SenseGate: Distributed and Reliable Monitoring for Flood-Protected Areas

Michel Rottleuthner
Hamburg University of Applied Sciences*
michel.rottleuthner@haw-hamburg.de

Leandro Lanzieri
Hamburg University of Applied Sciences
leandro.lanzieri@haw-hamburg.de

Timon Rupelt
Hamburg University of Applied Sciences
timon.rupelt@haw-hamburg.de

Jan Thies
Hamburg University of Applied Sciences
jan.thies@haw-hamburg.de

Marc Briede
Hamburg Port Authority†
marc.briede@hpa.hamburg.de

Thomas C. Schmidt
Hamburg University of Applied Sciences
t.schmidt@haw-hamburg.de

ABSTRACT

This paper reports about ongoing research for protecting critical infrastructure against floods. Aligned to a real use case in the Port of Hamburg, we present an approach for reliably monitoring floodgates. We analyze existing processes for safe floodgate operation and devise the design of SenseGate, a distributed monitoring system of increased resilience. We describe how the system integrates both autonomous sensors and technicians to digitize the polder status and to trace critical operations. Communication incorporates authenticated messaging and redundant communication links to improve the robustness of the system. Measurement challenges of reed switches and inductive sensors are discussed in the context of balancing energy consumption and accuracy. We propose designs for an energy neutral sensor, a companion device for field technicians, and a dashboard which reports cross-validated information to the overseeing operator. Finally, we systematically analyze how failure modes affect the SenseGate system and how impairments can be prevented.

Keywords

Dependability, IoT Sensing, Flood Protection, Critical Infrastructure

INTRODUCTION AND BACKGROUND

Global warming is expected to increase extreme weather conditions and flooding (Schiermeier 2011), with river floods imposing severe health and natural hazards (Jonkman and Kelman 2005; Ginkel et al. 2021). Cities with high flood risk therefore need robust counter measures to protect citizens and property. In the following, we outline the flood defense infrastructure and related processes in the city of Hamburg as the context for our solution.

The City of Hamburg and its Port

Hamburg is located at the Elbe river, 100 km upstream from the North Sea at 6 m above sea level. The strong tide with an average water level variation of 3.8 m and a history of catastrophic floods made flood risk management an integral part of the city (Guttry and Ratter 2022). In 1962, one of Germany's most severe flood incidents caused 347 deaths (Paprotny et al. 2024) with 315 of them in Hamburg (Jochner et al. 2013), after which the city initiated advancements in flood protection. Despite being designed at time in which future requirements of a digitized world were neither relevant nor applicable, these advancements are in operation today (Guttry and Ratter 2022).

In this work, we focus on digitizing critical infrastructure in the Port of Hamburg and in particular on adding resilience using the Internet of Things (Petersen et al. 2015).

*https://www.haw-hamburg.de
†https://www.hamburg-port-authority.de

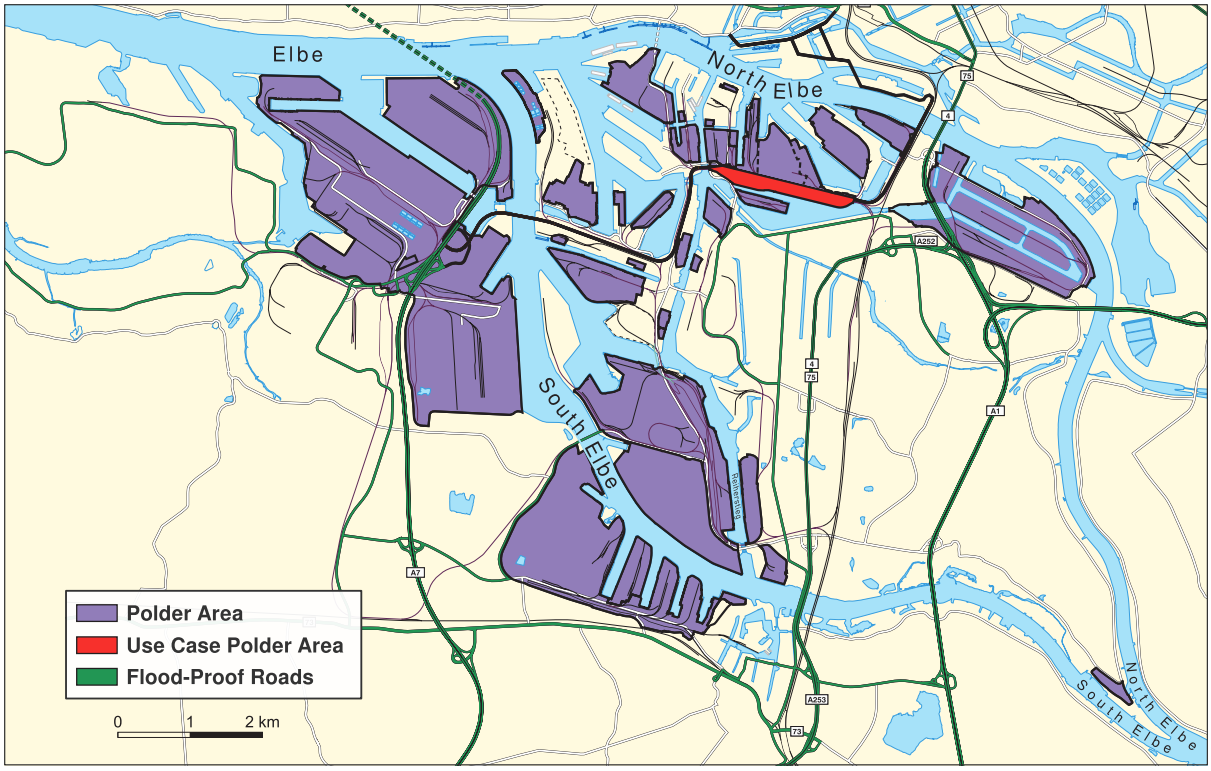

Figure 1. Map of privately operated polder areas in the Port of Hamburg

Polders: Water Barriers for Containing Floods

Hamburg has the largest seaport in Germany, spanning roughly 74 km^2. Without countermeasures, large areas within the port would be directly exposed to rising water levels several times every year. Polders in Hamburg serve as a reinforced and raised water barrier, protecting enclosed areas from floods. They typically host industrial facilities that benefit from nearby waterways but at the same time are exposed to a higher risk of flooding. At safe conditions, gates of various kinds provide direct access to polder areas for pedestrians, cars, and even trains.

Figure 1 displays a map of the privately operated polder areas in the Port of Hamburg. Our use case scenario happens in the polder highlighted in red. Polder areas are handled by *polder communities* (*cf.* Figure 2), which encompass defence, operation, and management of the polders by bringing together the private tenants and the Hamburg Port Authority. We conducted initial experiments and plan on a real-world deployment study in cooperation with the polder operator. This polder spans an area of about 380 000 m^2 and incorporates 27 distinct floodgates.

Safe Operation Requires Accurate Monitoring and Fallback Mechanisms

Nautical and weather forecasts are used by WADI, Hamburg's Port storm surge warning service, to predict arrival times and intensities of storm surges. Whenever predictions indicate critical water levels, threatened areas must be evacuated and polders must be protected by timely closing relevant floodgates to avoid damage and casualties. While closed, technicians regularly inspect gates to maintain a safe state of the polder. A polder-pilot (*i.e.,* head of operations) is responsible for action planning, coordination, and documentation of all operations. Once the polder is protected, the pilot informs the next level authority, Hamburg Port Authority's crisis staff, that all relevant flood defense mechanisms are in place.

The existing floodgates are actuated locally and always require manual intervention for opening or closing. Four different gate designs of various sizes are used. The basic variants are two meters wide swing-doors, locked by hand levers. More complex gates range over ten meters, are rail-guided and weigh several metric tons. Operation requires multiple semi-manual steps for correctly positioning and closing. Figure 3 shows examples of different gate designs in our experimentation polder.

Digitization of Manually Operated Floodgates

Flood protection is a worldwide and widespread issue with solutions trading off site-specific aspects such as costs, reliability, and environmental impact. The spectrum of defence implementations ranges from software-controlled

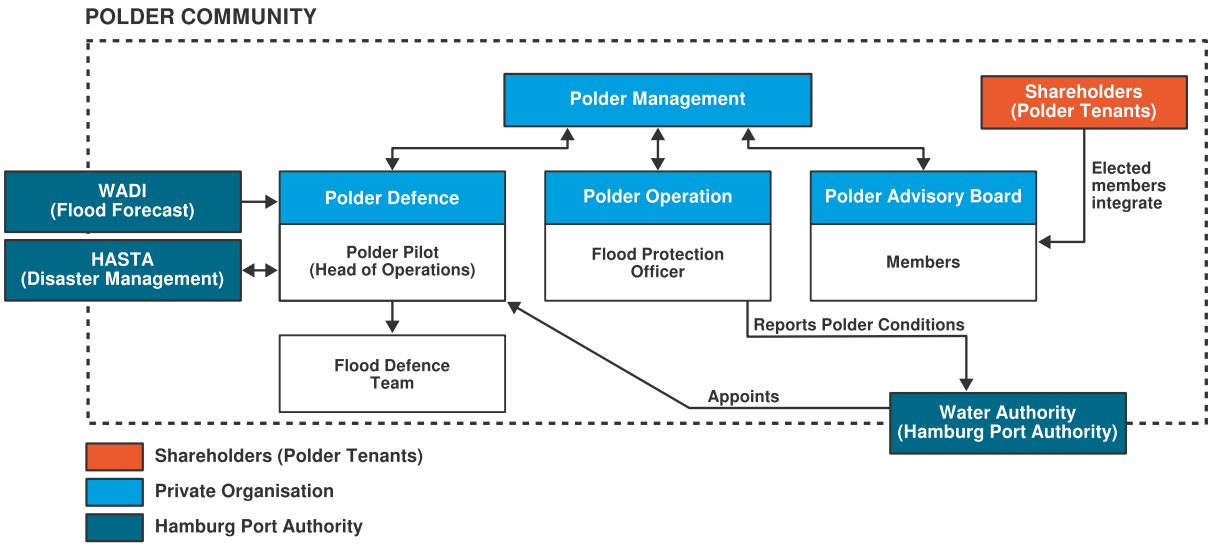

Figure 2. Organizational structure of a polder community in Hamburg.

Figure 3. Hinged floodgates with simple hand levers compared to more complex installations which require guide rails for positioning and geared transmissions for raising and lowering.

autonomous gates, such as the Maeslant storm surge barrier (Tretmans et al. 2001), through partial automation, such as the Thames barrier whose hydraulic system is manually triggered by operators (Kendrick 1988), all the way to manually operated gates, which is our use case. The floodgates in polder areas of Hamburg have been installed decades ago and are expected to operate for additional decades before upgrades to autonomous systems may be considered. Instead, the Hamburg Port Authority has decided to evaluate, as part of the RESCUE-MATE project[1], a digitalization approach based on a retrofit system compatible with the heterogeneous existing infrastructure, which is the scope of this work.

Challenges and Requirements

The various gate mechanisms require an easy-to-adapt sensing system of wide applicability. Due to the size and locations of polder areas, there is rarely access to the power grid or network infrastructure. The developed sensing system should therefore operate wirelessly and on batteries.

Polders are organized in a federated structure where stakeholders and operators are split between public and numerous, mostly private institutions. Distributed on-premise operation of the system is therefore preferred over a centralized single-operator system. The key users for our solution are private polder operators which are responsible for reporting the evacuation and flood protection status of the polder to the sub-unit of the disaster management team of the port (HASTA), which communicates flooding events to civilians and coordinates evacuations with the police. The high-level functional requirements of the proposed system were derived from user stories collected from polder operators and the flood defence team during the initial phase of the RESCUE-MATE project.

As floodgates are often accessible via public spaces, they are potentially subject to vandalism, unauthorized opening, and sabotage, demanding for threat detection and fallback mechanisms.

As clearly identified after the construction of the Maeslant storm surge barrier in the Netherlands, human involvement is crucial during operation to handle unforeseen circumstances, therefore it should be considered in the system design since the beginning (Vrancken et al. 2008). Since the prospective system will still highly rely on human

[1] https://www.rescue-mate.de/

intervention, it should seamlessly augment humans and specifically address challenges related to human errors. The system should then integrate feedback from both technicians and sensors to allow for cross validations of its integrity and the plausibility of observed events. Interactions with the gates should be logged for cross-checking against related events and serve as additional confirmation. Operators should be automatically warned about hazards and mistakes.

With the most critical usage scenario expected during exceptional weather conditions, the system should be made resilient against outages of power, network, and local subsystems.

Related Work

Most flood-related IoT solutions focus on measurements of rainfall, water levels, and flow rates (Bartos et al. 2018; Hussen Hajjaj et al. 2020; Esposito et al. 2022).

A comprehensive survey on IoT-based early warning systems identifies four critical requirements: battery life, fault-tolerance, coverage, and latency (Esposito et al. 2022). The analysis points out a gap for fault-tolerant solutions, which address communication and sensing failures. Our work explicitly address this gap in a real-world scenario, which combines the general requirements with the specific challenges of monitoring floodgates.

The open storm project (Bartos et al. 2018) introduced an open source platform for sensing and control of urban watersheds. A similar system demonstrates monitoring and control for small water locks (Hussen Hajjaj et al. 2020). Unlike those systems, we apply hybrid communication to improve the reliability compared to a single cellular link. We also follow the open source principles but focus on floodgates that need tight integration with human operators.

Optimizing node placement was shown to reduce cost while maintaining observability of water networks (Farahmand et al. 2022). This does not apply to our scenario because all floodgates must be observed separately. However, our system could be similarly optimized by sharing one long-range relay between neighboring nodes.

(Adesina et al. 2025) analyzed the costs and operation of flood sensor networks. The results show a predominance of wireless communication and a high relevance of the component and maintenance cost. Cellular and satellite systems are noted as costly solution for better coverage and resilience. Our approach addresses the need for cost-efficient and reliable systems in two ways: we build our system from commodity hardware, and we employ redundant links to improve communication resilience without expensive data plans.

DISTRIBUTED MONITORING OF POLDER AREAS

System Overview

In this work, we propose SenseGate: a robust approach to improve the reliability and traceability of the polder safety status in the Port of Hamburg. The objective is to augment the workflow of port operatives by automatically reporting the gate status (*i.e.,* closed or open), allowing manual reports by personnel on the field, and providing a traceable overview of the area. The SenseGate system consists of three building blocks: *SenseGate* nodes, *SenseMate* nodes, and a backend application. Figure 4 depicts the overview of the system deployment.

SenseGate Nodes are constrained embedded devices deployed at all polder gates. These devices have inductive and reed switch sensors, the data of which is fused locally to determine the gate status with high confidence. SenseGate nodes generate reports periodically and upon changes.

SenseMate Nodes are the handheld companions that augment field personnel when verifying gate status. The nodes serve three objectives: allowing workers to report the observed status of a gate, communicating tasks assigned by the polder pilot, and relaying messages from nodes to the backend and other nodes.

The **Backend Application** is a central coordination entity that provides the polder pilot with a status overview of the area, based on information collected by the nodes and operators. The backend collects all reports from SenseGate nodes and observations from SenseMate nodes, and estimates the status of all gates, as well as a confidence level on that status through an agreement mechanism.

In current polder defence operations, information flows from polders, through operatives, and to the Hamburg Port Authority mainly sequentially, following a single path, and scoped to single polders. Under this schema, the port authority receives merely asynchronous snapshots of the previous reported status, without any independent verification. As shown in the right-hand side of Figure 5, the introduction of SenseGate allows for a significant shift from sequential to multi-path reporting, mainly thanks to the following key points: i) gates turn into active information sources, and ii) the backend can aggregate and cross-validate polder information and provide it to the port authority continuously, even across polder areas.

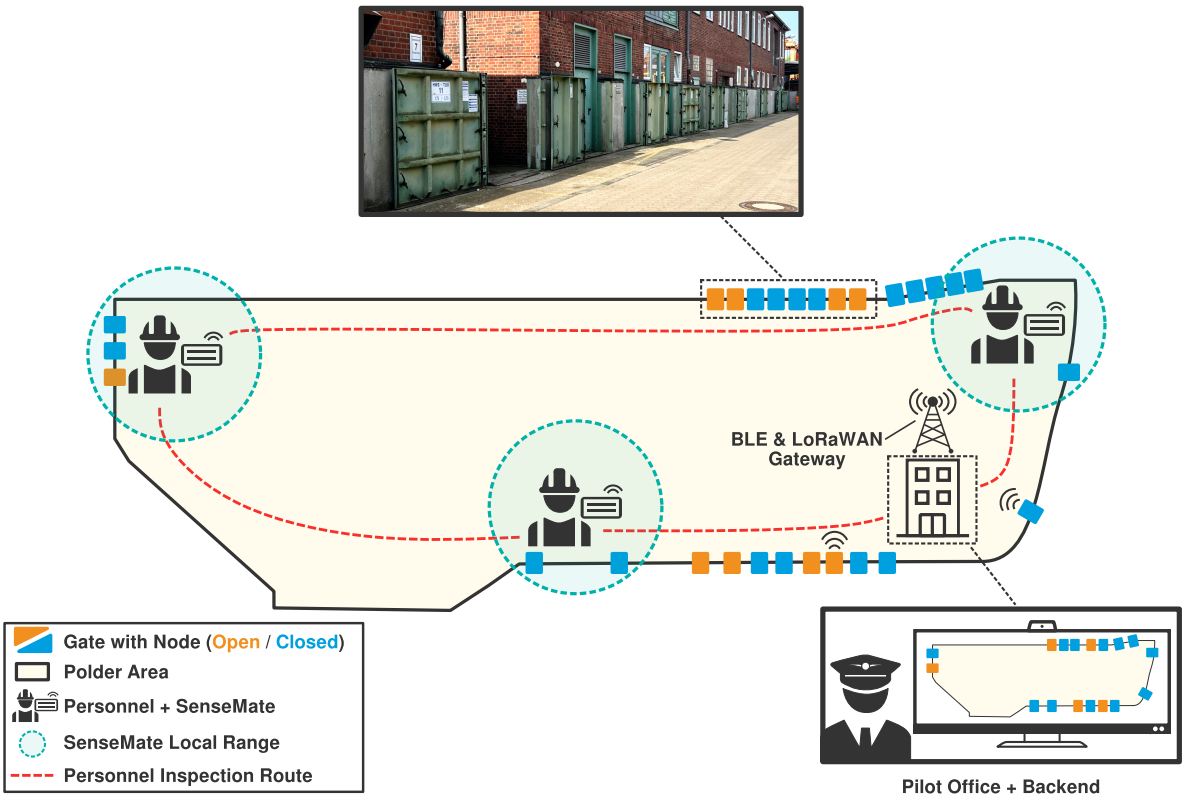

Figure 4. Overview of the SenseGate system deployment.

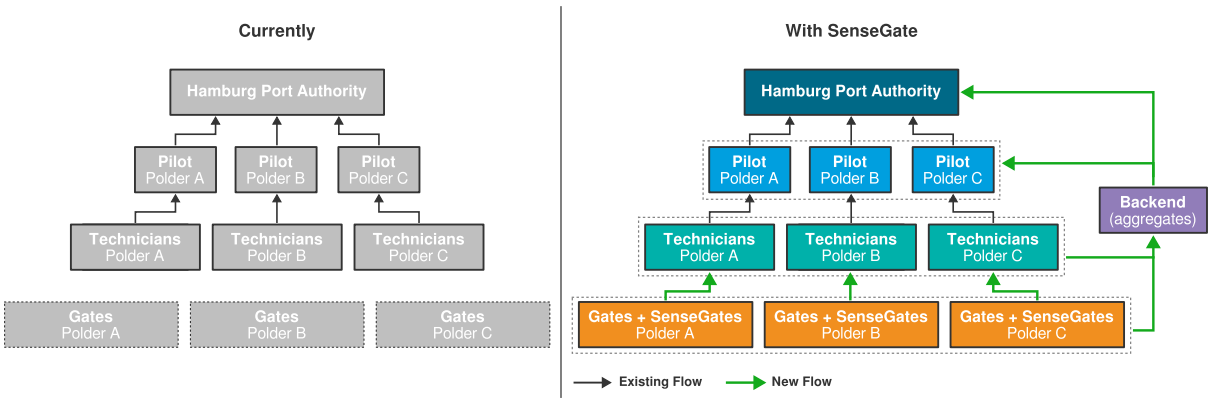

Figure 5. Information flow in polder defence operations: sequential single-path reporting currently (left) and cross-validated multi-path, redundant reporting with SenseGate (right).

Dependable Communication System

In SenseGate, information not only flows from sources to the central backend, but also distributes across nodes in the field. This ensures robustness and dependability, as it reduces the risk of a single point of failure, and allows the system to still operate in a degraded state if subsystems fail. Therefore, the system requires that all nodes acquire a global state that converges over time across nodes — following an eventual consistency.

Since the monitored structures of the SenseGate are part of the critical infrastructure, all communications must comply with security requirements. Sources of all application-level messages must be **authenticated** by the receivers (*e.g.,* gate reports and observations). As nodes relay messages from other sources, the protocol must ensure **non-repudiation**, which allows receivers of relayed messages to authenticate the source and verify the content is untampered.

Link Technologies

Without access to power and wired networking, SenseGate and SenseMate nodes must rely solely on low-power wireless communication. Specifically, the devices utilize two link technologies: Bluetooth Low Energy (BLE) for short-range communication between nodes, and LoRaWAN for long-range communication with the backend (Bluetooth Special Interest Group 2025; LoRa Alliance 2016).

LoRaWAN utilizes unlicensed sub-GHz radio bands and offers communication ranges between 2 km and 5 km under low energy requirements but with limited data rate (up to 21 kbit/s), low payload size (up to 222 B), and a latency in the order of seconds. The network has a star-of-stars topology, where end devices communicate with a server via transparent gateways. In the lowest energy setting (class A), downlink messages from the server can only be sent immediately after an uplink, which eliminates the need for continuous listening. LoRaWAN messages are end-to-end encrypted using AES-128, and devices are authenticated using pre-shared keys. A noteworthy limitation of LoRaWAN is the duty cycle compliance, which determines the maximum time on air for each radio, limiting the number of downlinks a gateway can send.

The BLE specification defines advertising as a connectionless low-power communication that allows devices to broadcast messages to scanners in ranges of several tens of meters. Unlike LoRaWAN, BLE advertisements provide data rates of up to 2 Mbit/s, allowing payloads of up to 1.6 kB. BLE advertisements provide neither encryption nor authentication methods built in.

Message Authentication

To ensure non-repudiation for unauthenticated BLE advertisements, our nodes cryptographically sign all generated messages. With the signature, all receivers can verify that the information originates from the device indicated in the message. Nodes create signatures in accordance to the CBOR object signing and encryption (COSE) standard (Schaad 2022), which defines efficient encoding of cryptographic signatures for constrained environments. It is worth noting that adding signatures does not provide confidentiality to the communication, but this is not a system requirement.

All messages sent via a BLE or relayed via a proxy include an Edwards-curve digital signature algorithm (EdDSA) signature of the originator. To authenticate a received message, the receiver has to verify the signature against the public key of the originator. Therefore, all devices require the public keys of all other devices in the network. In the current state, all devices ship with the required public keys, but the final solution will dynamically distribute signed public keys via C509 certificates.

As the LoRaWAN link provides built-in message authentication, devices do not include signatures over this link when sending messages generated by themselves. In contrast, when a device relays a message generated by a different device via LoRaWAN, the signature must be included for the backend to trust the message.

Application Messages

We define four types of application-specific messages: gate report (GR), gate observation (GO), gate command (GC), and gate job (GJ). All messages are encoded using concise binary object representation (CBOR), which re-utilizes the encoding and parsing logic that is already used for COSE. GRs are generated by each SenseGate node, and comprise a report of the sensed status of the monitored gate. GOs are generated by field personnel on their SenseMate upon inspection of a gate, and contain the observed status. Observations serve as a human confirmation of the gate status towards the pilot. GCs and GJs are produced by the pilot to communicate the intended status of a gate (*e.g., "gate 11 should be closed"*), and describe tasks to be carried out by field personnel. Unlike a GC, which can be executed by any worker, GJs specify a person that should execute the task.

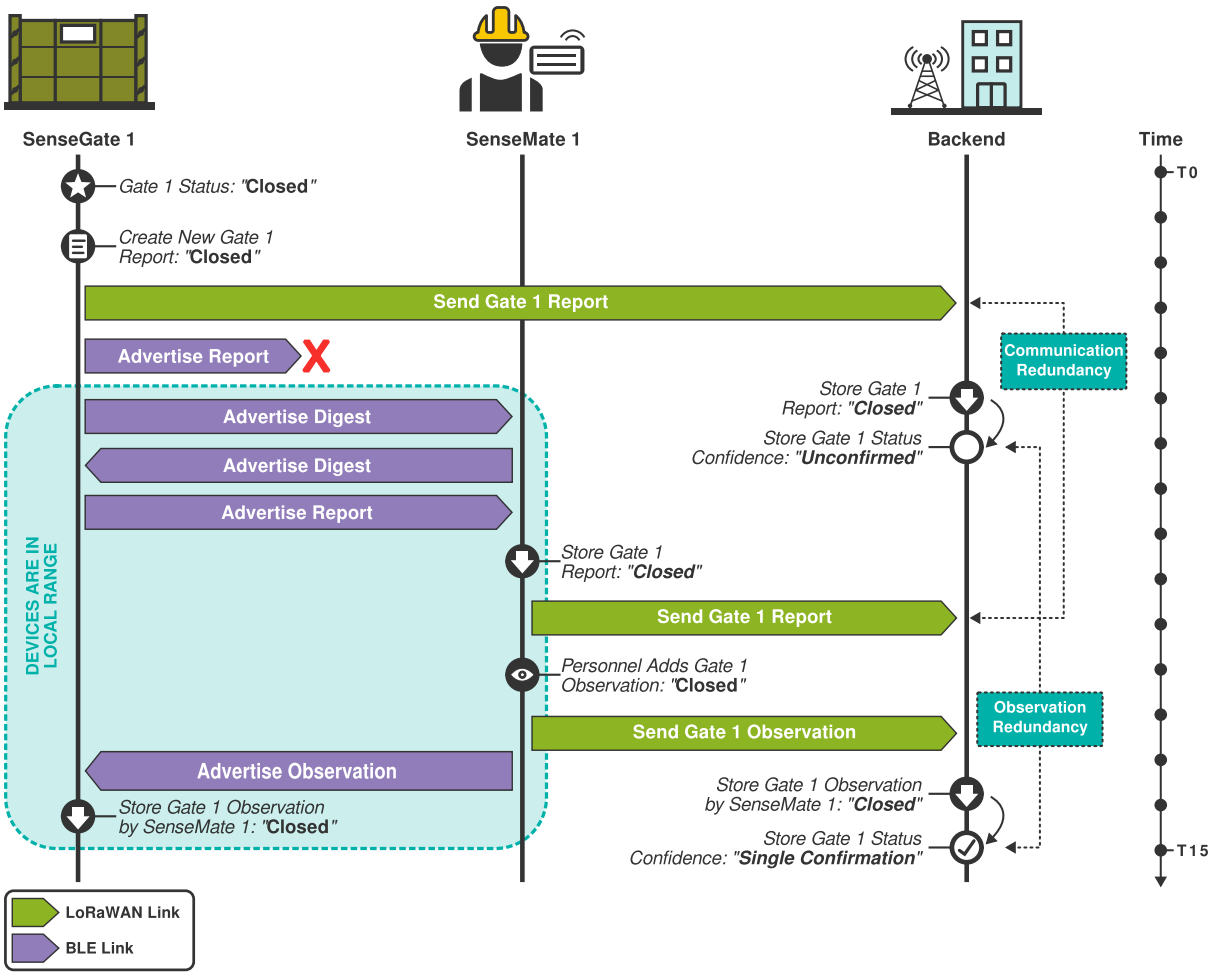

Figure 6. Sequence of an encounter between a SenseGate and a SenseMate.

Devices require timestamps to identify the freshness of messages. Given the constrained network, a perfect clock synchronization between devices is infeasible. Instead, we apply the hybrid logical clock (HLC) algorithm to attach timestamps to messages based on a local physical time and a logical clock (Kulkarni et al. 2014). The method allows to logically order events that are generated on different nodes of a distributed system, since it provides one-way causality information between events: if the event e happened before $f \Rightarrow \text{hlc.e} < \text{hlc.f}$. This guarantees that: i) events timestamped on the same node will be correctly ordered even if the wall clock is unstable, and ii) after a node A sends a message with events to node B, all events created in node B will be ordered after those sent from A.

Protocol for Message Distribution

The system desires that nodes in the field and the backend jointly converge on a common view of reality as close as possible. This contributes to the resilience and facilitates local operation even if the backend is not accessible. To this end, nodes communicate among themselves via the BLE interface when they come into local range, sharing fresher information that was locally produced or collected along the way.

The dissemination system for messages combines two different concepts, epidemic routing and data mules (Vahdat, Becker, et al. 2000; Shah et al. 2003). In these solutions, devices exploit the physical mobility of certain nodes to create ad-hoc networks and pass on data for delivery to a third party that is currently out of reach. Unlike those approaches, though, messages passed around in the SenseGate system do not have a particular final destination, the intention is rather to spread them among all participants.

The logic of the message exchange between devices is as follows. Whenever a node produces new information locally (*e.g.,* a SenseGate generates a GR), it first sends the corresponding message to the backend via LoRaWAN. Additionally, the message is broadcasted via BLE to the devices within the local range. Hereafter, a node will periodically advertise a digest message using BLE, which not only signals to other devices that the node is in the vicinity, but also summarizes the current information state of the node. Upon the reception of a digest, nodes

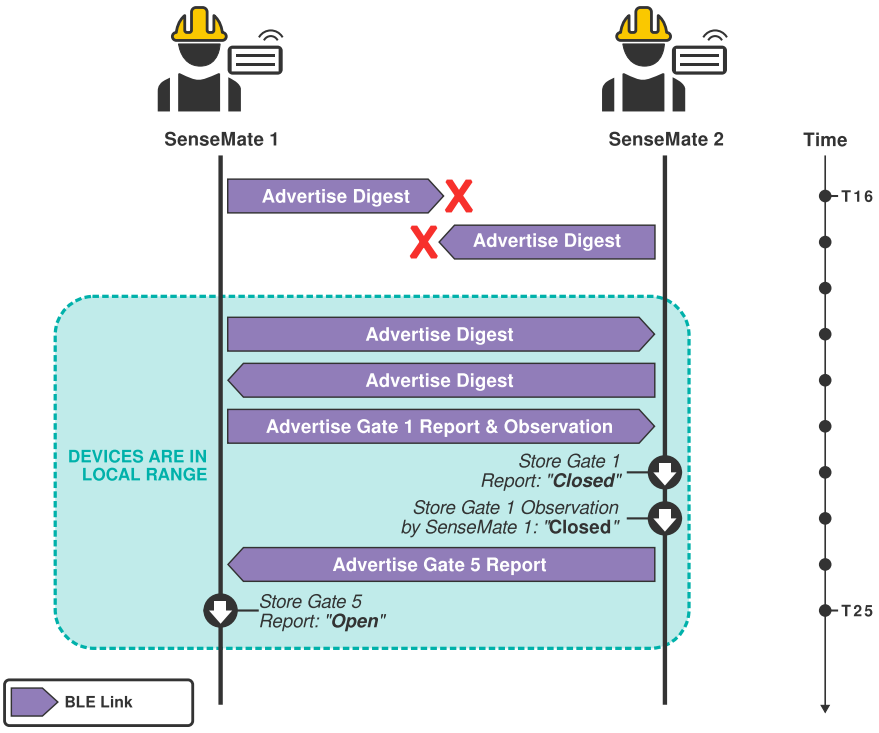

Figure 7. Sequence of an encounter between two SenseMates.

decide whether the sender is lacking fresh information that exists locally. In that case, they proceed to compose and broadcast a message containing the missing information.

In order to update the local information, nodes execute a merging procedure when receiving messages. First, the signatures are extracted and verified against the message author (which can differ from the node that relayed the message). Second, if the message is valid, the node verifies that the attached HLC timestamp is newer than the timestamp of the last equivalent information locally. For instance, upon reception of a GO from SenseMate N with the status of Gate M, the node will only merge it if the last received corresponding GO is older. After successfully merging a message, the local HLC value may be updated according to the HLC algorithm (Kulkarni et al. 2014).

To illustrate the mechanism of data distribution, Figure 6 shows an interaction between the backend, a SenseGate, and a SenseMate. At time T0, SenseGate 1 measures a change in Gate 1 and determines that it is now closed. The node creates a GR, which is first sent to the backend via LoRaWAN, and then via BLE to any nearby nodes (none in this case). The backend receives the gate status and updates it locally, marking it as *Unconfirmed*, meaning that no SenseMate has sent an observation of the gate status since it last changed. At some point, SenseMate 1 comes into reach of SenseGate 1, and they exchange digest messages. After comparing the digest of SenseMate 1 with the local, SenseGate 1 determines that it should send its last GR, which is missing in the neighbour. Having received the report, SenseMate 1 deems it new information, stores it, and sends it via LoRaWAN to the backend. Sending the gate report to the backend from two devices constitutes communication redundancy, which increases system reliability. The personnel in the vicinity of Gate 1 proceeds to indicate on its device that the gate is closed, triggering a local creation of a GO. The SenseMate 1 then sends this observation to the backend, and advertises it locally. With the received GO confirming the current status of Gate 1, the backend now increases the confidence to *Single Confirmation*. Figure 7 illustrates a subsequent encounter between two technicians carrying SenseMates. Upon coming into range, devices exchange digests and relay the missing information to each other: SenseMate 1 sends the GR and GO from Gate 1, and SenseMate 2 provides the GR from Gate 5 that SenseMate 1 had not received.

Gate Sensing

We combine two contactless methods for sensing the state of the floodgates, reed switches and inductive sensors, to operate on low energy but high accuracy. In this work, we focus on the general characteristics of both sensor types and the underlying measurement problem for a single axis of movement.

To assess whether a gate is fully closed, we monitor the distance between the sensor at a fixed location and a reference point on the moveable part of the gate. We select a reference point, which moves along a single axis

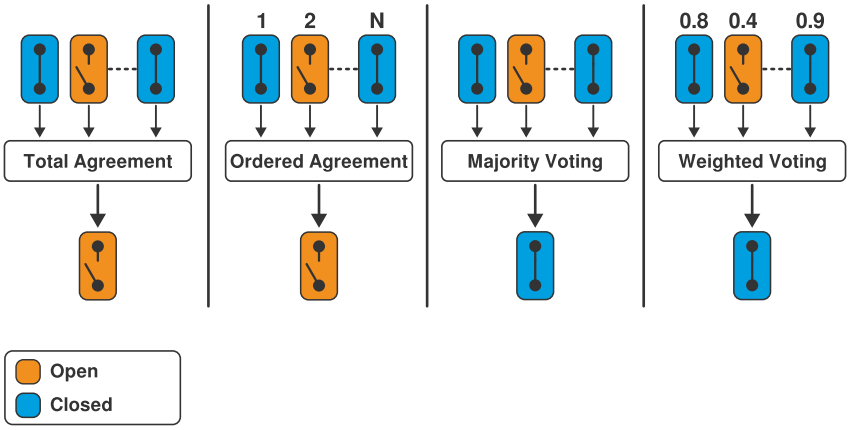

Figure 8. Simple sensor fusion approaches for interpreting multiple inconsistent readings.

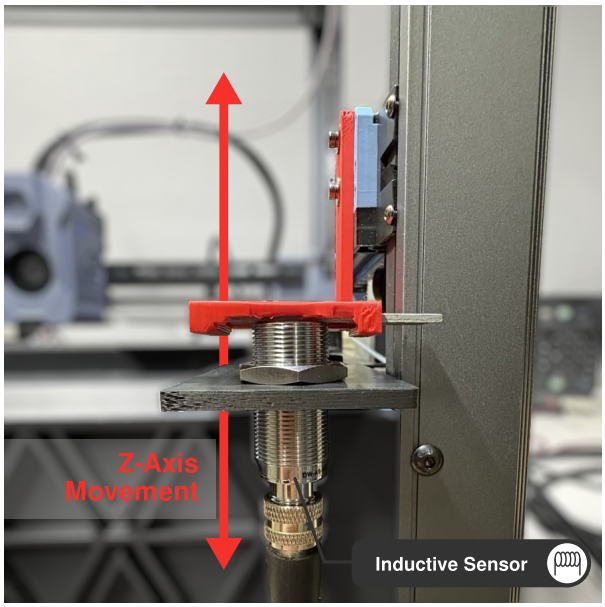

Figure 9. Automated response measurement of an inductive sensor with a custom hardware-in-the-loop setup based on a 3D printer.

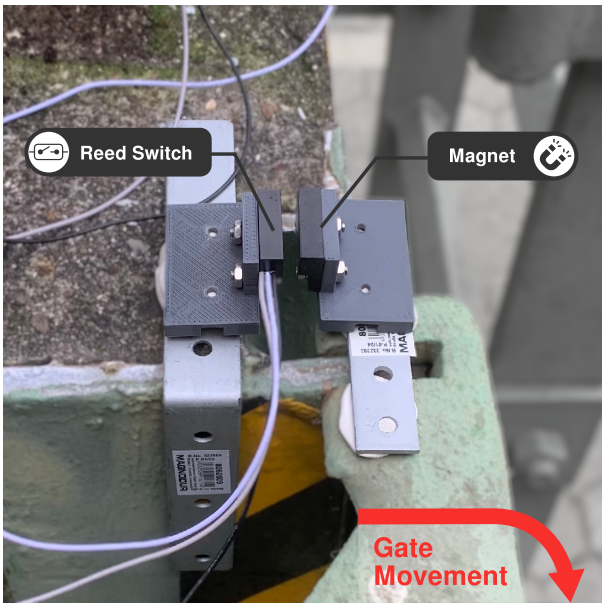

Figure 10. Reed switch with magnet counterpart mounted on a floodgate.

towards the final closing position. For precise characterization of the sensors we present our setup for reproducible sensor experiments in a controlled environment.

Figure 8 illustrates four simple methods to combine multiple sensor readings to improve either the redundancy or strictness of the measurement. For total agreement all sensor values must indicate the same gate state. Ordered agreement requires sensors to activate in the correct order at different offset thresholds during the closing procedure. In both cases, no state change can be asserted if one sensor mismatches. Majority voting (Khaleghi et al. 2013) interprets all sensors with the same significance but is less sensitive to mismatches. A majority threshold dictates the allowed number of conflicting sensors (*e.g.,* 2 out of 3). Weighted voting assigns a weight to each sensor, representing a sensor-specific trust value based on past decisions and behavior over time. State determination is not affected by malfunctioning sensors with lower weights.

Despite applying majority voting to balance between strictness and flexibility, we note that sensor mismatches should still be communicated to the operators to detect failing components and restore the original redundancy.

Sensor Characterization with Lab Experiments

We calibrate the system to the sensor responses with an automated hardware-in-the-loop setup based on a 3D printer. Figure 9 shows the lab setup with a rigidly mounted inductive sensor at the bottom. A bracket attached to the printer

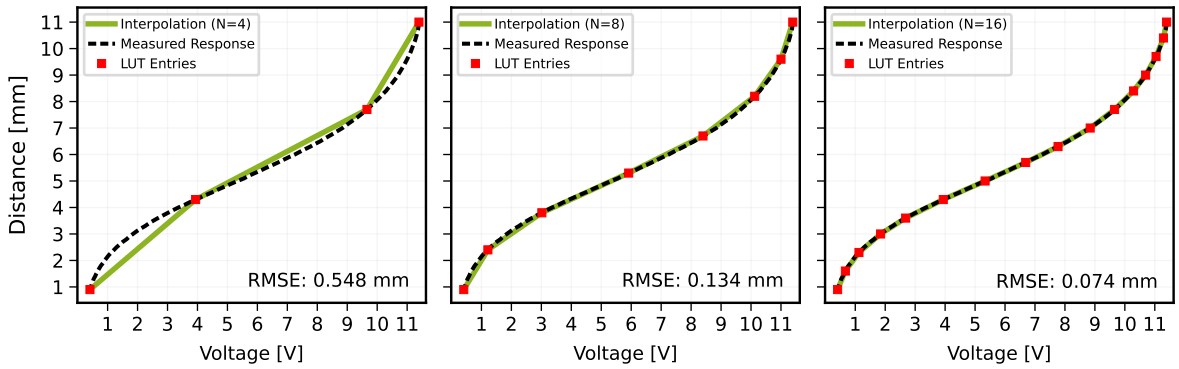

Figure 11. Encoding the response curve of the inductive sensor as lookup-table (LUT) and linear interpolation between LUT entries and the resulting root mean square error (RMSE).

Z-axis holds a metal piece to sense. A python script controls the system by sending G-code instructions to the 3D printer for moving to given positions. We sweep the offset in 0.1 mm steps across the entire measurement range of the sensor and simultaneously record measurements.

Reed Switches for Low-Power Sensing

Reed switches are passive electromagnetic switches that open or close when applying or removing an external magnetic field. In our context, the most relevant advantages of reed switches are the virtually zero power consumption, the very fast response time when used as interrupt-driven proximity sensor, the longevity, and a very low cost. However, reed switches only output a digital on/off-signal when their magnetic counterpart enters or leaves a certain range and are therefore unable to precisely determine a distance value. Figure 10 shows a reed switch mounted to a concrete surface on the polder wall and the magnet counterpart mounted to the moveable part of the floodgate.

Inductive Sensing for Precision

To counter the shortcomings of reed switches, we include inductive sensing for more fine-grained measurements, which utilizes the interaction between an electromagnetic field and metallic objects. More specifically, we evaluate a sensor that outputs an analog signal proportional to the distance to a metal object.

Figure 11 shows the response curve obtained from our automated experiment setup, and approximations using a look-up-table (LUT) of varied sizes. Values between LUT entries use simple linear interpolation.

The relatively high-power consumption of the inductive sensor (≈280 mW) strictly demands duty-cycled operation to maintain long battery life. Measurement settings will affect the response time, measurement accuracy, and energy consumption. The sensor specification states a stabilization time of 20 ms after power up. Therefore, after the mandatory startup delay, a few consecutive samples noticeably reduce the error without significantly increasing energy consumption.

Figure 12 relates measurement error to different oversampling configurations and energy consumption. We select a configuration of five samples per measurement, resulting in a favorable balance between measurement error and consumed energy.

Hardware Design

The hardware design of both SenseMate and SenseGate nodes utilizes commercial-off-the-shelf (COTS) components, which reduces production costs. Figure 13 shows the main components of the SenseMate prototype. It consists of a Nordic nRF52840 ARM Cortex-M4 microcontroller with a 2.4 GHz transceiver, a power delivery and battery charging circuit, and the RFM95W LoRa radio with a flexible antenna. In addition, SenseMates include a 0.96” OLED display to show information about nearby gates, a piezo buzzer to generate audible alerts, and a thumbwheel that allows technicians to input status reports.

The stationary SenseGate (see Figure 14) is designed for long-term outdoor deployment. A waterproof enclosure protects the internals from harsh weather conditions. Internally, the hardware is mostly identical to the SenseMate, but the power subsystem supports long-term operation and sensors with higher power demand. Energy harvesting from an 80 mm × 80 mm solar panel charges a 2500 mA h lithium cell for energy-neutral operation. A power-gated DC-DC (Direct Current) converter boosts the battery voltage up to 20 V for the inductive sensor. Sensing peripherals are externally connected via cables, so they can be flexibly mounted at their designated locations.

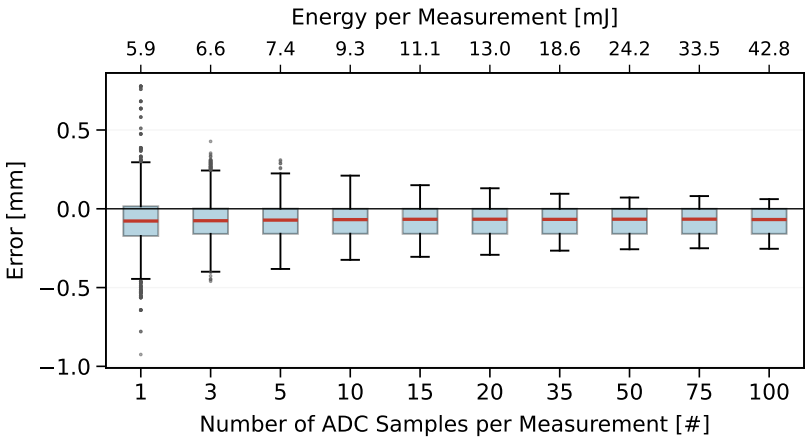

Figure 12. Error of the distance measurement of the inductive sensor versus number of averaged ADC samples and the consumed energy per measurement. Measurement range 1 – 10 mm. Boxes span the IQR (inter quartile range) from Q1 to Q3, box markers represent the median value, whiskers span from Q1 - 1.5 × IQR to Q3 + 1.5 × IQR, points mark outliers.

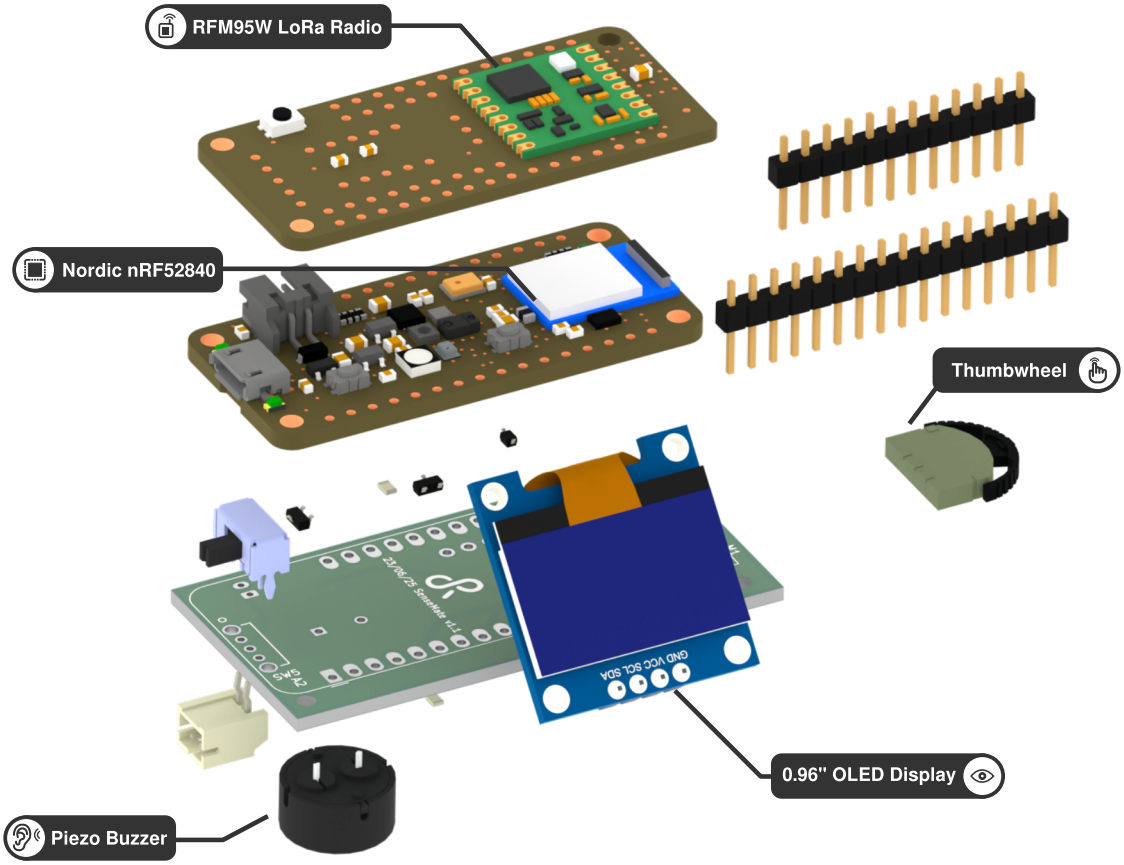

Figure 13. Detail of SenseMate hardware components.

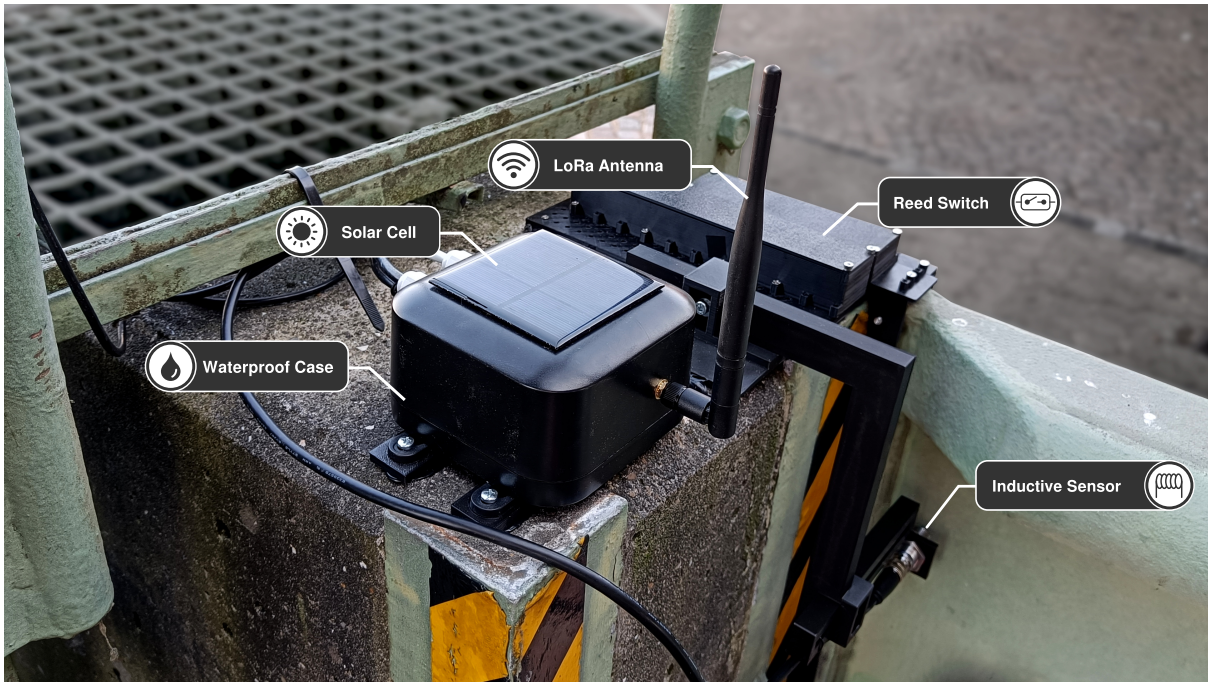

Figure 14. Field test deployment of the SenseGate node on a manually operated floodgate. The waterproof node is equipped with solar power, an external antenna, and sensors.

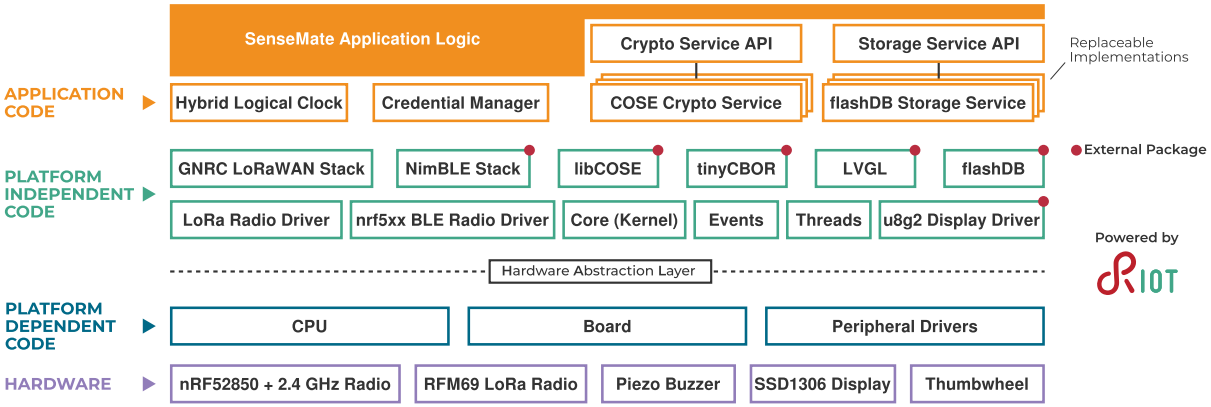

Figure 15. Illustration of the software stack for SenseMate devices.

Firmware

Given the networking requirements and the constrained nature of the hardware, the firmware of the nodes is implemented using RIOT (Baccelli et al. 2018). This open source embedded OS provides most of the building blocks required by the nodes, including a LoRaWAN stack, integration of the mynewt nimBLE BLE stack, and peripheral drivers. Figure 15 details the software architecture of the SenseMate nodes.

Backend Application

The backend application serves as the main interface between the polder pilot and the SenseGate system. It is in charge of collecting all gate reports and observations generated by nodes and communicating the current status of the polder to the pilot. Additionally, the pilot utilizes the backend to create GC and GJ to order technicians to verify or change the status of a gate.

As depicted in Figure 16, the backend is directly connected to the LoRaWAN application server, thus receives all uplinks from the nodes. Moreover, it has a BLE connector that allows to exchange information with SenseMates returning to the pilot office – a high-latency communication backup.

To confidently determine the status of a gate, the system implements a multi-source validation in its confidence logic. The backend receives data from multiple sources (*i.e.,* the gate and technicians), which is fused into a status

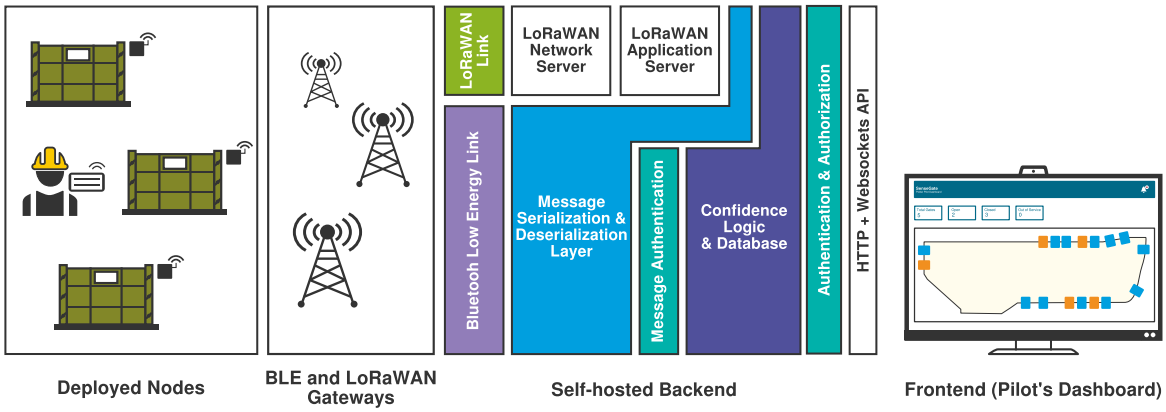

Figure 16. Main logic blocks of the backend architecture from edge nodes to pilot's frontend.

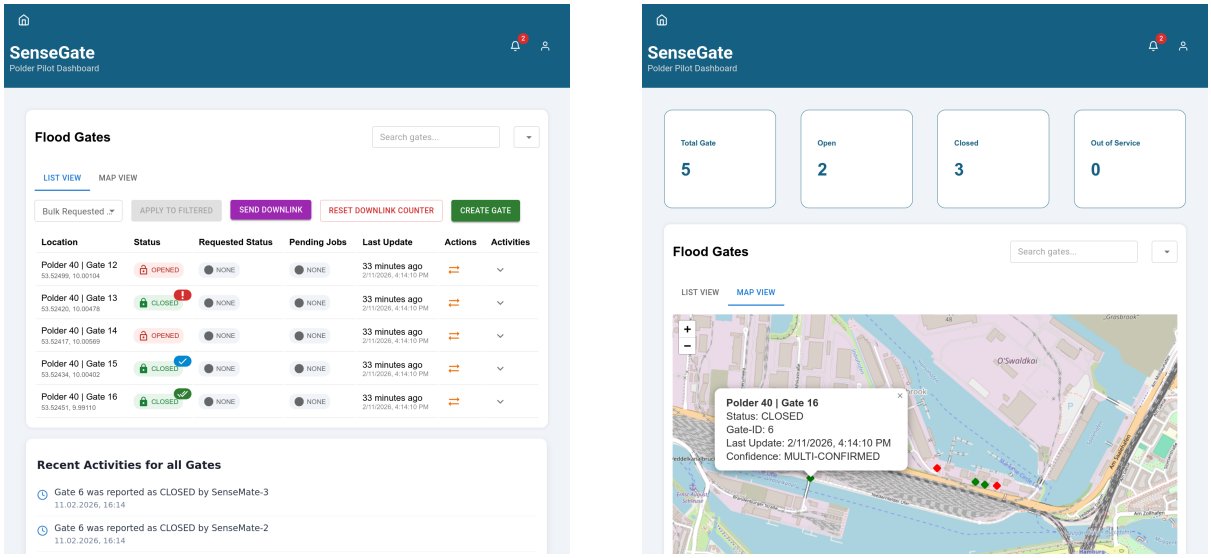

Figure 17. The SenseGate dashboard showing the status overview (left) and gate location map (right).

and a confidence level. The confidence of the reported status can be: i) Unconfirmed, ii) Confirmed, iii) Multiply Confirmed, and iv) Contradicting, depending on the observations issued by technicians on their SenseMate nodes, and the elapsed time since the last gate report. The backend keeps a count on the amount of observations that confirm or contradict the reported status by the gate, and this information is reported to the polder pilot on its dashboard.

The backend currently presents an HTTP RESTful application programming interface (API) to expose the available information and as entry point for pilot requests. Although at present the only client for this interface is the Pilot's dashboard, the backend could be extended to implement standard APIs, such as SensorThings API (Hertweck et al. 2019), to contribute to data reuse across Hamburg.

Polder Pilot Dashboard

The dashboard provides information to the pilot via two views, a status table and an overview map (see Figure 17). Each entry in the status tale represents the status of one floodgate and related metadata. Confidence values are displayed as small badges on the status field. A red badge with an exclamation mark visually emphasizes conflicting information which supports the pilot with prioritizing tasks. The table view is also used for assigning GC, and GJ to specific gates.

With the map view the pilot can observe the current situation via geolocated markers. This map enables easier short-term planning of polder accessibility from various pathways which are road accessible. Especially when communicating with external forces it is useful to relate gate closures to road names and geographical features. For example, when coordinating last-minute passage of motorized vehicles for evacuations or emergencies.

FAILURE MODE AND AVAILABILITY ANALYSIS

To ensure system reliability, we applied an evaluation technique inspired by failure mode, effects, and criticality analysis (FMEA) (Borgovini et al. 1993). Given a set of system properties required for successful operation, the technique examines potential failure modes and determines their effects by propagating the failure through the system. In this section, we analyze how critical system properties are threatened by failure modes of the principal components, and how the system design prevents or mitigates the effects.

Based on the system requirements, we define the following properties that are required for the system to operate:

(P1) Low Latency. Status updates are timely forwarded between nodes and the pilot.

(P2) Status Overview. The overall polder secured status is determined by the system.

(P3) Multi-source Validation. The gate status is cross-validated via multiple sources of information.

(P4) Traceability. Performed actions and occurred events can be analyzed in post-mission reviews.

(P5) Gate Status Alert. The operators are informed about unwanted states and potential errors.

(P6) Message Integrity. Transferred messages cannot be spoofed.

Table 1 describes the considered failure modes. For each failure, we specify the affected system property, its effects, and the mitigation strategy.

PRELIMINARY SYSTEM EVALUATIONS

The SenseGate system has so far been evaluated under two scenarios: a long-term deployment of gate sensors directly on the polder area, and as a scale model simulating multiple gates and operatives. As shown in Figure 14, the scenario of long-term deployments involves the installation of gate nodes on the actual gates of the use case polder, which are equipped with both reed switch and inductive sensors. The main objectives of the experiment are: evaluating the efficacy of the sensing mechanism, verifying the reliability of the long-range communication, and testing the resilience of the node enclosure. During the test, we record all reports generated by the gate nodes and are able to compare them with the gate status recorded by technicians, which are used as ground truth. In addition to the operative data, gate nodes record and transmit telemetry data about the sensors, allowing to evaluate the induction sensors in the real-world application. Figure 18 shows a deployment experiment for assessing how well the inductive sensor distinguishes between the closing mechanism being loosely engaged (left) and fully engaged (right). The system consistently reported a measurement delta of 4 mm between both positions. With the previous accuracy evaluation (Figure 12) we conclude that the measurement reliably indicates the final closing position.

For an initial integration test of the SenseGate system, a scale model of the gates was built and fitted with the reed switches and gate nodes. Modular and mobile, the model gates allow dynamically simulating deployment scenarios that also include SenseMate nodes to test complex communication scenarios under multiple physical distributions. For this evaluation, the server is deployed to keep track of the emulated gate activity and the generated observations from the SenseMate nodes carried around the model area. To validate the system design and collect expert feedback from stakeholders and practitioners, the model scenario has been demonstrated during the city-wide drill for crisis management of HASTA, Hamburg Port Authority's disaster management team, as depicted in Figure 19.

Figure 18. Deployment test of the inductive sensor to correctly detect the final closing positions of the gate.

Table 1. Summary of failure modes, their effects, and mitigations applied by the system.

Failure Mode	#	Affected Property	Failure Effects	Mitigation / Prevention
LoRaWAN infrastructure unavailable.	F1	Low Latency (P1)	No long-range link between backend and nodes are operational.	Multi-hop BLE provides fallback connectivity.
LoRaWAN interface of a node is inoperative.	F2	Low Latency (P1)	Affected node has no direct long-range link to the backend.	Proxied communication via BLE or other nodes LoRaWAN provides fallback connectivity.
A SenseGate or its sensing peripheral are inoperative.	F3	Status Overview (P2)	The affected SenseGate cannot report its status.	SenseMates send gate observations to the backend.
	F4	Multi-source Validation (P3)	The affected SenseGate cannot contribute to its status validation.	Observations by multiple SenseMates are used for validation.
	F5	Gate Status Alert (P5)	The affected node cannot trigger alerts on gate state changes.	Gate observations by SenseMates are used as fallback.
Backend is inoperative.	F6	Status Overview (P2)	The pilot cannot access the status overview via the dashboard.	A (high-latency update) status overview is accessible locally via a SenseMate.
	F7	Multi-source Validation (P3)	The backend cannot perform gate status validation nor display status.	SenseMates can validate gate status locally.
	F8	Traceability (P4)	The backend cannot be used for logging nor confirmations.	An event log can be extracted from the replicated data on SenseMates.
	F9	Gate Status Alert (P5)	The backend cannot communicate alerts to the pilot.	All nodes replicated the expected status and locally generate alerts on detected mismatches.
A SenseMate is inoperative.	F10	Status Overview (P2)	The affected SenseMate does not contribute to the status overview.	Other SenseMates and SenseGates still contribute to the status overview.
	F11	Multi-source Validation (P3)	The affected SenseMate cannot contribute to gate status validations nor proxy messages.	Other SenseMates can act as fallback.
	F12	Traceability (P4)	The affected SenseMate cannot contribute to the trace.	SenseGate and other SenseMates still report events.
A SenseGate cannot confirm the presence of a SenseMate.	F13	Traceability (P4)	Encounters between the SenseGates and SenseMates cannot be verified directly.	SenseMates still log gate observations, and nearby SenseGates can vouch for the presence of the SenseMate.
The sensing peripheral of a SenseGate is inoperative.	F14	Traceability (P4)	The input of the affected sensor cannot contribute to the trace.	Multi-source validation is used as fallback.
An attacker impersonates a node or the backend and sends messages.	F15	Message Integrity (P6)	There are messages with information controlled by the attacker in the network.	All nodes verify signatures and timestamps of received messages. Signature verification fails.
An attacker replays messages.	F16	Message Integrity (P6)	Old messages are sent by the attacker.	Old messages are discarded.
An attacker with physical access obtains private keys of a node.	F17	Message Integrity (P6)	The attacker impersonates the node by sending signed messages.	A tamper detection system erases keys when the enclosure is opened.
An attacker with access to a copy of the firmware obtains private keys of a node.	F18	Message Integrity (P6)	The attacker impersonates the node by sending signed messages.	To avoid obtaining keys from a firmware copy, nodes create keys locally during provisioning.

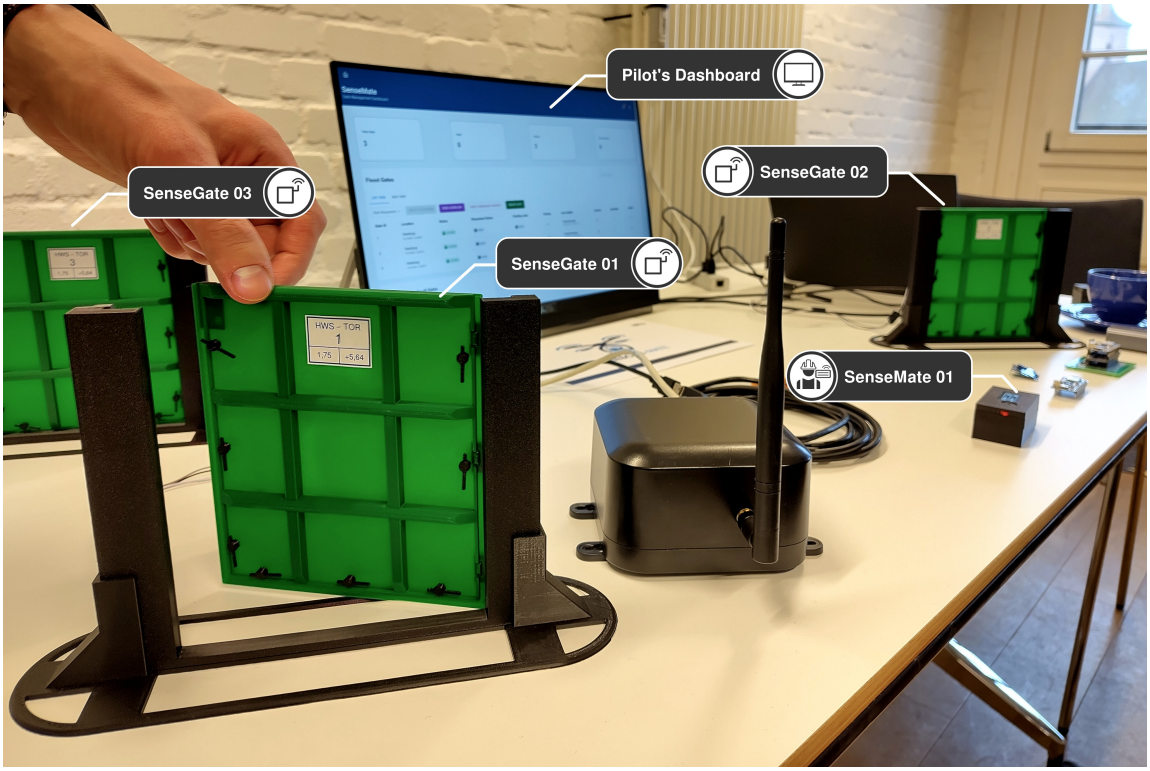

Figure 19. Demonstrator for interactive experiments with multiple SenseMate and SenseGate nodes.

CONCLUSION AND FUTURE WORK

In this paper, we introduced SenseGate, a distributed approach for reliable floodgate monitoring in polder areas. SenseGate digitizes the status of floodgates with wireless sensors and augments field technicians with a networked companion device. We showed that combining reed switches and inductive distance sensors allows for accurate measurements under tight energy constraints. The system integrates protective security, redundancy, and cross validation. Its dependability was confirmed by a comprehensive failure mode analysis.

In future work, we aim for extending the system with a smartphone app, dynamic credential management, over-the-air updates, and by deploying more sensors.

Reproducibility

We support reproducible research and open source development. All artifacts, including firmware and hardware design files are publicly available[2].

REFERENCES

Adesina, M., Brake, N., and Hariri Asli, H. (Oct. 2025). "A survey of flood warning sensor network operational and maintenance practices across the United States". In: *Developments in the Built Environment* 23, p. 100689.

Baccelli, E., Gündogan, C., Hahm, O., Kietzmann, P., Lenders, M., Petersen, H., Schleiser, K., Schmidt, T. C., and Wählisch, M. (Dec. 2018). "RIOT: an Open Source Operating System for Low-end Embedded Devices in the IoT". In: *IEEE Internet of Things Journal* 5.6, pp. 4428–4440.

Bartos, M., Wong, B., and Kerkez, B. (2018). "Open storm: a complete framework for sensing and control of urban watersheds". In: *Environmental Science: Water Research & Technology* 4.3, pp. 346–358.

Bluetooth Special Interest Group (2025). *Bluetooth Core Specification v6.2*. URL: https://www.bluetooth.com/specifications/specs/core-specification-6-2/ (visited on 03/22/2026).

Borgovini, R., Pemberton, S., and Michael, R. (1993). *Failure Mode, Effects and Criticality Analysis (FMECA)*. Tech. rep. Reliability Analysis Center, Department of Defense.

[2] https://git.inet.haw-hamburg.de/inetrg/rescuemate-sensing

Esposito, M., Palma, L., Belli, A., Sabbatini, L., and Pierleoni, P. (Mar. 2022). "Recent Advances in Internet of Things Solutions for Early Warning Systems: A Review". In: *Sensors* 22.6, p. 2124.

Farahmand, H., Liu, X., Dong, S., Mostafavi, A., and Gao, J. (May 2022). "A Network Observability Framework for Sensor Placement in Flood Control Networks to Improve Flood Situational Awareness and Risk Management". In: *Reliability Engineering & System Safety* 221, p. 108366.

Ginkel, K. C. H. van, Dottori, F., Alfieri, L., Feyen, L., and Koks, E. E. (Mar. 2021). "Flood risk assessment of the European road network". In: *Natural Hazards and Earth System Sciences* 21.3, pp. 1011–1027.

Guttry, C. de and Ratter, B. (Feb. 2022). "Expiry date of a disaster: Memory anchoring and the storm surge 1962 in Hamburg, Germany". In: *International Journal of Disaster Risk Reduction* 70, p. 102719.

Hertweck, P., Hellmund, T., Schaaf, H. v. d., Moßgraber, J., and Blume, J.-W. (2019). *Management of Sensor Data with Open Standards*. en.

Hussen Hajjaj, S. S., Hameed Sultan, M. T., Moktar, M. H., and Lee, S. H. (Feb. 2020). "Utilizing the Internet of Things (IoT) to Develop a Remotely Monitored Autonomous Floodgate for Water Management and Control". In: *Water* 12.2, p. 502.

Jochner, M., Schwander, M., and Brönnimann, S. (2013). "Reanalysis of the Hamburg Storm Surge of 1962". In: *Weather extremes during the past 140 years*, pp. 19–26.

Jonkman, S. N. and Kelman, I. (Feb. 2005). "An Analysis of the Causes and Circumstances of Flood Disaster Deaths: An Analysis of the Causes and Circumstances of Flood Disaster Deaths". In: *Disasters* 29.1, pp. 75–97.

Kendrick, M. (Oct. 1988). "The Thames barrier". In: *Landscape and Urban Planning* 16.1–2, pp. 57–68.

Khaleghi, B., Khamis, A., Karray, F. O., and Razavi, S. N. (Jan. 2013). "Multisensor data fusion: A review of the state-of-the-art". In: *Information Fusion* 14.1, pp. 28–44.

Kulkarni, S. S., Demirbas, M., Madappa, D., Avva, B., and Leone, M. (2014). "Logical Physical Clocks". In: *Principles of Distributed Systems*. Springer International Publishing, pp. 17–32.

LoRa Alliance (2016). *LoRaWAN Specification v1.0.1*. URL: https://resources.lora-alliance.org/technical-specifications (visited on 03/22/2026).

Paprotny, D., Terefenko, P., and Śledziowski, J. (Nov. 2024). "HANZE v2.1: an improved database of flood impacts in Europe from 1870 to 2020". In: *Earth System Science Data* 16.11, pp. 5145–5170.

Petersen, H., Baccelli, E., Wählisch, M., Schmidt, T. C., and Schiller, J. (2015). "The Role of the Internet of Things in Network Resilience". In: *Internet of Things. IoT Infrastructures. First International Summit, IoT360 2014, Revised Selected Papers, Part II*. Vol. 151. LNICST. Berlin, Heidelberg: Springer, pp. 283–296.

Schaad, J. (Aug. 2022). *CBOR Object Signing and Encryption (COSE): Structures and Process*. RFC 9052. IETF.

Schiermeier, Q. (2011). "Increased flood risk linked to global warming". In: *Nature* 470.7334, p. 316.

Shah, R. C., Roy, S., Jain, S., and Brunette, W. (Sept. 2003). "Data MULEs: modeling and analysis of a three-tier architecture for sparse sensor networks". In: *Ad Hoc Networks* 1.2–3, pp. 215–233.

Tretmans, J., Wijbrans, K., and Chaudron, M. (Sept. 2001). "Software Engineering with Formal Methods: The Development of a Storm Surge Barrier Control System Revisiting Seven Myths of Formal Methods". In: *Formal Methods in System Design* 19.2, pp. 195–215.

Vahdat, A., Becker, D., et al. (2000). *Epidemic routing for partially connected ad hoc networks*. Tech. rep. CS-200006, Duke University Durham.

Vrancken, J., Berg, J. V. d., and Soares, M. D. S. (2008). "Human factors in system reliability: lessons learnt from the Maeslant storm surge barrier in The Netherlands". In: *International Journal of Critical Infrastructures* 4.4, p. 418.