

# Roadmap

This document outlines potential improvements, new features, code cleanup opportunities, and technical debt for the Protolean project.

---

## Feature Proposals

### [Priority: High] Proto2 Syntax Support

**Description:** Add support for parsing and code generation from Proto2 syntax files. **Rationale:** Proto2 is still widely used in legacy systems. Many organizations have existing .proto files using Proto2 syntax with required fields, extensions, and groups. **Affected Files:** - /Users/Shared/Projects/lean-workspace/protolean/Protolean (add Proto2-specific constructs) - /Users/Shared/Projects/lean-workspace/protolean/Protolean/Parser/Proto.lean (parse “proto2” syntax and required/extensions) - /Users/Shared/Projects/lean-workspace/protolean/Protolean/Codegen (generate required field handling) **Estimated Effort:** Large **Dependencies:** None

### [Priority: High] Complete Well-Known Types Implementation

**Description:** Implement remaining Google well-known types: Any, FieldMask, Struct, Value, ListValue, and Type/SourceContext/Api. **Rationale:** The current implementation covers Empty, Timestamp, Duration, and Wrapper types, but many proto files use Any for dynamic typing and FieldMask for partial updates. **Affected Files:** - Create /Users/Shared/Projects/lean-workspace/protolean/Protolean/WellKnown/Any.lean - Create /Users/Shared/Projects/lean-workspace/protolean/Protolean/WellKnown/FieldMask.lean - Create /Users/Shared/Projects/lean-workspace/protolean/Protolean/WellKnown/Struct.lean - Update /Users/Shared/Projects/lean-workspace/protolean/Protolean/WellKnown.lean (update registry) **Estimated Effort:** Medium **Dependencies:** None

### [Priority: High] Proper Nested Message Decoding in Codegen

**Description:** The current code generation for decoding named types and nested messages uses skipField instead of actually decoding the embedded message. **Rationale:** This is a significant functionality gap. Lines 57-58 and 72-73 in /Users/Shared/Projects/lean-workspace/protolean/Protolean/Codegen/Decode.lean skip nested messages rather than decoding them properly. **Affected Files:** - /Users/Shared/Projects/lean-workspace/protolean/Protolean/Codegen/Decode.lean (implement proper embedded message decoding) - /Users/Shared/Projects/lean-workspace/protolean/Protolean/Codegen/Verify.lean (verify embedded encoding is correct) **Estimated Effort:** Medium **Dependencies:** None

### [Priority: Medium] JSON Serialization Support

**Description:** Add JSON encoding and decoding for protobuf messages following the canonical JSON mapping. **Rationale:** JSON is the standard format for REST APIs and debugging. The proto3 spec defines a canonical JSON mapping that should be supported. **Affected Files:** - Create /Users/Shared/Projects/lean-workspace/protolean/Protolean/JSON/Encoder.lean - Create /Users/Shared/Projects/lean-workspace/protolean/Protolean/JSON/Decoder.lean - Create /Users/Shared/Projects/lean-workspace/protolean/Protolean/JSON/Codec.lean **Estimated Effort:** Large **Dependencies:** JSON parsing library

### [Priority: Medium] Text Format Support

**Description:** Add support for Protocol Buffers text format (human-readable representation). **Rationale:** Text format is useful for configuration files, debugging, and human-editable proto data. **Affected Files:** - Create /Users/Shared/Projects/lean-workspace/protolean/Protolean/Text/Encoder.lean - Create /Users/Shared/Projects/lean-workspace/protolean/Protolean/Text/Decoder.lean **Estimated Effort:** Medium **Dependencies:** None

### [Priority: Medium] Unknown Field Preservation

**Description:** Implement full unknown field preservation during decode/encode roundtrips. **Rationale:**

The `UnknownField` and `UnknownFields` types exist in `/Users/Shared/Projects/lean-workspace/protolean/Protolean/M` (lines 9-20) but are not integrated into the code generation or decode loop. **Affected Files:** -

`/Users/Shared/Projects/lean-workspace/protolean/Protolean/Message.lean` - `/Users/Shared/Projects/lean-work`  
- `/Users/Shared/Projects/lean-workspace/protolean/Protolean/Codegen/Decode.lean` - `/Users/Shared/Projects/1`

**Estimated Effort:** Medium **Dependencies:** None

### [Priority: Medium] Proto Path Environment Extension

**Description:** Implement proper storage of `proto_path` declarations using Lean environment extensions.

**Rationale:** The `proto_path` command in `/Users/Shared/Projects/lean-workspace/protolean/Protolean/Codegen/Im` (lines 121-131) acknowledges the path but does nothing with it. Proto paths should be stored and used for im-

port resolution. **Affected Files:** - `/Users/Shared/Projects/lean-workspace/protolean/Protolean/Codegen/Import.1`  
- `/Users/Shared/Projects/lean-workspace/protolean/Protolean/Import/Resolver.lean` **Estimated**

**Effort:** Small **Dependencies:** None

### [Priority: Medium] Oneof Field Code Generation

**Description:** Improve oneof field support with proper encoding/decoding in generated `ProtoMes-`  
sage instances. **Rationale:** Oneof inductive types are generated but the encoding/decoding logic

for oneof fields is not included in the generated `ProtoMessage` instances. **Affected Files:** -

`/Users/Shared/Projects/lean-workspace/protolean/Protolean/Codegen/Encode.lean` - `/Users/Shared/Projects/1`

- `/Users/Shared/Projects/lean-workspace/protolean/Protolean/Codegen/Types.lean` **Estimated**

**Effort:** Medium **Dependencies:** None

### [Priority: Medium] ProtoEnum Code Generation

**Description:** Generate `ProtoEnum` instances for parsed enum definitions to enable enum encode/decode.

**Rationale:** Enum inductive types are generated in `/Users/Shared/Projects/lean-workspace/protolean/Protolean/Cod` but `ProtoEnum` instances (`toInt32/fromInt32`) are not generated. **Affected Files:** - `/Users/Shared/Projects/lean-worksp`

(add enum instance generation) **Estimated Effort:** Small **Dependencies:** None

### [Priority: Low] Derive Handler for ProtoMessage

**Description:** Create a `deriving ProtoMessage` handler that automatically generates encode/decode from

structure fields. **Rationale:** Would eliminate the need for manual `ProtoMessage` instance writing and im-

prove ergonomics for manually-defined types. **Affected Files:** - Create `/Users/Shared/Projects/lean-workspace/protol`

**Estimated Effort:** Large **Dependencies:** None

### [Priority: Low] gRPC Integration Documentation

**Description:** Add documentation and examples for using Protolean with the legate gRPC library.

**Rationale:** Protolean and legate are designed to work together, but there is no documentation

showing the integration pattern. **Affected Files:** - Create documentation/examples - Update

`/Users/Shared/Projects/lean-workspace/protolean/README.md` **Estimated Effort:** Small **De-**

**pendencies:** legate library

### [Priority: Low] Field Validation Support

**Description:** Add support for proto3 field validation annotations (like `validate.proto` patterns). **Rationale:**

Many organizations use field validation to ensure message integrity. Supporting common validation patterns

would be valuable. **Affected Files:** - Create `/Users/Shared/Projects/lean-workspace/protolean/Protolean/Validati`

**Estimated Effort:** Large **Dependencies:** None

## Code Improvements

### [Priority: High] Packed Repeated Field Decoding for All Types

**Description:** Add missing packed decoding support for sint32, sint64, int64, uint32, and sfixed types.

**Current State:** /Users/Shared/Projects/lean-workspace/protolean/Protolean/Repeated.lean has packed encoding functions for various types but the decode generation in /Users/Shared/Projects/lean-workspace/protolean/Protolean/Repeated.lean only handles unpacked repeated fields.

**Proposed Change:** The decode field generation should detect repeated scalar fields and generate code that handles both packed and unpacked formats.

**Benefits:** Correct proto3 semantics (proto3 defaults to packed encoding for repeated scalars)

**Affected Files:** - /Users/Shared/Projects/lean-workspace/protolean/Protolean/Codegen/Decode.lean

**Estimated Effort:** Medium

### [Priority: High] Eliminate Partial Functions in Decoder

**Description:** Replace partial decodeMessageLoop and decodeEmbeddedLoop with total functions using fuel or termination proofs.

**Current State:** /Users/Shared/Projects/lean-workspace/protolean/Protolean/Message.lean (lines 36, 54) use `partial` for the decode loops.

**Proposed Change:** Use fuel parameter or prove termination based on input byte consumption.

**Benefits:** Improved safety guarantees, better compile-time verification

**Affected Files:** - /Users/Shared/Projects/lean-workspace/protolean/Protolean/Message.lean

**Estimated Effort:** Medium

### [Priority: Medium] ByteBuilder Size Tracking

**Description:** Track size during building instead of computing at the end.

**Current State:** /Users/Shared/Projects/lean-workspace/protolean/Protolean/ByteArray/Builder.lean (lines 42-43) must build the entire ByteArray to compute size.

**Proposed Change:** Add a size field to ByteBuilder that is incremented during building operations.

**Benefits:** O(1) size queries instead of O(n), useful for length-prefixed encoding

**Affected Files:** - /Users/Shared/Projects/lean-workspace/protolean/Protolean/ByteArray/Builder.lean

**Estimated Effort:** Small

### [Priority: Medium] Unified Varint Encoding to Builder

**Description:** Emit varints directly to ByteBuilder instead of creating intermediate ByteArrays.

**Current State:** /Users/Shared/Projects/lean-workspace/protolean/Protolean/Encoder.lean (lines 44-49) create intermediate ByteArrays via `Varint.encodeUInt64/32`.

**Proposed Change:** Add `ByteBuilder.varint` that writes directly without intermediate allocation.

**Benefits:** Reduced allocations, better performance

**Affected Files:** - /Users/Shared/Projects/lean-workspace/protolean/Protolean/ByteArray/Builder.lean - /Users/Shared/Projects/lean-workspace/protolean/Protolean/Encoder.lean

**Estimated Effort:** Small

### [Priority: Medium] Type-Safe Field Numbers

**Description:** Use a dedicated FieldNumber type with validation instead of raw UInt32.

**Current State:** /Users/Shared/Projects/lean-workspace/protolean/Protolean/WireFormat.lean (line 43) defines `abbrev FieldNumber := UInt32`.

**Proposed Change:** Create a proper structure with smart constructor that validates field numbers are in range 1 to  $2^{29}-1$  and not in reserved range 19000-19999.

**Benefits:** Compile-time or construction-time validation of field numbers

**Affected Files:** - /Users/Shared/Projects/lean-workspace/protolean/Protolean/WireFormat.lean

**Estimated Effort:** Small

### [Priority: Medium] Better Error Messages in Parser

**Description:** Add source location information to parse errors.

**Current State:** /Users/Shared/Projects/lean-workspace/protolean/Protolean/Parser/Proto.lean (line 333) returns errors without position info.

**Proposed Change:** Track position during parsing and include line/column in error messages.

**Benefits:** Easier debugging of proto file syntax errors

**Affected Files:** - /Users/Shared/Projects/lean-workspace/protolean/Protolean/Parser/Proto.lean

/Users/Shared/Projects/lean-workspace/protolean/Protolean/Parser/Lexer.lean - /Users/Shared/Projects/lean-workspace/protolean/Protolean/WellKnown.lean  
**Estimated Effort:** Medium

#### [Priority: Medium] Replace HashMap with Lean4 Standard HashMap

**Description:** Use Lean.HashMap instead of Std.HashMap for consistency. **Current State:** Multiple files import Std.Data.HashMap (e.g., /Users/Shared/Projects/lean-workspace/protolean/Protolean/Map.lean, /Users/Shared/Projects/lean-workspace/protolean/Protolean/WellKnown.lean). **Proposed Change:** Evaluate whether Lean.HashMap or Std.HashMap is more appropriate and use consistently. **Benefits:** Reduced dependencies, consistency **Affected Files:** - /Users/Shared/Projects/lean-workspace/protolean/Protolean/Map.lean - /Users/Shared/Projects/lean-workspace/protolean/Protolean/WellKnown.lean - /Users/Shared/Projects/lean-workspace/protolean/Protolean/Decoder.lean **Estimated Effort:** Small

#### [Priority: Low] Float32 Precision Handling

**Description:** Improve float32/float64 conversion to handle edge cases better. **Current State:** /Users/Shared/Projects/lean-workspace/protolean/Protolean/Encoder.lean (lines 104-133) and /Users/Shared/Projects/lean-workspace/protolean/Protolean/Decoder.lean (lines 203-235) have inline float conversion with limited denormal handling. **Proposed Change:** Extract to separate module, add property-based tests for edge cases, consider FFI for IEEE754 compliance. **Benefits:** Correct handling of denormals, NaN payloads, and edge cases **Affected Files:** - Create /Users/Shared/Projects/lean-workspace/protolean/Protolean/Float.lean - /Users/Shared/Projects/lean-workspace/protolean/Protolean/Encoder.lean - /Users/Shared/Projects/lean-workspace/protolean/Protolean/Decoder.lean **Estimated Effort:** Medium

#### [Priority: Low] ProtoMergeable Implementation

**Description:** Implement proper message merging semantics. **Current State:** /Users/Shared/Projects/lean-workspace/protolean/Protolean/ProtoMergeable.lean (lines 88-95) has a ProtoMergeable class but the default instance just takes the later value. **Proposed Change:** Implement recursive merging for embedded messages, concatenation for repeated fields. **Benefits:** Correct proto semantics for duplicate fields **Affected Files:** - /Users/Shared/Projects/lean-workspace/protolean/Protolean/ProtoMergeable.lean - /Users/Shared/Projects/lean-workspace/protolean/Protolean/Codegen/Decode.lean **Estimated Effort:** Medium

---

## Code Cleanup

#### [Priority: High] Remove Unused elabCommandString Function

**Issue:** The elabCommandString function in /Users/Shared/Projects/lean-workspace/protolean/Protolean/Codegen/Tools.lean (lines 162-171) only logs generated code and has a TODO comment. The actual elaboration happens via elaborateCodeString in Import.lean. **Location:** /Users/Shared/Projects/lean-workspace/protolean/Protolean/Codegen/Tools.lean lines 162-176 **Action Required:** Remove elabCommandString, elaborateMessage, and elaborateEnum functions as they are unused (Import.lean uses elaborateCodeString and generateMessageString/generateEnumString directly). **Estimated Effort:** Small

#### [Priority: Medium] Consolidate Repr Instance for ByteArray

**Issue:** There are multiple Repr instances defined for ByteArray in different files. **Location:** - /Users/Shared/Projects/lean-workspace/protolean/Protolean/ByteArray/Basic.lean, lines 6-9 - /Users/Shared/Projects/lean-workspace/protolean/Tests/Scalar.lean, lines 13-14 **Action Required:** Keep one canonical Repr instance and remove duplicates. **Estimated Effort:** Small

### [Priority: Medium] Consistent Naming Convention for Proto Types

**Issue:** Some scalar wrapper types use camelCase (SInt32, SFixed32) while proto types use consistent casing. **Location:** /Users/Shared/Projects/lean-workspace/protolean/Protolean/Scalar.lean **Action Required:** Consider whether wrapper types should mirror proto naming exactly (sint32 -> Sint32 or SInt32). **Estimated Effort:** Small

### [Priority: Medium] Extract Duplicate Packed Encode/Decode Patterns

**Issue:** /Users/Shared/Projects/lean-workspace/protolean/Protolean/Repeated.lean has repetitive encode/decode functions that differ only in the emit/read call. **Location:** /Users/Shared/Projects/lean-workspace/protolean/Repeated.lean lines 18-148 **Action Required:** Create generic encodePackedWith and decodePackedWith functions that take the emit/read operation as a parameter. **Estimated Effort:** Small

### [Priority: Low] Unused ctx Parameter in Code Generation

**Issue:** The ctx parameter is often unused in code generation functions. **Location:** - /Users/Shared/Projects/lean-workspace/protolean/Codegen/Types.lean, line 75 (\_ctx) - /Users/Shared/Projects/lean-workspace/protolean/Codegen/Service.lean, line 103 (\_ctx) - /Users/Shared/Projects/lean-workspace/protolean/Codegen/Service.lean, lines 42, 50 (\_ := ctx) **Action Required:** Either remove unused parameters or document why they are kept for future use. **Estimated Effort:** Small

### [Priority: Low] Standardize Test Organization

**Issue:** Test file organization inconsistency - some tests are in Tests/ root, one is in Tests/Integration/. **Location:** - /Users/Shared/Projects/lean-workspace/protolean/Tests/ - /Users/Shared/Projects/lean-workspace/protolean/Tests/Integration/ **Action Required:** Either organize all tests by category (unit vs integration) or flatten to single directory. **Estimated Effort:** Small

### [Priority: Low] Document Internal Modules

**Issue:** Some internal modules lack module-level documentation. **Location:** - /Users/Shared/Projects/lean-workspace/protolean/Import/Loader.lean has minimal doc - /Users/Shared/Projects/lean-workspace/protolean/Protolean/Import/Loader.lean has minimal doc **Action Required:** Add module-level documentation explaining purpose and usage. **Estimated Effort:** Small

### [Priority: Low] Add Missing Hashable Instances

**Issue:** Map key types should have Hashable instances for use in HashMap. **Location:** /Users/Shared/Projects/lean-workspace/protolean/HashMap.lean **Action Required:** Ensure all scalar types that can be map keys have appropriate Hashable instances. **Estimated Effort:** Small

---

## Test Coverage Improvements

### [Priority: High] Add Map Field Tests

**Issue:** No test coverage for map field encoding/decoding. **Location:** Missing in /Users/Shared/Projects/lean-workspace/protolean/Tests/Map.lean **Action Required:** Add tests for map<string, T>, map<int32, T>, and nested map scenarios. **Estimated Effort:** Small

### [Priority: High] Add Nested Message Tests

**Issue:** No test coverage for nested message encode/decode roundtrips. **Location:** Parser tests verify parsing but no encode/decode tests for nested structures. **Action Required:** Add tests with multi-level nesting and recursive message references. **Estimated Effort:** Small

### [Priority: Medium] Add Float/Double Edge Case Tests

**Issue:** Float encoding tests are minimal; need edge cases like NaN, Infinity, denormals. **Location:** /Users/Shared/Projects/lean-workspace/protolean/Tests/Scalar.lean **Action Required:** Add tests for special float values and precision boundaries. **Estimated Effort:** Small

### [Priority: Medium] Add Proto File Syntax Error Tests

**Issue:** Parser tests only cover valid inputs; no tests for error handling. **Location:** /Users/Shared/Projects/lean-workspace/protolean/Tests/Parser.lean **Action Required:** Add tests verifying correct error messages for malformed proto files. **Estimated Effort:** Small

### [Priority: Low] Add Cross-Language Validation

**Issue:** Integration test fixtures exist but cross-validation with other implementations is limited. **Location:** /Users/Shared/Projects/lean-workspace/protolean/Tests/Integration/ **Action Required:** Add tests comparing output with reference protobuf implementation (e.g., Go, Python). **Estimated Effort:** Medium

---

## Performance Improvements

### [Priority: Medium] Lazy ByteArray Construction

**Description:** Consider using lazy evaluation for large message encoding. **Current State:** Entire message is encoded before any bytes are available. **Proposed Change:** Implement streaming encoder that yields bytes as they are produced. **Benefits:** Lower memory usage for large messages, streaming support. **Affected Files:** - /Users/Shared/Projects/lean-workspace/protolean/Protolean/Encoder.lean - /Users/Shared/Projects/lean-workspace/protolean/Protolean/ByteArray/Builder.lean **Estimated Effort:** Large

### [Priority: Low] Zero-Copy Decoding

**Description:** Avoid copying bytes during string/bytes field decoding where possible. **Current State:** /Users/Shared/Projects/lean-workspace/protolean/Protolean/Decoder.lean copies bytes via extract. **Proposed Change:** Return views/slices for read-only access scenarios. **Benefits:** Reduced memory allocations for read-heavy workloads. **Affected Files:** - /Users/Shared/Projects/lean-workspace/protolean/Protolean/Decoder.lean **Estimated Effort:** Medium

---

## Documentation

### [Priority: Medium] Add API Documentation

**Issue:** Most public functions lack detailed docstrings with examples. **Action Required:** Add comprehensive docstrings to all public API functions in Codec.lean, Message.lean, and the main Protolean.lean module. **Estimated Effort:** Medium

### [Priority: Low] Add Architecture Documentation

**Issue:** No high-level architecture documentation explaining the module structure. **Action Required:** Add ARCHITECTURE.md explaining the layered design (wire format -> codecs -> messages -> codegen). **Estimated Effort:** Small