

SF2 SUITE RT 1.1.90

Waterpump End-to-End-Tutorial

Von Rohsensordaten zur lokal in Betrieb genommenen Engine

Ein geführter Einstieg in Semantic Folding Sensor Fusion

ROHDATEN • UDTAT • UDM • UFP • STUDIO • ENGINEERT • SCADA

BENUTZERTUTORIAL

Dieses Tutorial ist der empfohlene erste vollständige Einstieg in die SF2-Technologie. Sie begleiten ein Waterpump-Projekt von einer CSV-Aufzeichnung über Datenvorbereitung, Training der semantischen Map, Fingerprint-Erzeugung, Erfassung von Fehlerreferenzen und Alarmkonfiguration bis zum signierten Runtime-Export, zur automatischen Erkennung einer lokalen Engine und zu einem realen SCADA-Alarm.

Der zentrale Gedanke ist die durchgängige Konsistenz. Ein SF2-Projekt besteht aus einer Kette versionierter Artefakte und nicht aus einer Sammlung voneinander unabhängiger Einstellungen:

sensor_in.csv -> UDTAT -> UDM -> UFP -> Studio Status Library -> signiertes Engine-Paket -> lokales EngineRT -> SCADA

Fahren Sie erst fort, wenn das im aktuellen Schritt erzeugte Artefakt vorhanden ist, sich erfolgreich öffnen lässt und zum selben vorgelagerten Projektstand gehört.

Bevor Sie beginnen

Planen Sie etwa 60 bis 90 Minuten ein, wenn Sie die Trainingsschritte selbst ausführen. Sie benötigen SF2 Suite RT, Schreibzugriff auf den ausgewählten Ordner `SF2-Suite-Data` und das mitgelieferte Waterpump-Projekt unter `SF2-Suite-Data/waterpump/project.sf2project.json`. Für die lokale Runtime-Demonstration werden außerdem der File Stream Server und der SCADA Simulator der Suite sowie ein lizenzierter lokaler Engine-Slot verwendet.

Die Screenshots wurden auf Apple Silicon erstellt. Dateiauswahldialoge und Bezeichnungen für Beschleuniger können auf anderen Plattformen abweichen; Dateinamen und Artefaktverträge bleiben jedoch gleich. Engine-Pakete für den Produktivbetrieb müssen mit einem vertrauenswürdigen Ed25519-Schlüssel signiert sein. Nicht signierte Pakete eignen sich nur für ausdrücklich dafür konfigurierte Entwicklungsumgebungen.

Zuständigkeit für Artefakte

Artefakt	Zuständig	Zweck
<code>waterpump_norm_binning.json</code>	UDTAT	Spaltenrollen, Normalisierung und Binning
<code>waterpump_numeric_data.csv</code>	UDTAT	Kontinuierlicher, normalisierter UDM-Input
<code>waterpump_binned_data.csv</code>	UDTAT	Diskreter UFP-Input
<code>waterpump_F_r64-0.02_e100_g64x64_MAP.jsonl</code>	UDM	Trainierte semantische Map
<code>waterpump_retina_64.jsonl</code>	UFP	Runtime-Fingerprint-Dictionary

Artefakt	Zuständig	Zweck
status_library.statuslib.json	Studio	Referenz-Fingerprints und Trigger-Regeln
runtime.sf2enginepkg	Studio	Signiertes, portables EngineRT-Paket

1. SF2 Suite RT starten

Beim Start fragt Studio, wo die Projektdaten gespeichert werden sollen. Wählen Sie den Ordner, der das Verzeichnis `SF2-Suite-Data` enthält. Mit der unten gezeigten Auswahl wird das mitgelieferte Waterpump-Projekt gefunden, ohne seine internen Pfade zu verändern.

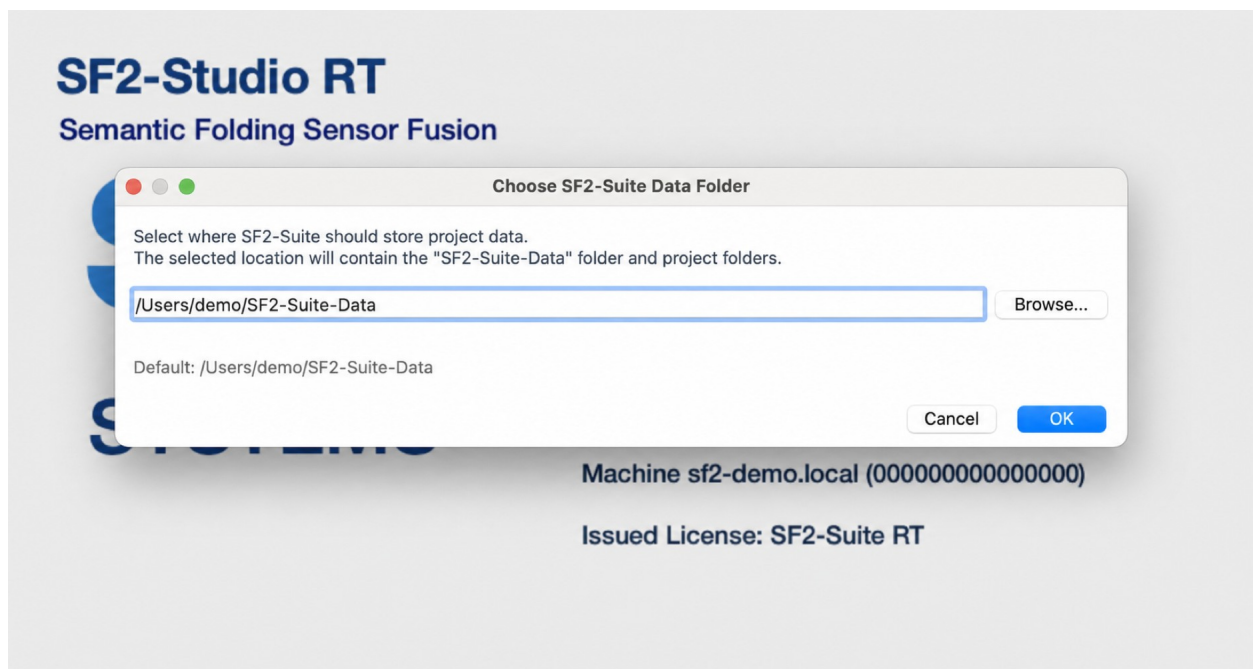


Abbildung 1. Startbildschirm und Auswahl des Datenstammordners sind die ersten sichtbaren Schritte des Tutorials.

Wählen Sie **OK**. Studio öffnet sich, ohne dass Sie zuvor ein leeres Projekt anlegen müssen.

Prüfpunkt: Der ausgewählte Datenstammordner ist beschreibbar und enthält einen Projektordner namens `waterpump`.

2. Das mitgelieferte Waterpump-Projekt öffnen

Wählen Sie **Open Project**, navigieren Sie zum Ordner `waterpump` und öffnen Sie `project.sf2project.json`.

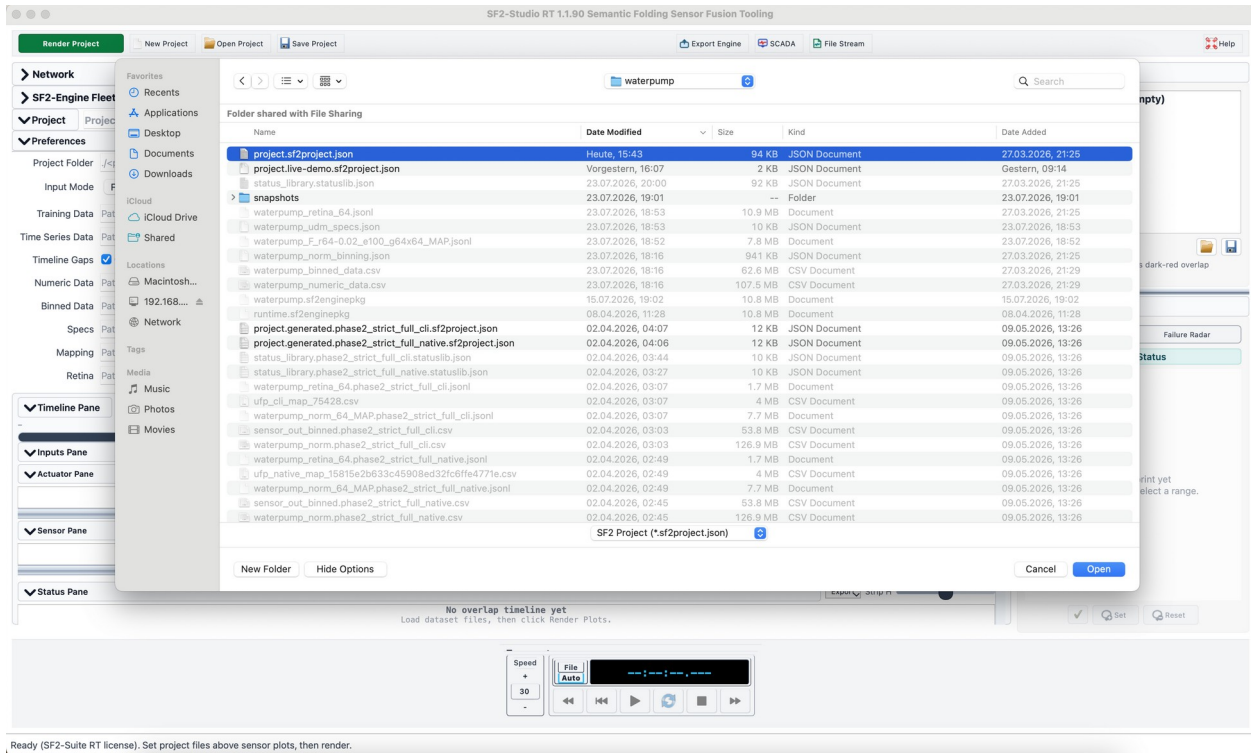


Abbildung 2. Öffnen Sie das mitgelieferte Projekt, anstatt seine Artefaktpfade manuell zusammenzustellen.

Klappen Sie **Project** und **Preferences** auf. Prüfen Sie die vollständige Artefaktkette:

Feld	Erwarteter Wert
Input Mode	File CSV
Training Data	sensor_in.csv
Time Series Data	sensor_in.csv
Specs	waterpump_norm_binning.json
Numeric Data	waterpump_numeric_data.csv
Binned Data	waterpump_binned_data.csv
Mapping	waterpump_F_r64-0.02_e100_g64x64_MAP.jsonl
Retina	waterpump_retina_64.jsonl

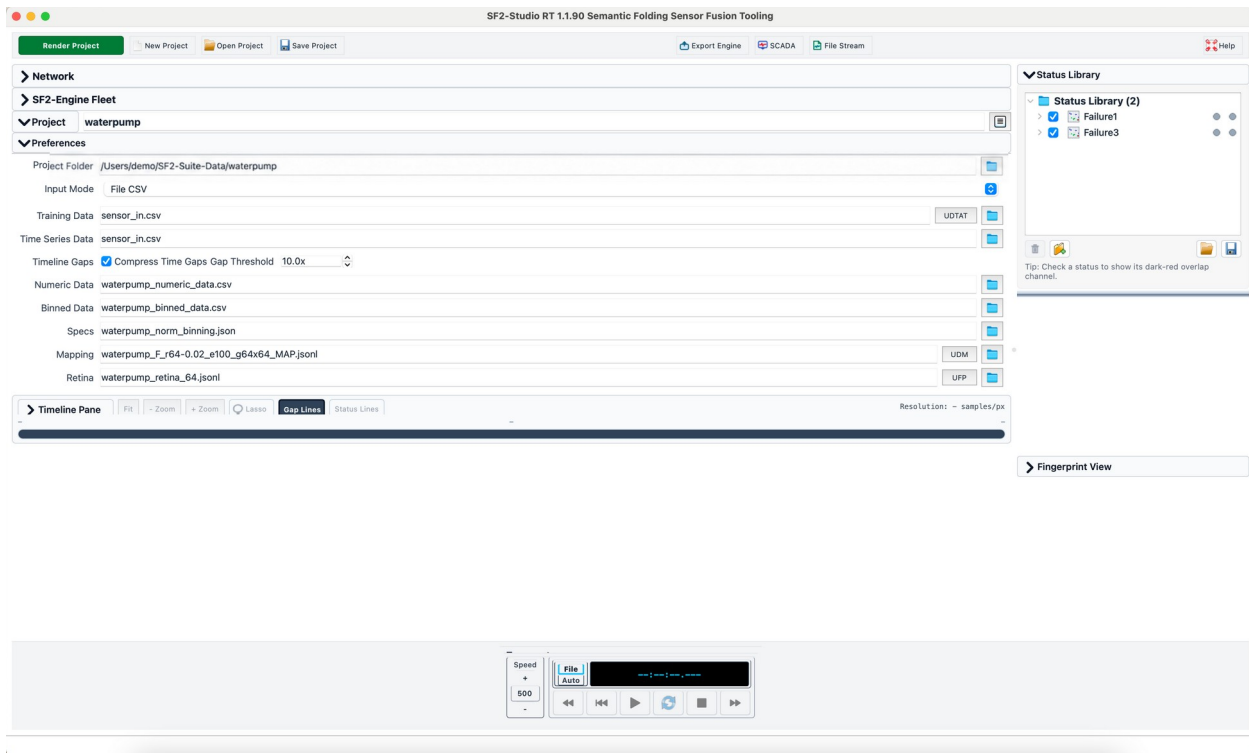


Abbildung 3. Das geladene Projekt verbindet Quelle, Rezept, numerische und gebinnte Daten, Map, Retina und Status Library.

Die Aufzeichnung enthält `id`, `timestamp`, `status` sowie `sensor_00` bis `sensor_51`, insgesamt also 52 rohe Sensorspalten. Das mitgelieferte Rezept entfernt den nicht verwendeten `sensor_15`; das trainierte Modell enthält daher 51 Sensoren. Für ein vollständig reproduzierbares Tutorial wird dieselbe Aufzeichnung bewusst für Training und Wiedergabe verwendet. Im Produktivbetrieb werden Trainings- und Validierungs- beziehungsweise Wiedergabedaten normalerweise getrennt.

Prüfpunkt: Kein Artefaktfeld ist leer, das Projekt heißt `waterpump` und die mitgelieferten Referenzen `Failure1` und `Failure3` sind vorhanden.

3. Das fertig konfigurierte Projekt verstehen

Bevor Sie das Projekt selbst nachbauen, sollten Sie seinen fertigen Zustand betrachten. Die folgende Übersicht zeigt das Ziel der Kapitel 4 bis 9: Das Waterpump-Projekt ist geladen, historische Sensor- und Statusdaten sind sichtbar, Referenz-Fingerprints stehen zur Verfügung und ihre Ähnlichkeitskurven sind auf derselben Zeitachse ausgerichtet.



Abbildung 4. Ein konfiguriertes Waterpump-Projekt. Die nummerierte Legende verbindet jedes Bedienelement mit seiner Aufgabe im Workflow.

Nr.	Oberflächenelement	Bedeutung in diesem Tutorial
1	Render Project	Lädt die konfigurierten Artefakte und rendert historische Timeline, Statuskanäle, Fingerprints und Overlap-Kurven.
2	Help	Öffnet die kontextbezogene Hilfe für die aktive Version von SF2 Studio.
3	Export Engine / SCADA / File Stream	Öffnet Deployment- und Simulationswerkzeuge. In der SE-Edition sind diese Schaltflächen inaktiv; die exportierten Artefakte gehören dennoch zum selben Workflow.
4	Network Pane	Konfiguriert Discovery und Alarm-Uplinks wie MQTT oder REST.
5	SF2-Engine Fleet Pane	Erkennt, übernimmt und überwacht EngineRT-Instanzen.
6	Project and Preferences Panes	Identifizieren das Projekt und zeigen Quelle, Preprocessing-Rezept, numerische Daten, gebinnte Daten, Map, Retina und die Schaltflächen zum Starten der Werkzeuge.

Nr.	Oberflächenelement	Bedeutung in diesem Tutorial
7	Timeline Pane	Stellt jeden ausgewählten Sensor auf einer gemeinsamen Zeitachse dar.
8	Status Pane	Zeigt aufgezeichnete Betriebszustände einschließlich der High-Phase, die für Failure1 verwendet wird.
9	Overlap Pane	Zeigt für jeden sichtbaren Library-Fingerprint einen Ähnlichkeitskanal sowie den Baseline-Kanal.
10	Cursor position	Die vertikale Linie wählt genau einen Datensatz aus und synchronisiert Sensoren, Status, Overlap und den aktuellen Fingerprint.
11	Tape control	Navigiert durch Daten im Dateimodus beziehungsweise spielt sie ab und zeigt den aktuellen Zeitstempel der Aufzeichnung.
12	Timeline controls	Fit zeigt die gesamte Aufzeichnung; Zoom verändert die Detailstufe; Lasso wählt einen Bereich; Gap Lines und Status Lines ergänzen Kontextmarkierungen.
13	Resolution	Gibt an, wie viele Samples durch einen horizontalen Pixel repräsentiert werden. Kleinere Werte zeigen mehr zeitliche Details.
14	Status Library	Speichert benannte, wiederverwendbare Referenzzustände und deren Alarmeinstellungen.
15	Failure fingerprints	Failure1 und Failure3 sind benannte numerische Referenzvektoren, die aus überprüften Zeitpunkten der Aufzeichnung erfasst wurden.
16	Load / save library	Öffnet oder speichert die vollständige Status Library unabhängig von der Projektdatei.
17	Fingerprint View	Vergleicht den aktuellen oder ausgewählten Fingerprint mit einer Library-Referenz.
18	Current + Status fingerprint	Blau beziehungsweise Türkis gehört zur aktuellen Auswahl, Gelb zur ausgewählten Referenz und Rot zu beiden.
19	Green check mark	Übernimmt die aktuelle Cursor- oder Lasso-Auswahl als neuen Library-Fingerprint.
20	Verknüpfung von `Failure1`	Der Library-Eintrag benennt die Referenz, der Timeline-Marker zeigt ihren Erfassungsort und der Overlap-Strip Failure1 zeigt für jeden Zeitpunkt die Ähnlichkeit mit ihr.

Die wiederholte Nummer 20 kennzeichnet den wichtigsten Zusammenhang. Ein Eintrag in der Status Library ist nicht bloß eine Bezeichnung: Er ist ein benannter Fingerprint-Vektor mit einem Ursprung in der Timeline und einem zugehörigen Overlap-Signal über die gesamte Aufzeichnung.

Prüfpunkt: Sie können Artefaktkette, Timeline-Bedienelemente, Status Library, Fingerprint-Vergleich und die drei miteinander verbundenen Darstellungen von `Failure1` lokalisieren.

4. Daten in UDTAT vorbereiten

UDTAT wandelt die Roh-tabelle in zwei synchronisierte Modelleingaben um. Öffnen Sie UDTAT immer aus dem Projektkontext: Starten Sie **SF2 Suite RT**, öffnen Sie das Waterpump-Projekt, klappen Sie **Preferences** auf und wählen Sie **UDTAT** neben dem Feld für die Trainingsdaten. Dadurch bleiben der aktive Projektkontext und seine konfigurierten Pfade erhalten.

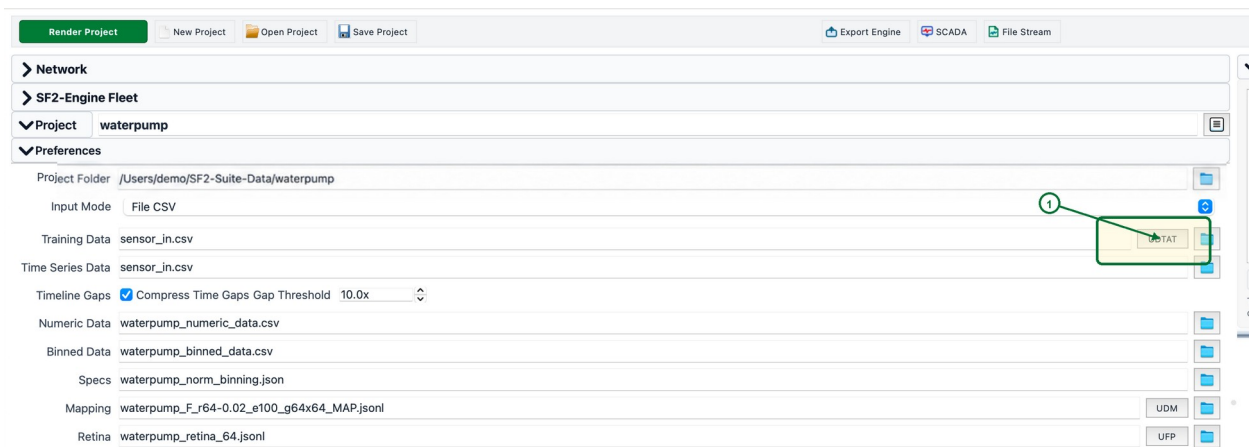


Abbildung 5. UDTAT wird aus dem geöffneten Projekt in SF2 Suite RT gestartet und nicht als losgelöster erster Arbeitsschritt.

4.1 Den Vertrag der Quelldaten prüfen

Öffnen Sie in UDTAT **Original** und prüfen Sie, dass 220.320 Zeilen vorhanden sind. Weisen Sie `id` und `timestamp` die Rolle **Ignore**, `status` die Rolle **Status** und den Sensorspalten die Rolle **Sensor** zu. Der Status ist übergeordnete Kontextinformation: Er muss für die spätere Interpretation verfügbar bleiben, darf aber nicht in die Trainingsmerkmale der Map einfließen.

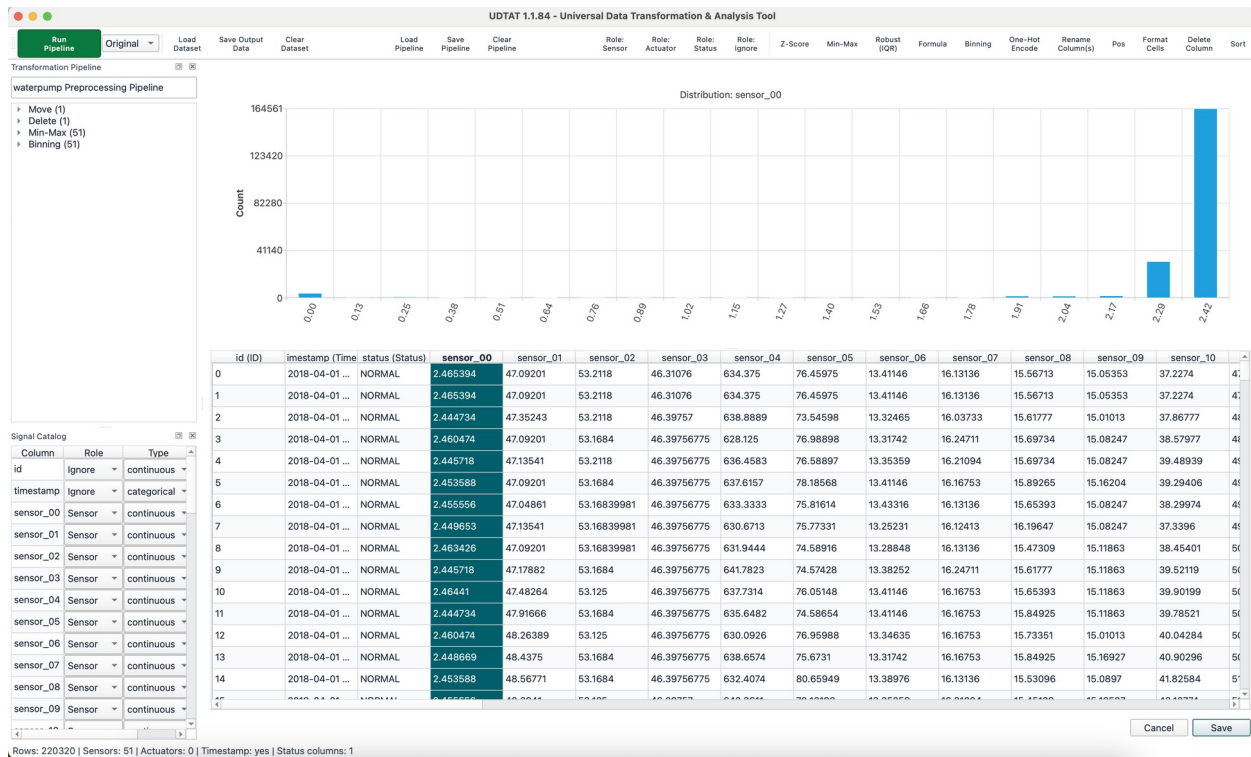


Abbildung 6. Quelltable und Signal Catalog legen fest, welche Spalten Merkmale, Metadaten und aufgezeichnete Zustände enthalten.

Die mitgelieferte Pipeline verschiebt die Statusspalte an die vorgesehene Ausgabeposition, entfernt den nicht verwendeten `sensor_15`, normalisiert die 51 verbleibenden Sensoren mit Min-Max und weist jedem normalisierten Wert ein semantisches Bin-Label zu. Wählen Sie **Run Pipeline** und wechseln Sie anschließend zwischen **Numeric** und **Binned**, um beide Ausgaben zu prüfen.

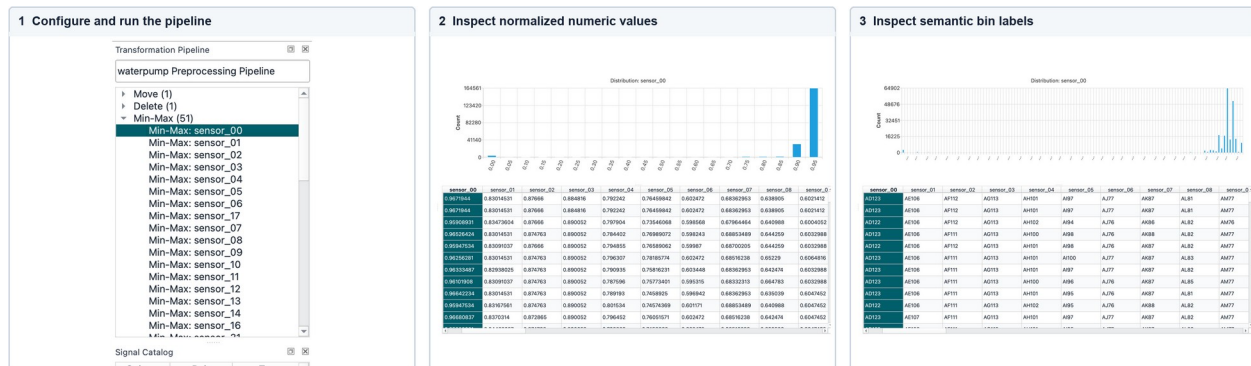


Abbildung 7. Drei fokussierte Zustände ersetzen wiederholte Vollbildansichten: Pipeline konfigurieren und ausführen, normalisierte Werte prüfen und anschließend semantische Bin-Labels kontrollieren.

Numerische und gebinnete Daten werden vom selben Rezept erzeugt und gehören zusammen. Die numerische Tabelle ist der kontinuierliche UDM-Input, die gebinnete Tabelle der symbolische UFP-Input. Ihre Zeilen müssen ausgerichtet bleiben, da beide dieselben physischen Datensätze beschreiben.

4.2 Die Pipeline prüfen und bearbeiten

Klappen Sie eine Operationsgruppe auf, um ihre einzelnen Schritte zu prüfen. Mit einem Doppelklick auf einen Eintrag öffnen Sie **Edit Pipeline Step**. Dort können Operation und Zielspalte kontrolliert oder geändert werden.

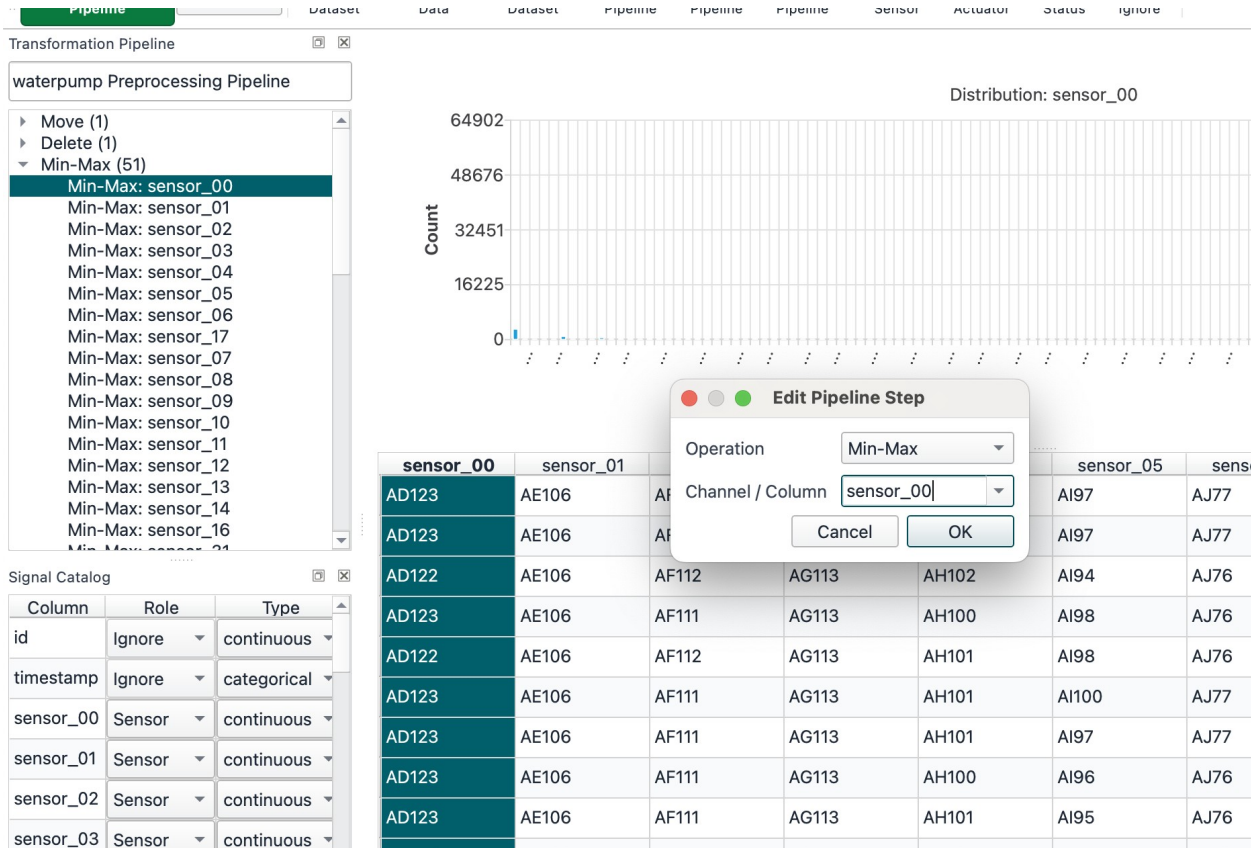


Abbildung 8. Ein Doppelklick auf einen Pipeline-Eintrag öffnet den Schritteditor. Änderungen sollten bewusst vorgenommen und anschließend mit einem vollständigen Pipeline-Lauf überprüft werden.

Speichern Sie das Rezept als `waterpump_norm_binning.json`. Diese Datei ist ein Runtime-Vertrag und keine entbehrliche Einrichtungsinformation. UDM, UFP, Studio-Export und EngineRT sind auf exakt dieselben Rollen, Normalisierungsgrenzen und Binning-Regeln angewiesen. Eine Map oder Retina, die mit einem anderen Rezept erzeugt wurde, lässt sich möglicherweise noch öffnen, beschreibt jedoch nicht mehr denselben numerischen Raum und kann nicht für eine korrekte Verarbeitung verwendet werden.

4.3 Das Verteilungsdiagramm lesen

Wenn Sie einen Pipeline-Eintrag oder eine Sensorspalte auswählen, wird deren Werteverteilung angezeigt. Jeder Balken repräsentiert ein Werteintervall; seine Höhe entspricht der Anzahl der Datensätze in diesem Intervall. Ein schmaler, hoher Cluster weist auf ein vergleichsweise stabiles Betriebsband hin. Ein kleiner, abgesetzter Cluster kann einen alternativen Betriebszustand, einen Stillstand, einen Signalausfall oder eine fehlerbezogene Population kennzeichnen. Eine breite Verteilung

bedeutet, dass der Sensor einen größeren Teil seines Wertebereichs durchläuft und daher meist mehr Bins belegt.

UDTAT distribution comparison on the original Waterpump sensor data

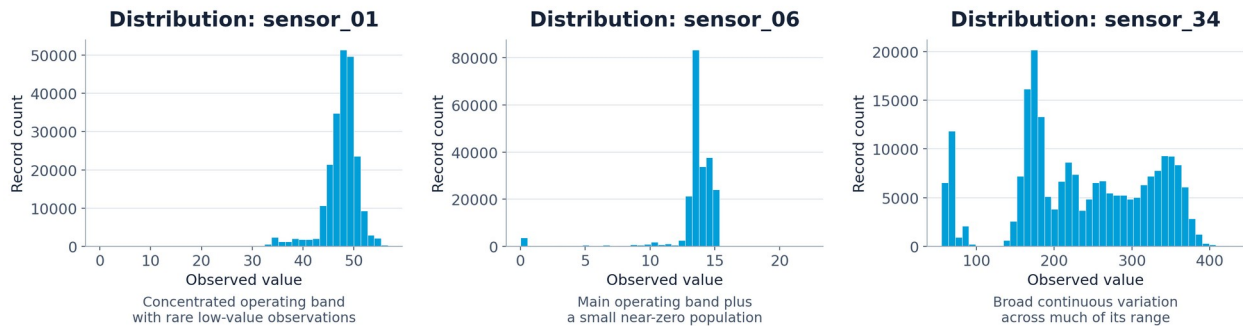


Abbildung 9. **sensor_01** besitzt ein konzentriertes Betriebsband mit seltenen niedrigen Werten; **sensor_06** zeigt zusätzlich eine kleine Population nahe null; **sensor_34** variiert breit über seinen Wertebereich.

Verwenden Sie das Diagramm als Prüfung der Datenqualität und Konfiguration. Unerwartete Spitzen, leere Wertebereiche oder eine Verteilung, die nicht zum Quellsignal passt, sollten vor dem Training untersucht werden. Das Diagramm beschreibt Häufigkeit und nicht Wichtigkeit: Ein seltener Wert kann dennoch der wertvollste Hinweis auf einen benannten Zustand sein.

Prüfpunkt: UDTAT erzeugt `waterpump_numeric_data.csv` und `waterpump_binned_data.csv` mit 220.320 ausgerichteten Zeilen, behält 51 modellierte Sensoren und speichert den Preprocessing-Vertrag, der von allen nachfolgenden Schritten verwendet wird.

5. Die semantische Map in UDM trainieren

UDM verteilt die Trainingsdatensätze über eine zweidimensionale semantische Map. Datensätze mit ähnlichen Kontexten aus 51 Sensoren werden nahe beieinander angeordnet. Benachbarte Map-Positionen repräsentieren daher verwandte Betriebssituationen.

Öffnen Sie ausschließlich das **Parameters Pane** und verwenden Sie folgende reproduzierbare Waterpump-Baseline:

Parameter	Wert
Grid Size	64
Map Topology	Flat Map
Initialization	Random Distribution
Epochs	100
Initial Radius	64.00

Parameter	Wert
Final Radius	1.00
Radius Decay Strategy	Exponential (fast start, slow end)

Wählen Sie im Performance-Dropdown eine verfügbare Trainingsimplementierung. Die genauen Bezeichnungen hängen vom Betriebssystem und von der CPU-/GPU-Kombination ab. Verwenden Sie ein unterstütztes beschleunigtes Backend, wenn eines angeboten wird, andernfalls die kompatible CPU-Implementierung.

UDM 1.1.90 - Universal Data Modeler

Start Training | New Map | Open Map | Save Map | New Specs | Load Specs | Save Specs

Parameters Panel

Input Data File: waterpump_numeric_data.csv [Browse...]

Algorithm: Batch - Metal (GPU Accelerated) [v]

Grid Size: 64 [v]

Map Topology: Flat Map [v]

Initialization: Random Distribution [v]

Epochs: 100 [v]

Initial Radius: 64.00 [v]

Final Radius: 1.00 [v]

Radius Decay Strategy: Exponential (fast start, slow end) [v]

Map File Name: waterpump_F_r64-0.02_e100_g64x64_MAP.json [Reset Map Name]

Signal Catalog Panel

Map	Signal	Role
☐	id	id
☐	timestamp	timestamp
☒	sensor_00	sensor
☒	sensor_01	sensor
☒	sensor_02	sensor

Input Data Panel

	id	timestamp	sensor_00	sensor_01	sensor_02	sensor_03	sensor_04	sensor_05	sensor_06	sensor_07	sensor_08	sensor_09	sensor_10	sensor_11	sensor_12	sensor_13	sensor_14
1	0.000000	2018-04-01 ...	0.967194	0.830145	0.876660	0.884816	0.792242	0.764598	0.602472	0.683630	0.638905	0.602141	0.489146	0.792070	0.691492	0.053911	0.8280
2	1.000000	2018-04-01 ...	0.967194	0.830145	0.876660	0.884816	0.792242	0.764598	0.602472	0.683630	0.638905	0.602141	0.489146	0.792070	0.691492	0.053911	0.8280
3	2.000000	2018-04-01 ...	0.959089	0.834736	0.876660	0.890052	0.797904	0.735461	0.598568	0.679645	0.640988	0.600405	0.497561	0.802954	0.713088	0.054781	0.8307
4	3.000000	2018-04-01 ...	0.965264	0.830145	0.874763	0.890052	0.784402	0.769891	0.598243	0.688535	0.644259	0.603299	0.506916	0.810935	0.703827	0.050643	0.8305
5	4.000000	2018-04-01 ...	0.959475	0.830910	0.876660	0.890052	0.794855	0.765891	0.598970	0.687002	0.644259	0.603299	0.518868	0.817716	0.710045	0.053990	0.8286

Heatmap Panel

Density: 0 to 359

Legend: Non-Empty Nodes

Loaded 220320 records, 51 source features, 51 map signals, 51 encoded features (numeric).

Cancel Save

Abbildung 10. Die Baseline definiert eine reproduzierbare 64-x-64-Map; die Performance-Auswahl zeigt die vom aktuellen System unterstützten Beschleuniger.

Schließen Sie **Parameters** und öffnen Sie das **Input Data Pane**. Prüfen Sie 220.320 Datensätze sowie jeweils 51 Source-, Map- und Encoded-Features. `id`, `timestamp` und `status` dürfen keine Trainingsmerkmale sein.

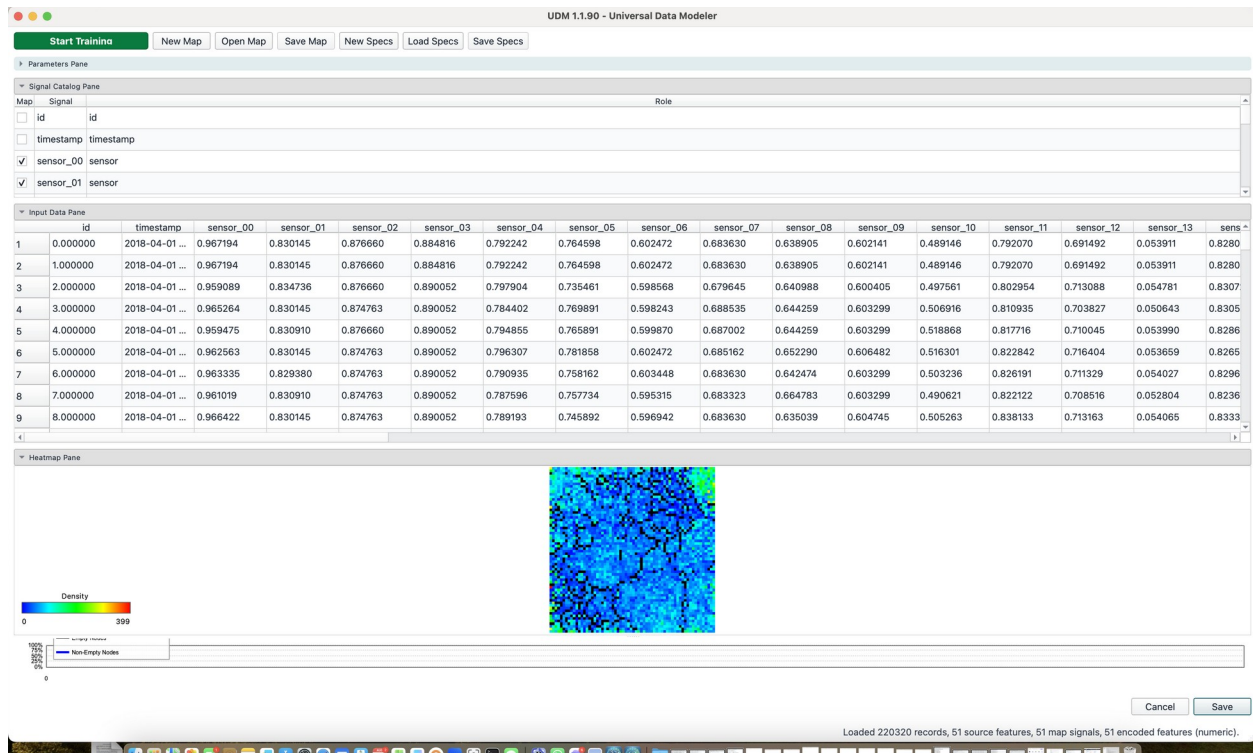


Abbildung 11. UDM liest die normalisierte numerische Ausgabe, die mit demselben UDTAT-Rezept erzeugt wurde.

Wählen Sie **Start Training**. Schließen Sie nach Abschluss des Trainings die sekundären Panes und geben Sie dem **Heatmap Pane** die Hauptfläche.

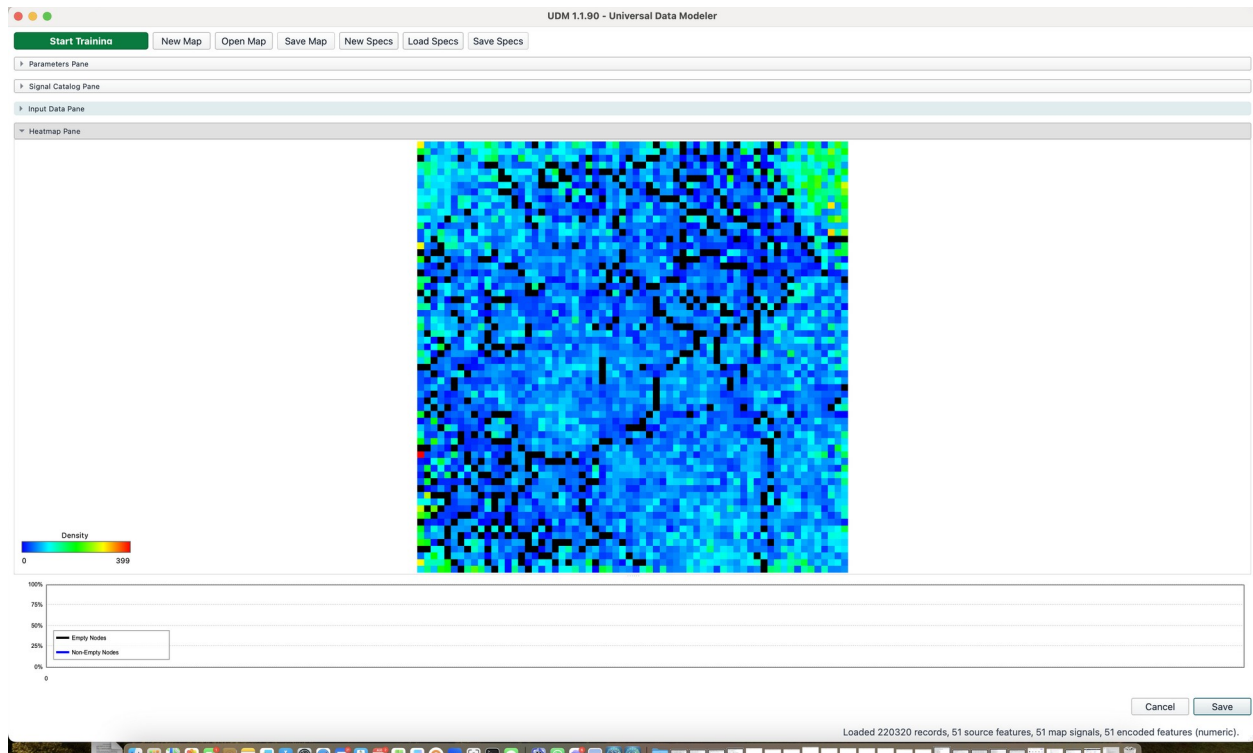


Abbildung 12. Die Heatmap ist das zentrale UDM-Ergebnis: Farbe und Belegung zeigen, wie die Trainingsdatensätze die semantische Map nutzen.

Interpretieren Sie das Ergebnis anhand von vier Aspekten:

1. **Räumliche Gruppierung:** Ähnliche Trainingsdatensätze sollten benachbarte Regionen bilden und nicht strukturlos verstreut sein.
2. **Nutzung des Map-Space:** Versuchen Sie, unbelegte Positionen zu minimieren oder ganz zu vermeiden, damit der verfügbare 64-x-64-Raum die Daten effizient repräsentiert. Große leere Bereiche weisen auf ungenutzte Kapazität oder ein Trainings- beziehungsweise Konfigurationsproblem hin.
3. **Verteilungsgraph und Legende:** Der Graph fasst zusammen, wie gleichmäßig die Datensätze zugeordnet sind. Die Legende ordnet den Farben die Anzahl der Datensätze pro Position zu.
4. **Prüfung per Hover:** Bewegen Sie den Mauszeiger über eine berechnete Map-Zelle, um Position und Anzahl der dort liegenden Datensätze abzulesen. Prüfen Sie sowohl dicht als auch schwach belegte Bereiche, anstatt nur die am stärksten gesättigte Farbe zu beurteilen.

Ein abgeschlossener Epochenzähler belegt lediglich, dass die Berechnung beendet wurde. Ein brauchbares Ergebnis verteilt die Datensätze über die Map und erhält dabei lokale Ähnlichkeit. Eine leere Map, eine einzelne übermäßig gesättigte Zelle, großflächig ungenutzter Raum oder eine abweichende Merkmalsanzahl müssen den Workflow stoppen.

Speichern Sie die Map als `waterpump_F_r64-0.02_e100_g64x64_MAP.jsonl`.

Prüfpunkt: Die 64-x-64-Heatmap ist über die Map hinweg belegt, UDM meldet 220.320 Datensätze und 51 modellierte Signale und die Hover-Prüfung liefert plausible Datensatzanzahlen.

6. Das Fingerprint-Dictionary in UFP erzeugen

UFP kombiniert die gebinnten Waterpump-Werte mit der trainierten semantischen Map. Für jedes Bin-Label jedes modellierten Sensors wird ein 64-x-64-Fingerprint erzeugt. Die Gesamtheit dieser Bin-Fingerprints bildet das Retina-Dictionary, mit dem Studio und EngineRT neue Datensätze kodieren.

Öffnen Sie **Data Sources** und prüfen Sie, dass alle drei Eingaben aus derselben Artefaktkette stammen:

Eingabe	Erwartete Datei
Binned CSV	waterpump_binned_data.csv
UDM Map File	waterpump_F_r64-0.02_e100_g64x64_MAP.jsonl
UDTAT Recipe	waterpump_norm_binning.json

The screenshot displays the 'Data Sources' and 'Fingerprint Configuration' sections of the SF2 Studio interface. The 'Data Sources' section contains four input fields, each with a 'Browse...' button: 'Binned CSV' (waterpump_binned_data.csv), 'UDM Map File' (waterpump_F_r64-0.02_e100_g64x64_MAP.jsonl), 'UDTAT Recipe' (waterpump_norm_binning.json), and 'Retina File Name' (waterpump_retina_64.jsonl). The 'Fingerprint Configuration' section shows 'Sensor: sensor_00' and 'Bin Index' set to a dropdown. Below this is a 64x64 heatmap visualization with blue dots representing active positions. At the bottom, it says 'Bin Label: | Active Positions: 136 | Total Counts: 10208' and 'Loaded existing retina dictionary: waterpump_retina_64.jsonl'. 'Cancel' and 'Save' buttons are at the bottom right.

Abbildung 13. Gebinnte Daten, Map und Preprocessing-Rezept müssen zur selben Artefaktkette gehören.

Behalten Sie im **Signal Catalog** die mitgelieferte Auswahl von 51 Sensoren bei. Lassen Sie in **Fingerprint Configuration** das quadratische Raster auf **64 px**, passend zur UDM-Map. Erzeugen und speichern Sie `waterpump_retina_64.jsonl`.

The screenshot displays the SF2 Suite RT interface. At the top, there are tabs for 'Data Sources', 'Signal Catalog', and 'Fingerprint Configuration'. The 'Fingerprint Configuration' tab is active, showing a 'Square Grid Size' of '64 px' and a 'User Notes' field. Below these is a green 'Generate Retina' button. The bottom section, titled '3. Fingerprint Inspection', shows a 'Sensor' dropdown set to 'sensor_00' and a 'Bin Index' dropdown. A large grid displays a heatmap of active positions (blue dots) on a grid. Below the grid, it states 'Bin Label: | Active Positions: 136 | Total Counts: 10208' and 'Loaded existing retina dictionary: waterpump_retina_64.jsonl'. At the bottom right are 'Cancel' and 'Save' buttons.

Abbildung 14. Mit der Erzeugung ist das zählwertbasierte Fingerprint-Dictionary für die Runtime vollständig.

6.1 Fingerprint-Dichte und Zählwerte verstehen

Geben Sie **Fingerprint Inspection** die gesamte Arbeitsfläche und prüfen Sie Bins aus mehreren Sensoren. Ein Bin-Fingerprint kann extrem spärlich sein oder fast die gesamte Map abdecken. Ein seltener Sensorwert ist möglicherweise nur an wenigen Trainingspositionen aufgetreten; ein stabiler oder häufig beobachteter Wert kann sich über viele Positionen verteilen. Die Dichte ist daher eine Eigenschaft dieses Sensorwerts in den Trainingsdaten und für sich allein kein Bestanden-/Nicht-bestanden-Kriterium.



Abbildung 15. Vier Bins aus drei Sensoren reichen von 65 bis 2.854 aktiven Positionen und machen die erwartete Variabilität sichtbar.

Diese Fingerprints sind **zählwertbasiert** und nicht binär. Ein belegtes Quadrat speichert, wie oft das Bin an dieser Map-Position aufgetreten ist. Dunkleres Blau steht für einen größeren, helleres Blau für einen kleineren Zählwert. Zwei Fingerprints können daher ähnliche Positionen belegen und dennoch unterschiedliche Häufigkeitsinformationen tragen.

Verwerfen Sie einen spärlichen Fingerprint nicht allein deshalb, weil große Teile des Rasters leer sind, und nehmen Sie nicht an, dass ein dichter Fingerprint wichtiger ist. Prüfen Sie stattdessen, ob das Muster nicht leer ist, wenn das Bin in den Daten vorkommt, ob Sensor- und Bin-Label stimmen und ob die Werte für aktive Positionen und Gesamtzählwert plausibel sind.

Prüfpunkt: UFP speichert eine 64-x-64-Retina, das Dictionary enthält Fingerprints für alle ausgewählten Sensor-Bins und die geprüften Beispiele zeigen plausible Zählwertschattierungen von spärlich bis dicht.

7. Die historische Zeitreihe in Studio untersuchen

Kehren Sie zu Studio zurück, speichern Sie das Projekt und klappen Sie die Einrichtungs-Panes zu. Wählen Sie **Render Project**. Falls Sensor-, Status- oder Overlap-Kurven nicht erscheinen, wählen Sie **Render Project** erneut.

Wählen Sie zuerst **Fit**. Dadurch erhalten Sie einen Überblick über die gesamte Aufzeichnung, bevor Sie Details untersuchen. Scrollen Sie im Timeline Pane ganz nach unten, bis **Status Pane** und **Overlap Pane** sichtbar sind. Lassen Sie rechts außerdem **Status Library** und **Fingerprint View** geöffnet. Schließen Sie Panes, die nicht zur aktuellen Untersuchung gehören.



Abbildung 16. Die vollständige Untersuchungsansicht zeigt Sensorkurven, aufgezeichneten Status, Overlap-Kanäle, Library-Referenzen und Fingerprint-Vergleich synchron in einem Bild.

Die drei Plotbereiche beantworten unterschiedliche Fragen:

Bereich	Frage
Sensor-Timeline	Was haben die gemessenen Signale getan?
Status Pane	Welcher Betriebszustand wurde zu diesem Zeitpunkt aufgezeichnet oder annotiert?
Overlap Pane	Wie ähnlich war der vollständige Sensorkontext zu jeder benannten Referenz?

Verändern Sie mit **+ Zoom** und **- Zoom** die zeitliche Detailstufe. Sobald Sie hineingezoomt haben, ziehen Sie die horizontale Scrollbar nach links oder rechts, um zu pannen. Der Wert **Resolution** gibt an, wie viele Samples durch einen Pixel repräsentiert werden. Er zeigt damit, ob Sie Monate, Tage oder einzelne Datensätze betrachten. **Fit** stellt die gesamte Aufzeichnung wieder her.

Aktivieren Sie **Lasso**, wenn Sie ein Intervall statt einer einzelnen Cursorposition untersuchen möchten. **Gap Lines** machen komprimierte Zeitlücken sichtbar; **Status Lines** führen Statusübergänge durch den

Sensor-Viewport. Alle Aktionen verwenden dieselbe Zeitachse. Cursorposition oder Lasso-Bereich bleiben daher über Sensor-, Status- und Overlap-Strips hinweg ausgerichtet.

Interpretieren Sie Overlap als Ähnlichkeitsmaß. Ein hoher Baseline-Overlap bedeutet, dass der aktuelle Kontext der Baseline-Referenz ähnelt; ein hoher `Failure1`-Overlap bedeutet Ähnlichkeit mit der gespeicherten Referenz `Failure1`. Ähnlichkeit ist ein Hinweis auf den Systemzustand, für sich allein jedoch weder eine garantierte Diagnose noch eine exakte Restlebensdauerprognose.

Prüfpunkt: Fit, Zoom, horizontales Panning und Lasso sind verstanden; Sensor, Status, Overlap, Status Library und Fingerprint View bleiben während der Interpretation gemeinsam sichtbar.

8. Eine Library benannter Betriebszustände aufbauen

Die Status Library verbindet zwei Beschreibungsebenen. Ein Name wie `Healthy - efficient operation`, `Failure1` oder `Maintenance - post service` ist symbolisch und für Bedienpersonal verständlich. Hinter diesem Namen liegt ein numerischer Kontextvektor, den Studio und EngineRT mit jedem Datensatz vergleichen können.

Historische Statuskanäle sind hierbei besonders wertvoll, weil sie zeigen, wann bekannte Betriebszustände vorlagen. Solche benannten Zeitpunkte müssen keine Fehler sein. Zeiträume, in denen die Pumpe perfekt arbeitete, Inbetriebnahmephasen, Wartungsereignisse oder zusätzliche Annotationen können zu nützlichen Referenzen werden, sofern sie einen überprüften physischen Zustand beschreiben.

8.1 Einen einzelnen Datensatz erfassen

Klicken Sie auf eine genaue Position in der Timeline, um einen einzelnen Zeitreihendatensatz zu verwenden. Für `Failure1` besteht eine gültige Methode darin, den Cursor exakt auf die steigende Flanke einer überprüften Phase mit `status = high` zu setzen.

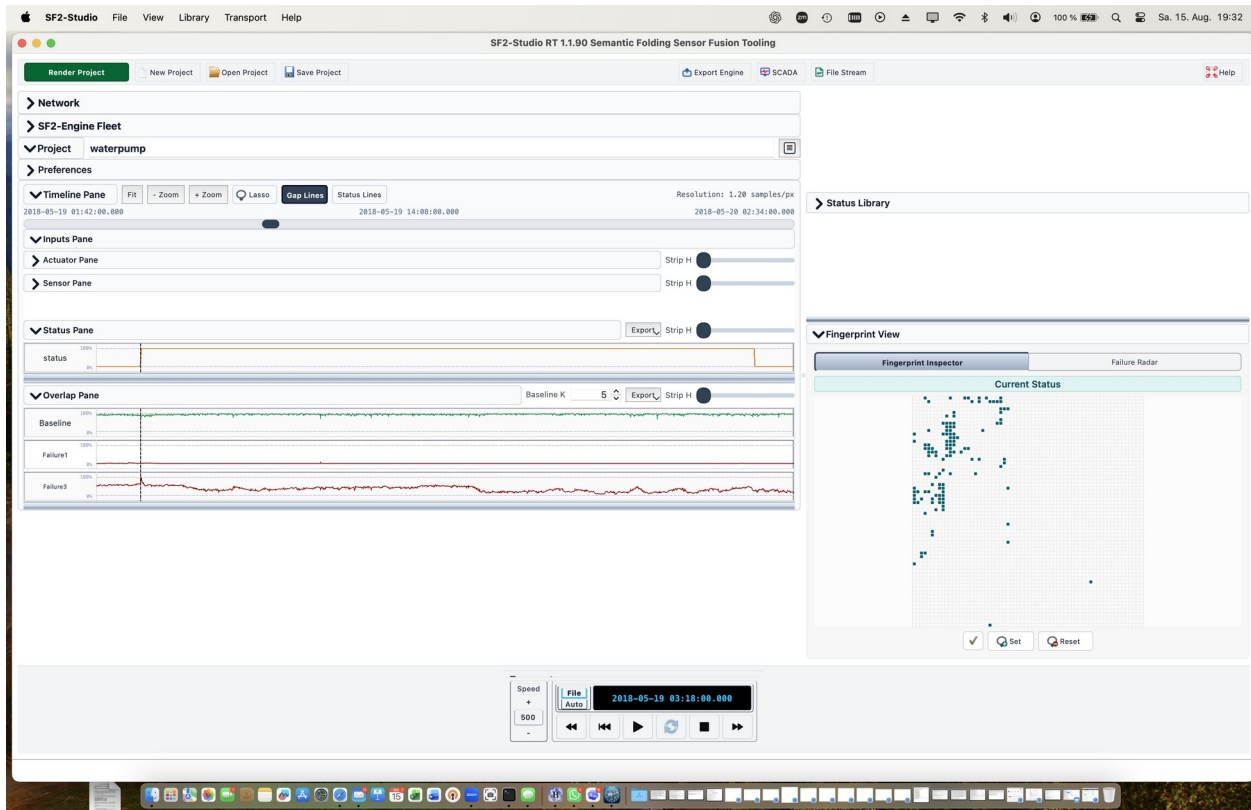


Abbildung 17. Ein einzelner Klick synchronisiert den Cursor über die Timeline und erzeugt den aktuellen Fingerprint aus einem Datensatz.

8.2 Einen Abschnitt erfassen

Falls ein einzelner Zeitpunkt zu empfindlich gegenüber Rauschen ist, aktivieren Sie **Lasso** und ziehen Sie über die gesamte zusammenhängende High-Phase. Studio erzeugt für jeden ausgewählten Datensatz einen Fingerprint und summiert diese zu einem aggregierten **Section Fingerprint**. Wählen Sie nur Datensätze aus, die tatsächlich zum selben benannten Zustand gehören. Normalbetrieb, Lücken oder ein anderer Betriebsmodus dürfen nicht allein wegen ihrer zeitlichen Nähe einbezogen werden.

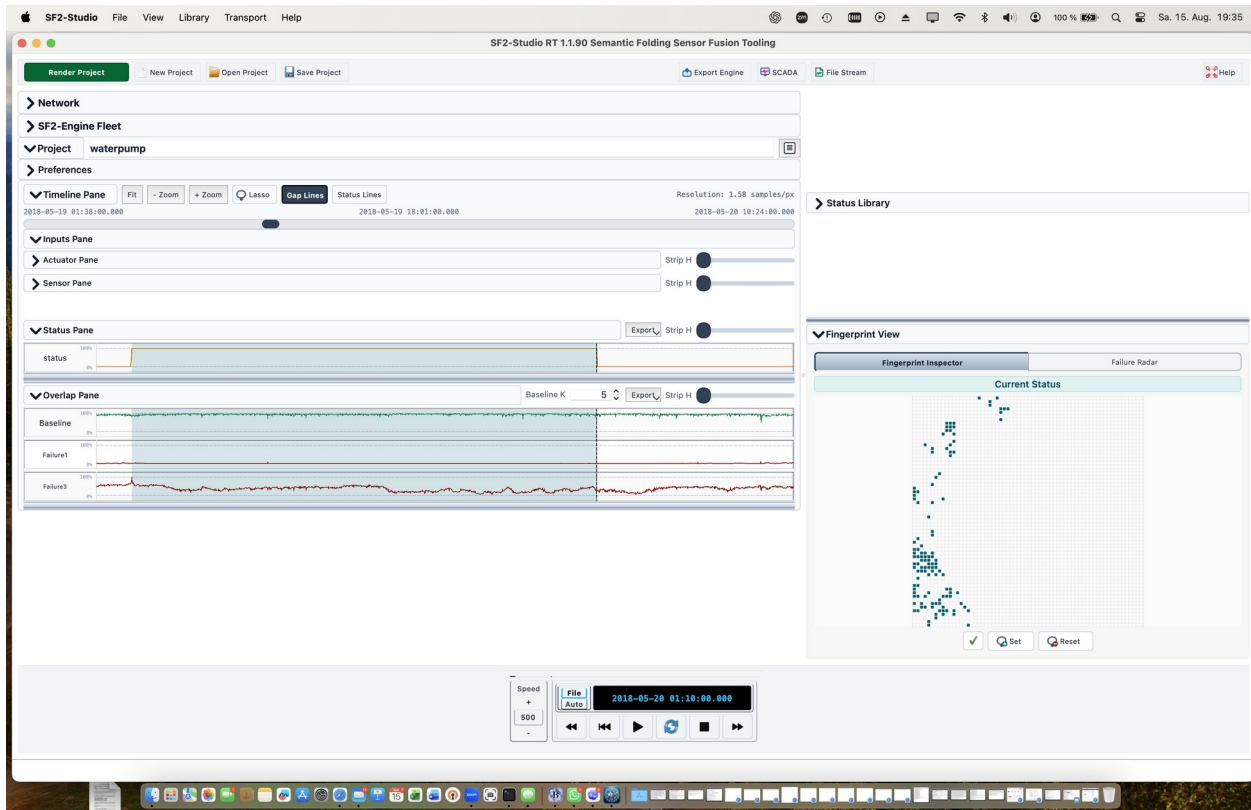


Abbildung 18. Eine Lasso-Auswahl aggregiert alle Datensätze einer überprüften High-Phase zu einem repräsentativeren Section Fingerprint.

Geben Sie **Fingerprint View** genügend Breite, damit Raster und Schaltflächen nicht übereinanderliegen. Unabhängig davon, ob die Quelle ein Datensatz oder ein Abschnitt ist, erscheint das Ergebnis im Fingerprint Inspector. Wählen Sie den grünen Haken mit der Funktion **Accept current fingerprint into Status Library**.

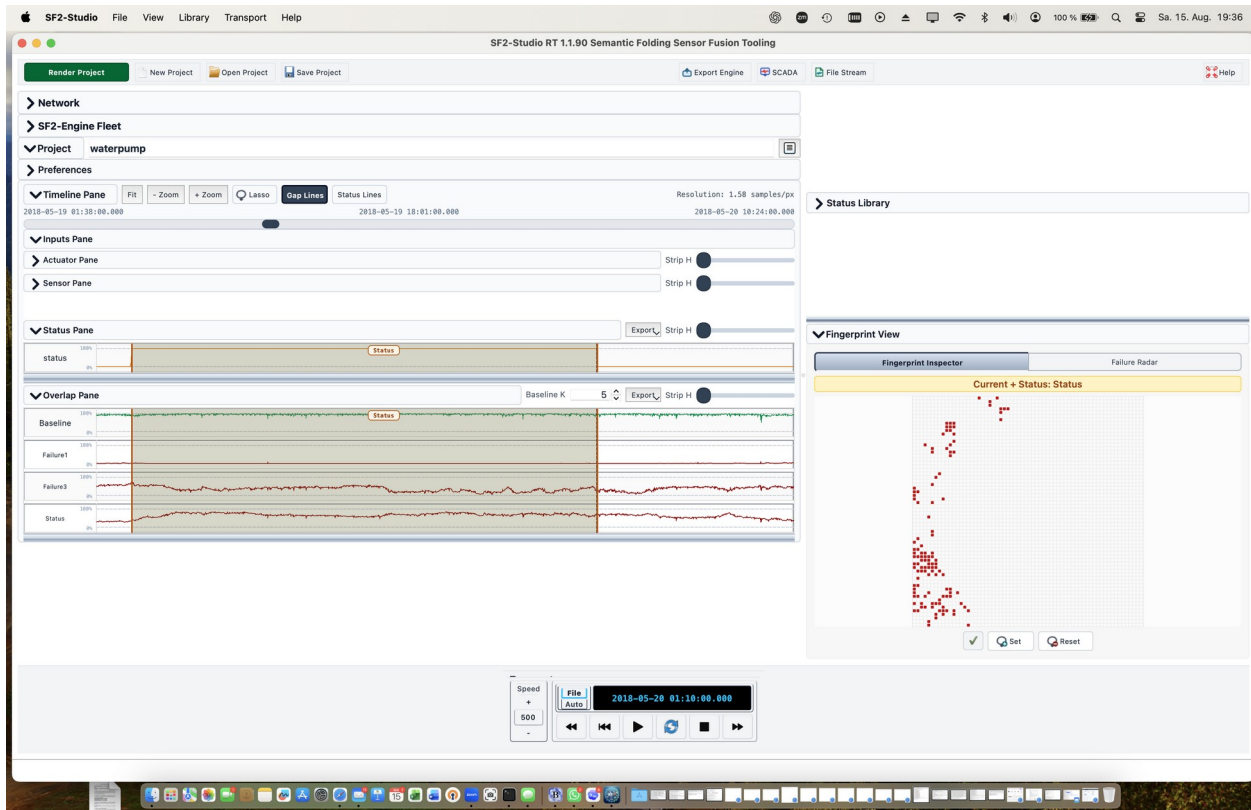


Abbildung 19. Der grüne Haken überträgt den numerischen Fingerprint in die Status Library.

Benennen Sie den neuen Eintrag sofort mit einer physischen, wiederverwendbaren Beschreibung. Im Tutorial wird **Failure - Pump degradation** verwendet; eine produktive Library kann beliebig viele benannte Zustände enthalten.

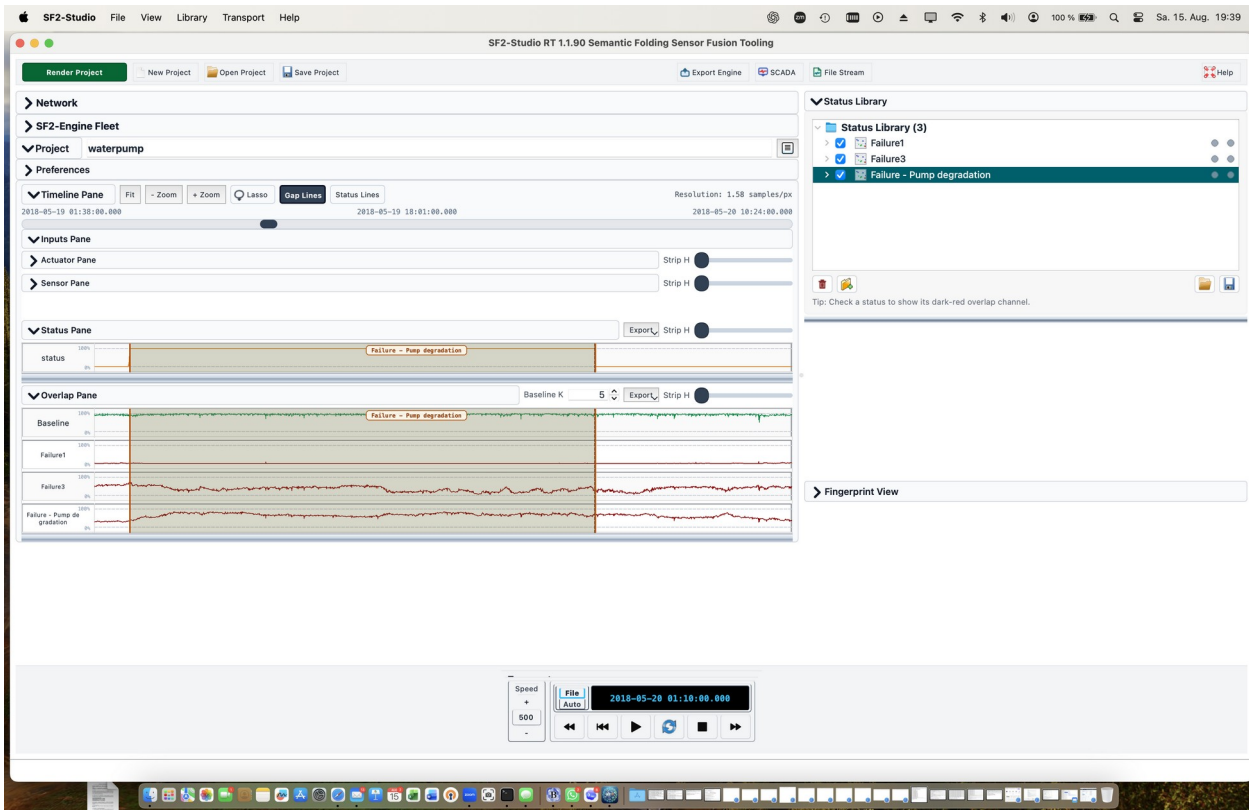


Abbildung 20. Durch die Benennung wird der symbolische Betriebszustand mit seinem numerischen Fingerprint-Vektor verknüpft.

8.3 Die ausgewählte Referenz interpretieren

Wählen Sie einen Library-Fingerprint aus. Die gespeicherte Referenz erscheint gelb, der aktuelle Cursor- oder Section-Fingerprint blau beziehungsweise türkis und gemeinsam belegte Positionen rot. Dieser Vergleich macht den numerischen Overlap sichtbar, anstatt ihn auf einen einzelnen Prozentwert zu reduzieren.

Gleichzeitig zeigt Studio den Ursprungsmarker der Referenz in der Timeline und erzeugt im Overlap Pane einen benannten Kanal. Für `Failure1` bleiben drei Darstellungen miteinander verknüpft:

1. Der **Eintrag in der Status Library** speichert und benennt den Referenzvektor.
2. Der **Timeline-Marker** zeigt den Datensatz oder Abschnitt, aus dem die Referenz erfasst wurde.
3. Der **Overlap-Strip `Failure1`** zeigt für jeden Zeitpunkt, wie stark der jeweilige Kontext der gespeicherten Referenz ähnelt.

Die Kurve stellt damit für jeden historischen Datensatz dieselbe Frage: *Wie ähnlich war der vollständige Pumpenkontext in diesem Moment dem Kontext mit dem Namen `Failure1`?*

Die Checkbox eines Library-Eintrags steuert nur die Sichtbarkeit der Overlap-Kurve; sie aktiviert keinen Alarm. Speichern Sie die Status Library über die Disketten-Schaltfläche und sichern Sie das Studio-

Projekt, wenn die Änderung dauerhaft erhalten bleiben soll. **Set** und **Reset** dienen Expertenänderungen im Fingerprint Inspector. Manuelle Änderungen sollten immer überprüft und dokumentiert werden.

Prüfpunkt: Sie können einen Fingerprint aus einem Datensatz oder Abschnitt erzeugen, übernehmen und benennen, den gelb-blau-roten Vergleich interpretieren und die Referenz vom Library-Eintrag über den Ursprungsmarker bis zur Overlap-Kurve verfolgen.

9. Von Fehlerähnlichkeit zu Predictive Maintenance

Wählen Sie **Failure1** in der Status Library. Zoomen und pannen Sie so, dass der Viewport am Anfang der Aufzeichnung beginnt und kurz nach der ersten High-Phase von **Failure1**, aber vor der nächsten Fehlerperiode endet. Dadurch werden spätere, nicht zugehörige Ereignisse ausgeblendet und die Entwicklung zum ersten Fehler wird lesbar.



Abbildung 21. Die ausgewählte Referenz **Failure1**, ihr Ursprungsmarker, der aufgezeichnete Statusübergang und der frühere Anstieg des Overlaps sind in einem fokussierten Fenster sichtbar.

Am Ursprungsmarker ist der ausgewählte Kontext die Referenz selbst und besitzt daher 100 % Selbst-Overlap. Entscheidend ist, was vor diesem Marker geschieht. Die Overlap-Kurve **Failure1** beginnt mehr als vier Tage vor dem aufgezeichneten Totalausfall anzusteigen. Ihr genauer Verlauf ist dynamisch, weil sich die Betriebsbedingungen weiterhin ändern. Die übergeordnete Bewegung bleibt jedoch bestehen: Je näher der Fehler rückt, desto ähnlicher wird der Pumpenkontext zu **Failure1**.

Dies ist der konzeptionelle Kern von SF2 Predictive Maintenance. Viele industrielle Ausfälle treten nicht spontan auf. Ein initiales Ereignis, etwa das Brechen einer Kugel in einem Wellenlager, verändert den

Systemkontext. Sekundärschäden und kompensierendes Verhalten bauen sich anschließend auf, bis der endgültige Ausfall aufgezeichnet wird. Ein im Endzustand erfasster Fingerprint erlaubt SF2, die vollständige multivariate Historie rückwärts zu durchsuchen und sichtbar zu machen, wann sich das System erstmals in Richtung dieses Kontexts bewegte.

Die Kurve behauptet weder, dass jeder Anstieg einen Fehler darstellt, noch berechnet sie unmittelbar die verbleibende Nutzungsdauer. Sie liefert eine interpretierbare Ähnlichkeitstrajektorie. Fachliche Prüfung, wiederholte Beispiele und Betriebskontext bestimmen, welcher Teil dieser Trajektorie früh genug für eine Maßnahme und zugleich spezifisch genug für eine verlässliche Aussage ist.

9.1 Eine überprüfte Trajektorie in einen Alarm überführen

Klappen Sie den ausgewählten Fingerprint in der Status Library auf. Jede Referenz besitzt eigene Alarm-Bedienelemente.

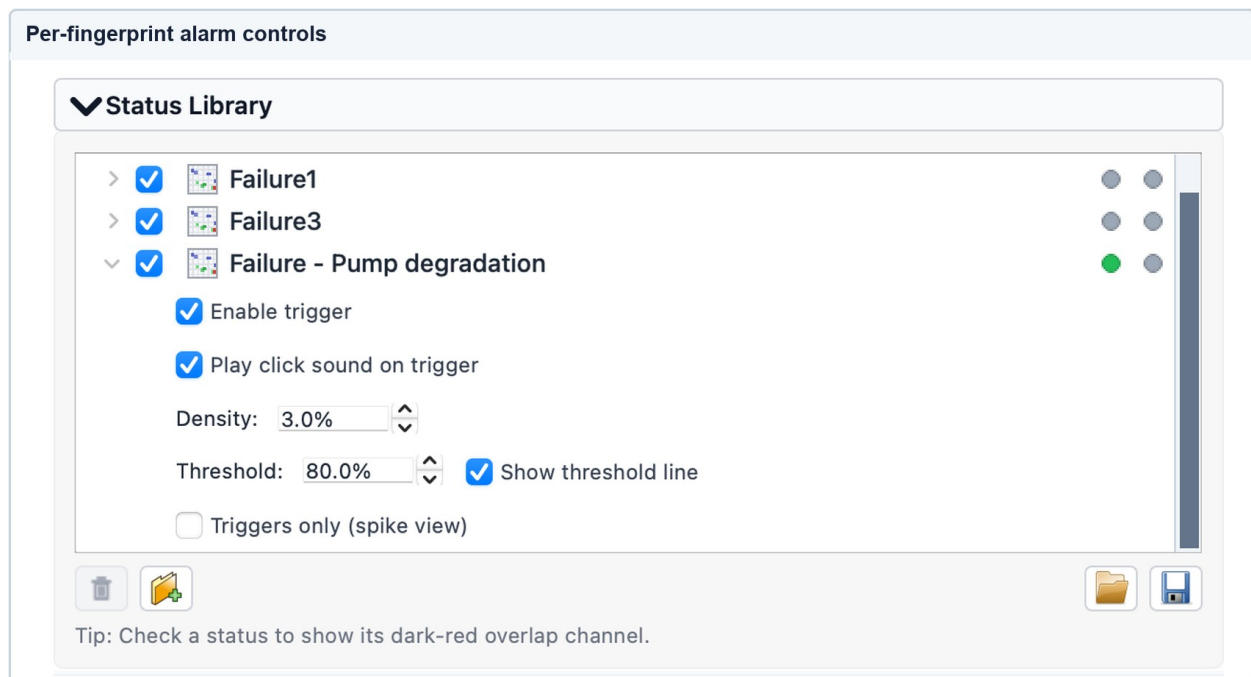


Abbildung 22. Der aufgeklappte Library-Eintrag zeigt Trigger-Aktivierung, Dichte, Overlap-Schwellwert, Schwellwertlinie, lokalen Ton und die Trigger-only-Darstellung.

Bedienelement	Zweck
Library-item checkbox	Blendet die Overlap-Kurve ein oder aus; der Trigger wird dadurch niemals aktiviert.
Enable trigger	Erzeugt FIRE beim Überschreiten und CLEARED beim Unterschreiten des Schwellwerts.
Play click sound on trigger	Gibt eine lokale Rückmeldung in Studio aus.
Density	Definiert die für die gespeicherte Referenz verwendete Sparsity und sollte vor dem Abstimmen des Schwellwerts geprüft werden.

Bedienelement	Zweck
Threshold	Legt den Overlap-Prozentwert fest, ab dem der benannte Zustand zu einem Ereignis wird.
Show threshold line	Zeichnet den konfigurierten Schwellwert direkt in den Overlap-Strip ein und erleichtert die visuelle Abstimmung.
Triggers only (spike view)	Ersetzt die kontinuierliche Kurve durch Ereignismarker, wenn nur Schwellwertüberschreitungen benötigt werden.

Bestimmen Sie den Schwellwert anhand repräsentativer historischer Timelines. Ein zu niedriger Wert kann zu übermäßig vielen Fehlalarmen führen; ein zu hoher Wert kann ein reales Ereignis übersehen oder zu spät erkennen. Das Ziel ist nicht die größtmögliche Vorwarnzeit, sondern der früheste Schwellwert, der handlungsrelevante Degradation noch zuverlässig von normaler Variation trennt. Lassen Sie **Show threshold line** aktiviert, während Sie mögliche Werte mit bekannten Statusperioden vergleichen.

Manche Systeme nähern sich wiederholt einem fehlerähnlichen Kontext, ohne tatsächlich auszufallen. Beispielsweise kann eine saisonal geringe Wassermenge die Ähnlichkeit jedes Jahr erhöhen, während der echte Fehler nur dann eintritt, wenn geringe Wassermenge und eine über mehrere Stunden erhöhte Umgebungstemperatur zusammentreffen. Solche Beispiele müssen in der Schwellwertvalidierung enthalten sein. Ein brauchbarer Alarm erkennt jeden überprüften Fehler früh genug und bleibt zugleich bei nicht ausfallenden Annäherungen stabil.

Die Bedienelemente pro Fingerprint bestimmen, *wann* ein Ereignis vorliegt. Die globalen Einstellungen unter **Network > Uplink** bestimmen, *wohin* es übertragen wird:

Alarmwirkung	Konfiguration
Nur visuelle Analyse	Overlap und Schwellwertlinie anzeigen; Enable trigger ausgeschaltet lassen.
Lokale Rückmeldung für Bedienpersonal	Trigger und Klickton aktivieren; externe Veröffentlichung ausgeschaltet lassen.
SCADA über REST	Trigger und Veröffentlichung über REST-Webhook aktivieren.
SCADA oder Ereignisverarbeitung über MQTT	Trigger und MQTT-Veröffentlichung aktivieren.
Redundante Ziele	REST und MQTT gemeinsam aktivieren.

Für MQTT stehen Host, Port, Client-ID, Topic, QoS, Zugangsdaten, Retry-Queue, Overflow-Policy und Spool-Einstellungen zur Verfügung. REST stellt Webhook-URL und Selbsttest bereit. **MQTT Self-Test** und **REST Self-Test** prüfen die Verbindung; **Send Trigger Test** prüft das Routing des Payloads. Diese Tests belegen nicht, dass der ausgewählte Fingerprint oder Schwellwert betrieblich korrekt ist.

Prüfpunkt: Die Kurve *Failure1* wird als frühe Ähnlichkeitstrajektorie verstanden, der Schwellwert wird aus realen Timelines abgeleitet und nicht geraten, und die Trigger-Aktivierung bleibt klar von Overlap-Sichtbarkeit und Transportkonfiguration getrennt.

10. Das signierte Engine-Paket exportieren und validieren

Wählen Sie **Export Engine**. Behalten Sie die portablen **Broadcast**-Deployment-Capabilities bei, damit Quell- und Zielpunkte beim Übernehmen der lokalen Engine aufgelöst werden können.

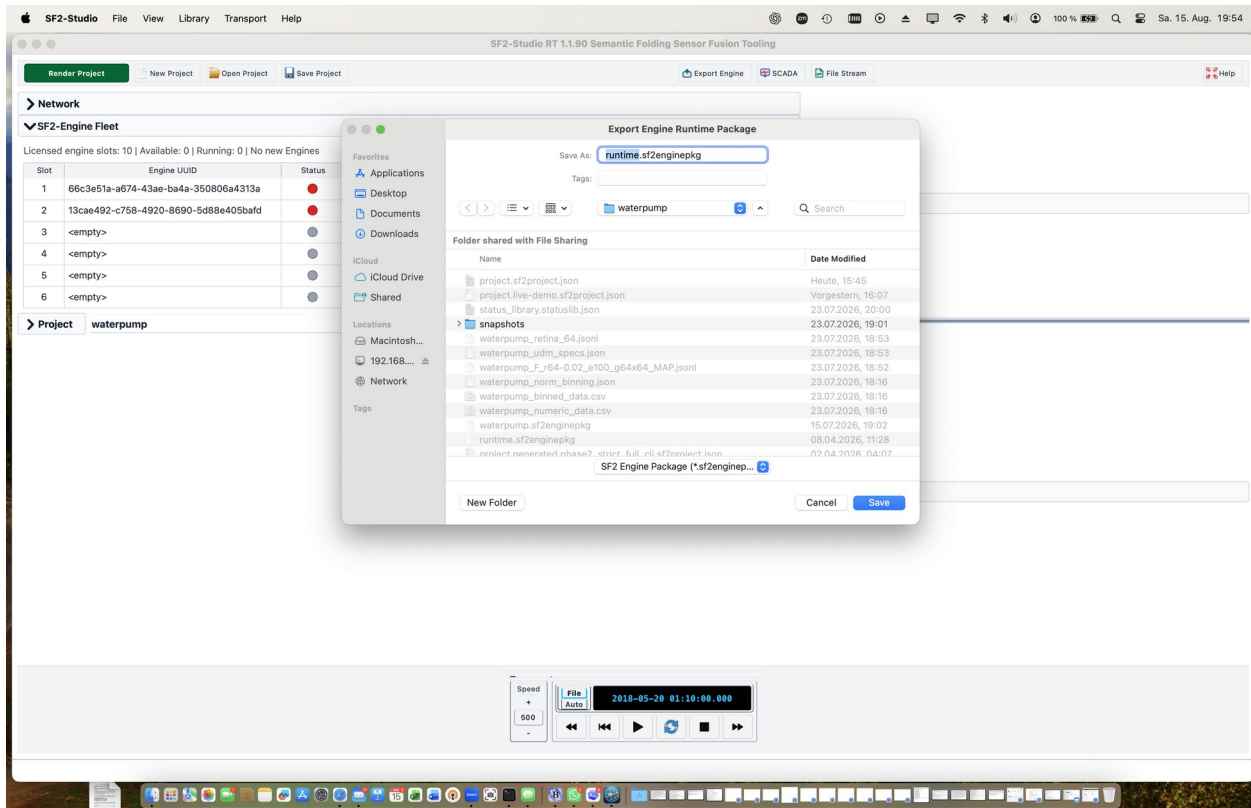


Abbildung 23. Studio bündelt Retina, Status Library, Trigger-Regeln und portable Runtime-Capabilities.

Exportieren Sie `runtime.sf2enginepkg`. Das Tutorial-Paket ist mit Ed25519 signiert und enthält die erforderlichen Map-, Retina-, Rezept- und Status-Library-Blobs. EngineRT prüft Signatur und Hashes vor der Ausführung. Im Produktivbetrieb muss die Vertrauenskonfiguration der Engine den Signaturschlüssel enthalten.

Beschreiben Sie das Paket, ohne es zu starten:

```
SF2-EngineRT \
--package /absolute/path/runtime.sf2enginepkg \
--describe-only
```

Der aktuelle Waterpump-Export meldet eine vorhandene Ed25519-Signatur, vier erforderliche Artefakt-Blobs, Broadcast-Capabilities und drei konfigurierte Trigger. Führen Sie danach den deterministischen Smoke-Test aus:

```
SF2-EngineRT --headless \  
--package /absolute/path/runtime.sf2enginepkg \  
--input-csv /absolute/path/sensor_in.csv \  
--run-smoke \  
--output-json /absolute/path/waterpump-engine-smoke.json
```

Der verifizierte Tutorial-Lauf wurde mit `ok: true`, 220.320 Eingabezeilen sowie gültigen erforderlichen Hashes und Signaturnachweisen abgeschlossen. Fahren Sie nicht fort, wenn das Paket nicht signiert ist, ein Blob-Hash abweicht oder die Zeilenanzahl unerwartet ist.

Prüfpunkt: `runtime.sf2enginepkg` ist signiert, die Describe-only-Validierung ist erfolgreich und der Smoke-Report akzeptiert alle 220.320 Zeilen.

11. Eine lokale Engine per Plug-and-Play-Adoption in Betrieb nehmen

Dieses Tutorial demonstriert **ausschließlich eine lokale Engine-Instanz**. Lassen Sie Studio und das Waterpump-Projekt geöffnet. Ein Neustart der Suite ist weder erforderlich noch erwünscht; setzen Sie beim Wiederholen der Sequenz nur die Engine-Instanz zurück.

11.1 Mit einer leeren Fleet beginnen

Stoppen und releasen Sie zuvor verwendete Tutorial-Engines, bis die Fleet keine zugewiesene oder erkannte Instanz mehr enthält.

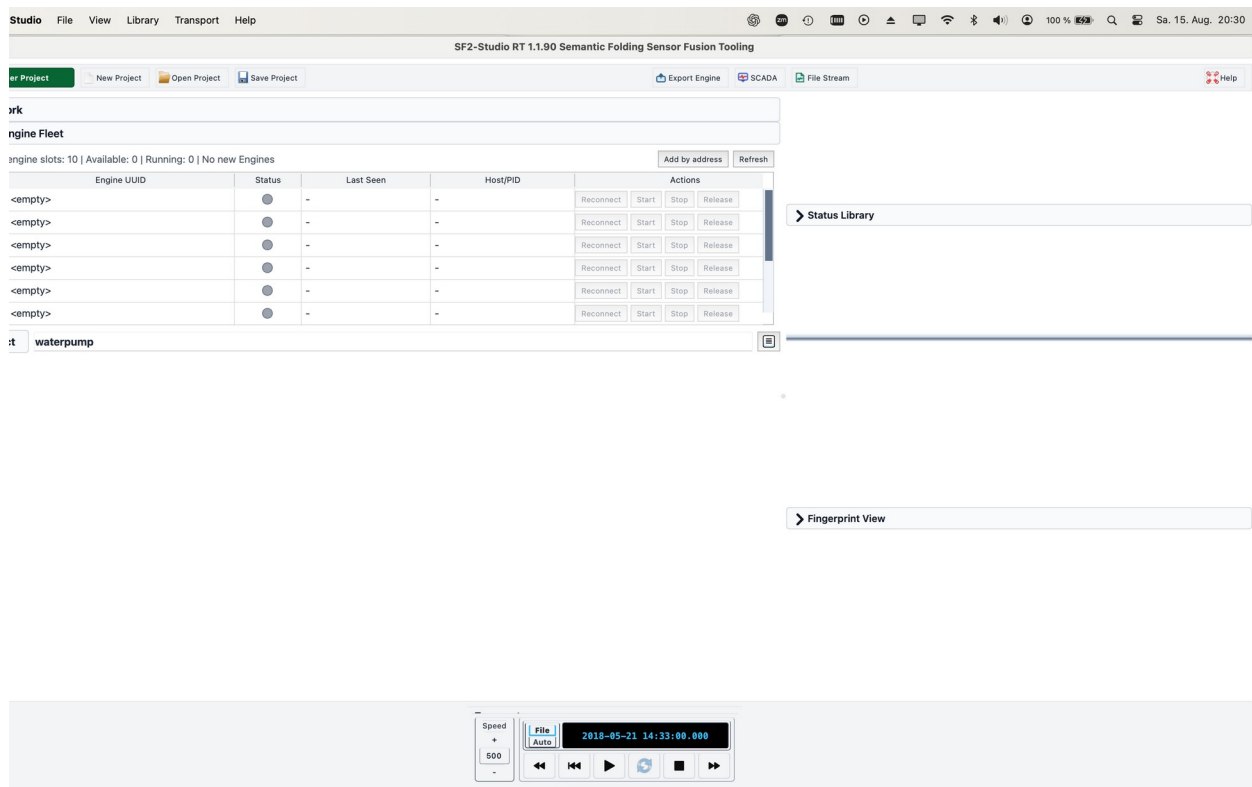


Abbildung 24. Die Sequenz beginnt mit einer leeren Fleet und vollständig freien lizenzierten Slots.

11.2 Den lokalen EngineRT-Dienst starten

Starten Sie eine lokale EngineRT-Instanz mit ihrer Websteuerung auf `127.0.0.1:10032`. Die Webseite bestätigt, dass der Dienst läuft, bevor er mit dem Projekt verbunden wird.



Abbildung 25. EngineRT ist lokal funktionsfähig und wartet auf die Inbetriebnahme.

Verwenden Sie in diesem Ablauf nicht **Add by address**. Der Plug-and-Play-Weg basiert auf automatischer LAN-Discovery. Warten Sie kurz oder wählen Sie **Refresh**, bis Studio **1 engine ready to adopt** meldet.

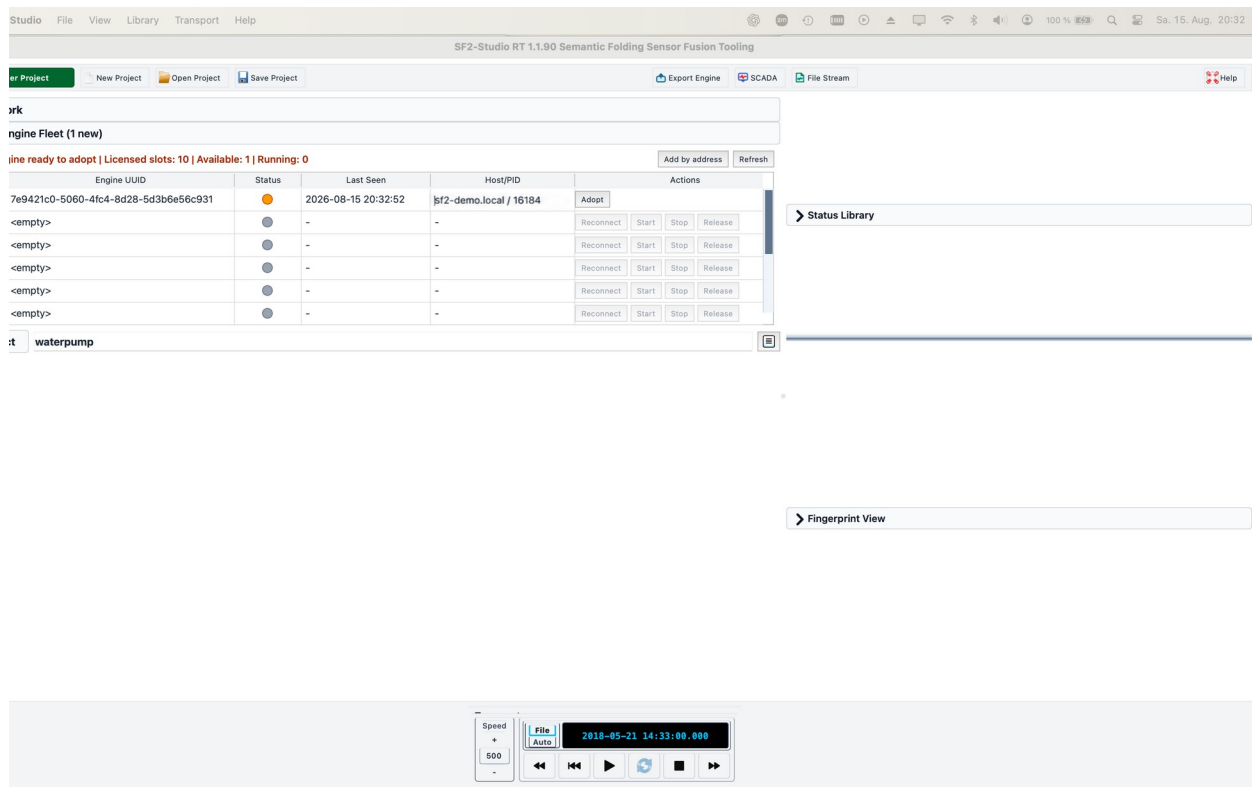


Abbildung 26. UUID und Host/PID der lokalen Engine erscheinen automatisch zusammen mit der Aktion **Adopt**.

11.3 Die erkannte Engine übernehmen und starten

Wählen Sie **Adopt**. Die Adoption belegt einen lizenzierten Slot, bindet diese Engine-UUID an den Gatekeeper der aktuellen Suite und löst die portablen Capabilities des Pakets auf, ohne das signierte Paket zu verändern.

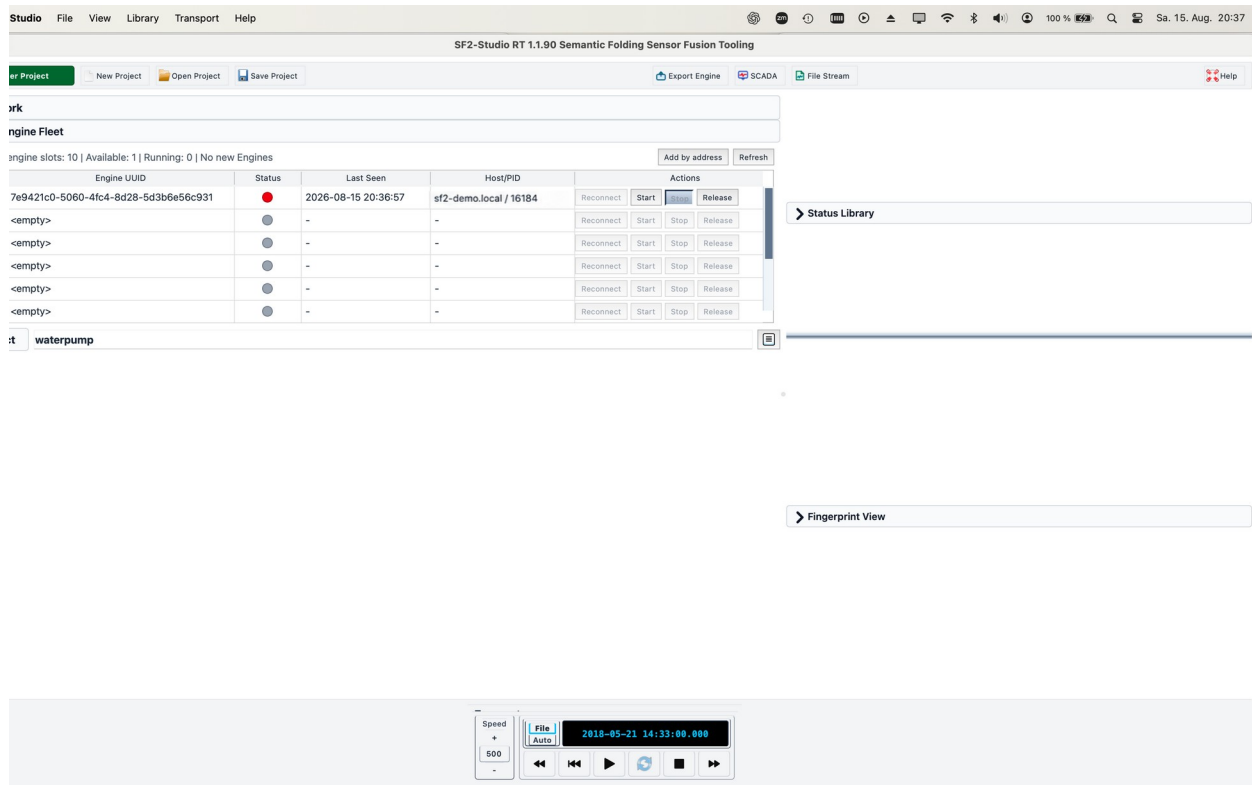


Abbildung 27. Die erkannte Engine ist nun der Suite zugewiesen.

Warten Sie, bis die Statusanzeige grün wird und die Fleet-Zusammenfassung **Available: 1** und **Running: 1** anzeigt.

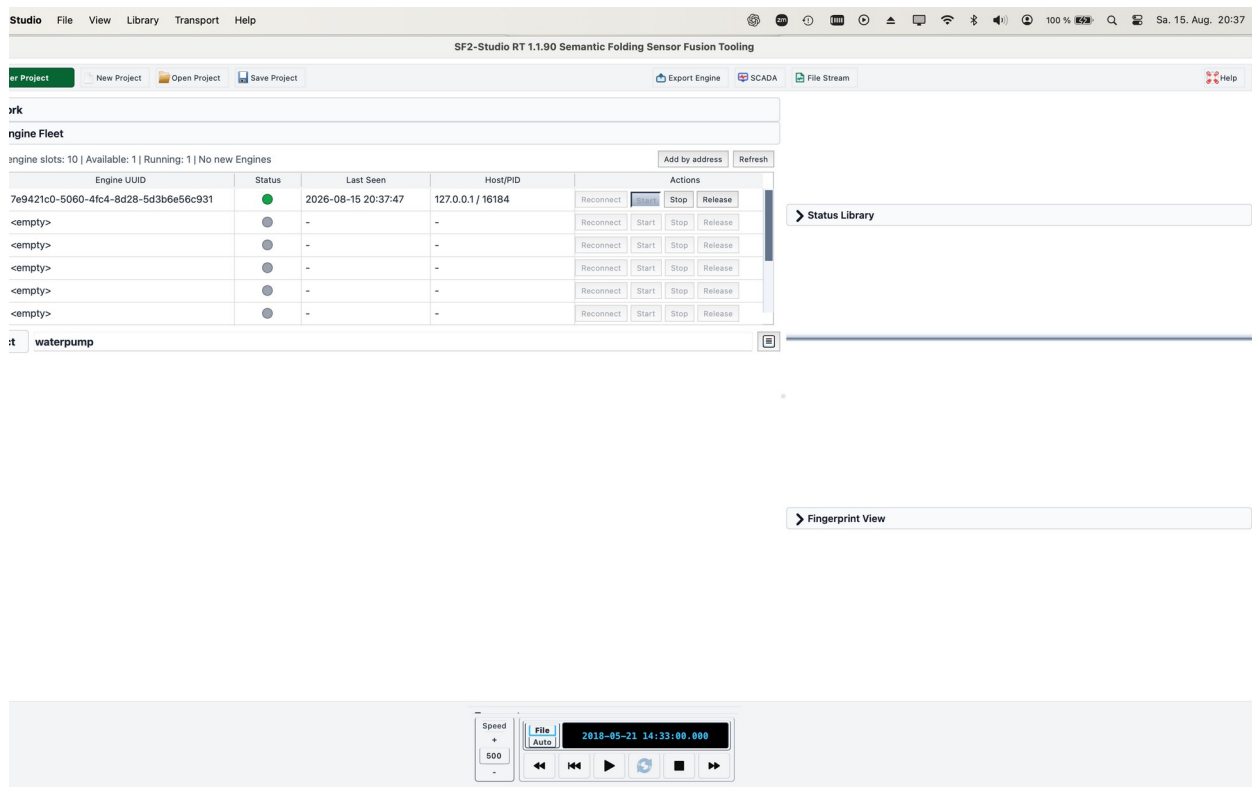


Abbildung 28. Die übernommene lokale Engine erreicht in der Fleet den Zustand Running.

EngineRT kann bereits Running sein, während es noch auf die ausgewählte Live-Quelle wartet. Vor dem Start von File Stream zeigt die Steuerungsseite eine gültige Lease und das Paket gemeinsam mit dem wartenden TCP-Zustand.



Abbildung 29. Die Inbetriebnahme ist abgeschlossen; der nächste Vertrag ist der Live-Input über TCP.

Fleet-Zustand	Richtige Reaktion
Ready to adopt	Für die erkannte Engine Adopt wählen
Waiting	Auflösung der Bindings, Gatekeeper-Lease oder Bereitschaft der Quelle prüfen
Running	Eingehende Frames und Trigger-Auswertung prüfen
Offline (license reserved)	Engine-Prozess beziehungsweise Route wiederherstellen und Reconnect verwenden

Release gibt die lizenzierte Zuweisung frei. Es ist keine Reconnect-Aktion.

Prüfpunkt: Eine automatisch erkannte lokale Engine ist übernommen, die Fleet zeigt eine Running-Instanz und es wurde keine Adresse manuell eingegeben.

12. Das lokale End-to-End-Szenario ausführen

Der Runtime-Nachweis verwendet eine einfache lokale Kette:

File Stream TCP 127.0.0.1:19001 -> EngineRT -> MQTT 127.0.0.1:1883 -> SCADA Simulator

12.1 File Stream konfigurieren

Öffnen Sie **File Stream** aus Studio. Wählen Sie unter **Data Source** `sensor_in.csv` und `stream_config.json` aus und schließen Sie die übrigen Panes.

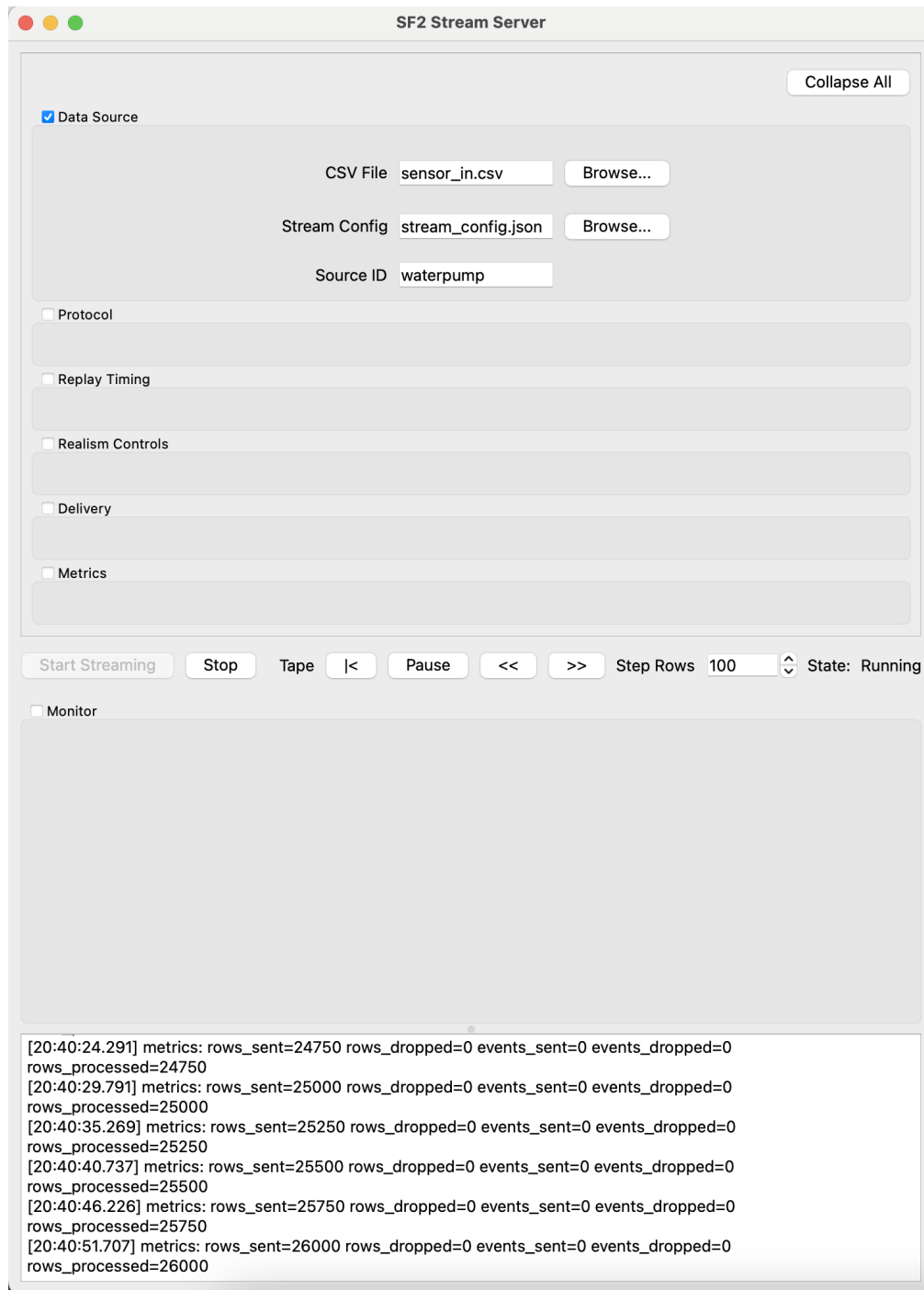


Abbildung 30. File Stream lädt die Waterpump-Aufzeichnung und ihre Stream-Konfiguration.

Öffnen Sie ausschließlich **Protocol & Endpoint**, wählen Sie TCP und verwenden Sie `127.0.0.1:19001`.

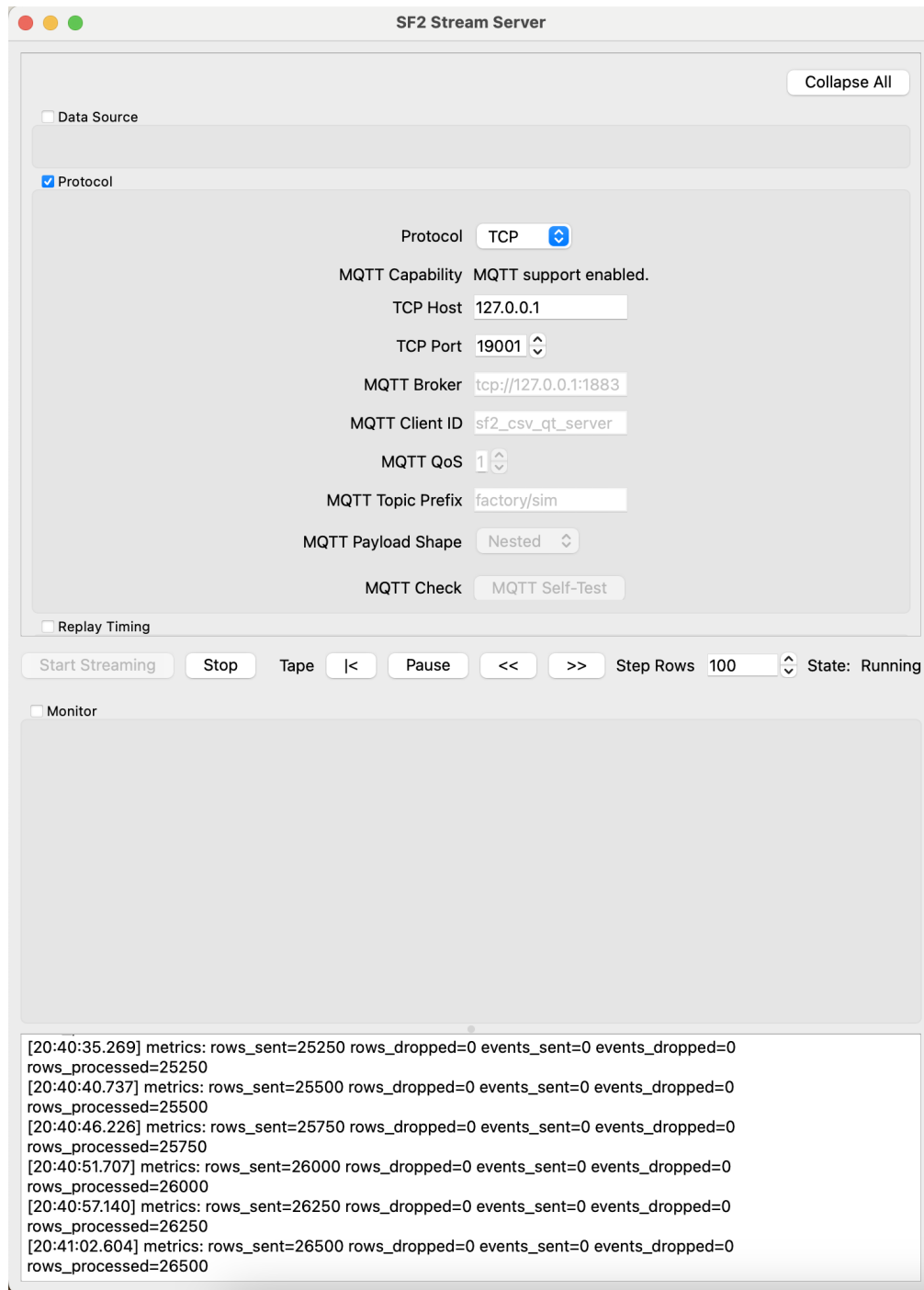


Abbildung 31. Die deterministische lokale Quelle entspricht der während der Adoption aufgelösten Capability.

Starten Sie den Stream und lassen Sie anschließend nur **Monitor** geöffnet. Die Quelle enthält 220.320 Zeilen und 52 rohe Sensorfelder; EngineRT wendet den im Paket enthaltenen Vertrag des 51-Sensor-Modells an.

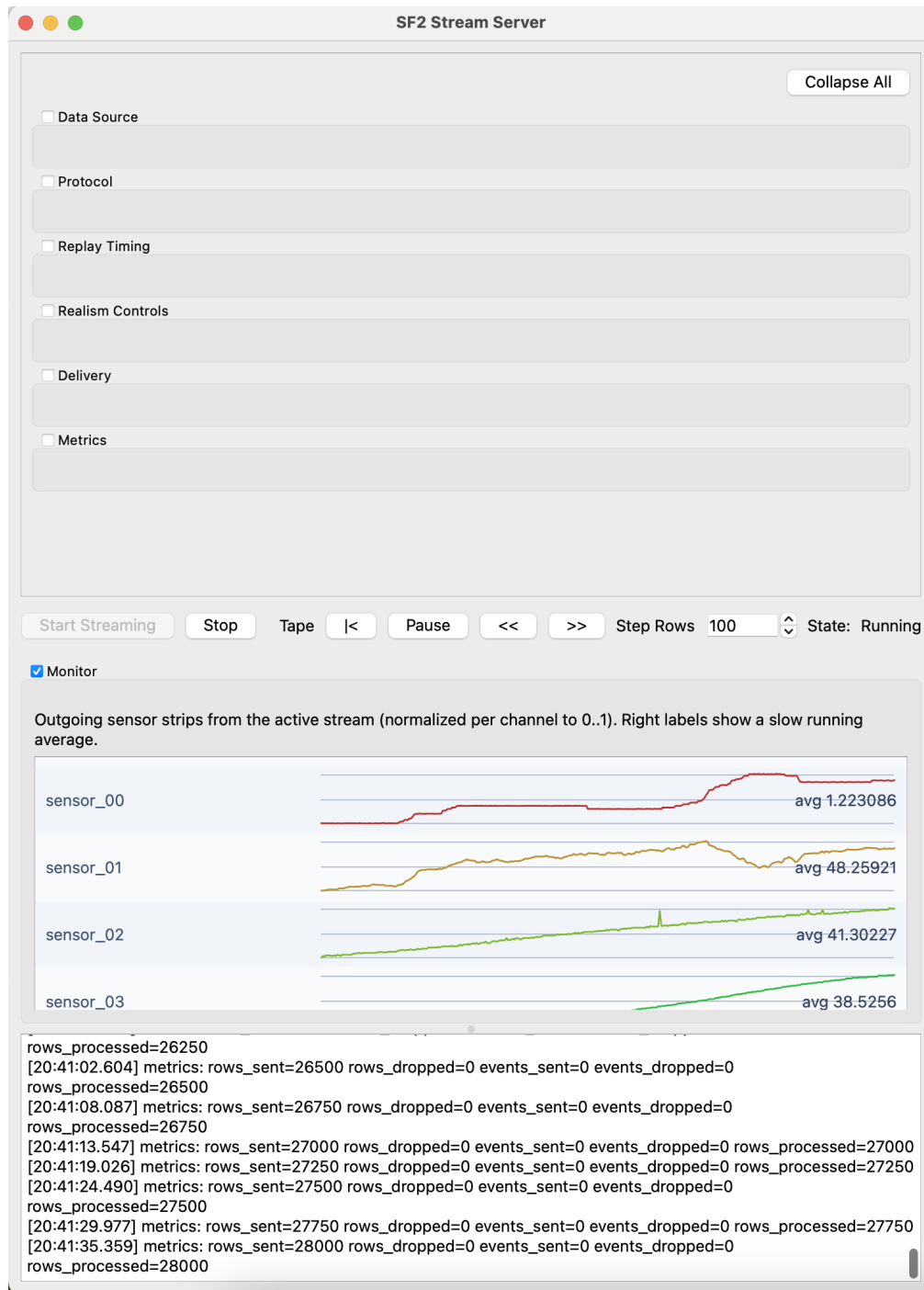


Abbildung 32. Running bestätigt, dass Frames an die lokale Engine ausgegeben werden.

Um die markierte Fehlerregion schnell zu erreichen, verwenden Sie einen großen Zeilenschritt wie 40000, anstatt den Alarmschwellwert zu verändern.

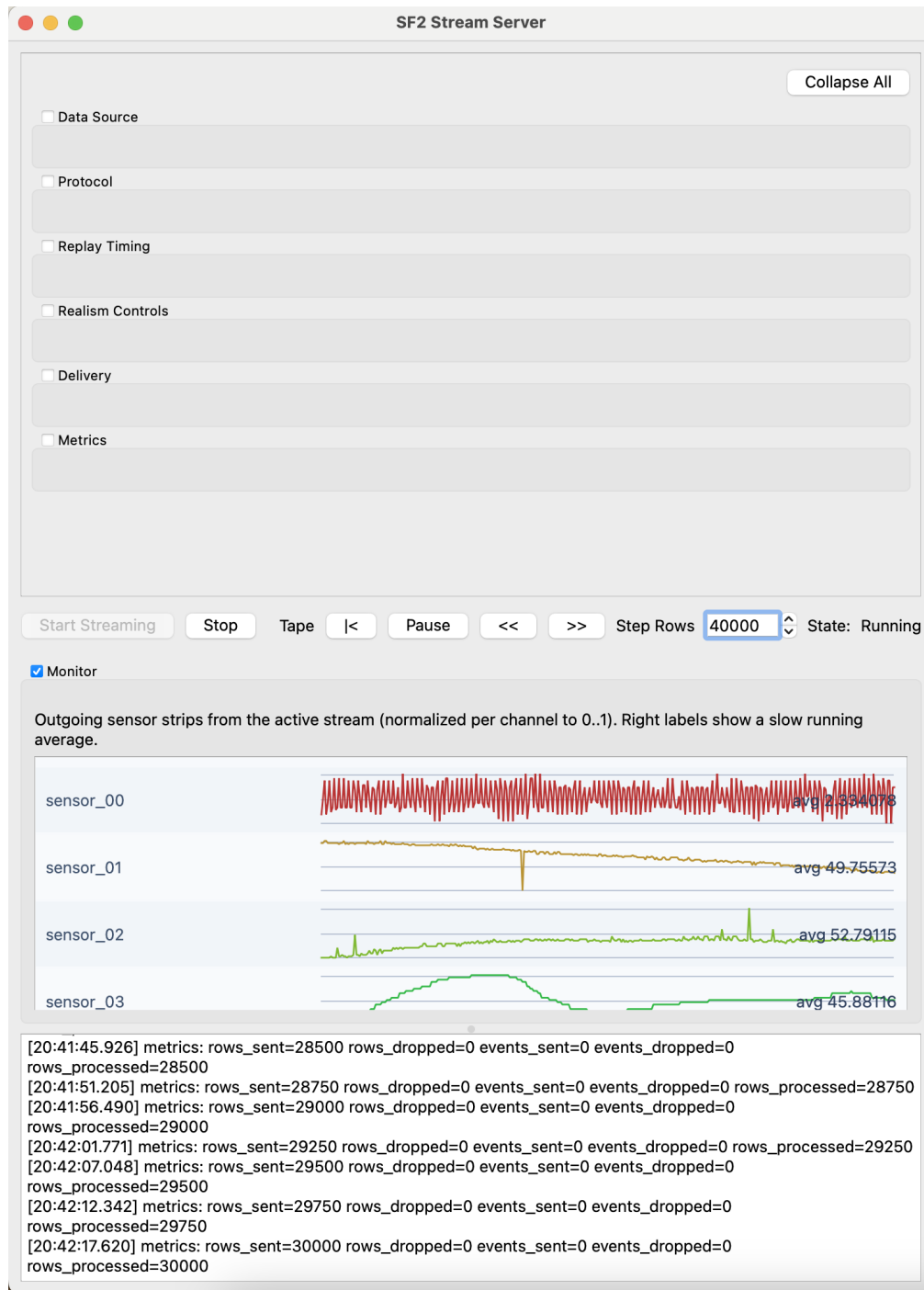


Abbildung 33. Fast-Forward behält den konfigurierten Detektor bei und bewegt sich schneller durch das aufgezeichnete Szenario.

Fahren Sie fort, bis sich der Stream in der Nähe von Zeile 71.847 befindet, an der der demonstrierte Fehlerübergang auftritt.

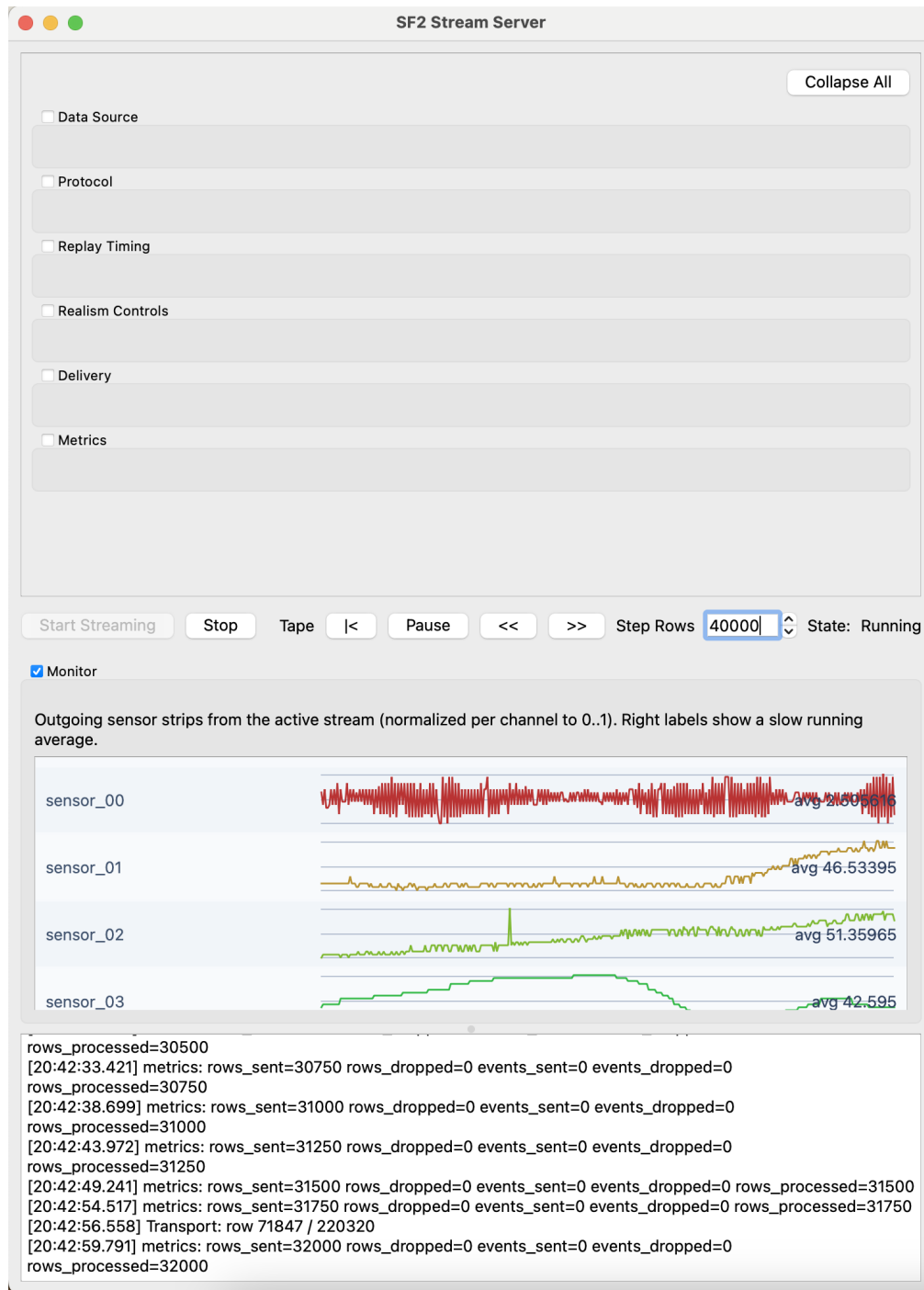


Abbildung 34. Die Wiedergabe befindet sich nun in dem Teil der Aufzeichnung, der für den Runtime-Alarmnachweis verwendet wird.

12.2 Den lokalen SCADA-Empfänger starten

Öffnen Sie **SCADA** aus Studio. Starten Sie den verwalteten MQTT-Broker auf 127.0.0.1:1883, abonnieren Sie mit dem Empfänger `sf2/uplink/alerts` und lassen Sie während der Einrichtung nur das aktive Konfigurations-Pane geöffnet.

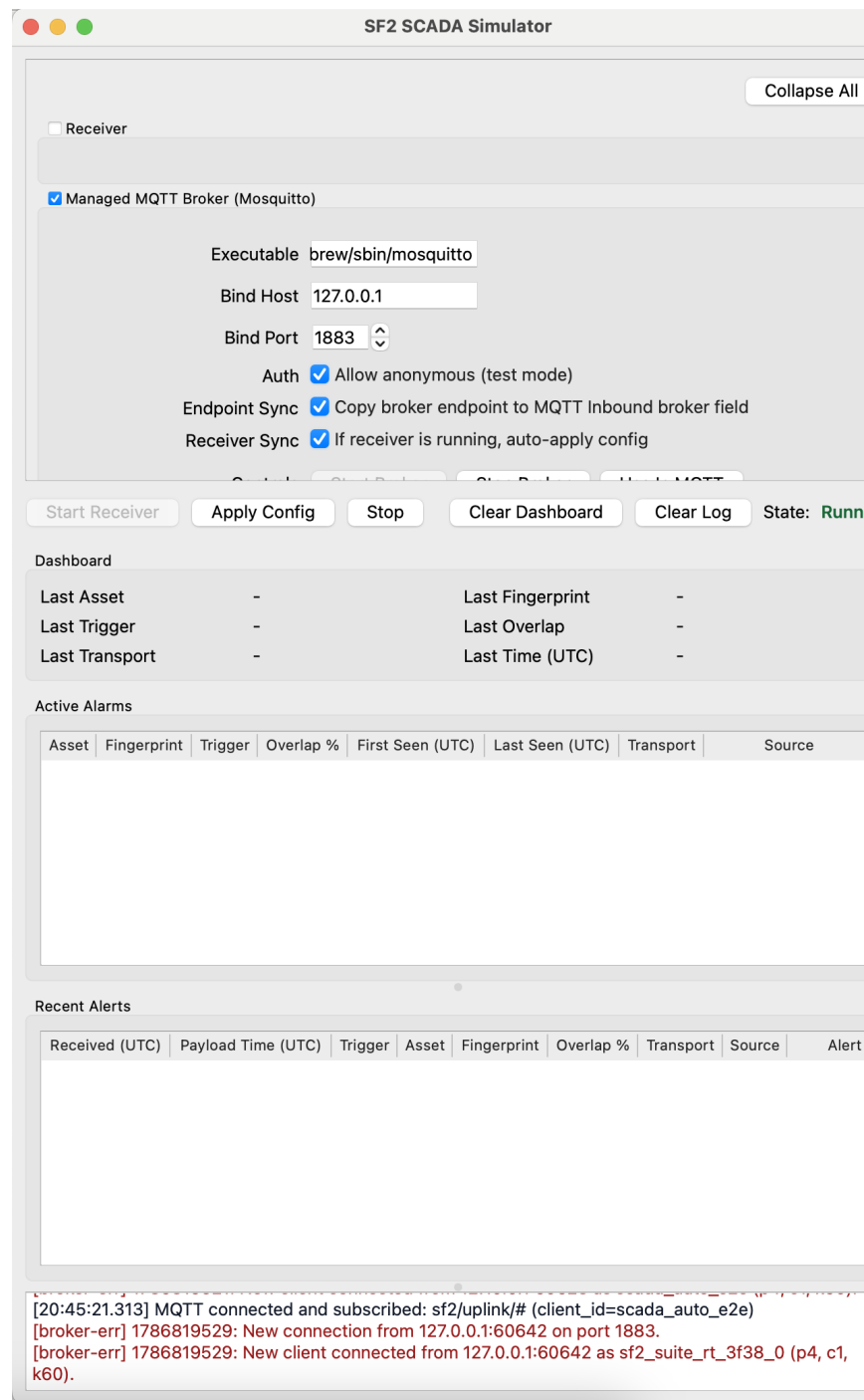


Abbildung 35. Der lokale Broker und das eingehende Abonnement stellen das Alarmziel der Engine bereit.

12.3 Den realen Alarm beobachten

Wenn File Stream die Fehlerbedingung durchläuft, gibt EngineRT die Trigger-Übergänge aus. SCADA empfängt über MQTT für `Failure - Pump degradation` am Asset `asset-001` zunächst `FIRE` und anschließend `CLEARED`.

The screenshot shows the SF2 SCADA Simulator interface. At the top, there are checkboxes for Receiver, Managed MQTT Broker (Mosquitto), MQTT Inbound, REST Webhook Listener, Diagnostics, and Telemetry. Below these are buttons for Start Receiver, Apply Config, Stop, Clear Dashboard, Clear Log, and a State indicator showing 'Runn'. The Dashboard section displays the following data:

Dashboard			
Last Asset	asset-001	Last Fingerprint	Failure - Pump degradation
Last Trigger	CLEARED	Last Overlap	68.852
Last Transport	mqtt	Last Time (UTC)	2026-08-15 18:45:32.348 UT

The Active Alarms section is currently empty. The Recent Alerts section shows the following data:

Received (UTC)	Payload Time (UTC)	Trigger	Asset	Fin
2026-08-15 18:45:32.348 UTC	2026-08-15 17:47:03.540 UTC	CLEARED	asset-001	Failure - Pur
2026-08-15 18:45:32.348 UTC	2026-08-15 17:47:03.535 UTC	FIRE	asset-001	Failure - Pur

Below the Recent Alerts table, there is a log of events:

```
Pump degradation | overlap=68.852 | source=sf2/uplink/alerts
[20:45:32.348] MQTT Rx topic=sf2/uplink/alerts bytes=324
[20:45:32.348] CLEARED a01e2042-56fb-485a-90e9-63e07bed1d03 | asset=asset-001 | status=Failur
- Pump degradation | overlap=68.852 | source=sf2/uplink/alerts
```

Abbildung 36. Das Dashboard enthält das reale, von der lokalen Engine empfangene FIRE/CLEARED-Paar.

Der zuletzt angezeigte Overlap-Wert 68.852 gehört zum Übergang **CLEARED**, nachdem der Overlap wieder gesunken ist. Deshalb kann er unter dem Auslöseschwellwert von 80 % liegen.

Prüfen Sie abschließend EngineRT. `frames_observed` und die Trigger-Edge-Zähler müssen ansteigen, die eingehende TCP-Verbindung muss bestehen und die Gatekeeper-Lease muss aktiv bleiben.

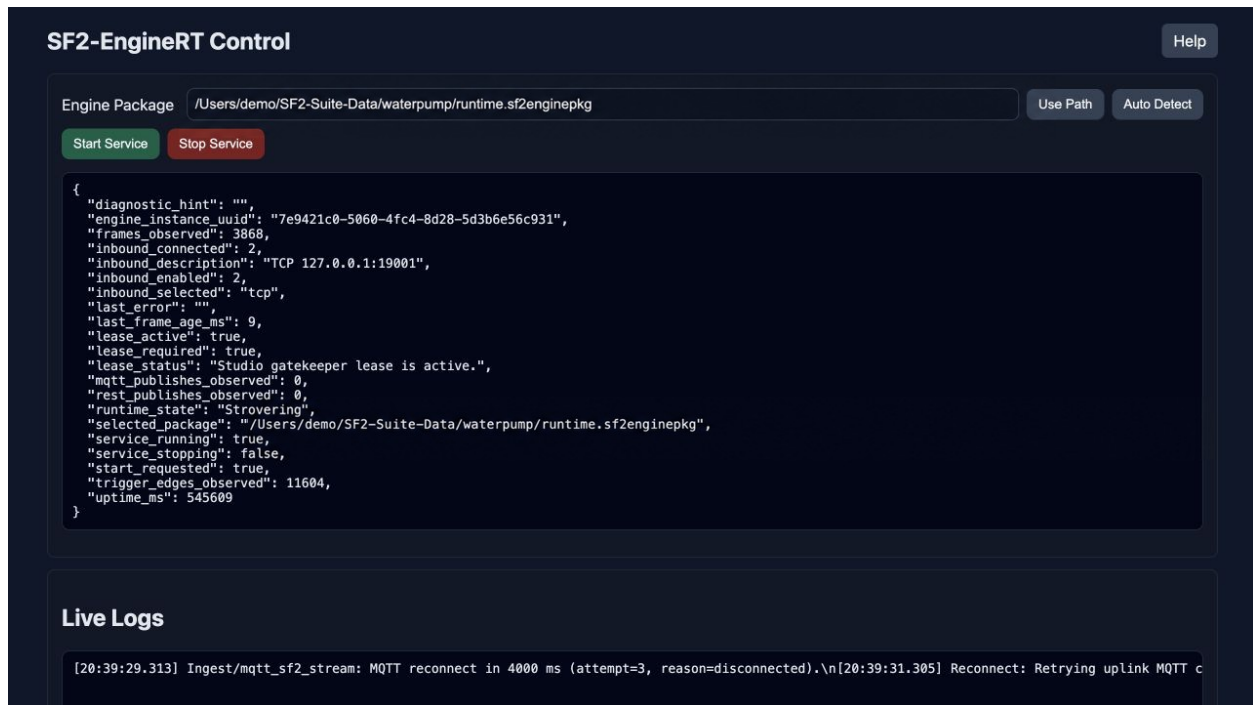


Abbildung 37. EngineRT zeigt das aktive Paket, den eingehenden Stream, fortschreitende Frames und ausgewertete Trigger-Flanken.

Prüfpunkt: File Stream ist Running, EngineRT empfängt Frames über TCP und SCADA zeigt die passenden MQTT-Ereignisse FIRE und CLEARED.

Fehlersuche beim frühesten verletzten Vertrag

Beginnen Sie bei der ersten Stufe, die ihre Ausgabe nicht nachweisen kann. Änderungen am Schwellwert können keine unpassende Retina reparieren, und ein Neustart von Studio kann keine gestoppte Quelle wiederherstellen.

Symptom	Zuerst prüfen
UDM-Input ist leer oder enthält falsche Spalten	UDTAT-Rollen, numerische Ausgabe und den Vertrag des gelöschten <code>sensor_15</code>
Heatmap ist leer oder kollabiert	UDM-Merkmalsauswahl, Anzahl der Eingabezeilen und Abschluss des Trainings
Retina ist leer oder Signale fehlen	Gebinnte Daten, Signalsatz der Map, Rezept und 64-x-64-Dimensionen
Das erste Studio-Rendering ist leer	Render Project erneut wählen
Overlap-Kurven fehlen	Overlap Pane sichtbar lassen und Render Project erneut wählen
Fehler-Fingerprint ist verrauscht oder gemischt	Einen überprüfen Datensatz oder eine vollständige zusammenhängende Phase ohne gemischte Betriebszustände auswählen

Symptom	Zuerst prüfen
Overlap ist sichtbar, aber kein Alarm wird ausgelöst	Eintrag aufklappen und Enable trigger , Dichte und Schwellwert prüfen
Lokaler Klickton ertönt, aber SCADA empfängt nichts	Globale MQTT-/REST-Veröffentlichung, Endpunkt, Empfängerzustand und Transport-Selbsttest
Engine erscheint nicht in der Fleet	Prüfen, ob der lokale EngineRT-Prozess läuft; anschließend warten oder Refresh wählen
Engine bleibt Ready to adopt	Adopt wählen; lokale Adresse nicht manuell hinzufügen
Engine ist funktionsfähig, bleibt aber Waiting	Gatekeeper-Binding, ausgewähltes Paket und Quellendpunkt
Engine ist Offline (license reserved)	Prozess wiederherstellen und Reconnect statt Release verwenden
Engine läuft mit null Frames	Running-Zustand von File Stream, TCP 127.0.0.1:19001 und Frame-Alter
EngineRT weist das Paket zurück	Signaturvertrauen, Paketsignatur und Artefakt-Hashes

Abschließende Abnahme-Checkliste

1. Der Startbildschirm öffnet sich und der ausgewählte Datenstammordner enthält das mitgelieferte Waterpump-Projekt.
2. Studio öffnet `project.sf2project.json` mit der vollständigen Artefaktkette.
3. UDTAT wird aus dem Preferences Pane des geöffneten Projekts gestartet und speichert das von allen nachfolgenden Stufen verwendete Preprocessing-Rezept.
4. UDTAT schließt Metadaten und Status aus den Merkmalen aus, entfernt `sensor_15` und erzeugt ausgerichtete numerische und gebündelte Ausgaben.
5. UDM trainiert aus 220.320 Zeilen und 51 Sensormerkmalen eine belegte 64-x-64-Map, deren Belegung bei der Hover-Prüfung plausibel ist.
6. UFP erzeugt ein passendes zählwertbasiertes 64-x-64-Dictionary mit plausiblen Fingerprints von spärlich bis dicht.
7. Studio rendert die Aufzeichnung; Fit, Zoom, Panning und Lasso funktionieren, während Sensor-, Status-, Overlap-, Library- und Fingerprint-Ansichten sichtbar bleiben.
8. Eine benannte Referenz kann aus einem überprüften Datensatz erfasst oder aus einem vollständigen zusammenhängenden Abschnitt aggregiert werden.
9. Die ausgewählte Referenz wird über Library-Eintrag, Ursprungsmarker, Fingerprint-Farben und Overlap-Kanal hinweg verstanden.
10. Der Anstieg des Overlaps `Failure1` mehr als vier Tage vor dem aufgezeichneten Fehler wird als frühe Ähnlichkeitstrajektorie verstanden.
11. Die Schwellwertauswahl wird anhand realer Fehler und nicht ausfallender Annäherungen validiert und bleibt von der Transportkonfiguration getrennt.
12. Das Engine-Paket besteht die Validierung von Signatur, Hashes, Artefakten und den Smoke-Test mit 220.320 Zeilen.
13. Die Engine Fleet ist zu Beginn leer und erkennt automatisch eine lokale Engine.
14. Die lokale Engine wird mit **Adopt** und ohne **Add by address** in Betrieb genommen und erreicht Running.
15. File Stream liefert TCP-Frames auf 127.0.0.1:19001.
16. Der verwaltete SCADA-Broker empfängt den MQTT-Alarm der Engine auf 127.0.0.1:1883.
17. Die Frame- und Trigger-Edge-Zähler von EngineRT steigen an und SCADA zeichnet die passenden Übergänge `FIRE` und `CLEARED` auf.