INCOSE 2002 Formalizing a Structured Natural Language Requirements Specification Notation

August 1, 2002

Kendra M.L.Cooper
The University of Texas at Dallas

Topics

- Introduction
  - Motivation
- The SRRS Notation
  - History, brief example
- The Formalization Process
  - IDEF0, 5 steps
- Summary

Introduction

- Since there are numerous formal methods available, why would we want to define another one?
  - Formal notations have numerous advantages:
    - Unambiguous
    - Precise
    - Automated tool support (detect parsing, typechecking, analysis errors; generate test specification/case generation…)
    - ...
  - However, the myths(?) continue that formal methods are difficult to read, write, and understand

Introduction

- Applying a process to a semi-formal notation and formalizing it is appealing
  - it has the potential to offer a *familiar* notation that is precise, unambiguous, and amenable to automated tool support
- Recent research in formalizing semi-formal notations includes work on structured analysis and UML
  - The notations are clearly and precisely defined
  - The process to accomplish the formalization not presented
- So, if we have a semi-formal notation, where would we start if we wanted to formalize it?

SRRS Notation

Update the Notation and Move to Research Environment

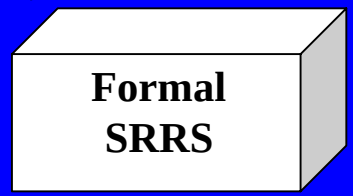
Industrial Notation used on large-scale, complex projects

- Strengths:
- External, Task- oriented organization
- Natural Language
- Template Phrasing
- …
- Weaknesses:
- Stimulus Response matching rules not defined
- Correct use of Capabilities not defined
- No tool support
- …

Formalize the Notation in Research Environment

Research Notation used in small, pilot project in industry

- Modifications
- Stimulus Response matching rules defined
- Threads and Capabilities generalized into Specification Units
- Used by Control group in experiment



Research Notation used in a controlled, lab experiment

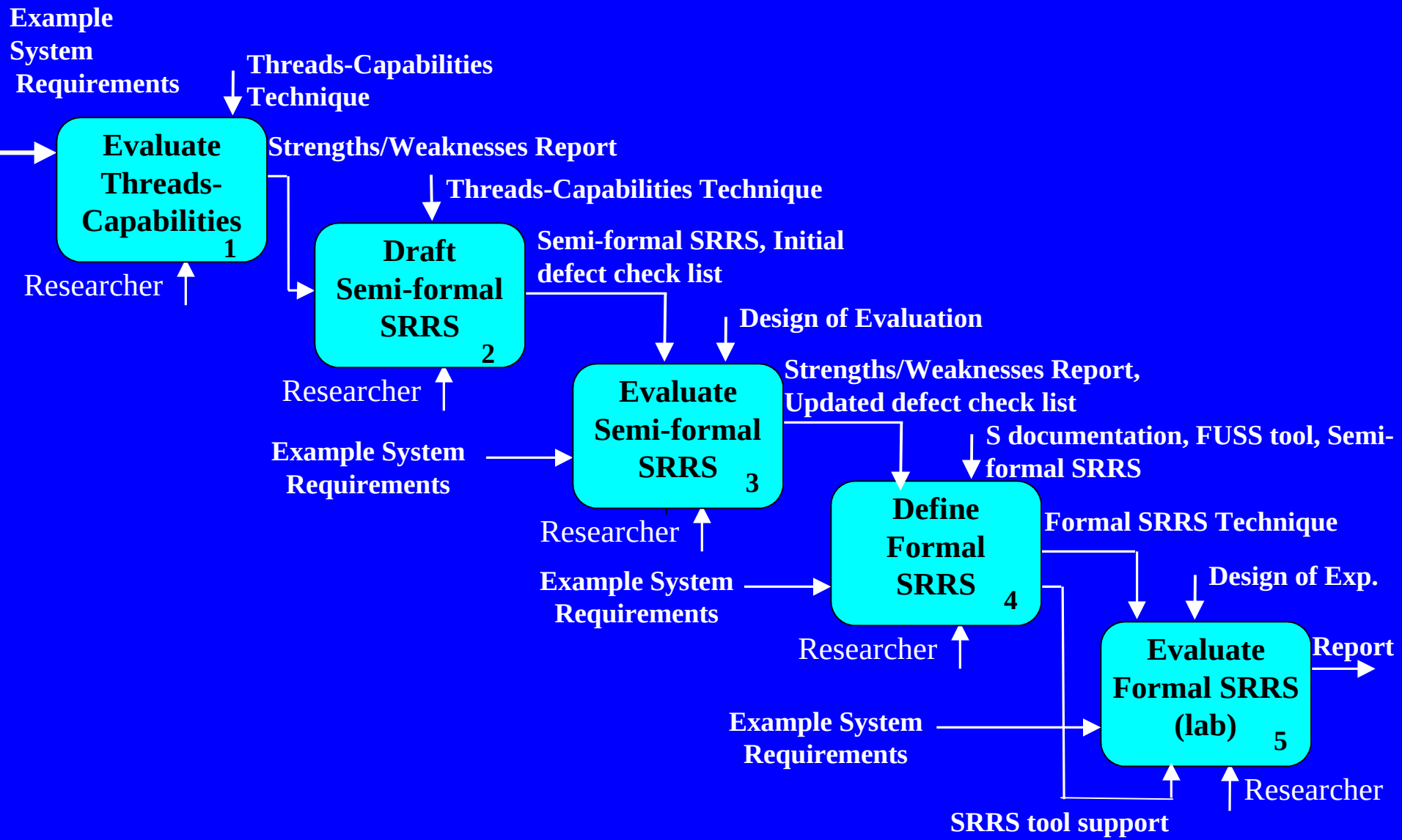
- Modifications
- Syntax and Semantics Formalized
- Tool support developed for automated error checking
- Tool support for automated test generation
- Training material developed
- Used by Experimental group in experiment

SRRS Notation

- **Example Requirement Statement in Semi-formal SRRS**

```
1. Upon receipt of a [sign out book request], if all of
   the following conditions are true:
      a) The borrower is a member of the library
      b) The borrower does not have outstanding
         fines
      c) The book is allowed to be signed out
the system shall send a [charge book response].
```

Formalization Process



Formalization Process

- **Step 1. Evaluate the strengths and weaknesses of the current notation**
  - **Always room for improvement**
  - **Strengths:**

external partitioning of the requirements blackbox style

abstraction, or grouping, of the stimuli and responses

readability of the notation

highly structured format

definition of terminology with a data dictionary

Formalization Process

- **Weaknesses:**
  lack of tool support to assist the authors in detecting defects
  difficulty in describing complex logical conditions
  lack of tool support to automate the development of system level (requirements based) testing
  lack of clearly defined matching rules used to pair a stimulus with a response
  lack of clearly defined re-use mechanism
  lack of training material

Formalization Process

- **Step 2. Draft an updated version of the semi-formal notation**
  - **Select the weaknesses to overcome**
    - **trade-offs with strengths**
  - **Define matching rules, define re-use mechanism, develop tool support (automated error checking and test specification)**
- **Step 3. Evaluate the semi-formal notation**
  - **small case study performed in industry**
  - **4 iterations to write,review, correct a non-proprietary spec.**
  - **Collected data, identified error checks (categories, types)**
  - **89% of all errors detected could be automatically detected with tool support**

Evaluation of the Formalized Notation

- Step 4. Formalize Notation
  - Formalize syntax
    - Describe the grammar in BNF
    - Develop tool support (lex, yacc) to verify
    - >5000 source lines
  - Formalize semantics
    - Select formal notation
      - "S" – a higher order logic
    - Identify what the formal notation offers
      - How to describe data, functions
      - S has 4 kinds of paragraphs (type and constant declarations; type and constant definitions)
    - Manually create the mapping, conversions

Sample Mappings

SRRS Component	S Translation
Title, Overview, Stimuli, Responses	N/A
Type Name Declaration	Constant declaration (Type Declaration)
Subtype Declaration	Constant declarations (type Declaration and function declaration)
…	

Formalization cont.

- Step 4. Cont.
- Develop algorithms to accomplish mappings
  - Implement algorithms, automated error checking, automated translation to test specifications
    - ~14,000 source lines

- Step 5. Evaluate Formal Notation
  - Controlled experiment, lab setting
  - 3 test hypotheses
    - defect rates
    - effort to write/review/correct
    - training time

Summary of Results

- **Reduction in Total Number of defects: 81%**

Group	Number of syntax defects per ROID	Number of type defects per ROID	Number of analysis defects per ROID	Number of total defects per ROID
Group 1	0.99	0.74	0.88	2.61
Group 2	0.09	0.01	0.39	0.49
% Difference	-90.01	-98.65	-55.68	-81.23

Summary of Results

- **Reduction in Average Total Time to Write/Review/Correct: 39%**

Group	Average time to write per ROID minutes	Average time to review and correct per ROID minutes	Average total time per ROID minutes
Group 1	17.58	15.28	32.86
Group 2	10.58	9.42	20.00
% Difference	-39.82	-38.35	-39.14

Summary of Results

- **Increase in Total Training Time: 186%**

Group	Formal Training Time minutes/author	Informal Training Time minutes/ROID	Total Training Time minutes/author
Group 1	420.00	0.32	448.33
Group 2	835.00	5.02	1285
% Difference	98.81	1468.75	186.62

Summary

- Presented a preliminary version of a formalization process
- Need to use this process on a variety of formalization problems and refine/improve it
  - Formalize SRRS using HOL
  - Formalize SRRS using VDM
  - Formalize parts of UML into O-telos
  - …

Author Contact Information

- **Kendra Cooper**
  kcooper@utdallas.edu
  www.utdallas.edu/~kcooper

- **Mabo Ito**
  mito@ece.ubc.ca