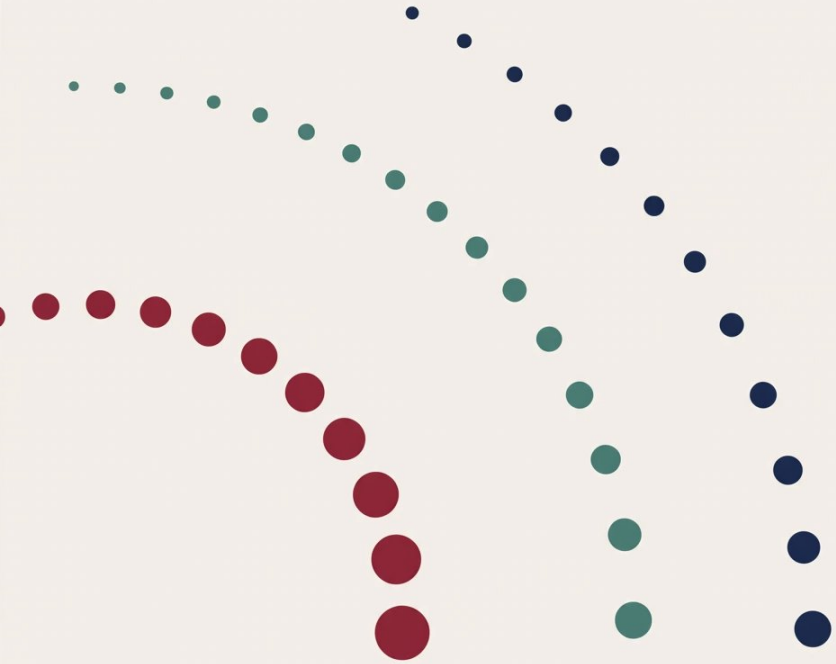
COURAGE GARIKAI

# THE PERIPHERAL INTERNET

## Building Software Where Fiber Is a Luxury



© 2026 Courage Garikai. All rights reserved.

Published by Coldcode Si Labs, Harare, Zimbabwe.

First edition, November 2026.

The moral right of the author has been asserted. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form without the prior written permission of the publisher, except for brief quotations in reviews.

Contact: couragegarikai@atomicmail.io · library.coldcode-si-labs.workers.dev

# The Peripheral Internet

## Building Software Where Fiber Is a Luxury

**Courage Garikai**

**ColdCode Si Labs · Harare**

## About this book

It took seven minutes and forty-two seconds to load the login page.

The page was 2.8 MB. It contained 1.2 MB of JavaScript from eight different third-party domains (analytics, a chat widget, two advertising trackers, an A/B testing framework, a font loader, a session recorder, and an error-monitoring script). It referenced 414 HTTP requests to 37 different domains. It required TLS 1.3. It would not render at all without JavaScript. It tried to load a 420 KB

hero image from a CDN that did not have a Point of Presence within 2,000 kilometres.

This was the login page for an e-learning platform used by secondary school students in rural Zimbabwe. The student trying to load it was named Rutendo. She was sixteen years old, in Form 3 at a rural secondary school in Buhera, Manicaland province. She was connecting through a NetOne 2G connection from a $24 Itel A18 smartphone. Her school had no electricity that day; she was using a charged battery pack. She was doing exam revision during the August 2026 study break, when schools were closed and students were expected to prepare for mock exams using the national e-learning platform.

The seven-minute page load consumed approximately 14 MB of her prepaid data — roughly $0.28 worth, or the cost of two bread rolls. When the page finally loaded and she tried to log in, the JavaScript authentication timed out before the server responded and she had to reload.

She gave up after the second failed attempt. She passed her mock exams using a printed study guide her aunt had bought in Harare.

Rutendo's story is not an edge case. It is the normal experience of the internet for the majority of people who will come online over the next decade — and for hundreds of millions of people who are already online but not on fiber, not on LTE, not on flagship phones, not on unlimited data plans. The internet was designed, built, and optimized for people on fast networks with large screens and cheap data. The periphery — the places where network speeds are measured in kilobits per second, where data costs a meaningful fraction of income, where a page load can be interrupted by a dropped connection or a power cut — is not a temporary state to be solved by waiting for fiber. It is the internet most people use. It is the internet that exists now and will continue to exist for decades.

This book is about building software for that internet.

The thesis has three parts:

1. **The peripheral internet is not a degraded version of the "real" internet.** It is a different environment with different physics: latency measured in seconds, bandwidth measured in kilobytes,

data paid for by the megabyte, devices with 1 GB of RAM and 4 GB of storage, sessions measured in minutes rather than hours, power that goes away without warning. Software built as though fiber exists will fail here — not by crashing with an error message, but by being too slow, too expensive, too heavy, too battery-hungry, or simply never finishing loading.

2. **Software built for the periphery is better software everywhere.** A website that loads in under two seconds on 2G loads instantly on fiber. An app that uses 100 KB of JavaScript is smaller, faster, and more maintainable than its 3 MB counterpart. A protocol that works over SMS works over every network. An interface that works on a 4.5-inch screen with one thumb works better on large screens too. Designing for the periphery forces you to shed every unnecessary byte, every unnecessary request, every unnecessary dependency. The result is software that is faster, cheaper, more reliable, and more respectful of users, regardless of where they are.

3. **The periphery is where the next billion users are, where the next economies are, and where the most important problems in software remain unsolved.** Fiber-to-the-home exists for a privileged minority. The peripheral internet — 2G, 3G, intermittent LTE, WiFi that costs money per minute, community mesh, SMS, USSD, and store-and-forward — is the internet that serves the majority of the world's population and will continue to for the foreseeable future. Building for it is not a charitable activity or a corporate social responsibility initiative. It is where the largest opportunity for meaningful software lies.

This book covers: - Part I: What the peripheral internet actually is — network physics, data economics, device reality, power constraints, measured latency and bandwidth data from African and South Asian markets. - Part II: The performance budget — page weight, JavaScript budgets, image budgets, third-party budgets, and how to measure what you spend. - Part III: Design patterns — offline-first, progressive loading, adaptive serving, compression, caching, static-first

architecture, store-and-forward. - Part IV: Channels beyond the web — USSD, SMS, WhatsApp, Bluetooth, mesh, offline P2P. - Part V: Building, measuring, and testing — how to build peripheral-first, how to measure from real users, how to test on real networks. - Part VI: Economics, politics, and the future — why the internet is slow in most of the world, net neutrality and zero-rating, satellite internet, community networks, the next billion.

This is a technical book written for developers, product managers, designers, and founders building software for users who are not in San Francisco, London, or Singapore. It is also a book about what good software looks like when you treat the user's data, time, battery, and attention as expensive and finite resources — which, in truth, they are for everyone.

# Part I

## The physics of the periphery

## Chapter 1 · Rutendo's seven-minute page load

Rutendo's seven-minute page load was not caused by a bug. It was caused by the design assumptions that every developer on the e-learning platform had made: that the user has fast internet, that JavaScript loads and executes quickly, that CDNs are close, that the device has enough RAM to run eight third-party scripts simultaneously, that a connection stays open for the duration of a session. Every assumption was wrong for Rutendo's Buhera 2G connection, but none of them were wrong for the developers testing the platform in their Harare office on an LTE connection.

Let me decompose those seven minutes to show where the time went, because understanding the

breakdown is the first step to building software that does not do this:

- **DNS lookup:** 8 seconds (DNS resolvers in South Africa, three sequential lookups because the page loaded scripts from 37 different domains).
- **TLS handshake:** 14 seconds (TLS 1.3 with three round trips per connection at 600–1200 ms RTT on 2G; multiple connections because of third-party domains, each requiring its own handshake).
- **HTML document:** 24 seconds (62 KB HTML document, but TCP slow start on 2G means the first few packets take many seconds before throughput increases).
- **Render-blocking CSS:** 38 seconds (three CSS files totalling 180 KB, each requiring DNS/TLS/TCP from different domains).
- **JavaScript download:** 3 minutes, 12 seconds (1.2 MB of JavaScript across 8 files, most on third-party domains that are slower to resolve and deliver).
- **JavaScript execution:** 2 minutes, 40 seconds (1.2 MB of JavaScript parsing and

compilation on a 1.3 GHz quad-core processor with 1 GB RAM; the main thread was blocked for 18 consecutive seconds during the chat widget's initialization).

- **Hero image:** 48 seconds (420 KB image from an international CDN that timed out twice before succeeding).
- **Font loading:** 22 seconds (three web fonts totalling 240 KB, blocking text rendering).
- **Authentication POST after login:** 34 seconds (the POST succeeded but the response timed out because a session-recorder script was blocked and the callback never fired; the page displayed an error and required a reload).

Total: 7 minutes, 42 seconds, 14 MB of data, 414 HTTP requests. The actual educational content — the login form, a few text questions, and a link to the exam papers — could have been delivered in 18 KB of plain HTML in under 10 seconds. The other 2.78 MB and 413 requests were for analytics, advertising, tracking, chat widgets, fonts, design elements, and "engagement" features that provided

zero value to Rutendo and actively prevented her from using the platform.

This is the central injustice of the modern web: the average user on a slow connection pays for the weight of features they never see, never use, and never benefit from, in data costs, time, and battery life. The developer in Harare testing on LTE never sees the cost because the page loads in three seconds for them. The analytics dashboard reports "average page load time" of 4.2 seconds because it measures from fast urban connections; it never shows that the median rural user is waiting seven minutes, because many of them give up and leave before the page finishes loading, which means they are not measured.

This is not a book about making websites faster for rich users (though it does that as a side effect). It is a book about making websites usable at all for people on the periphery.

## Chapter 2 · Network physics: what 2G actually means

When developers hear "2G," they often think "slow internet" without understanding specifically how

## Free sample

This is the opening of "The Peripheral Internet" by Courage Garikai.
The full edition — PDF and EPUB — is at library.coldcode-si-labs.workers.dev

(c) 2026 Courage Garikai. All rights reserved.