Using Process Definition and Analysis Techniques to Reduce Errors and Improve Efficiency in the Delivery of Healthcare

Leon J. Osterweil

(with Lori A. Clarke and George S. Avrunin)

Department of Computer Science

University of Massachusetts Amherst

[1] This material is based upon work supported by the National Science Foundation under Awards CCF-0820198, CCF-0905530 and IIS-0705772, and by a gift from the Baystate Medical Center, Rays of Hope Foundation.

The Large Issue: Continuous Improvement of Human-Intensive Systems

- How to deal with systems where the human contributions
  - Require considerable domain expertise
  - Have significant impact on the overall success or failure of the outcome
- Involve complex collaborations among large and changing configurations of humans, software systems, and hardware devices
- Can't be specified in detail all at once—maybe not ever
- Examples:
  - Healthcare: chemotherapy administration, blood transfusion, emergency department activities
  - Government activities: elections, dispute resolution, emergency response, search and rescue
  - Manufacturing and finance
  - Engineering and scientific investigation

Our Approach: Analysis Technology Support for *Continuous Improvement*

- Model systems to some level of detail
- Evaluate them
  - Using a wide variety of testing and analysis techniques
- Propose elaborations, modifications, enhancements
- Deploy them: Model-guided support
- Reevaluate in the domain setting and iterate

Shewhart/Deming Cycle

Various Drivers for Iterations

- New understandings of system details
- Defects detected
- Changes or additions
- Clearer understandings of emergent behaviors
- Involvement of new people with new behaviors or perspectives
- Changed environment or doctrine

Approach: Employ an Integrated Collection of Technologies Designed to Model and Analyze Human-Intensive Systems

- Powerful, rigorous, articulate language for modeling systems
  - Little-JIL system specification language
- Requirements engineering to capture properties
  - PROPEL (property elucidation system)
- Model checking to detect errors
  - FLAVERS (Flow Analysis for Verifying Systems)
- Safety analysis to reveal vulnerabilities
- Discrete event simulation to improve efficiency

Modeling Human-Intensive Systems

- Language requirements
  - Capture complexity of systems clearly, cleanly, in detail
  - Rich semantics
    (e.g. functionality, concurrency, resource utilization, exceptions, human participation)
  - Precise semantics to support static analysis, simulations, and executions
  - Understandable to the domain experts (facilitate validation that the definition models reality)

The Little-JIL Process Definition Language

- Blends proactive and reactive control
- Coordinates human and automated agents
- Emphasizes exception specification, management
- Facilities for abstraction, scoping, hierarchy
- Supports artifact flow
- Concurrency, synchronization with message-passing
- Articulate specification of resources
- Steps have agents that can be humans, software, hardware
- Semantics for aborting steps
- Pre/post condition constructs
- Facilities for human choice

Hierarchy, Scoping, and Abstraction in Little-JIL

- Definition is a hierarchical decomposition
- Think of steps as procedure invocations
  - They define scopes
  - Copy and restore argument semantics
- Encourages use of abstraction
  - Eg. system fragment reuse

Exception Handling: A Special Focus of Little-JIL

- Steps may have one or more exception handlers
- Handlers are steps themselves
  - With parameter flow
- React to exceptions thrown in descendent steps
  - By Pre- or Post-requisites
  - Or by Agents
- Four different continuations

Artifact flow

- Primarily along parent-child edges
  - As procedure invocation parameters
  - Passed to exception handlers too
  - Often omitted from coordination diagrams to reduce visual clutter
- This is inadequate
  - Artifacts also need to flow laterally
  - And subtasks need to communicate with each other

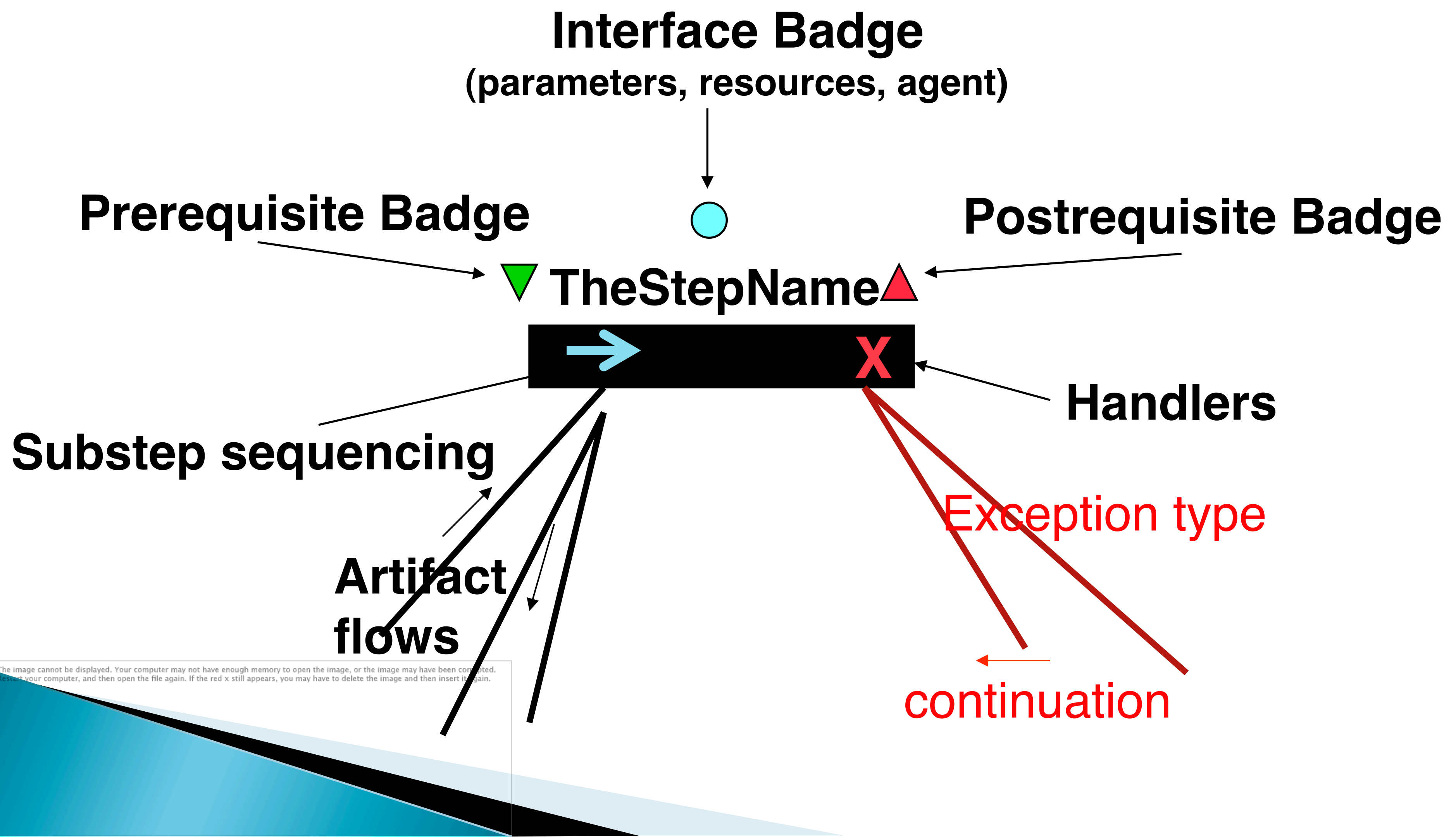
Resources

- Entities needed in order to perform step
- Step specifies resource needed as a type
  - Perhaps with attributes, qualifiers
- An EHR is a resource
- Resource instances bound at runtime
- Exception thrown when “resource unavailable”

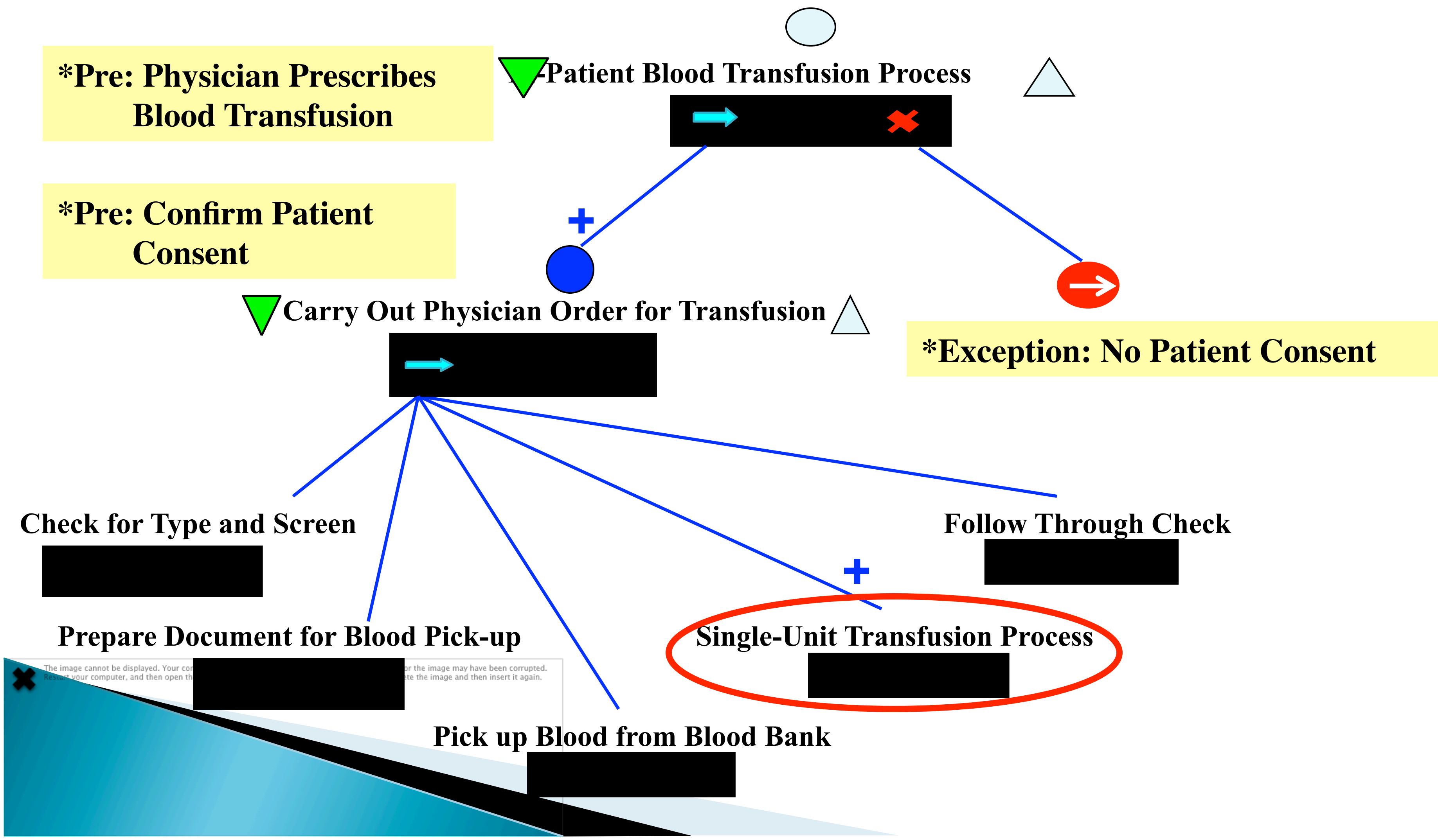
Agents

- Collection of all entities that can perform a step
  - Human or automated
- System definition is orthogonal to assignments of agents to steps
  - Path to automation of system model
- Have freedom to execute leaf steps in any way they want

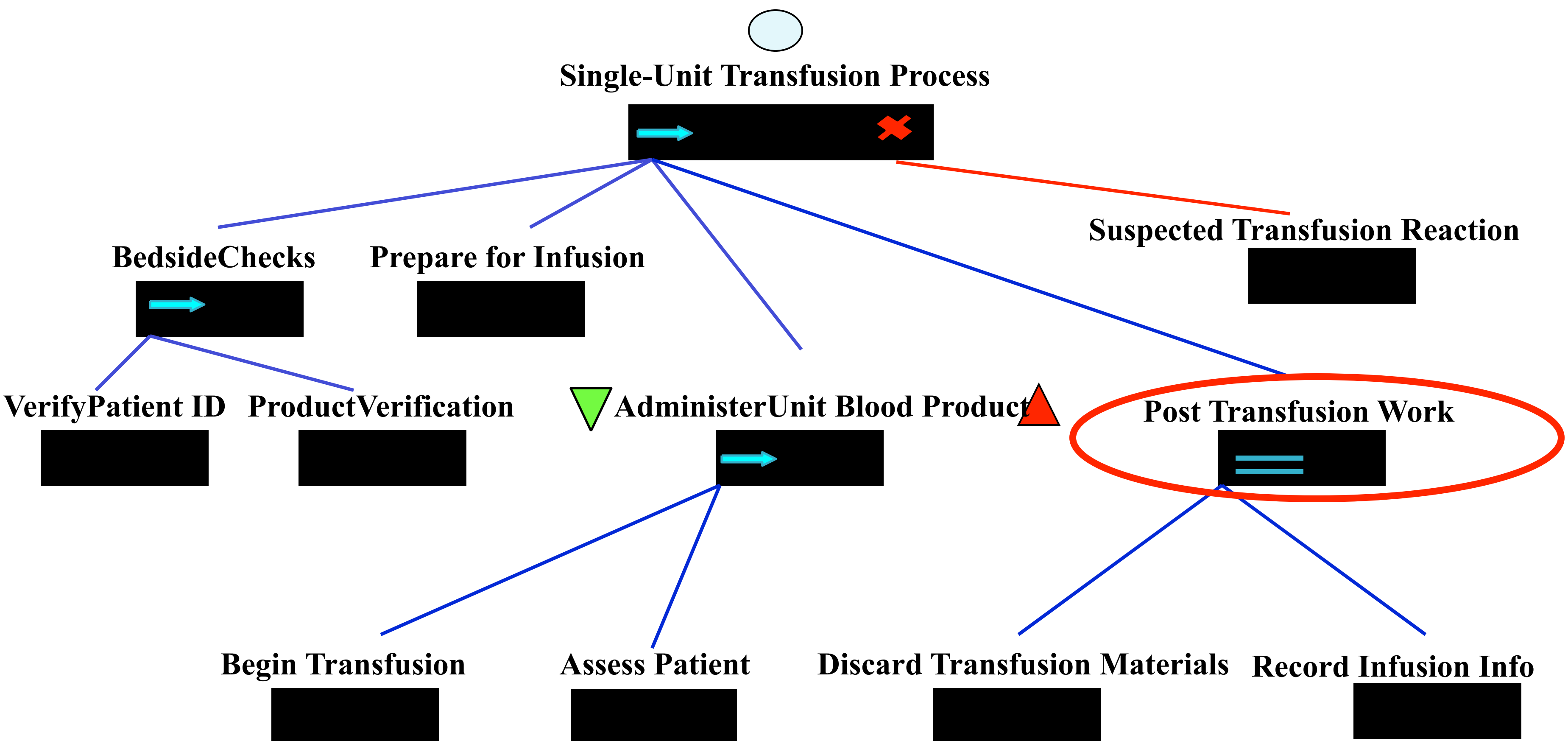
“Step” is the central Little-JIL abstraction



In-Patient Blood Transfusion



Single-Unit Transfusion Process



The image cannot be displayed. Your computer may not have enough memory to open the image, or the image may have been corrupted. Restart your computer, and then open the file again. If the red x still appears, you may have to delete the image and then insert it again.

Narrative View and ToC

3. Order Test(s)

3.1 order test(s) on computer

3.1.1 log into computer

3.2.1 select patient record

3.2.1.1 look for patient name on the alphabetical list

3.2.1.2 match additional info as needed (age, gender, complaint, location...)

- ...

3. **Order Test(s)**(part of perform Blood Specimen Labeling process)

To perform this step the *Provider* must have the *patient-name*.

The *Provider* should first order test(s) on computer,

and then order test(s) on patient chart.

During any of these steps, if the required resources are not available, *order test(s)* is considered to have failed.

Upon successful completion of this step,

continue to perform Blood Specimen Labeling process by proceeding to the next step in the sequence.

3.1 Order Test(s) on Computer (part of order test(s))

To perform this step *the Prov*ider must have *the patient*-name and the *CIS system*.

To order test(s) on computer the *Pro*vider should perform, in order, each of the following:

log into computer

select patient record in DB

verify the selected patient exactly matches desired patient

select test to order at least once

digitally sign the order(s)

During any of these steps, if the required resources are not available, *order test(s) on computer* is considered to have failed.

Upon successful completion of this step, continue order test(s) by proceeding to the order test(s) on patient chart step.

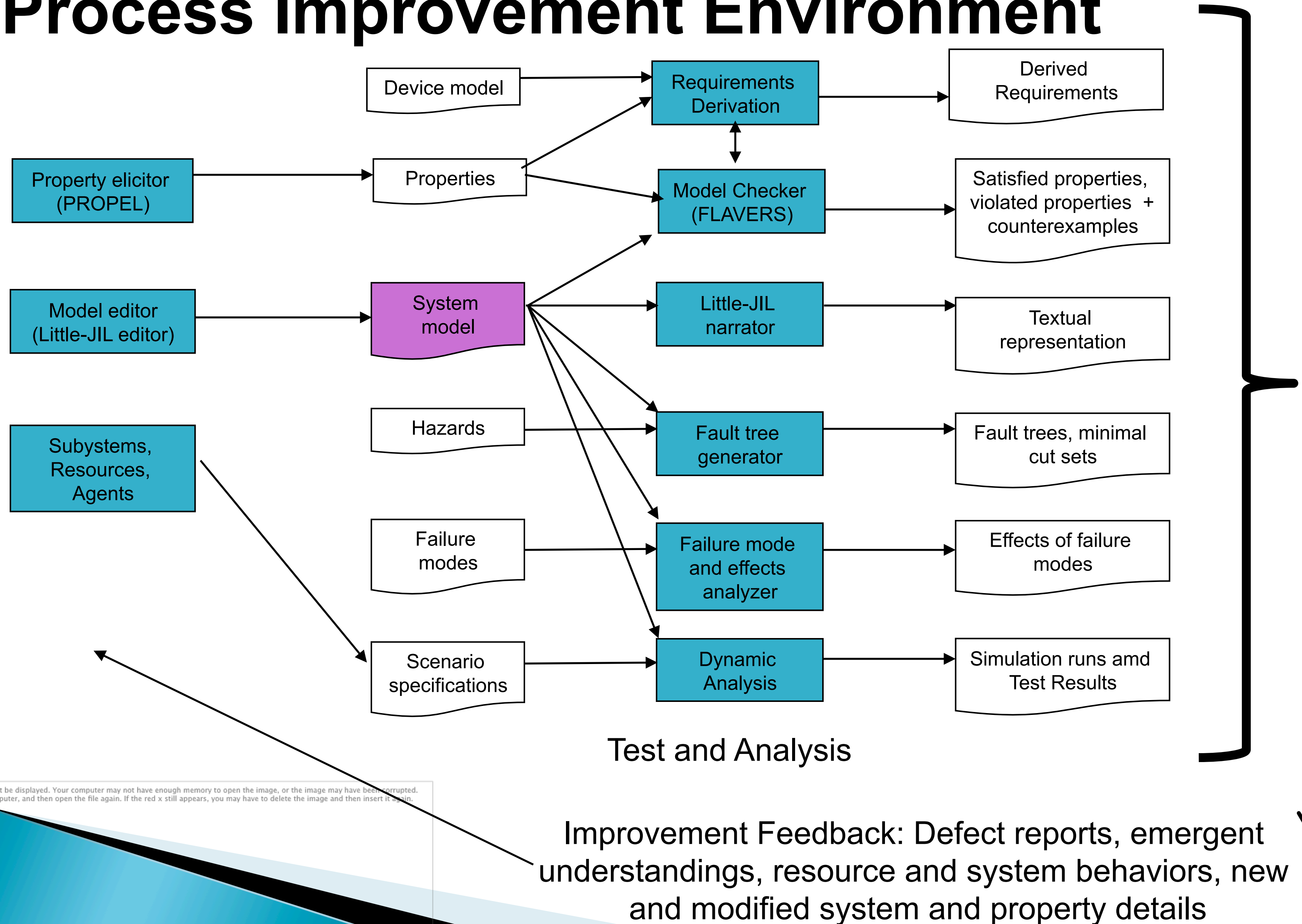
System Modeling Observations

- Systems are not well-understood
  - Individuals know their own activities, but misunderstand how they relate to others
    e.g., Artifacts created but not used
  - Need abstraction and hierarchical decomposition
- Iteratively add detail based on emerging concerns
  - Decisions about upper and lower bounds of the scope may change
  - Decisions about granularity of task decomposition may change
- Features of the language help guide the elicitation
  - E.g., What exceptions can arise and how are they treated?
- Need to consider specifications of desired behavior

Testing and Analysis of System Models

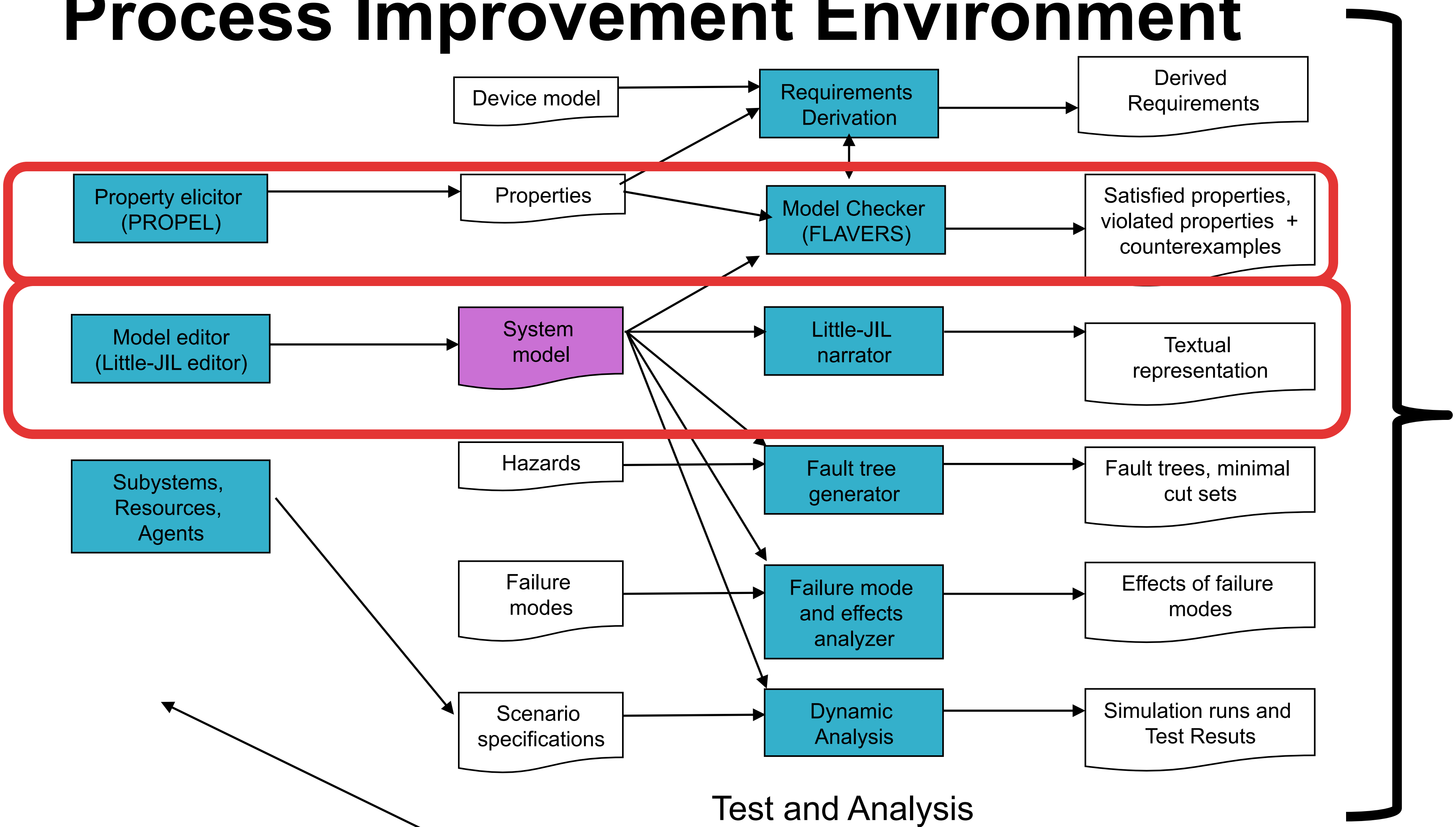
- **Model checking** to find erroneous sequences of events and system states
- **Failures mode and effects analysis** (FMEA) to determine how faults propagate through a system and lead to hazards
- **Fault tree analysis** (FTA) to find combinations of faulty events or agents events that lead to a hazardous situation
- **Discrete Event Simulation** to evaluate resource utilization and performance
- **Requirements generation** to automatically determine requirements for families of components; safe system composition and substitution

Iterative Improvement Driven by a Process Improvement Environment



The image cannot be displayed. Your computer may not have enough memory to open the image, or the image may have been corrupted. Restart your computer, and then open the file again. If the red x still appears, you may have to delete the image and then insert it again.

Iterative Improvement Driven by a Process Improvement Environment

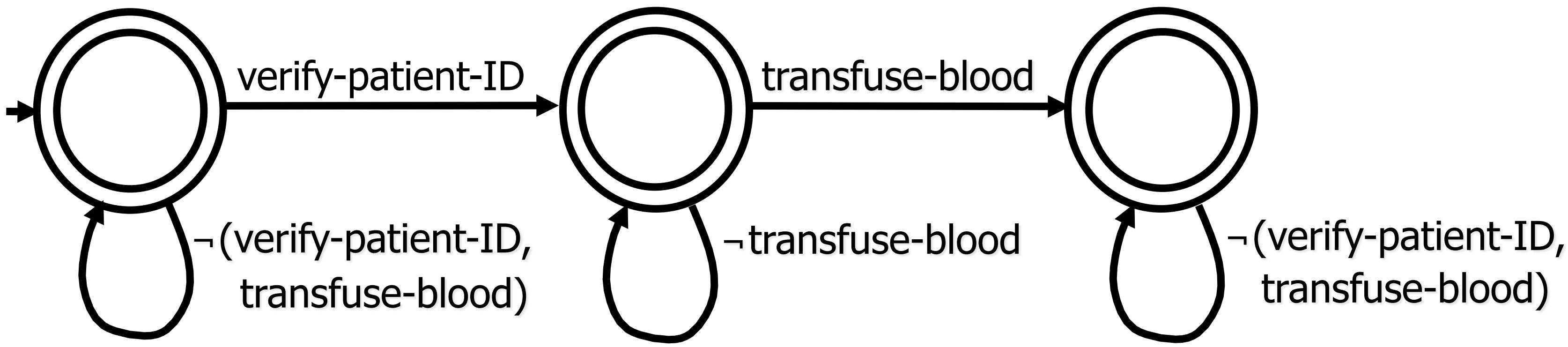


Improvement Feedback: Defect reports, emergent understandings, resource and system behaviors, new and modified system and property details

Example Property

The patient's identification must be verified prior to transfusing each unit of blood product.

A Specific, Detailed Elaboration



transfuse-blood cannot occur unless verify-patient-ID has already occurred.

It is acceptable for verify-patient-ID to not occur, but if it does not occur then transfuse-blood can never occur. Even if verify-patient-ID does occur, transfuse-blood is not required to occur.

Before the first verify-patient-ID occurs, the events in this property, other than transfuse-blood, can occur any number of times.

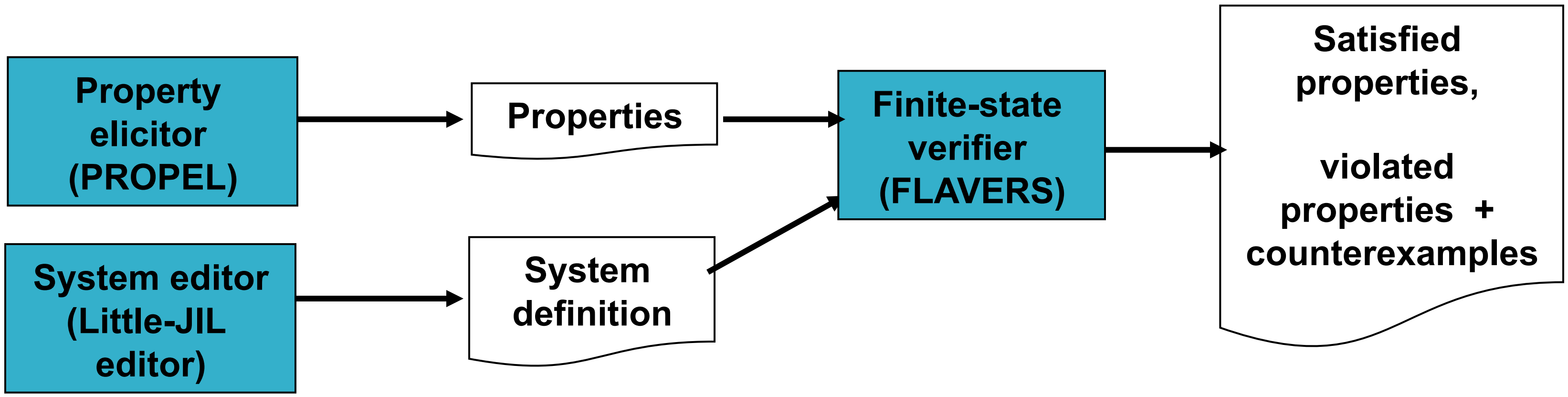
After verify-patient-ID occurs and before the first subsequent transfuse-blood occurs:

- the events in this property, including verify-patient-ID but not transfuse-blood, can occur any number of times.

After the first subsequent transfuse-blood occurs:

- the events in this property, other than verify-patient-ID or transfuse-blood, could occur any number of times;
- neither verify-patient-ID nor transfuse-blood can occur again.

Model Checking

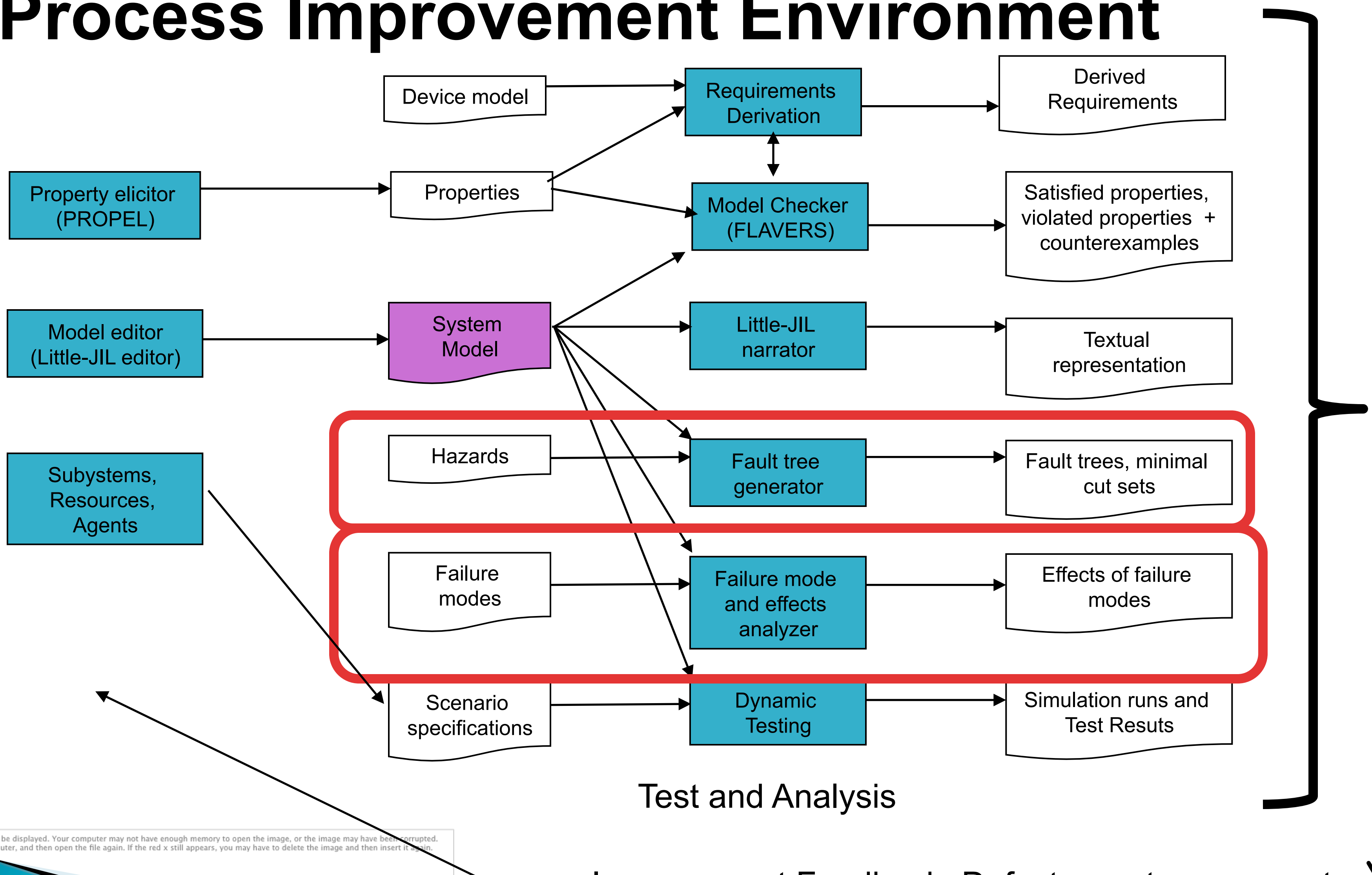


- Are there ***any*** traces through the system model that will violate a property?
  - e.g., is it possible for a required event to ever be missed or done out of order?
  - If so, provides counterexample traces
- Example errors
  - Deadlock - nurse waiting for bloodbank, bloodbank waiting for nurse
  - Missed event - no update on height on weight

Observations about Verifying Models

- Just doing the modeling helped uncover errors in the systems
- Initially mostly found errors in models and properties
- After fixing the modeling errors, we found errors in the real systems
  - Stale height and weight
  - Deadlock
- Fixing the errors often led to other errors
- ***If systems are complex enough to be modeled, the models must be carefully validated!***

Iterative Improvement Driven by a Process Improvement Environment



Improvement Feedback: Defect reports, emergent understandings, resource and system behaviors, new and modified system and property details

Faults versus Vulnerabilities

- **Model checking** assumes that the stated tasks are done correctly, but tries to determine if the tasks are always done in the right order with the right values

- **Safety analysis** tries to determine what harm might be done if the tasks are not done correctly
  - **Failure mode and effects analysis**
    - What hazards might arise, if there is a failure in the system?
  - **Fault tree analysis**
    - What are the ways in which a particular hazard might occur

Fault Tree Analysis (FTA)

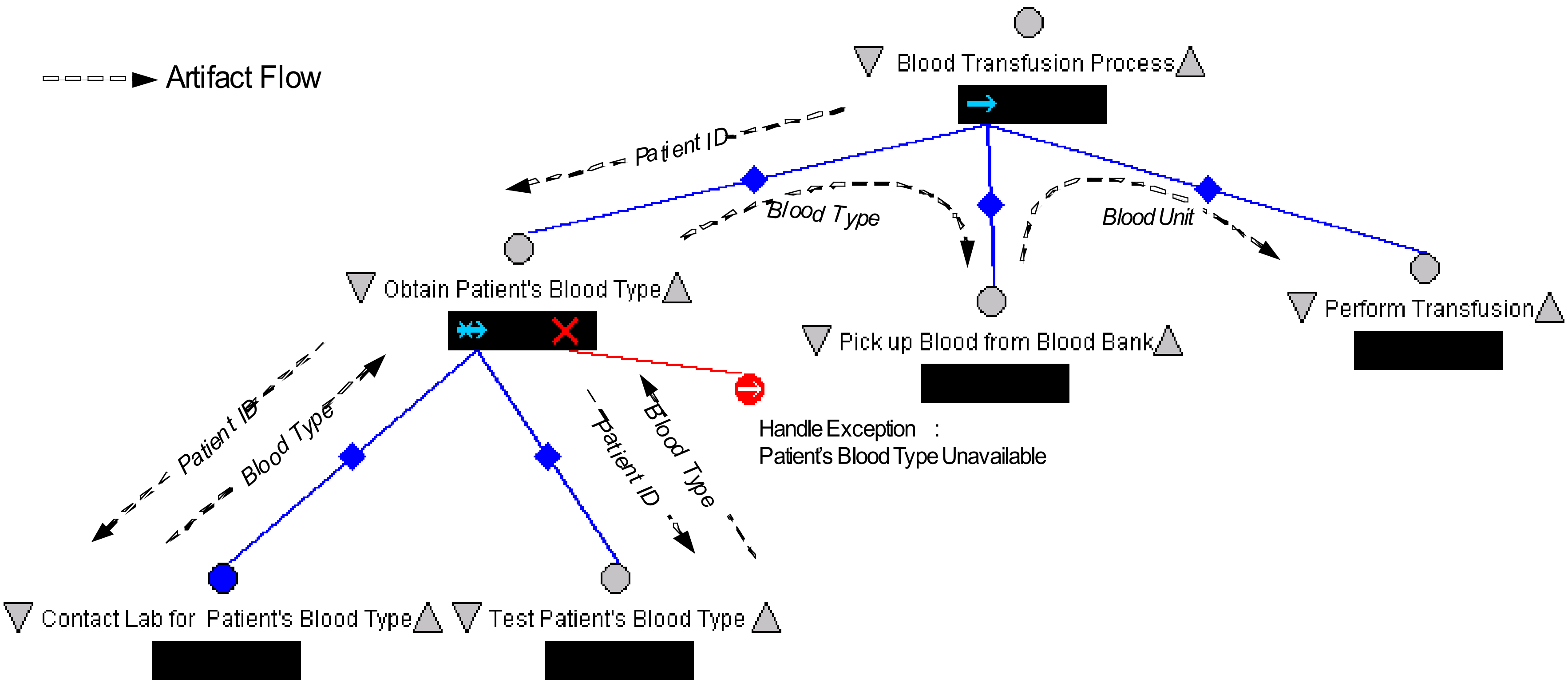
- A well accepted and widely practiced safety analysis technique that identifies all possible combinations of events that could lead to a given *hazard*
  - Hazard: A condition in which loss of life or serious loss of property becomes possible
- Approach
  - Specify a hazard that is of concern
  - Create a fault tree for that hazard
  - Derive Minimal Cut Sets (MCSs)--minimal event combinations that can cause the hazard

Our Approach: Generate Fault Tree from the System Model

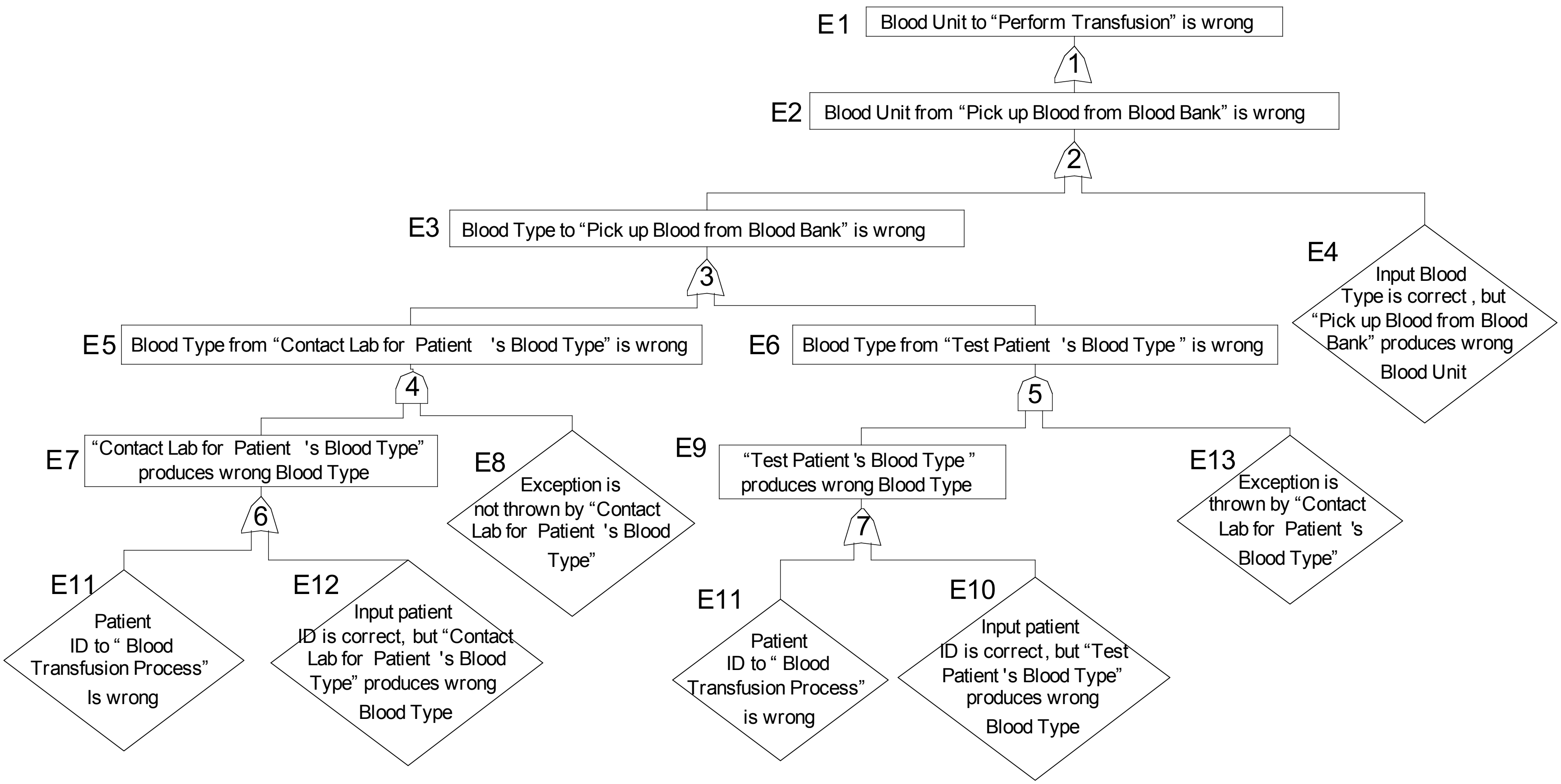
- Specify a hazard
  - Consider hazards created by the delivery of an incorrect artifact to a Little-JIL step
  - Generation based on templates for the semantics of the language

- Use Fault Tree Analysis to develop all Minimal Cut Sets
  - Automatically calculated from the fault tree using Boolean algebra

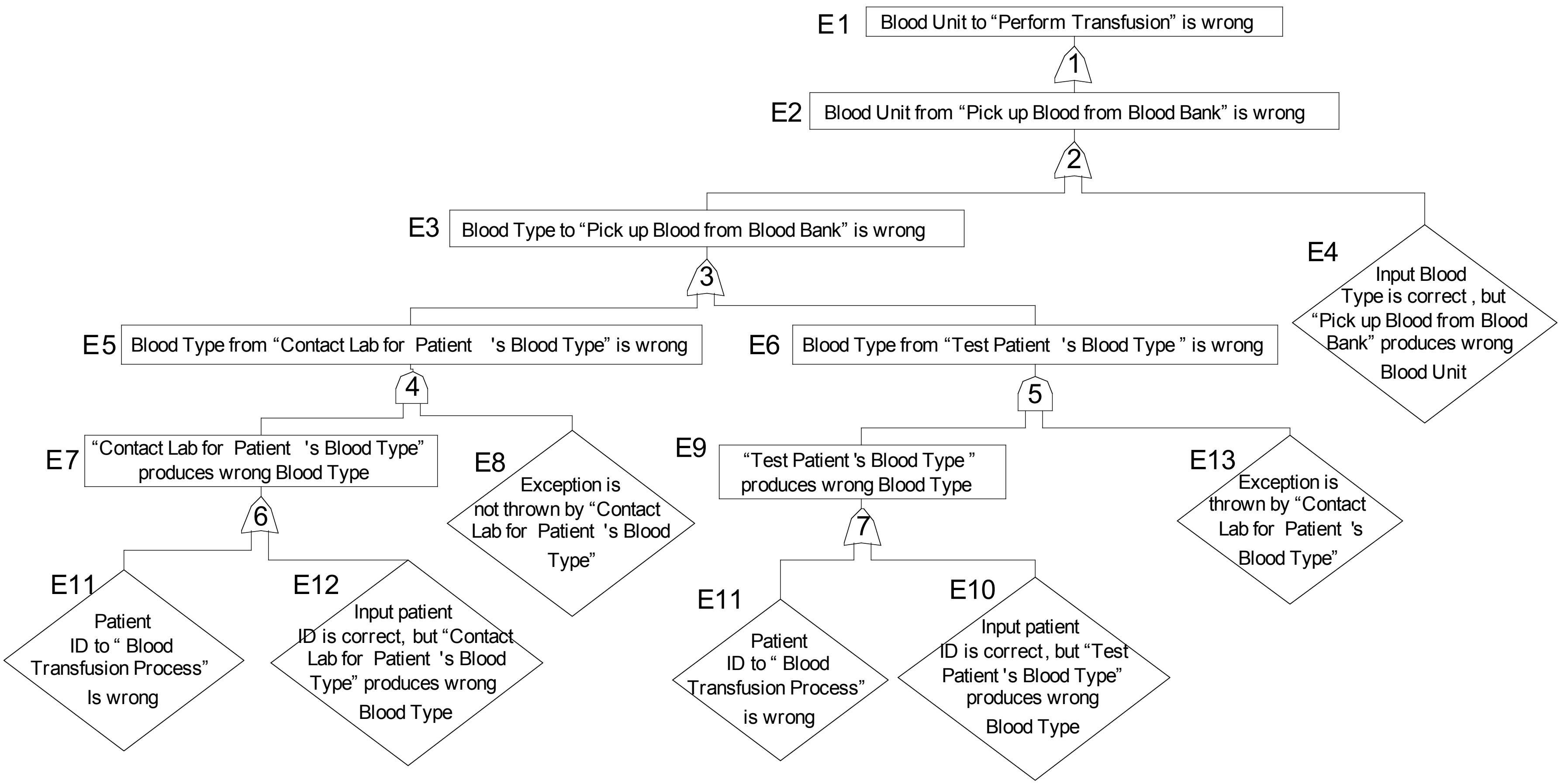
Simple Blood Transfusion Process



Example Fault Tree



Calculate MCSs

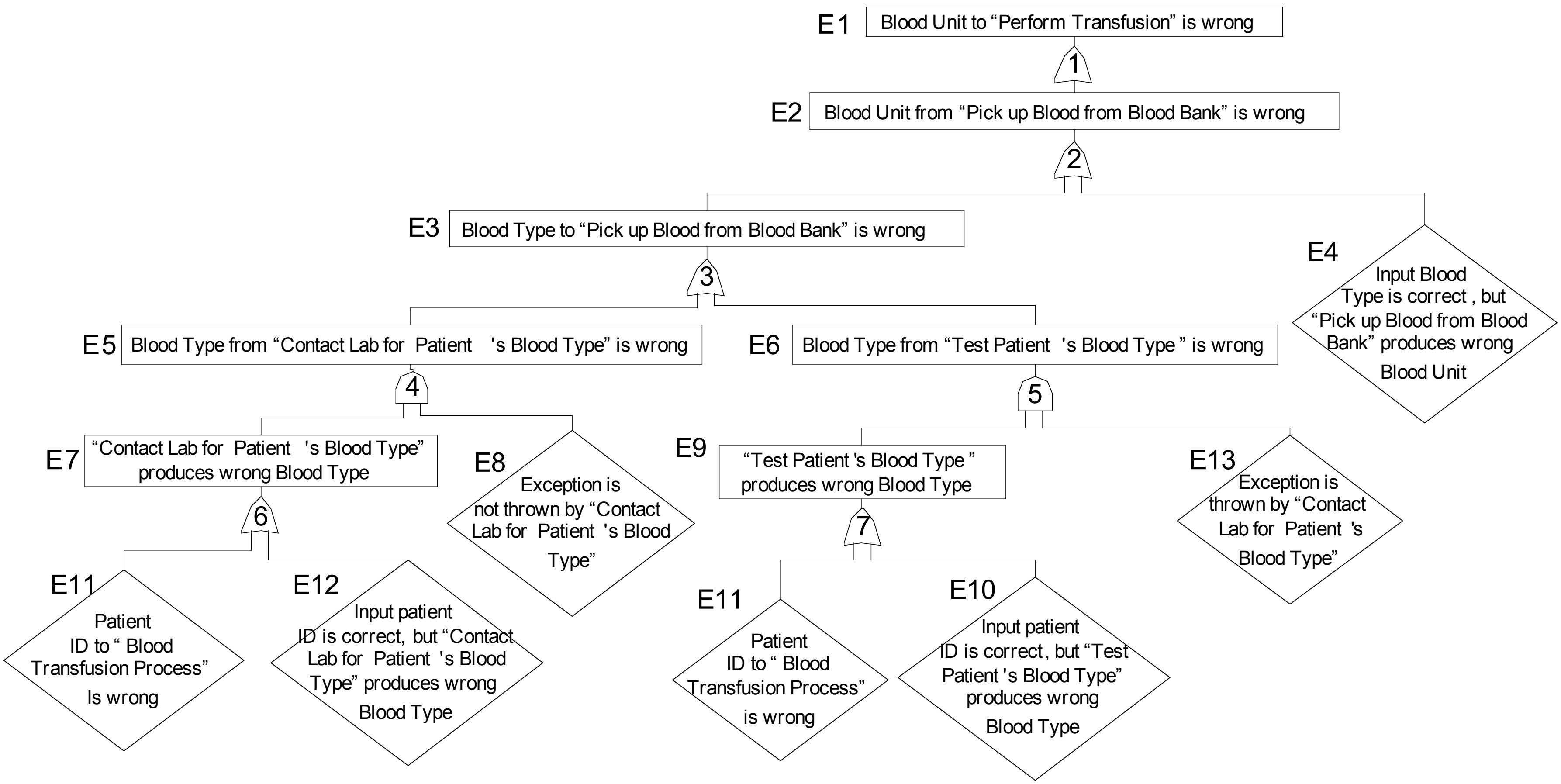


Each gate corresponds to an equation

1: E1 = E2 2: E2 = E3 + E4 3: E3 = E5 + E6 4: E5 = E7 • E8

5: E6 = E9 • E13 6: E7 = E11 + E12 7: E9 = E11 + E10

Calculate MCSs



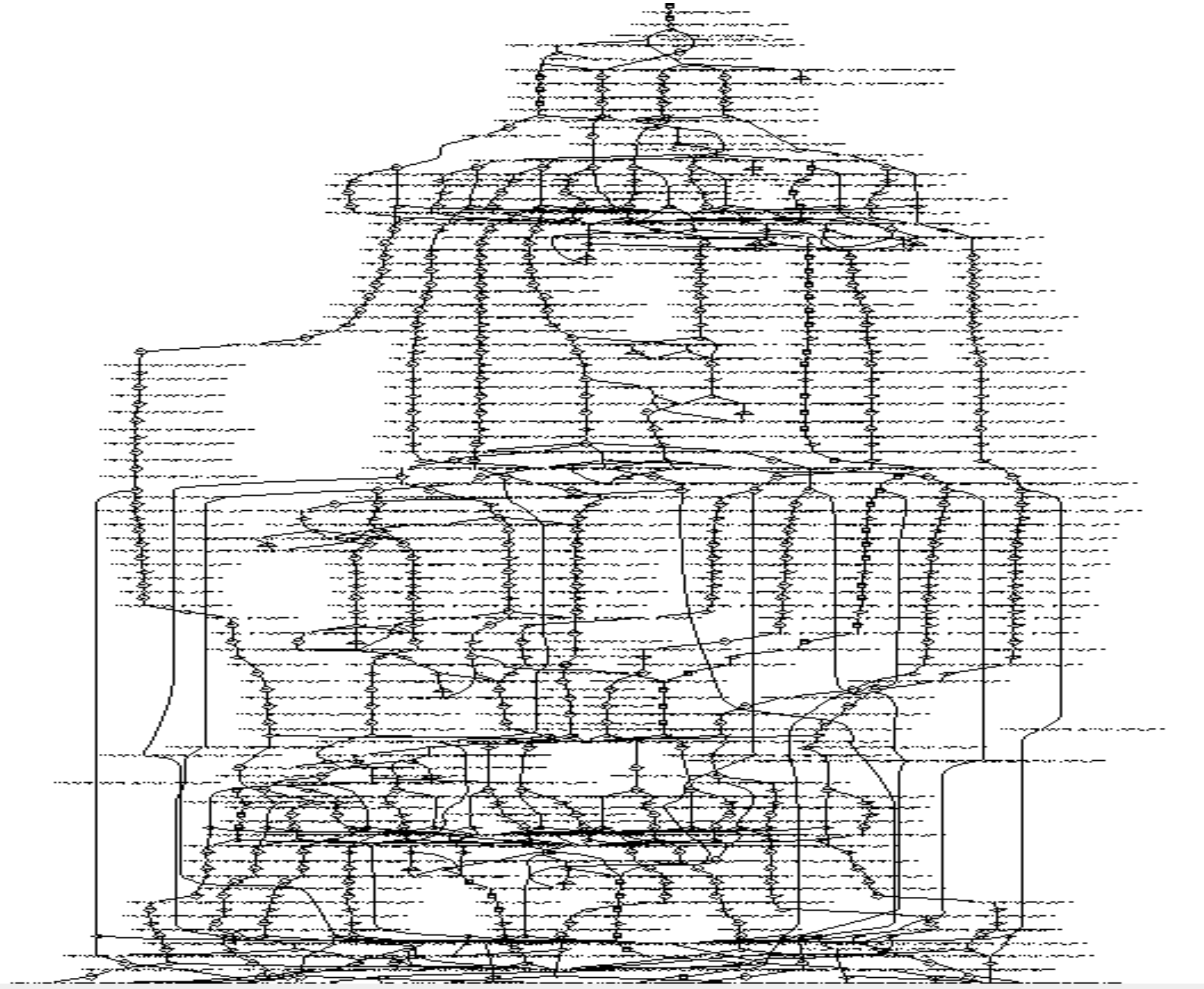
Derive an equation for E1 by eliminating and substituting the other intermediate events:

$$E1 = (E4) + (E11) + (E12 \bullet E8) + (E10 \bullet E13)$$

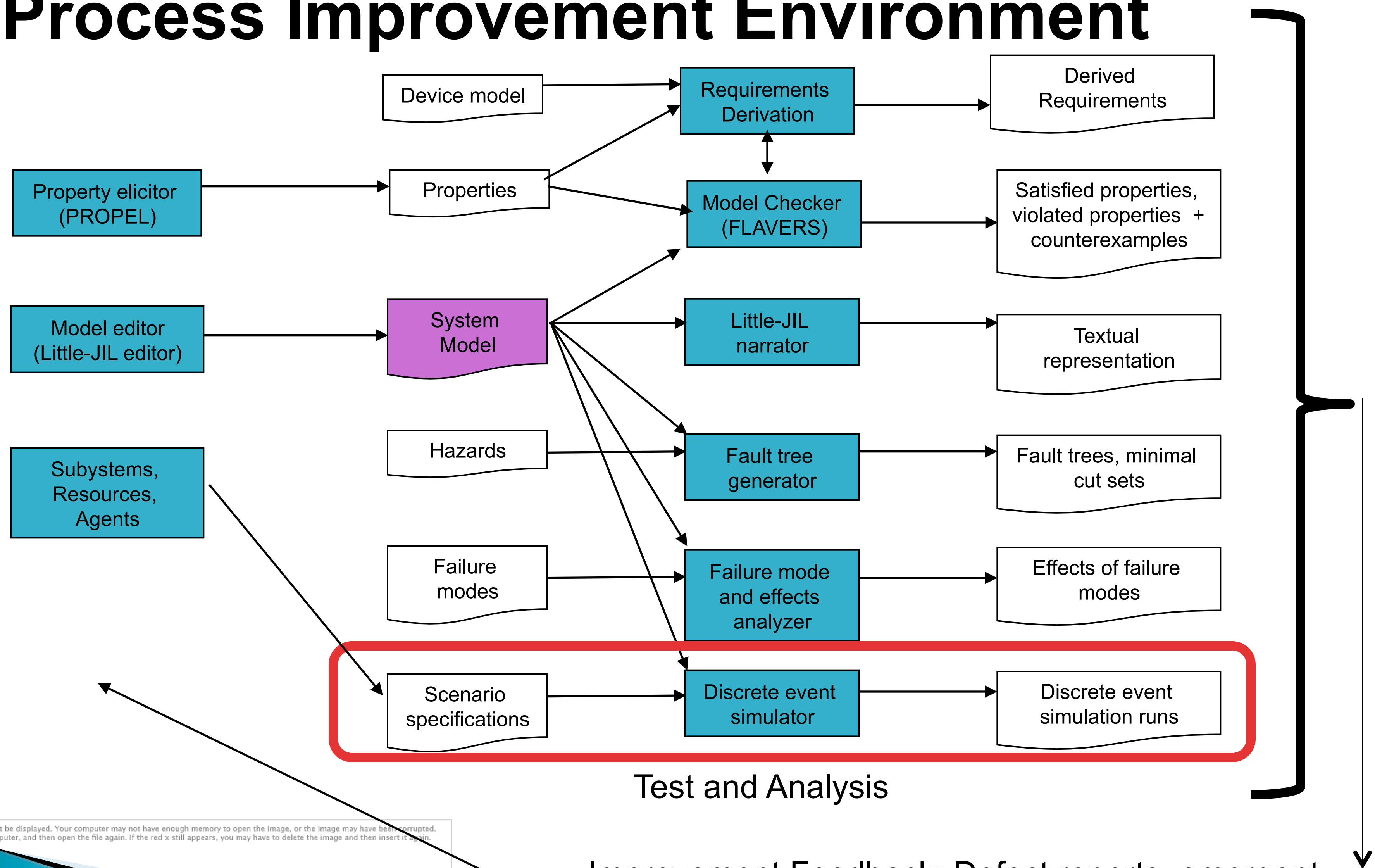
Observations about FTA and FMEA

- Usually fault trees and FMEA tables are created manually by safety engineers
  - Requires a deep understanding of the systems
  - Error prone and time consuming
- Using system models, we automatically derive fault trees and FMEA tables for multiple hazards/faults
  - Can easily be re-derived when the systems (and their models) are changed

Blood Transfusion Example: Generated Fault Tree



Iterative Improvement Driven by a Process Improvement Environment



Improvement Feedback: Defect reports, emergent understandings, resource and system behaviors, new and modified system and property details

What is a "resource"?

- A *resource* is an entity that is characterized by
  - Ability to provide one or more "capabilities"
    - Capability: The ability to support doing some task/activity/work
  - A set of descriptive attributes
    - Attribute: a (name, value) pair
- Capability set changes with context, circumstances
  - Attribute values do too
- A resource is a set of
  - Guarded capabilities
  - Guarded attributes

Example Resource Specifications from the Medical Domain

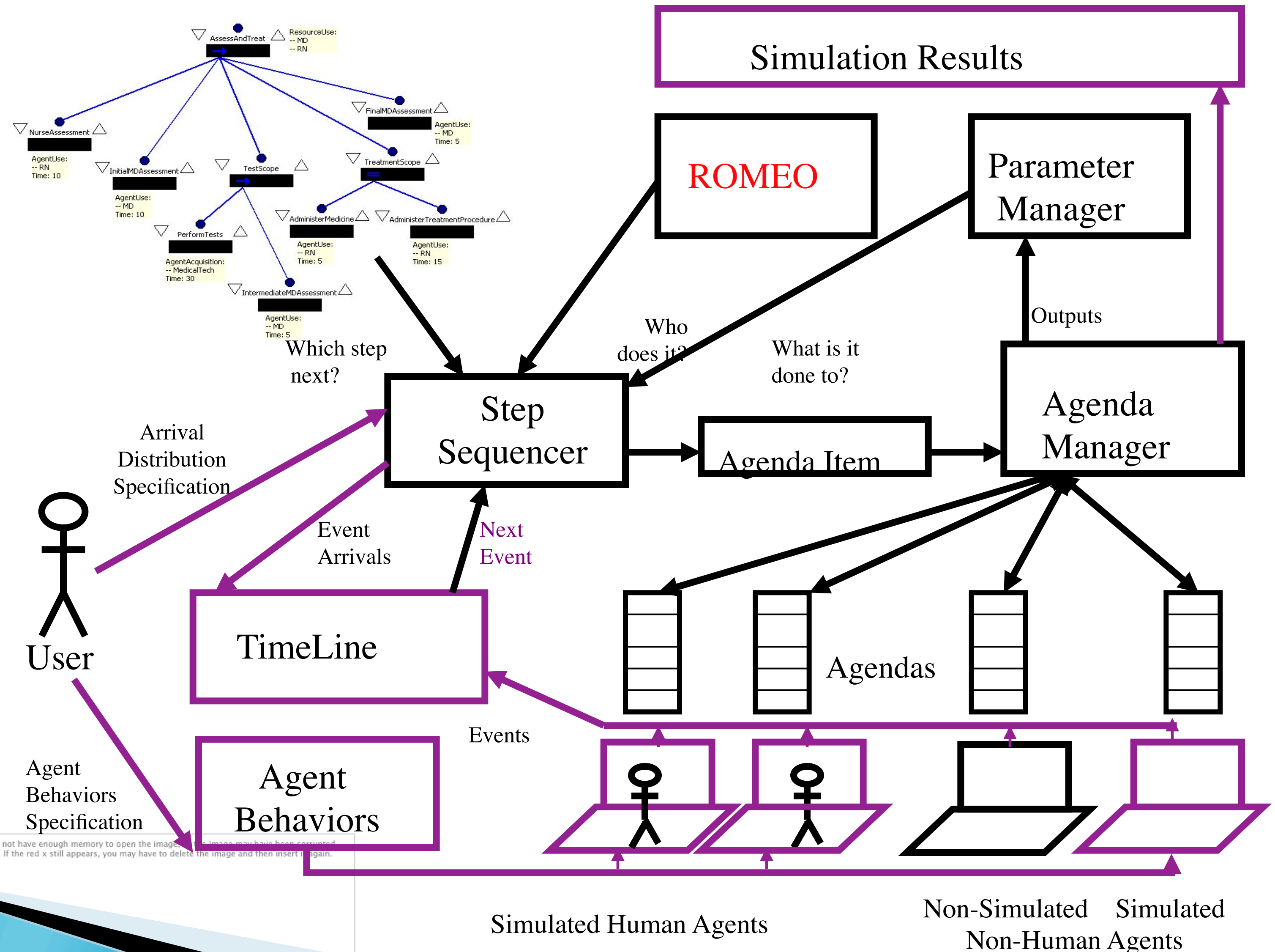
Name: John Smith
Job Title: Physician
Location: ED
Experience Level: 10
Cost: 100
Capacity: 8
Offered Capabilities:

(MDInitialAssessment, .true., 10, 2)
(MDProcedure, .true., 10, 5)
(MDFinalAssessmentandDecision, .true., 10, 3)
(RNPaperwork, [availability.nurse = 0 $\wedge$ crowding > 100], 3, 1)

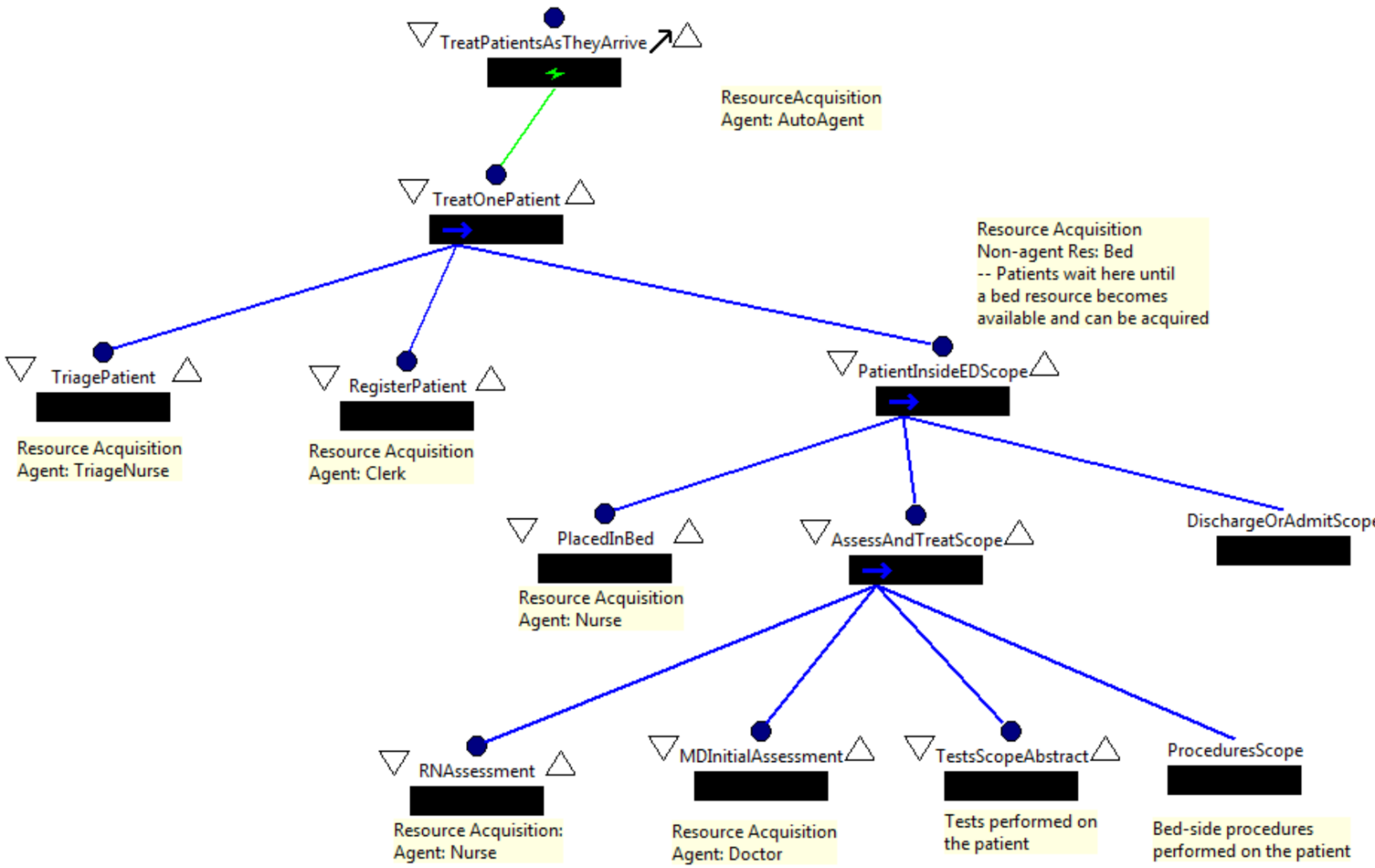
Name: Ellen Masterson
Job Title: Physician
Location: ED
Experience Level: 4
Cost: 80
Capacity: 8
Offered Capabilities:

(MDInitialAssessment, .true., 10, 2)
(MDProcedure, .true., 9, 5)
(MDFinalAssessmentandDecision, .true., 10, 3)
(RNPaperwork, [availability.nurse = 0 $\wedge$ crowding > 100], 3, 1)

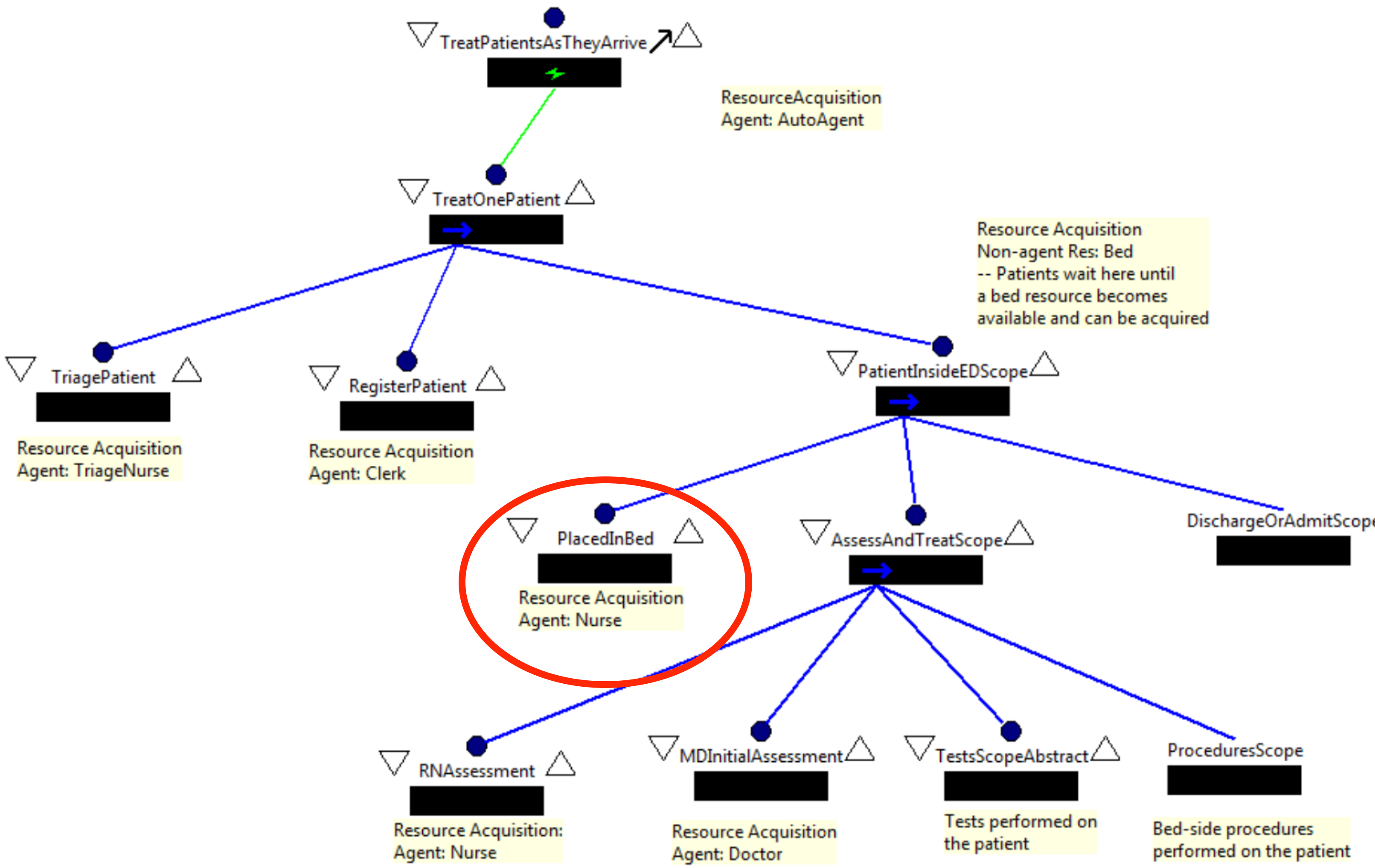
JSim: The Little JIL Simulator



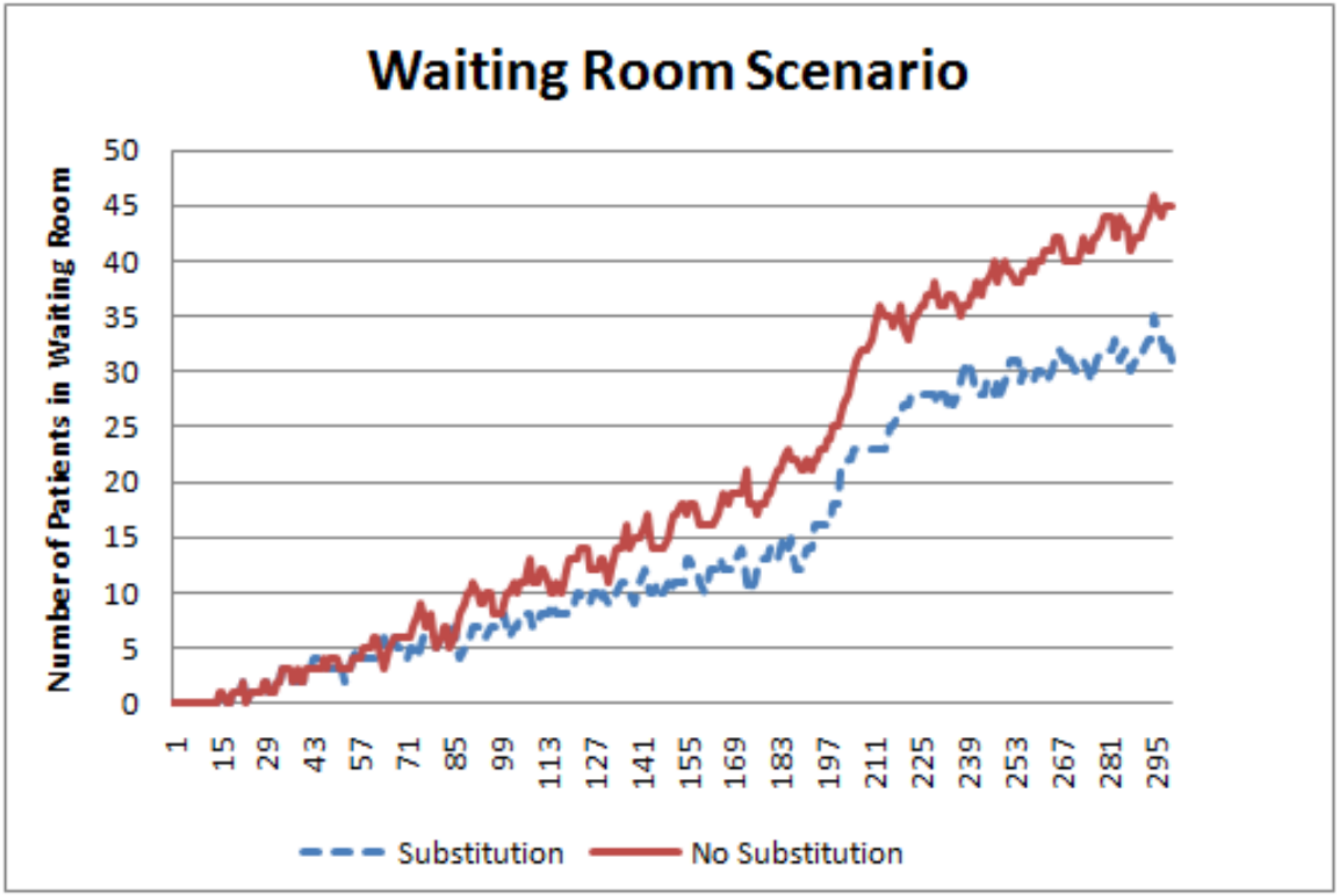
The “SimpleED” Process



The “SimpleED” Process With a Policy Change



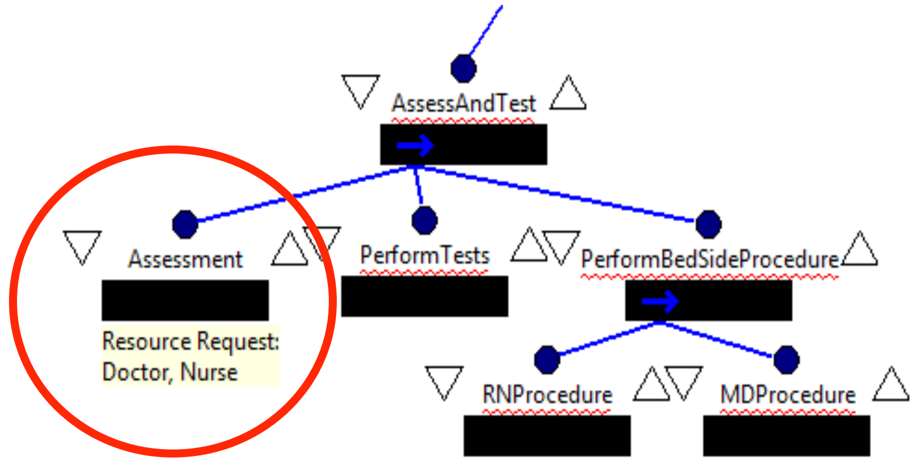
Triage Nurse can place patient in bed

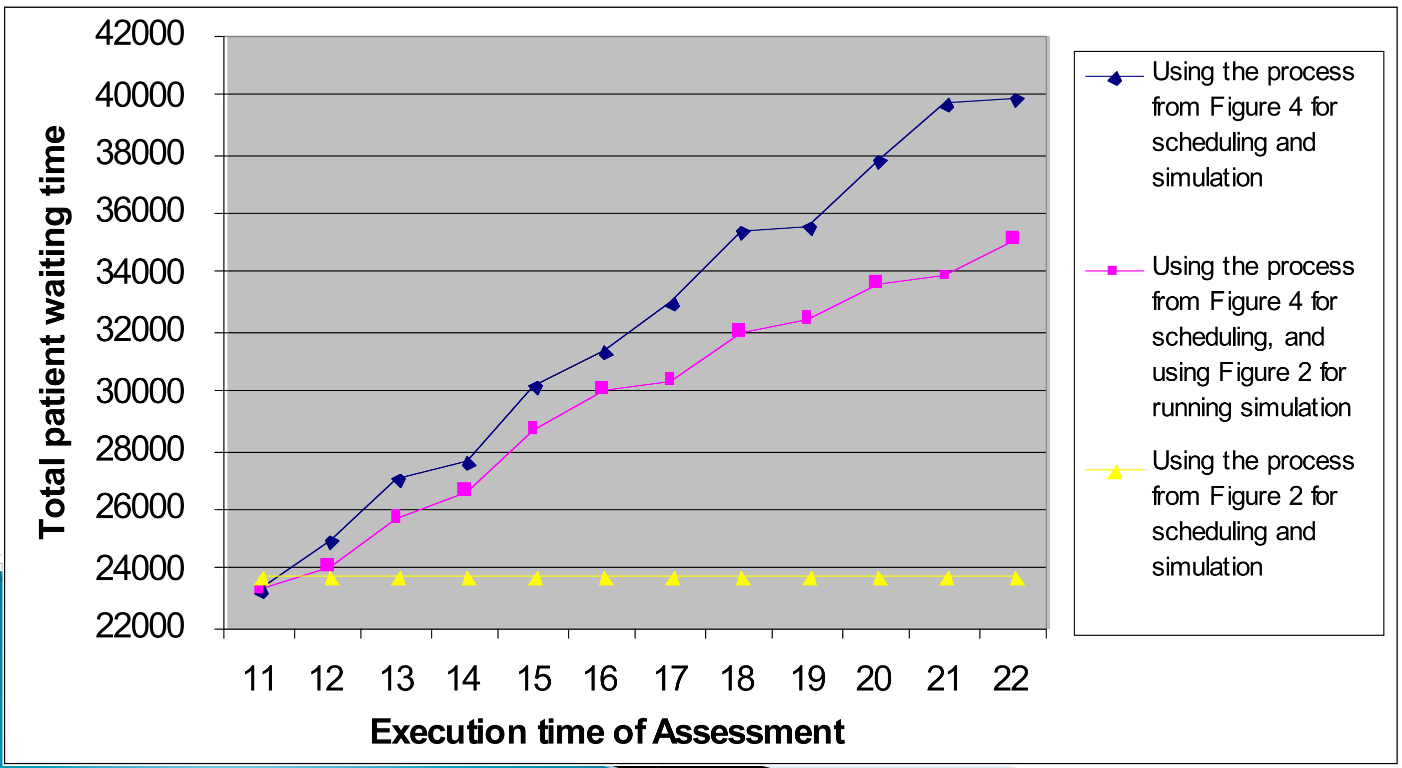


Elapsed time (in simulation time units)

Effect of process detail on scheduling effectiveness

Elaborate Assessment step with nurse and doctor doing assessments in parallel with each taking 11 time units





Observations about Simulations

- A number of simulation systems are commercially available
  - Usually based on queuing models
- System-based models more easily provide finer-grain control
- Initial studies seem to indicate that finer-grain control can increase simulation accuracy

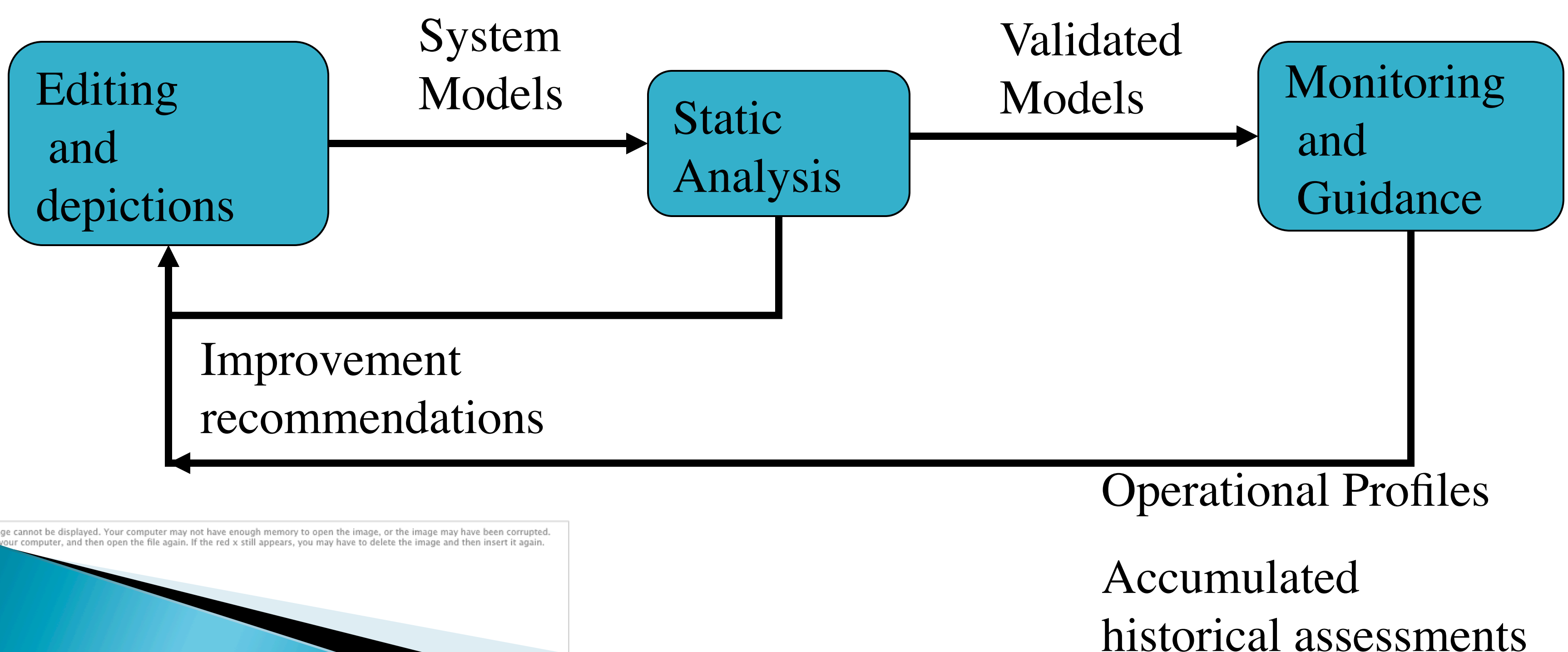
- Can leverage the investment in the model
  - Provides a basis for studying resource allocation using scheduling and planning

One Early Clinical Result

- Defined part of breast cancer chemotherapy process
  - Up to and including the first day of chemo
- Number of errors reaching the patient declined by ~70%
  - Due to errors found and/or heightened process awareness
- To appear in *Joint Commission Journal of Quality and Patient Safety*

Future Vision

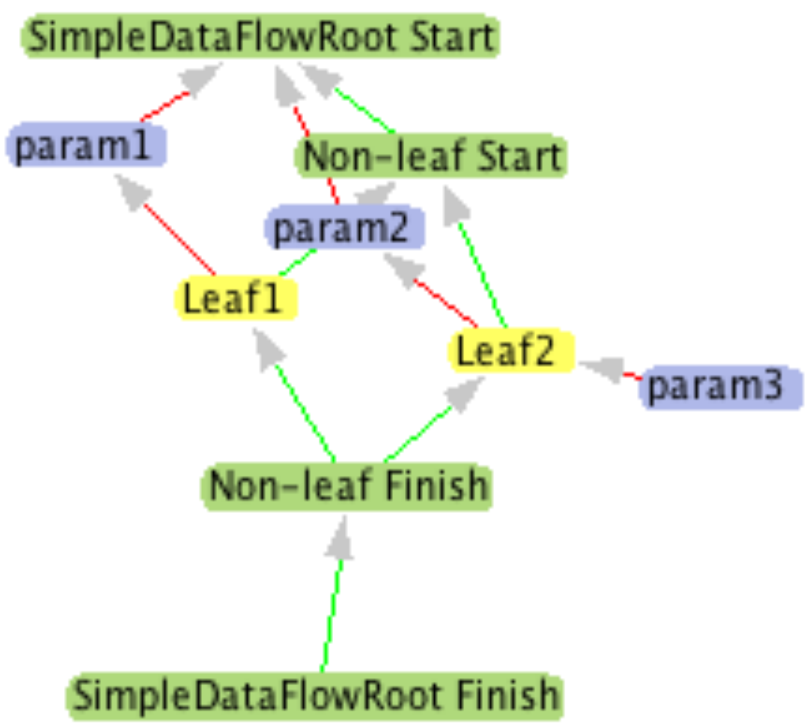
- Environment for evidence-based, systematic system improvement



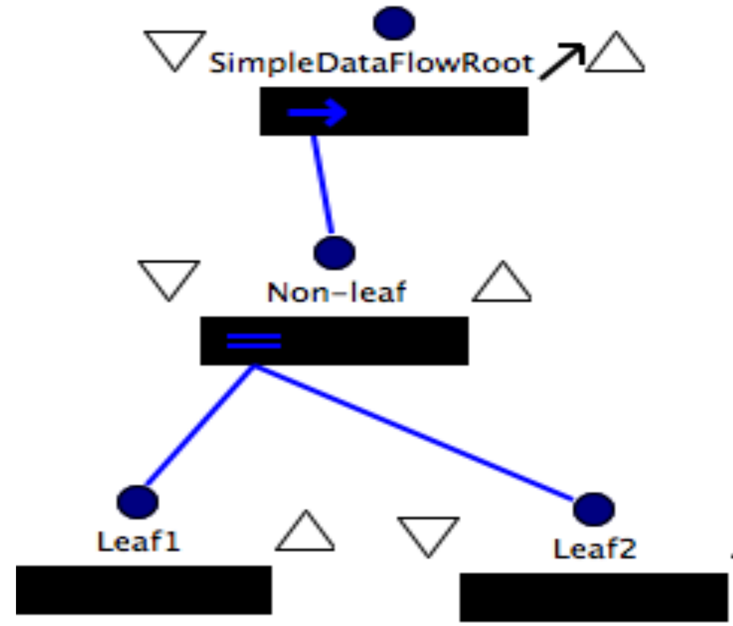
The DDG Project (Work being led by Barbara Lerner and Xiang Zhao)

- Defined templates for translating Little-JIL step executions into DAG fragments
- Gluing software for building DDGs from them
- Incorporates scoping, nesting, hierarchy information
- Links to previous values of artifacts
- Detailed history is inferrable
- Can generate DDGs dynamically while process is executing

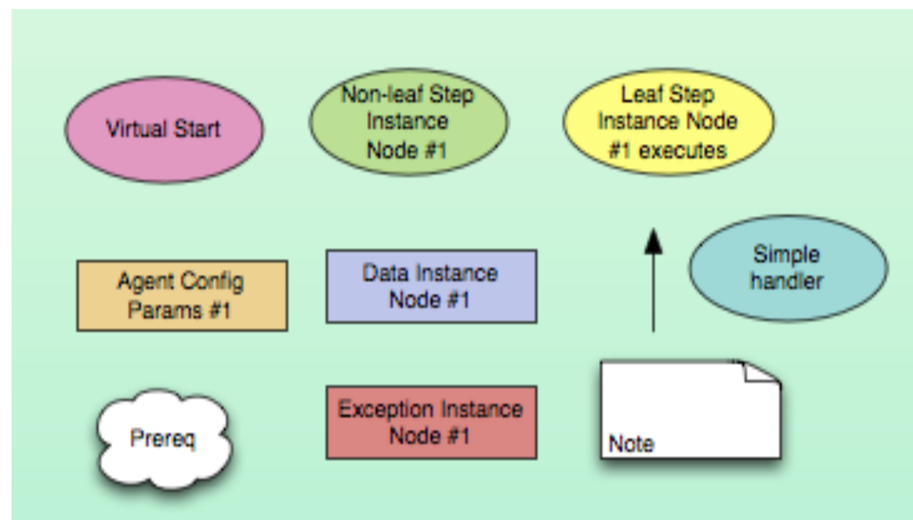
Example



DDG

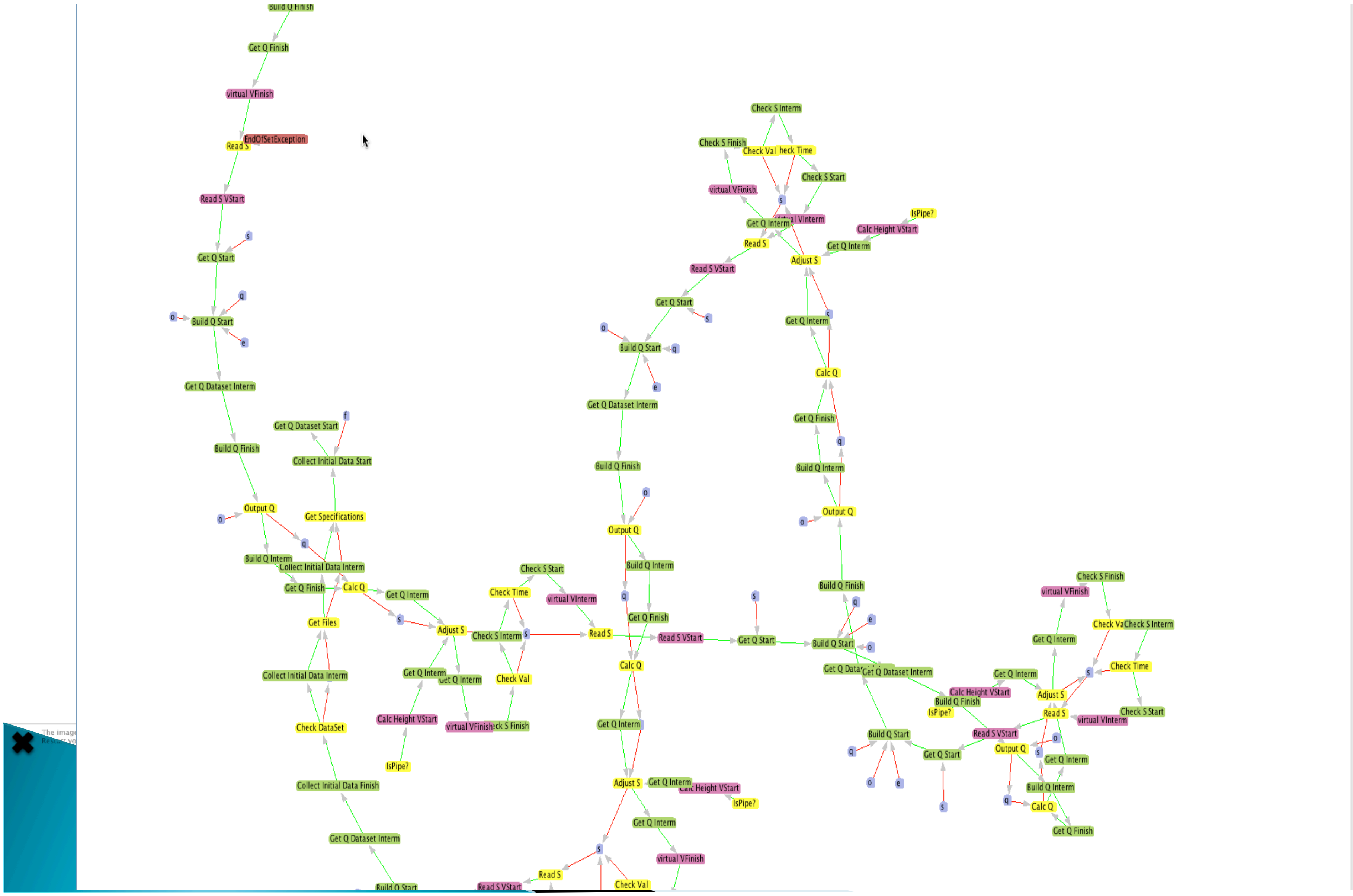


Little-JIL

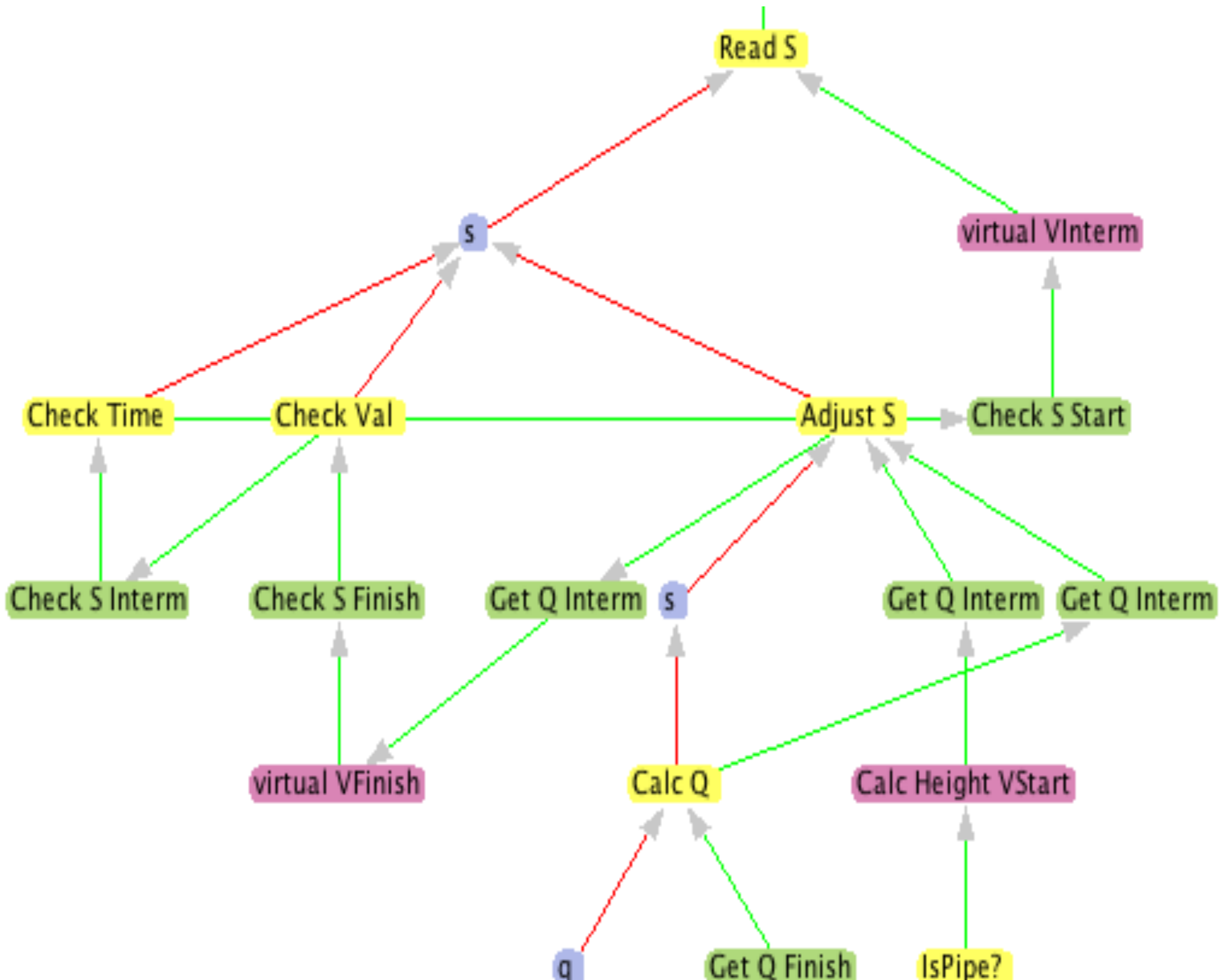


Legend

Actual Generated DDG (From Ecology Process Definition)



Enlargement of a Piece



Overall Observations

- Found important errors and inefficiencies in the systems that we modeled
  - Sequence errors
  - Deadlocks
  - Single points of failure
  - Resource allocation bottlenecks
- Mostly found errors in the system models and properties
  - But, correcting these is important if the other analysis results are to be trusted
  - Unable to do experimental before and after studies
- Testimonials
  - Medical colleagues, ecology researchers, claim that this approach has changed the way they view their systems and processes, the terms they use, and how they teach their disciplines

Conclusions

- Current approach
  - Analysis-driven approach to iterative improvement of human intensive systems of systems
  - Effective for identifying errors, vulnerabilities, emergent properties and behaviors in these systems
  - Indicating improvement directions
- Future work: monitoring and guidance based on validated system models
  - Basis for deviation detection
  - Framework for accumulating operational data, applying probabilistic analysis, and proposing **evidence-based** improvements

Questions?

Backup Slides

Four different continuations on exception handlers

- Complete
  - Handler was a "fixup"; substep is completed
- Continue
  - Handler cleaned up; parent step is completed
- Restart
  - Handler cleaned up; repeat substep (deprecated)
- Rethrow
  - Rethrow to parent step

Channels and Lateral flow

- Channel supports message passing
- Multiple steps can add artifacts
- And multiple steps that can take them
- Use for synchronization and passing artifacts

Pre- and Post-requisites

- Steps guarded by (optional) pre- and post-requisites
- Are steps themselves
- Can throw exceptions
- May be executed by different agents
  - From each other
  - From the main step

Another Resource Specification

Name: Bed 12
Job Title: Bed
Location: ED
Experience Level: 12
Cost: 250
Capacity: 1
Offered Capabilities:
(PatientInsideED, .true. , 10, 1)

PROPEL Templates

- Provides templates that explicitly indicate the options associated with each Property Pattern (Dwyer, Avrunin, and Corbett)
- Three coordinated representations
  - Question Tree
    - Helps select the appropriate pattern
    - Guides in the selection of options
  - Disciplined Natural Language (DNL)
    - Specifier selects from given optional phrases
    - Fully instantiated template is a sequence of English sentences
  - Extended Finite-State Automaton
    - Graphical FSA with optional transitions, labels, and accepting states
    - Fully instantiated template is a FSA defining a language of desirable sequences of events; basis for Model Checking

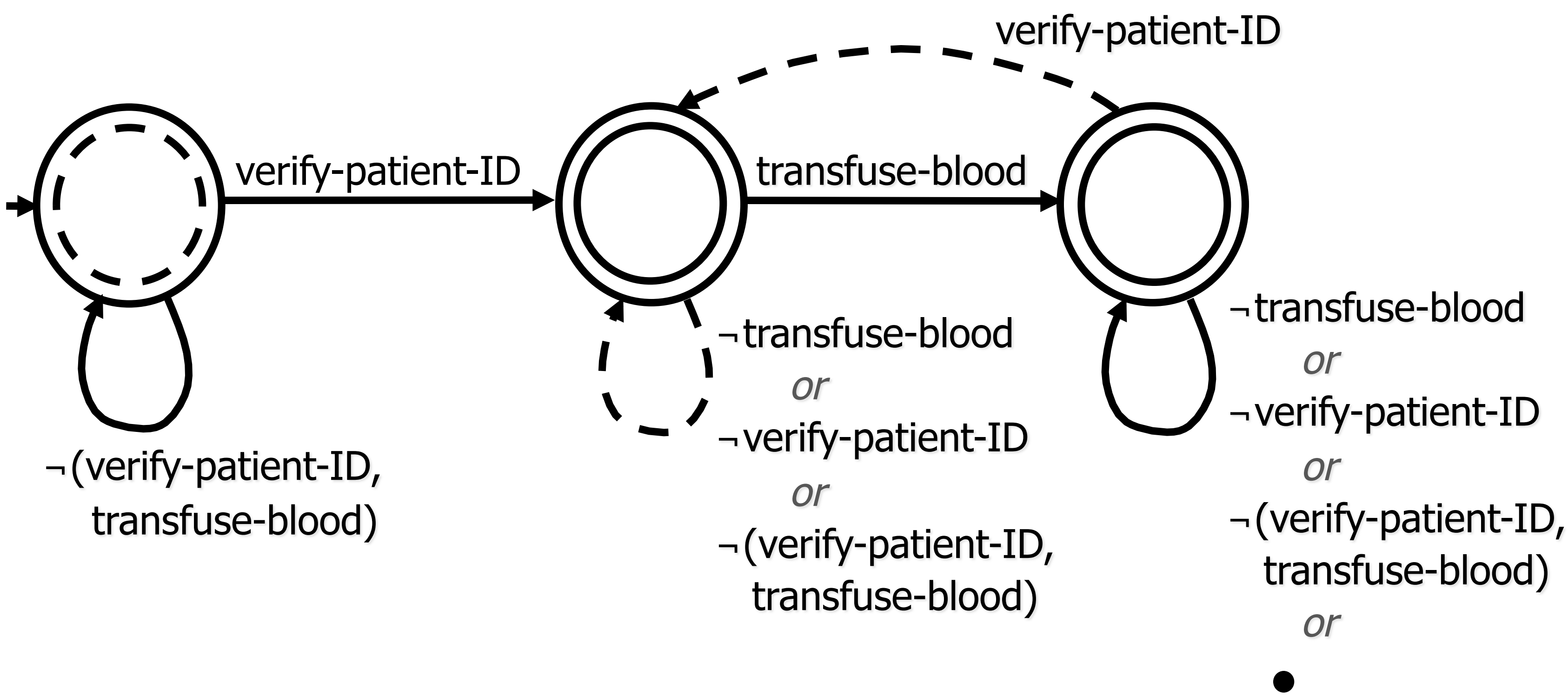
Question Tree View

How many events of primary interest are there?

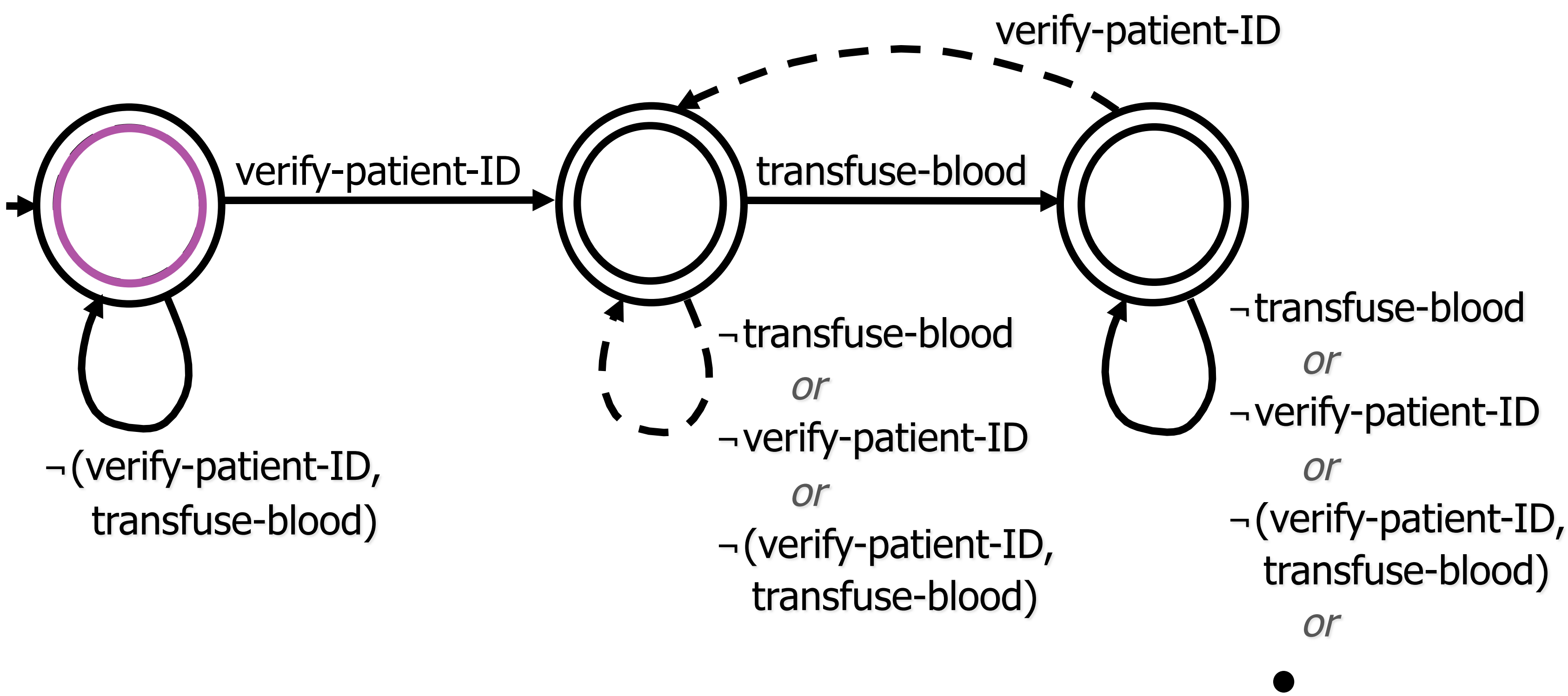
- One: event verify-patient-ID
- Two: events verify-patient-ID and transfuse-blood

  - After verify-patient-ID occurs, transfuse-blood is required to occur
  - transfuse-blood cannot occur until after verify-patient-ID has occurred

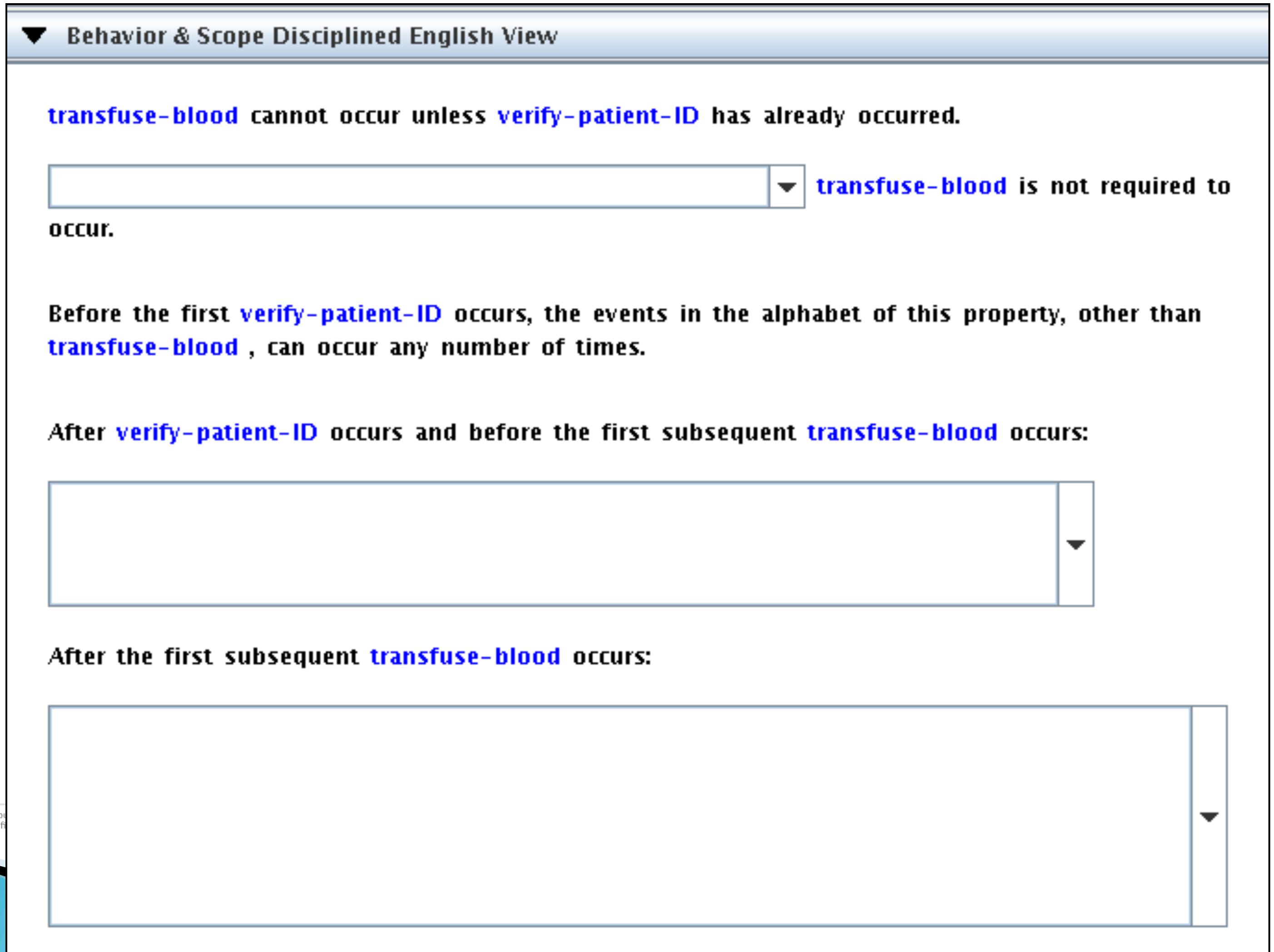
Precedence FSA Template



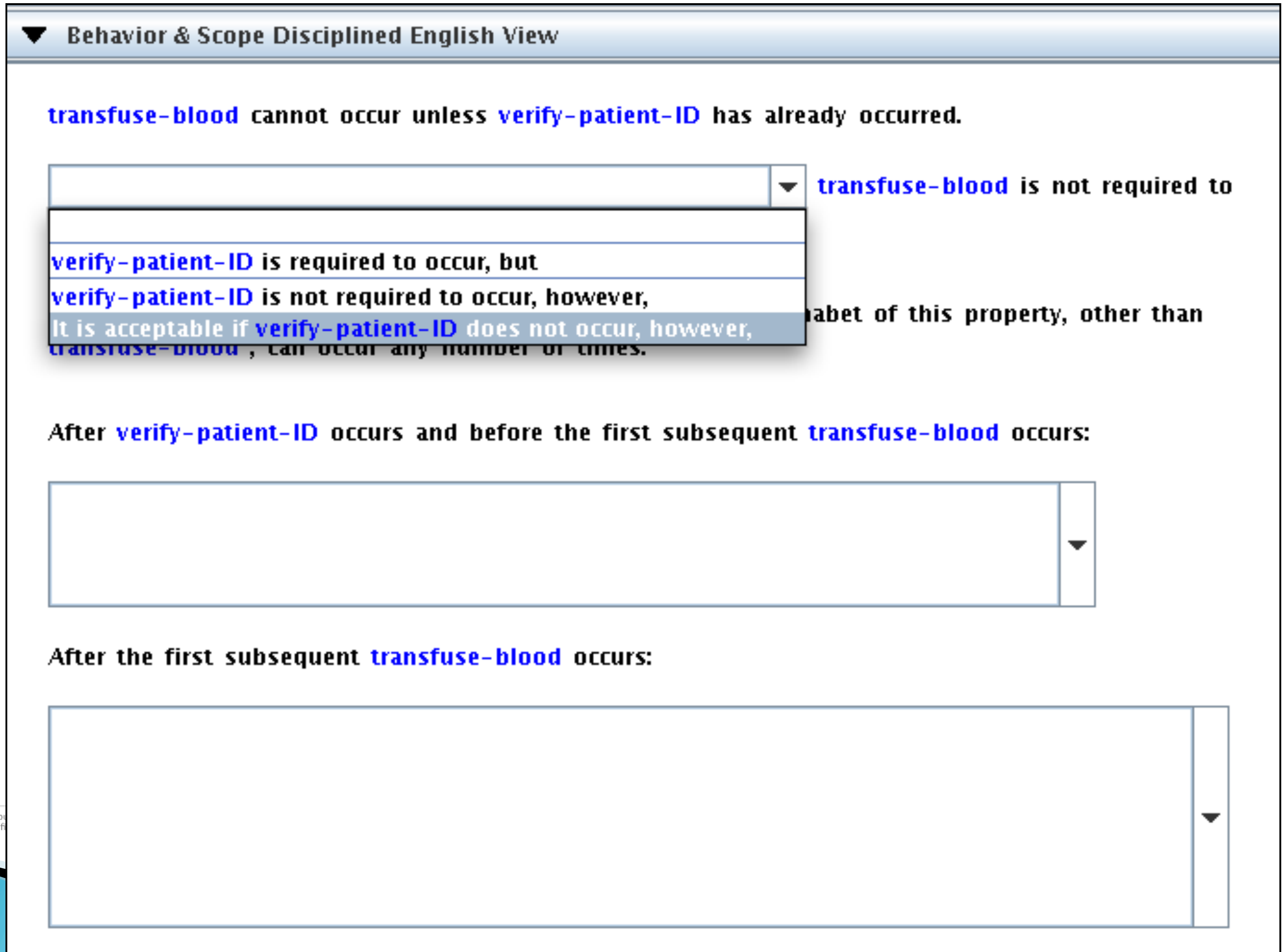
Precedence FSA Template



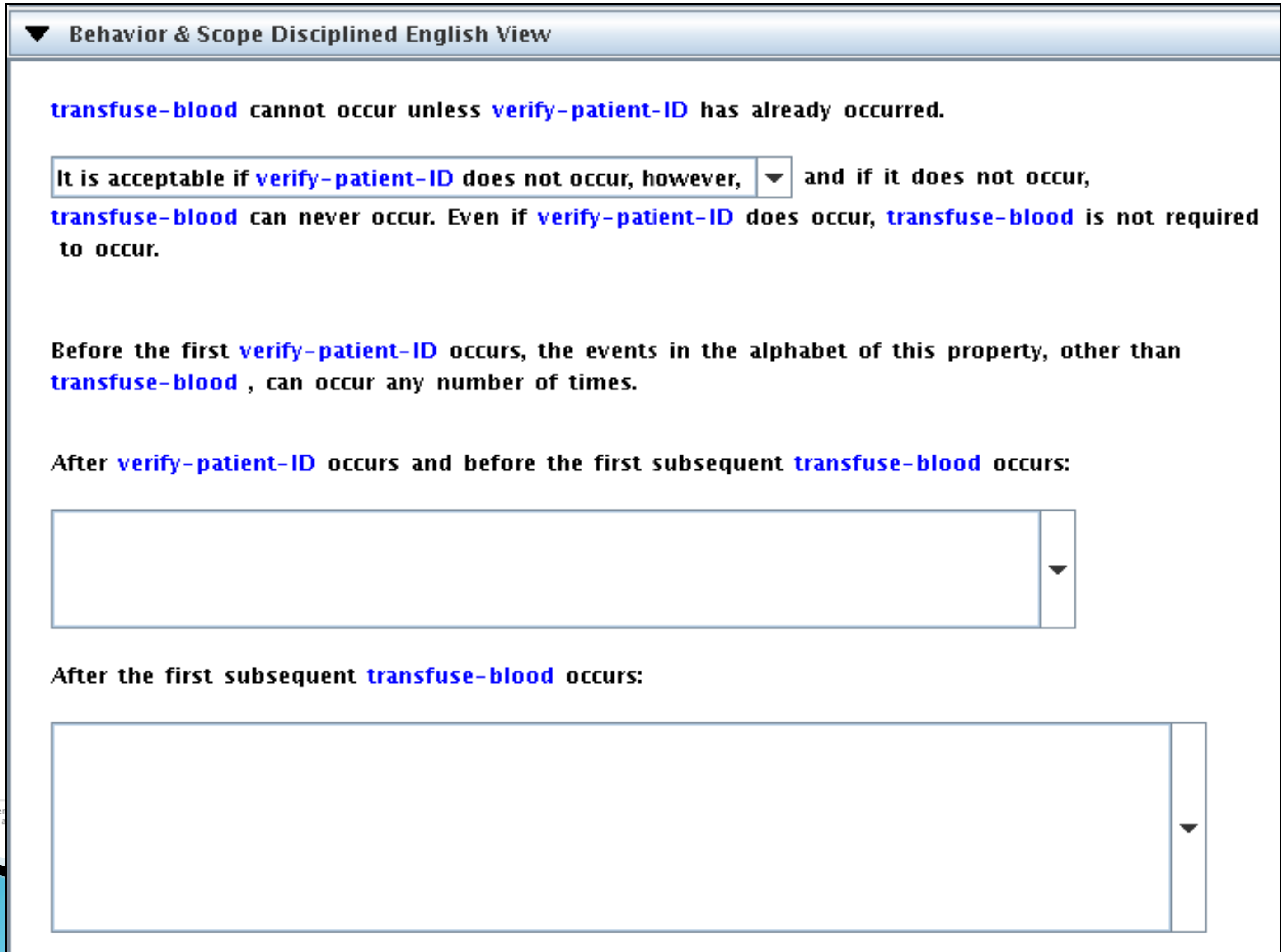
Precedence DNL Template



Precedence DNL Template



Precedence DNL Template



Observations about Specifying Properties

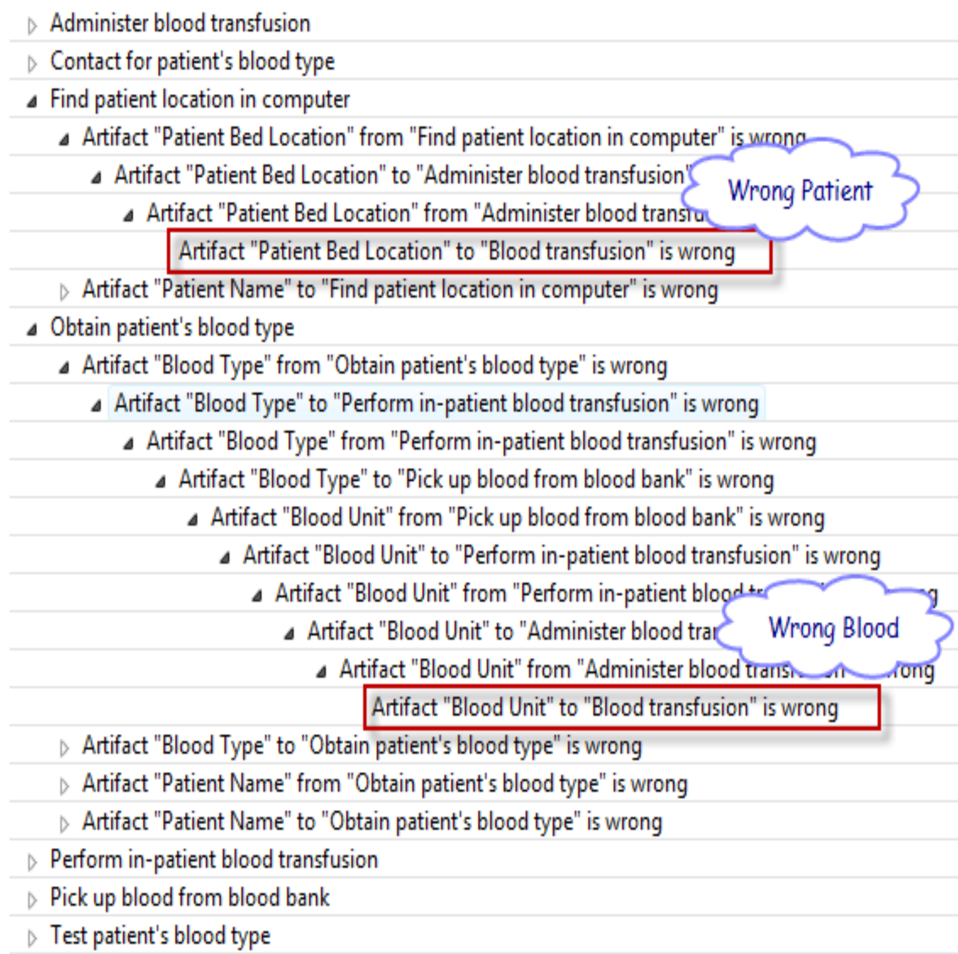
- Specifying the properties helped determine the scope/granularity of the system model

- Added the ability to specify properties in the context of exceptions
  - PropA is true unless exception X1 or X2 occurs

Identify Effect (s) for Each Failure Mode

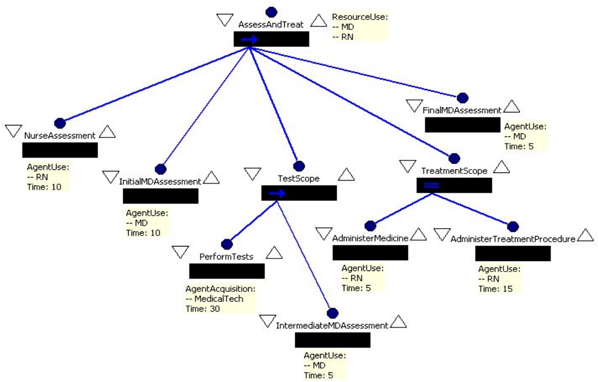
- Shows two potential hazards:
- “Patient Bed Location” is wrong =>
  wrong patient receives blood
- “Blood Type” is wrong =>
  patient receives wrong blood

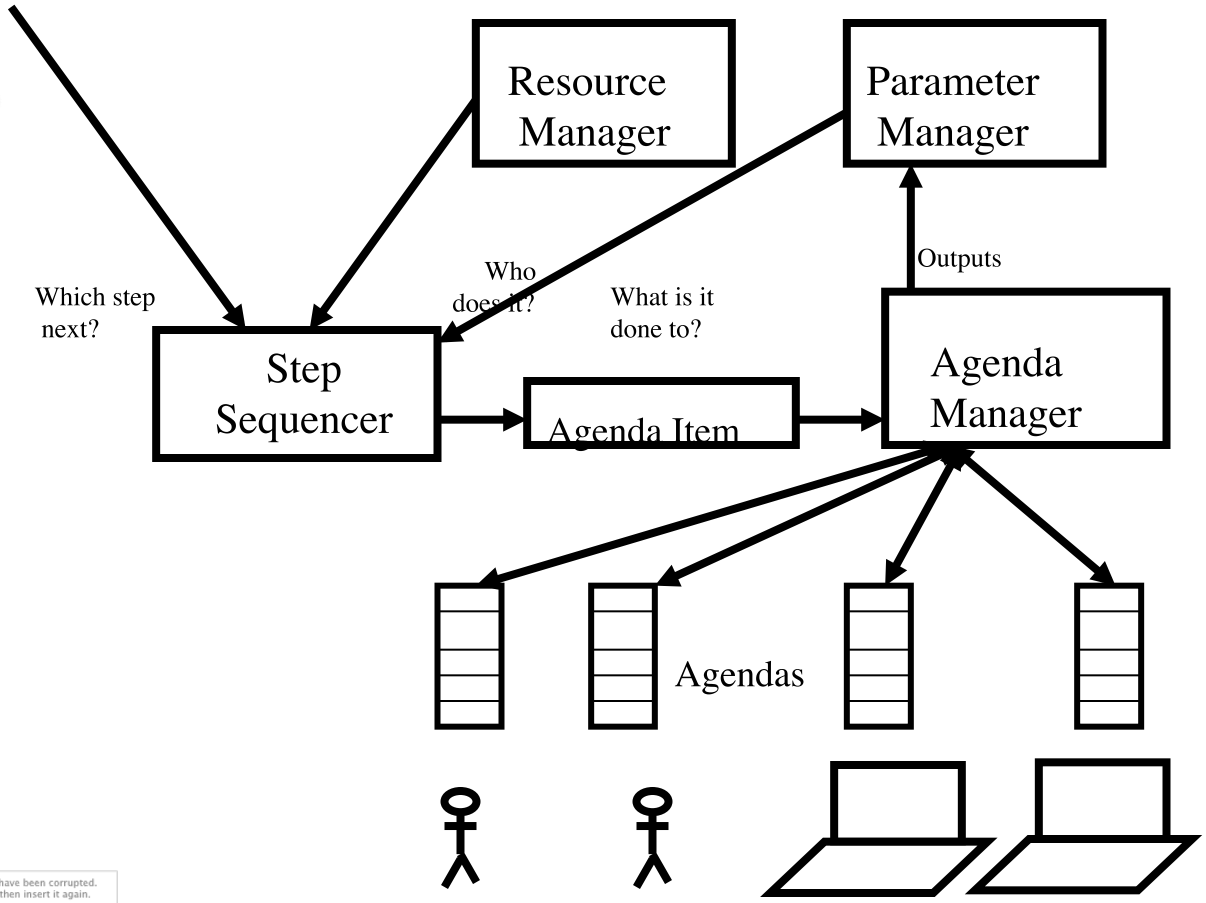
FMEA Table



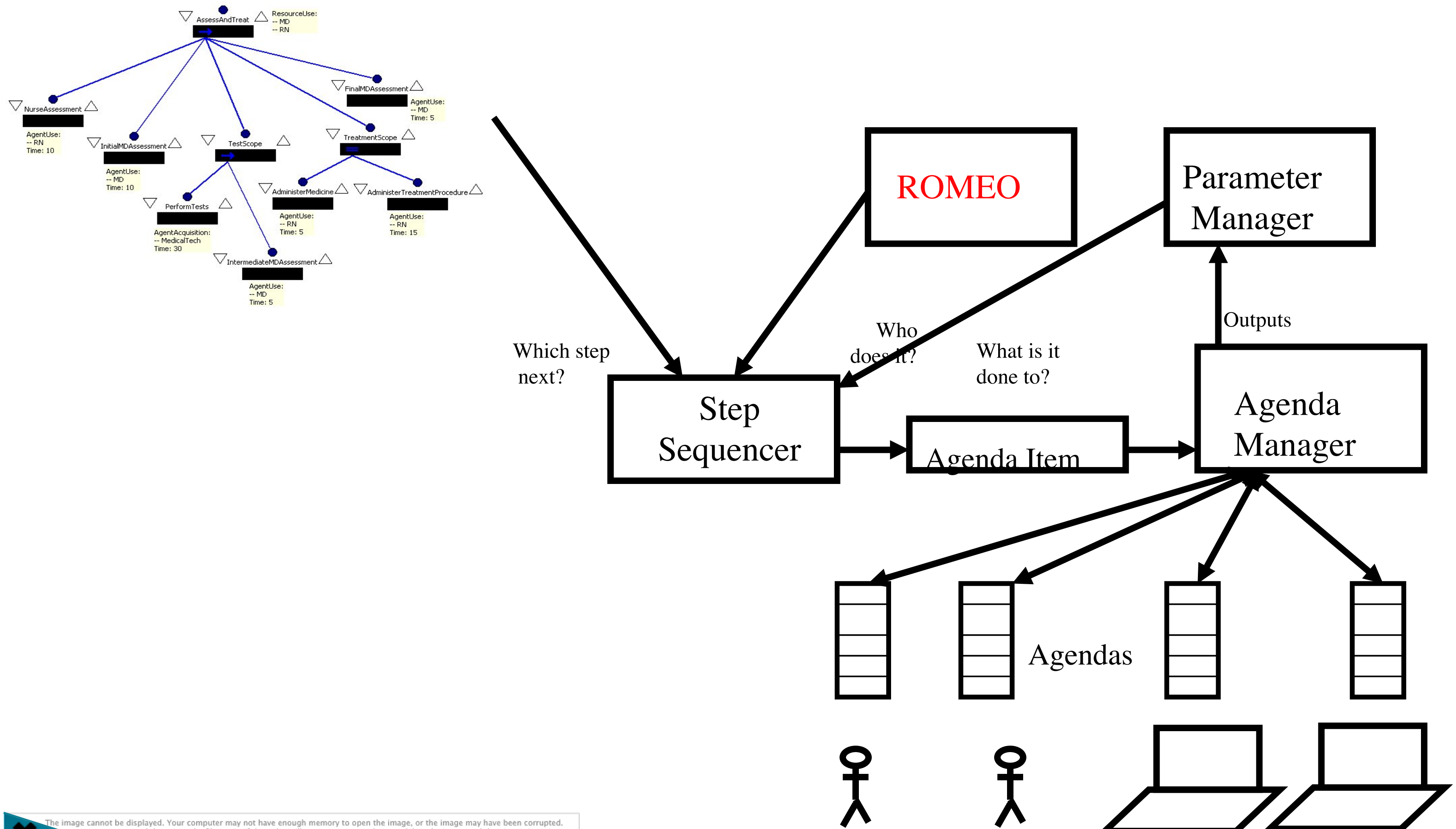
The image cannot be displayed. Your computer may not have enough memory to open the image, or the image may have been corrupted. Restart your computer, and then open the file again. If the red x still appears, you may have to delete the image and then insert it again.

Executing Little-JIL Process Definitions





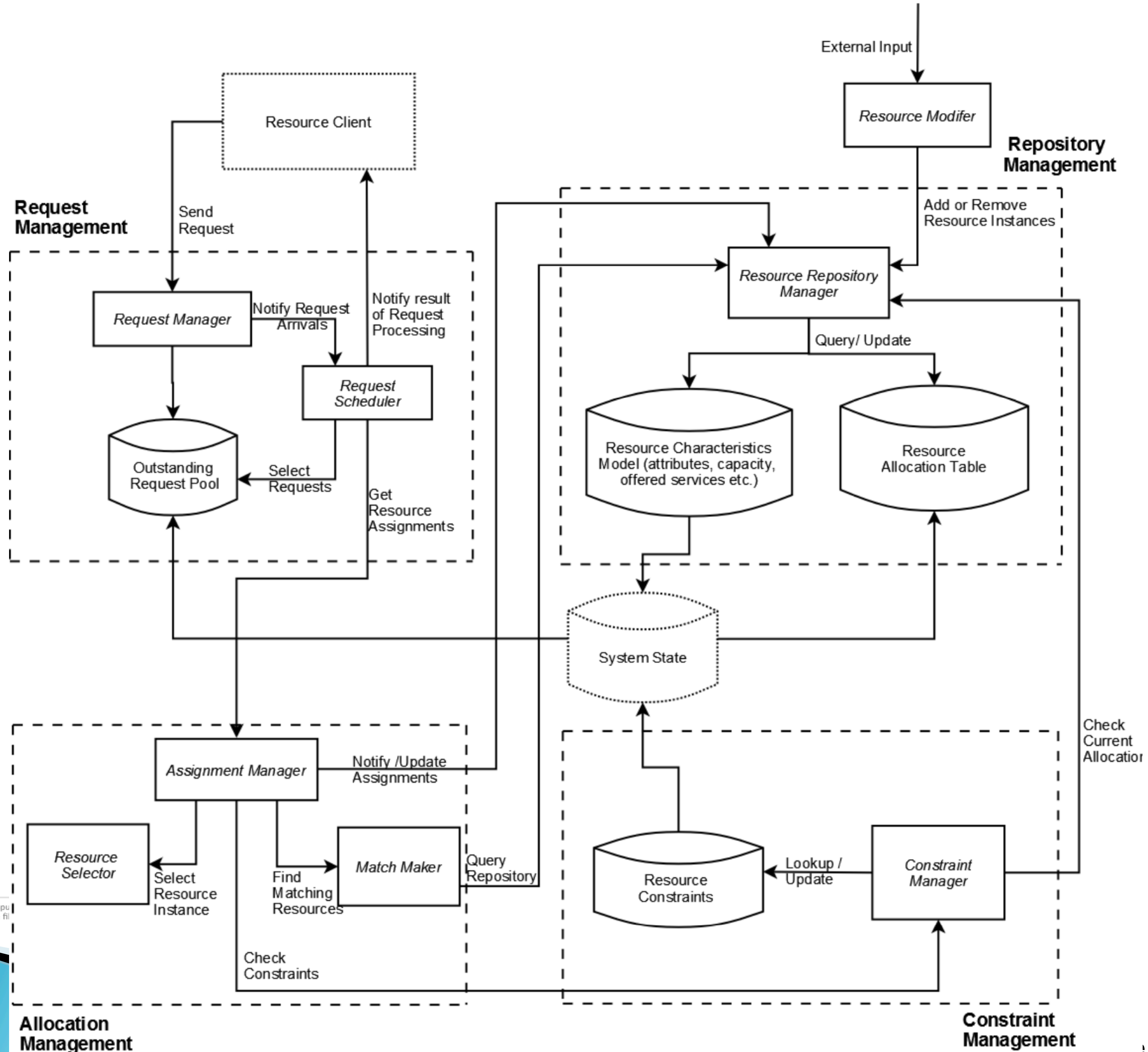
Executing Little-JIL Process Definitions



ROMEO approach to defining and managing resources

- Store resource entities as database relations
- Turn resource requests into queries
- Group sets of attributes into predefined queries
  - Serves some of the purposes of a type structure
- Add and delete resource instances dynamically as data base modifications
- Change attribute values and guards dynamically as DB modifications too

The ROMEO architecture

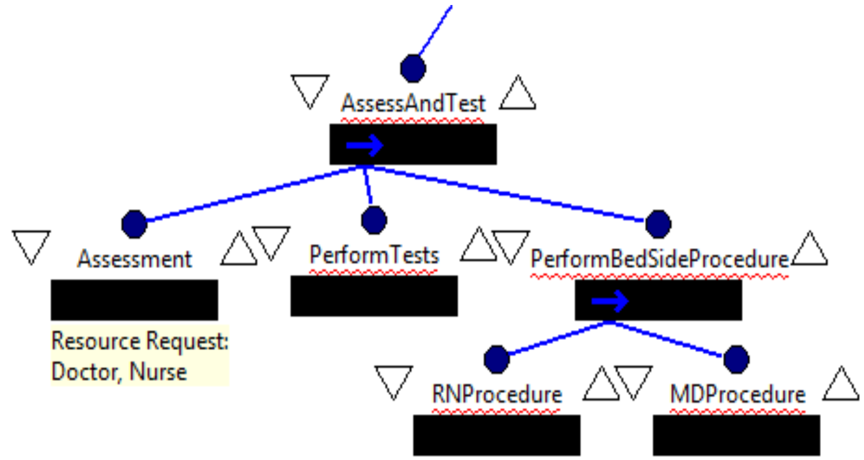


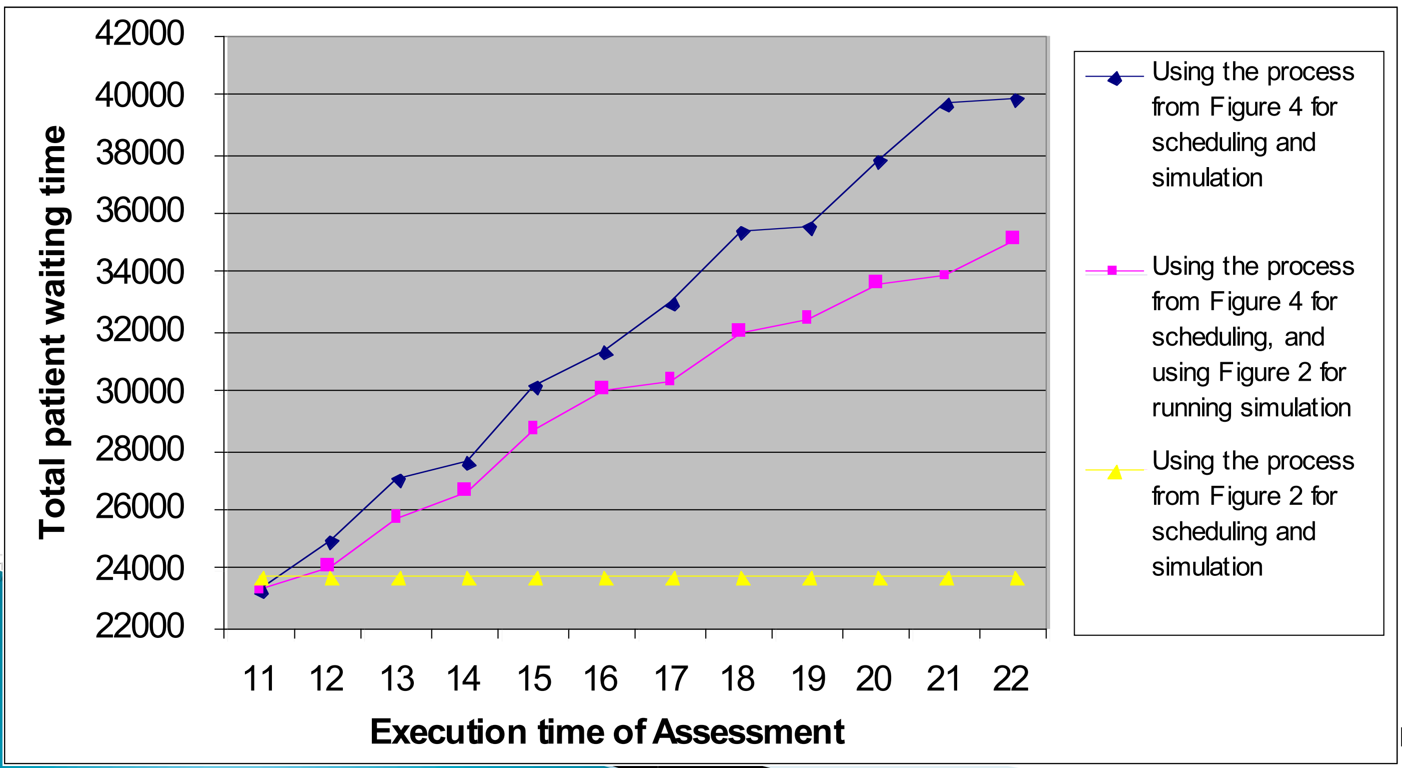
Incremental Resource Scheduling

- First-come-first-served is myopic
- End-to-end static scheduling breaks down in a dynamic environment
  - Unexpected events can negate entire schedule
  - Unanticipatable paths through the process can too
- Intermediate approach: Incremental scheduling
  - Define a window of upcoming events
  - Schedule over that window
  - Reschedule when scheduled tasks have been completed or when disruption negates schedule

Effect of process detail on scheduling effectiveness

Elaborate Assessment step with nurse and doctor doing assessments in parallel with each taking 11 time units



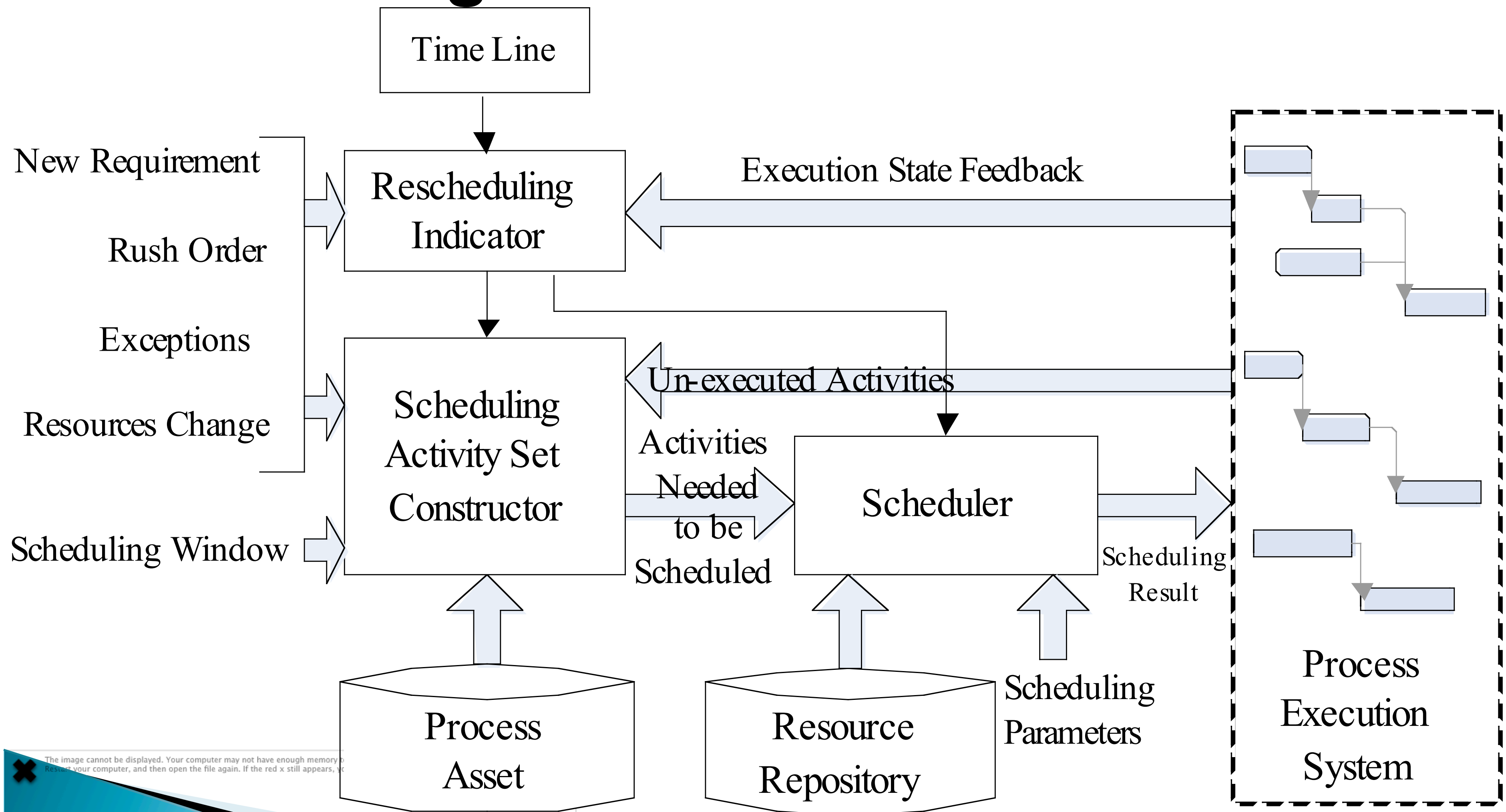


Both resources allocated for entire step

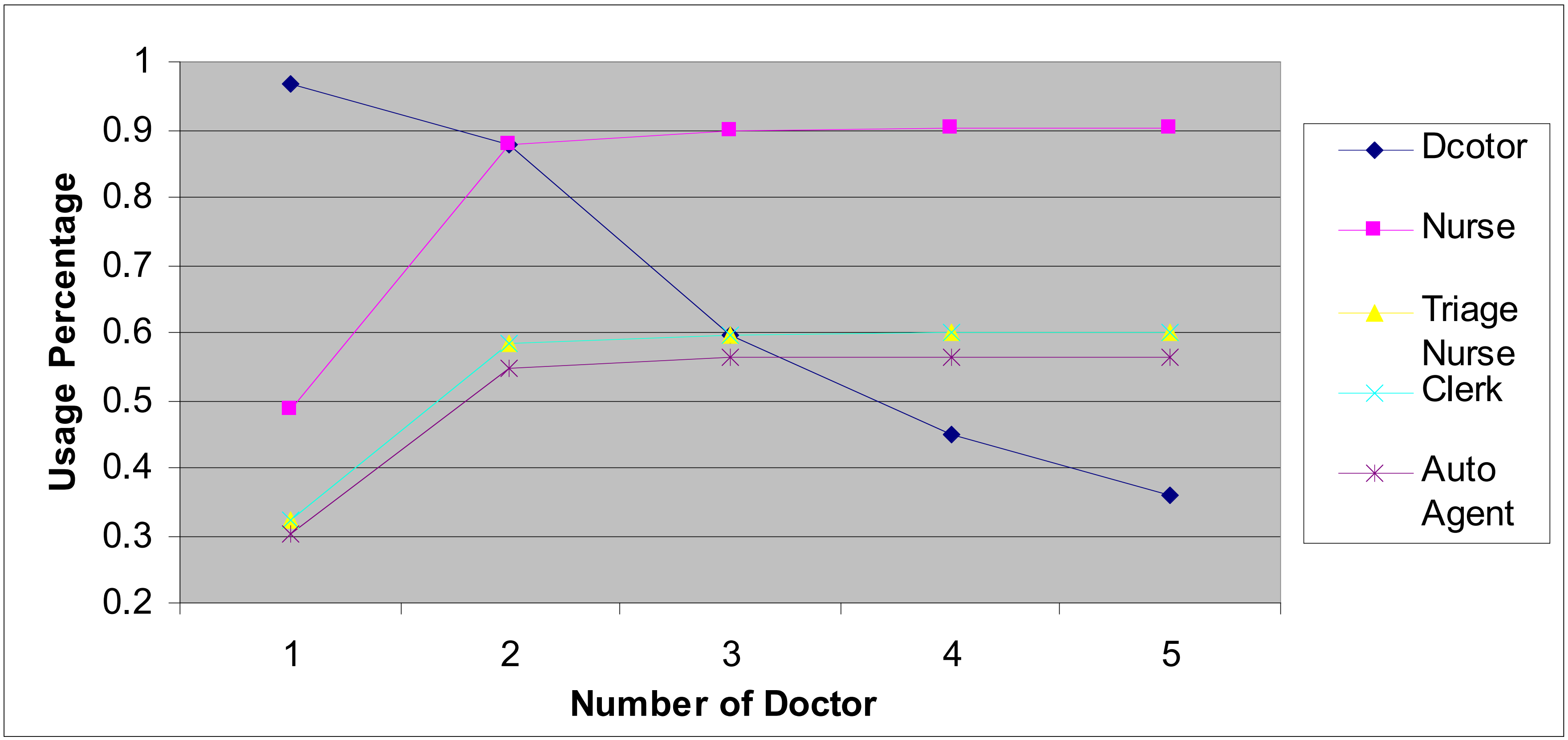
Allocation based on lack of substep detail, but assigned only when needed for substep

Each resource allocated only when needed for substep

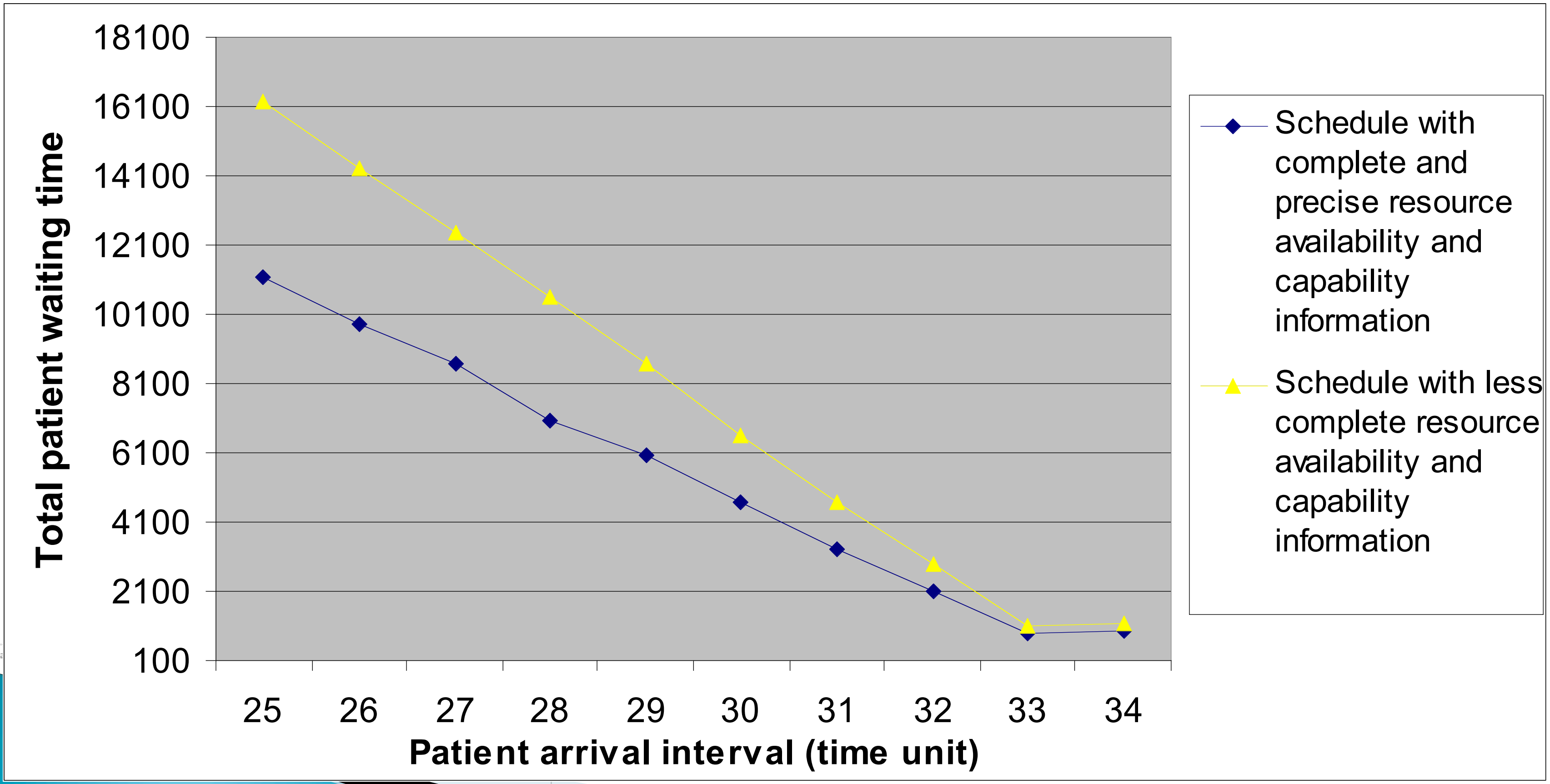
The TWINS Incremental Resource Scheduling Framework



Resource utilization rate as number of doctors increases

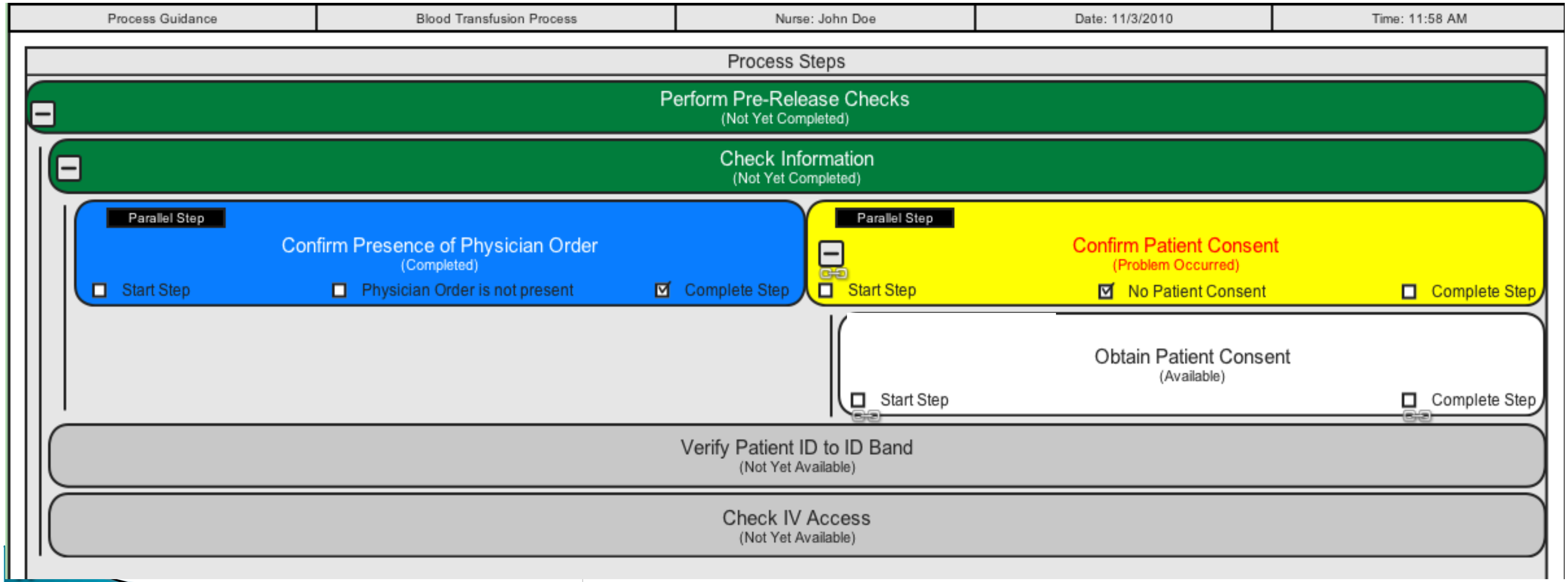


Effect of resource specification detail on scheduling effectiveness



How to communicate system information and provide guidance to humans?

- Visualization of current, historical, and prospective views
  - Warn of impending events
- Mock-up of process progress for a blood transfusion



How to gather, display, and exploit: Historical Execution Information

- Gather and display historical information
  - Present relevant contextual information
  - Summarize historical performance
  - Identify situations that tend to cause errors, exceptional circumstances, bottlenecks
- Gather probabilities that can sharpen the static analysis
  - More accurate projection of vulnerabilities
- Basis for process comparisons
  - Fine-grained assessment of differences, not just in terms of outcomes
- Basis for system and process improvement
  - In collaboration with domain experts