Software Requirements

Thomas Alspaugh

Informatics 221

2007 Oct 2

Overview of talk

- Requirements forms
- Requirements activities
- Requirements contexts and appropriate practices
- Active research areas in requirements
- Some current research

Some requirements forms

- Properties — the classic form ("The system shall ...")
- Narratives — the ubiquitous form (scenarios, use cases, user stories, ...)
- Goals (with tradeoffs, relationships)
- Ontologies (describing domain and system)
- Models,usually state models (MSC, SD, LTS, ...)
- Hybrid forms, often tabular (SCR, Problem Frames, ...)

Properties (“shalls”)

- Contractual
- Good for broadly-exhibited characteristics
- Can be good for analysis of later models
- Can be hard to analyze, infer from
- Bad for describing dynamic behavior (except temporal logics, which have their own drawbacks)
- Can be problem for nontechnical stakeholders

Example properties

Common Design System

System v.2 users will be able to transmit input to any Common Design System installation at any location. It will be able to receive output files in the same way.

Lotus Notes

Each local PC with *System* v.2 and Lotus Notes installed can use the automatic message transmittal features to send email and order information to any other email address.

Microsoft Office Applications

System v.2 will be able to save transmittable documents and drawings in a specified directory of the local PC in a file formats compatible with Microsoft Office Applications.

Narratives

- Almost universal (scenarios, use cases, prose)
- Sometimes the primary requirements — especially in the U.S.
- Other forms commonly accompanied by them
- Evocative, partial, concrete, widely understood
- Challenging to integrate, analyze, infer from
- Individual narratives are easy, groups are hard

ATM Scenarios

Contents

1. "Fast Cash"

EVENT CHAIN:

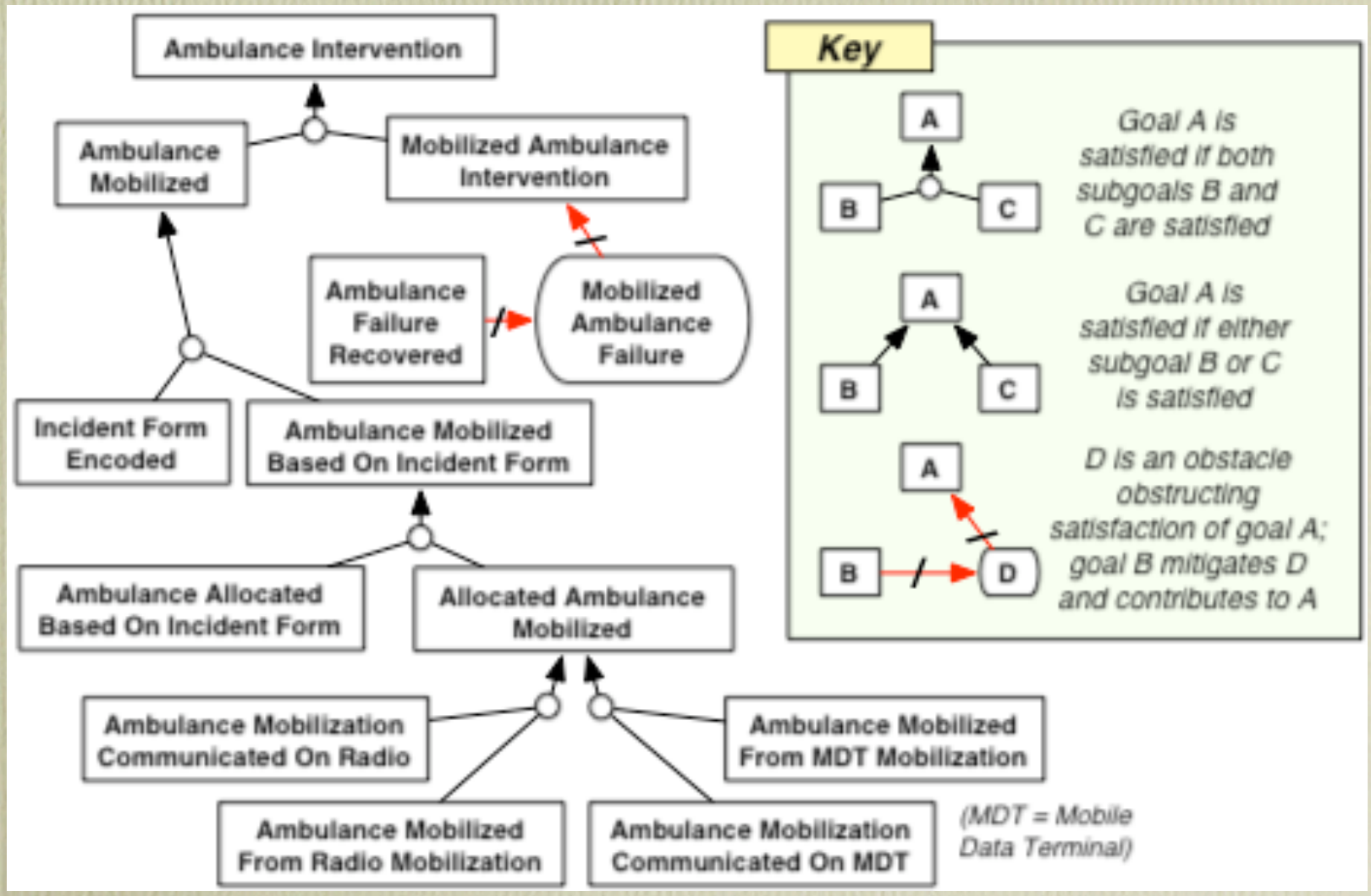
1. The ATM displays a "welcome" screen: "Insert card to begin".
2. A customer inserts an ATM card into the ATM .
3. The ATM displays "Please select your language preference", with choices "English" and "Spanish".
4. The customer selects "English".
5. The ATM displays "Please enter PIN", with choices "Please press cancel if error" and "Press if correct".
6. The customer enters a PIN and chooses "Press if correct".
7. The ATM displays "Please select a transaction. Please press cancel if error. Transfer, Deposit, Payment, Cash Check, Fast Cash From Checking, Withdrawal, Balance Inquiry."
8. The customer selects "Fast Cash From Checking".
9. The ATM sends a query "Checking - $160 OK?" to the credit union .
10. The credit union receives the query "Checking - $160 OK?" .
11. The credit union sends the response "Checking - $160 OK" to the ATM .
12. The ATM receives the response "Checking - $100 OK" .
13. The ATM dispenses $160 .
14. The customer takes the $160 .
15. The ATM displays "Please take your card / Thank you", and ejects the ATM card halfway.

Example narrative

Goals

- Explanatory power — why a requirement is there
- Other kinds of requirements usually are means
- More stable
- Have relationships that can be worked with
- Good for tradeoff analysis
- Stakeholders often more certain of goals

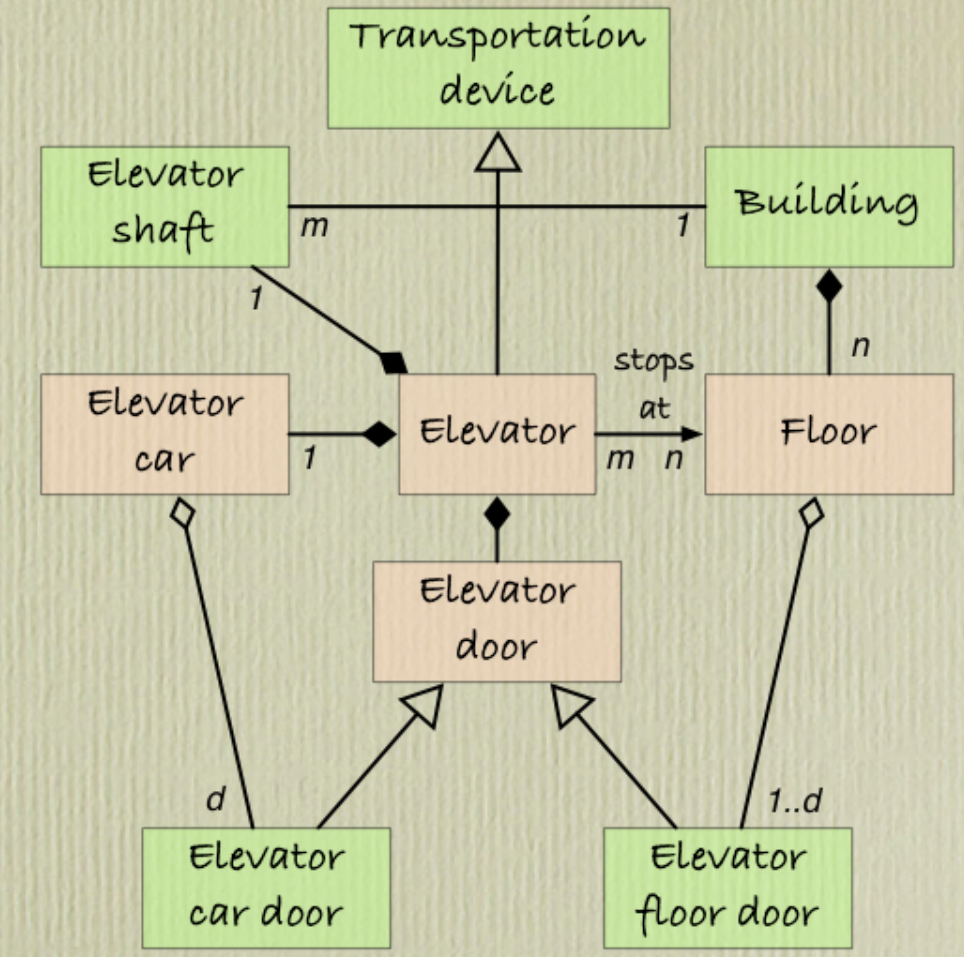
Example goals and relations



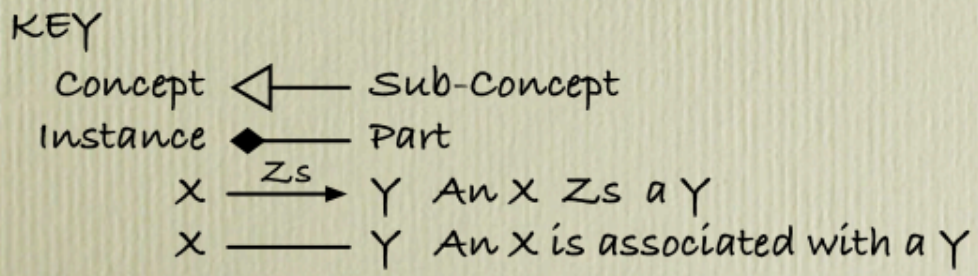
Ontologies

- Entities, sets of entities, relationships
- Define terminology
- Define the shape of the world in question
- Not widely used in requirements (except glossaries)
- Good ontologies are rare

Example ontology



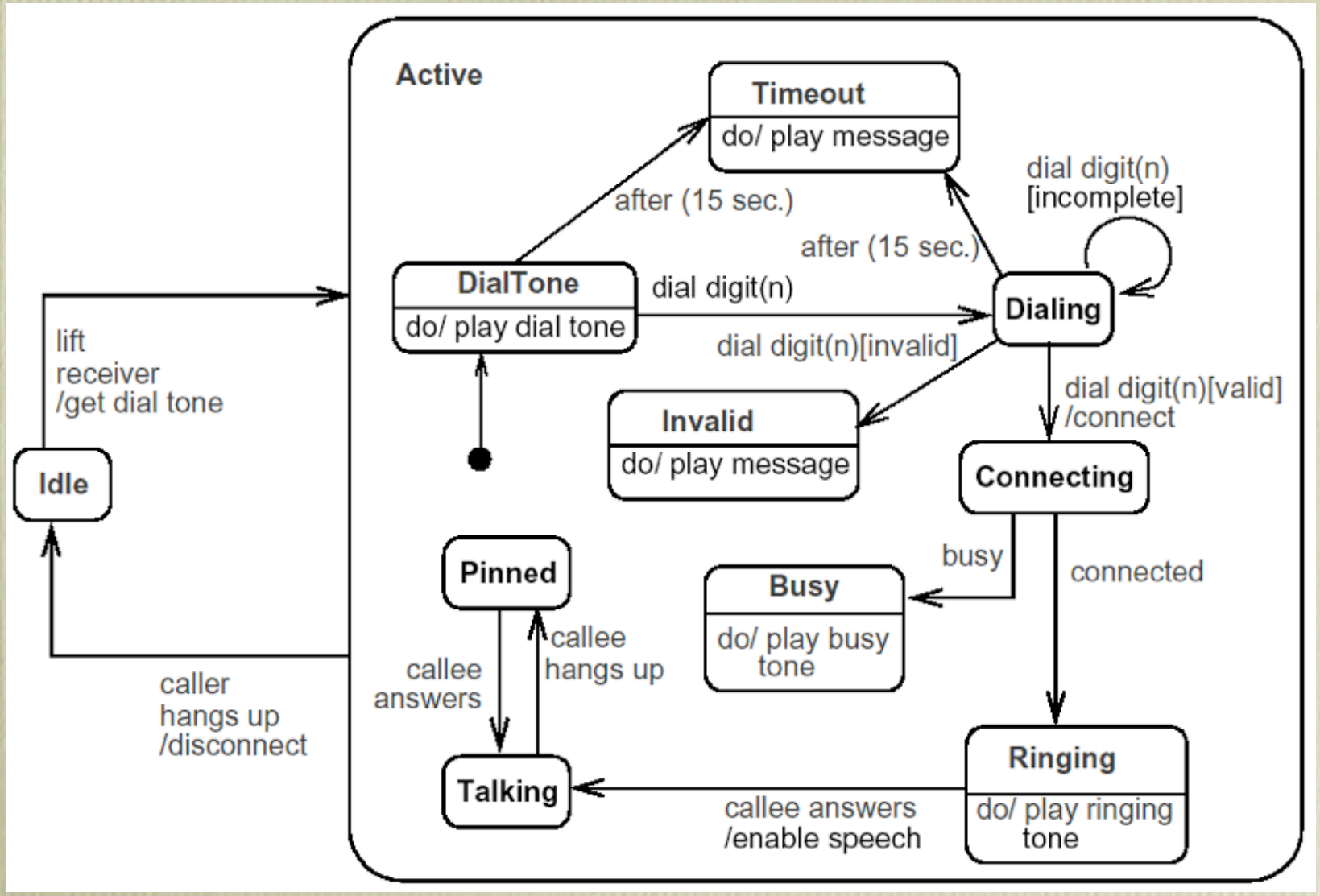
(this example has no glossary)



State models

- Good for analysis; powerful techniques, including model checking
- Especially good for concurrent systems and systems with high failure costs
- Models can be complete —completeness is problematic for all other forms
- Often stray into design
- Require training — stakeholders don't understand

Example state model



Hybrid forms

- Most often tabular
- Organize requirements for ease of reference
- Often integrate two or more forms
- May be analyzable (e.g. SCRTool)
- Usually best for one kind of system —e.g. SCR for embedded realtime
- Problem Frames designed to be flexible

Example hybrid form

Condition Table 4.2-d: Azimuth Cursor Position				
MODES	CONDITIONS			
RadarUpd *BOC* *BOCFlyTo0* *BOCoffset* *SBOC* *SBOCFlyTo0* *SBOCoffset*	90° < BRG < -90°	-90° ≤ BRG < -45°	-45° ≤ BRG ≤ 45°	45° < BRG ≤ 90°
CURSOR POSITION	out of view	left edge	BRG° from center	right edge

Requirements activities

- Elicitation
- Analysis (inference; formal properties)
- Presentation (esp. written)
- Negotiation
- Evolution (esp. throughout development process)
- Integration into other phases (e.g. testing)

What requirements are good for (or should be)

- Communication among all parties involved
- Stakeholder input, agreement, buy-in
- Analysis, inference, tradeoffs at inexpensive time
- Light showing where the end of the tunnel lies
- Context for all subsequent refinements, choices
- Criteria for testing, buyer satisfaction, sign-off

Arguments against and for

- Against: *Requirements are hard!*
- Against: *Requirements evolve, so why bother*
- Against: *Requirements don't reflect implementation*
- For: *If you don't know where you're headed ...*
- For: *The decisions you don't realize you make ...*
- For: *You can't recapture the requirements later*
- For: *Stakeholders understand requirements, only*
- For: *Requirements are cheap and effective*

The classical requirements context

- Big, expensive, one-off system —hundreds of developers working for years
- Developed on contract: customers vs. developers
- Waterfall model, Boehm statistics
- Ineffective tool support
- Lawyers, project managers, accountants
- The development process is an ocean liner

Most systems aren't developed like that

Dimensions of requirements context

- Novelty —*domain, system, implementation*
- Total cost of system development
  —*Requirements effort usually proportional (10-50%)*
- Cost of failing to meet requirements
  —*Not necessarily related to development cost*
- Stakeholder characteristics
  —*What form of requirements is effective for them?*
- System characteristics / Stakeholder goals
  —*How can what's important be expressed?*

Four contexts

- Project expensive, system failures expensive
  —*ATC*
- System failures expensive
  —*fly-by-wire, medical systems, HIPAA*
- Project moderate, system failures cheap, stakeholders nontechnical
  —*many business systems, most PC software*
- Small project, system failures inexpensive, system domain complex, medium to high novelty
  —*Embedded controllers, some business systems*

Context #1: expensive, high cost of failure

- Goals for tradeoffs, focus, rationale
- Models for convincing analysis of consequences
- Properties for contractual force
- Narratives to explain contexts, give immediacy
- Ontology (or at least glossary) for agreement

#2: high cost of failure

- *Similar, but different emphases*
- Models for convincing analysis of consequences
- Properties and narratives for verification
- Narratives to explain contexts, “same page”
- Ontology for domain understanding
- Goals for rationale, tradeoffs, focus

#3: limited failure cost, nontechnical stakeholders

- Narratives as primary form
- Ontology for domain understanding
- Goals for exploration, tradeoffs, rationales
- No properties (or few), no models

#4: small system, limited failure cost, complex domain

- *XP: 10 or fewer, highly-skilled, a year or less*
- Domain expert sitting with developers
- Requirements = the tests (specialized narratives)
- Implementation is what's analyzed
- Evolution expected, welcomed (in implementation)
- Requirements activities distributed throughout development, in small chunks

Hot research areas (RE'07)

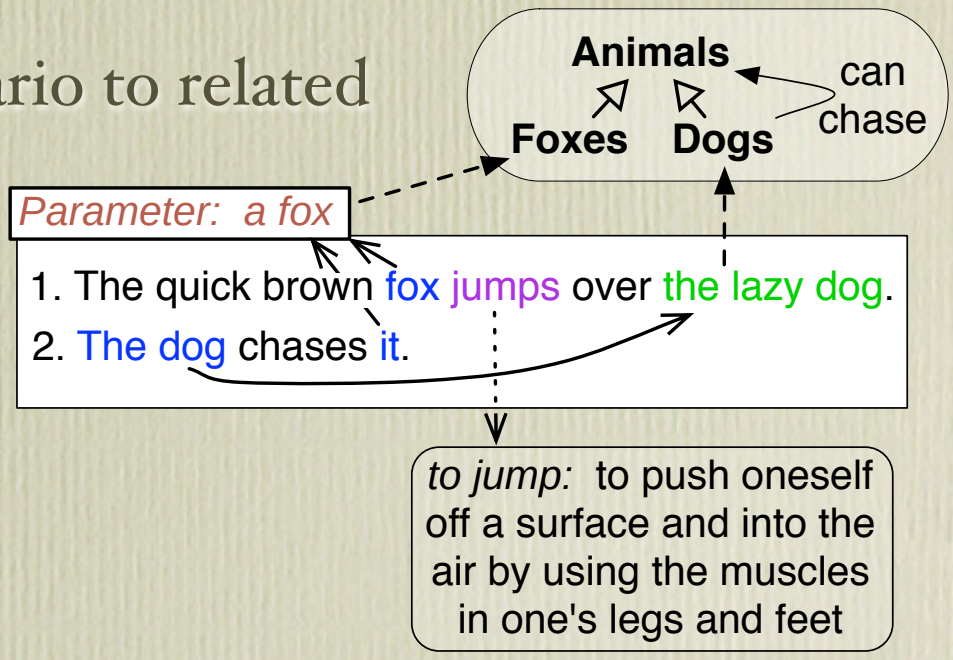
- The business view of requirements
- Globalization (highlighted in the CfP)
- Natural language processing
- Evaluating effects of requirements practices
- Goals, i*, scenarios
- Problem modelling, not behavior modelling
- Product line engineering (an isolated world)

My current research

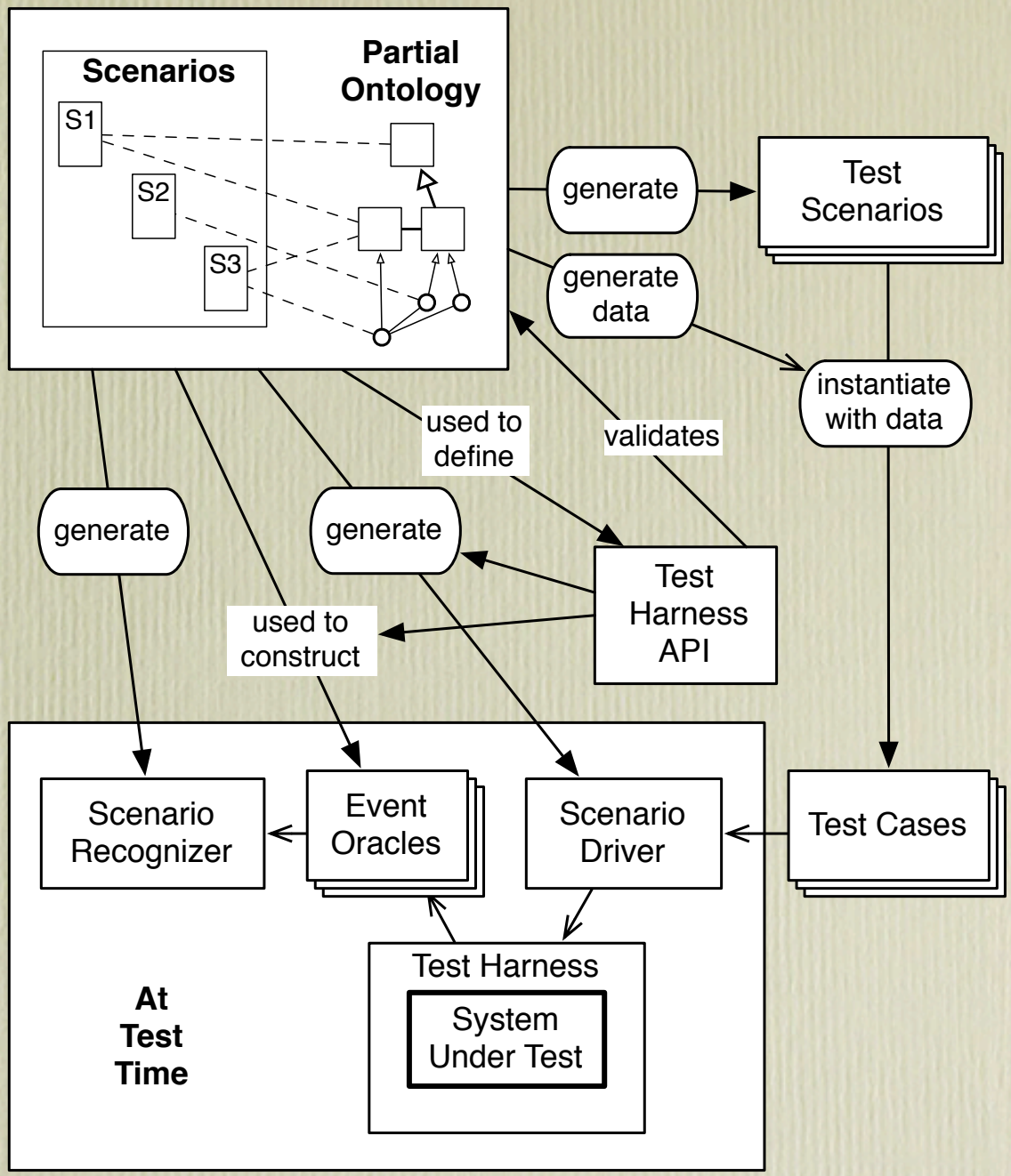
- Scenarios and the informal/formal boundary
- Automation with scenarios and of scenario work
- Scenarios and ontologies of their worlds
- Scenario-driven specification-based testing
- Scenarios and social interactions

Scenarios and ontologies

- A more exploratory view of ontologies
- Ontology describes structure of scenarios' world
- Connect parts of scenario to related parts of ontology
- Enhanced automation, semantic connection



Scenarios and testing



Scenarios as data structure

- Computed social worlds
- Social interactions driven by, recalled as scenarios
- Implementation uses ScenarioML scenarios and software for manipulating them
- Work with Bill Tomlinson and Eric Baumer

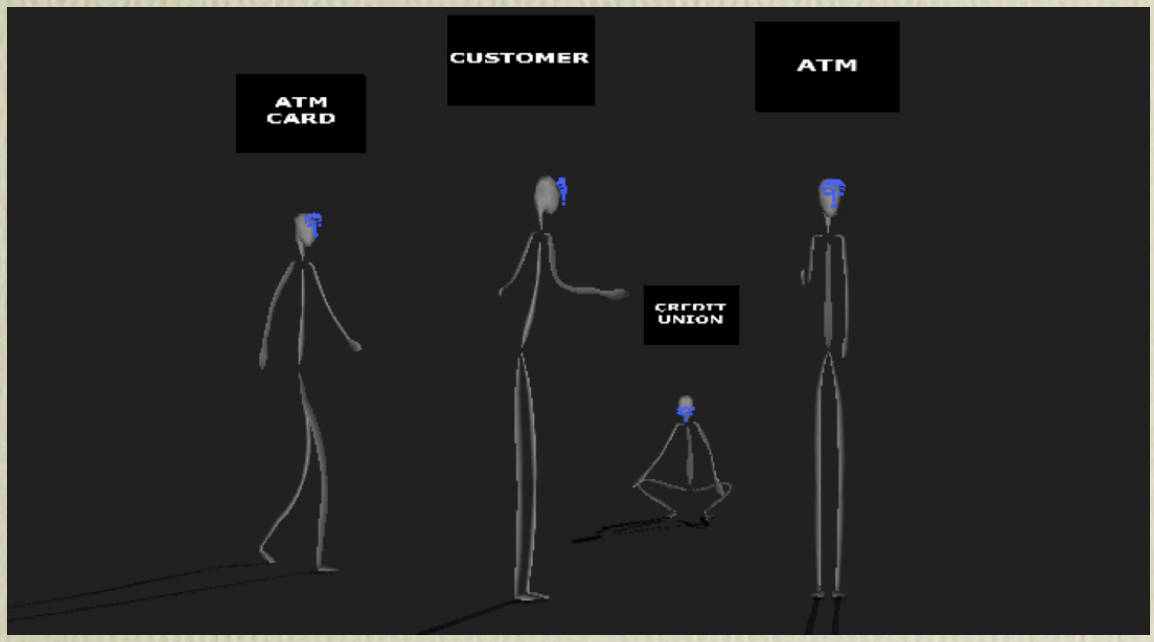
http://orchid.calit2.uci.edu/~ebaumer/aiide06/BaumerEtAlAIIDE06.mov

"feather"

"Hey, nice fire" "Thank you!"

Stakeholder visualizations

- Visualization created in real time, for almost-free
- Stakeholders understand better (dual-coding effect)
- Different audience, novel interactions, new ways
- Work with Bill Tomlinson and Eric Baumer



http://orchid.calit2.uci.edu/~wmt/movies/softvis.mov

More information

http://www.ics.uci.edu/~alspaugh/