**Motivation**
- Modern software engineering tools are driven by sophisticated automatic software analysis
- Automatic analysis of software’s natural language (user-defined names) requires accurate parsing of the multi-word names (e.g., `isPointInImage`)

**Research Question**
How can we automatically identify the parts of speech and parse program identifiers with high accuracy?

**Current Focus:** program method signatures

**Target Applications**
- Automatic comment generation
- Program search and navigation

**Parsing Method**
- Two step algorithm: tagging and chunking
- Tagging uses morphology rules (e.g., words ending in “-ity” are nouns, adjectives form adverbs when “-ly” is added)
- Chunking uses cases (e.g., begins with “is”, is a noun phrase, noun phrase + verb phrase, is a constructor)

**Evaluation**
- Ran on a set of 200 sample method names collected randomly from 25 Java programs
- 187 (93.5%) chunked correctly
- 13 (6.5%) chunked incorrectly

**Challenges**
- Part-of-speech tagging iterations on an example method name: `decodeRequest`

**Tagging**
- Most nouns do not follow any pattern and are difficult to identify
- Irregular verbs and adjectives are difficult to identify

**Chunking**
- Words with multiple parts-of-speech can give numerous parses for a single method name
- Syntax of method names are different from English
- There are different types of names which all have differing syntax
- Naming conventions differ between coding styles

**Current Status**
- All tagging rules are implemented
- Chunking constructors and method names starting with “is,” “can,” and “has” is implemented
- Basic phrases and combinations of phrases (NP, VP, NP+VP, NP+PP, etc) implemented

**Next Steps**
- Improve accuracy of tagging irregular nouns, verbs, and adjectives
- Evaluate how field names are used in program code
- Evaluate how common multiword phrases (e.g., “text field”) must be chunked in method names

**Summary**
- Various types of method names are parsed with high accuracy
- Results of research can be used to improve numerous software engineering tools

---

**Research Process**

**Phase 1: Part of Speech Tagging**
Find all parts of speech that can be applied to a given word through prefix and suffix patterns.

**Rule Iterations**
1. If `word` exists, `word` is a noun
2. If `word` and `worded` exist, `word` is verb
3. If `word` begins with “de,” `word` is only a verb, not a noun

**Example**
Part-of-speech tagging iterations on an example method name: `decodeRequest`.

**Phase 2: Phrase Chunking**
Find patterns in syntax to form phrases (i.e. noun, verb, prepositional, and adjective phrases).