. Gemeinsame Definition of Done über Teams hinweg, integrierte Branch-Strategie und automatisierte Integrationsumgebungen sind Voraussetzungen, damit Sprints nicht im Merge-Stau sterben. Ein Multi-Team-Review zeigt, was als Produkt-Increment wirklich existiert, und entlarvt Insel-Erfolge. Und wenn die Organisation auf Projekt-Budgets statt Produkt-Funding besteht, macht der Scrum Master die Kosten dieser Struktur sichtbar.

Skalierungs-Events sind nur so gut wie die Daten, die sie füttern. Ein Program Increment Planning ohne belastbare Durchsatz-Historie und ohne Monte-Carlo-Simulation ist eine politische Wunschliste. Ein Scrum Master bringt realistische Szenarien, Pufferlogiken und explizite Risiken ins Spiel. Er schützt Teams vor Übercommitment und verhandelt Kapazitäten, bevor sie in PowerPoint eingezeichnet werden. Er etabliert Objectives, die Outcome-orientiert sind, nicht Output-verliebt. Und er führt Stop-Start-Continue-Rituale ein, die PI für PI einen echten Lernzyklus ermöglichen.

Skalierung ohne DevOps ist Staffage. Wenn Deployment-Pipelines fragil, Testumgebungen rar und Feature-Toggles selten sind, skalierst du Bürokram, nicht Produktlieferung. Der Scrum Master kooperiert mit Architekten, Betriebs- und Sicherheitsteams, um technische Voraussetzungen zu befestigen. Er fördert Plattform-Teams, klare Schnittstellen und innere Open-Source-Praktiken. Und er sorgt dafür, dass Architekturentscheidungen die Flussfähigkeit erhöhen, nicht Meetings. Erst dann lohnt sich jede weitere Schicht skalierter Koordination.

Playbook für Scrum Master: in 12 Schritten zu messbarem

Impact

Gute Absichten sind nett, ein Playbook ist besser. Der folgende Ablauf liefert eine erprobte Sequenz, die aus Zerstreuung Planbarkeit macht. Er ist bewusst pragmatisch, datengestützt und konfliktfest. Wenn du die Schritte konsequent umsetzt, entsteht ein System, das liefert, lernt und skaliert. Kein Theater, kein Buzzword-Bingo, sondern Handwerk in klaren Etappen. Los geht’s, Schritt für Schritt und ohne Ausreden.

- 1. System scannen: Value Stream Mapping vom Konzept bis zur Produktion erstellen, Wartezeiten, Handovers und Engpässe markieren.
- 2. Board auf Realität trimmen: Workflow-Stufen definieren, WIP-Limits setzen, Blocker-Policy formulieren, Aging WIP täglich prüfen.
- 3. Datenbasis säubern: Historische Tickets aufräumen, Status konsolidieren, Felder normieren, Definitionen dokumentieren.
- 4. Metriken einführen: Throughput, Cycle Time (mit 50./85./95.-Perzentil), CFD und Flow Efficiency visualisieren und wöchentlich reviewen.
- 5. Forecasting umstellen: Monte-Carlo-Forecasts für Releases und Epics etablieren, Commitments probabilistisch kommunizieren.
- 6. Qualität absichern: DoD mit Testautomatisierung, Code-Review-Policy, Security-Checks und Deploy-Readiness vervollständigen.
- 7. Refinement professionalisieren: Akzeptanzkriterien testbar machen, Abhängigkeiten explizit tracken, kleine schneidbare Slices fördern.
- 8. Meetings schärfen: Timeboxing rigoros, Entscheidungslog führen, Daily auf Flow und Blocker fokussieren, Retro auf Ursachenarbeit ausrichten.
- 9. DevOps verzahnen: CI/CD-Hygiene pushen, Trunk-Based Development fördern, Feature-Toggles und Dark Launches ermöglichen.
- 10. Stakeholder führen: Review auf Outcomes ausrichten, Metriken erklären, Erwartungsmanagement datenbasiert etablieren.
- 11. Risiken managen: Service-Klassen definieren, Expedite-Regeln selten und explizit, technische Schulden sichtbar und begrenzt abbauen.
- 12. Lernen verankern: Verbesserungs-Experimente mit Hypothese, Messkriterium und Zeitrahmen; Impact monatlich validieren.

Dieses Playbook lebt von Konsequenz, nicht von Perfektion. Die Reihenfolge ist bewusst gewählt, weil Visualisierung und Daten Voraussetzung für Forecasts und sauberes Stakeholder-Management sind. Viele Teams stolpern, weil sie über Meetings nachdenken, bevor ihr Flow sichtbar ist. Ein Scrum Master hält die Reihenfolge, schützt die Metriken vor politischer Verformung und sorgt dafür, dass jede Veränderung gemessen wird. Wer so arbeitet, erzeugt Vertrauen, weil Prognosen mit der Zeit besser werden.

Widerstand kommt garantiert. Manche verwechseln Transparenz mit Kontrolle oder fürchten den Verlust von Alibi-Produktivität. Ein Scrum Master moderiert diese Spannungen mit ruhiger Hand und klarer Argumentation. Er zeigt, wie weniger WIP zu schnelleren Lieferungen führt, und wie kleine Slices das Risiko senken. Er beweist mit Daten, dass Fokus keine Mode ist, sondern harte Ökonomie. Und er sorgt dafür, dass die Organisation die Vorteile spürt: schnellere Feedback-Loops, verlässlichere Releases, weniger

Feuerwehreinsätze.

Skills, Zertifizierungen und Karriere: vom Anfänger zum Impact-Macher

Zertifikate sind Einlasskarten, keine Wirkungsgarantie. CSM, PSM, A-CSM, PSM II, ICP-ATF, ICP-ACC – all das ist gut, solange es Praxis trifft. Ein starker Scrum Master investiert in Facilitation-Techniken, Konfliktkompetenz und Statistik-Grundlagen. Er liest über Systemtheorie, Conway’s Law, Theory of Constraints und Organisationsdesign. Er trainiert Visualisierung, Story-Slicing und Architektursensibilität. Und er bleibt neugierig auf Tools, die Daten besser nutzbar machen.

Karrierepfade sind vielfältig: Senior Scrum Master, Agile Coach, Delivery Lead, Head of Delivery, Product Operations. Entscheidend ist nicht der Titel, sondern der Beitrag zum Wertstrom. Wer Teams messbar stabilisiert, Risiken reduziert und Produkt-Outcome verbessert, wird gebraucht. Wer in skalierten Umgebungen Abhängigkeiten neutralisiert und mit Technikteams auf Augenhöhe arbeitet, wird unverzichtbar. Und wer das alles transparent vermittelt, erzeugt Sponsorship auf Führungsebene.

Soft Skills sind Hard Skills im Tarnanzug. Aktives Zuhören, Nonviolent Communication, die Ladder of Inference – das klingt nach Psychologie, ist aber Produktionsmittel. Denn ohne klares Denken und sauberes Sprechen gehen Meetings in Nebel unter. Ein Scrum Master schafft sprachliche Präzision, macht implizite Annahmen explizit und baut Brücken zwischen Fach, Technik und Management. Er übersetzt Metriken in Entscheidungen, nicht in Schuldzuweisungen.

Am Ende zählt Lernfähigkeit. Die besten Scrum Master sind Studenten des Systems, nicht Prediger eines Frameworks. Sie prüfen Praktiken regelmäßig, werfen Überflüssiges über Bord und schärfen dort nach, wo es wirkt. Sie sind unbequem, wenn es nötig ist, und dienend, wenn es möglich ist. Und sie haben eine Allergie gegen Theater: Kein Ritual darf bleiben, wenn es keinen Wert liefert. Das ist Karriere in diesem Feld: weniger Glanz, mehr Ergebnis.

Fazit: Scrum Master als echte Führungskraft in agilen Organisationen

Ein Scrum Master ist weit mehr als Taktgeber für Meetings. Er ist die Führungskraft, die Systeme baut, Metriken etabliert und Blockaden beseitigt, damit Produktteams zuverlässig liefern. Wer Teamführung agil und erwachsen

denkt, investiert in Flow, Daten und DevOps-Kopplung. Wer an Ritualen festklebt, produziert Scheinbewegung. Die harte Wahrheit: Ohne einen starken Scrum Master kippt Agilität schnell in Chaos oder Bürokratie. Mit ihm wird sie ein Wettbewerbsvorteil, der sich in Durchlaufzeit, Qualität und Vorhersagbarkeit messen lässt.

Wenn du diese Rolle ernst nimmst, wirst du Polarisierung erleben – und Wirkung. Du wirst mit Stakeholdern verhandeln, mit Entwicklern experimentieren und mit Managern Daten sprechen. Du wirst Spielregeln ändern, nicht Menschen umerziehen. Und du wirst feststellen, dass echte Führung nicht laut ist, sondern konsequent. Das Ergebnis ist ein System, das liefert. Genau darum geht es. Alles andere ist Theaterbeleuchtung.