Timeline

- April 6: Java API, documentation, code stubs due. Coding begun.
  - First drafts circulated by Monday April 3
  - Including spec for RTW architecture which will use them
  - Form central registry of labels, their meanings, their sources
- April 13: Working project code (standalone) due
  - and integrated system based on code stubs
- April 20: Integrated system based on project code
  - Shortcomings identified
- April 27: Experiment with integrated code, and extensions
- May 4: Final system evaluation
- May 11: Final writeups due, in form to integrate into single unified RtW project report.

Task Threads

- Module development
  - Weeks 1-5
- Module-level evaluation
- Architecture development
- System integration
- End-to-end System Evaluation
- Final write-ups (modules)
- Final project report for the class

Action Items

- Tabulate steps in your module's operation (as per Andy S's example) by Monday
- Document the steps (on Monday or ASAP)
  - Text description, i/o, assumptions, constraints, etc.
  - Data flow diagram (reading/writing from disk, ADB, etc.)
- Domain model / type system
  - EHN distribute GALE type system
  - Set up submeeting to discuss further
    Jon, Jaime, Andy, Kevin, Justin, Nguyen, Laura, Ben, Scott
- EHN draft an overall vision document (Monday)
  - Global version of the tabulation of steps

APIs Posted to Kiva Site

- sfung: Coreferent Resolution API
- hazen: Coreference & Data Flow
- acarlson: Active Learning API
- belamber: Scone API
- jarguell: Relation Extraction API
- rcwang: Entity Association API
- jbetteri: Nominalization SRL initial API / Contribution Spec

The Big Picture

	LEARNING		RUN-TIME	
Module	**Input**	**Output**	**Input**	**Output**
Coreferent Resolution	Person names, web pages containing the name, profile info	Clustered profile vectors for each web page and name pair	Person name, web page containing the name	Information profiles for that name; best profile match
Coreference	Gold-standard data (text and annotations), features, unlabelled data	Classifiers & models	Document	Document with referent and antecedent annotations
Active Learning	Active Learning problem elements conforming to API	Active Learning outcomes	N/A	N/A
Scone API	Store facts that are learned by any module	Updates to shared KB	API calls to query the knowledge base	Relevant facts, relations, etc.
Relation Extraction	Relations & seeds; INDRI index with NE & SENTENCE annotations	Trained model (entity pairs and extraction rules) w/confidence for each relation	A trained model; an INDRI index; a file to annotate	For each relation, the most confident entity pairs that participate in the relation
Entity Association	Seed inputs, types	Entity co-occurrence graph,, ranked entities for each type	Input string, entity type	Ranked entities of that type which co-occur with the input
Nominalized SRL	Text annotated with NE, NP, VP, PP, & SRL information	Trained SRL model for training data	Document	Semantic role labels (annotations)

Levels of Representation in RTW

- Mention level: root text
- Instance level: instance of a KB fact, concept, event, attribute, etc. (e.g. “PERSON_1234”)
- Concept level: abstract class, relation, etc. in the KB (e.g. “PERSON”)
- Recognition_1: linking spans/tags at the mention level to tags at the instance level
- Reference resolution_1: linking mentions that refer to the same instance
- Etc.

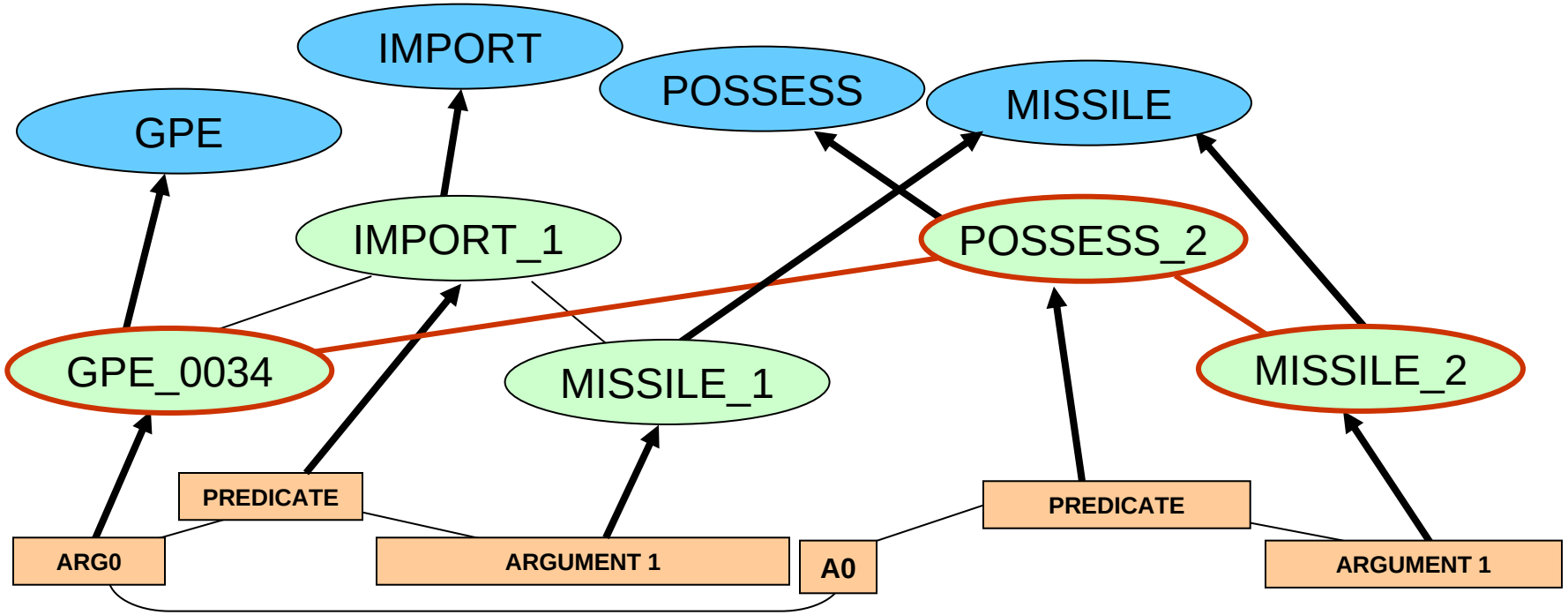
Another form of recognition

- Mentions lead to a local hypothesis of an entity; use EQ links on concepts
- Scone can represent multiple entity hypotheses and weights, with links to possible concepts
- Could be context-specific
- Outputs of name profile module (Nguyen and Simon)

Starting from the raw text

- Based on text & grammatical structure:
  - Annotate “referring expressions”
  - Link RE annotations with “refers_to” attribute
- Based on meaning of the text
  - Creating clusters of attributes
  - Checking clusters in Scone
  - Finding most likely referent for e.g. “Jim”

Co-Reference Resolution

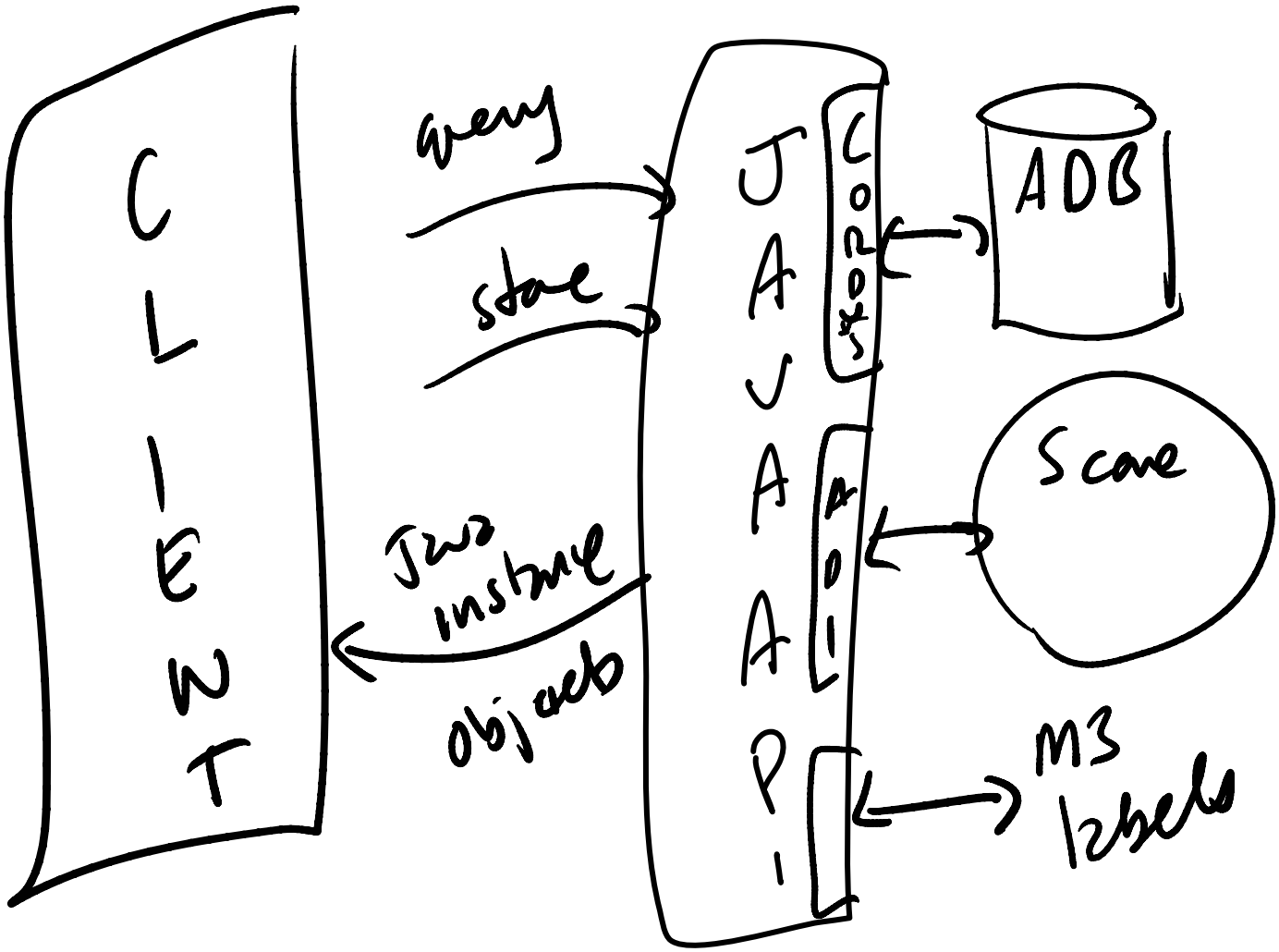


Egypt imports Scud-C missiles. **It** also possesses 50 Scud-Bs.

Heuristic Reference Resolution

- Identify predicate with anaphor in argument
- Search for “earlier” predicates that have potential antecedents
- Use Scone to sanity-check possible antecedents

Possible Approach



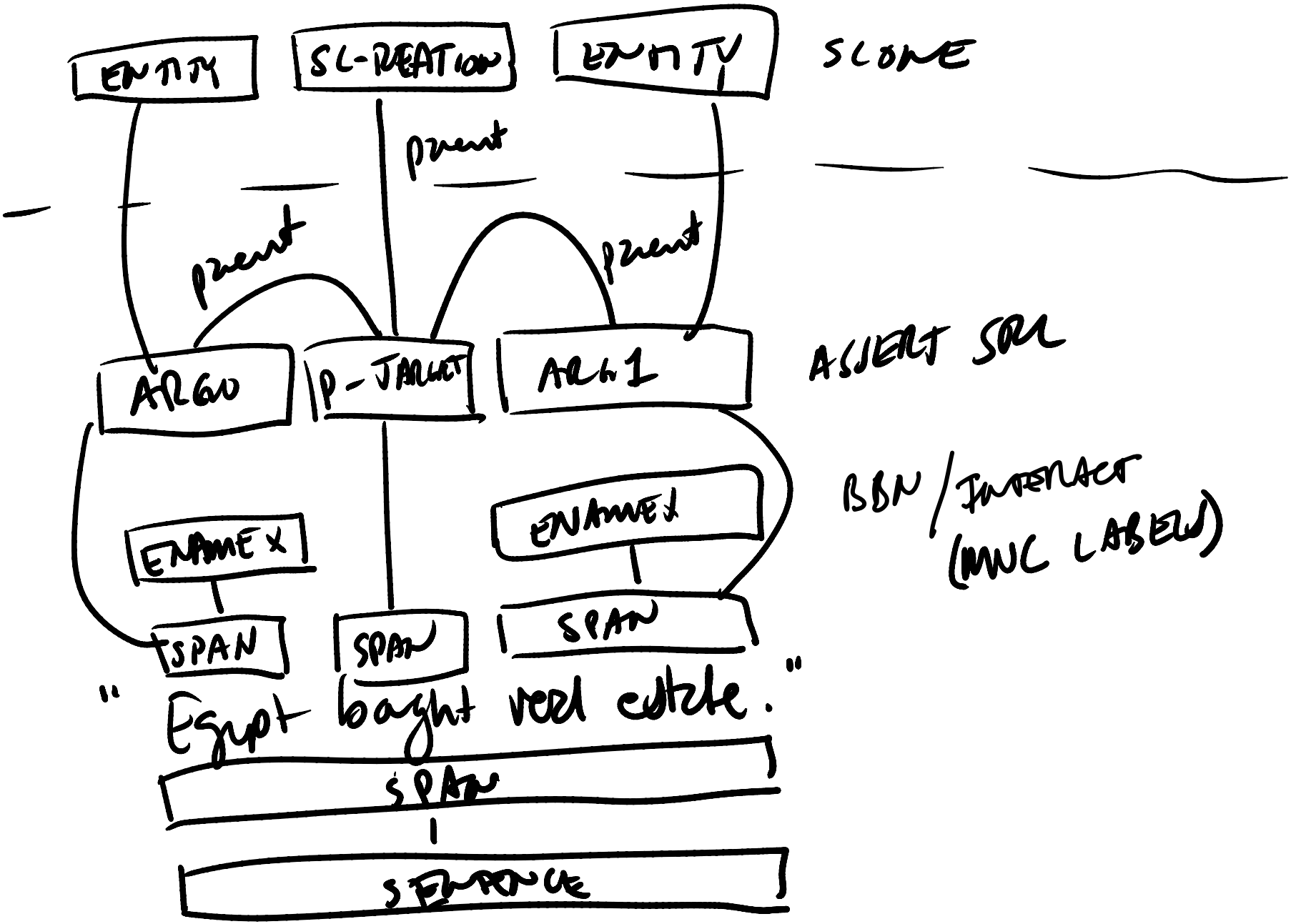
API Calls at Layer Boundaries

```
StoreMentor(RobotMentor);

Concept=FindInstance(RobotMentor);

StoreInstance(RobotMentor, Concept)
```

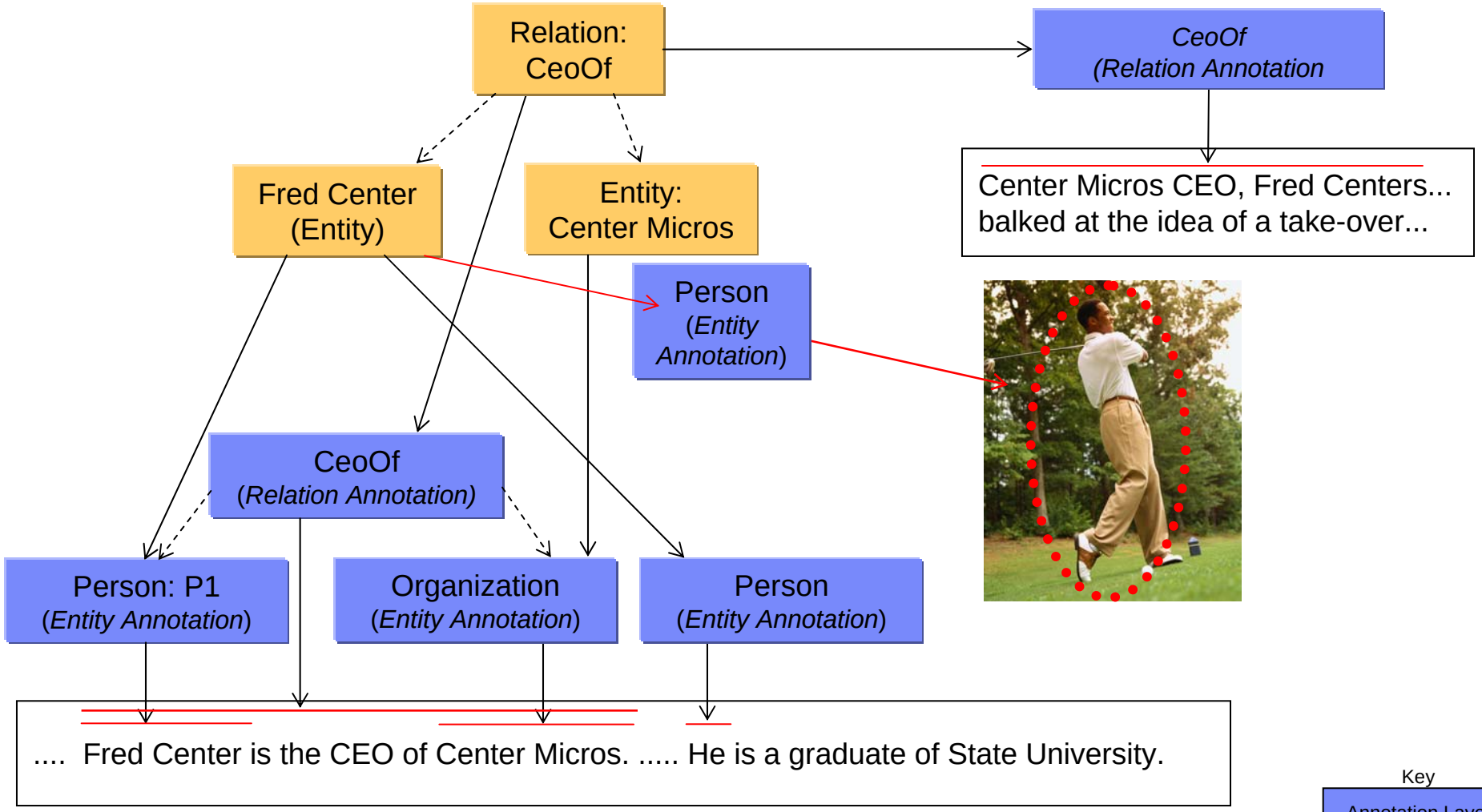
domain
& general
(upper)
SCONE LABELS
"instances"
PERSON
domain
(upper)
ACE LABELS
"mention level"
PERSON-MENTION
text
ADB LABELS
"text"
SPAN, POS, NP
ASSERT. ARG0

Action Items

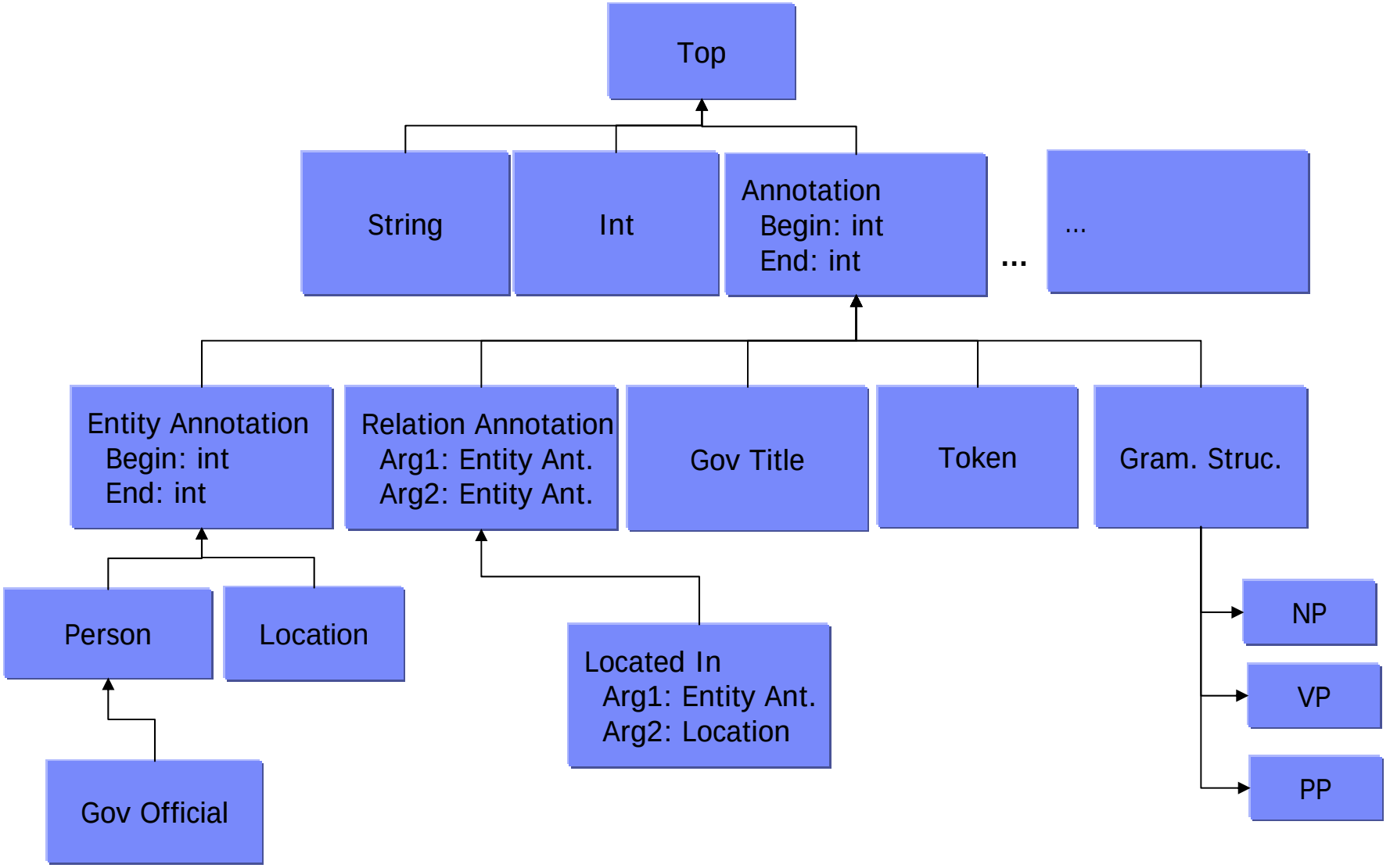
- Get input/output annotations (initial set plus examples) from each team
- Jon provide ACE annotation labels (Kiva)
- EHN provide ENAMEX scheme
- Question: can we bootstrap the RTW type system using ACE & current ADB annotation types?
- EHN: provide an end-to-end example showing how to use existing APIs (ADB, M3 & Scone) to decorate a sample text.


ENTITY
SL-RELATION
ENTITY
SCONE
parent
parent
parent
ARG0
P-OBJECT
ARG1
ASSERT SRL
ENAME X
ENAME 1
BBN / INTERACT
(NNC LABELS)
SPAN
SPAN
SPAN
" Egypt bought reed cattle. "
SPAN
SENTENCE

Annotations and Referents



Sample Type System



Partial *HUTT* Type System (254 concepts in total)