PELARON

SEGELN · WISSEN · PRAXIS

ENTWICKLERTAGEBUCH · GESAMTAUSGABE

Vom Refit-Notizbuch zum digitalen Co-Skipper

Wie aus einem echten Problem an Bord eine Offline-first-Plattform für Fahrtensegler wurde.

SVEN GRIESBACH

Stand 9. September 2026 · Teile 1 bis 36

DIE GESCHICHTE HINTER DER APP

Nicht am Whiteboard erfunden, sondern im Motorraum

PELARON begann nicht mit einem Geschäftsplan und auch nicht mit dem Vorsatz, noch eine Navigations-App zu bauen. Der erste Gedanke entstand während des umfangreichen Refits unserer Segelyacht Bluebaerry: Wie behält man an einem Boot den Überblick, wenn Arbeiten, Rechnungen, Fotos, Wartungsfristen, Ersatzteile und Erfahrungen über Tabellen, Notizen, Mailfächer und Papier verteilt sind?

> **Je mehr wir an Bluebaerry selbst machten, desto deutlicher wurde: Das eigentliche Problem war nicht fehlendes Wissen. Das Problem war, dass Wissen, Daten und Erinnerungen an zu vielen Orten lagen.**

Der Ausgangspunkt von PELARON

Bluebaerry - eine Prospect 900 von Ridgeway Marine, Baujahr 1979, mit Volvo Penta MD7A - wurde dabei zum ersten echten Testboot. Beim Refit von Rumpf, Durchführungen, Elektrik, Solar, Fenstern, Luken, Segeln, Rigg und Motor zeigte sich, wie schnell eine Bootsverwaltung unübersichtlich wird. Zuerst gab es eine Tabelle, dann mehrere. Dazu kamen Handyfotos, Rechnungen im Postfach und ein Logbuch, das am Ende eines langen Tages oft leer blieb.

Wir probierten vorhandene Apps aus. Nicht ihr Funktionsumfang störte am meisten, sondern ihre Grundannahmen: Konto als Pflicht, Daten auf fremden Servern, steigende Abos, verschwindende Funktionen und Bedienwege, die sich nach Büroarbeit anfühlten. Daraus entstand ein einfaches Ziel: ein Bordbüro, das zuerst auf dem eigenen Gerät funktioniert und dessen Cloud eine Option bleibt.

Hinweis zur Chronologie

Die frühen Gespräche und Entscheidungen werden in diesem Beitrag thematisch zusammengefasst. Ab V27 liegen konkrete Versionsarchive vor; von dort an ist die Entwicklung chronologisch und mit den jeweiligen Modulständen dokumentiert.

DER ERSTE PRODUKTKERN

Aus einem persönlichen Werkzeug wird ein System

Die erste brauchbare Fassung musste nicht alles können. Sie musste die Dinge verbinden, die an Bord täglich auseinanderfallen: Törnplanung, Logbuch, Wartung, Inventar, Dokumente, Crew, Kosten und Sicherheitsinformationen. Der entscheidende Schritt war deshalb nicht die nächste einzelne Funktion, sondern ein gemeinsamer Datenzusammenhang.

Vier Grundsätze standen schon früh fest:

- Offline zuerst: Logbuch, Wartung und Bordorganisation müssen auch ohne Empfang funktionieren. - Ohne Konto nutzbar: Wer PELARON öffnet, soll sofort beginnen können. - Keine erfundenen Live-Daten: Eine konfigurierte Quelle ist erst dann live, wenn wirklich ein Messwert ankommt. - Einmal erfassen, mehrfach nutzen: Motorstunden, Kosten, Positionen und Ereignisse sollen in mehreren Modulen weiterwirken.

So entstand die Idee der digitalen Bootsakte und später des digitalen Co-Skippers. Das Logbuch sollte nicht nur Beweisführung sein. Ein Defekt unterwegs sollte zur Wartungsaufgabe werden, das verbrauchte Ersatzteil das Inventar verändern, die Reparatur in der Bordkasse erscheinen und die Erfahrung im Saisonrückblick erhalten bleiben.

> **PELARON sollte nicht die nächste Insel-App werden. Es sollte das digitale Gedächtnis des Bootes sein.**

Der Funktionskern wuchs deshalb in sechs zusammenhängenden Bereichen:

STAND	SCHWERPUNKT	WAS SICH VERÄNDERTE
01	Fahrt	Törns, Etappen, GPS, Logbuch, Kartenansicht und später GPX/KML-Import.
02	Boot	Stammdaten, Motor, Anlagen, Tanks, Batterien, Inventar und Wartung.
03	Menschen	Crew, Rollen, Qualifikationen, Medizin und Notfallkontakte.
04	Sicherheit	Checklisten, Ausrüstung, Fristen, Ankeralarm und Notfallseite.
05	Organisation	Dokumente, Bordkasse, Einkauf, Backups und Auswertungen.
06	Verbindungen	Signal K, NMEA, Victron, Orca, optionale Cloud und Geräteabgleich.

V27 BIS V30

Das Logbuch wird professionell - und die App lernt, sich selbst zu prüfen

Mit V27 begann die nachvollziehbar archivierte Phase. Aus einem funktionalen Logbuch wurde ein Werkzeug, dessen Daten auch nach Änderungen und Updates belastbar bleiben sollten. Verantwortliche Personen, Änderungsgründe und Revisionen machten aus losen Einträgen eine Historie. Der Törnbericht ließ sich als PDF ausgeben; Sicherung und Wiederherstellung wurden Teil des Produkts.

STAND	SCHWERPUNKT	WAS SICH VERÄNDERTE
V27	Smart Logbook Professional	Vollsicherung, Wiederherstellung, Törn-/PDF-Bericht, Verantwortlichkeit, Änderungsgrund, Revisionshistorie, sichere Motor-/Segelzustände und CSV-Schutz.
V28	Stabilität & Diagnose	Zentrale Version, lokales Diagnoseprotokoll, Update-Prüfung, kontrollierter Service Worker, Offline-Fallback und Cache-Bereinigung.
V29	Health Dashboard	Selbstdiagnose für System, Offline-Status, Speicher, Datenintegrität und Performance; technische Details standardmäßig eingeklappt.
V30	Systemcockpit	Status nach aktuell, Hinweis und historisch; Diagnosefilter, lokale Wartung, persistente Untermenüs, Deep Links und Status-Badges.

Diese Phase war wichtiger, als sie auf den ersten Blick klingt. Eine PWA kann beim Nutzer alt aussehen, obwohl auf dem Server neue Dateien liegen. Lokale Daten dürfen durch Cache-Updates nicht verloren gehen. Und ein Fehler, der auf dem iPad auftritt, muss beschreibbar sein, ohne dass jemand Entwicklerwerkzeuge öffnen kann. Diagnose, Versionsanzeige und kontrollierte Updates wurden damit zu Produktfunktionen - nicht nur zu Entwicklerhilfen.

Was in dieser Phase bewusst getrennt wurde

Statische Prüfungen kontrollierten JavaScript-Syntax, Manifest, HTML-IDs, Cache und ZIP-Integrität. Reale Hardware, iPad-Safari, GPS, NMEA, Signal K und Live-Deployment blieben gesonderte Praxistests. Diese Grenze wurde in den Prüfberichten offen benannt.

V31 BIS V38

Vom Bordbüro zum herstellerunabhängigen Schiffsbetriebssystem

V31 markierte den größten konzeptionellen Sprung. Das Maritime Cockpit sollte nicht länger eine Sammlung von Modulen sein, sondern ein gemeinsames Lagebild: Was ist heute an Bord wichtig? Ist das Boot abfahrtsbereit? Welche Ressource wird knapp? Welche Wartung ist fällig? Die Antwort sollte aus den bereits vorhandenen Törn-, Logbuch-, Crew-, Wartungs-, Inventar- und Kostendaten entstehen.

STAND	SCHWERPUNKT	WAS SICH VERÄNDERTE
V31	Maritime Platform	Digitaler Bordcomputer mit Lagebild, Bereitschaft, Checklisten, Ressourcen und Bordchronik.
V32	Tankmanagement	Mehrere Kraftstoff- und Wassertanks, Kapazitäten, Arten, Warnschwellen, Summen und valide Füllstände.
V33	Gerätecenter	Datenquellen für Tanks und Batteriebänke: Signal K, NMEA 2000, Victron GX, MQTT, Modbus TCP, Bluetooth und manuell.
V34	Digitaler Zwilling	Bootsprofil, Motor, Grau-/Schwarzwasser, Energiekomponenten, Vollständigkeit und Adapter-Registry.
V35	Workflow Engine	Ereignis-Engine, zwölf Bordfähigkeiten, Regeln und Schnellabläufe für Törnstart, Tanken, Ankern und Motorstopp.
V36	Asset Management	Technische Stammdaten, Garantie, Wartung, Dokumente, Ersatzteile und Beziehungen als Wissensgraph.
V37	Marine Integration	Geräteübersicht, Live-Monitor, Messwertrekorder, Sensorkalibrierung, Diagnoseexport und vorbereitete Instrumente.
V38	Orca-Koexistenz	Orca bleibt Navigation; PELARON übernimmt Schiffsbetrieb. Parallelmodi, Konfliktregeln und gemeinsame Datenquelle ohne erfundene proprietäre API.

TECHNISCHE TIEFE, EHRLICHE GRENZEN

Nicht jede Idee gehört in eine reine Browser-App

Mit der Geräteintegration kam eine entscheidende Einsicht: Eine Browser-PWA darf ein lokales Bordnetz nicht beliebig scannen. Bluetooth, mDNS, NMEA 2000 oder Modbus benötigen je nach Umgebung Gateways, Server-Endpunkte oder eine native Ergänzung. PELARON sollte diese Grenze nicht kaschieren. Deshalb entstanden klare Zustände: konfiguriert, verbindend, live, offline oder Fehler. Live ist eine Quelle nur nach tatsächlich empfangenen Daten.

Ähnlich fiel die Entscheidung bei Orca. Statt eine nicht verifizierte proprietäre Schnittstelle vorzutäuschen, wurde Orca als führende Navigationsanwendung akzeptiert. PELARON sollte daneben laufen, seinen Zustand vor dem App-Wechsel sichern und gemeinsame Sensordaten über Signal K, NMEA oder ein Gateway nutzen. Das war keine Einschränkung der Vision, sondern ihre Schärfung.

> **PELARON ersetzt keine Karten- oder Autopilotlösung. Es verbindet den langfristigen Schiffsbetrieb, den andere Systeme nur in Teilen abdecken.**

Drei Architekturentscheidungen aus dieser Phase prägen die App bis heute:

- Signal K als bevorzugte Integrationsschicht, weil Offenheit wichtiger ist als die Bindung an einen Hersteller. - Digitaler Zwilling als gemeinsame Struktur, damit Boot, Tanks, Batterien, Motor und Anlagen nicht mehrfach gepflegt werden. - Ereignisse statt isolierter Formulare, damit ein Vorgang mehrere Module nachvollziehbar aktualisieren kann.

Der erste Komplexitätsalarm

Mit digitalem Zwilling, Gerätecenter, Workflows, Assets und Marine Integration wuchs die technische Stärke schneller als die Orientierung für Erstnutzer. Spätere Prüfungen zeigten deshalb: Die nächste Aufgabe war nicht noch mehr Funktion, sondern bessere Priorisierung, verständlichere Begriffe und kürzere Wege.

V39 BIS V45

Core, Adapter, Datensicherheit - und dann wieder: Was zählt heute?

Die nächste Stufe ordnete die gewachsene Plattform von innen. Ein zentraler Core, Rollen, ein Audit-Verlauf und eine herstellerunabhängige Adapter-Schicht sollten verhindern, dass jedes Modul seine eigene Wahrheit verwaltet. Gleichzeitig wurden Sicherung, Datenintegrität und eine optionale Synchronisation so ausgebaut, dass Offline-Nutzung das Fundament blieb.

STAND	SCHWERPUNKT	WAS SICH VERÄNDERTE
V39	PELARON Core	Snapshot-, Query-, Event- und Audit-Schnittstelle; Rollen für Eigner, Skipper, Crew, Technik/Werft und Gast.
V40	Adapter Layer	Registry und Signal-K-WebSocket-Adapter; Victron-Profil über Signal K; transparente Verbindungszustände.
V41	Orca & Companion	Gemeinsames Bordnetz, Parallelbetrieb und vorbereitetes Token-/Begleitgeräte-Modell ohne direkten API-Anspruch.
V42	Optionaler Sync	Datei-Vollsicherung, Wiederherstellung und optionales WebDAV für Nextcloud-/Synology-kompatible Server.
V43	Stability UI Layer	Null-sichere UI-Helfer, Selbsttest und Fehlerbehebung, damit optionale Elemente keinen App-Abbruch auslösen.
V44	Datenintegrität	Prüfsummen, Revisionen, Sicherungsvalidator, Plausibilitätsprüfung und datensparsamer Beta-/Supportbericht.
V45	Heute an Bord	Health Score, Kontextmodi und maximal vier priorisierte Empfehlungen statt weiterer Menüfülle.

V45 war bewusst eine Gegenbewegung. Die Plattform konnte inzwischen sehr viel, aber der Nutzer brauchte morgens keine Architekturübersicht. Er brauchte eine Antwort auf vier Fragen: Ist Versorgung okay? Ist Sicherheit vorbereitet? Ist Wartung fällig? Gibt es Dokumente mit Frist? Daraus entstand 'Heute an Bord' mit Kontexten wie unterwegs, vor Anker und im Hafen.

> **Ein guter Bordcomputer zeigt nicht alles gleichzeitig. Er zeigt zuerst das, was als Nächstes entschieden werden muss.**

V48 BIS V50

Von der Plattform zurück zum Menschen, der sie zum ersten Mal öffnet

Nach der Plattformphase änderte sich die Leitfrage. Nicht mehr 'Was kann PELARON noch?', sondern 'Was versteht jemand, der PELARON heute zum ersten Mal öffnet?' Wiederholte Prüfungen der App machten die Dimension sichtbar: 22 Ansichten, fünf Navigationsgruppen und rund 50 Reiter. Für Kenner war das leistungsfähig; für Neueinsteiger war es Sucharbeit.

STAND	SCHWERPUNKT	WAS SICH VERÄNDERTE
V48	Cloud & Zugriff	Optionales Konto, Crew-Einladungen, Rollen, Geräteverwaltung, Sicherheitsverlauf und verschlüsselte Sicherung. Lokaler Betrieb bleibt vollständig möglich.
V49.5	Erster Start	Auswahl zwischen eigenem Boot, gekennzeichneter Demo und vorhandener Sicherung; kein Pflichtschritt.
V49.6	Anlegeassistent	Hafen, Ankunftszeit und echte Wettervorhersage werden verbunden; ohne Daten keine erfundenen Werte.
V49.7	Wettermodelle	ECMWF, ICON und GFS im Vergleich, Konsens, Streuung, Welle und Wetterfenster über Open-Meteo.
V49.8	Fotos & Rückblick	Komprimierte Logbuchfotos in IndexedDB und Saisonrückblick aus echten Törn- und Logbuchdaten.
V49.9	Schnelllog & Backup	Häufige Ereignisse mit einem Tipp; nur noch der verschlüsselte Komplett-Sicherungsweg.
V50	Wegweiser	Aufgabensuche in Seglersprache und aufgabenorientierter Einstieg statt eines weiteren Einsteigermodus.

Besonders wichtig war die Entscheidung gegen ein bloßes Verstecken von Funktionen. Ein 'Einsteigermodus' hätte die Suche nur verlagert. Der Wegweiser übersetzt stattdessen Aufgaben in Ziele: Wer nach Ölwechsel, Geld, Rettungsinsel oder Hafen sucht, landet im passenden Modul - unabhängig davon, wie es intern heißt.

V50 BIS V54

Versprechen wurden zu echten Funktionen

Mehrere Ansichten trugen richtige Namen, konnten aber noch nicht leisten, was Nutzer darunter erwarteten. Die digitale Bootsakte verwaltete Textzeilen, aber keine Dateien. Der Assistent nahm Fragen an, beantwortete sie jedoch nicht. Sicherheits- und Analyse-Reiter zeigten Platzhalter. Diese Lücken wurden nicht umbenannt, sondern geschlossen.

STAND	SCHWERPUNKT	WAS SICH VERÄNDERTE
V50	Dokumente	Echte Dateien, Kategorien, Ablaufdaten, Warnungen und Verknüpfung zu Anlagen; Bilder komprimiert, PDFs unverändert.
V50	Assistenz	Sucht offline in App-Funktionen und online in PELARON-Wissensartikeln. Kein vorgetäuschtes Sprachmodell.
V50	Sicherheit & Analyse	Ausrüstung mit Seriennummern und Fristen sowie echte Kosten- und Motor-Auswertungen.
V50	Seewetter im Logbuch	Bewölkung in Achteln, Luftdruck und Sicht; vorbelegt, aber immer überschreibbar.
V51	Demo-Törn	Ein zusammenhängender Törn Warnemünde-Gedser mit Defekt, Reparatur, Inventar, Kosten und Saisonstatistik.
V51	Crew & Medizin	Crewakte, Wachen, Qualifikationen sowie lokaler Notfallzugriff auf medizinische Angaben ohne Konto und Netz.
V51/52	SOS & Mobile	UKW 16/DSC 70, 112 und zuständige Leitstelle; auf dem Telefon sind alle 22 Ansichten erreichbar.
V53	Konsistenz & Bedienung	Eine sichtbare Produktversion, eindeutige Navigation, größere kritische Trefferflächen, VoiceOver-Hinweise, korrekte Törnzustände und ein führender Bootsdatenbestand.
V54	Foto-/Dateiabgleich	Fotos und Dokumente auf Wunsch zwischen Geräten; keine stille Mobilfunkübertragung, medizinische Inhalte ausgeschlossen.

Der neue Notfalldialog erzeugt aus Bootsname, Rufzeichen, MMSI und Position einen vorbereiteten Funkspruch und unterscheidet zwischen PAN PAN und MAYDAY. Der Geräteabgleich verbindet heute unter anderem Logbuch, Törns, Wartung, Bordkasse und Inventar; Fotos und Dokumente werden nur nach transparenter Entscheidung übertragen.

Datenschutz als Produktlogik

Medizinische Angaben bleiben grundsätzlich lokal. Medizinische Logbucheinträge, medizinische Aufgaben und zugehörige Fotos sind vom Abgleich ausgeschlossen. Ein Cloud-Konto erweitert PELARON, ersetzt aber nicht den Offline-Betrieb.

PELARON.DE UND PRODUKTREIFE

Die Website musste dieselbe Geschichte einfacher erzählen

Parallel zur App entwickelte sich pelaron.de von einer Projektseite zur Wissens-, Praxis- und Werkzeugplattform für Fahrtensegler. Das Refit von Bluebaerry liefert die echte Praxis, Wissensbereiche erklären Technik und Sicherheit, Downloads bündeln Checklisten - und die App macht daraus ein nutzbares Bordbüro.

In mehreren Tiefenprüfungen wurden nicht nur Texte und Design betrachtet, sondern auch Übersichtlichkeit für Erstnutzer, Button- und Linkfunktionen, mobile Navigation, Verkaufsreife und die Frage, ob Website und App dasselbe Versprechen geben. Am 5. August 2026 waren auf der Website 46 HTML-Seiten und acht PDFs ohne gefundene 404- oder Alttext-Probleme erreichbar. Die Seite wurde als weitgehend veröffentlichungsreif bewertet; die App blieb bewusst eine offene Beta.

Die Positionierung wurde dabei präziser:

- Nicht 'noch eine Navigations-App', sondern Bordbüro, Betriebssystem und digitales Gedächtnis. - Nicht Cloud-first, sondern offline zuerst, Cloud freiwillig. - Nicht Theorie, sondern Wissen und Funktionen aus einem echten Refit- und Bordalltag. - Nicht 'alles kann alles', sondern sichtbare Grenzen und überprüfbare Aussagen.

Dein Bordbüro. Auch ohne Netz.

Die verdichtete Botschaft auf pelaron.de

Auch der Produktfilm folgte dieser Entwicklung. Das Drehbuch wuchs von einer reinen Funktionsschau zu einer nachvollziehbaren Bordreise mit Notfall, Fotos und Geräteabgleich. Die Produktion wurde bewusst zurückgestellt, bis der Technikbereich einfacher und der Erstnutzerweg klarer ist. Erst das Produkt schärfen, dann das Versprechen filmen.

MARKTBLICK

Breite allein ist kein Alleinstellungsmerkmal

Die Marktvergleiche mit Orca, Sentinel, PredictWind DataHub, Signal K, Victron VRM, Garmin ActiveCaptain und savvy navvy führten zu einer nüchternen Erkenntnis: In vielen Einzelfunktionen gibt es sehr starke Spezialisten. Orca ist bei Navigation und modernem Instrumentendesign führend, Victron bei Energie, Signal K bei offener Datenintegration, PredictWind bei Wetter und weltweiter Kommunikation, Garmin im eigenen Geräteökosystem. Sentinel liegt PELARON funktional am nächsten.

STAND	SCHWERPUNKT	WAS SICH VERÄNDERTE
Orca	Navigation	Karten, Routing, AIS und Instrumente. PELARON ergänzt langfristigen Schiffsbetrieb.
Sentinel	Monitoring	Stark mit eigener Hardware, Mobilfunk, Warnungen und Remote-Sensorik.
PredictWind	Wetter & Datenhub	Stärker bei Wetterrouting, Kommunikation, Tracking und NMEA-Datenhub.
Signal K	Offene Daten	Technische Plattform und bevorzugte Integrationsschicht unter PELARON.
Victron VRM	Energie	Tiefe Energieüberwachung; PELARON ordnet Energie in den Gesamtzustand ein.
Garmin	Ökosystem	Plotter, Karten, Routen, Updates und Community im Garmin-System.

PELARONs Differenzierung liegt daher nicht in einem einzelnen KI-Label und auch nicht allein in der Anzahl der Module. Sie liegt in der herstellerunabhängigen Verbindung des gesamten Schiffsbetriebs: Smart Logbook, Wartung, Assets, Tanks, Batterien, Crew, Dokumente, Kosten, Sicherheit, Workflows, Diagnose, Offline-Nutzung und optionale Synchronisation in einem gemeinsamen Datenzusammenhang.

Die strategische Konsequenz

PELARON kopiert keine Kartenengine von Orca oder Garmin. Es koexistiert mit Navigation und konzentriert sich auf den Bereich, der davor, daneben und Jahre danach wichtig bleibt: den Betrieb und die Geschichte des Bootes.

TESTEN, SCHEITERN, NACHBESSERN

Was die Prüfungen wirklich verändert haben

Die Entwicklung verlief nicht als gerade Linie. Online-Prüfungen, iPad-Aufnahmen und wiederholte Marktchecks zeigten immer wieder Abstände zwischen technischer Implementierung und Nutzerwahrnehmung. Ein Button konnte vorhanden sein und trotzdem nicht verständlich wirken. Ein Modul konnte korrekt rechnen und dennoch am falschen Ort liegen.

Zu den wichtigsten gefundenen Problemen gehörten:

- Uneinheitliche sichtbare Versionsstände aus App, Modul, Cache und Sync - später auf eine Produktversion zusammengeführt. - Doppelte aktive Navigation und widersprüchliche Überschriften - technisch klein, für Nutzer aber ein Vertrauensproblem. - Eine zu komplexe Technikstruktur - der Grund für die aktuelle Vereinfachungsrunde. - Hafensuche und externe Datenquellen, die unter realen Netzbedingungen ausfallen können - deshalb klare Offline-/Fehlerzustände. - Mobile Navigation, in der 17 von 22 Ansichten nicht erreichbar waren - mit dem 'Mehr'-Menü korrigiert. - Platzhalter, leere Reiter oder scheinbare Assistentenfunktionen - in V50/V51 durch echte Daten und nachvollziehbare Antworten ersetzt. - Alte unverschlüsselte Backup-Wege, die dem Datenschutzversprechen widersprachen - entfernt zugunsten einer verschlüsselten Komplettsicherung.

Die daraus abgeleitete Testregel ist einfach: PELARON wird nicht nur am Schreibtisch geprüft. Kritische Bedienungen müssen bei Bewegung, Außenlicht, mit nassen oder kalten Händen, mit Handschuhen und unter Zeitdruck funktionieren. Deshalb wurden wichtige Trefferflächen vergrößert, Status nicht nur über Farbe vermittelt und VoiceOver-Zustände ergänzt.

> **Wenn ein Wartungseintrag im Motorraum umständlich ist, hilft keine perfekte Architekturzeichnung.**

DIE STIMME HINTER DER CHRONIK

Sechsunddreißig Beiträge

Die bisherige Chronik ordnet Entscheidungen, Module und Versionsstände. Die folgenden sechsunddreißig Beiträge erzählen dieselbe Entwicklung aus der Perspektive dessen, der sie gefordert und geprüft hat: warum PELARON überhaupt begonnen wurde, was an einzelnen Tagen wirklich passiert ist, und woran sich die Zusammenarbeit zwischen Seemannschaft, Messung und Maschine tatsächlich entschieden hat.

Alle Beiträge sind zwischen dem 6. August und dem 9. September 2026 entstanden; die ersten wurden für diese Ausgabe neu geschrieben — nicht gekürzt, sondern aus der ersten Person erzählt. Die Zahlen darin sind gemessen, nicht geschätzt. Wo eine Messung sich später als falsch herausgestellt hat, steht das im selben Beitrag.

> **Die Maschine baut schnell. Die Praxis entscheidet, ob das Gebaute an Bord wirklich stimmt.**

Was diese Wochen verbindet, ist ein einziger wiederkehrender Befund: Eine Regel gab es meistens schon — und die anderen Stellen wussten nichts davon. Siebzehnmal steht dieser Satz im Arbeitsheft, jedes Mal mit einem anderen Gegenstand davor. Die Antwort war jedes Mal dieselbe: eine Stelle, von dort geholt, ohne Rückfallfassung.

INHALT

Sechsunddreißig Beiträge

INHALT

Sechsunddreißig Beiträge · Fortsetzung

ORIGINALBEITRAG · TEIL 1

Der Wassersammler, den keiner kannte

August 2026

Die ehrliche Antwort auf die Frage, warum ich noch eine Logbuch-App baue, lautet: weil ich keine gefunden habe, die tut, was ich brauche.

Das ist ein schlechter Grund. Wer so anfängt, endet meistens mit einem Werkzeug, das genau für eine Person taugt. Trotzdem war es der Grund.

Wie es wirklich anfing

Nicht mit Software. Mit einem Impeller.

Ich wollte einen auf Lager haben — und Motoröl gleich dazu, damit im Frühjahr nicht das Suchen losgeht. Also saß ich zu Hause und versuchte herauszufinden, welcher es denn nun ist. Nicht am Boot. Das Boot lag anderswo, und die Angaben, die ich brauchte, lagen dort auch: auf einem Aufkleber, an einer Stelle, an die man nur mit dem Kopf über der Maschine kommt.

Das ging noch. Beim nächsten Teil ging es nicht mehr.

Der Wassersammler im Auspuffstrang. Ich brauchte den Durchmesser, und **keiner wusste ihn.** Nicht das Forum, nicht der Händler, nicht die Herstellerseite. Was ich fand, war ein Beitrag von 2014, ein Video ohne Angabe zum Bootstyp und eine Seite, die alles verspricht und nichts erklärt.

Ich hätte fahren müssen, um nachzumessen. Wegen einer Zahl.

Das war der Moment. Nicht irgendein Einfall am Schreibtisch, sondern die simple Erkenntnis, dass das Wissen über mein eigenes Boot nicht bei mir liegt, sondern am Boot — und dass es dort auch bleibt, wenn ich es woanders brauche.

Zwei Dinge, die daraus wurden

Daraus entstand zuerst **pelaron.de**: Wissen an einem Ort, aus echter Praxis, mit Kosten und Fehlern statt Hochglanz. Was beim Refit der *Bluebaerry* gelernt wurde — Seeventile, Elektrik, Solar, Rigg, Motor —, sollte nicht wieder verstreut liegen. Welcher Querschnitt bei welcher Länge. Was nach fünf Jahren wirklich passiert. Was es am Ende gekostet hat.

Der Zettel kam später. Auf ihm standen die Dinge, die mich beim Segeln selbst störten:

Ein Logbuch, das ich bei sechs Windstärken mit einer Hand führen kann. Ein Ort, an dem der Motorstundenstand steht, wenn die Werft danach fragt. Eine Wartungsübersicht, die weiß, dass ein Impeller nach Betriebsstunden fällig wird und nicht nach Kalendermonaten. Und die Gewissheit, dass das alles funktioniert, wenn zwanzig Seemeilen vor der Küste kein Netz mehr da ist.

Nichts davon ist neu. Aber die Apps, die ich kannte, konnten immer einen Teil — und der Rest lag wieder auf Papier.

Was eine Maschine kann und was nicht

Ich bin kein Softwareentwickler. Den Code schreibt eine künstliche Intelligenz, der ich beschreibe, was gebraucht wird.

Wer daraus schließt, so etwas gehe schnell und ohne Reibung, irrt. Es geht schnell. Ohne Reibung geht es nicht.

Die Maschine tippt in Minuten, wofür ich Wochen bräuchte. Sie kennt Formate, Schnittstellen und Verschlüsselungsverfahren aus dem Stand. Was sie nicht kennt, ist das Wasser. Sie weiß nicht, wie es ist, mit nassen Fingern zu tippen. Sie weiß nicht, dass ein Ankeralarm, der einmal ohne Grund losgeht, beim zweiten Mal ignoriert wird.

Und sie weiß vor allem nicht, was sie nicht weiß. Deshalb muss jemand danebenstehen und fragen: Stimmt das so?

Neun Versionen, und in fast jeder ein „das hatten wir falsch"

Zwischen dem Zettel und heute liegen neun Versionen in wenigen Wochen.

Eine davon hat im Wesentlichen einen einzigen Punkt: *Optionale Elemente verursachen keinen Programmabbruch mehr.* Auf Deutsch: Die App stürzte ab, wenn ein Bedienelement fehlte, das gar nicht gebraucht wurde. Eine ganze Version, nur um das abzustellen.

Dann kamen die Integritätsprüfung, die Tagesübersicht, das Verbindungszentrum, die verschlüsselten Sicherungen, die Zugriffsverwaltung für die Crew.

In fast jeder steht mindestens ein Punkt, der eigentlich heißt: **Das hatten wir vorher falsch.**

Narben im Quelltext

An einer Stelle im Programm steht ein Kommentar, den ich mag:

> **Selbst wenn ein nachgelagertes Modul einen Fehler wirft, werden gespeicherte Logbucheinträge gelesen und angezeigt.**

Dahinter steckt ein Tag, an dem das Logbuch leer blieb, obwohl die Einträge da waren. Irgendein Modul weiter hinten war gestolpert und hatte die ganze Anzeige mitgerissen. Seither gibt es dieses Netz — und den Kommentar, damit niemand ihn später für überflüssig hält und wieder entfernt.

Ein anderer Fehler ist mir besonders geblieben, weil er so unauffällig war: Fotos landeten am falschen Logbucheintrag. Neue Einträge werden hinten angehängt, gesucht wurde vorne.

Solche Fehler machen kein Geräusch. **Sie schreiben still das Falsche, und man merkt es Wochen später.**

Deshalb steht in vielen PELARON-Dateien inzwischen nicht nur, was der Code tut, sondern auch, was ein früherer Entwurf falsch gemacht hat. Das ist ungewöhnlich für Programmtexte. Ich halte es für das Nützlichste, was wir uns angewöhnt haben.

Was daraus geworden ist

Heute führt PELARON das Logbuch mit einem Fingertipp, rechnet den Verbrauch zwischen zwei Tankfüllungen aus, kennt neunzehn Reviere von der Nordsee bis Estland, zeichnet die gefahrene Strecke auf einer Karte, die auch ohne Netz funktioniert — und sagt, was sie nicht kann.

Der letzte Punkt ist mir der wichtigste. Wenn dort steht, dass der Ankeralarm keine Ankerwache ersetzt, oder dass echtes Wetterrouting ein Polardiagramm bräuchte, das die App nicht hat, dann steht es dort, weil beim Bauen jemand gefragt hat: Stimmt das so?

Den Durchmesser des Wassersammlers habe ich übrigens irgendwann selbst nachgemessen. Er steht jetzt in der App.

Im nächsten Teil geht es um zwei Tage im August, an denen dieses *Stimmt das so?* mehrfach nötig war — und um den Moment, in dem ich schrieb: **Nicht raten.**

PELARON entsteht an Bord der Bluebaerry und am Schreibtisch in Erfurt. Fragen, Widerspruch und Fehlermeldungen sind willkommen.

ORIGINALBEITRAG · TEIL 2

„Nicht raten"

6. August 2026

„Das ist die Ursache."

Diesen Satz habe ich in zwei Tagen dreimal gehört. Zweimal war er falsch.

Der Abend, an dem die Karte grau blieb

Mittwochnachmittag. Die Törnkarte zeigte die gefahrene Strecke als blaue Linie, die Zoomknöpfe gingen, unten stand ordentlich „© OpenStreetMap-Mitwirkende".

Nur die Karte selbst fehlte. Graue Fläche, nichts darauf.

Die erste Erklärung kam nach zwei Minuten und klang überzeugend: eine Sicherheitsregel des Servers, die das Laden der Kartenkacheln verhindere. Ich bekam eine korrigierte Datei, lud sie hoch. Grau.

Die zweite kam nach zehn Minuten und klang ebenfalls überzeugend: eine zweite Sicherheitsregel aus einem übergeordneten Verzeichnis, die die erste überstimme. Neue Datei. Hochgeladen. Grau.

Die dritte betraf einen Zwischenspeicher, der am selben Nachmittag eingebaut worden war. Immerhin ein Fehler auf der richtigen Seite. Behoben hat es nichts.

Irgendwann habe ich geschrieben:

> **Langsam komme ich mir verarscht vor.**

Das war unhöflich. Es war auch berechtigt, und der Grund war nicht der Fehler.

Der Grund war, dass nicht gemacht wurde, was ich wollte — und dass jedes Mal etwas anderes herauskam. Ich hatte nicht um eine Erklärung gebeten. Ich hatte um eine Ursache gebeten. Bekommen habe ich drei Stunden lang plausible Geschichten, jede davon anders als die vorige, jede mit einer Datei zum Hochladen dran.

Wer dreimal hintereinander ein anderes Ergebnis bekommt, hat nicht ein schwieriges Problem. Er hat ein Verfahren, das rät.

Dreißig Sekunden

Also haben wir ein Prüfwerkzeug gebaut. Es fordert dieselbe Kachel auf vier verschiedene Arten an und schreibt auf, welche durchkommt.

Es lieferte die Antwort in **dreißig Sekunden.**

Die Ursache stand die ganze Zeit im Fehlerbericht der App. Sie war dreimal gesehen und dreimal als Nebenwirkung des eigenen Tests abgetan worden.

Kartenkacheln sind normalerweise Bildanfragen. Sobald aber ein Hintergrundprozess dazwischentritt, werden sie technisch zu etwas anderem, und die Sicherheitsregel behandelt sie nach anderen Maßstäben. Eine Zeile in einer Serverdatei. Ein halber Tag.

Seither steht das als Kommentar im Quelltext, damit es nicht verlorengeht:

> **Eine Meldung wegzuerklären, statt sie zu prüfen, führt dazu, dass man danach an der falschen Stelle weitersucht.**

Zwei Sätze, die keine Maschine weiß

Am nächsten Vormittag ging es um das Logbuch. Eingebaut war eine Regel, die seemännisch klang: Wer in einem Hafen angekommen ist, legt dort auch wieder ab. Der Zielhafen des letzten Eintrags wird also zum Ausgangshafen des nächsten.

Ich habe geschrieben:

> **Nein, es ist nicht seemännisch richtig. Wenn ich mich auf einem Törn von Rostock nach Vitte befinde, mache ich mindestens stündlich einen Logbucheintrag und will nicht ständig den Ausgangshafen ändern. Der ist und bleibt Rostock, bis ich in Vitte festgemacht habe.**

Zwei Sätze, und die Regel war widerlegt. Von und Nach gehören nicht zum einzelnen Eintrag, sondern zur Etappe. Zwischen Ablegen und Festmachen bleiben beide gleich, egal wie viele Stundeneinträge dazwischenliegen.

Das ist kein Programmierfehler. Es ist ein **Wissensfehler** — und keine Textvorhersage der Welt schließt ihn, weil die Antwort nur jemand kennt, der es getan hat.

Meine nächste Frage war, ob es dafür einen eigenen Knopf braucht. Die Antwort war nein, und sie war gut: Die Etappe steht längst im Logbuch. „Ablegen" eröffnet sie, „Festgemacht" schließt sie. Man muss sie nur lesen.

Nebenbei entstand dabei etwas, das niemand geplant hatte — ein Knopf für den Stundeneintrag und die Möglichkeit, ihn automatisch setzen zu lassen. Mit dem ausdrücklichen Hinweis, dass das nur funktioniert, solange die App offen und der Bildschirm an ist. Ein Browser darf im Hintergrund nichts tun. Das ist eine Einschränkung, und sie steht dort, wo man sie liest, nicht im Kleingedruckten.

Der Satz, der geblieben ist

Am Nachmittag hakte es an einer Auswahlliste. Ich bekam eine Erklärung, eine Korrektur, eine zweite Erklärung, eine zweite Korrektur.

Beim vierten Anlauf habe ich geschrieben:

> **Wieder ändert sich der Reiter nicht nach Änderung, nicht raten!**

Daraufhin wurde eine Messanzeige eingebaut, die auf den Bildschirm schreibt, was in dem Feld tatsächlich steht. Danach war es eine Sache von Minuten: ein überflüssiges Ereignis, ausgelöst an einer Stelle, an der es nichts zu melden gab — und dabei die eben getroffene Auswahl wieder verworfen.

Dasselbe Muster wie am Vortag mit der Karte. Vier Anläufe raten, dreißig Sekunden messen.

Die Messanzeige ist inzwischen wieder ausgebaut. Geblieben sind drei Werkzeuge, die es vorher nicht gab:

– eine Übersicht, welche Programmteile tatsächlich geladen sind,

– Fehlermeldungen, die den Dateinamen mitliefern statt nur den Wortlaut,

– eine Prüfung der externen Dienste, die zwischen „die App funktioniert" und „das Wetter kommt an" unterscheidet.

Alle drei sind entstanden, weil vorher zu lange im Dunkeln gesucht wurde.

Was sonst noch passiert ist

In der Aufzählung sieht es beeindruckend aus: eine zusammengeführte Wetteransicht statt zweier verwirrend ähnlicher, ein eigener Hafenbestand für neunzehn Reviere von der Nordsee bis Estland, Import und Export in fünf Formaten, ein neu geordnetes Eingabefenster, einundzwanzig neue Symbole, Kartenkacheln, die auch ohne Netz funktionieren.

So war es aber nicht. Es war ein Wechsel aus Bauen, Prüfen, Verwerfen und Nachbessern — und die Hälfte der Zeit ging für die drei Stellen drauf, an denen geraten statt gemessen wurde.

Was ich mitnehme

Die beiden Tage haben mir beigebracht, worauf ich seitdem bestehe, und es ist kein technischer Satz:

> **Eine Erklärung ist keine Ursache. Und wer bei jedem Anlauf ein anderes Ergebnis liefert, hat noch nicht angefangen zu messen.**

Das gilt für die Maschine. Es gilt aber genauso für mich, wenn ich am Boot stehe und meine, ich wüsste schon, woran es liegt.

Im nächsten Teil geht es um fünf Reiter, die nichts hielten — und um die Frage, was schlimmer ist: ein leeres Feld oder eine Beschriftung ohne Funktion.

ORIGINALBEITRAG · TEIL 3

Fünf Reiter, die nichts hielten

7. August 2026

„Digitale Bootsakte. Dateien, Fristen und Verknüpfungen zentral organisieren."

So stand es im Menü. Was das Modul konnte, war: eine Textzeile mit einem Erledigt-Haken.

Keine Datei. Kein Ablaufdatum. Keine Kategorie — obwohl fünf Reiter nach Kategorien benannt waren.

Was mich daran geärgert hat

Nicht der Fehler. Fehler passieren.

Was mich geärgert hat, war die Frage, **wie eine Maschine überhaupt etwas bauen kann, hinter dem keine Funktion steckt.** Die Anweisung war klar. Da stand nicht „mach irgendwas mit Dokumenten". Da stand, was das Modul können soll.

Gebaut wurde die Oberfläche. Fünf Reiter, saubere Beschriftungen, alles an seinem Platz. Und dahinter nichts.

Für mich heißt das: Der Maschine hat schlicht die Intelligenz gefehlt, weit genug zu denken. Sie hat gebaut, was sich beschreiben lässt — und nicht gemerkt, dass eine Beschriftung ohne Funktion schlimmer ist als gar keine Beschriftung.

Ich habe zwischendurch überlegt, ob ich es selbst so bestellt und nicht gemerkt habe. Habe ich nicht. Ich habe nachgesehen.

Und es war nicht die einzige Stelle

Beim Durchgehen der App fand sich dasselbe an mehreren Stellen zugleich.

Im Hafenmodul standen zwei Bedienelemente ohne Wirkung: eine Auswahl zweier Marinas, die nichts auslöste, und ein Feld für die geplante Ankunftszeit, das nirgends ausgewertet wurde. Der Windpfeil für das Anlegemanöver ließ sich von Hand verstellen — ein Schieberegler ohne Bezug zur Wirklichkeit.

Auf den Wetterkarten für ECMWF, ICON und GFS stand: **„Professionelle API-Anbindung folgt."** Eine Ankündigung ohne Datum. Dabei liefert Open-Meteo alle drei Modelle frei, ohne Schlüssel.

Unter Sicherheit zeigten zwei Reiter Erklärtext statt Daten. Bei den Fristen stand „keine Ablaufdaten hinterlegt" — und der Satz konnte gar nicht verschwinden, weil es keinen Weg gab, welche einzutragen.

Unter Analysen hieß es „Auswertung folgt", obwohl die Kosten- und Motordaten längst vorlagen. Es fehlte nur die Rechnung.

> **Ein leeres Feld ist ehrlich. Eine Beschriftung ohne Funktion dahinter ist es nicht.**

Wie so etwas entsteht

Niemand hat das absichtlich gebaut. Es entsteht in der Reihenfolge, in der gearbeitet wird: Zuerst steht die Ansicht, dann die Reiter, dann die Beschriftungen — und die Funktion kommt danach.

Nur kommt sie manchmal nicht, weil inzwischen etwas anderes wichtiger wurde. Zurück bleibt eine Oberfläche, die vollständig aussieht.

Das erklärt es. Es entschuldigt es nicht. Wer eine Überschrift setzt, hat damit ein Versprechen abgegeben — und wer das nicht mitdenkt, denkt nicht weit genug.

Zwei Wege, und nur einer taugt

Man kann so etwas auf zwei Arten auflösen.

Entweder man benennt um, bis die Beschriftung zur Funktion passt — aus „Bootsakte" wird „Notizen", und alles stimmt wieder. Oder man baut, was die Beschriftung verspricht.

Der erste Weg ist schneller und macht die App kleiner. Ich habe den zweiten genommen, überall dort, wo die Beschriftung das Richtige versprach.

Warum am Ende Fotos dazukamen

Bei der Törnauswertung ist mir aufgefallen, dass aus solchen Ansichten selten etwas herauskommt, das man gerne ansieht. Ich glaube nicht, dass das an fehlenden Feldern liegt. Es liegt daran, dass am Ende nur Zahlen stehen.

Also Fotos. Und damit sofort drei Entscheidungen, die mit Fotos nichts zu tun haben:

Sie liegen in einem eigenen Speicher, weil der normale rund fünf Megabyte für alles zusammen fasst und ein einziges Handybild das sprengt. Jedes Bild wird auf 1600 Punkte längste Kante verkleinert; aus vier Megabyte werden meist unter vierhundert Kilobyte. Und sie werden **nicht** abgeglichen — ein Törn mit zweihundert Bildern wären mehrere Gigabyte, dafür ist weder der Server gedacht noch die Sicherung.

Das steht in der App. Nicht im Kleingedruckten.

Was ich mitnehme

Die Runde hat kaum neue Ideen gebracht. Sie hat eingelöst, was schon dastand.

Das fühlt sich beim Arbeiten weniger nach Fortschritt an als eine neue Ansicht. Es ist trotzdem mehr wert, weil eine Beschriftung ohne Funktion Vertrauen kostet, das man später an ganz anderer Stelle braucht.

Seither gibt es eine einfache Prüfung, bevor etwas eine Überschrift bekommt:

> **Kann ich das, was da steht, in der App vorführen? Wenn nein, steht es zu früh dort.**

Im nächsten Teil geht es um eine Anzeige, die „System betriebsbereit" meldete, während zwei Kernfunktionen ausgefallen waren.

ORIGINALBEITRAG · TEIL 4

„Betriebsbereit"

9. August 2026

Die Diagnose meldete: **System betriebsbereit.**

Gleichzeitig fand die Hafensuche nichts, und der Wettermodellvergleich blieb leer. Zwei Kernfunktionen ausgefallen, und die App sagte, es sei alles in Ordnung.

Warum die Anzeige nicht gelogen hat

Sie hatte recht — nur über etwas anderes, als draufstand.

Geprüft wurde ausschließlich, was im Gerät liegt: Sind die Programmteile geladen? Ist der Speicher lesbar? Läuft der Offline-Dienst? Ist das Logbuch unbeschädigt?

Alles davon war in Ordnung. **Kein einziger Onlinedienst wurde angefasst.**

Das ist verständlich, wenn man weiß, wie so eine Prüfung entsteht. Man baut zuerst, was leicht messbar ist. Ob eine Datei geladen wurde, weiß das Gerät sofort. Ob ein fremder Server antwortet, dauert Sekunden und kann aus zehn Gründen scheitern, die mit der eigenen App nichts zu tun haben.

Also bleibt es beim Naheliegenden — und die Überschrift wächst trotzdem mit.

> **Eine Anzeige, die „betriebsbereit" sagt, während zwei Kernfunktionen nicht arbeiten, kostet mehr Vertrauen, als sie stiftet.**

Die Lösung war nicht, mehr zu prüfen. Sie war, genauer zu benennen. Die alte Aussage heißt jetzt, was sie immer war: *Die lokalen Kernfunktionen laufen.* Daneben steht eine zweite Prüfung, die tatsächlich bei den Diensten anfragt — und die zwischen „die App funktioniert" und „das Wetter kommt an" unterscheidet.

Der Fehler, der seinen Namen verschweigt

Am selben Tag ist mir zweimal dasselbe passiert. Ein Programmteil scheiterte beim Laden und hinterließ eine Fehlermeldung — die aber nicht verriet, **welche Datei** betroffen war.

Also habe ich an der falschen Stelle weitergesucht. Zweimal.

Daraus entstand eine Übersicht, die schlicht auflistet, welche Programmteile sich angemeldet haben. Jedes Modul meldet sich beim Laden an; fehlt die Meldung, liegt die Datei entweder nicht auf dem Server oder ist beim Laden gescheitert.

Das ersetzt keine Fehlermeldung. Aber es beantwortet die erste Frage, und zwar in Sekunden statt in einer Stunde. Dazu die kleinere, wirksamere Änderung: Fehlermeldungen liefern jetzt den Dateinamen mit.

Dieselbe Falle, einen Tag später — von der anderen Seite

Ich wollte wissen, wie dieses Tagebuch auf pelaron.de eigentlich verlinkt ist, und habe die Seite abrufen lassen.

Das Ergebnis war eindeutig: kein Eintrag in der Navigation, keiner in der Fußzeile, von keiner Seite ein Weg hierher. Dreimal abgerufen, dreimal dasselbe.

Dann habe ich pelaron.de auf dem iPad geöffnet. Zwischen Refit und Downloads steht: **Tagebuch.** Seit Tagen.

Die Auflösung ist banal. Der Menüpunkt wird beim Laden der Seite eingefügt, nicht in vierzig einzelne Dateien eingetragen — eine bewusste Entscheidung gegen vierzig Gelegenheiten, es zu vergessen. Ein Abruf, der eine Seite nur herunterlädt, sieht diesen Schritt nicht. Er bekommt das Gerüst, bevor die Ergänzung passiert.

Der Abruf hatte also **korrekt gemessen.** Er hatte nur eine Seite gemessen, die so nie jemand zu sehen bekommt.

> **Ein Messwert ist nur so gut wie das Wissen darüber, was das Messgerät nicht sieht.**

Die dritte Regel

Aus dem ersten Teil stammt die Arbeitsteilung: Die Maschine baut schnell, ich korrigiere. Aus dem zweiten die Regel, erst zu messen und dann zu erklären.

Diese hier ist die Fortsetzung, eine Ebene tiefer.

Es reicht nicht, zu messen. Man muss wissen, was das Instrument grundsätzlich nicht erfassen kann — sonst wird aus einer blinden Stelle im Werkzeug ein Befund über die Welt.

Das ist kein Sonderfall der Softwareentwicklung. Ein Echolot, das den Grund nicht findet, meldet keine Tiefe. Ob das heißt „hier ist es tief" oder „der Geber sitzt falsch", entscheidet nicht das Gerät.

Wer ein Prüfergebnis bekommt, sollte deshalb zwei Fragen stellen. Die erste ist: **Stimmt das?** Die zweite: **Konnte das Werkzeug das überhaupt sehen?**

Die zweite wird seltener gestellt und ist häufiger die wichtigere.

Beide Male hat der Test am Ende weniger als eine Minute gedauert. Hinsehen, wo vorher nur gerechnet worden war.

Nachtrag, drei Wochen später: Dieser Teil beschreibt eine Anzeige, die „betriebsbereit" sagte, ohne etwas zu prüfen. Am 28. August habe ich eine Kachel gefunden, die seit jeher „Keine kritische Frist" meldete — sie fragte nach Feldern, die in der ganzen App nirgends geschrieben werden. Und eine zweite, die dauerhaft „Bereit" sagte, weil keine einzige Datei sie je beschrieben hat.

Dieselbe Gestalt, drei Wochen später, an zwei neuen Stellen. Es ist offenbar keine Kinderkrankheit.

ORIGINALBEITRAG · TEIL 5

Der leere Notfallbildschirm

9. August 2026

Ich bin beim Durchsehen zufällig draufgestoßen. Der Notfallbildschirm zeigte keine Leitstelle mehr.

Nicht einmal UKW Kanal 16.

Mein erster Gedanke war: **kann doch nicht wahr sein.**

Von allen Bildschirmen in dieser App ist das der eine, der funktionieren muss. Alles andere kann warten, kann falsch sein, kann einen Tag später repariert werden. Der nicht. Und ich habe ihn nur gesehen, weil ich gerade zufällig durchgeklickt habe.

Niemand hätte es gemerkt. Nicht bis zu dem Tag, an dem jemand ihn braucht.

Was dahintersteckte

Ein halb angewendeter Umbau. Eine Zeile griff auf etwas zu, das es nach der Änderung nicht mehr gab. Das ergibt einen Laufzeitfehler, und der ganze Bereich bleibt leer.

Eine Prüfung des Programmtexts findet so etwas nicht — die Schreibweise war ja korrekt. Es fällt erst beim Ausführen auf.

Der Fehler selbst war in Minuten behoben. Die Frage danach war die wichtigere:

Warum konnte eine einzelne kaputte Zeile den gesamten Notruf ausblenden?

Weil alles in derselben Rechnung hing. Die Ermittlung der zuständigen Seenotleitstelle aus Position, Revier oder Zielhafen ist der aufwendige Teil — und sie stand **vor** der Anzeige. Scheiterte sie, war auch das weg, was gar nicht berechnet werden muss.

Jetzt steht es umgekehrt

UKW Kanal 16 und DSC Kanal 70 stehen immer und zuerst da. Sie gelten überall und brauchen keine Berechnung.

Danach 112, mit dem Hinweis, dass es nur in Mobilfunkreichweite hilft und kein Handy ein Schiff in der Nähe erreicht. Erst dann die Leitstelle.

> **Was ohne Rechnung gilt, darf nicht von einer Rechnung abhängen.**

Aus derselben Überlegung stammt eine zweite Entscheidung. Wenn Position und Zielhafen auf verschiedene Zuständigkeitsgebiete deuten, zeigt die App **beide** Nummern und sagt dazu, dass die Angaben auseinandergehen.

Es wäre eleganter gewesen, sich für eine zu entscheiden. Im Notfall ist eine zweite Nummer besser als eine stillschweigend verworfene — und wer über UKW ruft, bekommt die zuständige Stelle ohnehin von selbst ans Mikrofon.

Siebzehn von zweiundzwanzig

Am selben Tag fiel etwas auf, das größer war als der leere Bildschirm.

Auf dem Telefon wird die Seitenleiste ausgeblendet. Übrig blieben fünf Knöpfe: Home, Log, Anker, Bord, Notfall.

Damit waren **siebzehn der zweiundzwanzig Ansichten nicht erreichbar** — Wartung, Wetter, Häfen, Crew, Dokumente, Bordkasse. Also fast alles, was jemand unterwegs braucht.

Dazu zwei kleinere Ärgernisse. „Bord" führte zum Inventar, was niemand vermutet. Und die Ankerwache belegte einen der fünf Plätze für etwas, das man selten braucht — auf einer Leiste, auf der jeder Platz zählt.

Die neue Aufteilung heißt Start, Logbuch, Wetter, Mehr, SOS. Die ersten drei sind das, was unterwegs am häufigsten gebraucht wird. „Mehr" öffnet alle übrigen, nach denselben Gruppen wie am großen Bildschirm. SOS bleibt eigenständig und rot.

Eine Entscheidung, die man der App nicht ansieht

Beim Umbau der Bordkasse stand die Frage an, wie Zusatzangaben zu Personen gespeichert werden. Die naheliegende Lösung hätte die bestehende Struktur verändert — und wer eine ältere Sicherung eingespielt hätte, wäre seine Namen los gewesen.

Also liegen die Zusatzangaben jetzt daneben und werden über den Namen verknüpft. Weniger elegant, bricht aber nichts.

Solche Entscheidungen sieht man der App nie an. Sie sind trotzdem der Unterschied zwischen einer Aktualisierung, die man einspielt, und einer, nach der man das Logbuch neu tippt.

Ein Beispieltörn, der etwas erzählt

Zuletzt eine Kleinigkeit, die mir wichtiger wurde als gedacht.

Das Demo-Boot legte bisher einen Törn, drei Crewnamen und drei Wartungsaufgaben an. Das zeigt, dass es Module gibt — nicht, was sie miteinander tun.

Jetzt erzählt es einen zusammenhängenden Törn: Abfahrtscheck, Ablegen in Warnemünde, 58 Seemeilen nach Gedser bei auffrischendem Wind, Hafengeld. Am zweiten Tag ein Impellerschaden — Vorkommnis im Logbuch, Reparatur, Ersatzteil aus dem Bordvorrat, Kosten in der Bordkasse, Serviceeintrag in der Wartung, am Ende alles im Saisonrückblick.

Genau diese Verkettung ist der Grund, warum PELARON überhaupt gebaut wurde. Ein Defekt unterwegs soll nicht in vier verschiedenen Listen landen, die niemand miteinander verbindet.

Eine Funktionsliste kann das nicht zeigen. Ein Törn schon.

Was ich mitnehme

Der leere Notfallbildschirm hat mir zwei Sachen beigebracht, und nur eine davon ist technisch.

Die technische steht oben: Was ohne Rechnung gilt, darf nicht von einer Rechnung abhängen.

Die andere ist unbequemer. **Ich habe ihn durch Zufall gefunden.** Es gab keine Prüfung, die ihn gefunden hätte, keinen Ablauf, der ihn regelmäßig aufruft. Es gab mich, der an dem Abend zufällig durchgeklickt hat.

Darauf kann man ein Bordbüro nicht bauen.

ORIGINALBEITRAG · TEIL 6

Ein Cent, zwei Tage

11. und 12. August 2026

Bernd schuldete 35,54 Euro. Die App sagte 35,53.

Der Cent war das kleinste Problem dieser zwei Tage.

Der Plan für die Woche war ein anderer. Der Produktfilm war fertig, die Seite dazu gebaut, und ich wollte die letzten Ungereimtheiten aus der App räumen, bevor die Beta breiter läuft. Neun Punkte hatte ich mir aufgeschrieben, nachdem ich einen Abend lang mit dem iPad durch alle Bereiche gegangen war.

Zwei Tage später waren sieben davon erledigt. **Die Hälfte der Zeit war für etwas draufgegangen, das mit der Liste nichts zu tun hatte.**

Die Rechnung, die dreitausendmal nicht aufging

Die Bordkasse rechnet die Ausgaben eines Törns auf die Crew um. Drei Personen, vier Ausgaben, 211,30 Euro. Bernd überweist 33,57 an den Skipper und 1,97 an Anna. Macht 35,54. Bernd schuldet aber 35,53.

Ein Cent. Bei drei Leuten und einem Wochenende ist das egal. Bei sechs Crewmitgliedern und zwanzig Ausgaben über eine Saison ist es kein Cent mehr — und dann sitzt man abends im Cockpit und rechnet nach, warum es nicht aufgeht. Genau das soll die App einem abnehmen.

Die Ursache war das Rechnen mit Kommazahlen: Jeder Anteil wird geteilt, jede Überweisung einzeln gerundet, und die Reste verschwinden nach oben.

Umgestellt auf ganze Cent. Der unteilbare Restcent geht reihum, damit über viele Ausgaben hinweg niemand systematisch draufzahlt.

Wie verbreitet der Fehler war, zeigte ein Testlauf über fünftausend zufällige Abrechnungen mit zwei bis sieben Personen: **In gut dreitausend Fällen ging die alte Rechnung nicht auf.** Bei der neuen in keinem.

Dazu zeigt die Abrechnung jetzt eine Bilanz — wer hat wie viel gezahlt, wie viel getragen, wie ist sein Saldo. Und unten die Summe, die null sein muss. Wenn nicht, wird die Zeile rot und nennt den Betrag.

Ich will nicht, dass die App eine Zahl behauptet, die man ihr glauben muss.

Der Nachmittag, an dem das Design weg war

Am Dienstag öffnete ich die App, und sie sah aus, als hätte jemand die Gestaltung gelöscht. Riesige Schrift, keine Seitenleiste, Elemente ohne Abstände.

Ein paar Stunden Arbeit dahin, und ich war entsprechend gut gelaunt.

Die Ursache: Die neue Formatierung für die Bordkasse war an eine **alte Fassung** meiner Stilvorlage angehängt worden — an einen Projektstand aus dem Juli, nicht an die Datei, die tatsächlich auf dem Server lag. Zwischen beiden liegen sechs Versionen und rund fünfundsiebzig Kilobyte. Alles, was seitdem dazugekommen war, war weg: Wetterumschaltung, Modellvergleich, Hafensuche, Modulübersicht, Einrichtungsdialog.

Das Ärgerliche war nicht der Fehler. Das Ärgerliche war, was danach passierte.

Weil der Einrichtungsdialog plötzlich unformatiert dastand, haben wir eine Stunde lang gesucht, warum für ihn nie eine Gestaltung geschrieben worden sei — und schließlich eine neue geschrieben.

Sie war die ganze Zeit vorhanden gewesen.

Wir haben ein Symptom behandelt, das wir selbst verursacht hatten.

Gerettet hat mich eine Sicherung von Sonntagabend. Bitgleich, alles wieder da.

Der Ordner war nicht der Server

Ich lade meine Dateien aus einem iCloud-Ordner hoch. Also war ich sicher, dass dort dasselbe liegt wie oben auf dem Server.

Tat es nicht. Bei zwei Dateien lag ein alter Stand, vier fehlten ganz.

Wir haben eine Stunde auf Grundlage einer Datei diskutiert, die mit meiner nichts zu tun hatte, und dabei Fehler „gefunden", die längst behoben waren. Die Bugrollenhöhe, ein fest eingetragener Bootsname, der Profilknopf — lauter Phantome aus einer Fassung von Ende Juli.

Das war der Moment.

Und der Grund war kein hehrer. Ich wollte die Arbeit einfach nicht mehrfach machen.

Ich hatte inzwischen mehrmals dieselben Dateien hochgeladen — und jedes Mal war die Maschine anderer Meinung darüber, was oben liegt. Man lädt hoch, bekommt gesagt, das könne nicht sein, lädt wieder hoch, sucht in einer Datei nach einem Fehler, den es dort längst nicht mehr gibt. Irgendwann ist das keine Fehlersuche mehr, sondern nur noch Beschäftigung.

Seitdem verlange ich Prüfsummen. Und ich lasse online nachsehen, ob tatsächlich alles oben steht, was oben stehen soll — nicht was ich hochgeladen zu haben glaube, nicht was im Ordner liegt.

Die Lösung war ein Zweizeiler im Terminal. Statt aus der Wolke hochzuladen, hole ich mir den Stand seither vom Server herunter: ein Befehl, vierundachtzig Dateien, 1,7 Megabyte. Damit ist ausgeschlossen, dass jemand auf einem Stand arbeitet, den es nicht gibt. Und ich habe nebenbei eine Sicherung, die dem Server wirklich entspricht.

Dazu die Angewohnheit, die mir seitdem den meisten Ärger erspart: Nach jeder Lieferung frage ich die Prüfsumme ab — eine Zahlenfolge, die sich ändert, sobald sich ein einziges Zeichen ändert. Stimmt sie, liegt genau die Datei oben, die liegen soll.

Drei Sekunden. Und sie ersetzt jedes „müsste eigentlich".

> **Bei der Hauptdatei hatten wir die Prüfsumme von Anfang an abgefragt. Dort ist nichts kaputtgegangen. Bei der Stilvorlage nicht — und genau dort ging es schief.**

Hundertsechzehn Seemeilen, die es nicht gibt

Der Beispieltörn führt von Warnemünde nach Gedser. Er stand mit achtundfünfzig Seemeilen je Schlag in den Daten, hundertsechzehn gesamt, gefahren in fünfundzwanzig Stunden.

Zwischen Hohe Düne und Gedser liegen fünfundzwanzig Seemeilen. Mit Hafenmanövern siebenundzwanzig.

Eine erfundene Zahl in den eigenen Beispieldaten. Wer die App zum ersten Mal öffnet, sieht genau das.

Die Karte, die von Schweden bis Libyen zog

Zum Saisonrückblick gehört eine Karte. Eine Linie ohne Küste sagt niemandem, wo man war — das ist etwas zum Aufheben, nicht nur zum Ablegen. Also liegen jetzt Kartenkacheln darunter, aus demselben Zwischenspeicher wie in der App.

Beim ersten Versuch zog die Karte von Schweden bis Libyen.

Einer meiner Testeinträge hatte keine GPS-Position, und im Feld stand eine Null. Null ist eine gültige Zahl, also wurde daraus **null Grad Nord, null Grad Ost** — ein Punkt im Golf von Guinea, den Programmierer „Null Island" nennen.

Die Prüfung fragte, ob es eine Zahl ist. Sie fragte nicht, ob es ein Ort sein kann.

Auch das ist niemandem aufgefallen, solange der Hintergrund grau war.

Nachtrag vom 28. August: Diese Prüfung wurde an dem Tag eingebaut und steht seitdem in einer Datei, mit einem langen Kommentar davor, der genau dieses Problem beschreibt — bis hin zu dem Satz, dass die Karte sich von Schweden bis Libyen aufzieht. **Sechzehn Tage später fand ich denselben Fehler an drei anderen Stellen wieder.** Eine davon war die GPX-Ausfuhr: Meine Saison hätte achtundzwanzig Punkte vor Westafrika enthalten, in einer Datei, die man weitergibt. Die Regel gab es. Drei Stellen kannten sie nicht.

Was mich am meisten gekostet hat

Nicht die Fehler in der App. Die waren in Minuten behoben, sobald klar war, wo sie sitzen.

Gekostet hat mich die Arbeit an Dateien, die nicht meine waren. Zwei Tage, in denen ich Fehler gemeldet habe, die es nicht mehr gab, und in denen eine Korrektur eine funktionierende Datei überschrieben hat.

> **Bevor du etwas änderst, stelle fest, was tatsächlich läuft. Nicht was du hochgeladen zu haben glaubst, nicht was im Ordner liegt, nicht was letzte Woche stimmte. Was jetzt auf dem Server liegt.**

Drei Sekunden für eine Prüfsumme. Ich habe zwei Tage gebraucht, um mir das anzugewöhnen.

ORIGINALBEITRAG · TEIL 7

Zwei Fehler, die keine waren

14. August 2026

Auf meiner Liste standen zwei Punkte als „Doppelung".

Hätte ich sie abgearbeitet, wären zwei Funktionen weg gewesen.

Nach den zwei Tagen aus Teil 6 stand eine Liste mit neun Punkten. Sieben davon habe ich an diesem Abend abgearbeitet, zwischen kurz vor sechs und kurz nach sieben. Achtzehn Änderungen sind es am Ende geworden, weil unterwegs Dinge auftauchten, die nicht auf der Liste standen.

Und zwei Punkte haben sich in Luft aufgelöst, weil sie Fehler beschrieben, die es nicht gab. Das ist der Teil, über den ich eigentlich schreiben will.

Drei Fälle, nicht zwei

Der Törnbericht sagt seit Teil 6, dass er Einträge ohne Törnzuordnung weglässt. Wenn er das sagt, muss man auch sehen können, welche das sind. Also gibt es jetzt eine Übersicht im Logbuch.

Beim Entwurf war mir wichtig, dass sie keine Fehler meldet, sondern **fragt.** Denn es sind drei Fälle, nicht zwei:

– Törn vergessen anzulegen.

– Müll.

– Und der berechtigte Hafeneintrag: Ölstand geprüft, Winterlager, Werftbesuch, Motor probegelaufen.

Zwischen zwei Saisons ist der dritte Fall die Mehrzahl aller Einträge, und er ist kein Mangel.

> **Eine Prüfung, die richtige Einträge dauerhaft rot markiert, schaltet man nach zwei Wochen ab. Und dann fehlt sie, wenn man sie braucht.**

Deshalb: Wer einen Zeitraum als Hafenzeit bestätigt, bekommt ihn nie wieder vorgesetzt.

Beim Testen kam ein Fall dazu, an den ich nicht gedacht hatte — Einträge, deren Zuordnung auf einen **gelöschten** Törn zeigt. Die sind heimtückischer als gar keine Zuordnung: Sie sehen zugeordnet aus und tauchen trotzdem in keinem Bericht auf.

Erst der eigene Bestand, dann das Netz

PELARON führt einen eigenen Hafenbestand. Jede erfolgreiche Suche wird behalten, ganze Reviere lassen sich vorab laden. Bei mir liegen dreiundneunzig Häfen drin.

Die Marinasuche in der Kartenansicht hat davon nichts gewusst. Sie fragte einen einzigen fremden Server, ohne Zeitgrenze, und meldete bei Ausfall: *„Suche derzeit nicht möglich. Internetverbindung oder Kartendienst prüfen."*

Für ein Werkzeug, das an Bord gebraucht wird, ist das die falsche Reihenfolge.

Jetzt kommt zuerst der eigene Bestand — sofort und ohne Netz —, danach das Netz mit drei Spiegeln als Ergänzung. Und jede erfolgreiche Suche füllt den Bestand weiter.

Beim Testen war der fremde Dienst tatsächlich nicht erreichbar. Statt der Sammelmeldung stand da, wie viele Häfen aus dem eigenen Bestand gefunden wurden. Das ist der Unterschied zwischen „geht nicht" und „geht, nur anders".

Der Bote statt der Ursache

Am selben Abend noch ein Fund, der mich geärgert hat, weil er hausgemacht war.

Die Marinasuche probiert jetzt drei Server statt einem. Jeder fehlgeschlagene Versuch wurde vom Hintergrunddienst in eine eigene Fehlermeldung verwandelt. Der Fehler wurde sauber gefangen, die App zeigte die richtige Meldung — aber in der Konsole stand nicht mehr die Ursache, sondern **der Bote.**

Wer einen Ausfall sucht, sucht dann am falschen Ort. Fremde Adressen gehen jetzt unverändert durch.

Vier Bausteine, die es ohne Netz nicht gab

Beim Nachrechnen der Liste, die festlegt, was für den Betrieb ohne Netz vorgehalten wird, kam heraus: **Vier Bausteine waren gar nicht offline verfügbar.** Der Dateiabgleich zwischen den Geräten, die Namensprüfung, die neue Zuordnungsprüfung und die Installationshilfe.

Mit Netz merkt man das nie. Ohne Netz fehlen sie.

Und „läuft ohne Netz" ist die Zusage, mit der ich werbe.

Einen der vier hatte ich zwei Stunden vorher selbst hinzugefügt und die Liste vergessen. Die anderen drei fehlten schon länger.

Achtzehn Änderungen, jede belegt

Was mir am Ende dieses Abends am meisten wert ist, sind nicht die achtzehn Änderungen.

Es ist, dass für jede eine Prüfsumme im Verlauf steht: vor der Änderung die des Servers, nach dem Hochladen die neue, dazu ein Wort, das nur in der neuen Fassung vorkommt.

Das ist der Ertrag aus dem Debakel vom Mittwoch. Es kostet zehn Sekunden je Datei und ersetzt jedes „müsste eigentlich".

Was ich mitnehme

Zwei Punkte auf einer Fehlerliste waren keine Fehler. Ich hatte zwei Dinge für dasselbe gehalten, weil sie ähnlich aussahen — und beim Ansehen waren es zwei verschiedene Aufgaben.

Das ist mir seitdem noch mehrfach passiert, immer in dieselbe Richtung: Ich sehe eine Doppelung und will aufräumen.

> **Bevor man etwas als überflüssig entfernt, sollte man wissen, wofür es da war. Eine Liste sagt das nicht. Nur das Hinsehen sagt es.**

ORIGINALBEITRAG · TEIL 8

Die Ölmenge, die ich fast erfunden hätte

17. August 2026, nachmittags

Eine gute Idee, zwei Stunden Arbeit, ein Test mit vier Motoren — und dann habe ich meinen eigenen Vorschlag wieder eingesammelt.

Warum ich das überhaupt wollte

Beim Ansehen der Motordaten fiel mir auf, dass Öltyp und Ölmenge leer waren.

Und ich wollte das nicht. **Ich wollte die Ölmenge vorbelegt haben, um dem Skipper Arbeit abzunehmen.** Wer im Frühjahr Öl kauft, soll nicht erst das Handbuch suchen müssen. Die App kennt den Motor — warum trägt sie nicht ein, was der Hersteller angibt?

Das war meine Idee, nicht die der Maschine. Das ist mir wichtig, weil ich sonst so klinge, als hätte ich etwas verhindert, das mir jemand untergeschoben hat.

Ich habe es gebaut. Eine Spanne aus der Motorleistung, wie sie bei kleinen Marinedieseln üblich ist. Zwei Stunden.

Dann habe ich es geprüft

Gegen vier reale Motoren:

```
MOTOR               RECHNUNG      TATSÄCHLICH
Yanmar 1GM10        1–1,5 l       ca. 1,1 l
Volvo Penta MD7A    1–1,5 l       ca. 2,5 l
Yanmar 3YM30        2,2–3,5 l     ca. 2,9 l
Volvo D2-75         5,5–8,8 l     ca. 5,5 l
```

Einer von vieren lag außerhalb. **Ausgerechnet der, der in meinem eigenen Boot steht.**

Kleine Motoren führen verhältnismäßig mehr Öl, als eine Rechnung nach Leistung hergibt.

> **Eine Spanne, die den wahren Wert nicht enthält, ist schlechter als keine. Wer anderthalb Liter kauft und zweieinhalb braucht, steht am Samstag vor geschlossenen Läden.**

Das Ärgerliche daran: Zwei Stunden vorher hatte ich selbst gegen Vorbelegungen argumentiert. Auf demselben Bildschirm steht ein Satz von mir über Berechnungen, die „mit angenommenen Werten arbeiten".

Und dann hätte ich fast angenommene Werte eingetragen.

Was daraus geworden ist

Geblieben ist etwas Bescheideneres — aber es ist **dieselbe Absicht**, nur ehrlich gebaut.

Im leeren Feld für die Viskosität steht jetzt ein Beispiel, weil eine falsche Viskositätsklasse nichts verschüttet. Für die Menge steht dort „laut Handbuch", und darunter ein Satz, der verschwindet, sobald man etwas einträgt:

> **Beides steht im Motorhandbuch. Einmal eingetragen, steht es im Frühjahr da — dann weißt du beim Ölkauf, was und wie viel, ohne wieder zu suchen.**

Das Ziel ist genau das, was ich am Anfang wollte: Der Skipper soll im Frühjahr nicht suchen müssen.

Nur behauptet die App nichts mehr. Sie erinnert daran, es einmal einzutragen.

Achtzig Seemeilen sind kein Suchradius

Am selben Nachmittag der zweite Fall derselben Sorte.

Auf der Hafenseite standen zwei Suchen nebeneinander. Links „Im Bestand suchen", rechts „Marina oder Ort". Seit dem Umbau vom Freitag durchsucht die rechte den eigenen Bestand ohnehin mit. Zwei Felder für dieselbe Sache — die linke Karte sollte weg.

Beim Nachsehen steckte in der linken Karte noch etwas anderes: **„Ausweichhäfen in der Nähe".** Das sieht wie dieselbe Suche aus und ist eine andere.

Die Marinasuche fragt fünf bis fünfzig Kilometer um GPS oder einen eingegebenen Ort. Das ist Planung am Kartentisch.

Die Ausweichhäfen fragen zwanzig bis achtzig **Seemeilen** um die aktuelle Position, ausschließlich aus dem eigenen Bestand, mit Peilung und Wetter für die ersten fünf. Das ist Reichweite, und es funktioniert ohne Netz.

> **Achtzig Seemeilen sind kein Suchradius zum Stöbern. Das ist die Entfernung, in der man einen Hafen sucht, wenn der Wind auf die Nase dreht.**

Die Karte ist weg, die Ausweichhäfen sind als dritter Knopf geblieben. Beim Test fand er fünfundzwanzig Häfen im Umkreis von vierzig Seemeilen und zeigte beim ersten gleich das Wetter — alles aus dem Bestand, ohne eine einzige Anfrage ins Netz.

Das ist jetzt das dritte Mal, dass ich zwei Dinge für eine Doppelung gehalten habe und beim Ansehen zwei verschiedene Aufgaben vorfand.

Wer die Bootsmaße eintragen will, braucht keine Legende

Unter „Mein Boot" standen zwei Karten ohne ein einziges Eingabefeld. Eine hieß „Zentrale Geräte- und Datenquellenverwaltung" und bestand aus einem Erklärabsatz und einer Legende: Live, Berechnet, Manuell, Geschätzt, Fehler. Die andere zeigte einen Vollständigkeitsbalken.

Beides beschreibt nicht das Boot, sondern woher seine Werte kommen. Das ist eine berechtigte Frage — aber nicht die, die man hat, wenn man Länge, Breite und Tiefgang eintragen will.

Beide sind nach „Erweitert" gewandert.

Dabei fielen zwei Wegangaben auf, die auf Orte zeigten, die es nicht gibt: „Boot & Stammdaten" und „Boot konfigurieren ? Antrieb". Beide Namen existieren in der App nicht.

Ein Hinweis, der auf einen erfundenen Ort zeigt, ist schlechter als keiner. Man sucht und findet nichts.

Was das Inventar nicht wusste

Aus der Ölüberlegung kam dann die eigentliche Entdeckung des Tages.

Öl und Ölmenge gehören zum Motor. Impeller, Filter und Anoden gehören ins Inventar — und stehen dort auch.

Nur wusste das Inventar nichts von der Wartung.

Der Serviceeintrag hat ein Textfeld mit dem Platzhalter „Öl und Ölfilter gewechselt, Impeller geprüft". Man trägt genau das ein, speichert — und der Ölfilter im Inventar steht weiterhin auf eins. Bis irgendwann jemand daran denkt, ins Inventar zu gehen und „minus eins" zu drücken.

Jetzt gibt es im Servicedialog einen Abschnitt „Verbrauchte Teile" mit allen Artikeln, die Bestand haben, je mit einem Mengenfeld. Beim Eintragen wird abgebucht. Fällt ein Artikel unter den Mindestbestand, landet er von selbst auf der Einkaufsliste.

Wartung erzeugt Verbrauch. Verbrauch erzeugt Bedarf. Das war immer der Sinn der Sache — es hat nur niemand verkabelt.

Was ich mitnehme

Der Nachmittag hat mir gezeigt, dass meine besten Ideen dieselbe Prüfung brauchen wie alles andere.

Die Ölmenge war meine Idee, aus einem richtigen Motiv, und sie hätte Schaden angerichtet. Vier Motoren nachschlagen hat zwanzig Minuten gedauert.

> **Eine gute Absicht ersetzt keine Messung. Und wer nur gegen die Fälle prüft, die ihm einfallen, prüft gegen sich selbst.**

Der Motor, der die Rechnung widerlegt hat, steht in meinem eigenen Boot. Wenn ich drei andere genommen hätte, wäre die Funktion heute drin.

ORIGINALBEITRAG · TEIL 9

Die Prüfsumme, die nichts gemessen hat

17. August 2026, abends

Achtundachtzig Zeitgeber, drei Sekunden Startzeit, ein Vorhaben, vor dem ich Respekt hatte.

Die Ursache stand am Ende in einer Zeile, die richtig aussah — und in einem einzigen fehlenden Wort.

Teil 8 endete am Nachmittag. Was dann kam, war als Aufräumen gedacht: die Restpunkte von der Liste, ein Tippfehler, ein Filter, der nicht greifen wollte.

Am Ende des Abends standen fünf Lieferungen, drei Irrtümer und eine Ursache, die in keiner einzigen Zeile Programmcode steckte.

Eine Zahl für nichts

Seit Teil 6 frage ich vor jeder Änderung ab, was tatsächlich auf dem Server liegt. Das ist die Regel, die mir am meisten Ärger erspart hat.

An diesem Abend sind mir im Terminal zwei Zeilen zusammengelaufen. Aus zwei Befehlen wurde einer, der Dateiname war Unsinn, der Abruf lief nie — **und trotzdem kam ein Ergebnis.** Eine ordentliche, zweiunddreißigstellige Zahlenfolge, wie sie immer kommt.

Nur war es die Prüfsumme von nichts. Die Rechenvorschrift läuft auch, wenn vorne nichts ankommt, und liefert brav einen Wert für die leere Eingabe.

> **Es gibt Messungen, die auch dann eine Zahl abgeben, wenn sie gescheitert sind. Die sind gefährlicher als gar keine.**

Die Abfrage schreibt jetzt erst in eine Datei und misst danach. Scheitert der Abruf, gibt es keine Zahl statt einer falschen. Und die eine Zahlenfolge, die für „da war nichts" steht, kenne ich inzwischen auswendig.

Eine halbe Stunde später ist mir derselbe Fehler in anderer Form passiert. Beim Abgleich meldete eine Datei eine Abweichung. Gemessen war beide Male derselbe Wert — falsch **abgetippt** hatte ich den Sollwert aus meiner eigenen Liste.

Seitdem vergleicht das der Rechner und nicht mehr ich.

Nachtrag vom 28. August: Dieser Satz oben ist an dem Tag dreimal wahr geworden. Ein Prüfling meldete nichts, weil er nichts messen konnte — und weil „nichts" auf beiden Seiten gleich aussieht, wäre es fast durchgegangen. Elf Tage später, dieselbe Falle, in einem anderen Werkzeug.

Fünf auf dem Bildschirm, zwei in der Datei

Auf der Liste stand seit dem 14. August eine Beobachtung: Der Datumsfilter im Logbuch stand in beiden Feldern auf demselben Tag und zeigte trotzdem alle Einträge.

Reproduzieren ließ sich das nicht. Der Filter ist richtig gebaut, die Felder werden beim Laden nicht vorbelegt, und weder Neuladen noch Zurückblättern stellt einen Wert wieder her. **Ich habe die Ursache offen gelassen, statt mir eine auszudenken.**

Beim Suchen fiel etwas anderes auf, und das war schlimmer. Wenn Feldwerte und angezeigte Liste auseinanderlaufen, zeigt der Bildschirm fünf Einträge — und der Export schreibt **zwei** in die Datei. Die Liste stammt aus dem letzten Zeichnen, der Export liest die Felder im Moment des Klicks.

Das ist dieselbe Klasse Fehler wie beim Törnbericht in Teil 6: Zu viele Zeilen in einer Datei sieht man beim Durchsehen. Zu wenige nicht.

Der Export zeichnet die Liste jetzt zuerst neu und schreibt danach. Bei Uneinigkeit gewinnen die Felder — die sind, was jemand eingestellt hat. Die veraltete Liste ist der Fehler.

„Letzter Törn — Keiner"

Ein Bildschirmfoto vom iPad zeigte im Systembereich eine Kachel. Überschrift: „Letzter Törn". Wert darunter: „Keiner".

Zwei Zeilen tiefer, in derselben Ansicht: „Törns · 1 Eintrag".

Die Ursache ist eine Naht zwischen zwei Änderungen, die einzeln beide richtig waren.

Auch das ist eine Gestalt, die wiederkommt. Am 28. August habe ich zwei Kacheln gefunden, die seit jeher dasselbe sagten — eine, weil sie nach Feldern fragte, die niemand schreibt, und eine, die überhaupt niemand je beschrieben hat.

Ein Wort, das seit Wochen fehlte

Der eigentliche Fund des Abends stand in keiner Programmdatei.

Die Startzeit der App lag bei drei Sekunden. Auf der Liste stand das als Brocken, den man nur als eigenes Vorhaben angehen sollte — achtundachtzig Zeitgeber umbauen, Ladereihenfolgen ändern, Bausteine zusammenfassen. Ich hatte Respekt davor.

Dann haben wir nachgesehen, was der Server tatsächlich ausliefert.

Die Serverkonfiguration listet auf, welche Dateitypen gepackt übertragen werden sollen. Für Programmdateien stand dort ein Name, den der Server seit ein paar Jahren nicht mehr verwendet. **Zwei Schreibweisen für dasselbe. Eine stand da, die andere nicht.**

Stilvorlagen wurden gepackt. Programmdateien nicht — und die sind vier Fünftel der Übertragung.

Ein Wort, in zwei Zeilen ergänzt. Danach kam die Hauptdatei mit sechsundfünfzigtausend statt hundertdreiundsiebzigtausend Byte an. **Siebenundsechzig Prozent weniger**, hochgerechnet über achthundert Kilobyte bei jedem ersten Start.

Auf einer Mobilfunkverbindung an Bord ist das genau der Unterschied, den ich mir vom Umbau der Zeitgeber erhofft hatte. Und dafür musste keine einzige Zahl in einem Baustein angefasst werden.

Was ich mitnehme

Der größte Anteil an der Ladezeit lag außerhalb des Codes. Bei jedem Umbau der Bausteine wäre er unentdeckt geblieben. Ich hätte an Zeitpunkten gedreht, hätte hinterher Fehler gesucht, die vorher keine waren — und wäre am Ende bei denselben Sekunden gelandet.

Vier Dinge sahen an diesem Abend nach dem aus, was sie nicht waren: ein Tippfehler, den es nicht mehr gab. Ein Filter, der funktionierte. Eine Prüfsumme, die keine war. Ein fehlender Block, der längst dastand.

Und dreimal haben wir uns geirrt, zweimal mit ziemlicher Sicherheit im Ton. Geholfen hat jedes Mal dasselbe: **nicht diskutieren, sondern eine Messung bauen, die auch dann etwas sagt, wenn sie einem widerspricht.**

> **Nachtrag zu Teil 8: Dort steht, bei der Software, die mich geärgert hat, habe offenbar niemand nachgerechnet. Heute würde ich ergänzen: Nachrechnen allein reicht nicht. Man muss auch nachrechnen, ob man das Richtige nachgerechnet hat.**

ORIGINALBEITRAG · TEIL 10

Der fremde Prüfbericht

19. August 2026

Jemand anders hat sich PELARON angesehen. Nicht ich, nicht die Maschine, die den Code schreibt — ein Prüfbericht von außen, zwanzig Punkte lang, mit Noten von 9,1 für die Website bis 7,0 für die Verkaufsreife.

Die Versuchung ist, so etwas von oben nach unten abzuarbeiten. Ich habe es anders gemacht und die Punkte zuerst danach sortiert, **ob sie stimmen.**

Das hat einen Tag gekostet und war jede Stunde wert.

Messwert 0,0

Der schwerste Befund stand in der Mitte des Berichts, zwischen zwei Formulierungsfragen.

Das Demo-Boot hat keine Tanks, keine Batteriebänke und keine Motorquelle. Trotzdem meldete die App:

> **Kraftstoffreserve prüfen · Messwert 0,0**

Der Zeitstempel war von heute, 19:13 Uhr. Kein alter Verlaufseintrag — die Regel feuerte gerade.

Die Ursache ist eine Zeile, und sie ist lehrreicher als der Fehler.

Die Funktion, die den Messwert liefert, macht **alles richtig**: Ist kein Tank eingerichtet, gibt sie ausdrücklich „kein Wert" zurück und nicht etwa null. Genau die Unterscheidung, die der Prüfbericht fordert, war eingebaut.

Die Auswertung hat sie unbrauchbar gemacht. Sie wandelt den Wert in eine Zahl um und prüft **danach**, ob es eine gültige Zahl ist. Nur ergibt die Umwandlung von „kein Wert" die Zahl null, und null ist gültig. Danach ist null kleiner als zwanzig, die Regel greift, und die Meldung schreibt „Messwert 0,0" — weil der Wert an dieser Stelle tatsächlich null ist.

Die Schutzabfrage stand da. Die Umwandlung davor hat sie ausgehebelt.

Das Bemerkenswerte: Bei einem fehlenden Feld hätte es funktioniert. Es scheitert ausgerechnet an dem Wert, den die richtige Seite absichtlich sendet, um zu sagen, dass sie nichts weiß.

Vier Zeilen. Vorher prüfen statt nachher.

Dreißig Sekunden

Der Prüfer suchte Häfen bei Rostock und wartete rund dreißig Sekunden, während nur „Marinas werden gesucht …" dastand.

Die Marinasuche fragt drei Kartendienst-Spiegel ab, jeden mit einer Zeitgrenze von zwanzig Sekunden. **Nacheinander.** Antwortet der erste nicht, wartet man seine volle Grenze ab, bevor überhaupt jemand anders gefragt wird. Im schlimmsten Fall dreimal zwanzig.

Und jetzt kommt der Teil, der mich beschäftigt hat.

Über der Abrufschicht in einer anderen Datei steht seit dem 14. August ein Kommentar von mir, der **genau davor warnt**: nacheinander summieren sich die Wartezeiten. Gemeint waren dort die Spiegel innerhalb einer Abfrage. Eine Ebene höher passierte dasselbe, in einer anderen Datei, unbemerkt.

Das ist, soweit ich sehe, das erste Mal, dass diese Gestalt in meinem Tagebuch steht: Eine Regel gab es schon — und eine andere Stelle kannte sie nicht. Am 28. August habe ich sie an einem einzigen Tag siebzehnmal wiedergefunden.

Jetzt laufen die drei gleichzeitig. Aus fünfundvierzig Sekunden im schlimmsten Fall werden fünfzehn, aus achtzehn werden drei.

Nebenbei fiel auf, dass die App nach zwanzig Sekunden auflegte, während die Abfrage selbst bis fünfundzwanzig rechnen durfte — wir haben also Antworten weggeworfen, auf die wir gewartet hatten.

Ein Fähranleger ist keine Marina

Unter den Treffern standen ein Fährterminal, ein namenloser Eintrag „Marina / Hafen" und mehrere Stege ohne erkennbaren Gastliegeplatz.

Das kommt aus der Abfrage. Sie holt drei Merkmale aus OpenStreetMap, und zwei davon erfassen jeden Hafen — auch einen Fähranleger, auch einen Umschlaghafen.

Wer als Segler einen Hafen sucht, sucht einen Liegeplatz. Nicht jede Stelle, an der ein Schiff festmachen kann.

Zwei Zeichen für dieselbe Marke

pelaron.de und pelaron.app trugen verschiedene Zeichen. Die Website die Wortmarke mit den zwei Bögen und den Segeln, die App ein quadratisches Symbol.

Erklärbar ist es — das Quadrat ist als Symbol für den Startbildschirm entstanden, wo eine Wortmarke nicht funktioniert, und dann in die Kopfzeile gewandert. Wann das passiert ist, weiß ich bis heute nicht.

Jetzt steht in beiden dasselbe Zeichen. Das Quadrat bleibt, wo es hingehört: Startbildschirm, Favicon, Kachel.

Was ich mitnehme

Zwanzig Befunde. Davon zwölf bestätigt, einer ernst, zwei bereits behoben, **einer selbst falsch**, zwei nicht nachvollziehbar.

Ein Tag Arbeit, um sie zu sortieren, bevor eine einzige Zeile geändert wurde.

> **Ein fremder Befund ist eine Beobachtung, keine Aufgabe. Vor der Aufnahme in die Liste steht die Frage, womit gemessen wurde.**

Denn ein Abruf ohne Skriptausführung sieht die halbe Seite nicht — genau der Fehler, der mir in Teil 4 selbst passiert ist, als das Tagebuch angeblich nicht verlinkt war. Eine Ladezeit vor einer Serveränderung ist keine Ladezeit danach: Die im Bericht genannten 4,9 Sekunden waren vor der Kompression gemessen, heute sind es 1159 Millisekunden auf dem iPad über Hotspot.

Und eine Angabe über mein eigenes Boot kann falsch sein, auch wenn vier andere davor stimmten.

Die beiden Korrekturen, die den Tag am meisten wert waren, standen in keinem Bericht: die Seemeilen und die Länge des Wetterfensters.

Beide kamen nicht vom Prüfen, sondern vom Segeln.

ORIGINALBEITRAG · TEIL 11

Ein Schönheitsfehler

20. August 2026

„Warum sind die Auswahlfelder nicht alle auf einer Höhe?"

Die Frage kam mittendrin, zwischen zwei Punkten aus dem Prüfbericht. Sie wurde beantwortet wie eine Nebensache — und genau das war der Fehler des Tages.

Was vorher lief

Der Vormittag war gut.

Die verschlüsselte Sicherung enthält keine Fotos und kann es nicht — sie liest den einen Speicher, die Bilder liegen im anderen. Das ist keine Nachlässigkeit, sondern eine Folge davon, wo Bilder liegen müssen: In den Speicher, den die Sicherung liest, passen rund fünf Megabyte für alles zusammen; ein einziges Handybild sprengt das.

Gefährlich ist daran nicht, dass die Fotos fehlen. **Gefährlich ist, dass es niemand merkt.** Wer zurückspielt, bekommt Logbuch, Wartung und Bordkasse wieder — und nichts sagt ihm, was nicht dabei war.

Dieselbe Klasse Fehler wie beim Törnbericht: Zu viel sieht man. Zu wenig nicht.

Jetzt wird beim Erstellen gezählt, was in den Bilddatenbanken liegt, und die Zahl wandert verschlüsselt in die Sicherung hinein. Das ist der Kniff: Beim Zurückspielen stammt der Hinweis **aus der Datei** und beschreibt das Gerät, auf dem sie entstand — nicht das leere, auf dem gerade wiederhergestellt wird.

Sonst meldete ausgerechnet das Gerät, dem alles fehlt, es fehle nichts.

Zwei weitere Punkte fielen an dem Vormittag: „1 Törns" heißt jetzt „1 Törn". Und der Filter „Törn" im Logbuch war leer, obwohl ein Törn vorhanden war — die Liste füllte sich nur beim Öffnen des Dialogs für einen neuen Eintrag.

Und dann die Felder

Fünf Felder in einer Zeile: Suchfeld, zwei Auswahllisten, zwei Datumsfelder. Die beiden Datumsfelder saßen höher als der Rest, ihre Beschriftungen klebten an der Karte darüber.

Dreimal wurde eine Ursache genannt, und dreimal lag sie daneben.

Beim ersten Mal fehlte angeblich eine Ausrichtungsregel — sie stand längst da, nachgesehen wurde in einer veralteten Kopie. Beim zweiten Mal sollte die Schreibweise der Regel schuld sein, zu neu für den Browser — der versteht sie seit fünf Jahren. Beim dritten Mal sollte ein Zusatz das Innere des Kastens nach unten schieben — er änderte nichts.

Zwischendurch wurde mir zweimal angeboten, den Punkt liegen zu lassen. Es sei ja nur die Optik.

Ich habe geschrieben:

> **Und das nächste Mal nicht Dinge nach hinten schieben, die Optik ist ebenso ein wichtiger Punkt.**

Der Satz sitzt, und er stimmt auch sachlich. Schief sitzende Felder beschädigen keine Daten. Sie sind der erste Eindruck. Wer die App öffnet und die Filterzeile sieht, schließt daraus auf den Rest — zu Recht, denn ich beanspruche Sorgfalt bis auf den Cent.

Bei der Konkurrenz für achtzig Euro im Jahr sitzt jedes Feld gerade.

Woran es wirklich lag

Als endlich statt zu raten die Hilfslinien des Browsers eingeschaltet wurden, war es in **dreißig Sekunden** klar.

Alle fünf Felder lagen in einer Zeile, unten bündig. Die Zeile war richtig ausgerichtet. Schief saß nur das Innere der beiden Datumsfelder.

Der Grund: Diese beiden waren anders gebaut als ihre Nachbarn. Sie hatten eine sichtbare Beschriftung über dem Feld, die anderen drei gar keine. Zwei Bauformen nebeneinander, und die eine dehnte sich anders als die andere.

Die Lösung war deshalb nicht, weiter zu flicken, sondern beide Bauformen anzugleichen.

Eine Behauptung, von der ich wusste

Zum Schluss noch etwas, das mich mehr geärgert hat als die Felder.

Auf der App-Seite stand unter „Verschlüsselte Sicherung": *Der komplette Datenstand lässt sich herunterladen.*

Das stimmt nicht, **und ich weiß das seit dem 12. August.** In der App ist die Beschriftung seither ehrlich. Auf der Seite, mit der ich für die App werbe, stand die alte Behauptung noch — zwei Abschnitte über dem Kapitel, das erklärt, wofür man PELARON nicht verwenden darf.

Dieselbe Gestalt wie überall: Die richtige Auskunft war da. Nur nicht dort, wo man sie liest.

Was ich mitnehme

Bei einer falschen Zahl wäre sofort ein Prüfwerkzeug gebaut worden. Bei zwei schiefen Feldern wurde aus der Ferne geraten — und der Bootseigner als Messgerät benutzt.

> **Was der Nutzer aufbringt, wird abgearbeitet, nicht bewertet. Und ein Darstellungsfehler bekommt dieselbe Messung wie ein Rechenfehler.**

Der zweite Teil ist der eigentliche. Es wurde nicht dreimal danebengelegen, weil die Ursache schwer war — sie war in dreißig Sekunden sichtbar, sobald das richtige Werkzeug lief.

Es wurde danebengelegen, weil die Sache für zu klein gehalten wurde, um ein Werkzeug dafür einzuschalten.

ORIGINALBEITRAG · TEIL 12

Dreimal dieselbe Datei

21. August 2026

Am Abend standen sechsundzwanzig Lieferungen auf dem Server, jede vor und nach dem Hochladen gemessen. Das ist die Zahl, die man in so einen Text schreibt.

Die Zahl, die haften bleibt, ist eine andere: **drei.**

So oft habe ich an diesem Tag dieselbe Datei hochgeladen. Sie lag schon beim ersten Mal richtig auf dem Server.

Das ist neun Tage nach dem Tag, an dem ich angefangen habe, Prüfsummen zu verlangen — und zwar genau deswegen: weil ich die Arbeit nicht mehrfach machen wollte. Die Prüfsumme sagt mir, was oben liegt. Sie sagt mir nicht, ob jemand sie richtig liest.

Der Faktor zehn

Der Tag fing mit einer Frage an, die länger offen war: Warum sind Fotos nicht in der Sicherung?

Die Antwort stand seit Wochen im Quelltext: *„Ein Törn mit 200 Bildern wären mehrere Gigabyte, und dafür ist weder der Server gedacht noch die Sicherung."*

Ein klarer Satz, eine klare Begründung — und falsch.

Denn zwei Absätze weiter oben steht in **derselben Datei**, was die App tatsächlich speichert: verkleinert auf 1600 Punkte, JPEG mit 82 Prozent, „aus 4 MB werden meist unter 400 KB". Zweihundert Bilder sind damit rund 80 MB. Nicht mehrere Gigabyte. Um etwa den Faktor zehn daneben.

Die Begründung galt für die Originale. Gespeichert wird aber nicht das Original.

Niemand hat nachgerechnet, weil die Zahl in die gewünschte Richtung zeigte.

Wichtiger als der Rechenfehler ist das, was darunter lag: Fotos liegen in einem anderen Speicher als der Rest — und dieser Speicher wird genauso abgeräumt. Genau die Sieben-Tage-Regel, vor der die App selbst warnt, löscht auch die Bilder.

Wer pflichtbewusst gesichert hat, rettete sein Logbuch und verlor trotzdem jedes Foto.

Seit dem Tag gibt es eine eigene, verschlüsselte Bilddatei. Nicht als Anhang an die bestehende Sicherung, sondern **daneben** — aus einem Grund, der keine Ordnungsliebe ist: Die Vollsicherung ist der einzige Weg, der die Borddaten rettet. Käme die Bildermenge in denselben Verschlüsselungsvorgang, hinge die Rettung des Logbuchs daran, dass auch der Bilderdurchgang gelingt.

Gemessen wurde, was dabei im Arbeitsspeicher liegt: Eine Datei von 40,7 MB kostet 0,1 MB. Und ein absichtlich beschädigter Block hat sich selbst übergangen — fünf von sechs Bildern kamen zurück, das sechste wurde gemeldet.

Wo die Sekunden lagen

Zweiter Punkt: Beim Anlegen der Beispieldaten steht mehrere Sekunden ein leeres, unbenanntes Boot auf dem Schirm. Der naheliegende Gedanke ist eine Ladeanzeige.

Erst messen. Das Anlegen selbst kostet vierzehn Schreibvorgänge, 9.706 Bytes, **unter zwanzig Millisekunden.**

Die Sekunden lagen woanders: Die App lädt sich nach dem Anlegen komplett neu, und das sind 86 Skripte mit zusammen 1,29 MB, alle nacheinander. Auf dem Mac 1.363 Millisekunden. Dazu 700 Millisekunden reines Warten, damit ein Hinweistext lesbar bleibt, den die Seite gleich darauf verwirft.

Eine Ladeanzeige in der Demo hätte einen von fünf Wegen abgedeckt. Dasselbe leere Boot erscheint nach jeder Wiederherstellung und bei jedem gewöhnlichen Start.

Die Abdeckung sitzt jetzt in der Seite selbst und gilt für alle fünf. Die 700 Millisekunden sind ersatzlos weg.

Vier Menüs auf einer Website

Am Nachmittag kam heraus, dass 59 Seiten **vier verschiedene Menüs** führten. Kein Fehler, den jemand gemacht hat — eines wurde kopiert, dann eines geändert, dann wieder eines.

Die Folgen waren messbar. Nur 9 von 59 Seiten führten überhaupt zum Entwicklungstagebuch — ausgerechnet die Tagebuchbeiträge selbst.

Vier Messungen, die nicht das getroffen haben, was sie treffen sollten

Am Abend kam eine Prüfung der Bedienbarkeit dazu, und die hat vor allem sich selbst geprüft.

Die erste Zahl: null von 274 bei der verbreitetsten Bauform — ein Artefakt des Prüfskripts, nicht der Seite.

Die zweite: 45 Stellen, an denen dieselbe Ansage dreimal hintereinander steht. Ebenfalls ein Artefakt. Der Browser führt ein Element, seinen Textknoten und dessen Textkasten getrennt, alle mit demselben Namen. Eine Vorlesehilfe sagt das nicht dreimal. Echt war genau **eine** Stelle.

Übrig blieb aber ein Befund, der es in sich hat: **46 Knöpfe hatten als einzigen Text ein Symbol und keine Beschriftung.** Eine Vorlesehilfe sagt dann das Zeichen an — „mal", „Häkchen", „minus eins".

Wer nicht sieht, worauf er drückt, drückt nicht.

Seit dem Abend sind es null.

Nachtrag vom 28. August: Diese Knöpfe waren der Grund, warum ich sieben Tage später bei der Messung 398 Knöpfe mit Namen und null ohne bekam. Die Zahl stimmte. Sie hat mir trotzdem nicht gesagt, dass der Logbuchdialog mit einer Vorlesehilfe unbenutzbar ist.

Was ich mitnehme

Vier Fehler an einem Tag, und alle vier haben dieselbe Form: **Es wurde gemessen, und die Messung traf nicht das, was sie treffen sollte.** Die falsche Adresse. Die nicht verfolgte Umleitung. Das Skript, das die häufigste Bauform übersah. Die Baumdarstellung, die für das Ergebnis gehalten wurde.

> **Eine Messung an der falschen Adresse ist keine Messung. Und wenn ein Test bestätigt, was man erwartet hat, ist das der Moment, ihn zu prüfen — nicht der Moment, ihm zu glauben.**

Die Regel stand schon vorher im Aufgabenheft. Sie steht dort seit dem Tag, an dem ein Cent zwei Tage gekostet hat.

An diesem Tag habe ich gelernt, dass sie auch für Zahlen gilt, die ich selbst erzeuge — besonders für die.

ORIGINALBEITRAG · TEIL 13

Der zweite fremde Prüfbericht

21. August 2026

Am Nachmittag kam ein zweiter Prüfbericht zurück. Erstellt nach der Anweisung, die ich am selben Tag geschrieben hatte, von jemandem, der die App nur von außen kennt. Neunzehn Seiten, elf Befunde, Gesamturteil 8,4 von 10 und vier Auflagen, die vor einem bezahlten Start erfüllt sein müssen.

Im Aufgabenheft steht seit dem ersten fremden Bericht eine Regel dazu: *Ein fremder Befund ist eine Beobachtung, keine Aufgabe. Vor der Aufnahme klären, womit gemessen wurde.*

Genau das habe ich getan. Es kam etwas dabei heraus, mit dem ich nicht gerechnet hatte.

Eine Uhrzeit auf die Millisekunde

Der Bericht nennt seinen Messzeitpunkt selbst, und zwar genau: **2026-08-21T12:04:26.989Z**. Das ist 14:04 Ortszeit.

Die Aussagen über die App können aber nicht von 14:04 stammen. Sie beschreiben die verschlüsselte Bildersicherung und einen Offline-Speicher namens **pelaron-62.2.0** — beides ging erst am Nachmittag online.

Der Bericht hat also zwei Zeitpunkte: einen für die Website, einen späteren für die App.

Das ist kein Vorwurf. Der Prüfer hat gemessen, was zum Zeitpunkt der Messung dastand, und den Zeitpunkt protokolliert. **Nur genau deshalb ließ sich überhaupt feststellen, dass zwei seiner Befunde einen Stand beschreiben, den es abends nicht mehr gab.**

Hätte er die Uhrzeit weggelassen, wäre beides ungeprüft in die Aufgabenliste gewandert.

> **Die Sorgfalt des Prüfers hat seinen eigenen Fehler sichtbar gemacht. Das ist der Unterschied zwischen einem Bericht, dem man glauben muss, und einem, den man nachrechnen kann.**

Der Befund, der keiner war

Ein Punkt hieß: In den Hauptansichten stehe wiederholt direkt hintereinander „Anzeigen Anzeigen Anzeigen" — drei Aktionen ohne unterscheidbaren Namen.

Ich habe die App dafür lokal aufgebaut und den Barrierefreiheitsbaum ausgelesen, den ein Vorleseprogramm benutzt.

Es gibt drei Aufklappbereiche, jeder mit einem eigenen, sprechenden Namen. Das Wort „Anzeigen" steht nicht im Text, sondern in einer Gestaltungsregel als angehängter Hinweis, und es wechselt beim Öffnen auf „Verbergen".

Im Barrierefreiheitsbaum: **null Treffer.** Im sichtbaren Text: null.

Die empfohlene Änderung hätte einen Namen geändert, den niemand hört, an einem Element, das schon einen hat.

Woher kommt der Befund dann? Vermutlich hat das Prüfwerkzeug den gerenderten Text ausgelesen und dabei den Gestaltungshinweis mitgenommen.

Das ist derselbe Fehlertyp, in den ich am selben Vormittag gelaufen war. Ein Skript meldete mir fünfundvierzig Stellen mit dreifacher gleicher Ansage — es zählte in Wahrheit Element, Textknoten und Textzeile desselben einen Elements.

Wir haben denselben Fehler gemacht, unabhängig voneinander, am selben Tag.

Der Knopf, der das Versprechen abräumt

Acht der elf Punkte halten der Nachmessung stand, vier davon konnte ich präziser machen, weil ich in den Quelltext sehen kann und er nicht.

Der schwerste: Ein Knopf in den Einstellungen räumt den Offline-Speicher ab.

Offline zu arbeiten ist nicht eine Eigenschaft von PELARON unter vielen. **Es ist das Versprechen.** Und der Knopf, der es abräumt, heißt „Wartung" — gedrückt wird er genau dann, wenn an Bord etwas klemmt, also genau dann, wenn kein Netz da ist, um den Schaden wieder aufzufüllen.

Zwei Preise ohne Quelle

Der Bericht enthält auch eine Marktübersicht mit fünfzehn Produkten. Zwei davon waren bei der eigenen Recherche nicht auffindbar — nicht das Produkt, nicht der Anbieter, nicht der Preis. Bei einem existiert die genannte Firma, aber als Yachtüberführungsunternehmen, nicht als Softwarehaus.

Das heißt nicht, dass es die Produkte nicht gibt. Es heißt, dass hinter einem Preis eine Quelle stehen muss.

Zwei von fünfzehn Zeilen ohne Beleg sind kein Beinbruch — aber sie sind der Grund, warum ich die Tabelle neu erhoben habe, statt sie zu übernehmen.

Was bleibt

Von elf Befunden hält einer der Nachmessung nicht stand. Zwei hängen an einem Stand, den es seit dem Nachmittag nicht mehr gibt. Acht sind richtig.

Das ist eine gute Quote für eine Prüfung von außen, und sie hat sich gelohnt — auch die widerlegten Punkte. Denn das Widerlegen hat mich gezwungen, richtig zu messen statt zu vermuten.

> **Ein guter Prüfbericht ist nicht der, der recht hat. Es ist der, bei dem man nachrechnen kann, wo er danebenliegt.**

ORIGINALBEITRAG · TEIL 14

Die Antwort stand vier Zeilen über dem Fehler

21. August 2026

Nach dem fremden Bericht habe ich eine eigene Prüfung gemacht. Beide Seiten, nach derselben Anweisung, aber mit einem Werkzeug, das der externe Prüfer nicht hatte: Zugriff auf den ausgelieferten Quelltext.

Eigenes Urteil: **7,9 von 10.** Der externe Bericht sagte 8,4.

Der ganze Unterschied liegt in einem Wort — Gestaltung wurde **gemessen** statt angesehen.

Angesehen sieht PELARON gut aus. Ruhige Farben, klare Karten, ordentliche Typografie. Gemessen kommt etwas anderes heraus.

Vier Zeilen

Die Stildatei der App beginnt mit einem Kommentar. Er ist gut, er ist richtig, und er ist der Beweis, dass die Erkenntnis längst da war:

> **Textfarbe getrennt von der Markenfarbe. #f27600 erreicht auf Weiß nur 2,85:1 — zu wenig für Text, den man im Cockpit bei Sonne lesen soll. #b34700 erreicht 5,50:1 und wirkt nebeneinander noch als dieselbe Familie. Das helle Orange bleibt für Flächen, Ränder und Symbole, wo Kontrast keine Rolle spielt.**

Vier Zeilen darunter, in derselben Datei, steht die Regel für den Hauptknopf: eine orange Fläche, die weißen Text trägt.

Also genau der Fall, den der Kommentar vier Zeilen vorher ausschließt.

Gemessen 2,85:1, verlangt sind 4,5:1. Und nicht an einer Randstelle, sondern am wichtigsten Knopf fast jeder Ansicht: „+ Logbucheintrag", „Törn starten", „Ankerposition setzen", „+ Dokument hinzufügen".

Neunundzwanzig Stellen in sechzehn von einundzwanzig Ansichten.

Es ist aber ein Fehler, nicht neunundzwanzig. Eine Regel, eine Zeile.

Dieselbe gute Entscheidung, am falschen Ort

Auf der Website gibt es denselben Kommentar und dieselbe Farbvariable. Und dort kennzeichnet eine Regel den Menüpunkt der Seite, auf der man gerade ist — mit genau dieser Textfarbe.

Die Kopfleiste ist dunkel. Die Farbe ist für dunklen Text auf hellem Grund gerechnet. Ergebnis: **1,88:1.**

Der aktive Menüpunkt ist damit der am schlechtesten lesbare Punkt im Menü. Die Regel, die die aktuelle Seite hervorheben soll, macht sie unkenntlich. Auf jeder einzelnen Seite.

Die helle Gegenvariante steht bereits eine Zeile höher in derselben Datei und erreicht dort 5,61.

Es fehlte nicht das Wissen. Es fehlte der Gedanke, dass eine Farbe zwei Varianten braucht — eine für hellen und eine für dunklen Grund.

Eine Schrift, die nie ankommt

Die Stildatei benennt zwei Schriften: Inter für den Fließtext, Montserrat für die Wortmarke.

Gesucht und nicht gefunden in der gesamten Auslieferung: **eine Schriftdatei.** Keine, in keinem Format. Auch kein Verweis auf einen Schriftendienst — die Sicherheitsregel der App würde ihn ohnehin blockieren.

Auf einem iPad ist weder Inter noch Montserrat installiert. Der Fließtext fällt also auf die Systemschrift zurück, was gut aussieht. Die Wortmarke fällt zwei Stufen weiter, bis auf Helvetica.

Das ist kein Fehler, der auffällt — es ist einer, der nicht auffällt. Die App sieht ordentlich aus, weil die Systemschriften ordentlich aussehen.

Der einzige Punkt, an dem wir etwas sagen können

Zur Prüfung gehörte ein Vergleich mit sechzehn Produkten. Ein Befund war deutlicher als erwartet:

Bei keinem der neun direkten Wettbewerber gibt es eine belastbare Aussage zur Verschlüsselung. Einer nennt Konformität und ein Audit, aber kein Verfahren. Einer wirbt mit „end-to-end encrypted" und wird vom eigenen Store-Eintrag widerlegt, der nur Transportverschlüsselung angibt und einräumt, dass Finanzdaten mit Dritten geteilt werden können.

PELARON nennt Verfahren, Rundenzahl, Salz- und IV-Länge und eine Prüfsumme. Das ist der einzige Punkt im ganzen Feld, an dem wir etwas sagen können, das sonst niemand sagt.

Und es ist kein Marketingsatz, sondern eine Zeile im Quelltext, die man nachlesen kann.

Was bleibt

Von den Punkten, die in der eigenen Prüfung Abzug kosten, sind die meisten Zeilenänderungen, keine Umbauten.

Und in fast jedem Fall **steht die richtige Lösung bereits im eigenen Bestand**: die Farbvariable in derselben Datei, die Meldezeile im Nachbarmodul, das Exportmuster in der Bildersicherung von gestern, die korrekte Aufräumlogik im Offline-Dienst.

> **Ein Blick sagt, ob etwas gut aussieht. Eine Messung sagt, ob man es bei Sonne im Cockpit lesen kann. Das sind zwei verschiedene Fragen, und nur eine davon stand bisher in der Prüfanweisung.**

Nachtrag vom 28. August: Der Satz „vier Zeilen darunter" ist der kürzeste Ausdruck für etwas, das mir seitdem in fast jeder Woche begegnet ist. Am 28. August habe ich eine Datei gefunden, in der die richtige Wartungsregel im Kommentar steht — und **fünfzig Zeilen darunter**, in derselben Datei, rechnet die Kennzahl anders. Der Abstand wächst. Die Gestalt bleibt.

ORIGINALBEITRAG · TEIL 15

Eine Zeile statt sieben

22. August 2026

Zwölf Lieferungen, jede vor und nach dem Hochladen am Server gemessen. Am Abend steht die App bei null Kontrastfehlern, wo es morgens vierundsechzig waren, und bei null Ansichten mit waagerechtem Überlauf, wo es neun waren.

Das ist die Zahl für die Bilanz. Der Tag hatte eine andere Pointe.

Der Knopf, der das Versprechen abräumt

Die erste Lieferung war die wichtigste und bestand aus einer **gelöschten Zeile.**

„Lokale Wartung" hatte eine zweite Aufräumroutine für die Zwischenspeicher, deren Ausnahme nie zutraf — siebenundneunzig Einträge im Offline-Speicher wurden zu null. Genau der Knopf, den man drückt, wenn an Bord etwas klemmt. Also genau dann, wenn kein Netz da ist, um den Schaden wieder aufzufüllen.

Entfernt, nicht repariert.

Der Offline-Dienst räumt im eigenen Ablauf bereits richtig auf und kennt dort beide Speichernamen; die Zeile in der App konnte den Namen gar nicht kennen.

> **Zwei Fassungen derselben Aufgabe waren der Fehler, nicht die eine falsche Zeile.**

Im Heft steht dafür eine Regel: Wer dreimal flickt, hätte umbauen sollen. Hier reichte einmal.

Ansichten sind keine Zustände

Bei den Farben war gründlich gemessen worden: einundzwanzig Ansichten, drei Bildschirmbreiten, jedes sichtbare Textelement gegen seinen tatsächlichen Hintergrund gerechnet. Vierundsechzig Fehlstellen.

Beim Schreiben der Ersetzungen ist die Stildatei noch einmal von Hand durchgesehen worden — und dabei tauchten zwei Stellen auf, die im Durchlauf nicht vorgekommen waren.

Eine davon ist die Auswahl der Notrufstufe. Bei 2,85:1.

Warum hatte der Durchlauf sie nicht? **Weil sie ein Zustand ist, keine Ansicht.** Die Farbe erscheint erst, wenn die Stufe gewählt ist. Das Werkzeug hat einundzwanzig Ansichten geöffnet und in keiner davon etwas ausgewählt.

> **Ein Durchlauf durch alle Ansichten prüft Ansichten. Zustände findet er nie.**

Dasselbe galt für eine Ablaufkachel, die es nur mit angelegten Bootsdaten gibt. Und für die Ausrichtung der Bedienzeilen, die schlicht nie gemessen worden war: Die Filterzeile im Logbuch stand um sechzehn Punkte schief.

Ich habe das in drei Sekunden auf dem Bildschirm gesehen. Der Prüfaufbau in einem ganzen Tag nicht.

Eine Zeile statt sieben

Die Ersetzungsliste für die Farben hatte sieben Einträge.

Beim Nachzählen der Erwartungswerte — jede Zahl, die in einen Prüfbefehl geht, wird vorher im eigenen Bestand gezählt — fiel auf, dass das Suchmuster eine Schreibweise übersehen hatte: eine Variable mit Ersatzwert in Klammern.

Also wurde nachgesehen, was diese Variable überhaupt tut. Ergebnis: **159 Verwendungen, ausnahmslos als Textfarbe.** Kein einziges Mal für einen Rand, keine Fläche, kein Symbol. Über drei Dateien hinweg.

Damit war der ganze Rest eine Zeile: Die Variable zeigt jetzt auf ihre bessere Fassung.

Sieben Ersetzungen wären richtig gewesen. Eine ist besser.

Vier Seiten, die niemand erreicht

Beim selben Nachsehen kam noch etwas heraus: **Vier Seiten hängen gar nicht am gemeinsamen Stylesheet.** Sie tragen ein eigenes, im Kopf der Seite.

Die Kontrastkorrektur von diesem Tag hat sie nicht erreicht. Und jede künftige Änderung wird sie auch nicht erreichen.

Im Heft stand dazu bisher „vier Seiten ohne Menü". Der Befund ist größer, als er klang.

Was ich mitnehme

Der beste Fund des Tages war nicht der kaputte Wartungsknopf. Es war die eine Zeile, die 159 Stellen auf einmal richtig macht — und sichtbar wurde sie nur, weil die eigene Suche vorher danebengegriffen hatte.

Der zweitbeste war eine Zahl, die seit Wochen falsch im Heft stand, ohne jemals falsche Ergebnisse zu produzieren. Sie hat einfach jedes Mal eine Seite vergessen. Die wichtigste.

> **Ein Werkzeug, das ein Ergebnis liefert, hat nicht bewiesen, dass es das Richtige gemessen hat. Gerade dann nicht, wenn das Ergebnis zur Erwartung passt.**

Das steht seit dem Tag als Regel im Heft. Nicht als Vorsatz, sondern als Ablauf.

Nachtrag vom 28. August: Der Satz mit den Zuständen hat sich sechs Tage später noch einmal gemeldet, in seiner schärfsten Form. Alle Auszeichnungen für Vorlesehilfen waren richtig — 398 Knöpfe mit Namen, null ohne. Gemessen war ein **Zustand**. Nicht gemessen war das **Verhalten über Zeit**: ein Hinweis, der sich jede Sekunde neu schrieb, ohne sich zu ändern. Ansichten, Zustände, Verläufe. Drei Stufen, und ich war bis dahin auf der zweiten.

ORIGINALBEITRAG · TEIL 16

Eine Schrift, die nie ankam

22. August 2026

„Einheitlich und auf allen Geräten anwendbar, egal ob iOS, macOS oder Android oder Windows, immer gleich schick."

So kam die Aufgabe von mir. Das klingt nach Geschmack.

Es war eine Messung.

Null

Zwei Stildateien, die der App und die der Website. Beide nennen seit Monaten dieselben zwei Schriften.

Gezählt wurde dann Folgendes:

```
Schriftdefinitionen            null
Verweise auf eine Schriftdatei null
in beiden Dateien
```

Die Namen standen da. **Ausgeliefert wurde keine der beiden Schriften — nie.**

Jedes Gerät hat sich seine eigene Ersatzschrift gesucht: eine auf dem iPad und dem Mac, eine andere auf Windows, eine dritte auf Android.

Gemerkt hat es niemand, weil alle Beteiligten auf Apple-Geräten hingesehen haben.

Das ist die unangenehmste Sorte Fehler. Nichts war kaputt, nichts hat gemeldet, keine Prüfung schlug an. Es sah überall ordentlich aus — nur eben überall anders.

Die Zahl, die die Entscheidung abgenommen hat

Bleibt die Frage, was man ausliefert: acht einzelne Schnitte oder eine Schrift mit stufenlosem Gewicht. Normalerweise eine Abwägung zwischen Dateigröße und Bequemlichkeit.

Hier nicht. Gezählt wurde, welche Strichstärken die Stildateien tatsächlich verlangen: **400, 500, 600, 650, 700, 750, 800, 850, 900.**

Drei davon gibt es in keinem festen Schnitt.

Eine Schrift mit stufenlosem Gewicht war damit keine Wahl mehr, sondern die einzige Möglichkeit, die vorhandene Gestaltung überhaupt darzustellen.

Und noch etwas kam dazu, das die Frage längst vorentschieden hatte: In der Serverkonfiguration steht seit Wochen, dass Schriften nur vom eigenen Server geladen werden dürfen. Eine Einbindung von außen wäre also ohnehin blockiert worden — und ohne Netz hätte sie sowieso nicht funktioniert.

> **Eine Sicherheitszeile, geschrieben aus einem ganz anderen Anlass, hatte eine Gestaltungsfrage beantwortet — zwei Monate bevor sie gestellt wurde.**

Was „überall gleich" nicht einschließt

Beim Bauen fiel etwas auf, das die Zusage einschränkt, und es gehört hierher, weil es sonst niemand merkt.

Quer durch die App stehen **zweiundvierzig verschiedene Sonderzeichen in einhundertsechzig Vorkommen**: Pfeile, geometrische Formen, ein Anker, ein Segelboot, ein Telefon, ein Zahnrad, ein Rettungsring. Der Pfeil allein einundfünfzigmal.

Keines davon steckt in einer der ausgelieferten Schriften. Sie kommen weiterhin vom Betriebssystem — und sechs von ihnen werden auf manchen Geräten farbig gezeichnet und auf anderen einfarbig.

Ein Anker, der auf dem iPad bunt und auf Windows grau ist, ist das genaue Gegenteil dessen, was an dem Tag erreicht werden sollte.

Der ehrliche Weg sind gezeichnete Symbole statt Schriftzeichen. Das steht als eigener Punkt im Heft, nicht als Nebenbemerkung.

Zwei Regeln, die an dem Tag entstanden sind

Die eine: **Jeder Verbesserungsvorschlag — eigener wie fremder — kommt zuerst ins Heft**, mit Nummer, Herkunft und Status, und wird dann Punkt für Punkt abgearbeitet. Kein Vorschlag lebt nur in einem Prüfbericht, den niemand wieder aufschlägt.

Die andere ist die Lehre aus einem überschriebenen Textbaustein: **Jede Lieferung nennt hinzugefügte und entfernte Zeilen** gegen den Ausgangsstand.

Eine Prüfsumme sagt, dass eine Datei angekommen ist. Sie sagt nicht, dass sie richtig ist. *Achtunddreißig Zeilen dazu und null weg* — das hätte den Fehler in dem Moment gezeigt, in dem er entstand, statt zwanzig Minuten später.

> **Beide Fehler des Tages haben dieselbe Wurzel, und es ist nicht Unachtsamkeit. Es ist die Annahme, eine kleine Änderung brauche das Verfahren nicht.**

Was am Abend stand

Dreiundsechzig von dreiundsechzig Seiten mit ausgelieferter Schrift, wo es am Morgen null waren. Die Dokumente zum ersten Mal in einer Sicherung — Angaben und Dateien. Und im Heft ein Register mit fünfunddreißig durchnummerierten Punkten, von denen fünf erledigt sind.

Die Zahlen sind das Ergebnis. **Die zwei Regeln sind der Ertrag.**

Nachtrag: Aus diesen fünfunddreißig Punkten sind bis zum 28. August hundertsechs geworden, und hundertfünf davon waren an dem Abend erledigt. Das Register ist das Einzige aus dieser Zeit, das ich heute nicht anders machen würde.

ORIGINALBEITRAG · TEIL 17

Vorhanden ist nicht erreichbar

23. August 2026

„Nur ist die Academy-Seite nirgends zu finden."

Ein Halbsatz von mir am Nachmittag, während wir eigentlich an etwas anderem arbeiteten. Er hat den Tag zusammengefasst, ohne dass das jemand geplant hätte.

Drei Dinge, die es gab

Der Tag hat drei Sachen ans Licht gebracht, die sich gleichen. **Alle drei waren gebaut, ausgeliefert und funktionsfähig. Keine davon war erreichbar.**

Der Wissenstest. Auf der Academy-Seite steht ein Test mit zehn Fragen zu je drei Antworten, 6.494 Zeichen Programmcode. Er lag im Quelltext der Seite selbst.

Vor Wochen hat die Website eine Sicherheitskopfzeile bekommen, die Skripte nur noch aus eigenen Dateien erlaubt — ein sinnvoller Schutz, der genau das tut, was er soll. Seitdem wurde der Test vom Browser gar nicht erst ausgeführt.

```
Antwortknöpfe auf dem Bildschirm, unter der echten Kopfzeile:   0
nach dem Auslagern in eine eigene Datei:                       30
```

Nichts hat gemeldet. Die Seite lud, sah vollständig aus, und wo der Test hätte stehen sollen, stand nichts — kein Fehler, keine Lücke, einfach ein Stück Seite, das aufhört.

Das Menü. Beim Nachmessen fiel auf, dass vier weitere Seiten dasselbe Problem haben: Impressum, Datenschutz, Nutzungsbedingungen und die App-Seite. In allen vieren steht derselbe Block, 385 Zeichen, Zeichen für Zeichen identisch. Er lässt die Menütaste am Telefon funktionieren.

Auch er wurde nie ausgeführt. Und weil auf schmalen Bildschirmen die Menüleiste ausgeblendet ist und nur die Taste bleibt, gab es von diesen vier Seiten aus **überhaupt keinen Weg zurück.**

Wer mit dem Telefon im Impressum landete, war dort gefangen.

```
Erreichbare Menüpunkte nach dem Antippen, 390 Punkte breit:
vorher 0 · nachher 9
```

Ausgerechnet Impressum, Datenschutz und Nutzungsbedingungen. Die drei Seiten, die es aus rechtlichen Gründen geben muss.

Die Seite selbst. Und dann mein Halbsatz. Die Academy-Seite gibt es, sie ist ausführlich, sie steht in der Sitemap, sie sagt auf sich selbst, sie gehöre unter „Wissen".

Von 64 Adressen der Website verweist **eine einzige** auf sie — ein Satz im Fließtext einer anderen Seite. Nicht in der Kopfnavigation, nicht in der Fußzeile.

Suchmaschinen finden sie. Menschen nicht.

Der vierte Fall: eine Antwort, auf die niemand wartet

Ein fremder Prüfbericht hatte notiert, die Hafensuche brauche im ungünstigsten Fall 32 Sekunden. Ich hatte die Zahl im Heft stehen, ohne sie nachzumessen — und die Begründung des Berichts war falsch, was es leicht machte, die Zahl gleich mit abzutun.

Die Zahl stimmte.

Der Weg ist: acht Sekunden für die Ortssuche, danach bis zu drei Umkreisabfragen zu je acht Sekunden, nacheinander. Acht plus dreimal acht.

In dieser ganzen Zeit stand ein einziger Satz auf dem Bildschirm. Kein Fortschritt, nichts zum Abbrechen.

Im Prüfstand nachgestellt, der Datendienst antwortet absichtlich nicht: 31,6 Sekunden, zehn abgesetzte Abfragen, zwei Meldungen auf dem Schirm.

Jetzt sind es zwölf.

Zwei Sachen, die mir die Arbeit erschwert haben

Zwei kleinere Befunde betrafen nicht die App, sondern meinen eigenen Ablauf.

Der erste: Dateien kamen mit Namenszusätzen an und zwangen mich, vor jedem Hochladen umzubenennen — Arbeit, die niemand braucht, und eine Gelegenheit, die falsche Datei am falschen Ort abzulegen.

Der zweite kam aus einem Ausfall: Mein Schreibtisch liegt in einer Cloud, die hochgeladene Dateien auslagert und nur Platzhalter zurücklässt. Ein halbes Megabyte ist an dem Tag zweimal ins Leere gelaufen, bevor der Grund klar war.

Übergaben laufen seitdem über einen Ordner ohne Synchronisierung.

Was ich mitnehme

Alle vier Befunde des Tages haben dieselbe Form: **Etwas ist da, gebaut, gemessen, ausgeliefert — und niemand kommt hin.**

> **Eine Prüfsumme sieht das nicht. Sie bestätigt, dass die richtige Datei angekommen ist, und schweigt darüber, ob jemand sie je zu Gesicht bekommt.**

Am Abend stehen der Wissenstest und die Menütaste wieder, die Hafensuche antwortet in zwölf statt in zweiunddreißig Sekunden, und im Register stehen vierzig durchnummerierte Punkte.

Und einer davon ist die Frage, wie man eine Seite findet, die es gibt.

Nachtrag vom 28. August: Fünf Tage später habe ich in der App eine Spurenliste gefunden, die vollständig gebaut war und hinter einer Zeile lag, die nach Hinzufügen klang. Ich habe die App gebaut und den Löschknopf nicht gefunden. Und am selben Abend: fünfundzwanzig Dialoge, jeder mit einer Überschrift, keiner mit einem Namen für die Vorlesehilfe. **Vorhanden ist nicht erreichbar** ist der Satz, der mich am längsten begleitet hat.

ORIGINALBEITRAG · TEIL 18

Fünfmal das Falsche gemessen

23. August 2026

Der Beitrag von heute Vormittag hört mittags auf. Der Tag nicht.

Am Vormittag ging es um Dinge, die da sind und die niemand erreicht. Am Abend um etwas anderes: um Messungen, die eine Zahl liefern, die stimmt — und die trotzdem nicht die Frage beantworten, um die es geht.

Fünfmal an einem Abend.

Ein Band über einem Band

Zuerst das, was einfach fertig geworden ist: Die Academy hat eine Kachel statt eines Satzes im Fließtext. Und das gesammelte Tagebuch steht auf 86 Blättern statt 82.

Beim Fortschreiben kam etwas heraus, das ich nicht gesucht hatte. Auf dem Deckblatt stand sichtbar „Teile 1 bis 16". In der Textebene darunter stand eine allein stehende **15**.

Das Deckblatt hatte zwei Aufdruckbänder übereinander. Beide zeichnen dasselbe dunkle Rechteck über das Foto, das ältere mit der alten Zahl, das jüngere mit der neuen. Das zweite lag über dem ersten und hat es unsichtbar gemacht. Sichtbar war alles richtig. Beim Kopieren, beim Suchen und in jeder Textmessung kam die alte Zahl mit.

> **„Ein Deckel ist keine Entfernung" stand seit dem Vortag in meinem Heft. Gestern war es ein weißes Rechteck über einer Fußzeile. Heute ein Band in Hausfarbe über einem Band in Hausfarbe — und ich hatte es selbst dorthin gelegt.**

Jetzt sind beide weg, nicht überdeckt. Der Zeichenaufruf und das Objekt.

Derselbe Befund, drei Fassungen

Dann der Punkt, der den Abend gekostet hat. Die App lädt beim Start 87 Programmdateien. Dreiundvierzig Module rufen ihre eigene Startfunktion nicht sofort auf, sondern mit einer von Hand gesetzten Verzögerung, 400 bis 2000 Millisekunden. Eine Leiter aus dreiundvierzig Stufen.

Erste Fassung: „Dreiundvierzig Module an einer Zeitleiter, das kostet." Fundstellen gezählt und die Anzahl für ein Maß der Kosten gehalten.

Zweite Fassung, nach der ersten Messung: Die dreiundvierzig Startfunktionen kosten zusammen 54 Millisekunden Rechenzeit, die teuerste einzelne 9,4. Also nichts zu holen, der Umbau nicht begründet.

Dritte Fassung, nach der Messung am iPad: Das Gerüst steht dort nach 984 Millisekunden. Die letzte Stufe der Leiter feuert 2000 Millisekunden danach.

> **Alles zwischen 984 Millisekunden und drei Sekunden ist Warten, nicht Arbeiten. Die Zeitleiter kostet keine Rechenzeit — sie ist die Wartezeit.**

Die zweite Messung war richtig und hat das Falsche gemessen: Rechenzeit, wo es um Wartezeit ging. Die erste hatte gar nicht gemessen, sondern gezählt.

Gemessen am falschen Gerät

Der Umbau lag nahe: Statt auf die Uhr zu warten, sollen die Module starten, sobald der Hauptfaden frei ist. Dafür gibt es eine Browserfunktion. Eingebaut, zwölf Module umgestellt, im Prüfstand gemessen: Das letzte Modul war statt nach 2158 Millisekunden nach 192 fertig. Zwei Sekunden gewonnen.

Am iPad, nach dem Hochladen: **beiRuhe** lief nach 2002 ms. Und auf die Frage, ob die Funktion überhaupt da ist: **false**.

Safari kennt sie auf diesem Gerät nicht. Der eingebaute Rückfall lief in seine Frist — genau das Verhalten, das ersetzt werden sollte. Die Messung war nicht falsch. Sie war nicht übertragbar: gemessen in einem Browser auf einem Rechner, gebaut für ein iPad.

Die zweite Fassung hängt jetzt an nichts, was ein Browser haben kann oder auch nicht. Und ihre Wirkung liest man ab, statt sie zu glauben: **[3, 12, 0, 2077]** — dritte Fassung, zwölf Starts, **null** über die Frist, letzter fertig nach 2077 Millisekunden. Die Null ist die wichtigste Zahl.

Nebenbei fiel dabei ein echter Fehler auf: Ein Riegel arbeitete die Warteschlange genau einmal ab. Wer sich danach anmeldete, stand in einer Schlange, die nie wieder anlief. Für die zwölf Module folgenlos. Für jeden späteren Aufrufer ein stiller Ausfall.

Fünf fremde Befunde, drei halten nicht

Zum Schluss habe ich zwei Prüfberichte gegen mein Register gehalten und fünf Befunde gefunden, die dort in keiner Zeile standen. Eingetragen — und dann gemessen.

Die Datenschutzerklärung sollte zehn fremde Dienste nicht nennen. Der Serverstand nennt alle zehn. Erledigt, bevor ich es eingetragen habe. Zwei echte Fehler standen trotzdem darin: Zwei Dienste waren als auswählbare Kartenschicht beschrieben. Nachgemessen kommen ihre Adressen im ganzen Bestand nur an einer Stelle vor, als Testkacheln eines Werkzeugs zur Fehlersuche. *Die Erklärung beschrieb eine Funktion, die es nicht gibt, und verschwieg die, die es gibt.*

Der waagerechte Überlauf bei 320 Pixeln war mein eigener Messfehler. Fünf von acht Ansichten, schlimmster Fall 738 Pixel, „Verursacher jedes Mal ein Knopf". Buchstäblich richtig und bedeutungslos: Die Knöpfe stehen in einer Leiste, die von sich aus seitlich rollt. Sauber nachgemessen, acht Ansichten und neunundzwanzig Unteransichten einzeln aufgeklappt: **null**.

Zwei von fünf halten stand, einer war schon erledigt, zwei gibt es nicht.

Was ich mitnehme

In meinem Heft steht: Ein fremder Befund ist eine Beobachtung, keine Aufgabe. Daneben: Jeder Vorschlag kommt zuerst ins Heft. Ich habe die zweite Regel befolgt und die erste dabei übersprungen. Ein Register, in das ungeprüfte Befunde wandern, ist kein Register. Es ist eine zweite Ablage für PDF-Inhalte.

> **Fünf Messungen an einem Abend haben etwas gemessen, das nicht die Frage war: gezählte Fundstellen statt Kosten, Rechenzeit statt Wartezeit, einen Rechner statt ein iPad, eine Kopie statt den Server, Knöpfe in einem rollenden Kasten statt einen Überlauf. Jede einzelne lieferte eine Zahl. Keine davon war falsch.**

Das ist die unangenehme Sorte Irrtum. Ein Rechenfehler meldet sich. Eine Messung, die das Falsche misst, sieht aus wie ein Ergebnis.

Nachtrag vom 28. August: Dieser Satz hat fünf Tage später den ganzen Tag bestimmt. Ich hatte für die Bedienung mit Sprachausgabe alles gemessen, was sich messen lässt — jede Auszeichnung, jede Beschriftung, jeden Namen. Sechs Fehler haben sich davon nicht erwischen lassen, drei hätte kein Prüfprogramm der Welt gefunden. Gemessen war die Auszeichnung. Nicht gemessen war das Verhalten über Zeit. Am 23. August habe ich fünfmal das Falsche gemessen und es abends gemerkt. Am 28. war es dasselbe, nur einen Tag lang.

ORIGINALBEITRAG · TEIL 19

Was auf dem Bildschirm steht, gilt

24. August 2026

Gestern um 13:39 habe ich mich beim Apple Developer Program angemeldet. Heute um 11:54 ist der Antrag draußen, der über eine der wichtigsten Funktionen der geplanten iPhone-Fassung entscheidet.

Dazwischen liegen zweiundzwanzig Stunden und vier Stellen, an denen der Bildschirm etwas anderes sagte als meine eigene Anleitung.

Dieser Beitrag holt etwas nach. Die Anmeldung fiel mitten in den Nachmittag, über den Teil 18 berichtet — und kommt dort nicht vor. Ich habe gestern Abend über das geschrieben, woran ich gearbeitet hatte, nicht über das, was der Tag enthielt. Das ist ein Unterschied, und er gehört bemerkt.

Der Ausweis, den es nicht gab

Zur Anmeldung braucht Apple einen Lichtbildausweis. Ich hatte mir vorher eine Anleitung geschrieben und darin den Reisepass empfohlen — gestützt auf Apples allgemeine Seite, die von „passports in most regions" spricht. Maschinenlesbar, international, eindeutig.

Die App bot in Deutschland keinen Reisepass an. Nur den Kartenführerschein.

> **Nicht falsch recherchiert. An der falschen Stelle recherchiert. Die allgemeine Seite beschreibt, was Apple weltweit anbietet. Der Bildschirm zeigt, was hier gilt.**

Gekostet hat es eine Minute Verwirrung. Es ist trotzdem dieselbe Sorte Fehler, die mich in diesem Projekt schon mehrfach eingeholt hat: eine Quelle lesen, die allgemein richtig ist, und sie für die Antwort auf die konkrete Frage halten.

Drei Zahlen für denselben Betrag

Die Mitgliedschaft kostet 99 US-Dollar im Jahr. So steht es auf Apples Seite.

Die App zeigte beim Kauf **$98.99**.

Abgebucht wurden **98,99 Euro**.

Drei Angaben, ein Vorgang. Keine davon ist gelogen, und keine zwei stimmen überein. Es gilt, was auf der Rechnung steht — und das ist nicht die Seite, auf der man vorher nachgesehen hat.

Nebenbei, klein gedruckt und wichtig: Die Anmeldung über die App ist ein Abonnement. Sie verlängert sich am 23. August 2027 automatisch. Kündigung spätestens einen Tag vorher.

Eine Tür, die zufällt

Bei der Anmeldung fällt eine Entscheidung mit, die man leicht übersieht: Einzelperson oder Organisation.

Als Einzelperson erscheint im App Store der Rechtsname als Anbieter — mein Name, nicht „PELARON". Der Name der App bleibt davon unberührt. Für „PELARON" als Anbieter bräuchte es eine Kapitalgesellschaft mit eigener Kennnummer.

Ein Einzelunternehmen lässt sich später nicht umstellen. Das ist keine Einstellung, die man ändert, sondern ein anderes Konto. Ich habe es trotzdem so gemacht, weil die Alternative eine Frage an den Steuerberater ist und den Antrag um Wochen verzögert hätte.

Aber es ist eine Tür, die zufällt, und ich schreibe sie deshalb auf.

Vierzehn Stunden, siebenundvierzig Minuten

Apple sagt zur Prüfung des Ausweises: Freischaltung binnen 24 Stunden.

Die beiden Bestätigungen kamen heute um 04:26 und 04:27. Vierzehn Stunden und siebenundvierzig Minuten nach der Anmeldung.

Eine Zusage, die eingehalten wurde, ist keine Nachricht wert. Ich schreibe sie trotzdem auf, weil ich sonst nur die Fälle notiere, in denen etwas schiefgeht.

Eine Kennung, die für immer bleibt

Der erste Schritt danach war kein Programmieren, sondern ein Formular: die Bundle-Kennung. Der eindeutige Name der App bei Apple, weltweit einmalig, nach der ersten Einreichung nie wieder änderbar.

de.pelaron.app. Passt zu beiden Adressen, pelaron.de und pelaron.app, und bleibt gültig, egal welchen technischen Weg die native Fassung später nimmt.

Registriert um 11:36. Der billigste Schritt des ganzen Vorgangs und einer der wenigen, die man nicht nachholen kann, wenn jemand anders schneller war.

Warum es überhaupt eilte

Die Ankerwache braucht ein besonderes Recht, das Apple einzeln vergibt: kritische Hinweise. Der Unterschied zur nächstschwächeren Stufe ist genau ein Satz aus Apples eigener Beschreibung.

Eine zeitkritische Mitteilung „breaks through system notification controls". Nur eine kritische Mitteilung „bypasses the mute switch to play a sound".

Nur die kritische Stufe durchbricht den Stummschalter. Und der ist nachts an Bord gesetzt.

> **Ein Schiff liegt vor Anker. Die Crew schläft unter Deck, das Gerät ist gesperrt, stumm und am Ladekabel. Wenn der Anker bei auflandigem Wind slippt, sitzt das Boot in Minuten fest. Eine Meldung, die auf einem stummen Gerät keinen Ton machen kann, ist dasselbe wie keine Meldung.**

Das ist kein Bequemlichkeitsargument und keine Frage von Sekunden. Es ist der Unterschied zwischen einer Crew, die aufwacht, und einer, die es nicht tut.

Fünf Felder, wo drei stehen sollten

Ich hatte den Antrag vorbereitet: App-Name, Bundle-Kennung, Begründung. So stand es in meinem eigenen Entscheidungspapier, gestern nachgeschlagen.

Das Formular verlangt zwei Auswahlfelder und **drei** Textfelder.

Beim App Type stehen vier Möglichkeiten: Gesundheit, öffentliche Sicherheit, Personen- und Objektsicherheit, Sonstiges. Gesundheit wäre falsch, öffentliche Sicherheit meint Behörden und Rettungsdienste, Sonstiges hieße, Apple sortiert selbst ein. Es bleibt eine.

Bei der Häufigkeit drei: selten, häufig, regelmäßig geplant. Ein Sicherheitsalarm, der regelmäßig kommt, ist kein Alarm mehr, sondern ein Kanal. Es bleibt eine.

Und dann drei getrennte Textfelder: was die App ist, welche Mitteilungen kritisch verschickt werden, und warum das Recht nötig ist. Hätte ich meine vorbereitete Begründung als Block eingefügt, stünde alles in einem Feld und zwei wären leer geblieben. Das sieht ein Prüfer sofort.

> **Der häufigste Ablehnungsgrund bei diesem Antrag ist eine allgemein gehaltene Begründung. Ein Formular, das dreimal nachfragt, will genau das verhindern — und bestraft den, der es nicht vorher aufmacht.**

Um 11:54 war er raus. Bestätigung, Vorgangsnummer, und der Satz: wir melden uns.

Was ich mitnehme

Viermal an einem Tag wich das, was auf dem Bildschirm stand, von dem ab, was ich vorbereitet hatte. Der Ausweis, der nicht angeboten wurde. Der Preis in drei Währungen. Das Formular mit fünf Feldern statt drei. Und der Weg zum Antrag selbst, der nicht über das Kontaktformular führte, sondern über einen Reiter an der App-Kennung, den ich nicht kannte.

Keine dieser Abweichungen war ein Fehler von Apple. Alle vier waren Abweichungen zwischen einer allgemeinen Beschreibung und dem konkreten Fall.

> **Was auf dem Bildschirm steht, gilt. Nicht, was die allgemeine Seite sagt, nicht, was in der eigenen Anleitung steht, und schon gar nicht, was man vorgestern darüber notiert hat.**

Das ist derselbe Satz, der in diesem Projekt seit Wochen für Prüfsummen gilt: zuerst messen, dann behaupten. Er gilt offenbar auch für Formulare.

Nachtrag vom 28. August: Auf den Antrag ist bis heute keine Antwort gekommen. Fünf Tage, keine Mail, kein Zwischenstand. Er steht als eigener Punkt im Register und ist der einzige, an dem ich nichts tun kann außer warten.

ORIGINALBEITRAG · TEIL 20

Als die Prüfung ihre eigene Antwort abschrieb

24. August 2026

Zehn Punkte weg vom Register, sechs Lieferungen auf den Servern. Das ist der schöne Teil.

Der lehrreiche ist ein anderer. Dreimal an diesem Tag hat eine Prüfung bestätigt, was erwartet worden war — ohne irgendetwas gemessen zu haben. Und jedes Mal sah das Ergebnis besser aus als eine echte Messung.

Der Kasten, den ich falsch erwischt habe

Am Nachmittag war eine Lieferung für pelaron.de oben: siebzig Dateien, ein neues Stylesheet, eine mitgelieferte Schrift. Wie immer kamen zwei Kästen — oben der Befehl zum Kopieren, darunter die erwartete Ausgabe zum Vergleichen.

Am Terminal kam eine Ausgabe heraus, in der jede einzelne Zahl stimmte. Alle Prüfsummen, alle Größen, alle Zähler.

Sie war wertlos. Kopiert hatte ich den zweiten Kasten.

Bash hatte die Sollwerte als Befehle auszuführen versucht — daher hinter jeder Zeile ein **command not found**, und die Eingabezeile stand noch im Heimatverzeichnis statt im angelegten Prüfordner. Zwei Kleinigkeiten am Rand, die nichts mit den Zahlen zu tun hatten, waren das Einzige, was den Unterschied verriet.

> **Zwei gleich aussehende Kästen untereinander, beide grau hinterlegt, beide voller Dateinamen und Ziffern. Wer den falschen erwischt, bekommt eine Bestätigung geschenkt.**

Daraus ist die dreiundzwanzigste Arbeitsregel geworden, und sie ist kurz: Ein Prüfbefehl geht allein hinaus. Die Sollwerte kommen erst, nachdem die Ausgabe da ist. Dazu gibt der Befehl seither seinen eigenen Standort aus — *Ich bin in: …* als erste Zeile. Eine Messung, die nicht sagt, wo sie gemessen hat, ist dieselbe Falle in Grün.

Zwei Sollzahlen, geschätzt statt gezählt

Am Vormittag hieß es zu einer Lieferung: „und darunter achtmal eine 1." Es waren sieben Zähler. Der achte gibt gar keine Zahl aus, sondern einen Namen.

Am Nachmittag dasselbe noch einmal: „Source Serif 4 kommt dreimal in der Stildatei vor." Gemessen waren es vier — die Erwähnung in der Kommentarzeile war übersehen worden.

Beides ist folgenlos geblieben, weil mir die Abweichung aufgefallen ist statt dass ich sie überlesen hätte. Aber der Mechanismus ist derselbe wie beim Kasten oben, nur an der anderen Seite der Gleichung: Wer die Erwartung schätzt, hat beim Vergleich nichts in der Hand. Eine Abweichung heißt dann nicht mehr „die Datei ist falsch", sondern „irgendetwas von beidem ist falsch".

Und das ist keine Prüfung. Das ist ein Gefühl.

Prüfsummen aus einer Zeit, als die Frage noch anders lautete

Die dritte Sorte hat die meiste Zeit gekostet. Für dieselbe Lieferung waren morgens die Prüfsummen aller vier umgebauten Seiten notiert worden. Am Mittag kam die Entscheidung für die mitgelieferte Schrift dazu, und die änderte die Stildatei. Deren Prüfsumme steht in jeder der siebenundsechzig Seiten als Versionsnummer hinter dem Dateinamen.

Ein Stempellauf, alle Seiten angefasst, alle Prüfsummen neu. Nur die Notiz war von morgens.

Die Größen stimmten weiter, denn der Stempel ist immer gleich lang. Die Prüfsummen nicht. Ich hätte vier Abweichungen gefunden, die keine Fehler waren — der lästigste Fall, weil er Vertrauen kostet, ohne einen Mangel zu zeigen.

Die Regel dazu stand längst im Heft. Die Lücke war nicht das Messen, sondern die Reihenfolge. Also ein Zusatz: **Die Prüfsummen einer Lieferung werden erst gemessen, wenn keine Datei des Pakets mehr angefasst wird — und zwar am gepackten Ordner, nicht am Arbeitsstand.**

Ist es da? war die falsche Frage

Elf Zeichen in der App sollten durch gezeichnete Symbole ersetzt werden. Der Prüflauf danach meldete brav: „Bild an allen zehn Stellen vorhanden: ja."

Stimmte. Das Bildschirmfoto zeigte Symbole von zweiundsiebzig Pixeln, wo zweiundzwanzig hingehörten. Eine Zeichnung ohne Maßangabe füllt ihren Behälter aus — im Menü fällt das nicht auf, weil dort eine Regel die Größe setzt; in der unteren Leiste gibt es diese Regel nicht.

Die Prüfung hatte gefragt: *ist es da?* Die Frage wäre gewesen: *wie sieht es aus?*

Dreimal hat dasselbe Messen den Tag gerettet

Damit das kein Klagelied wird. Dieselbe Haltung hat heute dreimal eine Antwort umgedreht, bevor Arbeit hineinfloss.

Ein Umbau, der null Millisekunden gebracht hätte. Offen stand, gut dreißig Module vom starren Zeitplan auf den neuen Startpfad umzustellen. Vor dem Bauen gemessen, wann die Startfunktionen tatsächlich laufen: Der Startpfad hat einen Anker bei tausend Millisekunden; wer sich mit einer kürzeren Frist anmeldet, wird ohnehin vorher über seine eigene Schranke abgerufen. Bei vierhundert Millisekunden lief die Funktion nach 429 — mit und ohne Umstellung. Einunddreißig Dateien anzufassen hätte exakt nichts geändert.

Zweiundvierzig Zeichen, von denen dreizehn übrig blieben. Im Register stand seit Tagen, dass zweiundvierzig Sonderzeichen in hundertsechzig Fundstellen vom Betriebssystem kommen. Das war eine Zählung im Quelltext. Gemessen im laufenden Programm — alle zwanzig Ansichten geöffnet, jeden Textknoten durchgegangen — waren es dreizehn Zeichen an dreiundzwanzig Stellen. Der Punkt schrumpfte auf ein Viertel, bevor die erste Zeile geschrieben war.

Eine Schriftfrage, die keine war. Vier Stellen der Website standen in Georgia, das es auf Android nicht gibt. Es lag schon ein sauberer Schriftstapel mit ausgeschriebenen Ersatzfamilien fertig da — und wurde wieder verworfen, denn die Gattung „serif" löst auf denselben Geräten ohnehin in dieselben Familien auf. Vierundsechzig Dateien angefasst, nichts geändert.

Was wirklich passiert, zeigte erst die Messung an einem einzigen Wort. Das griechische *???????*, hundertfünfzehn Pixel groß, nur die Schrift getauscht:

```
DejaVu Serif, mit Namen angefordert        441px
der Georgia-Stapel (Georgia fehlt)         328px
serif, generisch                           328px
eine erfundene, nirgends vorhandene Schrift 328px
```

Bis zu vierunddreißig Prozent Unterschied — und die letzte Zeile ist der eigentliche Befund.

> **Ohne Georgia ist das Ergebnis von dem einer Schrift, die es gar nicht gibt, nicht zu unterscheiden. Das Gerät entscheidet, nicht die Seite.**

Die Antwort war dann kein besserer Stapel, sondern eine mitgelieferte Schrift: Source Serif 4, Regelschnitt, Latein und Griechisch, achtundzwanzigtausend Byte. Vorher gemessen, welche Schnitte die vier Stellen überhaupt brauchen — alle vier stehen auf Normalgewicht und aufrecht. Ein fetter oder kursiver Schnitt wäre totes Gewicht gewesen.

Zwei Bauformen, die eine geworden sind

Der größte Umbau des Tages hatte damit nichts zu tun und passt trotzdem dazu. Acht Reiterleisten in der App liefen über zwei verschiedene Konstruktionen — sieben über die eine, die Technikleiste über eine eigene. Beim Nachmessen zeigte sich: Es waren nicht zwei Schreibweisen für dasselbe, sondern zwei verschieden gute Leisten. Die eigene hatte keine Pfeiltastenbedienung, merkte sich den gewählten Reiter nicht über das Schließen hinaus und schrieb die Adresszeile nicht mit. Dazu ein Widerspruch, den vorher niemand bemerkt hatte: Im Markup war ein Reiter als offen markiert, im Skript ein anderer — und das Skript gewann.

Dieselbe Geschichte auf der Website. Vier Seiten hatten ein eigenes Stylesheet, eine eigene Kopfleiste, ein eigenes Logo im Quelltext und ein eigenes Menüskript. Sechsundzwanzig Kilobyte doppeltes CSS sind dabei verschwunden. In einer der Dateien stand der Grund im eigenen Kommentar:

> **„Dies ist eine Zwischenlösung. Punkt 13 des Aufgabenhefts führt die vier Seiten auf die gemeinsame Vorlage zurück; danach fällt diese Datei weg."**

Sie ist heute weggefallen.

Was ich mitnehme

Die drei Fälle vom Anfang sehen verschieden aus und sind derselbe: eine Prüfung, deren Antwort schon feststand, bevor sie gestellt wurde. Einmal, weil die Antwort abgeschrieben wurde. Einmal, weil die Erwartung geraten war. Einmal, weil sie aus einer Zeit stammte, in der die Frage noch anders lautete.

Das Tückische ist, dass alle drei gut aussehen. Ein Fehler meldet sich. Eine Bestätigung, die keine ist, meldet sich nicht.

> **Der Satz hieß: zuerst messen, dann behaupten. Er heißt jetzt: zuerst messen, dann behaupten — und vorher nachsehen, ob die Messung überhaupt gemessen hat.**

Nachtrag vom 28. August: Regel 23 ist die, die von allen siebenundzwanzig am häufigsten gebraucht wurde. Seit dem 24. August kommt jeder Prüfbefehl allein, nennt seinen Standort und enthält kein Ausrufezeichen. Und trotzdem ist mir heute noch dreimal eine Prüfung untergekommen, die nichts gemessen hat und trotzdem eine Zahl ausgab. Die Regel verhindert das Abschreiben. Sie verhindert nicht, dass man die falsche Frage stellt.

ORIGINALBEITRAG · TEIL 21

Die Regel gab es längst

25. August 2026

An Bord tippt man mit nassen Fingern. Bei Seegang, mit Handschuhen, gegen die Sonne. Deshalb steht in meinen Anforderungen seit Monaten, dass kein Bedienfeld kleiner als 44 Punkte sein darf, und dass Text genug Kontrast haben muss.

An diesem Tag habe ich gelernt, dass das Aufschreiben einer Regel und das Einhalten einer Regel zwei verschiedene Arbeiten sind — und dass ich die zweite nie gemacht hatte.

Zehn Punkte sind vom Register verschwunden, neun Lieferungen sind auf den Server gegangen. Das ist die Bilanz. Interessanter ist, was beim Messen herauskam: Vier Befunde, die nichts miteinander zu tun zu haben schienen, hatten am Ende dieselbe Form.

Vier Befunde, eine Form

Der erste war ein Überlauf nach rechts. Bei 1024 Bildschirmpunkten ragte die Startseite 213 Punkte über den Rand hinaus. Sehen kann man das nicht. Man merkt es nur daran, dass sich die Seite seitlich schieben lässt.

Die Ursache war eine Zeile, die man tausendfach so schreibt. Sie bedeutet „nimm den verfügbaren Platz" — heißt aber in Wahrheit „nimm mindestens so viel, wie der Inhalt braucht". Passt der Inhalt nicht, wächst die Spalte über ihren Behälter hinaus, statt umzubrechen.

Nachgemessen über vier Bildschirmbreiten waren es nicht zwei Überläufe, wie im Heft stand, sondern vier. Und die Reparatur vom Vortag hatte in einem Block gestanden, der nur unterhalb von 560 Punkten gilt. Sie hat den Fall behoben, an dem gemessen worden war. Nicht die Ursache.

Der zweite waren Bedienfelder unter 44 Punkten. Im Heft stand seit Tagen die Zahl 25, mit dem Vermerk, sie sei nicht nachmessbar. Sie war es inzwischen. Und sie war in beide Richtungen falsch.

Das angebliche kleinste Element mit 13 × 13 Punkten war eine Messung am falschen Ding — bei einem Ankreuzfeld ist auch die Beschriftung daneben anklickbar, und die ist hoch genug. Dafür fehlten neunzehn Befunde, die niemand gezählt hatte: die gesamte Seitennavigation. Knöpfe 42 Punkte, Gruppenüberschriften 38.

Und dann kam die Zahl, die mir den Tag verdorben hat. Dieselbe Datei setzt an **neunundzwanzig** Stellen ausdrücklich 44 Punkte als Mindesthöhe. An **sechzehn** Stellen setzt sie etwas darunter.

Die Regel war da. Sechzehn Stellen hielten sich nicht daran.

Der dritte war Kontrast. Ein Prüfläufer meldete zwanzig Stellen unter dem Grenzwert. Nachgerechnet waren es zwanzig Stellen, aber nur fünf Farbpaare, und davon drei echte. Die schlechteste: 3,05 zu 1, wo 4,5 verlangt sind — ein Datum in der Törnübersicht.

Die drei Farben standen in zwei Dateien, die nicht die Hauptstildatei sind. In der Hauptstildatei steht seit dem Sommer eine dunklere Variante der Markenfarbe, eigens angelegt, weil die helle für Text zu schwach war. Und ein Grauton, der drei Tage vorher an 159 Stellen nachgezogen wurde, aus demselben Grund.

Zwei Dateien haben von beidem nie etwas mitbekommen. Diesmal war die Grenze keine Bildschirmbreite, sondern eine Dateigrenze.

Der vierte war der Schwojkreis in der Ankerwache, der aus seiner eigenen Karte lief. Ursache: dieselbe wie beim ersten, nur eine Ebene tiefer.

Was das gemeinsam hat

Vier Symptome, ein Muster: **Es fehlte nirgends die Regel. Es fehlte überall die Reichweite.**

Das ist etwas anderes als Nachlässigkeit, und es ist unangenehmer. Wer eine Regel aufstellt, hat das Gefühl, das Problem erledigt zu haben. Der Unterschied zwischen „wir schreiben 44 Punkte vor" und „nirgends steht weniger als 44" ist genau die Arbeit, die dazwischenliegt.

Und die sieht niemand, weil sie aus Nachzählen besteht.

Keiner der vier Befunde ist dadurch gefunden worden, dass jemand hingesehen hat. Alle vier kamen daher, dass ein Werkzeug alle zwanzig Ansichten in zwei Bildschirmbreiten durchgeklickt und jede Zahl aufgeschrieben hat.

Das Werkzeug, das dreimal korrigiert werden musste

Dieses Werkzeug war am Vortag entstanden und wurde am Vormittag eingesetzt. Am Nachmittag stellte sich heraus, dass es in drei von sechs Prüfungen nichts gemessen hatte.

Nicht „nichts gefunden". **Nichts gemessen.**

Die App versteckt ihre Ansichten über eine Stilregel. Das Merkmal, nach dem das Werkzeug suchte, kommt in der ganzen Anwendung nicht vor. Also traf es immer die erste Ansicht im Dokument — fast immer eine unsichtbare. Deren Elemente haben die Größe null und wurden von der eigenen Sichtbarkeitsprüfung sauber aussortiert.

„Kontrast: nichts gefunden" war also keine Auskunft. Es war eine leere Zeile. Und sie stand in drei Lieferberichten.

Der zweite Fehler: Gemessen wurde, ohne die Bewegung abzuschalten. Ein Knopf, der beim Antippen auf 48 Punkte wächst, wird kurz danach als 42 gemessen.

Der dritte: Bei halbdurchsichtigem Hintergrund nahm die Rechnung die erste nicht ganz durchsichtige Schicht als Grund. Fünf Prozent Weiß auf dunklem Marineblau ist dunkel, nicht weiß — der Zähler meldete weißen Text auf weißem Grund. Zwei der zwanzig Kontrastbefunde waren auf diese Weise erfunden.

Dreimal an einem Tag korrigiert, und jedes Mal in dieselbe Richtung:

> **Das Werkzeug hat mehr behauptet, als es gemessen hat.**

Dass es nach der Korrektur sofort etwas fand, ist übrigens der einzige Beleg dafür, dass es vorher nichts gemessen hat. „Nichts gefunden" bedeutet erst etwas, wenn dasselbe Werkzeug an anderer Stelle etwas findet.

Eine Reparatur, die eine andere kaputtgemacht hat

Am Nachmittag kam für die Überläufe eine Regel dazu, die Knopfzeilen umbrechen lässt. Damit das funktioniert, müssen die Knöpfe darin unter ihre eigene Breite schrumpfen dürfen.

Zwei Stunden später, beim Nachmessen der Zielgrößen, tauchte ein Knopf auf, der 41 Punkte breit war statt 44.

Es war die eigene Zeile von vorhin. Sie hat genau das getan, was sie sollte, und dabei etwas anderes mitgenommen.

Gefunden hat das nicht das Nachdenken. Gefunden hat es die nächste Messung, zwei Stunden später, an einer Stelle, die mit dem Überlauf nichts zu tun hatte.

Und dann meine Terminalausgabe als Foto

Beim Prüfen einer Lieferung mit dreißig Dateien habe ich die Terminalausgabe als Bildschirmfoto geschickt, wie immer.

Neunundzwanzig Prüfsummen stimmten. Bei der dreißigsten meinte die KI, eine Abweichung zu sehen: **d6e9** gegen **de69**, in einer Schrift, in der beides gleich aussieht.

Sie hat nachgefragt statt behauptet — und die Datei war richtig. Falsch war das Ablesen.

> **Ein Bildschirmfoto ist keine Messung. Wo eine Zahl zählt, muss sie als Text kommen.**

Was an dem Tag nicht gemacht wurde

Ein Befund ist offen geblieben: 16 Punkte Höhenversatz zwischen zwei Bedienelementen in einer Zeile. Er ließ sich außerhalb des Prüfwerkzeugs nicht erzeugen — auch nicht, indem dessen Durchlauf nachgebaut wurde.

Also wurde nicht repariert, sondern das Werkzeug so erweitert, dass es beim nächsten Lauf sagt, welche zwei Elemente es sind.

Auf Verdacht zu liefern wäre derselbe Fehler wie eine Zahl im Heft, die niemand nachgemessen hat. Nur schneller.

Was ich mitnehme

Der Satz, den ich mir seit Wochen aufschreibe, hieß zuletzt: zuerst messen, dann behaupten — und vorher nachsehen, ob die Messung überhaupt gemessen hat.

Jetzt kommt der Teil dazu, der die vier Befunde verbindet:

> **Eine Regel, die es gibt, ist noch keine Regel, die gilt.**

Zwischen beidem liegt eine Zählung. Wer sie nicht macht, hat am Ende neunundzwanzig Stellen, die sich daran halten, und sechzehn, die niemand je gefragt hat.

ORIGINALBEITRAG · TEIL 22

Wer entscheidet das

26. August 2026

Fünf Fragen an einem Tag, die nichts miteinander zu tun hatten. Mann über Bord. Ankeralarm auf zwei Geräten. Android. Aufbewahrungsfristen. Und ein Satz im medizinischen Katalog.

Am Abend war es dieselbe Frage: **Wer darf das eigentlich entscheiden — und tut die Software so, als könnte sie es?**

Der Plotter navigiert, die App schreibt mit

Angefangen hat es mit einer Funktion, die es noch nicht gab: Mann über Bord. Die Datenstruktur liegt seit Monaten im Programm — Art des Vorfalls, betroffene Person, Zustand. Es fehlte nur der Knopf.

Die naheliegende Frage war, ob eine Rückpeilung dazugehört. Also: die Richtung zurück zur Person im Wasser.

Die Antwort war nein, und die Begründung hat mir eingeleuchtet. Das macht der Kartenplotter. Der kennt Kreiselkurs, Strom und Kartenbild. Ein zweites Gerät, das eine zweite Zahl anzeigt, macht eine Rettung nicht sicherer.

> **Eine App, die sich im Notfall neben ein Gerät stellt, das es besser kann, ist nicht hilfreich. Sie ist im Weg.**

Was bleibt, ist genau das, was der Plotter nicht tut: mitschreiben.

Dazu kam mir dann noch etwas, das vorher niemand auf dem Zettel hatte, und der Gedanke dahinter ist ganz einfach: **Irgendwann muss der Notfall auch als beendet gelten.** Sonst fehlt der Logbucheintrag — und an dem hängt alles. Der Törn, die Dauer, die Auswertung, das Dokument, das man aufbewahren muss. Ein Vorfall ohne Ende ist in einem Logbuch kein Vorfall, sondern eine Lücke.

Also braucht es nach einem Mann-über-Bord einen zweiten Knopf: die Person ist wieder an Bord.

Und der hat beim Bauen eine Entscheidung erzwungen, an die ich selbst nicht gedacht hatte. Nach einem MOB ist viel zu tun. Die Person ist vielleicht verletzt. Man drückt den Knopf also spät — eine Viertelstunde später, eine halbe. Stünde dann die Druckzeit im Logbuch, wäre sie falsch. Und das Logbuch ist ein Dokument, das aufbewahrt werden muss.

Die App schlägt jetzt die aktuelle Uhrzeit vor und lässt sie ändern. Die Dauer rechnet sie weiter aus dem echten Auslösezeitpunkt. Wer die Uhrzeit korrigiert, korrigiert damit nicht die Wahrheit über die Dauer.

Und es gibt ein Ende, das nicht „wieder an Bord" heißt. Freier Text, keine Auswahlliste. Die App benennt so einen Ausgang nicht. Ohne das bliebe entweder ein Vorgang, der ewig offen steht — oder ein Logbucheintrag, der nicht stimmt, weil es der einzige Knopf war.

Zwei Geräte, eine Wache

Die zweite Frage betraf die geplante native App. Das iPad stellt den Ankeralarm, das iPhone soll nachts wecken. Was heißt dann „unabhängig voneinander"?

Ich hatte die Frage falsch herum gestellt: *Über welchen Weg erreicht der Alarm das zweite Gerät?*

Richtig ist: über gar keinen. Ein Gerät, das nachts weckt, muss selbst messen. Eigenes GPS, eigener Ankerpunkt, eigener Ton. Alles andere hängt an fremder Infrastruktur, und kein Zustelldienst sagt dir zu, wann eine Nachricht ankommt. Oder ob.

Wenn aber beide Geräte ohnehin selbst wachen, trägt die Verbindung zwischen ihnen nur noch zwei Hausmitteilungen: „Ankerpunkt versetzt" und „quittiert". Beides ist selten, bei beidem steht ein Mensch daneben, beides kann man zur Not zweimal von Hand machen.

Daran hing die Entscheidung, und der Satz dazu ist einfach:

> **Jeder Fehlerfall ist dann ein Fehlalarm, nie ein verpasster.**

Bricht die Verbindung ab, läutet ein Gerät weiter, obwohl das andere quittiert hat. Oder es rechnet gegen einen Ankerpunkt, der nicht mehr gilt. Beides laut. Kommt das Boot in den Kreis zurück, sehen es beide selbst.

Eine Wache, die zum Fehlalarm hin ausfällt, ist lästig. Eine, die zum Schweigen hin ausfällt, ist gefährlich.

Eine Zahl statt einer Stimmung

Die dritte Frage war Android. Sie stand seit Wochen im Register als „ob überhaupt — und nach welchem Kriterium". Das Kriterium fehlte.

Beim Nachsehen kam heraus, dass es sich gar nicht bestimmen lässt. Weder die Website noch die App enthalten irgendeine Zählung. Kein Analysewerkzeug, kein Zählpixel. Die einzige Verbindung nach draußen ist der Wetterdienst.

Das ist eine Haltung, keine Nachlässigkeit. Sie hat nur den Preis, dass ich die Frage nach dem Anteil nicht beantworten kann.

Der Ausweg braucht keinen Einbau: In den Zugriffsprotokollen des Servers steht die Gerätekennung ohnehin. Vier Wochen auswerten, Schwelle **vorher** festschreiben, fertig. Die Zahl steht jetzt im Anforderungspapier.

Vorher, nicht hinterher. Das ist der ganze Unterschied zwischen einem Kriterium und einer nachträglichen Begründung.

Drei Jahre, die ich nicht zusagen kann

Am Vortag hatte ich nebenbei gesagt, der Logbucheintrag sei rechtlich vorgeschrieben. Das wollte ich nicht auf einer Annahme stehen lassen, also habe ich kurz einen befreundeten Rechtsanwalt angerufen. Zwei Minuten Telefon, und die Zahl stand: **drei Jahre** ab dem Tag der letzten Eintragung. Das amtliche Merkblatt sagt dasselbe.

Manchmal ist die schnellste Messung ein Anruf.

Drei Jahre stehen gegen sieben Tage. Denn eine Webanwendung, die man nicht installiert hat, verliert auf iOS ihre Daten nach sieben Tagen ohne Benutzung.

Damit ist „echte lokale Datenbank" in den Anforderungen keine Bequemlichkeit mehr, sondern das Rückgrat. Und der Grund dafür stand bis dahin nirgends. Wer eine Anforderung später abwägt, wägt sonst ohne ihren Grund ab.

Die Entscheidung, was PELARON daraus zusagt, ist gegen die große Zusage gefallen. Drei Jahre sichere ich nicht zu. Ein verlorenes, zurückgesetztes oder verkauftes Gerät kann keine Software verhindern.

Zugesagt wird stattdessen, was trägt: Es löscht nichts von selbst. Es erinnert an die Sicherung. Es gibt alles heraus, jederzeit, ohne Konto und ohne Netz. Und es sagt dir, dass die Aufbewahrung deine Sache ist — an der Stelle, an der es zählt, nicht im Kleingedruckten.

Die Frage, die den Tag beendet hat

Im medizinischen Katalog steht bei „Mann über Bord" ein Schritt: *Auch nach scheinbarer Erholung medizinisch beurteilen lassen.*

Mir war aufgefallen, dass drei der vier Schritte Handgriffe während der Rettung sind und nur dieser eine echte Nachsorge. Ich habe gefragt, ob der Katalog getrennt gehört.

Die Antwort hat meine Frage umgestellt, und zwar zu Recht. Wenn jemand über Bord geht, setzt man einen Notruf ab und übergibt die Person den Seenotrettern — ob man sie selbst herausgeholt hat oder nicht. Kein Skipper traut sich zu, medizinisch zu beurteilen, ob mit dem Geretteten alles in Ordnung ist. Und eine Rechtsfrage ist es obendrein.

Der Schnitt war also ein anderer als gedacht. Nicht „während" gegen „danach". Sondern **was du tust** gegen **wer entscheidet.**

Der Satz im Katalog ist in der Sache richtig und in der Form falsch. Er nennt keinen Handelnden. In einer Liste, die dem Skipper vorgelegt wird, liest sich „beurteilen lassen" wie eine Aufgabe für ihn — und die einzige Person an Bord, die sie erledigen könnte, ist er selbst.

Beim Nachgehen kam dann der eigentliche Fund. Die App enthält **an keiner Stelle** einen Hinweis auf funkärztliche Beratung. Null Fundstellen, über alle Dateien gemessen.

Dabei steht die Nummer, über die sie läuft, seit Monaten im Programm: Die App wählt die zuständige Seenotleitstelle nach Position aus. Und genau dort sitzt rund um die Uhr medizinisches Personal.

Der Notfallbildschirm hat seit dem Abend einen eigenen Kasten dafür. Blau, nicht rot, und unter den Notrufwegen — ärztlicher Rat ist kein Notruf, und im Ernstfall muss auf einen Blick klar bleiben, welcher Kasten zuerst gilt.

> **PELARON gibt keine medizinische Einschätzung ab und verlangt keine von dir. Ob jemand unverletzt ist, entscheidet nicht der Schiffsführer.**

Zweimal hat ein Prüfbefehl nichts geprüft

Zum Handwerklichen. Jede Lieferung geht mit einem Prüfbefehl hinaus, der nach dem Hochladen am Server nachzählt. Zweimal hat der an diesem Tag nichts gemessen und trotzdem eine Zahl geliefert.

Am Vormittag suchte er nach einem Wort in Behelfsschreibweise — *ausgeloest* statt *ausgelöst* —, weil beim Tippen des Prüfbefehls derselbe Fehler gemacht wurde, den er finden sollte. Ergebnis: null Treffer für eine Datei, die in Ordnung war.

Am Abend suchte er nach einem ganzen Satz, den der Quelltext über zwei Zeichenketten umbricht, weil die Zeile sonst zu lang wäre. Im Browser steht der Satz vollständig da. Das Suchwerkzeug sieht ihn nicht, weil es Zeilen sieht und keine zusammengesetzten Texte.

Zwei verschiedene Ursachen, dieselbe Form: **ein Prüfbefehl, der eine andere Vorstellung von der Datei hat als die Datei.** Beide Male hat es der Probelauf gefunden, nicht das Nachdenken.

Und einmal hat das Hinsehen gefunden, was die Messung nicht sehen

Ein dritter Fall gehört dazu, weil er die Grenze in die andere Richtung zeigt.

Eine Messung lief über alle sichtbaren Bedienelemente in allen zwanzig Ansichten und meldete: alles in Ordnung. Das Kontrollbild danach zeigte einen deaktivierten Löschen-Knopf mit weißer Schrift auf grauem Grund. Unlesbar.

Der Grund: Dieser Knopf war in keiner der zwanzig Ansichten sichtbar. Er lebt in einem Dialog.

Für die Frage, wie ein Knopf im deaktivierten Zustand aussieht, spielt Sichtbarkeit aber keine Rolle. Die Regel gilt auch für den Knopf in einem geschlossenen Dialog. Gezählt wird seitdem über alle 222 Marken statt über die 121 sichtbaren.

Was ich mitnehme

Der Satz vom Vortag hieß: eine Regel, die es gibt, ist noch keine Regel, die gilt. Heute kommt einer dazu, und der ist unbequemer, weil er nicht die Software betrifft, sondern die Haltung dahinter.

> **Software soll nicht so tun, als könnte sie entscheiden, was sie nicht entscheiden kann.**

Nicht die Richtung zur Person im Wasser, wenn ein besseres Gerät sie kennt. Nicht die Aufbewahrung über drei Jahre, wenn ein Gerät verloren gehen kann. Nicht den Anteil einer Plattform, wenn nichts gezählt wird. Und schon gar nicht den Zustand eines Menschen.

Was sie kann, ist erstaunlich viel. Festhalten, was wann geschehen ist. Erinnern, wenn etwas offen bleibt. Sagen, wen man fragen kann.

Und den Rest ehrlich als das benennen, was er ist: die Entscheidung von jemand anderem.

ORIGINALBEITRAG · TEIL 23

Der Katalog, der mich beurteilen lässt, was ich nicht beurteilen kann

27. August 2026

Der Satz ist mir am Abend vorher gekommen, beim Durchklicken der medizinischen Notfallhilfe in der eigenen App. Ich habe ihn genau so aufgeschrieben:

> **Der Katalog lässt den Skipper beurteilen, was er nicht beurteilen kann.**

Fünfzehn Einträge stehen da drin, von Seekrankheit bis Bewusstlosigkeit. Bei „Mann über Bord" steht als einer der Schritte:

Auch nach scheinbarer Erholung medizinisch beurteilen lassen.

Klingt vernünftig. Ist es auch — an Land. Dreißig Seemeilen vor der Küste ist es eine Aufgabe an jemanden, der sie nicht erfüllen kann. Ich bin kein Arzt. Ich habe keine Praxis in Reichweite. Und der Satz sagt mir nicht, wen ich stattdessen fragen soll.

Beim Nachdenken darüber ist mir der Unterschied klar geworden, um den es eigentlich geht: **was du tust** gegen **wer entscheidet.** Eine Notfallhilfe darf mir das eine abverlangen. Das andere muss sie abgeben.

Sechsmal 112, keinmal Funk

Die KI hat daraufhin alle fünfzehn Einträge durchgezählt, danach, was sie als Weg nach draußen nennen.

Sechsmal steht dort 112. Keinmal UKW Kanal 16.

Das ist kein Schönheitsfehler. Die 112 erreicht in Mobilfunkreichweite eine Leitstelle an Land — und kein Schiff in der Nähe. Bei den meisten Seenotfällen ist das nächste Schiff die schnellste Hilfe. Der Katalog war für Land geschrieben.

Und dann kam die Stelle, an der ich kurz geschluckt habe. Auf dem Notfallbildschirm derselben App steht seit Monaten, direkt unter dem 112-Knopf:

> **Nur in Mobilfunkreichweite. Auf See ist UKW der sichere Weg — ein Handy erreicht kein Schiff in der Nähe.**

Zwei Bildschirme weiter sagt der Katalog sechsmal „sofort 112 anrufen". Ohne diesen Satz.

Die richtige Auskunft stand in meiner App. Nur nicht dort, wo man entscheidet.

Dasselbe gilt für sechs Sätze, die Hilfe verlangen und nicht sagen wie — „ärztlich abklären", „professionelle Hilfe anfordern", „medizinischen Rat einholen". An Land ist jeder davon vollständig: man fährt zum Arzt. Auf See ist keiner davon vollständig. Und die Antwort gibt es in der App seit ein paar Tagen: funkärztliche Beratung über die Seenotleitstelle, rund um die Uhr. Der Katalog wusste nichts davon.

Geändert wurde der Weg zur Hilfe, nicht die Erste Hilfe. Die Handgriffe bleiben, wie sie sind. Daran rührt niemand von uns, und das ist auch richtig so.

Einmalhandschuhe

Ein Detail am Rande, das ich mir gemerkt habe, weil es so schön dumm ist.

Die erste Zählung meldete zwei Einträge, in denen UKW oder DSC vorkommt. Der eine stimmte. Der andere war der Eintrag zur Schnittwunde — und der Treffer lag mitten im Wort **Einmalha-ndsc-huhe**.

Eine Suche nach Buchstabenfolgen findet in deutschen Wortungetümen Sachen, die es nicht gibt. Nachgezählt mit richtigen Wortgrenzen: null von fünfzehn.

Die Zahl, die oben steht, ist die zweite Zählung. Nicht die erste.

Ein Knopf, der prüft, was er nicht prüft

Beim Durchsehen einer ganz anderen Sache fiel ein Knopf auf: **Verbindungen prüfen**, in den technischen Einstellungen.

Er prüft keine Verbindung. Er prüft, ob die eingetippte Adresse richtig geschrieben ist.

Und selbst wenn er wollte, könnte er nicht: Von einer verschlüsselt ausgelieferten Seite kommt eine unverschlüsselte Verbindung zu einem Gerät im Bordnetz gar nicht erst zustande. Der Browser lässt es nicht zu. Der Knopf konnte nie halten, was sein Name verspricht.

Denselben Befund gab es vor ein paar Tagen schon einmal, an einer anderen Stelle. Die zweite ist vier Tage später aufgefallen als die erste.

Wer etwas repariert, sollte am selben Tag nachsehen, wo dasselbe noch steht. Sonst findet man es wieder — nur später.

Der Nachmittag, an dem der Nachweis misslang

Seit dem 22. August steht ein Punkt auf der Liste, der mir wichtiger ist als die meisten: eine Sicherung auf einem leeren zweiten Gerät zurückspielen und nachzählen. Ich behaupte an mehreren Stellen, die Daten gehören dem Nutzer. Belegt war das nie.

Der Versuch am iPad hat drei Befunde geliefert und keinen Nachweis.

Der erste war den ganzen Nachmittag wert. Auf dem iPad ließ sich keine einzige Datei auswählen. Alles grau, sogar fremde PDF.

Die Ursache lag in meiner eigenen App. Die Einspielfelder sagen dem Gerät, welche Dateitypen sie annehmen — und ich habe meinen Sicherungen eigene Endungen gegeben, **.pelaron**, **.pelaronfotos**, **.pelarondocs**. iPadOS kennt die nicht. Was es nicht zuordnen kann, blendet es aus.

Also: **Eine Sicherung, die PELARON selbst erzeugt, ließ sich auf einem iPad nicht einspielen.** Auf genau dem Gerät, für das die native App gedacht ist. Ohne Entwicklerkonsole hatte ein Nutzer keine Chance.

Belegt haben wir es, indem die Angabe über den Web-Inspector entfernt wurde. Danach ließ sich dieselbe Datei sofort auswählen. Vorher unmöglich, nachher möglich, sonst nichts geändert.

Der zweite und dritte standen auf einem Bildschirm, untereinander. Oben:

Kein Konto eingerichtet. Alles ist auf diesem Gerät gespeichert; es gibt nichts abzugleichen.

Zehn Zentimeter tiefer: *3 Konflikte auf einmal auflösen* — mit Versionsnummern.

Ein Kasten sagt, es gebe keine Cloud. Der nächste bietet an, drei Datensätze aus ihr zu übernehmen.

Und der schwerere von beiden: Der hervorgehobene, orange Knopf hieß **Alle: Cloud übernehmen**. Wer gerade eine Sicherung eingespielt hat, drückt den auffälligen — und macht sie damit unbemerkt zunichte. Der unauffällige weiße war der, der die Wiederherstellung bewahrt.

> **Der betonte Knopf war der zerstörende.**

Warum es trotzdem kein Beweis war

Am Ende standen auf dem iPad einundfünfzig Einträge gegen einundfünfzig auf dem Mac. Neunundvierzig davon aufs Stück gleich. Zwei waren größer, keiner kleiner. Verloren war nichts.

Nur: beide Abweichungen gingen nach oben. Es war *mehr* da, als in der Sicherung stand. Woher, blieb offen.

Der Grund dafür war ein Fehler der KI, und sie hat ihn deutlich aufgeschrieben: Sie hatte den Ausgangszustand des iPads im privaten Fenster gezählt, gemerkt, dass das die falsche Umgebung ist, eine Wiederholung angefordert — und ist dann ohne sie weitergegangen.

Ohne gemessenen Ausgangszustand ist ein Endzustand kein Beweis. Das ist keine Feinheit. Das ist der ganze Punkt.

Und dann noch mein eigener Fehler

Spät am Nachmittag habe ich die Dateien einer Lieferung hochgeladen — auf pelaron.de statt auf pelaron.app.

Beide Adressen liegen im selben Webspace, direkt nebeneinander, und die Ordner heißen fast gleich. Die Startseite von pelaron.de war überschrieben und musste aus dem Stand vom Vorabend zurückgeholt werden.

Passiert ist es nicht aus Schlamperei. In der Anleitung stand nirgends, wohin. Seitdem steht die Zieladresse in der ersten Zeile jeder Lieferanleitung, groß, vor allem anderen.

Wer eine Anweisung gibt, nennt ihren Ort. Klingt banal. Hat einen halben Tag gekostet.

Was der Tag gezeigt hat

Sechs Lieferungen, sieben neue Punkte, ein misslungener Nachweis.

Der Faden war nicht „etwas fehlte". Es war: **etwas war da und stand am falschen Ort.** Der Satz über UKW stand in der App, nur nicht im Katalog. Der Befund über den Knopf stand seit Tagen im Heft, nur für eine andere Datei. Der richtige Ausgangszustand war angefordert, nur nicht abgewartet. Und die Zieladresse der Lieferung war klar, nur nicht aufgeschrieben.

Es ist leichter, etwas Neues zu bauen, als das Vorhandene dorthin zu bringen, wo es gebraucht wird.

Das ist keine Ausrede. Das ist die Arbeit.

ORIGINALBEITRAG · TEIL 24

Vier Sachen, die mir aufgefallen sind, während die Maschine gemessen hat

28. August 2026

Freitagmorgen, iPad am Kabel, Kaffee daneben. Der Plan war klein: einmal beweisen, dass meine Daten von einem Gerät aufs andere kommen. Es wurde der längste Arbeitstag, den dieses Projekt bisher hatte.

Ich baue PELARON für Segler. Zuerst für mich — mein Logbuch, meine Törns, die Fotos vom Warnemünder Segel-Club. Irgendwann für andere mit demselben Problem: dass an Bord alles auf Zetteln steht und nichts zusammenpasst.

Deshalb ist die Sache mit den Daten keine Fleißaufgabe. Wenn ich einem anderen Segler sage „deine Daten gehören dir", dann muss ich das einmal gezeigt haben. Nicht behauptet. Gezeigt.

Das Werkzeug, das kaputtgemacht hätte, was es messen soll

Am Vortag war der Versuch schon einmal schiefgegangen, aus einem dummen Grund: Ich hatte den Ausgangszustand des iPads im privaten Fenster gezählt. Falsche Umgebung. Ohne Ausgangszustand beweist ein Endzustand nichts.

Also von vorn. Mac zählen, sichern, iPad leeren, iPad zählen, einspielen, wieder zählen.

Und dann hat die KI ihr eigenes Zählwerkzeug noch einmal geprüft, bevor sie es auf das leere iPad losließ — und es fiel durch. Das Ding hätte beim Zählen die Fotodatenbank angelegt. Leer, aber angelegt. Und weil sie dann schon existiert, hätte die App ihre eigene nicht mehr bauen können.

Das Einspielen der Bilder wäre gescheitert. An der Messung, nicht an der App.

> **Ein Messgerät gehört an dem Zustand geprüft, den es messen soll. Nicht an dem, der gerade da ist.**

Am Vortag war es nicht aufgefallen, weil da die Datenbanken schon existierten. Der Fehler steckte nur im Fall „gibt es noch nicht" — also genau dort, wo das Werkzeug gebraucht wurde.

Der Beweis

Das iPad war leer. Siebzehn Einträge im Speicher, alle von der App selbst angelegt, kein einziger mit Daten von mir. Kein Logbuch, keine Fotos, keine Dokumente.

Dann kam beim Öffnen etwas, mit dem ich nicht gerechnet hatte: ein Begrüßungsdialog. *Wie möchtest du anfangen?* Und die dritte Zeile: **Vorhandene Sicherung einspielen.**

Ich habe die Tür selbst gebaut und in dem Moment vergessen gehabt, dass es sie gibt. Ganz gut so — jetzt weiß ich, dass sie funktioniert, wenn man nicht daran denkt.

Passphrase, Datei, Seite lädt neu. Nochmal Passphrase, Bilder, Dokumente.

Danach: einundfünfzig Einträge wie auf dem Mac, fünfunddreißig Logbucheinträge, fünf Bilder, drei Dokumente. Alles da, nichts doppelt, nichts weg.

Eine Sache hatte die KI vorher schriftlich festgelegt, und das war klug. Am Vortag hatte das iPad mit **sechs** Bildern geendet, obwohl der Mac fünf hatte, und niemand wusste warum. Also stand vorher da: *genau fünf. Kommen sechs heraus, legt das Einspielen Bilder an, statt sie zu ersetzen.*

Es waren fünf. Die sechste vom Vortag kam aus dem Altbestand des Geräts.

Eine Vorhersage, die vor der Messung feststeht, kann man hinterher nicht zurechtbiegen. Eine, die man danach formuliert, immer.

Und dann stand da eine Vier

In der Seitenleiste, neben „Logbuch": **4**.

Ich habe fünfunddreißig Einträge. Ich hatte gerade eine Sicherung eingespielt, in der fünfunddreißig drin sind. Und da steht vier.

Das ist der Moment, für den man so einen Test macht. Nicht wegen der Zahl — wegen des Gefühls. Wenn mir das als normaler Nutzer passiert, nach dem Wechsel auf ein neues iPad, dann denke ich: **meine Daten sind weg.** Und dann traue ich mich nicht mehr, irgendetwas anzufassen.

Ich habe aufs Logbuch getippt. Fünfunddreißig von fünfunddreißig. Alles da.

Die KI hat zuerst vermutet, die Sicherung sei gar nicht eingespielt worden. Das war falsch, und sie hat es selbst widerlegt, bevor daraus eine Änderung wurde.

Was wirklich dahintersteckte, hat mich dann fast gefreut. Die Seitenleiste holt sich ihre Zahl, indem sie nach einem Speicher sucht, in dessen Namen irgendwo „log" vorkommt. Der erste, den sie findet, ist nicht das Logbuch. Es ist mein **Wartungs-Servicebuch**. Vier Einträge.

Die Wartung hat sich als Logbuch ausgegeben.

Und das Ärgerlichste daran: Welcher Speicher zuerst gefunden wird, hängt an der Reihenfolge, in der der Browser sie aufzählt — und die ist nicht festgelegt. Auf dem Mac war die Anzeige über Tage richtig. Auf dem frisch wiederhergestellten iPad falsch. Weil das Wiederherstellen die Einträge in anderer Reihenfolge schreibt.

Ein Fehler, der immer falsch ist, wird gefunden. Einer, der meistens richtig ist, nicht.

Die Spur nach Afrika

Nachmittags habe ich eine Testspur eingelesen, vier Punkte vor Warnemünde. Auf der Karte lief eine dicke blaue Linie von der Ostsee schräg nach unten aus dem Bild.

Ich habe geschrieben: *die Spur ist wahrscheinlich komplett falsch.* Die KI war in dem Moment mit Dateiwählern beschäftigt.

War sie auch. Achtundzwanzig von meinen sechsunddreißig Logbucheinträgen haben keine Position — da war kein GPS an, als ich sie geschrieben habe. Und aus „keine Position" hat die Karte gemacht: **null Grad Nord, null Grad Ost.** Das ist im Golf von Guinea, vor Westafrika. Viereinhalbtausend Seemeilen von Warnemünde.

Darunter stand „35 erfasste Positionen". Es waren acht.

Jetzt kommt der Teil, der mich beschäftigt hat. Die Prüfung, die genau das verhindert, gibt es seit dem **12. August**. Sie steht in einer anderen Datei, mit einem langen Kommentar davor, der das Problem beschreibt — bis hin zu dem Satz, dass die Karte sich dadurch von Schweden bis Libyen aufzieht.

Drei andere Stellen, die dieselben Daten lesen, wussten nichts davon.

Eine davon ist der Export. Wenn ich meine Saison als GPX-Datei ausgebe und in OpenCPN lade, wären achtundzwanzig Punkte vor Afrika drin gewesen. Auf einer Karte sieht man sowas. In einer Datei, die man weitergibt, nicht.

> **Eine Regel, die in einer Datei steht, schützt eine Datei.**

„1 Spuren"

Kleinigkeit. Hat mich trotzdem geärgert. Oben rechts auf der Karte stand **1 Spuren**. Wenn dann 1 Spur.

Ich bin da nicht pedantisch aus Spaß. Wenn eine App an so einer Stelle schludert, denkt der Nutzer: *hier hat sich keiner Mühe gegeben.* Und dann glaubt er dir auch bei den Dingen nicht mehr, bei denen du dir Mühe gegeben hast.

Nachgezählt kam heraus: An siebenundsiebzig Stellen steht in der App eine Zahl direkt vor einem Hauptwort. An zweiunddreißig war es richtig gemacht, an achtunddreißig nicht. „1 Häfen gespeichert" gab es neunmal. „1 Personen an Bord". „1 Wachen über 24 Stunden".

Zwei Zeilen unter dem falschen „1 Spuren" stand übrigens „1 eingelesene Spur" — richtig. Gleiche Karte, gleiche Zahl, eine Fassung richtig und eine falsch.

Das war an dem Tag zum wievielten Mal? Ich habe aufgehört zu zählen.

Wo ist denn das Löschen?

Dann sollte ich zwei von drei Spuren löschen, damit man sieht, ob die Einzahl wirklich funktioniert. Ich habe gesucht. Und gesucht.

Es gab keinen Löschknopf. Nirgends.

Oben rechts stand „3 Spuren". Darunter eine Zeile: *Spur aus einem anderen Programm ergänzen.* Das klingt nach Hinzufügen. Dass man diese Zeile antippen muss, damit sich eine Liste aufklappt, in der die drei Spuren mit Löschknopf stehen — darauf muss man erst mal kommen.

Ich habe die App gebaut. Ich habe es nicht gefunden.

> **Eine Zahl anzuzeigen, ohne zu sagen, wo die Sache liegt, ist eine halbe Auskunft.**

Abends, iPhone statt iPad

Das iPad war weg, also das iPhone. AirDrop fand es nicht. Brauchte es auch nicht — der Ordner liegt in iCloud, und der Dateiwähler in Safari kommt da direkt ran.

Auf dem iPhone hat es dann in einem Rutsch funktioniert. Erst eine Spur, dann zwei, dann eine gelöscht. Alle drei Zahlen sind sofort mitgegangen, ohne Neuladen. Genau das war der Punkt.

Was ich mitnehme

Zehn Sachen sind an dem Tag geschlossen worden. Vier davon habe ich gefunden, einfach dadurch, dass ich die App benutzt habe, während die KI sie gemessen hat.

Das ist keine Kritik. Das ist die Arbeitsteilung, und sie funktioniert. Ein Prüfprogramm findet, was man ihm zu suchen sagt. Ein Nutzer findet, was ihn stört. Und was einen stört, stört meistens auch den nächsten.

Was mich wirklich beschäftigt, ist etwas anderes: **Fünfmal an einem Tag war die Lösung schon im Programm** — an einer anderen Stelle, in einer anderen Datei, manchmal zwei Zeilen daneben.

Ich habe immer gedacht, Software wird schlecht, weil jemand schlampt. Ich glaube inzwischen, sie wird schlecht, weil dasselbe an fünf Stellen entschieden wird und nur an vieren richtig.

Seit dem Tag liegen drei Entscheidungen an genau einer Stelle: ob eine Position gültig ist, ob es Einzahl oder Mehrzahl heißt, und was in der Seitenleiste steht. Ohne Rückfalllösung. Eine Rückfalllösung ist nur eine Doppelung, die man noch nicht bemerkt hat.

Von Apple ist übrigens immer noch keine Mail da. Vierter Tag.

ORIGINALBEITRAG · TEIL 25

Meine Messungen sagten, es sei in Ordnung

28. August 2026, zweite Hälfte

Der Tag hatte zwei Hälften, und ich habe das erst hinterher gemerkt.

Vormittags war ich der Nutzer. Ich habe die App benutzt, während sie gemessen wurde, und dabei vier Sachen gefunden, die kein Prüfprogramm gefunden hätte — die Spur nach Afrika, das „1 Spuren", den fehlenden Löschknopf, die Vier in der Seitenleiste. Das steht im Teil davor.

Nachmittags war ich der Auftraggeber. Ich habe gesagt: **erst alle Fehler ausmerzen, damit endlich die Arbeit an der nativen App beginnen kann.** Und dann sind an einem Nachmittag zwanzig Lieferungen rausgegangen.

Aber das, was ich mir von diesem Tag merken werde, ist keins von beidem.

Fünf Antworten auf eine Frage

Auf der Startseite gibt es eine Kachel „Wartung". Die sagte, solange es sie gibt: **Keine kritische Frist.** Immer.

Der Grund war schlicht. Sie fragte an den Wartungseinträgen nach drei Feldern, die in der ganzen App nirgends geschrieben werden. Alle drei Bedingungen konnten nie zutreffen. Eine Kachel, die immer dasselbe sagt, ist keine Kachel, sondern ein Bild.

Interessant wurde es beim Nachsehen. Die Frage „wann ist eine Wartung überfällig?" stand im Programm in **fünf** Fassungen:

```
die Zeile pro Anlage        Datum ODER Betriebsstunden   — richtig
die Kennzahl darüber        nur Datum
der Filter „Nur fällige"    nur Datum
die Anlagenliste            nur Datum, ausgemusterte mitgezählt
die Kachel                  Felder, die niemand schreibt
```

Die erste ist die richtige, und die Datei sagt das selbst. Direkt darüber steht seit dem Bau der Satz: *„Herstellerangaben lauten typischerweise ‚alle 250 Betriebsstunden ODER jährlich, je nachdem was zuerst eintritt'. Genau dieses ODER wird hier abgebildet."*

Vier Stellen kannten diesen Satz nicht. **Zwei davon in derselben Datei, fünfzig Zeilen darunter.**

Nachgestellt sah es so aus: Die Kennzahl sagte „3 überfällig", und drei Zentimeter darunter stand in der Zeile der Hauptmaschine „50 Bh überfällig". Zwei Antworten in einer Ansicht, übereinander.

Ein Datum im falschen Jahr

Der zweite Fund war kleiner und hat mich mehr gestört.

Ein Ablaufdatum ist ein Kalendertag, keine Uhrzeit. Wenn man es dem Rechner ohne Uhrzeit gibt, macht der daraus Mitternacht Weltzeit — und westlich von Greenwich ist Mitternacht Weltzeit noch der Vortag. In New York nachgestellt:

```
Ablaufdatum 15.09.2026   stand da als   14.09.2026
Ablaufdatum 01.03.2027   stand da als   28.02.2027
Ablaufdatum 01.01.2026   stand da als   31.12.2025
```

Der letzte ist der, bei dem ich kurz die Luft angehalten habe. Ein Datum im **falschen Jahr**. In einer nach Saison geordneten Übersicht landet der Eintrag damit in der falschen Spalte, und niemand sucht ihn dort.

Und wieder dieselbe Gestalt, diesmal fast komisch deutlich: An zehn Stellen im Programm wird eine Uhrzeit angehängt, damit genau das nicht passiert. An sechs nicht — und fünf davon sind **wörtlich dieselbe Zeile**, kopiert in fünf verschiedene Dateien.

Fünf Kopien desselben Fehlers. Kein einziger Blick über die Dateigrenze.

Zwei Meter Wasser

Der dritte gehört zu denen, bei denen ich froh bin, dass wir ihn vor dem Sommer gefunden haben und nicht danach.

Auf dem Marine-Dashboard gibt es eine Kachel, die heißt **„Tiefe"**. Sie holt sich ihren Wert aus dem Bordnetz, indem sie nach dem Wort „depth" sucht. Das Bordnetz kennt aber drei Tiefen: unter dem Kiel, unter dem Geber, unter der Wasserlinie. Alle drei enthalten „depth".

Welche gewinnt, hängt daran, welche das Bordnetz zuerst aufzählt. Das sichert niemand zu.

```
Bordnetz meldet alle drei, Kiel zuletzt aufgezählt

vorher    Tiefe   9,1 m     (unter Wasserlinie)
nachher   Tiefe   7,3 m     „Tiefe · unter Kiel"
```

1,8 Meter zu optimistisch. Das ist der Unterschied, der ein Boot auf Grund setzt — und die Kachel hieß nur „Tiefe", ohne zu sagen, welche der drei Zahlen sie gerade zeigt.

Beim Wind dasselbe: „wind.speed" trifft auch den scheinbaren Wind. Die Kachel „Wind" zeigte 12,8 Knoten scheinbar statt 9,4 wahr, ohne es zu sagen.

Jetzt steht dabei, welcher Wert es ist. Das ist eigentlich die ganze Reparatur: nicht eine andere Zahl, sondern eine Zahl, die sich benennt.

Die halbe Stunde, die drei wurden

Punkt 8 steht seit dem 20. August im Heft: *eine halbe Stunde mit einer echten Vorlesehilfe.* Acht Tage lang wurde er weitergereicht. Zweimal hat die KI stattdessen selbst gemessen und daraus geschlossen, es sei in Ordnung.

Am Abend war das iPad wieder da, und ich habe gesagt: machen wir.

Ich hatte VoiceOver noch nie benutzt. Das erste, was ich gelernt habe, ist, dass sich mit VoiceOver **jede** Geste ändert — ein Tippen löst nichts mehr aus, es liest nur vor. Wer das nicht vorher weiß, kommt aus den Einstellungen nicht mehr raus. Also erst den Notausstieg einrichten: dreimal die obere Taste. Einmal an, einmal aus, dann konnte nichts mehr schiefgehen.

Dann der Reihe nach durch die App.

Was mich zuerst überrascht hat, war, wie viel stimmte. „Aktuelle Seite" bei Bordzentrale. „Banner", „Navigation, Orientierungspunkt". Alle Knöpfe hatten Namen, auch die, die nur ein × zeigen — die heißen im Betrieb „Schließen".

Und dann habe ich den Logbuchdialog geöffnet.

„Noch keine geprüfte Position verfügbar"

Und nochmal. Und nochmal. Und nochmal.

In Dauerschleife, ohne Pause, über alles andere drüber. Man kommt in dem Dialog nicht vorwärts, weil jede Meldung die vorherige unterbricht. Man erreicht die Eingabefelder nicht. Der wichtigste Dialog der App war mit einer Vorlesehilfe schlicht **nicht bedienbar**.

Was ich in dem Moment gedacht habe, war nicht „gut, dass wir das machen". Es war: *na super, noch was.*

Ich hatte eine halbe Stunde eingeplant für etwas, das eine Pflichtübung war. Jetzt saß da ein Fehler, der nach Arbeit aussah, und es war halb sechs.

Die Ursache war dann in fünf Minuten gefunden und hat mich mehr beschäftigt als der Fehler selbst.

Der Hinweis unter dem Positionsfeld ist als „meldender Bereich" ausgezeichnet — er soll sich melden, wenn sich etwas ändert. Geschrieben wird er aus einer Funktion, die **im Sekundentakt** läuft. Der Text war dabei jedes Mal derselbe. Aber ihn zu setzen tauscht den Textknoten aus, und das gilt als Änderung.

```
in zehn Sekunden   vorher   20 Schreibvorgänge   0 echte Änderungen
                   nachher   0 Schreibvorgänge   0 echte Änderungen
```

Zwanzig überflüssige Schreibvorgänge alle zehn Sekunden. Dauerhaft. Auf einem Boot ist Strom nicht beliebig.

Und sichtbar war davon nichts. Wer sieht, merkt von zwanzig identischen Schreibvorgängen gar nichts. Wer hört, kann die App nicht benutzen.

Der Satz, um den es geht

Die KI hatte die Bedienbarkeit **vorher gemessen**. 398 Knöpfe mit Namen, null ohne. Die aktuelle Seite richtig gekennzeichnet. Jeder Bereich ausgezeichnet. Kein Bild ohne Alternativtext. Fünfunddreißig Reiter, alle mit Auswahlzustand.

Alles davon stimmte. Ich habe die Zahlen selbst gesehen.

> **Gemessen war die Auszeichnung. Nicht gemessen war das Verhalten über Zeit.**

Das ist für mich der Satz des Tages, und er hat mit Blindheit nichts zu tun. Ein Prüfprogramm sieht einen Zustand. Es sieht nicht, was zwischen zwei Zuständen passiert. Und eine ganze Klasse von Fehlern lebt genau dort.

Danach kamen noch vier: Fünfundzwanzig Dialoge ohne Namen. Der Name, den niemand vorliest, weil der Fokus auf dem Schließen-Knopf landet. Der Hinweis, der zweimal kommt. Und das große rote SOS auf dem Notfallbildschirm, das VoiceOver **buchstabiert** — „Großes S, Samuel".

Das letzte habe ich nicht gehört, sondern gelesen: VoiceOver schreibt mit, was es sagt, unten am Bildschirmrand. Da stand es.

Fünf Tasten ohne Namen

Zum Schluss noch die Karte. Ich wische mich durch, und es kommt fünfmal hintereinander **„Taste"**. Ohne Namen. Doppeltippen tut nichts Erkennbares. Erst danach „Zoom in", „Zoom out", „Leaflet".

Leaflets eigene Knöpfe hatten Namen. Unsere Marken nicht — jeder Logbuchpunkt, jedes Foto, jede eingelesene Spur. Sie tragen alle einen Aufklapptext mit Datum, Ereignis und Route. Der steht nur erst da, wenn man draufgetippt hat.

Und bei der Fotomarke war es am schönsten: Da steht ausdrücklich, dass das Symbol nicht vorgelesen werden soll. Richtig. **Nur hat nichts es ersetzt.**

> **Halb gelöst ist bei Bedienbarkeit schlimmer als gar nicht gelöst. Das Zeichen war weg, und übrig blieb eine namenlose Taste.**

Jetzt sagt jede Marke, was sie ist: „Logbuchpunkt · 18.8.2026 10:00 · Etappe 3 · Warnemünde nach Kühlungsborn". Und die Zoomknöpfe heißen auf Deutsch.

Wozu das alles

Am Ende habe ich gefragt, wie viele blinde Segler es auf der Welt gibt und ob die Stunden sinnvoll waren.

Die Zahlen: weltweit 43 Millionen blinde Menschen, 295 Millionen stark sehbehinderte, 1,1 Milliarden mit irgendeiner Sehbeeinträchtigung. Bei der Blindsegel-Weltmeisterschaft 2019 traten vierzehn Mannschaften an; 2017 waren es fünfzig Segler aus fünf Nationen, davon die Hälfte sehbehindert. Der organisierte Blindsegelsport weltweit umfasst einige hundert Menschen.

Ich habe nicht gefragt, um die KI zu prüfen. **Ich hatte echte Zweifel, wofür der Aufwand überhaupt betrieben wird.** Drei Stunden für eine Gruppe, von der ich nicht wusste, ob sie in meinem Fall überhaupt existiert — und mitten in einem Tag, an dem ich eigentlich die Fehlerliste leerräumen wollte, damit endlich die native App drankommt.

Die ehrliche Antwort war: **Wenn die Frage lautet, wie viele blinde Menschen ihr Bordbuch in PELARON führen werden, ist die Antwort vermutlich: niemand.** Und rechtlich verpflichtet bin ich auch nicht — das Barrierefreiheitsgesetz nimmt Kleinstunternehmen aus, und eine Bordbuch-App fällt ohnehin nicht darunter.

Was bleibt, ist etwas anderes, und das überzeugt mich mehr als beides:

Die Dauerschleife war **kein Fehler für Blinde**. Sie war ein Fehler, den eine Vorlesehilfe nur hörbar gemacht hat. Zwanzig überflüssige Schreibvorgänge alle zehn Sekunden hätte niemand gefunden — kein Blick auf den Bildschirm, kein Prüfprogramm.

Und die Gruppe, die etwas davon hat, sind nicht die 43 Millionen, sondern eher die 1,1 Milliarden. An Bord ist sie noch größer: Sonne aufs Display, Gischt, Seegang, Nachtsicht, die Lesebrille unter Deck. Segler sind im Schnitt nicht jung. Nachlassendes Sehen ist auf einem Boot der Normalfall.

Die KI hat auch gesagt, wo es sich **nicht** gelohnt hat: Etwa die Hälfte der Zeit ging in Dialognamen und das buchstabierte SOS. Das hilft Vorlesehilfen-Nutzern und sonst niemandem. An erwartbaren Nutzern gemessen war diese Hälfte nicht zu rechtfertigen.

Damit kann ich leben. Ich hätte es nicht ausgehalten, wenn sie mir erzählt hätte, es sei alles gleich wichtig gewesen.

Was mich an dem Abend am meisten geärgert hat, war übrigens weder die App noch die Frage nach dem Sinn. **Es war das Verfahren selbst.**

Mit VoiceOver kann man nicht eben mal etwas ausprobieren. Jede Prüfung geht so: einmal wischen, aufschreiben, was gesagt wurde, weitergeben, warten, wieder wischen. Wörtlich mitschreiben, weil eine Zusammenfassung nichts wert ist. Und zwischendurch immer wieder Tab schließen, neu laden, von vorn anfangen, weil eine neue Fassung online ist.

Das ist kein Fehler von jemandem. Es liegt in der Sache: Ich bin das Messgerät, und ein Messgerät, das tippen muss, ist langsam. Aber nach der fünften Runde „wisch einmal nach rechts und sag mir, was kommt" war ich es leid.

Dass am Ende sechs Befunde standen, ändert daran nichts. Es hat sich gelohnt. Angenehm war es nicht.

Und dann noch aufräumen

Zum Schluss zwei Sachen, die keine Fehler waren, sondern Entscheidungen.

Die erste: Beim Bauen der Wartungskachel kam heraus, dass die Datei für die Bordzentrale zur Hälfte ins Leere schreibt. Von sechsunddreißig Anzeigen, die sie befüllt, gibt es siebzehn im Programm gar nicht — Törnkachel, Sicherheitscheckliste, Bereitschaftsbalken, Zeitleiste, Crew-Zahl, Ausgabensumme.

Das waren keine falschen Rechnungen. Die Rechnungen waren richtig. Eine davon, der Bereitschaftsbalken, war am selben Vormittag sogar noch verbessert worden.

Nur sah sie niemand. Seit wer weiß wie lange.

Ich habe entschieden: raus. Dreitausend Zeichen weniger, und die Datei sagt jetzt, was sie tut.

Die zweite war unangenehmer, weil sie eine eigene Regel betraf. In einer Datei stand noch eine zweite Prüffunktion für Sicherungen, „als Rückfall, falls die richtige nicht lädt". Genau diese Doppelung war am Vormittag die Ursache eines Fehlers gewesen — der Knopf griff an der Reparatur vorbei auf die alte Fassung und meldete jede gültige Sicherung als ungültig.

Repariert wurde der Aufruf. Stehen blieb die Doppelung. Und im Heft steht, vom selben Tag:

> **Eine Rückfallfassung ist nur eine Doppelung, die man noch nicht bemerkt hat.**

Also auch raus. Eine Stelle, kein Rückfall. Wenn die Prüfung fehlt, sagt der Knopf das jetzt — statt still anders zu antworten.

Was ich mitnehme

Der Tag endet mit einem leeren Register: 106 Punkte, 105 abgeschlossen. Offen ist einer, und der ist meine eigene Zurückstellung.

Aber die Zahl ist nicht das, was hängen bleibt.

Es ist der Umstand, dass eine Messung sagen kann, alles sei in Ordnung, und alles ist trotzdem nicht in Ordnung. Nicht weil die Messung falsch gerechnet hätte, sondern weil sie das Falsche angesehen hat.

Die KI hat sich das an einem Tag dreimal selbst nachgewiesen: Ein Prüfling hat nichts gemessen und trotzdem eine Zahl geliefert — und weil „nichts" auf beiden Seiten gleich aussieht, wäre es beinahe durchgegangen. Einmal hätte eine Nachbildung sogar einen Fehler behauptet, den es gar nicht gibt; da war mein iPad die Instanz, die widersprochen hat.

Und achtmal — achtmal an einem Tag — hat ein Zähler den eigenen Erklärkommentar mitgezählt. Das ist fast lustig. Die Lehre daraus ist nicht „vorsichtiger kommentieren", sondern: den Zähler über etwas legen, das ein Kommentar nie enthalten kann.

Ich glaube, das ist die eigentliche Arbeit an einer Software, die man alleine baut. Nicht die Fehler zu finden. Sondern herauszufinden, wo man systematisch nicht hinsieht — und dafür braucht man etwas, das anders hinsieht als man selbst.

Heute war das ein iPad, das vorliest.

ORIGINALBEITRAG · TEIL 26

Svenner ist mein iPhone

28. und 29. August 2026

Am Freitagabend um 19:38 lief zum ersten Mal Programmcode für die native App durch, und vierunddreißig Prüfungen meldeten null Fehler. Auf Anhieb.

Ich habe darauf nicht mit Erleichterung reagiert. Ich habe gedacht: **na endlich, jetzt beginnt der Bau der nativen App.**

Das ist die ehrliche Reaktion nach einem Tag, an dem es von morgens bis abends um Fehler ging. Die grünen Prüfungen waren nicht das Ereignis. Der Anfang war es.

Was die Maschine nicht kann, und was daraus folgt

Vor der ersten Zeile stand eine Messung, an die ich selbst nicht gedacht hätte. Der Rechner, auf dem die Maschine arbeitet, hat **kein Swift** — und kann sich auch keines holen. Zwei Adressen, beide abgewiesen, gemessen um 17:31.

Für die Webseiten war es dieses Jahr die tragende Regel: erst am Server messen, dann behaupten. Neun Funde in einer einzigen Woche verdanken sich dieser Schleife. Bei einer nativen App gibt es keinen Server. Es gibt Xcode auf meinem MacBook und ein iPad in meiner Hand.

Die Antwort darauf war ein Schnitt, den ich für den wichtigsten dieser Woche halte:

> **Alles, was eine Entscheidung trägt — die Geometrie des Schwojkreises, wann ein Alarm gilt, welche Position zählt — kommt in einen Teil, der ohne Gerät und ohne Simulator prüfbar ist. Die Oberfläche und alles, was das System anfasst, bleibt außen vor.**

Der eine Teil lässt sich in Sekunden messen. Der andere nur, indem ich draufdrücke. Wie richtig dieser Schnitt war, hat sich schneller gezeigt, als mir lieb ist.

Zwei Fehler, die der Übersetzer gefunden hat

Der erste Bauversuch brachte zehn Fehler, alle mit derselben Ursache: eine fehlende Zeile in zwei Dateien. Beim Umbau war sie herausgefallen. Kein Denkfehler, ein Abschreibfehler.

Der zweite war interessanter. Eine gelbe Warnung, die man leicht wegklickt: *„Result of 'try?' is unused."*

Dahinter steckte etwas Echtes. Die Zeile springt ans Ende der Protokolldatei, bevor eine neue Zeile angehängt wird. Mit dem Fragezeichen wird ein Scheitern **verschluckt** — und dann schreibt der nächste Befehl da weiter, wo der Griff gerade steht: am Dateianfang. Die neue Ortsmeldung hätte den Anfang der Nacht überschrieben.

> **Ausgerechnet in der Datei, die beweisen soll, dass die Wache durchgelaufen ist.**

Beide Fehler standen in dem Teil, für den es keine Prüfungen gibt. Der geprüfte Teil war größer und hatte keinen.

Und dann Samstagmorgen

Ich hatte Freitagabend abgebrochen, weil ich zu müde war. Das lag nicht an der Arbeit. Es lag daran, dass ich in der letzten Stunde einen Fehler gejagt habe, den es nicht gab.

Xcode meldete immer wieder: **„Developer Mode disabled"** — der Entwicklermodus sei auf dem Gerät nicht eingeschaltet. Ich habe ihn eingeschaltet. Am iPad, in den Einstellungen, ganz unten. Neustart. Bestätigt. Der Schalter stand auf **Ein**.

Die Meldung kam wieder.

Kabel ab, Kabel dran. Xcode beendet, Xcode gestartet. iPad neu gestartet. Entwicklermodus aus, Neustart, wieder ein, Neustart. Die Meldung kam wieder.

Am Samstagmorgen habe ich dann das Fenster geöffnet, in dem Xcode auflistet, welche Geräte es sieht. Da standen **zwei**.

| | | |---|---| | **Svenner** | iPhone 17 Pro — mit dem roten Fehler | | **iPad Pro** | 12,9 Zoll — **ohne jeden Fehler** |

„Svenner" ist mein **iPhone**.

Die ganze Zeit war eingestellt, dass die App auf das iPhone soll. Dort war der Entwicklermodus aus. Am iPad war alles in Ordnung, von Anfang an. Es war nur nie gemeint.

Und die Maschine hatte mir am Abend geschrieben: *„Svenner ist dein iPad."* Angenommen, weil der Name auftauchte, als ich das iPad anschloss. Nachgesehen hat sie nie — obwohl in genau dem Fenster, das die Antwort enthielt, beide Geräte untereinander standen.

> **Das ist derselbe Fehler, gegen den dieses Heft seit vier Wochen anschreibt, nur diesmal von der anderen Seite: eine Behauptung ohne Messung. Sie hat mich einen Feierabend und einen Vormittag gekostet.**

Die Katze und ihr Schwanz

Als die App dann endlich auf dem iPad lag, stand oben brav das Recht auf Ortung — und darunter *„Noch keine Position."* Der Knopf „Anker gefallen" war grau.

Der Grund war wieder logisch, nicht technisch: Das Einschalten der Ortung steckte im Ankersetzen. Also keine Position, weil nicht geortet wird; nicht geortet, weil kein Anker gesetzt ist; kein Anker möglich, weil keine Position da ist.

An Bord ist die Reihenfolge umgekehrt und jeder weiß das: **Erst weiß man, wo man ist, dann fällt der Anker.**

Eine Zeile, und die Position stand da. Auf fünf Meter genau.

Die Zahl, die sich von selbst geändert hat

Und dann kam der Moment, an dem ich gesehen habe, dass das Ding wirklich rechnet und nicht nur anzeigt.

Vor der Ortung stand der Alarmradius bei **50 Metern**. Nachdem die Position da war, standen dort **55**.

Der Unterschied sind die fünf Meter, auf die das iPad seine eigene Position genau kennt. Die gehen in den Kreis ein, weil die Unschärfe zweimal wirkt: beim Setzen des Ankerpunkts und beim Vergleich. Niemand hat das eingetippt. Die Rechnung hat sich genommen, was sie brauchte, sobald es da war.

29 Meter waagerecht, 11 Meter Boot, 5 Meter Ortung, 10 Meter Reserve.

Was ich mitnehme

Ich habe von Xcode keinerlei Ahnung. Ich bin da absoluter Laie, und ich habe ein Werkzeug bedient nach einer Anleitung von jemandem, der es **selbst noch nie gesehen hat** — das stand ehrlich oben drüber.

Es war die einzige Möglichkeit, außer mit anderen Werkzeugen von vorn anzufangen. Am Ende hat es geklappt, und das freut mich.

Aber der Preis stand in derselben Anleitung, in einem Satz, den ich schon im August aufgeschrieben hatte:

> **Was auf dem Bildschirm steht, gilt. Nicht, was die allgemeine Seite sagt, nicht, was in der eigenen Anleitung steht.**

Am Freitagabend stand auf dem Bildschirm eine Liste mit zwei Geräten. Wir haben beide nicht hingesehen — sie nicht, weil sie es für erledigt hielt, ich nicht, weil ich es ihr geglaubt habe.

Am Donnerstag fahre ich aufs Boot. Dann wird sich zeigen, ob die App mit dem Bordnetz redet — und das kann sie im Browser nicht, aus einem Grund, den wir im August gemessen haben. Deshalb der ganze Umbau.

Fünf Tage.

Nachtrag vom selben Vormittag, 08:21. Die Frage, wegen der das alles gebaut wurde, ist beantwortet. Ich habe den Anker gesetzt, das iPad gesperrt und achtzehn Minuten liegen lassen.

1245 Meldungen. 97 Prozent davon aus dem Hintergrund. Größte Lücke: null Minuten.

Die Ortung läuft also weiter, während das Gerät zu ist. Das konnte die Webfassung nie, und deshalb der ganze Umbau.

Die interessanteste Zahl ist aber eine andere: **vier Meter Drift.** Das iPad lag still auf dem Schreibtisch, und trotzdem ist die gemeldete Position um bis zu vier Meter gewandert. Das ist keine Bewegung, das ist die Ortung selbst.

Damit weiß ich seit heute etwas, das ich vorher nur vermutet hatte: **Ein Ankeralarm mit zehn Metern Radius geht an einem Boot los, das gar nicht geankert hat, sondern nur daliegt.** Die zehn Meter Reserve in der Rechnung sind keine Vorsicht. Sie sind das Mindeste.

ORIGINALBEITRAG · TEIL 27

Zum Anschauen, nicht zum Bedienen

29. und 30. August 2026

Wenn ich diese zwei Tage auf einen Satz bringen müsste, wäre es meine eigene Antwort auf eine Frage, die die Maschine mir am Sonntagmittag gestellt hat. Sie wollte wissen, wann mir der Verdacht gekommen ist, dass die native App kleiner wird als pelaron.app. Ich habe geschrieben:

> **heute, beim logbuch und bei der törnplanung, sah alles so aus, als ob man da nicht bedienen, sondern nur anschauen kann**

Das ist der Faden dieser zwei Tage. Es ging beide Male um dasselbe, einmal am Samstagnachmittag und einmal am Sonntagvormittag, und ich habe es beide Male selbst finden müssen.

Samstagmittag: ein Vergleich, der uns beim Namen nennt

Angefangen hat der Samstag harmlos. Ich hatte eine achtseitige Gegenüberstellung zwischen PELARON und Rubberduck geschickt und um Verbesserungsvorschläge gebeten.

Der Bericht war ehrlich gearbeitet — er trennt sauber, was geplant ist, von dem, was es gibt. Nur macht er bei PELARON denselben Fehler, den wir uns monatelang selbst durchgehen ließen: Häkchen ohne Deckung. „MQTT/Modbus vorbereitet". „Digitaler Zwilling: ja."

Und dann stand da der harte Befund, den ich mir merken werde: **Eine Webseite kann keine rohe TCP-Verbindung öffnen.** Kein Yacht Devices, kein Actisense. Übrig bleibt Signal K über WebSocket. Damit bestätigt der Bericht der Konkurrenz das stärkste Argument für die native App — und ist zugleich die klarste Begründung dafür, die wir je hatten.

Dazu kam das Symbol. Farben und Geometrie sind aus dem vorhandenen **icon512.png gemessen** worden, nicht neu erfunden: Navy #082943, Orange #F27600, das Vorsegel #DBE7EE. Beim ersten Versuch waren die Bögen ausgefranst — als dicke Linie gezeichnet statt als Fläche — und die Wortmarke saß 38 Punkt zu hoch. Beides hat die Maschine selbst gefunden, bevor ich es gesehen habe. Das kommt vor und ist die angenehmere Sorte Fehler.

Und ich habe gefragt, was ich jetzt wohin schieben muss. Antwort: nichts. Das Projekt ist mit Xcode 26 angelegt und zieht neue Dateien im Ordner von allein ins Projekt. Das ganze „Add Files"-Geklicke entfällt. Seither legt die Maschine ab, und ich baue.

Vier Übersetzungsfehler in einer halben Stunde

Zwischen halb zwei und zwei gab es vier Fehler hintereinander, alle von der Maschine, und der vierte ist der interessante.

import Combine vergessen — zum zweiten Mal in zwei Tagen. Dann ein Wettlauf in der Gerätesuche: eine Variable wurde vom einen Faden geschrieben und vom anderen gelesen. Das hätte ein gefundenes Gateway als „keine Verbindung" melden können. Am Steg wäre das der teuerste Fehler von allen, weil man dann das halbe Boot auseinandernimmt und der Fehler in der App sitzt.

Die Berichtigung war selbst falsch. Zweimal.

Und beim vierten Anlauf hat sie in den Bauplan geschaut. Dort stand ein Wort: **SWIFT_DEFAULT_ACTOR_ISOLATION = MainActor**. Danach war es erledigt.

Drei Versuche im Nebel, und im Bauplan stand die Antwort. Das ist dasselbe Muster wie mit den Prüfsummen im Sommer: erst raten, dann messen. Es kostet jedes Mal denselben Umweg.

Aus meiner Konsole kam noch ein Fund: **No symbol named 'anchor' found in system symbol set**. Sie hatte das Zeichen erfunden. Ein erfundener Symbolname fällt nicht auf — er zeichnet einfach nichts. Jetzt heißt es **circle.dashed**, der Schwojkreis, was ohnehin besser passt.

Die Generalprobe, und was danach kam

Um halb drei lief eine Generalprobe, die mir gefallen hat. Ein erfundenes Gateway am Mac gibt sich als Yacht Devices aus, damit die ganze Kette zu Hause messbar ist. Ergebnis: **aus 264 Rahmen wurden 66 vollständige Nachrichten**, kein Stück verloren. Zwei Tanks getrennt gezählt. Unter „Was dieses Boot nicht sendet" standen genau die zwei PGN, die ich ausgelassen hatte. Der Mitschnitt: 1.664 Zeilen, null unlesbar.

Und dann habe ich hingeschrieben, was mir die ganze Zeit im Kopf herumging:

> **wie soll ich am freitag, wenn ich einen törn fahre, die native app benutzen um diese ausführlichen tests zu unterziehen, so jedenfalls nicht**

Alles bis dahin war ein **Messgerät**. Adresse eintippen, Mitschnitt starten, Zahlen ablesen, Datei sichern. Das geht am Steg mit einem Kaffee daneben. Es geht nicht, wenn ich am Ruder stehe.

Die Maschine hatte für **Donnerstag** gebaut — den Tag, an dem ich am Boot stehe und messe — und dabei übersehen, dass **Freitag** das Gegenteil davon ist. Daraus wurde der stille Zeuge: nicht der Mensch prüft die App, sondern die App prüft sich selbst und sagt am Abend, was war. Aus drei Vorschlägen habe ich genau den genommen, und der Name stimmt.

Dazu kam die ehrliche Rechnung, um die ich nicht herumkomme: Die Webfassung hat 4.775 Bedienelemente. Die native App hatte an dem Nachmittag zwei Bildschirme. Bis Freitag wird daraus nichts. Also fährt der Törn auf pelaron.app, und die native App läuft daneben mit und zeichnet auf. Und das MacBook kommt mit, damit Fehler Freitagabend im Hafen behoben werden und die Rückfahrt am Samstag ein zweiter Realtest wird.

Ein Ding aus diesem Gespräch halte ich für das nützlichste des Tages: den **Befundknopf**. Eine Berührung, Zeit und Position sind gemerkt, die Worte kommen abends. Ein Fehler, der um halb drei auffällt und um neun als „irgendwas mit dem Logbuch" erinnert wird, ist kein Befund.

Der Abend: acht Fehler, und alle acht waren meine Prüfungen

250 Prüffälle, acht Fehler — und keiner davon im Kern. Alle acht saßen in den Prüfungen selbst.

Sechs auf einen Schlag, weil die Testdaten **genau eine Minute** auseinander lagen und genau eine Minute die Schwelle für eine Lücke ist. Zwei weitere, weil ein Zeitpunkt um elf Tage danebengeschrieben war — aus dem Kopf überschlagen statt gerechnet.

Danach 250, null Fehler. In Xcode blieb einer übrig: ein Datumsformat ohne **public**. Den konnte der Prüflauf nicht finden, weil die Prüfungen **im** Kern laufen und nicht davor.

Das Muster des Tages, viermal in verschiedenen Kleidern: **hingeschrieben statt gerechnet.** Ein Byte um eine Stelle verschoben. Eine gemerkte Uhrzeit als Uhrzeit genommen. Instanzen falsch nummeriert. Elf Tage. Jedes Mal hat die Prüfung es gefunden — und dort, wo nicht geprüft werden kann, habe ich es gefunden.

Sonntagmorgen: durchgelaufen

Um zehn nach acht standen zwei Antworten da, auf die ich seit Donnerstag gewartet hatte.

Die Ankerwache hat **8,8 Stunden** durchgehalten, 2.218 Meldungen, alle aus dem Hintergrund, größte Lücke null Minuten. Vom Symbol gestartet, ohne Fehlersucher am Kabel. Das ist genau das, was die Webfassung nie konnte, und der ganze Grund für den Umbau.

Der stille Zeuge lief daneben: 2.126 Meldungen, keine Unterbrechung, und das Logbuch hat sich **stündlich von allein geschrieben** — 23:15, 0:15, 1:15, 2:15, 3:15, 4:15.

Und dann hat derselbe Lauf eine Antwort widerlegt, die ich vor zwei Tagen noch für gesichert hielt.

| Lauf | Dauer | Drift | |---|---|---| | 1a | 0,3 h | 4 m | | 1b | 2,8 h | 16 m | | 2 | 8,8 h | **6 m** |

Der längste Lauf hat die zweitkleinste Drift. Die Kurve, die aus den ersten beiden Punkten gebaut worden war, sagte 28,4 Meter voraus. Gemessen sechs. Um den Faktor 4,7 daneben.

Dass zwei Punkte keine Kurve tragen, stand sogar in der Datei. Es hat trotzdem gereicht, um daraus eine Wurzelfunktion und einen Alarmradius von 84 Metern zu bauen. **Ein Vorbehalt im Text hindert niemanden daran, die Zahl zu benutzen** — mich eingeschlossen.

Nebenbei ist auch die Stromsorge geplatzt: 4,2 Meldungen je Minute, nicht 59. Faktor 14 zu groß geschätzt.

Sonntagvormittag: zum Anschauen

Und dann kam der Vormittag, der diesem Teil den Namen gibt.

Ich habe mir das Logbuch angesehen und gefragt, wie man daran ein Logbuch führen soll. Man kann es nämlich nicht. Es war eine Liste. Man konnte hineinsehen und nichts hineinschreiben. Dasselbe bei der Törnplanung: Man konnte einen Törn ansehen, den es nicht gab.

Das ist es, was ich meinte. **Es sah aus, als ob man da nicht bedienen, sondern nur anschauen kann.**

Das Logbuch wurde daraufhin an einem Stück nachgebaut — dieselben Felder wie in pelaron.app, dieselben Wörter, „Groß 1. Reff" und „Vorsegel stark gerefft", zwölf Standardereignisse, und beim Ändern ein Pflichtgrund. Ein Logbuch, das sich spurlos ändern lässt, ist als Nachweis wertlos.

Dann die Törnplanung, ebenfalls in einem Zug: Kopfdaten, Etappen, Abfahrtscheck mit den sechs Punkten im Wortlaut, Zielhafen mit Koordinaten, und die Historie **aus den Logbucheinträgen**. Der Satz, der in meiner Webfassung steht und den ich hier wiedergefunden habe, gefällt mir: *Wo keine Einträge vorliegen, steht ein Strich — nicht die geplante Distanz, die etwas anderes bedeutet.*

Zeile 53

Ein Fehler des Vormittags gehört hierher, weil er so klein und so lehrreich ist.

Der Bau brach ab. Gemeldet wurde Zeile 55, der eigentliche Fehler stand in Zeile 53:

```
for punkt in Abfahrtspunkt.allCases { p.haken(punkt, auf: true) }
```

haken ist kein Befehl, sondern eine Menge. Die Maschine hatte den Befehl kurz vorher umbenannt — und dabei eine Ersetzung benutzt, die nur die Aufrufe traf, deren erstes Glied mit einem Punkt beginnt. Vier von fünf Stellen.

Eine Umbenennung, die vier von fünf trifft, ist keine.

Daraus ist eine neue Prüfregel geworden, und die Begründung dafür finde ich besser als die Regel selbst: Der erste Versuch war allgemein gehalten und meldete prompt zwei Fehlalarme — zwei Aufrufe, die völlig in Ordnung waren. Eine Prüfung mit Fehlalarmen wird nach zwei Tagen abgeschaltet und fehlt dann genau an dem Tag, an dem sie zuträfe. Jetzt steht schlicht eine Liste da: Was umbenannt wurde, darf nirgends mehr stehen.

Die Frage, die ich stellen musste

Dann habe ich nach den Häfen gefragt. Warum man beim Starthafen keine Marina wählen kann und beim Zielhafen auch keine angezeigt wird. Ob die Häfen so gespeichert werden wie in pelaron.app. Ob die Marinas in der Karte auswählbar sind.

Vier Fragen, vier Treffer. Der Starthafen war nur ein Textfeld. Im Törnformular hing die falsche Suche — eine **Ortssuche**, die „Warnemünde" kennt, aber keine „Marina Hohe Düne". Es gab keinen Hafenbestand, während meine Webfassung seit V53 einen hat, revierweise geladen, mit dem ausdrücklichen Grundsatz: *erst der eigene Bestand, dann das Netz* — weil ein Werkzeug an Bord ohne Empfang sonst nichts zeigt. Und Marinas auf der Karte gab es gar nicht.

Da ist mir der Kragen geplatzt, und ich habe geschrieben, was ich schon dreimal gesagt hatte:

> **mann wenn ich sage vollen funktionsumfang wie pelaron.app, dann meine ich es auch so, sage das nicht aus spaß**

Und dann die Frage, auf die es mir wirklich ankam: **Wolltest du etwas anderes bauen als pelaron.app?**

Die Antwort war: zum Teil ja. Nicht als Plan, aber im Ergebnis. An jeder Weggabelung stand „nachlesen, wie pelaron.app das macht" gegen „selbst entscheiden", und fast jedes Mal wurde selbst entschieden, weil das schneller ging. Und fast jedes Mal fiel die Entscheidung kleiner aus als meine Fassung.

Danach wurde die Webfassung endlich vollständig ausgelesen. Das Ergebnis in drei Zahlen: **20 Ansichten, 11 Dialoge, rund 130 Speicherschlüssel.** Die native Navigation kannte zwölf Bereiche. Acht Ansichten kamen darin überhaupt nicht vor — nicht als Baustelle, sondern gar nicht: SOS und Notfall, Medizin, Sicherheit, Wartung, BSH-Nachrichten, Analysen, Assistent, System.

Der Satz, der daraus in den Quelltext gewandert ist, trifft es:

> **Wer einen Bereich weglässt, statt ihn als offen zu zeigen, entscheidet still, dass es ihn nicht gibt. Ein Baustellenschild ist ehrlich; eine Leerstelle ist es nicht.**

Seither stehen alle zwanzig drin, in meinen Gruppen und mit meinen Namen. Neun gebaut, zwölf mit Schild.

Handschuhe

Der letzte Punkt kam mir beim Ansehen. Die neue Leiste sah gut aus — und genau das war das Problem.

> **nein, sah auch so schön aus, aber fragt mich dann, wie ich das auf einem boot was schwankt und wenn es kalt ist, noch mit handschuhen bedienen soll**

Einundzwanzig Einträge standen plötzlich in zwei Spalten, ohne Gruppenüberschriften, jeder klein. Man sieht alles und findet nichts. Ich habe gefragt, warum es keine aufklappbaren Menüs sind — meine Webfassung macht das seit jeher so, zwei Gruppen offen, drei zu.

Und aus dem Handschuh wurde eine Regel, die mir gefällt, weil sie eine Zahl begründet statt sie zu setzen: Apples 44 Punkt sind an einem trockenen Zeigefinger auf einem ruhigen Tisch gemessen. Beides trifft an Deck nicht zu. Ein Segelhandschuh legt einen Zentimeter zwischen Fingerkuppe und Glas, und das Schiff steht nicht still. Seither gilt für die Navigation eine **Griffläche von 56 Punkt**, rund neun Millimeter.

Damit begründen sich die aufklappbaren Gruppen von selbst: Größere Ziele und alle einundzwanzig Bereiche gleichzeitig geht nicht. Weniger Ziele heißt größere Ziele. Was ich unter Segeln brauche, steht offen; was ich am Steg brauche, ist einen Griff entfernt.

Warum ich am Ende gelobt habe

Um kurz nach zwölf habe ich ein Lob verteilt, und das kam nach zwei ziemlich deutlichen Ansagen. Die Maschine wollte wissen, was sich dazwischen geändert hat. Meine Antwort:

> **das aussehen und das die KI es endlich kapiert hat, dass ich eine funktionsfähige und auch bedienbare native APP haben will**

Beides zusammen. Nicht das eine oder das andere. Eine App, die gut aussieht und die man nur anschauen kann, ist mir auf einem schwankenden Boot bei Kälte nichts wert. Eine, die alles kann und aussieht wie ein Messgerät, auch nicht.

Am Ende stand meine eigene Rechnung: 93 Prozent des Nutzungslimits verbraucht, Rücksetzung in einer Stunde. Ich habe gefragt, ob das noch reicht, um etwas zu bauen. Die Antwort war nein, und sie war richtig begründet — sieben Prozent reichen für eine Lieferung, aber nicht für den Umlauf dahinter. Wenn das Limit mittendrin greift, liegen Dateien auf meinem Mac, die nicht übersetzen, und ich komme eine Stunde lang nicht weiter.

Also wurde geschrieben statt gebaut. Auch das ist eine Antwort, die ich hören wollte.

ORIGINALBEITRAG · TEIL 28

Gehts noch?

30. August 2026, Nachmittag und Abend

Es gibt einen Satz aus diesem Abend, den ich hier voranstelle, weil alles andere danach kam:

> **da muss ich dir widersprechen, die 3 punkte brauchst es, und den wichtigsten sicherheit und notfall willst du weglassen, gehts noch????**

Vier Fragezeichen. Ich schreibe selten vier Fragezeichen.

Die Maschine hatte mir eine ordentlich gemachte Tabelle vorgelegt, Spalte für Spalte, was meine Webfassung auf der Bordzentrale zeigt und was die native App davon hat. Drei Spalten waren sauber. In der vierten stand bei vier Karten „nein, noch nicht": Bordkasse, Inventar, Aufgaben — und **Sicherheit & Notfall.**

Am Donnerstag stehe ich am Boot. Am Freitag fahre ich.

Warum das kein kleiner Fehler war

Die Begründung klang sogar vernünftig. Ein SOS-Knopf, der auf eine leere Seite führt, sei schlimmer als keiner. Eine Karte mit drei Nullen sei keine Karte. Das stimmt für die Bordkasse. Für den Notruf stimmt es nicht.

Und der eigentliche Fehler steckte eine Ebene tiefer. In der Navigation stand „SOS & Notfall" auf **noch nicht gebaut**, und daraus wurde geschlossen, es sei nicht baubar. Nachgesehen hatte niemand.

Als dann nachgesehen wurde, lag alles längst da. In meiner Webfassung steht seit V52 eine Datei mit **elf Seenotleitstellen**, jede mit ihrem Zuständigkeitsgebiet als Rechteck, den Rufnummern, UKW 16 und DSC 70 an erster Stelle. Und im Kern der nativen App lagen seit Tagen Bootsname, MMSI, Rufzeichen, Personen an Bord, Position, Kurs und Fahrt. Es fehlte **nichts**.

Die Antwort darauf gefiel mir:

> **Ich habe „SOS steht auf Nochnichtgebaut" geschrieben und daraus geschlossen, es ginge nicht — das war die falsche Reihenfolge: erst nachsehen, dann urteilen.**

Ich habe dann gesagt, was ich meinte: **alles.** Auch Inventar, auch Aufgaben, auch Bordkasse. Wenn ich alles sage, meine ich auch alles; das hatten wir an diesem Tag schon zweimal.

Was in vier Stunden entstanden ist

Es wurde dann tatsächlich alles gebaut, und zwar in dieser Reihenfolge, weil der Kalender sie vorgibt: erst der Notfall, dann der Rest.

Der Notfallbildschirm hat eine Reihenfolge, die ich für die eigentliche Entscheidung halte. Ganz oben stehen die vier Angaben, die in den Funkspruch gehören — Boot, MMSI, Rufzeichen, Personen. Weil man sie abliest, **während jemand schon spricht.** Dann UKW 16 und DSC 70, die überall gelten und nicht fehlschlagen können. Dann 112, mit dem Satz, dass ein Handy kein Schiff in der Nähe erreicht. Dann erst die zuständige Stelle.

Die Position steht in **Grad und Dezimalminuten**. Nicht in Dezimalgrad. Wer „54,18" durchgibt, zwingt die Gegenstelle zum Umrechnen, und im Funk wird nicht gerechnet.

Und im kopierten MAYDAY-Text bleibt eine Zeile absichtlich leer:

```
Art des Notfalls: [ERGÄNZEN]
```

Die App weiß nicht, was passiert ist, und darf es nicht raten. Ein vorbelegtes Feld wird im Ernstfall unverändert abgelesen.

Danach kamen Wartung, Sicherheit, Inventar, Bordkasse und Crew — sieben Bildschirme, 1.716 Zeilen. Die Eingabefelder habe ich gegen meine Webfassung zählen lassen, Feld für Feld: neun beim Artikel, sechs bei der Ausgabe. Was ich dort seit Monaten eintippe, tippe ich jetzt auch nativ.

Vier Dinge sind bewusst anders als bei mir, und alle vier finde ich besser:

– **Knapp und abgelaufen sind zwei Sachen.** Eine Dose Diesel läuft nicht ab, eine Signalrakete schon — und nachkaufen hilft dort nicht.

– **Die Wiederkehr rechnet vom Erledigungstag**, nicht von der alten Frist. Wer den jährlichen Ölwechsel im März statt im Januar macht, hat danach bis März Ruhe. Sonst ist die Aufgabe beim Abhaken schon wieder überfällig.

– **Die Bordkasse rechnet den Ausgleich.** Nicht „Malte: ?45,00 €", sondern „Malte zahlt 45,00 € an Sven". Das eine ist eine Deutungsaufgabe, das andere eine Handlung. Und so wenige Überweisungen wie möglich.

– **Was Handlung braucht, steht oben.** Meine Webfassung zeigt erst die ganze Liste und die Warnungen darunter. „87 Artikel" verlangt nichts.

Der Dunkelmodus, den keiner bestellt hatte

Um kurz vor neun ist mein iPad von selbst in den Dunkelmodus gewechselt. Nicht in unseren Nachtmodus — in den des Systems. Und die halbe App war weg.

> **damit sind einige buttons nicht erkennbar bzw. das was darin steht**

Überschriften unsichtbar. „Ausgaben und Ausgleich", „Bordbestand und Nachschub", „Seenotfall". Die Knöpfe „Einkauf" und „MRCC Bremen anrufen" ebenso. Die Zahlen in der Bordkasse. Ein dunkelblaues Nichts.

Die Ursache ist die Sorte Fehler, über die ich mich am meisten ärgere, weil sie so dumm ist: Es gab **zwei Eigenschaften für dieselbe Sache.** Eine, die mit dem Systemmodus mitwandert, und eine feste. Benutzt wurde an neunzehn Stellen die feste. Am Tag sieht man den Unterschied nicht.

Dann wurde gemessen statt geraten, und die Zahlen sind hübsch:

| Farbe | auf dunklem Grund | auf hellem | |---|---|---| | Navy #041F33 | **1,03 : 1** | 16,81 : 1 | | Orange #F27600 | 5,74 : 1 | **2,85 : 1** | | Rot #C2261F | **2,80 : 1** | 5,85 : 1 |

1,03 zu 1. Das ist kein schwacher Kontrast, das ist gar keiner.

Und die 2,85 kannte ich schon. Genau diesen Wert habe ich am 22. August in meiner Webfassung beanstandet, und genau diesen Fehler hatte die native App an zehn weiteren Stellen geerbt — orange Schrift, die am Tag zu blass ist. Beides ist jetzt beseitigt. Der einzige feste Wert, der bleibt, ist Navy auf Orange beim Hauptknopf: dieselbe Paarung in beiden Modi, 5,23 zu 1. Helle Schrift auf Orange käme auf 2,9 und wäre schlechter.

Der Knopf, der nicht stumpf wurde

Punkt 10 meiner Prüfliste: Mann über Bord auslösen, danach muss der Knopf stumpf werden. Er wurde es nicht.

Im Quelltext stand **.disabled(…)**, sauber und richtig. Nur dimmt SwiftUI eben nur die **mitgelieferten** Knopfarten. Ein selbstgebauter Knopf zeichnet stur weiter, solange er nicht selbst nachfragt. Alle drei unserer Knopfarten taten das nicht.

Ein Knopf, der aussieht wie ein Knopf und nichts tut, ist an Bord schlimmer als gar keiner. Man drückt ihn dreimal und sucht den Fehler woanders.

Dreimal derselbe Name

Punkt 32: Steht der Bereichsname irgendwo dreimal? Ja. Bei Crew stand „Crew" in der Seitenleiste, oben mittig, und noch einmal darunter.

Der erste Versuch, das zu beheben, war ein **leerer** Eintrag an der Mittelstelle der Leiste. Klingt vernünftig, funktioniert nicht: SwiftUI überspringt ein leeres Element und zeichnet den Titel doch. Es half nur, gar keinen zu setzen.

Danach habe ich gefragt, was ohnehin auf der Hand lag:

> **warum verschieben wir den bereich nicht nach oben, platz wäre doch jetzt da?**

Die Leiste trug kein Wort mehr und kostete trotzdem rund fünfzig Punkt. Beim ersten Versuch, sie zu verstecken, war der Inhalt unter die Statusleiste gerutscht — „Törn starten" stand quer über der Uhrzeit. Der Grund dafür war aber nicht die Leiste, sondern **ihr Sicherheitsabstand**, der mit ihr verschwand.

Jetzt bringt die Arbeitsfläche einen eigenen mit: acht Punkt statt fünfzig. Und zwar als Sicherheitsabstand, nicht als Innenabstand — der Unterschied zählt beim Scrollen. Ein Innenabstand gehört zum Inhalt und läuft mit durch; die Überschrift stünde nach zwei Fingerbreit doch unter der Uhrzeit.

Und dann stand es doch wieder da

Um zwanzig nach neun habe ich die neue Bordzentrale angesehen. Sie sieht gut aus. Oben rechts, neben dem Wort BORDZENTRALE, stand:

unbekannt.

Genau das Wort, das ich Stunden vorher beanstandet hatte. Sechs Stellen waren berichtigt worden — und die siebte, die über allen anderen steht, war übersehen.

Der Hergang ist lehrreich. Beim ersten Versuch war ein eigenes Wort an die Überschriftentafel geschrieben worden. Der Prüfer meldete: *diese Ansicht hat kein Feld dafür.* Und daraufhin wurde das Wort **weggenommen**, statt das Feld **einzubauen**. Der Prüfer hatte recht und wurde falsch verstanden.

Ich habe dann gesagt: beseitige den Fehler, und schau am besten alles durch, damit du nicht wieder was übersiehst.

Das Ergebnis dieser Durchsicht: **24 Überschriftentafeln mit Lampe, 36 Tafelköpfe. Kein einziger trug ein eigenes Wort.** Alle 24 haben jetzt eins, dazu die 19 Tafelköpfe, bei denen „unbekannt" herauskommen konnte.

Der Satz, der dazu in den Quelltext gewandert ist, trifft es:

> **„unbekannt" ist ein Zustand des Prüfers, kein Zustand des Bootes. Wer am Steg steht, will „Am Boot" lesen.**

Die Prüfsumme, die nicht stimmte

Ein Zwischenfall des Abends gehört hierher, weil er mich zwanzig Minuten gekostet hat und am Ende keiner war.

Ich habe einen Prüfbefehl laufen lassen, und die Gesamtprüfsumme lautete **84a180b0d29c66b1**. Die Maschine hatte kurz vorher **510c7ae58a3b14f6** gemessen. Bei denselben Dateien.

Es lag nicht an den Dateien. Der Befehl hängte die Einzelprüfsummen in der Reihenfolge von **sort** aneinander — und **sort** ordnet je nach Spracheinstellung anders. Mein Terminal sortiert anders als das der Maschine. Gleiche Bytes, andere Reihenfolge, andere Summe.

Dieser Fehler steckte **seit dem ersten Tag** in jedem Prüfbefehl. Aufgefallen ist er erst, als zum ersten Mal ein anderes Terminal rechnete. Seitdem wird über die Prüfsummen selbst sortiert — die bestehen nur aus Ziffern und Buchstaben, und die ordnet jede Sprache gleich.

Daraus ist die 39. Regel geworden: **Ein Prüfbefehl darf nicht von der Spracheinstellung abhängen.**

Dreizehn neue Regeln an einem Abend

Der Prüfer, den wir uns gebaut haben, ist an diesem Abend von 20 auf 33 Regeln gewachsen. Jede einzelne entstand aus einem Fehler, der tatsächlich passiert ist — keine ausgedachten.

Ein paar davon gefallen mir besonders:

– Eine Liste, die **String** und **String?** mischt. Der Bau brach ab, und **dieselbe Zeile stand ein zweites Mal** in einer anderen Datei. Ein Bau meldet nur den ersten Fund; ohne die Regel hätte ich sie noch einmal über meinen Mac gefunden.

– Ein SwiftUI-Bauteil mit einem Argumentnamen, den es dort nicht gibt. Das kam daher, dass ein Ersetzungslauf **Klammern gezählt hatte statt Aufrufe zu lesen** — in vier von sechs Fällen landete er an der falschen.

– Einen Typ zweimal bauen. Das ist an diesem einen Tag **viermal** passiert.

– Eine feste Farbe als Schrift — der Dunkelmodus.

– Eine eigene Knopfart, die nicht dimmt — der Mann-über-Bord-Knopf.

Und zwei der neuen Regeln haben sich beim ersten Anlauf **selbst blamiert.** Die eine meldete eine Funktion als nicht vorhanden, die es sehr wohl gibt — sie steht nur in einer Erweiterung, in der alles öffentlich ist, ohne dass es dasteht. Die andere war so breit, dass sie 36 Stellen meldete, von denen 17 völlig in Ordnung waren.

Beides wurde berichtigt, und die Begründung dafür finde ich wichtiger als die Regeln: **Eine Prüfung, die Richtiges meldet, ist schlimmer als keine. Sie erzieht zum Wegsehen.**

Wo wir jetzt stehen

Am Nachmittag waren neun von einundzwanzig Bereichen gebaut. Am Ende des Abends sind es **fünfzehn**. Offen bleiben Medizin, BSH-Nachrichten, Dokumente, Analysen, Assistent und System.

Die Prüfliste ist auf 54 Punkte gewachsen und liegt als PDF bei mir — ich hatte darum gebeten, weil ich sie am Stück abarbeiten wollte, während mein Sohn das iPad hatte. Das hat gut funktioniert und wird jetzt bei jeder Lieferung fortgeschrieben.

Was noch fehlt, weiß ich genau. Der Tidenhub braucht eine Quelle — ich habe mich für die Zwölferregel entschieden, plus die amtliche Vorhersage, wenn Empfang da ist. Dazu kam eine Anmerkung, die ich mir merke: In der Ostsee liegt der astronomische Tidenhub bei zehn bis zwanzig Zentimetern. Für den Schwojradius ist das Rauschen. Was dort zählt, ist der **Windstau** — und den liefert eine Gezeitentafel nicht.

Am Donnerstag stehe ich am Boot.

ORIGINALBEITRAG · TEIL 29

Der Deckel war null Punkt hoch

31. August 2026, ganzer Tag

Drei Fehler an diesem Tag haben eines gemeinsam: Sie sind **fehlerfrei übersetzt worden.** Xcode hat nichts gesagt. Es gab keine Warnung, keine gelbe Zeile, kein Ausrufezeichen. Der Code war richtig geschrieben — er tat nur nichts.

Ich schreibe diesen Teil deshalb von hinten nach vorn.

Der Törn rutscht, und das ist alles

Am Morgen stand über der Aufgabenliste noch „noch drei Tage". Am Vormittag kam die Starkwindwarnung für Freitag und Samstag. Ich habe den Törn um eine Woche verschoben.

Ich wurde gefragt, was das mit mir macht. Die ehrliche Antwort: nichts. Eine Woche später ist eine Woche später. Wenn überhaupt, habe ich gedacht: na wenigstens etwas mehr Zeit zum Bauen der nativen App.

Das schreibe ich hier auf, weil in der ersten Fassung dieses Absatzes eine Enttäuschung stand, die es nicht gab. Sie war gut formuliert und komplett erfunden. Seit Teil 26 gilt bei mir die Regel, dass vor jedem Tagebuchteil gefragt wird, statt geraten. Das hier ist der Grund dafür.

Die Zahl, die nicht geschätzt war

Der Tag fing mit den Gezeiten an. Für den Schwojradius braucht man den Wasserstand, und ich hatte mich für die **Zwölferregel** entschieden — der alte Seemannsdreisatz: In sechs Stunden Tide fallen 1, 2, 3, 3, 2, 1 Zwölftel des Hubs.

Im Kern stand als Kommentar, die Regel weiche „rund vier Hundertstel des Hubs" von der Sinuskurve ab. Das war geschätzt. Also wurde nachgerechnet, über die ganze Kurve, in feinen Schritten. Es sind **2,555 Prozent**. Der Kommentar ist berichtigt, und ein Prüffall hält die Zahl jetzt fest.

Danach wollte ich es an echten Werten sehen, und dabei sind zwei Dinge schiefgegangen, die ich beide unterhaltsam finde:

Der erste Versuch, die Pegeldaten für Warnemünde zu holen, lief auf einen Fehler 404. Die Adresse hieß **WARNEMUENDE**, die Station heißt **WARNEMÜNDE**. Umlaut. Nicht mein Umlaut — der Behörde ihrer.

Der zweite Versuch fand die Hoch- und Niedrigwasser über einen Vergleich mit den Nachbarwerten. Bei einer Reihe mit Zentimeterauflösung stehen aber immer wieder zwei gleiche Werte nebeneinander, und die galten damit als beides gleichzeitig. Ich bekam **tausende** Scheitel für einen Tag.

Am Ende: 1436 Messwerte, Wasserstand zwischen 497 und 512 Zentimetern. Fünfzehn Zentimeter Unterschied über einen ganzen Tag. Das ist die Ostsee, und es ist genau der Grund, warum ich mir bei diesem Bereich am wenigsten vormache: Was hier zählt, ist der Windstau, nicht der Mond.

Die Taste, die nichts tat

Nachmittags habe ich geschrieben:

> **taste als tidenhub übernehmen ist ohne funktion**

Und weil ich weiß, dass die Frage kommt: **der rest hat funktioniert.**

Der Gezeitenrechner steckte in einem Aufklapper, und dieser Aufklapper hatte eine zweizeilige Überschrift. Das ist der Unterschied. Ein Aufklapper legt seine Aufklapp-Geste über seine Überschrift; ist die nur ein Wort, bleibt sie dort. Ist sie ein Block mit eigener Höhe, greift sie weiter — und dann verliert eine Taste darunter den Tastendruck gegen sie. Textfelder und Datumswähler nicht, die bringen eine höher gestellte Bedienung mit.

Genau das war zu sehen: alles ging, nur die Taste nicht.

Ersetzt wurde der Aufklapper durch einen eigenen, der keine fremde Geste mitbringt — ein Knopf und ein **if**, mehr nicht. Und daraus wurde Regel 34 im Prüfer.

Ich habe damals gedacht: gut, das ist erledigt.

Die Bootsakte, und warum ich scannen wollte

Am Nachmittag kam der Bereich Dokumente. Meine Webfassung kann dort Dateien anhängen, Fristen führen, Kategorien. Was sie nicht kann: die Kamera als Scanner benutzen. Ein Browser darf das nicht.

Ich habe geschrieben, das soll die native App können. Und dazu: ansehen, an jeder Zeile, ohne Umweg.

Ich wurde gefragt, ob ich dabei an ein bestimmtes Papier gedacht habe. Habe ich nicht. Ich habe einfach das Gehirn angeschaltet — so, wie es einem Selbstständigen mit gewissen Papieren in der täglichen Arbeit eben passiert. Man kennt das Gefühl, etwas zu suchen, von dem man genau weiß, dass man es hat.

Gebaut wurde es mit dem Bauteil, das auch in Notizen und Dateien den Scanner stellt: Kantenerkennung, Entzerrung, mehrere Seiten hintereinander. Die Seiten werden zu **einem** PDF gerechnet, jede in ihrer eigenen Größe — ein Bootsschein hat zwei Seiten, eine Werftrechnung fünf, und fünf Einzeldateien wären kein Beleg.

Eine Sache daran gefällt mir besonders, weil sie ohne mich entstanden ist: Nichts wird abgelegt, bevor „Sichern" gedrückt ist. Wer eine Police durch eine falsche ersetzt und dann abbricht, hat die alte noch. Der erste Entwurf hatte sofort kopiert. Die Begründung für die Änderung stand im Code, bevor ich sie lesen konnte:

> **Ein Bogen, der „abgebrochen" sagt und die Platte trotzdem verändert hat, ist eine Lüge.**

Zwei Fragen, zwei Felder

Abends kam Medizin — Bordapotheke, Vorfälle, Notfallangaben der Crew.

Beim Auslesen meiner Webfassung fiel etwas auf, das mir in Jahren nicht aufgefallen ist: Ich habe dort **zwei verschiedene Wortlisten für dieselbe Frage.** Im Logbogen steht „stabil, verschlechtert sich, bessert sich, nicht ansprechbar". Im Vorfallbogen steht „stabil, weiter beobachten, ärztliche Hilfe nötig, erledigt".

Beim Nachsehen ist das nicht eine Liste zu viel, sondern **zwei Fragen, die durcheinandergeraten sind.** Wie geht es der Person — und was folgt daraus. Nativ sind daraus zwei Felder geworden. „Bessert sich, aber ärztliche Hilfe nötig" kann ich in meiner Webfassung nicht aufschreiben. Sie muss sich entscheiden.

Der Funkspruch hat zwei Stufen, und **PAN PAN ist vorbelegt**. Der Grund steht seit Wochen in meiner eigenen Webfassung, und ich zitiere ihn hier, weil er richtig ist: MAYDAY bindet Rettungsmittel, die anderswo fehlen können. Die Höherstufung muss eine bewusste Handlung sein und darf nicht aus einem Zustandsfeld abgeleitet werden.

Und was ins Logbuch geht, ist weniger als das, was im Bereich steht: „Ein medizinischer Vorfall lag vor." Kein Name, keine Diagnose. Ein Logbuch wird vorgezeigt; eine Krankenakte nicht. Das steht jetzt nicht nur als Satz da, sondern im Code — der Eintrag bekommt die Felder für Person und Ereignis ausdrücklich **nicht**, obwohl es sie gibt.

Drei Fehler, die fehlerfrei übersetzen

Und jetzt zu dem, was diesen Tag ausmacht.

Der erste. Seit ich die Navigationsleiste habe verstecken lassen — sie kostete fünfzig Punkt für ein leeres Feld —, läuft beim Scrollen der Inhalt oben ins Freie. Am Vormittag stand „Törns und Ereignisse" quer über der Uhrzeit. Es wurde ein Deckel gebaut, ich habe ihn geprüft, er stand in der Prüfliste, ich habe abgehakt.

Am Nachmittag habe ich in der Bootsakte gescrollt, und da stand „Digitale Bootsakte" wieder halb im Bild und der orange Knopf über der Batterieanzeige. Ich habe das Foto geschickt und gefragt, ob etwas auffällt.

Der Deckel sah so aus:

```
Gestalt.papier
    .frame(height: 0)
    .ignoresSafeArea(edges: .top)
```

Er war **null Punkt hoch**. Der Befehl, der ihn in den Systembereich hätte wachsen lassen, vergrößert keine Ansicht — er erlaubt einer Ansicht nur, dort hineinzuzeichnen. Eine Fläche ohne Höhe hat nichts zu zeichnen.

Der zweite. Abends habe ich den Nachtmodus eingeschaltet und geschrieben:

> **die STATUSANZEIGEN nicht mehr sehe im nachtmodus, also „offen" „abgeschlossen" etc.**

Der rote Filter lag als Multiplikation über der App. Eine Multiplikation behält nur den Rotkanal. Nachgerechnet:

Lampe	Farbe	nach dem Filter
kritisch	#C0211F	(192, 7, 4) — kräftiges Rot
Handlungsbedarf	#F27600	(242, 26, 0) — Rot
geprüft	#198754	**(25, 30, 10) — fast schwarz**
unbekannt	#6A7B87	**(106, 27, 16) — kaum etwas**

Dazu die Verdunklung. Grün landete bei (16, 20, 7). Alles, was seine Helligkeit aus Grün oder Blau bezieht, war nachts weg — ausgerechnet bei den Wörtern, die sagen, ob etwas offen ist.

Die Berichtigung ist eine Zeile: erst entsättigen, dann einfärben. Dann bleibt die Helligkeit stehen und nur der Farbton geht. Der Nachtmodus ist jetzt einfarbig rot statt „rötlich mit Farbresten", und das ist kein Verlust, sondern der Zweck. Nachts unterscheidet man nach dem Wort, nicht nach der Farbe.

Der dritte. Eine halbe Stunde später:

> **bei längeren drücken des mondes passiert .... nix, es verändert sich nix**

Der Mondknopf schaltet den Nachtmodus mit einem Tipp; gedrückt halten sollte die Einstellungen öffnen. Es war ein **Button** mit einem angehängten langen Druck. Ein Knopf bringt seine eigene Gestenerkennung mit — was von außen danebengelegt wird, verliert gegen sie.

Das ist **derselbe Fehler wie am Vormittag**, nur umgekehrt. Beim Aufklapper lag die fremde Geste unter dem Bedienelement, hier liegt sie darauf. Beide Male gewinnt die eingebaute Erkennung. Beide Male passiert einfach nichts.

Der schlimmere Teil war ein anderer: Dass man den Mond gedrückt halten *kann*, stand nur **im** Blatt. Also dort, wo man erst hinkommt, wenn man es schon weiß. Jetzt steht der Weg sichtbar unter System & Diagnose.

Eine Prüfung, die nicht scheitern kann

Der Deckel ist mir am meisten nachgegangen, und zwar nicht wegen des Codes.

Punkt 1 meiner Prüfliste vom Vormittag lautete: „Überschrift und Knöpfe verschwinden hinter einem Streifen in Papierton." Es gibt einen Streifen — acht Punkt Sicherheitsabstand, die es schon vorher gab. Der Inhalt läuft hinter ihm durch und darüber weiter ins Freie. Ich habe hingesehen, den Streifen gesehen und abgehakt.

Der Prüfpunkt konnte bestehen, ohne dass die Sache funktioniert.

In Teil 28 steht der Satz, eine Prüfung, die Richtiges meldet, sei schlimmer als keine, weil sie zum Wegsehen erzieht. Jetzt habe ich den Gegenfall: Eine Prüfung, die nichts melden kann, erzieht zum Abhaken. Beides sind keine Prüfungen.

Am Abend ist das noch einmal passiert, und diesmal auf der anderen Seite. Für den Mondknopf wurde eine neue Prüferregel gebaut, mit einem eingebauten Probefehler getestet — und sie hat geschwiegen. Sie lief bis zur ersten schließenden Klammer, und das ist bei einem Knopf mit zwei Bauteilen erst die Hälfte. Nachgebessert, noch einmal geprobt, dann meldete sie.

Dritte Regel an einem Tag, deren erste Fassung nachgeschärft werden musste. Und die einzige, die dabei **gar nichts** meldete statt zu viel. Ich weiß jetzt, welche der beiden Sorten mir mehr Sorgen macht.

Drei Geräte, ein Stand

Zwischendurch habe ich aufgeschrieben, was ich mir für später wünsche: Logbuch unterwegs auf dem iPad, Nachbereitung zu Hause am MacBook. Ich arbeite zu Hause mit dem Rechner — Törn planen, etwas suchen, andere Dinge. Eine haptische Tastatur ist mir dabei immer lieber als das Glas.

Und dann kam die Nachfrage, ob das eine Kopie sein soll oder derselbe Stand. Derselbe Stand, natürlich. Es nützt nichts, wenn ich daheim etwas einfüge und an Bord nicht mehr finde. Das beste Beispiel liegt gerade auf dem Tisch: **An Bord sind die Versicherungsunterlagen — und die Werft braucht davon eine Kopie fürs Winterlager.**

Damit ist eine Entscheidung von heute vormittag hinfällig. Ich hatte zugestimmt, dass die Dokument*angaben* in die Sicherung gehen und die *Dateien* nicht, wegen der Größe. Für eine Sicherung stimmt das. Für den Abgleich nicht — eine Bootsakte, die auf dem Rechner nur sagt „Haftpflicht 2026, gültig bis 21.04.2027", ohne dass ich die Police herausgeben kann, ist ein Verzeichnis und keine Akte.

Dazu kommt das iPhone. Das ist nicht abends zu Hause, sondern mit mir an Bord, während das iPad am Kartentisch liegt. Drei Geräte gehen nicht mehr paarweise. Sie brauchen eine gemeinsame Stelle — und daneben einen Weg, auf dem iPhone und iPad sich auch ohne Netz einig werden, im Boots-WLAN.

Gebaut ist davon nichts. Vorbereitet ist zweierlei: Die medizinischen Daten tragen seit heute ein Zeichen, das sagt „geht nie über das Netz", damit es beim Bau des Abgleichs nicht vergessen werden kann. Und die neuen Datensätze tragen einen Änderungszeitpunkt — ohne den kann kein Abgleich sagen, welche von zwei Fassungen die jüngere ist, und einer, der rät, ist schlimmer als keiner.

Von acht Dateien im Kern haben ihn jetzt drei. Fünf fehlen. Das ist der nächste Schritt, und er ist billig, solange die Dateien klein sind.

Wo wir jetzt stehen

Mein iPad war den halben Tag beim Sohn. Gebaut habe ich trotzdem — über WLAN aufs Gerät, aus dem Nebenzimmer. Aufgefallen ist es an einem Ordnernamen: **Debug-iphoneos** statt **Debug-iphonesimulator**. Die Maschine hatte vermutet, es sei der Simulator, und musste die Vermutung zurücknehmen.

Am Ende des Tages:

	gestern Abend	heute
Bereiche gebaut	15 von 21	**18 von 21**
Kern	67 Dateien, 470 Prüffälle	**78 Dateien, 624 Prüffälle**
App-Schicht	47 Dateien, 33 Regeln	**62 Dateien, 42 Regeln**

Dreizehn Lieferungen an die native App, fünf an meine Webfassung, eine an die Website. **Neun neue Prüferregeln an einem Tag**, jede einzelne aus einem eigenen Fehler entstanden, dazu eine im Kern.

Offen sind noch drei Bereiche: BSH-Nachrichten für Seefahrer, Analysen, der Assistent.

Und der Satz, den ich mir aus diesem Tag mitnehme, ist keiner über Segeln:

Ein Fehler, der eine Fehlermeldung erzeugt, ist der harmlose. Der andere übersetzt sich sauber, startet ordentlich und tut einfach nichts — und du findest ihn erst, wenn du ihn brauchst.

Am Donnerstag stehe ich vielleicht am Boot. Der Törn ist jetzt eine Woche später.

ORIGINALBEITRAG · TEIL 30

Das Feld stand die ganze Zeit da

1. September 2026, ganzer Tag

Gestern waren es drei Fehler, die fehlerfrei übersetzt haben. Heute ist es eine Stufe subtiler: **Code, der funktioniert und nichts sagt.**

Ein Feld, das da ist und keine Beschriftung trägt. Ein Knopf, hinter dem neunzehn Reviere liegen und der aussieht wie „neu laden". Eine Zahl in einer Tafel, die etwas anderes verspricht, als sie zeigt. Xcode sagt bei keinem davon etwas — es ist ja alles richtig geschrieben.

Warum ich mir die anderen angesehen habe

Ich habe mir heute Mittag einen Wettbewerbsvergleich geben lassen. Der Grund war nicht Neugier auf fremde Funktionslisten.

Der Grund war, dass in meiner eigenen App **einzelne Bereiche nicht miteinander verknüpft waren.**

Die App soll dem Eigner Arbeit abnehmen und nicht noch mehr Arbeit machen. Wer Wasser bunkert, muss die Kosten eintragen — und der Zustand des Tankinhalts muss sich ändern. Beides, aus einem Vorgang. Eine intelligente, smarte App soll entstehen, keine Sammlung von Formularen, die nebeneinander liegen und sich gegenseitig ignorieren.

Aus dem Vergleich ist mir ein Satz hängengeblieben, und ich habe ihn weitergegeben:

> **Die Differenzierung sollte konsequent von „mehr Funktionen" zu „bessere, geprüfte und miteinander verstandene Daten" verschoben werden.**

Dazu habe ich geschrieben: **diese Logik fehlt bei dir manchmal.**

Der Wasserfall, der die Bordkasse nie erreichte

Mein Beispiel war absichtlich das einfachste, das mir eingefallen ist. Wasser bunkern, dafür bezahlen. Zwei Dinge müssen passieren, und beide gehören zu einem Vorgang.

Beim Nachsehen kam heraus: **Genau das konnte die App schon — aber nur für Kraftstoff.** Im Code stand wörtlich **tankliste(.kraftstoff)**.

Frischwasser und Abwasser waren seit Monaten angelegt. Sie hatten Füllstände, Warnschwellen, Geber am Bordnetz. Und keinen einzigen Weg, fortgeschrieben zu werden — außer die Zahl unter „Mein Boot" von Hand zu überschreiben. Die ganze Maschinerie lag daneben und wurde nicht benutzt.

Aus „Tanken" ist „Bunkern" geworden, mit zwei Schiebern: was, und in welche Richtung. Abwasser stellt sich von allein auf Abpumpen — die Absauganlage im Hafen kostet ja genauso Geld wie der Zapfhahn. Achtzig Liter Wasser für zwölf Euro ergeben jetzt drei Dinge auf einmal: Füllstand hoch, Ausgabe in der Bordkasse, Eintrag im Logbuch.

Und wenn mehr eingetragen wird, als hineinpasst, sagt das Blatt, **wie viele Liter es nicht einträgt.** Stillschweigend zu deckeln wäre dasselbe wie stillschweigend zu übernehmen: Die Zahl stimmt dann zwar, aber der Tippfehler dahinter bleibt.

Zwei Zahlen, die nichts voneinander wussten

Derselbe Gedanke, zweimal weitergedreht.

Der Luftdruck war eine Zahl. 1008 Hektopascal. Was einen Segler angeht, ist aber die zweite Zahl: 1008 nach einer ruhigen Nacht sind etwas anderes als 1008 nach 1016 am Morgen. Die Reihe dafür steht längst im Logbuch — jeder Eintrag mit Luftdruck ist eine Messung. Gelesen hat sie bis heute niemand. Jetzt steht unter dem Feld, was der Druck in den letzten sechs Stunden getan hat, mit der Zahl der Messungen und der Herkunft dabei.

Was dort **nicht** steht, ist eine Bewertung. Kein „Sturm kommt". Fünf Hektopascal in drei Stunden sind eine Messung; daraus eine Warnung zu machen wäre eine Vorhersage, und dafür bräuchte es Schwellen, die niemand nachgelesen hat. Dieselbe Zurückhaltung wie bei der Douglas-Skala.

Der Stundenzähler war zweimal da. Das Bordnetz meldet die Betriebsstunden der Maschine, und unter „Mein Boot" steht eine Zahl von Hand. Aus der rechnet die Wartungsfrist. Standen die beiden auseinander, merkte es niemand — die App hatte beide Zahlen und hat sie nie nebeneinandergelegt.

Jetzt tut sie es, mit einer Regel, die ich mir aufschreiben lassen habe:

> **Ein Zähler, der von 3000 auf 12 springt, ist keine Berichtigung, sondern eine Frage.**

Übernommen wird deshalb nur auf Knopfdruck. Neuer Motor, getauschtes Instrument, falscher Geber — wer das stillschweigend beantwortet, verschiebt eine Wartungsfrist um Jahre, und das fällt erst auf, wenn der Motor steht.

Das Jahr, das im April anfängt

Am Nachmittag kam das, was ich vorher als „so eine Art Haushaltsbuch" beschrieben hatte. Werft, Winterlager, Liegeplatz Wasser, Versicherung, Segelmacher, Rigger, allgemeine Instandhaltung, Neuanschaffung — dieses Frühjahr zum Beispiel ein neuer Autopilot. Es gehen halt auch mal Dinge kaputt.

Dazu habe ich gesagt: Liegeplatz Wasser immer von April bis Oktober, Winterlager November bis März.

Genau daran hängt die Entscheidung, die ich am besten finde: Das Bootsjahr läuft **vom 1. April bis zum 31. März.** Ein Kalenderjahr zerschneidet meine Saison in der Mitte. Das Winterlager 2026/27 stünde halb im einen und halb im anderen Jahr, und keine der beiden Zahlen wäre eine Saison.

Und dann kam die Frage, was ich mit der Zahl am 31. März eigentlich anfangen will. Die Antwort ist kürzer, als die Frage vermuten lässt:

Speichern. Das ist eine digitale Bootsakte — für einen Verkauf zum Beispiel.

Ein Boot, dessen Kosten und Arbeiten über Jahre dokumentiert sind, ist etwas anderes als eines, bei dem der Verkäufer erzählt, er habe sich gut gekümmert.

Vier Fragezeichen

Und dann wollte ich ein Winterlager eintragen.

> **jetzt verrate mir wo ich jahresausgaben wie winterlager eintragen soll????**

Vier Fragezeichen. Die Antwort war: Bordkasse, Ausgabe erfassen, runterscrollen bis „Für wen", das orange Wort antippen.

Das Feld stand die ganze Zeit vor mir. Es hatte nur keinen Namen.

Der Grund ist eine Zeile, die im Quelltext richtig aussieht:

```
Picker("Posten im Bootsjahr", selection: $wahl)
    .pickerStyle(.menu)
```

Da steht die Beschriftung doch. Ja — aber außerhalb eines **Form** zeigt ein Klappmenü **nur den Wert.** Der Titel bleibt unsichtbar. Auf meinem Bogen standen dadurch drei orange Wörter untereinander — „Hafen", „Sven", „nicht zugeordnet" — und nichts sagte, was sie bedeuten.

Nachgesehen wurde daraufhin die ganze App. **Sechzehn Stellen.** Kategorie, Bezahlt von, Warengruppe, Stauort, Rolle, Bauart, Wiederkehr, Abstand — überall dasselbe.

Das ist kein Schönheitsfehler. Es ist genau das Gegenteil von dem, was ich heute Mittag beschrieben habe: Eine App, die dem Eigner Arbeit abnehmen soll, und ein Feld, das er nicht findet, obwohl es da ist, macht ihm Arbeit.

Der Posten steht jetzt oben bei Betrag und Kategorie, wo man sagt, was es war — und nicht mehr zwischen den Häkchen der Crew. Eine Werftrechnung hat mit „für wen" nichts zu tun.

„Ohne Zuordnung" war zweimal falsch

In der Jahresübersicht stand eine Zeile „Ohne Zuordnung · 170,00 €". Ich habe sie als **Törnzuordnung** gelesen, und dazu geschrieben: Eine Törnzuordnung ergibt bei der Jahresübersicht keinen Sinn und interessiert keinen.

Beim Nachsehen war sie zweimal falsch. Erstens das Wort: „Zuordnung" heißt in dieser App an drei anderen Stellen genau das, was ich gelesen habe. Sie heißt jetzt „Ohne Posten".

Zweitens die Zahl. Von den 170 Euro waren 160 Euro Diesel und Essen der Crew — Geld, das geteilt und über den Ausgleich zurückgezahlt wird und mich nichts kostet. In einer Tafel mit der Überschrift „Was das Boot im Jahr kostet" hatte es nichts verloren. Aufgeteilt wird jetzt nur, was als Bootsausgabe eingetragen ist.

Was ich am Saisonende nicht will

Mir ist noch etwas aufgefallen, das erst später weh tut: Die Ausgabenliste wird immer länger. Richtung Saisonende erleichtert das die Bedienung nicht gerade.

Ich habe gefragt, ob man das in der Kachel scrollbar machen könnte. Die Antwort war ein Nein mit Begründung, und die Begründung leuchtet mir ein: Eine Scrollfläche **innerhalb** einer scrollenden Seite sind zwei Scrollbereiche übereinander. Wo der Finger aufsetzt, entscheidet, welcher sich bewegt — und mit Handschuhen erwischt man regelmäßig den falschen.

Stattdessen: fünfzehn Zeilen in der Kachel, und „Alle zeigen" öffnet ein eigenes Blatt mit Suche, Saison und Postenfilter. Der Filter „ohne Posten" ist dabei die eigentliche Arbeit am Saisonende — er zeigt genau die Ausgaben, denen noch ein Posten fehlt.

Dasselbe Problem sehe ich beim Logbuch, und dazu hatte ich einen eigenen Vorschlag: Einträge, die einem Törn zugeordnet sind und deren Törn **abgeschlossen** ist, müssen auf der Seite nicht mehr sichtbar sein. Weg sind sie ja nicht — im jeweiligen Törn stehen sie auch nach Jahren noch.

Das ist so gebaut worden, und mir wurde gesagt, es sei die bessere Lösung als der Deckel nach Anzahl, den es vorher gab: Der schneidet nach Menge ab und weiß nicht, was er abschneidet. Meiner schneidet nach Bedeutung. Ein abgeschlossener Törn ist erzählt.

Darunter steht, wie viele Einträge ruhen, und ein Knopf holt sie zurück. Eine Liste, die stillschweigend etwas weglässt, ist der Anfang von „wo ist mein Eintrag hin".

Der Knopf, der aussah wie ein Kreispfeil

Noch so ein Fall von fertig gebaut und nichts gesagt.

Ich habe gefragt, wo ich weitere Reviere laden kann, wenn ich das Revier verlasse, das ich einmal am Anfang eingegeben habe.

Es gab den Knopf. Er hieß „Revier laden" und trug einen **Kreispfeil** — das Zeichen für „neu laden". Wer 56 Häfen aus der Mecklenburger Bucht im Bestand hat und nach Bornholm will, liest das als „meine Häfen auffrischen" und sucht woanders weiter.

Dahinter lagen die ganze Zeit **neunzehn Reviere**, von der Nordsee bis Estland. Der Knopf heißt jetzt „Weiteres Revier laden — 18 zur Auswahl" und trägt ein Plus.

Die Zeile, an der das ganze Logbuch hing

Am Abend kam noch etwas, das mir vorher nicht klar war, und ich schreibe es auf, weil es mich beim Lesen kurz kalt erwischt hat.

Eine Zeile im Kern sah so aus:

```
anlass = try k.decode(Loganlass.self, forKey: .anlass)
```

Sechs Fälle kennt dieses Feld: Törnbeginn, Zeit, Strecke, Kursänderung, Törnende, Von Hand. Steht dort ein siebtes Wort — weil eine neuere Fassung der App eines erfunden hat und die Datei auf ein Gerät mit der älteren wandert, beim Abgleich also ständig —, dann scheitert der Leser des Eintrags. Und weil das Logbuch als **eine Liste** gelesen wird, ist dann nicht ein Eintrag weg, sondern **alle.** Beim nächsten Start. Ohne Meldung.

Das Tückische daran: Die naheliegende Absicherung sieht aus, als wäre sie schon da. **decodeIfPresent(...) ?? ersatz** liest sich nachsichtig. Ist es aber nicht — **decodeIfPresent** gibt nur dann nichts zurück, wenn der Schlüssel **fehlt.** Steht ein unbekanntes Wort da, wirft es, und das **?? ersatz** dahinter kommt nie zum Zug.

Dreiunddreißig solche Stellen gab es im Kern. Alle umgestellt. Und der Prüfer hat eine neue Regel bekommen, die die Namen der Aufzählungstypen **selbst aus den Quellen holt**, damit niemand eine Liste pflegen muss.

Die Regel hat in derselben Minute, in der sie entstand, zwei Stellen gefunden, die beim Durchgehen von Hand übersehen worden waren. Genau dafür sind Prüfer da — nicht um zu bestätigen, was man ohnehin weiß.

Die Bootsakte, und warum sie mir wichtig ist

Zum Schluss ein Punkt, der seit Tagen offen war.

Wir werben mit einer digitalen Bootsakte und machen es nicht. Das geht gar nicht.

Ich will nicht noch einen Ordner zu Hause haben. Das Beispiel ist immer dasselbe: die Versicherungskopie für die Werft, wegen des Winterlagers. An Bord liegt die Police, zu Hause sitze ich am Rechner, und die Werft will sie jetzt.

An jeder Dokumentzeile steht jetzt ein Teilen-Zeichen. Betreff ist der Titel, die Nachricht der Kennsatz. Mail, AirDrop, Dateien, Drucken — was das Gerät eben anbietet. Bei einem Dokument ohne Datei ist das Zeichen nicht da; ein Knopf, der ins Leere greift, ist schlimmer als keiner.

Und am Abend, als ich darüber nachgedacht habe, ist mir aufgefallen, dass ich noch weiter gehen würde. In die App gehören auch die **Bedienungsanleitungen der verbauten Geräte** — das Raymarine Axiom, der Wasserboiler, was sonst noch an Bord ist. Die zieht man ohnehin alle als PDF aus dem Netz.

Denn wenn ich ehrlich nachzähle, habe ich fürs Boot allein **drei Ordner** im Regal:

1. einen mit **Anleitungen** 2. einen mit **Rechnungen fürs Refit** 3. einen dünnen über die **laufenden Kosten**

Und wenn ich mir ansehe, was heute gebaut wurde, dann sind das genau diese drei Ordner. Die laufenden Kosten sind seit heute Nachmittag die Saisonübersicht. Die Refit-Rechnungen sind Bootsausgaben mit einem Posten, und sie werden später die Servicehistorie sein.

Und die Anleitungen haben ihr Fach längst: Unter Dokumente gibt es die Kategorie **Handbuch**, und darunter steht als Hinweis „Motor, Elektrik, Navigation". Die Kategorie **Rechnung** heißt dort „Belege und Werftrechnungen". Zwei meiner drei Ordner haben also schon ihr Fach in der App.

Was fehlt, ist nicht der Ort. Es ist die **Verbindung.** Eine Bedienungsanleitung ist kein Dokument mit einer Frist — sie gehört zu einem Gerät. Das Axiom hat eine Anleitung, eine Seriennummer, ein Kaufdatum, eine Garantie und irgendwann eine Rechnung über die Reparatur. Solange die App keine Liste der verbauten Anlagen führt, liegen diese fünf Dinge in fünf Listen nebeneinander statt beieinander.

Und damit bin ich wieder bei dem Satz von heute Mittag, nur eine Etage höher: Es geht nicht um mehr Funktionen. Es geht darum, dass die Daten voneinander wissen.

Und auf dem SOS-Bildschirm steht endlich der **gesprochene Wortlaut** und nicht nur der Merkzettel mit den Angaben. Dreimal die Anrufung, dreimal der eigene Name, Position in Grad und Dezimalminuten. Was passiert ist, steht in Klammern und wird angesagt, nicht abgelesen — die App weiß es nicht, und ein vorbelegtes Feld wird im Ernstfall unverändert vorgelesen.

Wo wir jetzt stehen

| | gestern Abend | heute | |---|---|---| | Kern | 78 Dateien, 624 Prüffälle | **86 Dateien, 769 Prüffälle** | | App-Schicht | 62 Dateien, 42 Regeln | **66 Dateien, 45 Regeln** | | Kern-Prüfer | 6 Regeln | **8 Regeln** |

Dreizehn Lieferungen, alle Prüflisten abgearbeitet, kein Rückstand.

Der Satz, den ich mir aus diesem Tag mitnehme, ist der Gegensatz zu dem von gestern. Gestern: Ein Fehler, der eine Fehlermeldung erzeugt, ist der harmlose; der andere übersetzt sich sauber und tut nichts.

Heute geht es eine Stufe weiter:

Es gibt einen Fehler, der noch schwerer zu finden ist als der, der nichts tut — nämlich der, der alles richtig tut und es nur nicht sagt. Den siehst du nicht im Code. Den siehst du erst, wenn du selbst davorsitzt und nicht weiterkommst.

ORIGINALBEITRAG · TEIL 31

Der Anker war bereits oben

2. September 2026, ganzer Tag

Der Tag hatte zwei Hälften, und sie hätten kaum verschiedener sein können.

Die erste bestand daraus, dass ich immer wieder dieselben Fehler zurückgemeldet habe. In der zweiten sind sieben Lieferungen gebaut worden, Apple hat nach zehn Tagen die Freigabe geschickt, und am Ende hat mir die eigene Totmannschaltung einen Fehler gezeigt, der größer war als sie selbst.

Dazwischen liegt ein Satz, den ich mittags getippt habe und der den Rest des Tages verändert hat.

Ein Komma, das nicht kommen wollte

Angefangen hat es klein. Die Tiefenwache hat eine Schwelle — wie viel Wasser unter dem Kiel bleiben muss, bevor es klingelt. Ich wollte sie auf 1,5 Meter stellen.

> **die schwelle kann ich nicht über 0,99 setzen, kommen nicht an die vorkommastelle**

Das Feld ließ mich nur zwei Nachkommastellen tippen und schluckte alles davor. Es kam eine Berichtigung. Dann das nächste:

> **leerer bekomme ich das feld nicht, wieder ein fehler :-(**

Und danach noch eins:

> **null bleibt auch stehen und springt nicht zurück auf 0,5, wieder fehler**

Drei Anläufe für ein Zahlenfeld. Ich stand immer noch bei **Punkt 1** der Prüfliste.

Das Ärgerliche daran ist nicht das Feld. Es ist, dass ich jedes Mal bauen musste, um zu sehen, dass es wieder nicht stimmt. Jeder Anlauf kostet mich eine Viertelstunde am Mac und eine Übertragung aufs iPad.

Und es ist nicht der einzelne Fehler, der einen fertigmacht. Es ist das Gefühl, **alles mehrfach sagen zu müssen** — und die Arbeit kommt trotzdem nicht so zurück. Ein Fehler, den ich einmal beschreibe, darf beim zweiten Mal nicht wieder dastehen. Hier stand er über **mehrere Lieferungen** hinweg immer wieder da, und das ist etwas anderes als ein Versehen.

Die gepackten Dateien

Dann kam etwas, das mich mehr aufgeregt hat als das Feld.

Ich habe die Lieferungen als gepackte Archive bekommen — Dateien, die ich selbst entpacken und einsortieren soll. Dabei steht seit Tagen im Aufgabenheft, dass die Dateien direkt in den Projektordner gehören. Der Ordner liegt offen, Xcode nimmt neue Dateien von selbst auf, weil das Projekt seine Quellen aus dem Dateisystem liest und nicht aus einer Liste.

> **und warum hast du wie im aufgabenheft nicht entpackt?**

Und als klar wurde, dass ich vorher schon mehrere solche Pakete von Hand ausgepackt hatte:

> **ich brauche diese scheiß gepackten dateien nicht!!!!**

Es geht dabei nicht um Bequemlichkeit. Jedes Paket, das ich selbst auspacke, ist eine Stelle, an der eine Datei im falschen Ordner landen kann — und dann suche ich einen Fehler, den es gar nicht gibt.

Zwanzig Dokumente, eine Regel

Der eigentliche Bruch kam am frühen Nachmittag. Ich hatte mehrmals auf Regeln verwiesen, die längst aufgeschrieben waren, und bekam trotzdem etwas anderes.

> **steht alles im aufgabenheft, liest du das überhaupt?? dran halten tust du dich jedenfalls nicht**

Beim Nachsehen kam heraus: Die Regeln standen tatsächlich alle da. Nur eben verteilt über **mehr als zwanzig Dokumente** — jede in dem Übergabepapier des Tages, an dem sie entstanden war. Zusammen zweiundvierzig Stück, keine davon an einer Stelle, an der man alle sieht.

Und dann noch, als die Suche erst nach dem Bauen begann:

> **warum suchst du jetzt erst, dass hast du dir durchzulesen bevor die arbeiten beginnen und dich dann danach zu richten**

Seitdem stehen sie in **einer** Datei, und die wird zu Sitzungsbeginn gelesen. Ich habe an dem Nachmittag noch etwas anderes geschrieben, und das war eigentlich die ganze Anweisung:

> **denke lieber länger nach, prüfe und messe und liefere erst dann**

Danach lief es. Nicht weil irgendetwas Neues gebaut wurde, sondern weil vor jeder Lieferung erst gemessen wurde. Sieben Lieferungen am Nachmittag und Abend, sechs davon auf Anhieb bestanden.

Der Fehler, der auf meinem Boot gar nicht auffällt

Einer der Befunde hat mich beim Lesen stutzen lassen, weil er mich selbst nie getroffen hätte.

Die App kann Bordnetzdaten lesen. Auf der Bluebaerry hängt ein NMEA-2000-Netz, und das versteht sie. Der ältere Standard — **NMEA 0183**, die Sätze, die mit **$GPRMC** und **$SDDBT** anfangen — wurde zwar empfangen, gezählt und als „nicht verstanden" beiseitegelegt. **Gedeutet wurde kein einziger.**

Auf meinem Boot merkt das niemand. Auf einem Boot mit einem älteren Gateway hätte die App überhaupt nichts angezeigt: Verbindung steht, Daten kommen an, Bildschirm leer.

Ich hatte das vor Wochen schon einmal gesagt, in einem anderen Zusammenhang: Wir bauen das für alle Netzwerkarten, nicht nur für meine. Jetzt werden vierzehn Satzarten gedeutet — Position, Kurs, Fahrt, Tiefe, Wind, Log, Steuerkurs, Wassertemperatur.

Der Seewetterbericht kommt jetzt vom Amt

Zwei Sachen, die ich mir gewünscht hatte, sind an dem Nachmittag gebaut worden.

Die **Wellenperiode** — bis dahin stand im Logbuch nur die Wellenhöhe. Zwei Meter mit vier Sekunden Periode sind ein steiler, unangenehmer Seegang; zwei Meter mit elf Sekunden ist eine lange Dünung, in der man schlafen kann. Die Höhe allein sagt darüber nichts.

Und der **Seewetterbericht des Deutschen Wetterdienstes.** Nicht die Modellzahlen, die die App ohnehin schon holt, sondern der Bericht, wie er im Seefunk verlesen wird: Nord- und Ostsee, neun Gebiete, mit Seegang. Dazu die Küstenvorhersage mit acht Gebieten.

Mein Wunsch dazu war, dass die App das Gebiet einblendet, das zur Position des Schiffes passt, und die Wellenperiode mit ins Logbuch schreibt. Je genauer und detaillierter das Logbuch, desto besser.

Was ich nicht wollte: die Umlaute zurückwandeln. Der Rohtext des DWD schreibt **ae**, **oe**, **ue**, und ich habe zuerst gefragt, ob man das nicht lesbarer machen kann. Dann kam die Überlegung, dass eine Rückwandlung raten muss — bei Eigennamen und Abkürzungen geht das schief, und ein Wetterbericht, der Wörter erfindet, ist schlimmer als einer, der **Boeen** schreibt. Also bleibt der Wortlaut, wie ihn das Amt sendet. Er steht auf einem eigenen Blatt, umschaltbar zwischen gegliederter Darstellung und Wortlaut, mit dem Quellenvermerk darunter.

Die Mail um halb sechs

Am späten Nachmittag kam etwas, worauf ich seit dem 24. August gewartet habe.

Apple hat die kritischen Hinweise freigegeben.

Das ist die Erlaubnis, eine Mitteilung zu schicken, die den Lautlos-Schalter und den Fokusmodus durchbricht. Für einen Wecker, der um drei Uhr nachts sagt, dass das Boot aus dem Ankerkreis läuft, ist das keine Spielerei — ein Alarm, den „Nicht stören" schluckt, ist kein Alarm.

Zehn Tage Wartezeit, und man bekommt sie nicht einfach so — man muss begründen, wofür man sie braucht.

Was ich beim Lesen empfunden habe, war schlicht **Erleichterung**. Nicht Triumph, nicht abgehakt. Der Ankeralarm ist der Grund, aus dem diese App überhaupt angefangen hat, und ohne diese Erlaubnis hätte er nachts gegen einen Schalter verloren, den man abends aus ganz anderen Gründen umlegt.

Eingerichtet war es am selben Abend. Die Berechtigung ist übrigens die einzige in Xcode, die man **nicht** anklicken kann — sie muss von Hand in eine Datei geschrieben werden. Dafür gab es dann eine eigene Anleitung, Schritt für Schritt, weil ich das an einem Abend nach der Arbeit nicht suchen wollte.

Fünf Meter gibt es nicht

Zwischendurch ein kleiner Punkt, der zeigt, wie schnell etwas Erfundenes in eine Prüfliste rutscht.

In der Prüfliste stand, ich solle den Ankerradius auf 5 Meter stellen. Ich habe es versucht:

> **ankerradius geht auch nicht auf 5 Meter**

Ging auch nicht, und zwar zu Recht: Der Schieber geht von **10 bis 250 Meter**, und das steht auf dem Bildschirm daneben. Die 5 waren einfach hingeschrieben worden, ohne nachzusehen.

Das ist dieselbe Sorte Fehler wie die erfundenen Prüfsummen vom Vormittag — es sieht plausibel aus, es ist nur nicht gemessen. Eine Prüfliste, die eine Einstellung verlangt, die es nicht gibt, verbrennt meine Zeit an einer Stelle, an der gar nichts kaputt ist.

Der Anker war bereits oben

Und dann der Moment, um den es in diesem Teil eigentlich geht.

Ich saß zu Hause im Garten. Der Anker war im Garten gesetzt — das klingt albern, ist aber die einzige Art, eine Ankerwache zu prüfen, ohne dafür rauszufahren.

Ich habe die Totmannschaltung geprüft. Sie ist dafür gebaut, dass ich merke, wenn die Wache **aufhört** zu wachen — weil das iPad ausgeht, die App abstürzt, iOS sie schließt. Kommt eine Viertelstunde lang keine Position mehr an, meldet sie sich.

Sie hat funktioniert. Der Alarm kam, und mein erstes Gefühl war Erleichterung: **Sie meldet sich also wirklich.** Das ist die eine Sache, die diese Schaltung können muss, und sie konnte es.

Dann habe ich die App geöffnet.

> **die totmannschaltung hat gegriffen, hat alarm gemeldet, allerdings war nach öffnen der app, der anker bereits oben**

Und da war die Freude wieder weg. Der Wecker hat geklingelt, und was ich danach in der Hand hatte, war nichts.

Der Ankerpunkt lag nur im Arbeitsspeicher. Position, Radius, Ankergrund, meine Bemerkung zum Ankerplatz, der gemerkte Fallpunkt — alles weg, sobald iOS die App beendet.

Das heißt: Genau in dem Fall, für den die Totmannschaltung überhaupt gebaut ist, weckt sie mich, ich stehe nachts an Deck, ich mache das iPad an — und die Ankerwache ist leer. Ich weiß nicht mehr, wo der Anker gefallen ist, und ich weiß nicht, ob das Boot noch dort liegt.

Die Totmannschaltung hat bei ihrem ersten Einsatz etwas gefunden, das größer war als sie selbst.

Eine Wache, die sich still wieder aufnimmt, lügt

Bei der Frage, was die App nach einem Neustart tun soll, gab es zwei Möglichkeiten: die Wache selbst wieder scharf stellen, oder fragen.

Ich habe gesagt: **immer fragen.**

Der Grund ist derselbe, aus dem hier schon manches nicht gebaut wurde. Eine Wache, die sich nach einem Neustart still selbst wieder aufnimmt, behauptet damit, sie hätte durchgewacht. Hat sie aber nicht. In der Lücke — fünf Minuten oder fünf Stunden — hat niemand hingesehen, und das Boot kann in dieser Zeit geschwojt sein oder der Anker geslippt.

Also steht die wiedergefundene Wache jetzt ganz oben auf dem Ankerbildschirm, mit der Länge der Unterbrechung, dem Ankerpunkt in Grad und Dezimalminuten, dem Radius, dem Ankergrund und meiner Bemerkung. Zwei Knöpfe: wieder aufnehmen, oder verwerfen, weil der Anker oben ist. Und darunter der Satz, dass ich vorher prüfen soll, ob das Boot überhaupt noch da liegt.

Die Länge der Unterbrechung zählt weiter, solange die Tafel dasteht. Sie endet nicht damit, dass ich die App öffne — sie endet, wenn ich antippe. Solange ich davorstehe und überlege, wacht auch niemand.

Die Lieferung, die nie ankam

Zum Schluss noch etwas, das mir gefallen hat, obwohl es ein Fehler war.

Vor dem Ablegen der letzten Lieferung wurde nicht einfach kopiert, sondern vorher der ganze Projektordner gegen den Arbeitsstand gemessen — **alle 166 Swift-Dateien einzeln**, mit Prüfsumme. Erwartet waren vier Unterschiede: die vier Dateien der neuen Lieferung.

Es waren fünf.

Die fünfte war eine Datei vom Nachmittag, die berichtigt worden war und **nie bei mir angekommen ist.** Ich hatte seit 16:41 Uhr mit der alten Fassung gebaut. Auf den Prüfpunkt, an dem ich gerade saß, hatte das keine Auswirkung — aber gefunden hat es niemand durch Nachdenken. Gefunden hat es das Messen.

Genau das ist der Punkt, an dem ich heute Mittag gestanden habe. Eine Lieferung, die „geliefert" heißt und nicht auf meinem Rechner liegt, ist keine Lieferung. Und sie fällt sonst erst auf, wenn irgendein Prüfpunkt scheitert und man den Grund an einer ganz anderen Stelle sucht.

Wo wir jetzt stehen

	gestern Abend	heute
Kern	86 Dateien, 769 Prüffälle	**96 Dateien, 870 Prüffälle**
App-Schicht	66 Dateien	**70 Dateien**
Lieferungen am Tag	dreizehn	**acht, sechs geprüft und bestanden**

Sechs Prüflisten abgearbeitet, eine liegt für morgen bereit.

Was ich mir vom Vormittag merke, ist nicht der einzelne Fehler. Es ist, dass derselbe über mehrere Lieferungen hinweg stehen blieb, obwohl ich ihn beim ersten Mal beschrieben hatte. Ein Fehler ist ein Fehler. Ein Fehler, den man zum dritten Mal erklärt, ist eine Arbeitsweise.

Und der Satz, den ich aus dem ganzen Tag mitnehme, ist der, den ich mittags selbst getippt habe — und den ich abends von der anderen Seite wiederbekommen habe:

Prüfen und messen kommt vor dem Liefern. Auch dann noch, wenn schon geliefert ist.

ORIGINALBEITRAG · TEIL 32

Ein Kreis, den es nicht gab

3. September 2026, ganzer Tag

Am Nachmittag stand ich mit dem iPad in der Hand da und habe auf einen Bildschirm gesehen, der mir zwei Zahlen gezeigt hat. Vierzehn Meter vom Ankerpunkt. Alarmradius zehn Meter.

Und nichts ist passiert.

Mein erster Gedanke war nicht, dass die Anzeige falsch sein könnte. Mein erster Gedanke war: **Die App weckt nicht.**

Das ist der Punkt, an dem so eine Sache gefährlich wird. Ein Ankeralarm hat genau eine Aufgabe, und wenn ich anfange, ihm nicht mehr zu glauben, kann ich ihn auch gleich ausschalten. Ich habe in dem Moment nicht überlegt, ob da vielleicht ein Anzeigefehler steckt. Ich habe überlegt, ob ich einer Software vertraue, die mich nachts wecken soll.

Was wirklich los war

Gerechnet hat sie richtig.

Ich hatte den Anker mit fünfundfünfzig Metern gesetzt und danach den Regler auf zehn gezogen. Der Regler stellt aber nur ein, was beim **nächsten** Mal gelten soll. Die laufende Wache hat weiter über ihre fünfundfünfzig Meter gewacht — und bei vierzehn Metern ist das kein Alarm, das ist ein Boot, das brav in seinem Kreis liegt.

Falsch war nur das, was auf dem Schirm stand. Die Kopfzeile hat den Regler angezeigt statt den Kreis, über den gewacht wird. Die Scheibe mit dem gestrichelten Ring auch.

Ich habe also einen Kreis gesehen, den es nicht gab.

Es ist behoben. Der Regler wirkt jetzt sofort auf die laufende Wache, der Ankerpunkt bleibt liegen, und in der Liste der Vorgänge steht danach eine Zeile, dass ich den Radius geändert habe. Aber der Ärger bleibt: Das war kein Rechenfehler. Das war ein Bildschirm, der etwas behauptet hat.

Sachen, die niemand gesucht hat

Das Merkwürdige am heutigen Tag ist, dass fast alles, was gefunden wurde, in keiner Aufgabe stand.

Da war eine Zeile im Gezeitenrechner, die den Tiefgang meines Bootes holen sollte. Sie hat unter einem Namen gesucht, den es nicht gibt — einen Buchstaben daneben, englisch statt deutsch. Was das gekostet hat: Der Satz „Beim tiefsten Stand bleiben so und so viel Meter unter dem Kiel" ist nie erschienen. Und mit ihm nicht die Warnung, dass das Boot aufsetzt.

Das ist die wichtigste Zeile in dem ganzen Rechner. Sie war seit dem einunddreißigsten August tot, und **es hat nichts gemeldet.** Keine Fehlermeldung, keine leere Stelle. Der Satz wurde einfach nicht gebaut.

Dasselbe an einer anderen Stelle: Die Wetterangaben, die seit Tagen in jeden Logbucheintrag geschrieben werden, haben zum Anzeigen nach einer Kennung an der Zeile gesucht. Die Kennung gab es nicht. Also standen Bewölkung, Luftdruck und Sicht fünf Tage lang im Speicher und nirgends auf dem Schirm.

Und der dritte Fall von derselben Sorte, der mich am meisten stört: Wenn die Windrichtung fehlt, stand auf dem Wetterblatt **Nord**. Nicht ein Strich, nicht „unbekannt". Nord.

Bei einer Zahl, die fehlt, springt die Rechnung auf Null — und null Grad ist nun mal Norden. Dasselbe Missverständnis gab es heute schon dreimal an anderen Stellen: aus einer fehlenden Wellenhöhe wurde „spiegelglatte See", aus einer fehlenden Position wurde ein Punkt im Golf von Guinea.

Was fehlt, ist keine Null. Der Satz klingt banal. Er hat heute vier Befunde erklärt.

Was mir selbst um die Ohren geflogen ist

Ich habe heute Abend die Prüfliste angesehen, die ich am Donnerstag mit aufs Boot nehmen soll. Sechsundsechzig Punkte, die ich dort abhaken will.

Statt der Kästchen stand da sechsundsechzigmal ein Platzhalter. Irgendein Werkzeug hatte ihn nicht ersetzt, und **niemand hat das fertige Blatt angesehen** — es wurde gebaut, abgelegt, für erledigt gehalten.

Dazu kam noch, dass in den Prüflisten des Tages die fett gesetzten Sollwerte gar nicht fett waren. Auch das fällt nur auf, wenn man das fertige Blatt anschaut statt den Text, aus dem es gemacht wurde.

Und dann die Uhrzeit. Mir wurde am frühen Abend geschrieben, der Sonnenuntergang sei schon gewesen und die Prüfung der Ankerlaterne verpasst. War sie nicht — es war noch nicht mal acht. Zwei Stunden Unterschied zwischen zwei Uhren, und eine davon war einfach nicht nachgesehen worden.

Es sind Kleinigkeiten. Aber es sind Kleinigkeiten derselben Art wie die großen: etwas hingeschrieben, statt nachzusehen.

Wo ich am Abend stehe

Ich bin ehrlich: **An der nativen App hat der Tag auch heute nicht viel gebracht.**

Es sind Sachen dazugekommen, die stimmen — die Wache merkt sich jetzt, was ihr in der Nacht zugestoßen ist, und das überlebt auch, wenn das iPad die App abschießt. Vorher stand am Morgen „ein Eintrag" da, egal was nachts los war. Das ist richtig so, und es war nötig. Aber es fühlt sich noch nicht an wie Vorankommen. Es fühlt sich an wie Aufräumen.

Die Webfassung dagegen hat ordentlich aufgeholt. Da ist heute wirklich etwas passiert: Die Ankerlaterne meldet sich, der Seenotruf spricht den richtigen Wortlaut, die Position steht überall in der Schreibweise, die man ins Funkgerät gibt. Und seit heute Abend gibt es dort eine Tiefenwache, die weckt, wenn zu wenig Wasser unter dem Kiel steht — mit einem Filter für die Aussetzer, die es beim Schwojen zwangsläufig gibt.

Am Abend war dann alles durch. Vierzehn Lieferungen an einem Tag, jede einzelne geprüft und bestanden — auch die beiden, die erst nach dem Abendessen fertig wurden. Und die Ankerlaterne hat pünktlich um kurz vor acht in beiden Fassungen umgeschaltet, ohne dass ich etwas antippen musste. Beide nennen denselben Sonnenaufgang, auf die Minute. Das ist eine Kleinigkeit, aber es sind zwei getrennt gebaute Rechnungen, und sie kommen auf dasselbe.

Neun Sachen sind heute aufgefallen, die auf keiner Liste standen. Ich habe gemischte Gefühle dabei. Es beunruhigt mich, dass sie da waren — manche seit Tagen, mitten in Funktionen, die ich für fertig gehalten habe.

Aber **froh bin ich trotzdem, dass sie aufgefallen sind.** Lieber jetzt am Schreibtisch als in einer Nacht vor Anker.

Was ich mitnehme

Gestern hieß die Lehre: erst messen, dann liefern.

Heute kommt eine dazu, und sie ist unbequemer. Wenn eine Datei bei einer anderen etwas sucht — einen Namen, eine Kennung, einen Wert —, dann muss jemand **nachzählen**, ob es das gibt. Sonst passiert genau das, was heute dreimal passiert ist: Die Sache ist still. Kein Fehler, kein Hinweis, nichts. Nur eine Zeile, die nie erscheint.

Und das andere, was ich mir merke, ist nichts Technisches. Als heute Nachmittag nichts passiert ist, habe ich der App misstraut und nicht dem Bildschirm.

Das sagt mehr über den Fehler als jede Messung.

ORIGINALBEITRAG · TEIL 33

Das Bild war schöner als die Wirklichkeit

4. September 2026, ganzer Tag

In meiner SKS-Ausbildung habe ich einen Seewetterbericht in der Hand gehabt, wie er aus dem Drucker kam. Ein schmaler Streifen, vielleicht zehn Zentimeter breit, in einer Schrift, bei der jeder Buchstabe gleich viel Platz bekommt. Sah aus wie ein Fax. Der Deutsche Wetterdienst hat ihn damals so über Funk verschickt, und die Empfänger an Bord haben ihn so ausgedruckt — oder nur auf dem Bildschirm gelesen, wer keinen Drucker hatte.

Ich hätte nicht gedacht, dass mir dieser Streifen drei Tage lang einen Fehler verstecken würde.

Drei Tage lang falsch, und keiner hat es gesehen

Der Seewetterbericht in PELARON kam seit Tagen an, aber er sah nicht richtig aus. Zeilen fehlten. Nicht irgendwelche — genau die, die im Original als Fortsetzung unter einer anderen Zeile stehen: der zweite Teil einer Windangabe, der Rest einer Warnung. Diese zweiten Zeilen waren weg. Das Ende des Berichts, die Fußnote mit den rechtlichen Hinweisen, war dagegen plötzlich achtzehn Zeilen lang.

Der Grund war der Fernschreiber. Der Bericht, so wie der Wetterdienst ihn heute noch bereitstellt, trägt am Anfang, am Ende und an jedem Zeilenende Steuerzeichen aus der Zeit der Papierdrucker. Am Zeilenende stehen zwei Wagenrückläufe statt einem. Ein Programm, das Zeilen zählt, sieht dort eine Leerzeile, wo keine ist — und eine Leerzeile bedeutete in unserer Deutung: Hier endet ein Absatz, die nächste Zeile ist keine Fortsetzung mehr.

Warum hat das drei Tage keiner gemerkt? Weil jeder Prüftext, mit dem die Deutung gebaut worden war, vorher durch einen kleinen Aufbereiter gelaufen war. Der hatte die Steuerzeichen sauber entfernt. Alle Prüfungen bestanden. Sie prüften nur eine Fassung des Berichts, die es in Wirklichkeit nicht gab.

Und dann noch einmal, am selben Tag

Am Vormittag ging der Seewetterbericht auch in die Webfassung. Ich habe ein Bild bekommen, wie das Blatt aussehen würde: ordentlich, mit Abstand zum Rand, sauber gegliedert. Dann habe ich es auf dem iPad geöffnet.

Total hässlich. Die Kopfzeile klebte am Rand und lief über den weißen Bereich hinaus, der Text stand ohne Abstand an der Kante, und am Schließen-Kreuz saß ein orangefarbener Kreis, der da nichts zu suchen hatte. Was vorher mit den Kacheln wirklich hübsch ausgesehen hatte, war jetzt ein Klotz.

Mein erster Gedanke war nicht, dass da ein Programmierfehler steckt. Mein erster Gedanke war: Da hat die KI wieder einmal etwas zerstört, was vorher vom Design her gestimmt hat.

Die Ursache war dieselbe wie beim Fernschreiben, nur an anderer Stelle. Das Bild, das ich vorher gesehen hatte, war mit einer selbst geschriebenen Ersatzvorlage für die Gestaltung gerendert worden — weil die echte Stildatei der Seite nicht vorlag. Der Ersatz gab dem Blatt einen Innenabstand, den die echte Datei an dieser Stelle gar nicht kennt. Wieder war gegen eine Fassung geprüft worden, die es nicht gibt.

Die echte Stildatei liegt jetzt im Übergabeordner. Wer künftig ein Bild vom fertigen Blatt macht, macht es damit — oder gar nicht.

Der Nachmittag

Ich hatte gesagt: Baue an der nativen App weiter, die ist wichtiger. Beim Auslesen der Webfassung fiel etwas auf, das nichts mit dem Wetter zu tun hatte. Derselbe Törn stand mit zwei verschiedenen Motorstundenzahlen auf dem Schirm. In der Törnhistorie mit der Summe der Zählerstände, im Törnbericht mit ihrer Differenz. Aus fünf Betriebsstunden wurde in der Historie eine Zahl, die eher nach dem Zählerstand selbst aussah als nach einer Fahrt.

Ein Feld, zwei Bedeutungen. Die Webfassung führt dort eine Laufzeit je Eintrag und darf addieren. Die native App führt einen Zählerstand und darf es nicht. Beim Nachbauen war die Rechenart mitgekommen, die Bedeutung nicht.

Und als die Berichtigung geprüft werden sollte, kam die Überraschung des Tages: Die Prüffälle des Kerns liefen gar nicht. Seit dem dritten September nicht mehr. Drei Prüfdateien hatten den Bau blockiert — Prüffälle in der falschen Klasse, falsche Feldnamen, Vergleiche gegen einen Wert, der fehlen darf. Und die Zahl, die mir jeden Tag als „Prüffälle bestanden“ genannt worden war, war die Zahl der Zeilen im Quelltext, in denen das Wort Prüffall vorkommt. Gezählt, nicht gelaufen.

Ich habe gefragt, ob die langen Prüfungen der letzten Tage damit vergebens waren. Die Antwort war: zum Teil. Alles, was ich selbst am Bildschirm geprüft hatte, galt weiter. Die Webfassung war nicht betroffen. Aber vier Lieferungen für den Kern standen als bestanden im Heft, ohne dass ein einziger ihrer Prüffälle gelaufen war.

Was der erste echte Lauf fand

Als der Kern wieder baute, fielen sechs Prüfungen durch. Keine davon war ein Fehler im Programm, alle waren Fehler in den Prüfungen selbst — und jede hatte eine Geschichte. Manche verlangten von der Bordapotheke, dass sie das Zeichen „bleibt auf dem Gerät“ trägt, obwohl im Quelltext seit Anfang September mit Begründung steht, dass sie es nicht tragen darf. Eine verglich eine Windrichtung, die praktisch Nord war, mit Nord und nannte den Abstand riesig — weil sie den Kompass nicht als Kreis las. Eine scheiterte an Mikrosekunden, weil eine Zeitangabe beim Speichern auf ganze Sekunden gerundet wird.

Und eine hatte eine falsche Zahl im Prüffall stehen, und hätte beinahe den Code verbogen. Die Wiedergabe einer Aufzeichnung setzt nach einer Pause bei der letzten gespielten Zeile wieder an, nicht beim Stand der Uhr. Das ist richtig so: Der Abstand zwischen zwei Sätzen ist der Inhalt einer Aufzeichnung. Wer bei der Uhr fortsetzt, verkürzt ihn — und misst dann nicht mehr die Aufzeichnung, sondern seine eigene Pause. Der Prüffall hatte das Gegenteil verlangt. Wir haben den Prüffall geändert, nicht den Code.

Was bleibt

Zwei Regeln, die an einem Tag zweimal gebrochen wurden und jetzt im Heft stehen. Ein Prüfbild wird gegen die echte Stildatei gerendert, sonst misst es den eigenen Ersatz. Und eine Zahl, die auch bei einem Fehlschlag herauskommt, ist keine Prüfung — zu jeder Lieferung für den Kern gehört die Zeile, die der Prüfläufer selbst ausgibt, mit der Zahl der gelaufenen Fälle und der Zahl der Fehler. Ohne sie steht im Heft nichts von bestanden.

Der Seewetterbericht kommt weiter als Fernschreiben an, mit allen Steuerzeichen, und wird auch so gespeichert. Nur die Deutung sieht jetzt darüber hinweg. Der schmale Streifen aus der SKS-Ausbildung war die ganze Zeit die Wahrheit. Man musste ihn nur einmal so ansehen, wie er wirklich ist.

ORIGINALBEITRAG · TEIL 34

Verbunden, aber still

5. bis 7. September 2026

Am Samstagmorgen war das iPad wieder da. Zwei Prüflisten lagen seit dem Abend davor, die Motorstunden in der Törnhistorie und der neue Analysenbildschirm mit dem Saisonrückblick. Beide bestanden, in einer halben Stunde. Dann habe ich etwas anderes auf den Tisch gelegt, das mit der App nur am Rand zu tun hatte: ein Hardwaretagebuch und den Entwurf einer Steuerung.

Was ich eigentlich wissen wollte

Die Idee war schon länger da. Auf Bluebaerry hängen Kühlschrank, Instrumente, Lichter, die Standheizung und die Toilette an einer Schalttafel mit Wippschaltern, und jedes Gerät, das mir seinen Zustand meldet, tut das über eine eigene App. Victron hat eine, der Autopilot hat eine, das Echolot spricht mit dem Kartenplotter. Ich wollte eine Platine, die die Verbraucher schaltet und zurückmeldet, was wirklich fließt — und eine App, die alles zeigt.

Vorgelegt habe ich das der KI mit einer klaren Erwartung: einen Vergleich, weltweit. Hat das schon einmal jemand gemacht? Kann man so etwas kaufen? Und während ich darauf wartete, kam der Gedanke, der die Sache erst rund gemacht hat: Wenn PELARON ohnehin auf dem iPad läuft und den Bus liest, warum dann nicht auch die Steuerung darüber — über eine App statt über fünf.

Die Antwort war eine Einschätzung, keine Liste von Produkten. Der Grundgedanke sei besser als das, was man kaufen kann: Fast jedes Produkt am Markt ersetzt die Schalttafel und wird damit selbst zum Einzelpunkt des Versagens — hier bleibt der Wippschalter die Wahrheit, und die Software darf nur einschränken. Und es schließe die eigentliche Lücke von PELARON: Bis heute liest und merkt sich die App, aber sie hat keine Hände.

Was die App dafür heute hat, war schnell gemessen: nichts. Kein Begriff für Verbraucher, kein Schalten, kein Relais im ganzen Kern. Aber es gibt eine Stelle, an der es anknüpfen kann. Die App unterscheidet bereits, wie sicher ein Wert ist — übernommen, gerechnet, gemessen, beobachtet — und zeigt das an. Meine Kette für einen Schaltbefehl, gesendet, bestätigt, Ausgang aktiv, Strom fließt, ist dieselbe Idee für einen Zustand statt für einen Messwert. Das muss nicht erfunden werden, nur übertragen.

Die große Platine

Dann kam der Rat, mit dem ich nicht einverstanden war. Erst eine kleine Platine bauen, vier Kanäle, daran lernen, dann die große. Klingt vernünftig. Ich mache es genau andersherum. Für Bluebaerry, das eigene Boot, würde ich immer die große Version wählen. Bei Tests kann ich alles testen und muss danach nicht wieder von vorn anfangen. Was man am kleinen Aufbau lernt, ist nicht das Schwierige; Wärme, Strompfade, Masse und die Firmware für viele Kanäle zeigen sich erst in voller Größe. Die Platine wird gerade so gezeichnet: WLAN, Bluetooth und Mobilfunk auf einem Brett.

Und es ist ein Winterprojekt. Ein Blick aufs Datum reicht: In zwei Monaten kommt das Boot aus dem Wasser, dann ist die Saison vorbei, und Löten, Firmware und App konkurrieren mit nichts mehr. Die KI hatte als eigentliches Risiko „die Abende“ genannt, dass die Hardware die Software auffrisst. Im Winter gibt es diese Konkurrenz nicht.

Was von ihrer Einschätzung geblieben ist: In den Stromkreis der Navigationsinstrumente gehört kein selbstgebautes Relais, dort hängt der Autopilot. Dieser Kanal wird gemessen, das Relais bleibt unbestückt. Und die Standheizung wird der erste Kanal, der in Betrieb geht — sie ist an Bord, der Nutzen ist sofort da, und sie beantwortet an einem einzigen Verbraucher die Frage, ob PELARON nur ausschalten oder auch einschalten darf.

Der Sonntag

Am Samstag hatte ich der KI noch gesagt: Entwickle die native App so weit, dass einem Törn auf dem Wasser und einem ausführlichen Test nichts mehr entgegensteht. Sie hat daraufhin die Aufzeichnung des Bordnetzes umgebaut. Die endete bisher nach einer Viertelstunde, weil sie alles im Arbeitsspeicher hielt und dann aufhörte — und die Liste für den Donnerstag verlangt eine halbe Stunde AIS am Stück. Jetzt schreibt sie in Blöcken auf die Platte, legt die Datei beim Start an, und der Bildschirm bleibt an, solange sie läuft.

Ich habe am Sonntag nichts davon geprüft. Ich habe an der Hardware gearbeitet, den ganzen Tag. Es gab auch Momente an der App, aber die Idee der Steuerung hat sich weiterentwickelt, und beim Ausprobieren und Setzen der ersten elektronischen Bauteile kam der Spaß an der Sache wieder durch, wie zu Jugendtagen. Die App war zwei Tage still. Das war in Ordnung.

Der Montag am Schreibtisch

Am Montagnachmittag saß ich wieder am iPad, ohne Boot, ohne Bordnetz. Die Frage war, wie man eine Aufzeichnung prüft, wenn nichts ankommt. Die erste Antwort lautete: liegen lassen bis Donnerstag. Ich habe es trotzdem probiert, und nach „Mitschnitt beenden“ kam der Knopf zum Verschicken der Datei nicht. Die App zeigte ihn nur, wenn mindestens eine Zeile aufgezeichnet war. Ohne Bordnetz kommt keine Zeile. Die Datei gab es trotzdem längst, seit dem Start. Der Bildschirm hat eine Datei geleugnet, die da war.

Die Datei, als ich sie dann hatte, trug keine einzige Zeitangabe. Der Kopf wurde beim Start geschrieben, die Startzeit aber erst mit der ersten Zeile gesetzt — in jeder Datei hätte sie gefehlt, nicht nur in der leeren. Zwei Lieferungen später stand die Startzeit im Kopf, und die Datei vermerkte, wann die Verbindung stand, wartete oder verloren ging. Denn das war der nächste Punkt: Vierzig Sekunden ohne WLAN sahen in der Aufzeichnung genauso aus wie vierzig Sekunden ohne Echolot. Am Donnerstag will ich wissen, ob die Tiefenwache Aussetzer zählt. Ohne diesen Vermerk hätte ich es nicht unterscheiden können.

Dann der Test, den ich am Schreibtisch nicht für möglich gehalten hatte. Der Mac spielt das Gateway: ein Befehl im Terminal, der auf dem Anschluss des Bordnetzes lauscht, im iPad die Adresse des Macs eingetragen. Ich habe es gestartet, und es passierte rein gar nichts. Weder am iPad noch am Mac. Ich habe gefragt, ob es daran liegt, dass das iPad noch per Kabel am Mac hängt. Dann ein Blick auf den Bordnetz-Bildschirm: verbunden, aber still. Der Mac hatte die Verbindung längst angenommen — der Befehl schweigt, wenn er das tut. Und in der Zeile darunter stand, was das iPad in der Zwischenzeit erlebt hatte: mehrere Abbrüche, Dutzende Versuche, jedes Mal binnen Sekunden wieder verbunden. Die Wiederverbindung tat genau das, was der Quelltext versprach. Ich hatte eher mit einem Fehlverhalten meinerseits gerechnet. Ich bin froh, dass es ohne weitere Fehler bestanden hat.

Was noch auffiel

Beim Benutzen, nicht beim Prüfen: Wenn ich den Anker über die Ankerwache setze, steht davon nichts im Logbuch. Wenn ich ihn über die Logbuchseite wieder hole, steht das drin. Die Webfassung schreibt beides. Das ist notiert und kommt als Nächstes.

Und am Abend, beim Setzen der Gesamtausgabe, fand die KI zwei Fehler in der Ausgabe vom Donnerstag, die seit vier Tagen auf pelaron.de lag: rohe Formatierungszeichen in den Zwischenüberschriften des vorigen Teils, und alle Seitenzahlen im Inhaltsverzeichnis um eins zu klein, weil der Setzer mit zwei Verzeichnisblättern gerechnet und eines gesetzt hatte. Niemand hatte das fertige Heft aufgeschlagen. Die Regel, die genau das verlangt, steht seit dem dritten September im Heft — und galt offenbar nur für Prüflisten. Jetzt gilt sie für alles, was gesetzt wird.

Was bleibt

Die App ist bereit für einen Törn. Die Aufzeichnung läuft stundenlang, der Bildschirm bleibt an, die Datei weiß, wann sie begonnen hat und wann die Verbindung fehlte. Was noch offen ist, braucht das Boot: die Datei vom echten Bus, AIS eine halbe Stunde, eine Nacht Ankerwache. Und ab jetzt zählt jede Aufzeichnung doppelt. Im Winter, wenn das Boot an Land steht, ist die Wiedergabe das einzige Boot, das ich habe. Was ich jetzt nicht mitschneide, fehlt beim Programmieren.

Das Hardwaretagebuch bekommt auf pelaron.de eine eigene Reihe. Es trägt die Adresse im Fuß schon.

ORIGINALBEITRAG · TEIL 35

Ein Ankern, ein Eintrag

8. September 2026, abends

Tagsüber war Arbeit. Die Vision PELARON ist momentan eher Privatsache als Haupterwerb, und so bleibt für sie der Abend. Heute waren es gut anderthalb Stunden am Schreibtisch, und es war ein guter Fluss: viele Dinge abgearbeitet, Lieferung auf Lieferung, Prüfliste auf Prüfliste, und am Ende war die Liste dessen, was am Schreibtisch noch zu prüfen wäre, leer bis auf einen einzigen Schritt.

Was gestern Abend aufgefallen war

Gestern hatte ich beim Benutzen gemerkt, dass der Anker, den ich über die Ankerwache setze, nicht im Logbuch steht — während „Anker auf" über die Logbuchseite drinsteht. Heute Nachmittag hat die KI das nachgemessen, bevor sie gebaut hat, und das Ergebnis war deutlicher, als ich es vermutet hatte: Die Wache hatte den Logbuchdienst der App noch nie aufgerufen. Nicht beim Setzen, nicht beim Alarm um drei, nicht beim Aufholen. Kein Vorgang der Wache stand je im Logbuch.

Und unter der Korrektur des Ankermittelpunkts stand seit sechs Tagen der Satz: „Der vorherige Punkt steht im Logbuch." Er stand dort nicht. Ein Satz auf dem Bildschirm ist eine Behauptung — die Regel dazu gibt es seit dem dritten September, und genau so ein Satz hatte sich trotzdem gehalten.

Jetzt schreibt die Wache dieselben neun Ereignisse ins Logbuch wie die Webfassung, wortgleich: Anker gesetzt, Anker geborgen, Ankerposition korrigiert, die Alarme, die Tiefenwache. Aus einer Wiedergabe kommt weiterhin nichts ins Logbuch — eine abgespielte Nacht darf nicht mit dem heutigen Datum im Logbuch landen.

Im Wachstand der App gab es bisher acht Stellen, die einen Vorgang in die Tafel schrieben, und jede einzelne hatte das Logbuch vergessen. Jetzt gibt es eine Stelle, durch die alles geht.

Zwei Einträge für ein Manöver

Beim Bauen fiel das Nächste auf. Wer auf der Logbuchseite „Ankern" tippt, bekam einen Handeintrag „Anker gesetzt" und wurde zur Wache geführt; setzt er dort den Anker, schrieb jetzt auch die Wache einen Eintrag. Zwei Zeilen für ein Ankern, und der Saisonrückblick zählte zweimal. Die Webfassung hat dieselben zwei Einträge, zählt aber nur die von Hand — dort geht die Rechnung zufällig auf.

Ich hatte zwei Fassungen zur Wahl: Entweder schreibt der Knopf im Logbuch nichts mehr und führt nur zur Wache, oder der Rückblick zählt zwei Einträge binnen zehn Minuten als einen. Ich habe die erste genommen. Ausschlaggebend war, dass der Nutzer bei Betätigung der Schnellauswahl trotzdem zur Eingabe der sicherheitsrelevanten Daten gezwungen wird — die passende Kettenlänge bei der jeweiligen Wassertiefe, der Ankergrund, der Alarmradius. Ein Knopf, der das alles überspringt und trotzdem „Anker gesetzt" schreibt, wäre eine Einladung, es zu überspringen.

Das Gegenstück kam gleich danach: Wer nachts im Logbuch „Anker auf" drückt, beendet damit jetzt auch die Wache, und die Wache schreibt den Eintrag — mit Dauer, Radius und Grund. Bisher lief sie einfach weiter, und morgens fand man den Handeintrag neben einer Wache, die noch wachte. Läuft keine, schreibt der Knopf wie bisher von Hand; wer ohne Wache geankert hat, soll es weiter eintragen können.

Elf Zeilen für einen Ausfall

Dann der Test, der gestern offen geblieben war: die Aufzeichnung des Bordnetzes soll vermerken, wann die Verbindung weg war und wann sie wieder da war. Der Mac spielte wieder das Gateway. Ich habe die Verbindung gekappt, eine knappe Minute gewartet, wieder verbunden, die Datei verschickt.

In der Datei standen für diesen einen Ausfall elf Vermerkzeilen. „Verloren", „wartet", „verloren", „wartet" — je ein Paar für jeden gescheiterten Wiederholungsversuch, und die Versuche kommen in immer längeren Abständen. Eine Nacht mit wackligem WLAN hätte Hunderte solcher Zeilen erzeugt. Die Prüfliste hatte etwas anderes versprochen: eine Zeile für weg, eine für wieder da, keine doppelt. Die KI hatte nicht nachgerechnet, was ein Wiederholungsversuch in der Datei hinterlässt, und es aufgeschrieben, als wüsste sie es.

Dabei stand das Wichtige drin: Der Ausfall dauerte 48 Sekunden, und die Wiederholung lief genau so, wie der Quelltext es sagt. Nur die Form war falsch. Jetzt ist ein Ausfall zwei Zeilen — „Verbindung verloren" mit Uhrzeit, „Verbindung steht" mit Uhrzeit, Dauer und Zahl der Fehlversuche. Ob ein Wechsel eine Zeile wert ist, entscheidet der Kern; „wartet" gibt es in der Datei nicht mehr.

Beim zweiten Lauf spielte der Mac die alte Aufzeichnung vom Ankerplatz Zeile für Zeile ein, und die Datei sah aus, wie sie soll: Kopf, Datenzeilen, verloren, steht, Ende. Damit waren zugleich die letzten beiden Schritte erledigt, die seit Tagen auf einen laufenden Bus gewartet hatten.

Die Analysen sind vollständig

Als aus der Wache nichts mehr offen war, hat die KI sich das Nächste genommen, was die Webfassung hat und die native App nicht: die drei Reiter unter Analysen. Saison gab es seit letzter Woche. Heute kamen Kosten und Motor dazu, und die Kachel „Noch nicht gebaut" ist von dem Blatt verschwunden.

Kosten ist aus der Bordkasse: gesamt, Buchungen, Schnitt, Ausgabenarten, darunter Balken nach Art, nach Monat und danach, wer gezahlt hat — die letzten beiden nur, wenn es mehr als einen gibt. Ohne Ausgaben steht da, dass keine da sind, und nicht null Euro.

Motor war die interessantere Sache, weil hier die Lehre von letzter Woche durchschlägt: In der Webfassung ist die Motorstunde im Logbuch eine Laufzeit, in der nativen App ein Zählerstand. Also wird nichts addiert. Betriebsstunden im Jahr sind letzter minus erster Stand des Jahres, und die Stunden zwischen dem letzten Stand eines Jahres und dem ersten des nächsten zählen zu keinem — das steht so auf dem Schirm. Einen „längsten Motorlauf", wie die Webfassung ihn zeigt, gibt es nicht, weil kein Eintrag eine Dauer trägt; was es gibt, ist der größte Zuwachs zwischen zwei Einträgen, und so heißt er auch. Der Zählerstand kommt von Hand, sonst vom Bus, sonst aus dem Logbuch, und die Kachel sagt, woher. Die Wartungsfrist rechnet mit dem Stand, der gilt — auch dann, wenn unter „Mein Boot" keiner steht.

Was bleibt

Der Kern hat heute 39 Prüffälle dazubekommen und jeden Lauf mit der Zeile bestanden, die seit letzter Woche Pflicht ist: die des Läufers selbst, mit Zahl und null Fehlern. Die App ist damit an dem Punkt, an dem ich sie vor dem Törn haben wollte. Was noch offen ist, braucht das Boot — oder, für den letzten Schritt, eine Aufzeichnung auf dem iPad.

Ein guter Fluss. Viele Dinge abgearbeitet.

ORIGINALBEITRAG · TEIL 36

Einundzwanzig von einundzwanzig

9. September 2026, abends

Neben der hauptberuflichen Arbeit sitze ich täglich noch einmal vier bis fünf Stunden an allem, was PELARON ist; am Wochenende, wenn privat nichts anliegt, deutlich mehr. Heute war es ein langer Tag in kurzen Stücken — morgens, nachmittags, abends —, und am Ende steht eine Zahl, auf die ich seit dem 30. August hingearbeitet habe: Die native App hat einundzwanzig Bereiche, und einundzwanzig sind gebaut.

Die Nachrichten für Seefahrer

Der vorletzte Bereich war das BSH. Die Webfassung hat dort drei Kacheln und einen Knopf, der die amtliche Seite öffnet — bewusst keine Kopie, damit niemand eine veraltete Warnung liest. Die KI hat vorher gemessen, was das BSH anbietet: die Warnnachrichten des Seewarndienstes Emden als laufend aktualisierte Datei je Seegebiet, das wöchentliche Heft als PDF, beides kostenlos unter festen Adressen. Ich habe die zweite Fassung gewählt, wie beim Seewetterbericht: eine Kopie auf dem Gerät, aber mit sichtbarem Alter. Am Ankerplatz ohne Netz ist die Warnnachricht von Freitag mehr wert als ein Knopf, der nicht lädt — solange dabei steht, von wann sie ist.

Beim Auslesen fiel auf, dass der Knopf der Webfassung seit unbekannt wann ins Leere führt: Die Adresse stand in Kleinschreibung, und das BSH antwortet darauf mit 404. Das war der erste Webpunkt für den Abend.

Was am Schreibtisch noch ging

Zwischendurch die kleinen Sachen, die noch auf einer Liste standen. Die Wiedergabe einer Aufzeichnung löst den Alarm aus, schreibt aber nichts ins Logbuch — so soll es sein. Die Punkte der Ankerwache, die vor einer Woche mit einer fehlerhaften Zeitquelle geprüft worden waren, sind mit der richtigen nachgefahren. Und mit dem Mac als Gateway ließ sich prüfen, was bisher aufs Boot gewartet hatte: Startet eine Wiedergabe, wird die echte Verbindung getrennt. Nur kam sie danach nicht von selbst zurück — die Bootsprüfliste versprach das seit zwei Wochen, und der Code tat es nicht. Eine Zeile.

Beim Prüfen dieser Zeile lag der Fehler dann bei der KI: Ihr Ersatz für das Gateway nahm nach dem Trennen keine neue Verbindung mehr an, und die App klopfte einunddreißigmal an, ohne dass jemand hörte. Die Prüfliste hatte behauptet, der Ersatz könne das. Dieselbe Sorte Fehler wie gestern bei den elf Zeilen: etwas versprochen, was nicht durchgespielt war. Mit einem richtigen Prüfgateway ging es dann.

Und eine Frage, die mir wegen Freitag kam: Wie bekomme ich alle Daten aus der App wieder heraus? Der Törn soll mit leerem Bordbuch beginnen, nicht mit den Prüfeinträgen der letzten Wochen. Jetzt gibt es unter System zwei Stufen — nur das Bordbuch, oder alles. Was geht und was bleibt, steht vor dem Knopf, gezählt und nicht geschätzt; die Stammdaten bleiben, die Sicherungen sowieso.

Der Assistent

Der letzte Bereich. In der Webfassung ist der Assistent bewusst kein Sprachmodell: eine Wortsuche über die Stellen der App und die Artikel auf pelaron.de, und wo nichts passt, sagt er das. Die KI hat das Verfahren Regel für Regel nachgebaut und die Erwartungswerte aus dem Web nachgerechnet, mit einer Nachbildung der Rechnung, bevor eine Zeile Swift stand.

Aber das ist nicht, was ich am Ende will. Dies soll eine Art KI sein, die dem Nutzer bei Fragen zur Verfügung steht, auch mit Wissen aus dem Netz. Ich könnte selbst, unbewusst, Fehler in der Wissensdatenbank angelegt haben — und dann vertraue ich doch der Schwarmintelligenz des Internets. Die KI der App soll intelligent und allumfassend sein. Ein Modell auf dem Gerät allein, wie Apple es seit iPadOS 26 anbietet, hätte das Netz nicht; als Stufe ohne Netz darf es dabei sein, mit einem Hinweis auf Geräten, die es nicht können — und den Grund dafür meldet das System selbst. Sprachein- und -ausgabe will ich nicht. Der Anbieter ist Claude, über einen Dienst auf pelaron.de, damit kein Schlüssel in der App liegt.

Weil die App nicht nur für mich ist, sondern verkauft werden soll, ging es dann um Geld. Eine Frage mit Netzsuche kostet nach Preisliste etwa drei Cent. Die KI hat mir fünf Preismodelle nebeneinandergelegt, vom Kontingent im Kaufpreis bis zum eigenen Schlüssel des Käufers. Ich habe den Vorschlag angenommen: ein kleines Freikontingent, darüber ein Abo — und als ersten Schritt nur das Kontingent, damit wir sehen, wie viel wirklich gefragt wird, bevor Preise feststehen. Gebaut ist davon nichts; die Wortsuche ist der Boden, auf dem es steht.

Fünf Fehler, die kein Prüfer sah

Der erste Lauf des Assistenten brach ab. Zwei Zeichenketten in den Prüffällen waren mit dem falschen Anführungszeichen geschlossen, und dreimal hieß eine Konstante wie die Funktion, die sie füllt — das liest sich natürlich und übersetzt nicht. Beide Prüfer hatten „keine Befunde" gesagt. Der App-Prüfer kannte die Regel für die Anführungszeichen seit dem 30. August, der Kern-Prüfer nicht, und die Prüffälle sind voller deutscher Sätze. Jetzt kennen beide beide Regeln, und auf den alten Dateien finden sie genau die fünf Stellen und sonst nichts. Ich baue nicht, bis die Fehler weg sind; das habe ich so gesagt, und das galt auch heute.

Der falsche Ordner

Abends die Webfassung. Der BSH-Link war eine Zeile und zwei Buchstaben. Ich habe die beiden Dateien hochgeladen, gemessen — und der Server zeigte den alten Stand. Noch einmal gemessen, wieder alt. Dann sah ich es: Ich hatte den falschen Ordner gewählt. Die Startseite der App lag im Ordner von pelaron.de, und wer in dieser Viertelstunde die Website aufrief, bekam die App.

Ich war einfach zu schnell und zu unkonzentriert. Die Wiederherstellung bei IONOS hatte ich schon einmal benutzt, und nach einer kurzen Frage an die IONOS-KI war das kein Problem: die Startseite aus der Sicherung von gestern Abend zurück, die fremde Datei gelöscht. Die KI hatte derweil nach einer Kopie gesucht und keine gefunden — und mir dann einen Sollwert aus dem Heft vom 27. August genannt, der nicht mehr stimmte. Die Sicherung war der letzte Stand; die Aufzeichnung im Heft war es nicht. Das ist die Regel, die sie selbst seit dem 22. August aufschreibt.

Ihr eigener Anteil an dem Abend: Die Lieferung hatte keine LIESMICH mit der Zeile, wohin sie gehört. Diese Zeile gibt es seit dem Fehlupload vom 27. August genau für diesen Fall. Seit heute Abend steht sie wieder auf jeder Lieferung, als erste Zeile.

Die Webfassung zieht nach

Danach ging es durch: Der Assistent der Webfassung kennt jetzt die zwei Artikel, die seit dem 30. August auf pelaron.de stehen, und zehn Füllwörter aus der nativen App — vorher fand „Wie funktioniert die Zwölferregel?" zuerst den Seenotruf, weil „funktioniert" in dessen Titel steht. „Ankern" im Logbuch schreibt nichts mehr, sondern führt zur Ankerwache, und „Anker auf" beendet sie — das, was ich gestern für die native App entschieden habe, gilt jetzt auch im Browser. Drei Dialoge, die ohne Innenabstand an der Kante klebten, haben ihn. Und die Wetterangaben stehen an den Logzeilen, der letzte offene Schritt einer Liste vom 3. September.

Was bleibt

Spät am Abend war das iPad wieder frei, und die dreizehn Schritte des Assistenten gingen durch: die Beispiele, das Springen in den Bereich, der Verlauf, der nach dem Neustart noch da ist, die Frage, zu der er nichts weiß und das auch sagt. Einundzwanzig von einundzwanzig — gebaut und geprüft. Jeder Bereich der Webfassung hat jetzt sein Gegenstück auf dem iPad; was noch fehlt, ist das, was die Webfassung selbst nicht hat, und das ist für später.

Vor Freitag wird das Bordbuch geleert, und der Törn beginnt mit einem leeren Logbuch.

Morgen Nachmittag fahre ich wieder aufs Boot. Ich bin tierisch gespannt, ob das Empfangen der Daten aus dem Netzwerk gelingt und reibungslos verläuft — oder ob noch Umbauten an der App nötig sind.