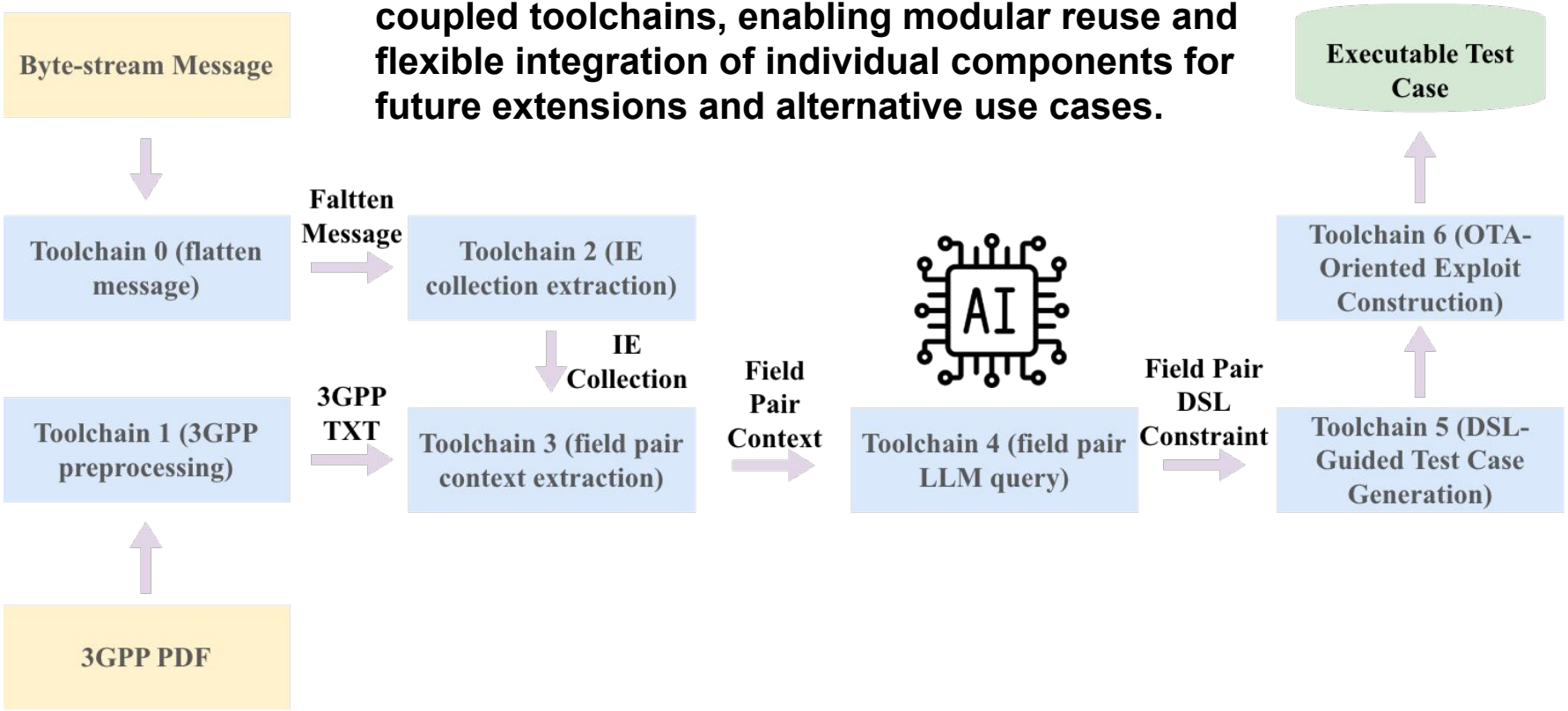
A2_pipeline_overview

- Toolchain 0 (flatten message)
  - Purpose: Given byte-stream message, generate flatten message
- Toolchain 1 (3GPP preprocessing)
  - Purpose: PDF to TXT
  - Goal: Keep the information as much as possible
- Toolchain 2 (IE collection extraction)
  - Purpose: From the give flatten message, extract IE collections
  - Goal: extracted IE collection can reach good coverage of the message
- Toolchain 3 (field pair context extraction)
  - Purpose: From the given IE collection. Perform intra-IE and inter-IE field pair context extraction
- Toolchain 4 (field pair LLM query)
- Toolchain 5 (DSL to testcase)
- Toolchain 6 (generate OTA testcase)

A2_pipeline_overview

We design the pipeline as a set of loosely coupled toolchains, enabling modular reuse and flexible integration of individual components for future extensions and alternative use cases.



Toolchain 1 (3GPP preprocessing) - Line Merging

- Stage 1: Intra-Page Merging
  - Hyphenated word breaks: "Config-" + "uration" → "Configuration"
  - Mid-sentence breaks: Lines ending without period + next line starts lowercase → Merge
  - Unclosed brackets/parentheses: Incomplete pairs → Merge until closed
  - Connecting words: Lines ending with "the", "of", "to", "for", etc. → Merge with next
- Stage 2: Cross-Page Merging
  - Strict conditions (all must be true):
  - Page 1 ends without sentence terminator (.!?)
  - Page 1 doesn't end with list item or colon
  - Page 2 starts with lowercase letter
  - Page 2 doesn't start with list/heading
- To be continued

Toolchain 1 (3GPP preprocessing) - Challenge

Table 10.1-3A: Maximum number $C_{\mathrm{PDCCH}}^{max,(X,Y),\mu}$ of non-overlapped CCEs in a span for combination (X,Y) for a DL BWP with SCS configuration $\mu \in \{0,1\}$ for a single serving cell

	Maximum number $C_{\mathrm{PDCCH}}^{max,(X,Y),\mu}$ of non-overlapped CCEs per span for combination (X,Y) and per serving cell		
μ	(2, 2)	(4, 3)	(7, 3)
0	18	36	56
1	18	36	56

Table 10.1-3B provides the maximum number of non-overlapped CCEs, $C_{\mathrm{PDCCH}}^{max,X_s,\mu}$, for a DL BWP with SCS configuration μ that a UE is expected to monitor corresponding PDCCH candidates per group of X_s slots for combination (X_s, Y_s) for operation with a single serving cell.

Table 10.1-3B: Maximum number $C_{\mathrm{PDCCH}}^{max,X_s,\mu}$ of non-overlapped CCEs in a group of X_s slots for any combination (X_s, Y_s) for a DL BWP with SCS configuration $\mu \in \{5,6\}$ for a single serving cell

	Maximum number of non-overlapped CCEs in a group of X_s slots per combination (X_s, Y_s) and per serving cell $C_{\mathrm{PDCCH}}^{max,X_s,\mu}$			
μ	(4, 1)	(4, 2)	(8, 1)	(8, 4)
5	32	32	-	-
6	16	16	32	32

If a UE

- does not report *pdcch-BlindDetectionCA* or is not provided $BDFactorR, \gamma = R$

```
8283  [TABLE_START | COLS: 4 | ROWS: 6]
8284  |  | (cid:6) , , , Maximum number (cid:3)(cid:4)(cid:5)(cid:5)(cid:6) of no
      (cid:2)(cid:0) no(cid:4)verlapped (cid:3) |  |  |
8285  |---|---|---|---|
8286  |  | CCEs per span for combination , and per |  |  |
8287  | (cid:7) | serving cell |  |  |
8288  |  | (2, 2) | (4, 3) | (7, 3) |
8289  | 0 | 18 | 36 | 56 |
8290  | 1 | 18 | 36 | 56 |
8291  [TABLE_END]
8292
8293
8294  [TABLE_START | COLS: 5 | ROWS: 6]
8295  |  | Maximum number of non-overlapped CCEs in a group of (cid:2) slots
      (cid:2)(cid:0) (cid:3) (cid:4) (cid:6)(cid:7)(cid:8)(cid:9)(cid:15)
      (cid:0) |  |  |  |
8296  |---|---|---|---|---|
8297  |  | per combination (cid:2), (cid:2) and per serving cell (cid:3)(cid:4)
      (cid:5)(cid:5) , (cid:6)(cid:0), |  |  |  |
      | (cid:5) |  |  |  |  |
8299  |  | (4, 1) | (4, 2) | (8, 1) | (8, 4) |
8300  | 5 | 32 | 32 | - | - |
8301  | 6 | 16 | 16 | 32 | 32 |
8302  [TABLE_END]
```

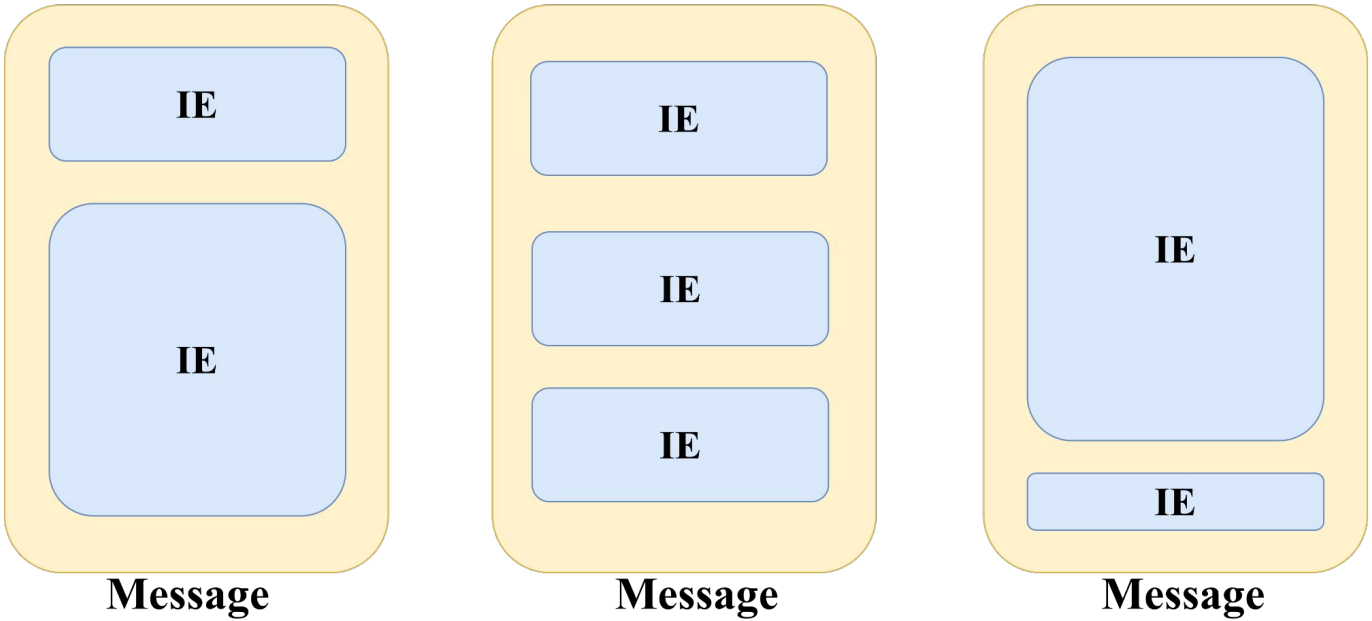
Toolchain 1 (3GPP preprocessing) - switch to OCR solution

- **Solution**
  - Apply OCR and convert the extracted content into LaTeX format to preserve as much information as possible.
  - Correct line break artifacts introduced during OCR.
  - Preserve mathematical formulas.
  - Preserve tabular structures.

Toolchain 2 (IE collection extraction): Intra-IE

- Problem Definition:
  - Original message contains **940 fields**
- Brute force approach:
  - Process **879 IEs (all candidates)**
  - Generate 1,148,652 field pairs
  - Estimated LLM cost: $11,487-34,460
- Challenge:
  - Too expensive and time-consuming for practical use
- Goal:
  - Select minimal IE collection to maximize the message coverage
  - Control LLM query cost

Toolchain 2 (IE collection extraction): Intra-IE



There has multiple IE combo to cover the message.

Toolchain 2 (IE collection extraction): Intra-IE

Solution: Greedy Set Cover with Parameter Search

- Parameter Space Search
  - search min_fields × max_fields combo
  - For example: min ∈ [3,5], max ∈ [10,30]
- Greedy Selection
  - Iterate min_fields × max_fields combo
  - Use Greedy algorithm for search
- Scoring & Ranking
  - Score = Coverage(×1.0) + IE_Count(×0.3) + Pair_Count(×0.2)
- Optimization Goals:
  - Coverage Rate: ≥90%
  - IE Count: 50-150 (optimal)
  - Field Pairs: 1,000-8,000 (cost control)

Toolchain 2 (IE collection extraction): Intra-IE

Metric	Before	After	Reduction
IE Count	879	83	**91% ↓**
Field Pairs	1,148,652	7,019	**99% ↓**
Coverage	100%	90.5%	9.5% ↓
LLM Cost	$11,487-34,460	$70-211	**99% ↓**

- Parameter Selection:
  - **min_fields = 3**
  - **max_fields = 30**
- Results analysis:
  - Message Coverage: 90.53%
  - IE number: 83
  - Field Pair Count: 7019
- Trade-off:
  - Sacrifice 9.5% coverage, to reduce 99% cost reduce

Toolchain 2 (IE collection extraction): Inter-IE

- Problem Definition:
  - Extract reference constraints across different IEs
  - Example: pucch-Config.bwp-Id MUST reference bwp-UplinkCommon.bwp-Id
- IE Candidate Pool:
  - 299 IEs (after filtering & deduplication)
  - ~3,000 fields across these IEs
  - 32 reference fields identified (ending with 'Id' or 'ID')
- Challenge - The Blind Search Problem:
  - Without knowing which field pairs are references:
  - Potential cross-IE field combinations: 3,000 × 3,000 = 9,000,000
  - Which pairs represent actual references? Unknown
  - Testing approach: Try everything and hope to find references

Toolchain 2 (IE collection extraction): Inter-IE

- Brute Force Approach:
  - Test all 9M possible field pair combinations
  - Cannot distinguish reference pairs from non-reference pairs
  - Estimated LLM cost: $90,000 - $270,000
  - Result: most wasted effort on meaningless combinations
- Goal:
  - Identify WHICH field pairs to test (not test everything blindly)
  - Focus only on meaningful reference relationships
  - Dramatically reduce search space while maintaining high coverage
  - **Select the IE with the highest coverage of citation relationships.**

Toolchain 2 (IE collection extraction): Inter-IE -> Reference Driven

- Step 1: Reference Field Identification
  - Strategy: Only fields ending with 'Id' or 'ID' form references
  - From 940 total fields → Identified **32** reference fields
  - Examples: *bwp-Id*, *controlResourceSetId*, *searchSpaceId*
- Step 2: Calculate Meaningful Reference Pairs
  - Logic: Same reference field name in different IEs = reference pair
  - Example: bwp-Id appears in 33 IEs → C(33,2) = 528 potential pairs
  - Process: Enumerate all same-name field pairs across IEs
  - Result: 3,361 meaningful reference pairs identified
  - Search Space Reduction: **9,000,000 → 3,361 (99.96% decrease)**

Toolchain 2 (IE collection extraction): Inter-IE -> Reference Driven

- Step 3: Greedy Selection for Coverage
  - Goal: Select minimal IEs to cover maximum reference pairs
  - Scoring Formula: Score = New_Ref_Pairs(×100) + New_Ref_Fields(×10) + New_Field_IDs(×1) + New_Field_Names(×0.1)
  - Priority: Reference pairs > Reference fields > Coverage
- Parameter Selection:
  - min_fields = 1 (include single-field reference IEs)
  - max_fields = 70 (no upper limit for reference IEs)
  - Tested 24 parameter combinations: min $\in$ [1,2,4,5] × max $\in$ [20-70]
- Results:
  - IEs Selected: 295 (from 299 candidates)
  - Reference Field Coverage: 100% (32/32 fields)

Toolchain 3 (field pair context extraction) Overview

- Goal:
  - Extract text contexts with potential constraints from 3GPP specifications
  - Feed high-quality constraint candidates to Toolchain 4 (LLM-based DSL generation)
- Two Categories:
  - Intra-IE: Constraints between fields within a single IE
  - Inter-IE: Constraints between fields across different IEs (reference integrity)
- Pipeline Position:
  - Toolchain 2 (IE Extraction) ->***Toolchain 3 (Context Extraction)*** ->Toolchain 4 (LLM-based DSL Generation)
- Output:
  - **141,666** constraint contexts ready for LLM processing
  - Intra-IE: **22,456** contexts
  - Inter-IE: **119,210** contexts

Toolchain 3 (field pair context extraction): Intra-IE

- Definition: Extract constraints between fields within a single Information Element
- Example:

  IE: PDSCH-Config

  Constraint: "If mcs-Table is qam256, then

  maxNrofCodeWordsScheduledByDCI shall be 1"

  Field Pair: mcs-Table × maxNrofCodeWordsScheduledByDCI

- Scale:
  - IEs Processed: 83
  - Field Pairs: All combinations (C(n,2) + n for self-reference)
  - Output Contexts: 22,456
  - Files: 22,456 (one per field pair)

Toolchain 3 (field pair context extraction): Intra-IE Methodology

- 1. Field Pair Generation
  - Generate all pairwise combinations for each IE
  - Include self-reference (same field with conditional constraints)
- 2. Constraint Detection
  - Keyword Matching: 50+ constraint keywords
    - Mandatory: shall, must, required, mandatory
    - Conditional: if, when, provided that, only if
    - Dependency: depends on, determined by, according to
    - Reference: reference, correspond, match
  - Context Window: ±2 lines around keyword
  - Field Co-occurrence: Both fields appear in nearby lines
- 3. Section Relevance Scoring
  - Calculate how well section titles match IE names
  - Scoring rules:
    - Exact match: 50 points
    - Contains match: 10-30 points
    - Word overlap: 5 points per word

Toolchain 3 (field pair context extraction): Intra-IE

- Confidence Levels (based on section relevance):
  - HIGH (≥30): Constraints in IE definition sections
  - MEDIUM (15-30): Constraints in related sections
  - LOW (5-15): Constraints in weakly related sections
  - VERY_LOW (<5): Constraints in generic sections
- Distribution:
  - HIGH: ~15%
  - MEDIUM: ~25%
  - LOW: ~35%
  - VERY_LOW: ~25%
- Design Philosophy:
  - Conservative threshold (0.0) - keep all potential constraints
  - Rely on Toolchain 4 (LLM) for final filtering
  - Section relevance provides quality signal for LLM prioritization

Toolchain 3 (field pair context extraction): Intra-IE Challenge & Solution

- Challenge 1: Field Name Diversity
  - Problem: Same field expressed in multiple ways
    - CamelCase, hyphenated, abbreviated
  - Solution: Multi-pattern matching + protocol-specific acronym mapping
    - Example: "schedulingRequestID" → "SR", "sr", "scheduling request id"
- Challenge 2: Complex Constraint Expressions
  - Problem: Natural language variety in specifications
  - Solution: 50+ keywords + 7 special patterns
    - Patterns: [same_value], [range_constraint], [conditional_reference]
- Challenge 3: Section Relevance Assessment
  - Problem: How to judge constraint quality automatically
  - Solution: IE name variant generation + section title matching
    - Variants: CamelCase, Config→Configuration, acronyms
- Performance Optimization:
  - Pre-compiled regex patterns
  - Batch field lookups
  - Caching mechanisms
  - Multi-process parallelization

Toolchain 3 (field pair context extraction): Inter-IE

- Definition: Extract constraints between fields across different IEs (reference integrity)
- Example:

  IE1: MAC-LogicalChannelConfig IE2: PUCCH-Config

  Constraint: "schedulingRequestID shall reference valid resourceId in PUCCH-Config"

  Field Pair: schedulingRequestID × resourceId

- Scale:
  - IE Pairs: 9,159 (after cluster optimization)
  - Field Pairs: ~200 per IE pair (average)
  - Output Contexts: 119,210
  - Processing Time: 1.56 hours
- Key Difference from Intra-IE:
  - Constraints context appear in procedure chapters, not IE definition sections
  - Massive combinatorial explosion without optimization

Toolchain 3 (field pair context extraction): Inter-IE challenge

Fundamental Differences from Intra-IE:

Dimension	Intra-IE	Inter-IE
Location	IE definition sections	Procedure/flow chapters
Section Relevance	High (title contains IE name)	Low (title describes process)
Field Pair Scale	Manageable	Explosive growth
Constraint Type	Value/conditional	Reference integrity

- Core Problems:
  - Problem 1: Section Mismatch
    - Constraint appears in: "UL Grant Reception" (process chapter)
    - Not in: "MAC-LogicalChannelConfig" (IE definition section) → Section relevance score = 0 for most Inter-IE constraints
  - Problem 2: Combinatorial Explosion
    - Without optimization:
    - 295 IEs × 295 IEs × 10 fields × 20 fields = 17M+ field pairs
    - Impossible to process!

Toolchain 3 (field pair context extraction): Inter-IE design

- Design 1: Smart Field Pair Filtering
  - Reference Field Priority:
    - 1. Field ending with "Id"/"ID" → MUST generate pair
    - 2. Same field name across IEs → MUST generate pair
    - 3. Name similarity > 0.3 → Generate pair
    - 4. Others → SKIP
  - Effect: 70-80% reduction in invalid field pairs
- Design 2: Cluster-based IE Pairing
  - Protocol Layer Clustering:
  - Clusters: PHY, MAC, RRC, PDCP, RLC...
  - Strategy: Only pair IEs from related clusters
  - Effect: 295×295 = 86,025 → 9,159 pairs (89% reduction!)

Toolchain 3 (field pair context extraction): Inter-IE design

- Design 3: Multi-Signal Confidence (v2)
  - Score = Keyword Strength (0-40 pts) + Field Match Quality (0-30 pts) + Section Relevance (0-30 pts, bonus)
- Grading:
  - ≥60: HIGH | 20-40: MEDIUM
  - 40-60: HIGH | <20: VERY_LOW
    - HIGH: ~20%
    - MEDIUM: ~40%
    - LOW: ~30%
    - VERY_LOW: ~10%
- Key Insight:
  - Not rely on section matching for Inter-IE constraints
  - Use constraint keywords as primary signal
  - Field co-occurrence as validation
  - Section relevance as optional bonus

Toolchain 4 (field pair LLM query)

- If using GPT-4 (our choice)
  - Intra-IE: 144 DSLs, 386 field pairs
  - Inter-IE: 593 DSLs, 1230 field pairs
  - Total: 737 DSLs, 1616 field pairs
- If using GPT-4o
  - Intra-IE: 22 DSLs, 32 field pairs
  - Inter-IE: 39 DSLs, 165 field pairs
  - Total: 61 DSLs, 197 field pairs

Toolchain 5: DSL-Guided Test Case Generation

- **Purpose**
  - To interpret synthesized DSL rules and apply semantic mutations to protocol messages in a controlled and systematic manner.
- **Functionality**
  - Parse DSL rules generated from prior analysis stages.
  - Execute a DSL engine that translates high-level semantic constraints into concrete message-level mutations.
  - Map field-level modifications specified by DSL rules onto structured protocol messages.
  - Ensure that generated test cases remain syntactically valid while intentionally violating targeted semantic constraints.
- **Outcome**
  - A set of mutated protocol messages that precisely reflect the intended semantic manipulations described by the DSL.
  - Test cases suitable for controlled evaluation in simulation environments.

Toolchain 6: OTA-Oriented Exploit Construction

- **Purpose**
  - To further transform DSL-guided test cases into exploit-ready inputs suitable for over-the-air (OTA) testing scenarios.
- **Functionality**
  - Adapt simulation-level test cases to account for OTA transmission requirements and runtime constraints.
  - Construct exploit-oriented test cases that can be injected through the gNB message delivery path.
  - Preserve the semantic violations introduced by the DSL while ensuring compatibility with OTA execution workflows.
- **Outcome**
  - OTA-oriented exploit test cases that enable evaluation of real-world protocol implementations.
  - A bridge between abstract semantic mutation and practical over-the-air testing.