SATURN 2007

Automated Requirements Processing Overview

David L. Curry
Software Systems Engineering
NAWC WD, China Lake
760.939.6121
david.l.curry@navy.mil

NAVAIR

Introduction (1 of 2)

Systems continue to become more software-intensive, more complex and, thus more expensive to produce. While great strides have been made in the areas of software development tools, processes and management, the quality, understanding and interpretation of driving requirements has not been sufficiently addressed. Several tools are available that can be of some assistance, including DoDAF, OSATE and ARM. However, each provides only a part of the solution, and none work together directly. Since we have no automated means of converting textual requirements to analytical models, far too much of the system and software design process depends on human interpretation of textual requirements and standards.

Introduction (2 of 2)

1990's-era NASA developed the Automated Requirements Measurement (ARM) tool

- Appears to be older MS BASIC code turned into a Visual Basic project.
- Good at what it does
  - » Parse out requirements and identify weaknesses
- Identifies subjects, but then does nothing with them
- No known tool uses the outputs for further analysis

2007 – The production of software systems is almost entirely dependent upon the ability of many human beings to accurately and completely understand standards and requirements produced by other human beings, usually in complete isolation from one another.

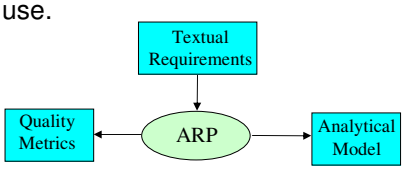
NAVAIR

3

Overview (1 of 2)

What is needed is a tool that can:

- Isolate individual requirements
- Identify system components and attributes
- Identify problematic areas
- Produce a product that is directly importable by an architectural modeling tool such as OSATE
- Learn new requirements statement patterns with minimal user input and no understanding by the user of the underlying technology, thereby becoming more capable with use.



NAVAIR

4

Overview (2 of 2)

The Automated Requirements Processor (ARP) is currently under development at NAWC Weapons Division, China Lake, CA, to address this need

- D. Curry (parsing)
- R. Stallcup (OSATE)

Phase I will

- Integrate existing natural language processing technology and report generation software
- Generate requirements quality report files
- Generate an XML file that can be imported by OSATE

Phase II will

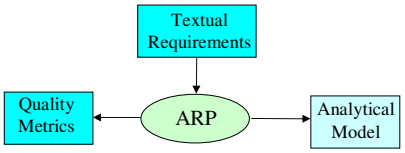
- Refine the requirements parsing capability so that the OSATE model can express the required quality attributes
- Implement a learning capability, so that various writing styles and paragraph identification tokens can accommodated

Goals (1 of 2)

The goal of ARP is NOT to make the tool happy.

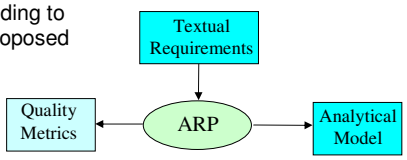
"Section 3" Analysis for Real People

- Increase Requirements Quality
  - » Clarity
  - » Correctness
  - » Completeness
- Reduce
  - » Complexity
  - » Ambiguity
  - » Rework
  - » Negotiation and conflict
  - » Cost, schedule and technical risk
  - » Interpretation by evaluators and designers



Goals (2 of 2)

Generate AADL (OSATE) Models
- Repository of Models for
  - » Standards
  - » Common components and component types (generalization)
  - » Design patterns
- Generation of Program Reference Models
  - » Customers know what they are asking for
  - » Suppliers know what they are bidding to
  - » Evaluators know what is being proposed
  - » Designers know what to design
  - » Analysts can "run the model"
- Ontological Awareness
  - » Identify "what's out there"
  - » Mitigate semantics
  - » Identify opportunities for re-use and optimization



NAVAIR

Approach

Inputs
- Plain text document fragments
  - » Tables and figures ignored

Process
- Identify paragraph identifiers and body
- Parse each paragraph body to sentences
- Parse each sentence to one or more requirements
- Parse each requirement into
  - » Subject
  - » Action(s) and/or State(s)
- Evaluate each requirement by looking for key words
  - » Shall, must = stronger
  - » Should, will = weaker
  - » Could, may = optional
  - » And, or = complex
  - » At least, etc. = incomplete
- Build tables
  - » Subject -> Requirements
  - » One table for each type of deficiency
  - » Sentences that could not be parsed to one or more requirements

Outputs Saved to Files
- Summary results in text file
- Detailed results in text file
- Model in XML file

NAVAIR

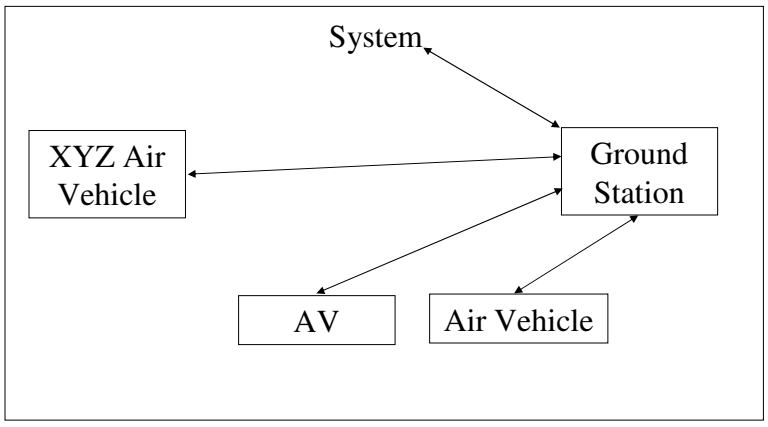
Clarity & Completeness Example

"The XYZ system shall have a range of 2000 to 2500 nm."

- "The XYZ system" is an unmanned air system that includes a ground control station
  - » Does the ground station have to fly 2500 nm also, or can it stay on the ground?
  - » "Air vehicle" might be a better term to use
- "2000 to 2500" – what about 3000 nm?
  - » Not a deficiency, but is this constrictive
  - » Instructions to offerors should prioritize "range" against other quality attributes

NAVAIR 9

Model Example



NAVAIR 10

Actual Summary Output

Summary Report 10/4/2006 15:40:41
Document: XYZ Performance Spec (Section 3)
Version: v1.0 draft 3
Released: 21 Sep 2006

Number of Paragraphs: 240
Document structure depth: 6
Number of sentences: 394
Average sentences per paragraph: 1.64

Number of Requirements: 271
Number of requirements containing SHALL: 249 = 91.88%
Number of otherwise STRONG requirements: 24 = 8.85%
Number of WEAKENED requirements: 67 = 24.72%
Number of OPTIONAL requirements: 14 = 5.16%
Number of INCOMPLETE requirements: 2 = 0.73%
Number of COMPLEX requirements: 209 = 77.12%
Number of phrases not understood: 122

Number of subjects: 111

Actual Detail Output
(Excerpts)

Number of Requirements: 271
- Paragraph #: 3.1.1.4
- Text: The <component> shall <do something> and be supportable.
- Analysis: SHALL COMPLEX

...
Number of requirements containing SHALL: 249 = 91.88%
...
Number of COMPLEX requirements: 209 =77.12%
- Paragraph #: 3.1.1.4
- Text: The < component > shall <do something> and be supportable.
- Rationale: and

Model Output

- Architectural model files comply with Open Source AADL Tool Environment (OSATE) XML schema
- Importable to OSATE
- Gets the ball rolling by quickly producing "the big picture"
- Initial release will only identify components
  - Each object at the same level
- Future releases will add details
  - Associations / Relationships / Heirarchy
  - Associated Quality Attributes
  - Auto-generation of notional graphical output
    - » Persist prior layout work when possible

ARP's Role

ARP should be used by

- Requirements and standards writers to evaluate and improve their own work, and show them what they are specifying
- Suppliers to evaluate RFPs and similarly evaluate their own work
- Evaluators to evaluate proposals
- Designers to help understand more-detailed requirements and designs
- Analysts to help evaluate products against models

ARP should not be used to rate requirements packages

- Doubtful that any body of text will pass through ARP with no "deficiencies"
- Metrics may not be useful for proposal evaluation
- ARP does not have "intelligence" or domain knowledge
- ARP only highlights areas of potential risk
- Making the tool "happy" is not a goal

Connection to Software Architecture

ARP is applicable to all requirements levels
- Enterprise
- System
- Segment
- Component
- Software

ARP will
- Improve the quality of requirements
- Provide reference models
- Motivate population and use of pattern and component model repositories

ARP directly turns textual requirements into a system/software model that can be analyzed and evaluated

Outlook

Phase I FY07
- Basic functionality using regular expression parsing
- Minimal XML document importable to OSATE
- Windows PC only

Phase II FY08
- Implement Link Grammar
- Identify Quality Attributes
- Implement learning capability
- More model details

Follow-on:
- Further parse the requirements and match actions and attributes to predefined Quality Attribute patterns, leading to a fully analytical/executable architectural model in OSATE.
  - » Acquisition engineers will be able to see a nominal model of the system they are specifying prior to RFP.
  - » They will also be able to see a nominal model of any submitted proposal specifications, to aid in evaluation.
  - » Once a vendor is on contract, evaluators will be able to continually update and refine the model based on detailed sub-segment and software specifications.
- Rehost to a cross-platform language
- Ultimately, we expect ARP to become a core component of all NAVAIR software system development

Questions?

NAVAIR

17