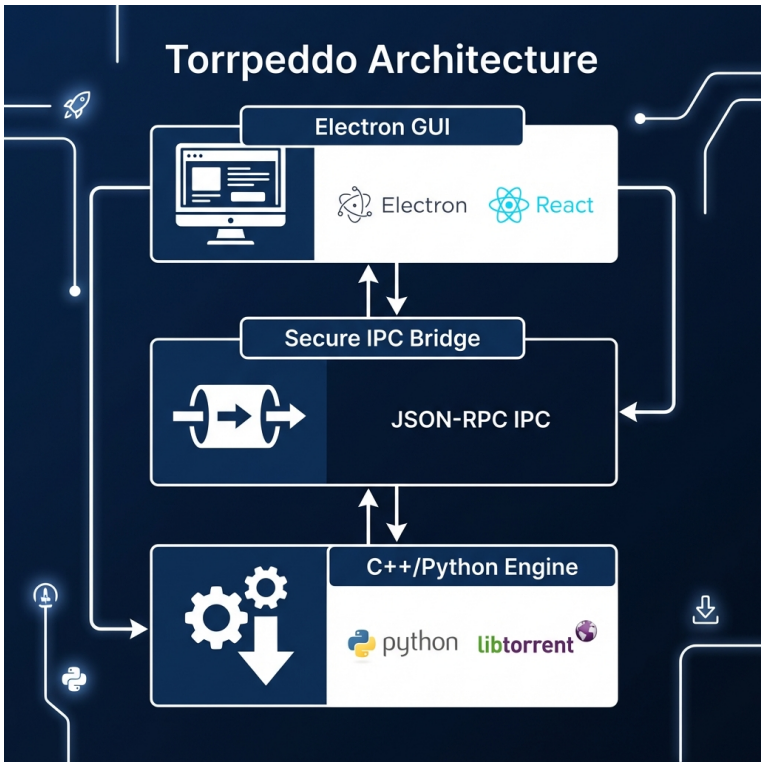
TORRPEDDO PROJECT BOOK



Executive Summary

Torrpeddo is an industrial-grade, premium torrent client designed for the modern desktop. Built primarily with Python and the Electron framework, Torrpeddo leverages the powerful `libtorrent` suite to offer a seamless, high-performance experience that bridges the gap between complex network protocols and professional user interfaces.

Architectural Deep Dive

Torrpeddo follows a decoupled architectural pattern, separating the presentation layer from the core logic and network engine. This is achieved through three primary layers:

1. Frontend: Electron Framework

What is Electron?

Electron is an open-source framework developed by GitH that allows developers to build cross-platform desktop applications using web technologies: HTML, CSS, and

JavaScript. It combines the Chromium rendering engine (
the UI) and the Node.js runtime (for system-level access)

Benefits for Torrpeddo:

- Visual Excellence: Leveraging the full power of modern
CSS and web components to create a "WOW" factor UI th
feels premium.

2. The Bridge: IPC (Inter-Process Communication)

What is IPC?

IPC, or Inter-Process Communication, is a mechanism tha
allows different processes to share data and coordinate
actions. In Torrpeddo, we use a custom IPC bridge to con
the Electron frontend with the Python backend.

Implementation: Secure JSON-RPC

Communication is handled via a JSON-RPC protocol ove
stdin/stdout channels.

Why this approach?

- Decoupling: The engine can be updated, debugged, or
even replaced without touching the UI.

3. Backend Engine: Python & libtorrent

The Core: libtorrent with Python Bindings

At the heart of Torrpeddo is `libtorrent`, a feature-complet
BitTorrent implementation. While the underlying engine is
implemented in high-performance C++, Torrpeddo utilizes
official Python bindings for rapid development and seamle
integration with the bridge logic.

Multi-threaded Performance:

- Engine Level: The `libtorrent` 2.0+ engine utilizes an internal thread pool for disk I/O, network polling, and piec validation. This allows for parallel processing of multiple torrent fragments simultaneously.

Development Process & Methodology

The Torrpeddo project followed a "Platform-First" methodology:

1. Language Choice: Python was selected for its exten library support and ease of integration with libtorrent and protocols.
