ArchE – An Architecture Design Assistant

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Len Bass
August 2, 2007

Outline

What is ArchE?

What problem are we going to demonstrate?

What is input to ArchE?

What is a reasoning framework?

What is ArchE?

ArchE is a software architecture design assistant, which:

- Takes quality and functional requirements as input
- Elicits key quality attribute information to refine quality requirements
- Elicits key architectural information
- Derives candidate architectures
- Evaluates whether quality requirements are satisfied
- Identifies tradeoffs
- Suggests alternative architectures

ArchE is implemented in Eclipse using Java and the JESS expert system.

What does ArchE "know"?

ArchE "knows":

- Architecture design process – how to get an architecture from requirements
- Quality knowledge – how to achieve required qualities in an architecture design
- What questions to ask – how to get the architect to think precisely about architectural design.

Key principle: Quality attribute requirements are primary drivers for architecture design and models capture the relations between architecture and desired results.

Sample Problem - Clemson Transit Assistance System (CTAS)

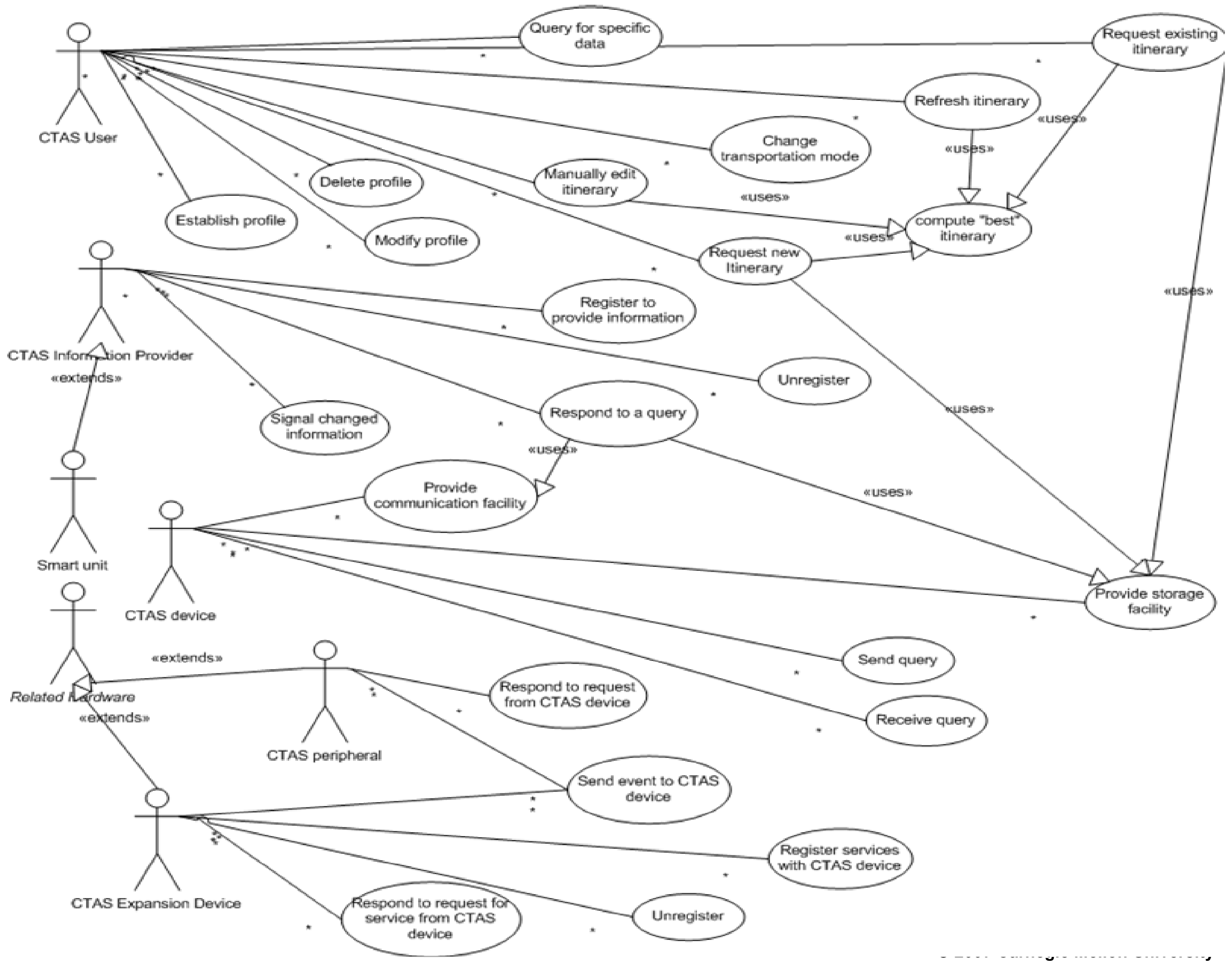
Wireless hand held itinerary planning system

User can plan routes and modes of transportation

Traveler can periodically update information on CTAS and reconsider itinerary.

External information services (hotel, transit systems, parking lot information) assumed.

Use Cases

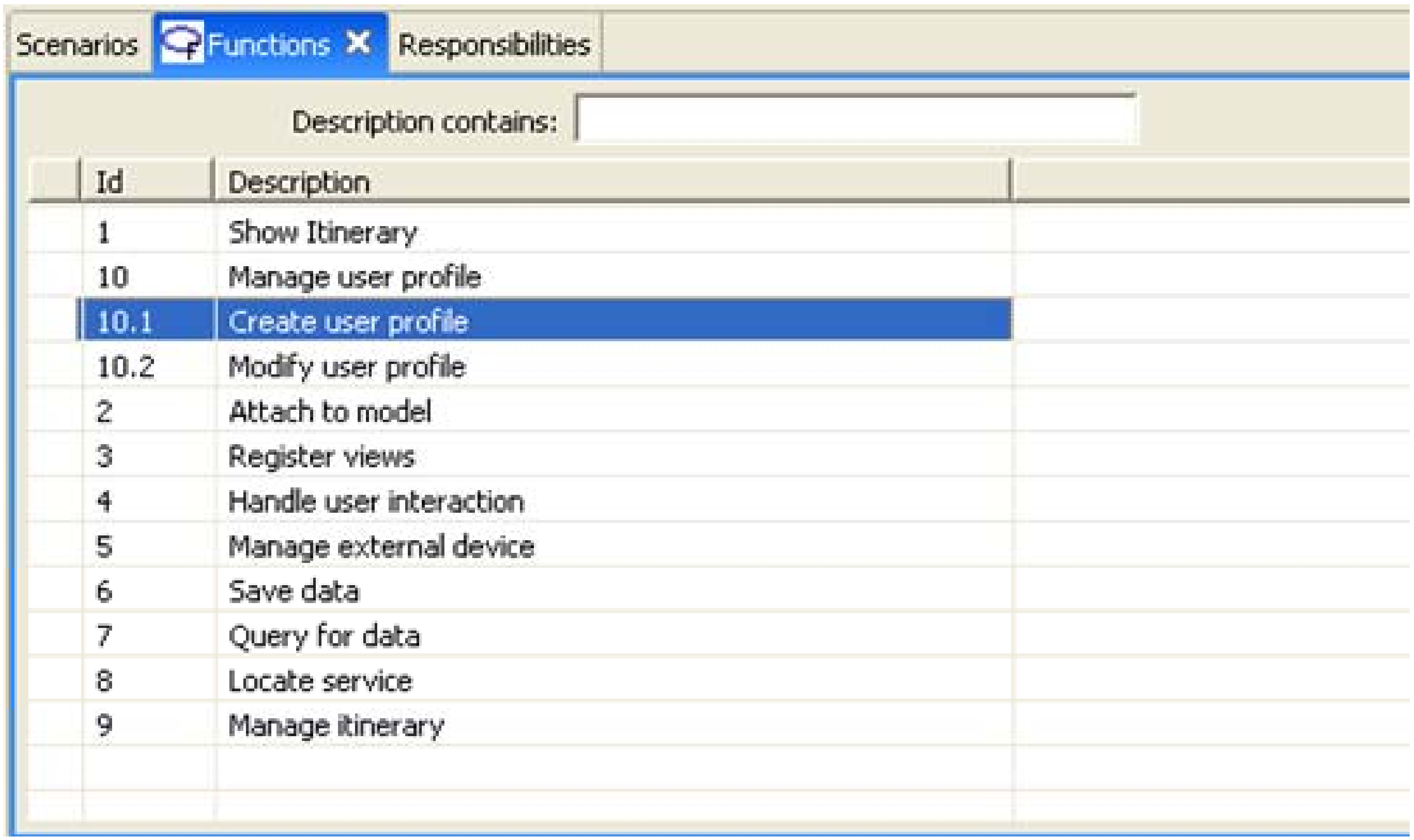

CTAS User
Query for specific data
Request existing itinerary
Refresh itinerary
Change transportation mode
Manually edit itinerary
Delete profile
Establish profile
Modify profile
compute "best" itinerary
Request new Itinerary
«uses»
CTAS Information Provider
Register to provide information
Unregister
Respond to a query
Signal changed information
«extends»
Smart unit
Provide communication facility
CTAS device
Provide storage facility
Send query
Receive query
Related Hardware
CTAS peripheral
Respond to request from CTAS device
Send event to CTAS device
CTAS Expansion Device
Register services with CTAS device
Unregister
Respond to request for service from CTAS device

Initial Input to ArchE

Functions with dependency relations

Quality requirements expressed as quality attribute scenarios

Initial Functions for CTAS



Description contains:

Id	Description
1	Show Itinerary
10	Manage user profile
10.1	Create user profile
10.2	Modify user profile
2	Attach to model
3	Register views
4	Handle user interaction
5	Manage external device
6	Save data
7	Query for data
8	Locate service
9	Manage itinerary

Responsibilities[1]

“Responsibilities” are fundamental to the design process ArchE supports

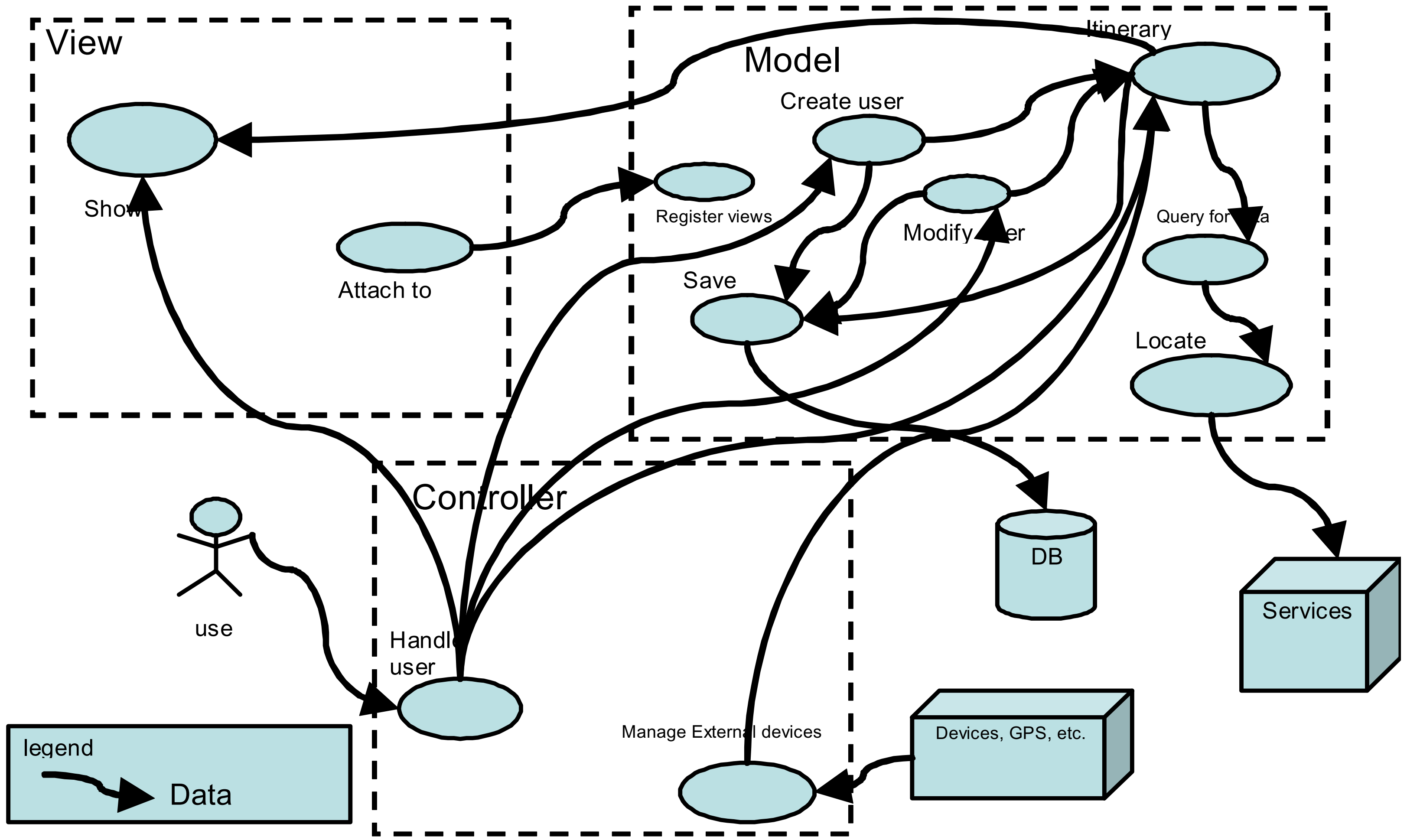
Responsibilities are general statements about an architectural element and include: the actions an element performs, the knowledge an element maintains, major decisions an element makes that affect others.

ArchE maps functions into responsibilities

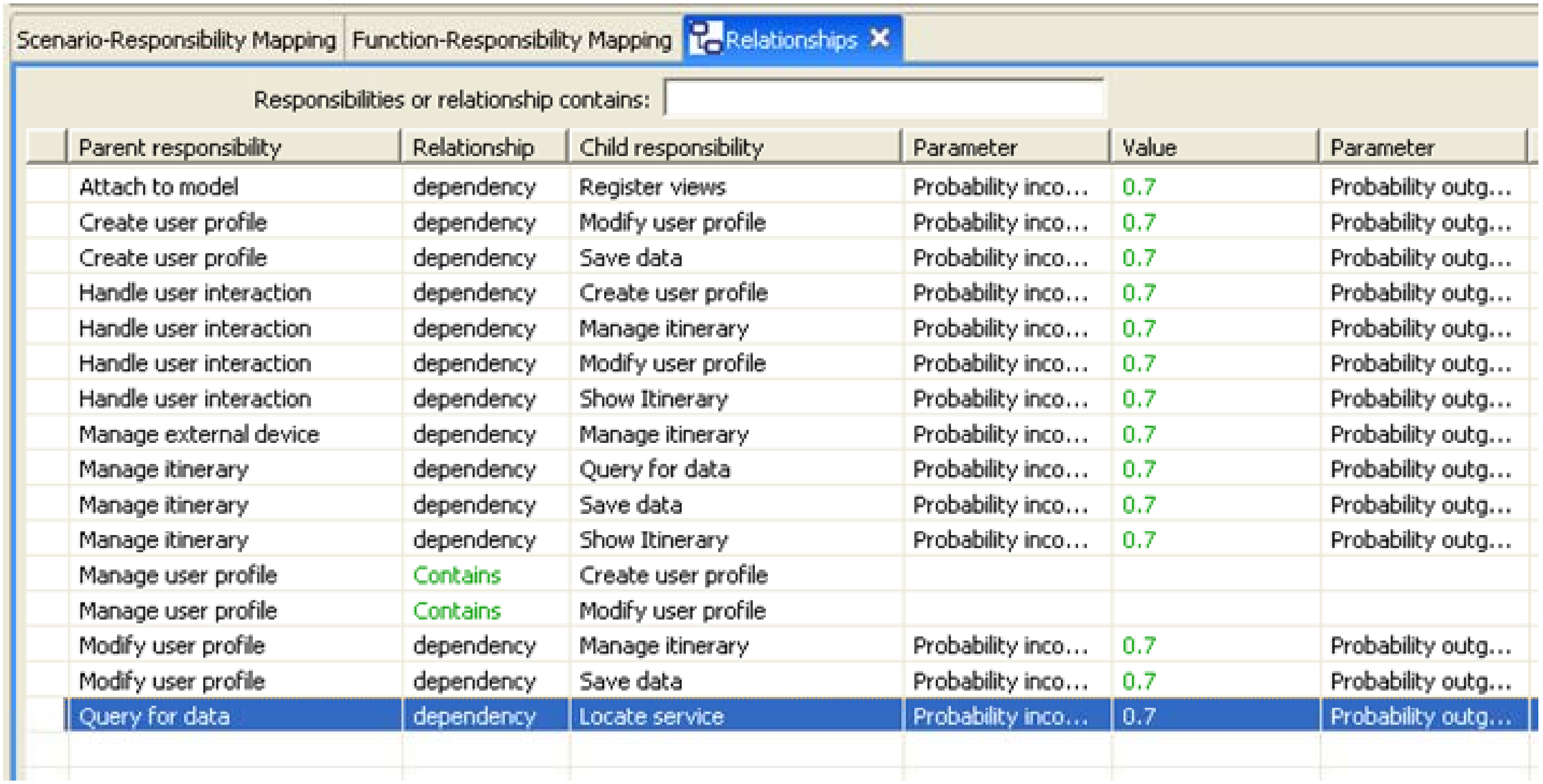
User specifies relationships among responsibilities manually.

1. Wirfs-Brock, R. and McKean, A. *Object Design*. Boston, MA: Addison-Wesley, 2003.

Responsibility graph for CTAS



Relationships among responsibilities


Scenario-Responsibility Mapping
Function-Responsibility Mapping
Relationships

Responsibilities or relationship contains:

	Parent responsibility	Relationship	Child responsibility	Parameter	Value	Parameter
	Attach to model	dependency	Register views	Probability inco...	0.7	Probability outg...
	Create user profile	dependency	Modify user profile	Probability inco...	0.7	Probability outg...
	Create user profile	dependency	Save data	Probability inco...	0.7	Probability outg...
	Handle user interaction	dependency	Create user profile	Probability inco...	0.7	Probability outg...
	Handle user interaction	dependency	Manage itinerary	Probability inco...	0.7	Probability outg...
	Handle user interaction	dependency	Modify user profile	Probability inco...	0.7	Probability outg...
	Handle user interaction	dependency	Show Itinerary	Probability inco...	0.7	Probability outg...
	Manage external device	dependency	Manage itinerary	Probability inco...	0.7	Probability outg...
	Manage itinerary	dependency	Query for data	Probability inco...	0.7	Probability outg...
	Manage itinerary	dependency	Save data	Probability inco...	0.7	Probability outg...
	Manage itinerary	dependency	Show Itinerary	Probability inco...	0.7	Probability outg...
	Manage user profile	Contains	Create user profile			
	Manage user profile	Contains	Modify user profile			
	Modify user profile	dependency	Manage itinerary	Probability inco...	0.7	Probability outg...
	Modify user profile	dependency	Save data	Probability inco...	0.7	Probability outg...
	Query for data	dependency	Locate service	Probability inco...	0.7	Probability outg...

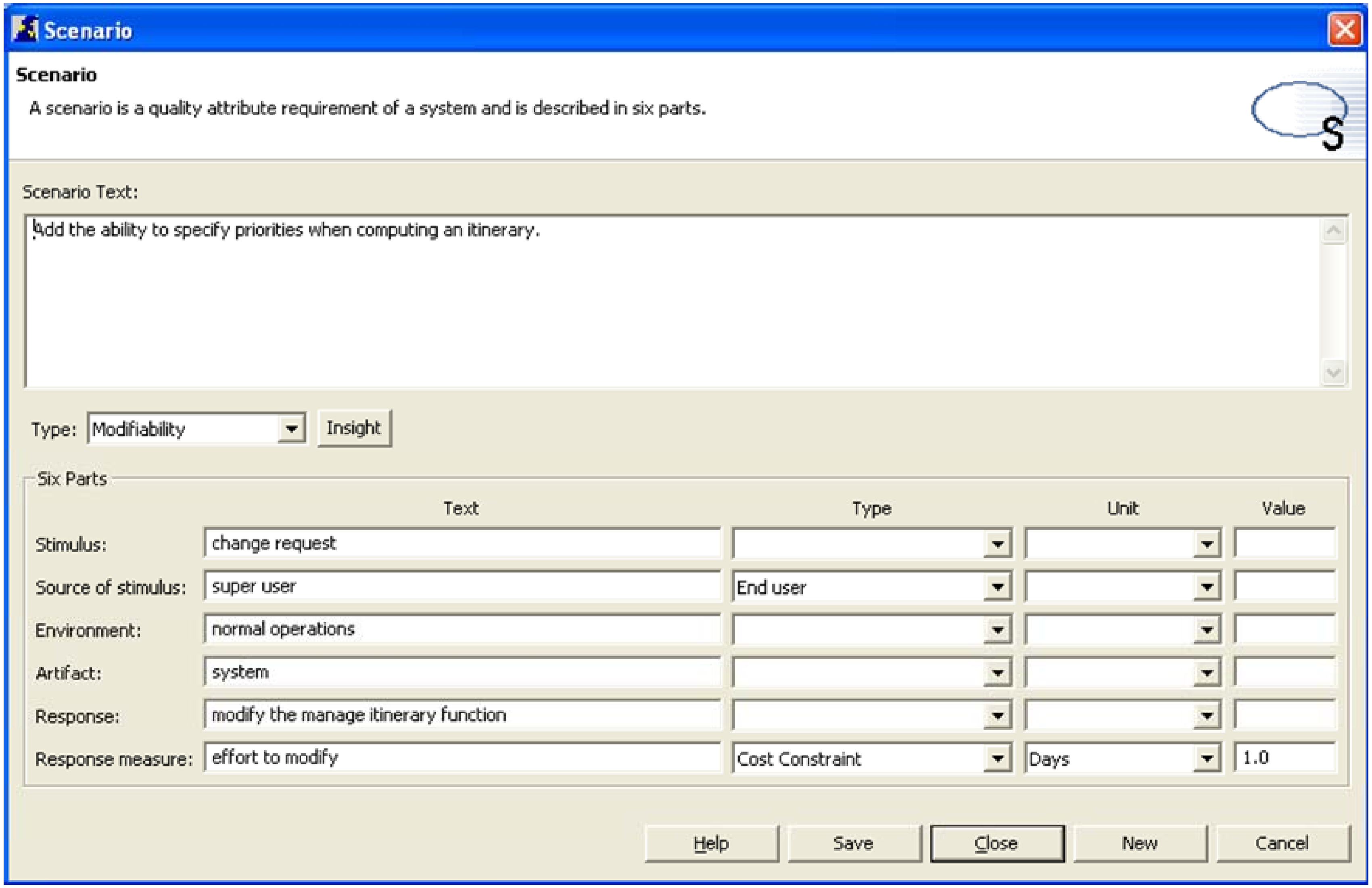
Quality Attribute Scenarios

Two modifiability scenarios for now:

1) Add the ability to specify priorities when computing an itinerary. The effort for adding the function should be less than 1 person day.

2) Add a function to notify others of late arrival. The effort for adding the function should be less than .5 person days.

Scenario addition screen



Scenario

A scenario is a quality attribute requirement of a system and is described in six parts.

Scenario Text:

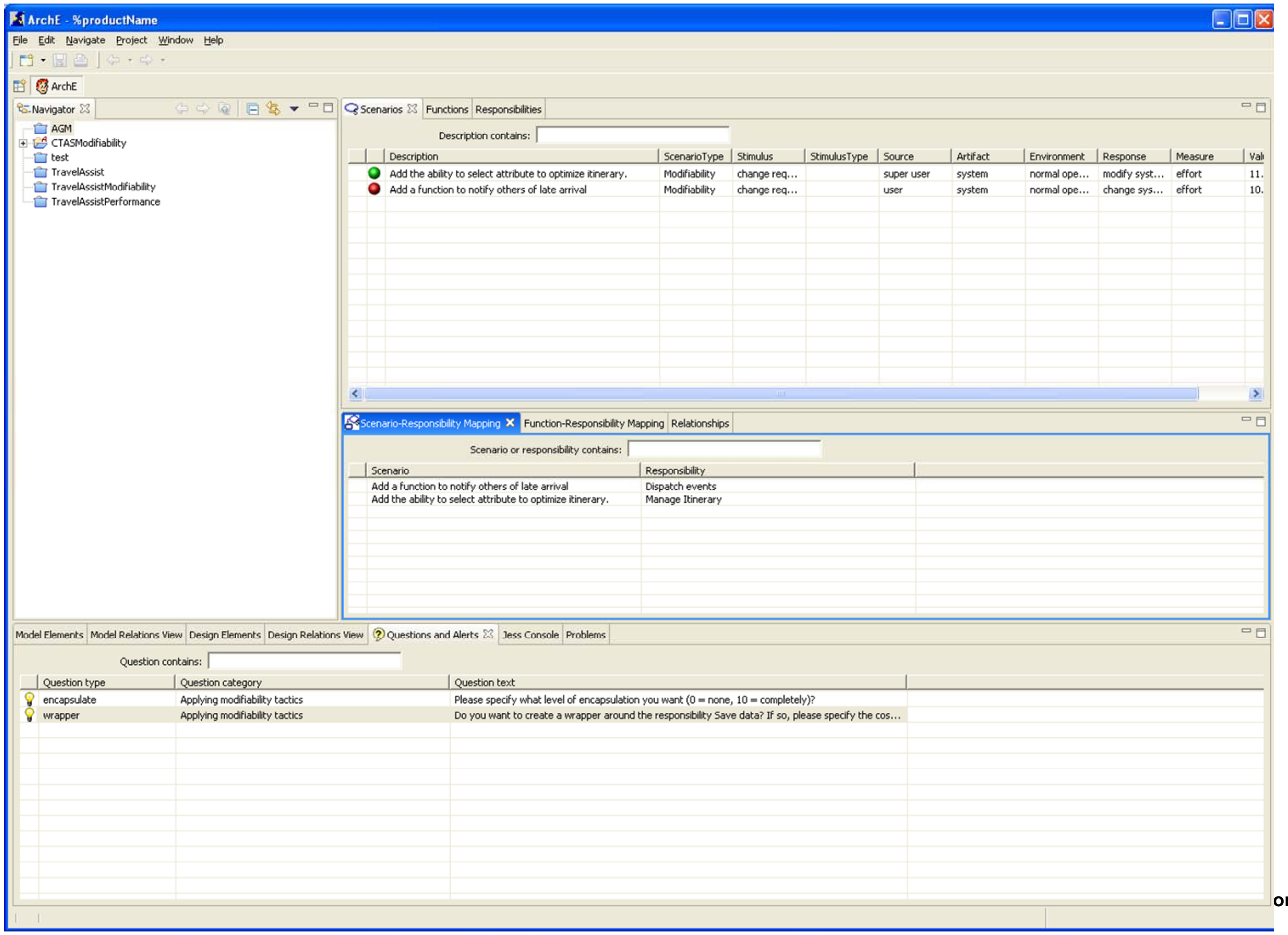
Add the ability to specify priorities when computing an itinerary.

Type: Modifiability

Six Parts

	Text	Type	Unit	Value
Stimulus:	change request			
Source of stimulus:	super user	End user		
Environment:	normal operations			
Artifact:	system			
Response:	modify the manage itinerary function			
Response measure:	effort to modify	Cost Constraint	Days	1.0

Scenarios must be related to responsibilities (manually)



Description contains:

	Description	ScenarioType	Stimulus	StimulusType	Source	Artifact	Environment	Response	Measure	Val
🟢	Add the ability to select attribute to optimize itinerary.	Modifiability	change req...		super user	system	normal ope...	modify syst...	effort	11.
🔴	Add a function to notify others of late arrival	Modifiability	change req...		user	system	normal ope...	change sys...	effort	10.

Scenario or responsibility contains:

Scenario	Responsibility
Add a function to notify others of late arrival	Dispatch events
Add the ability to select attribute to optimize itinerary.	Manage Itinerary

Question contains:

Question type	Question category	Question text
encapsulate	Applying modifiability tactics	Please specify what level of encapsulation you want (0 = none, 10 = completely)?
wrapper	Applying modifiability tactics	Do you want to create a wrapper around the responsibility Save data? If so, please specify the cos...

ArchE reasoning framework

ArchE uses a modifiability reasoning framework to reason about the scenarios.

What is a reasoning framework?

What is the modifiability reasoning framework?

Reasoning Frameworks

A reasoning framework is a vehicle for encapsulating the quality attribute knowledge and the tools needed to analyze the behavior of a system with respect to some quality attribute

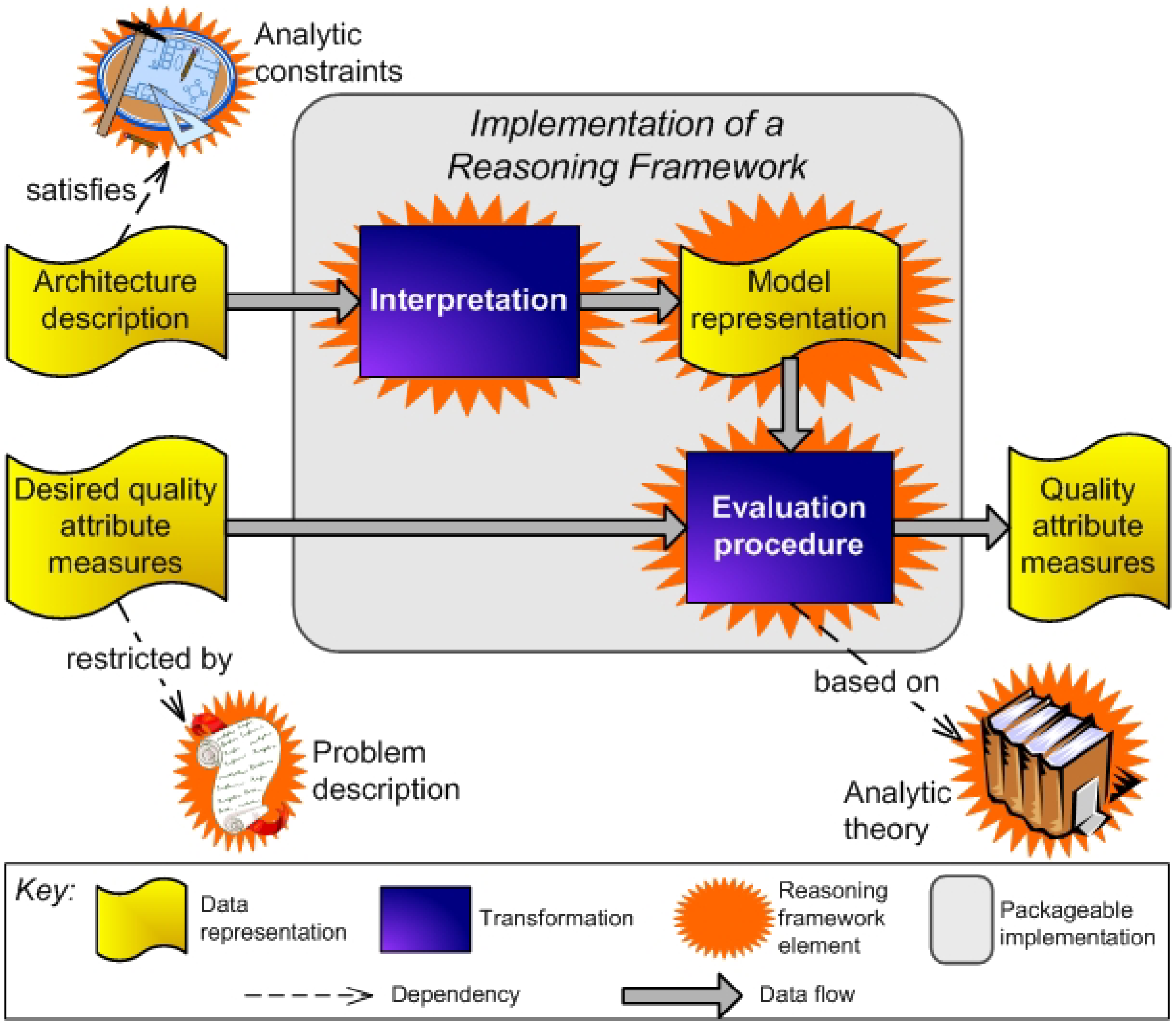
Can be used:

- To predict behavior before the system is built
- Understand behavior after it is built
- Make design decisions while it is being built

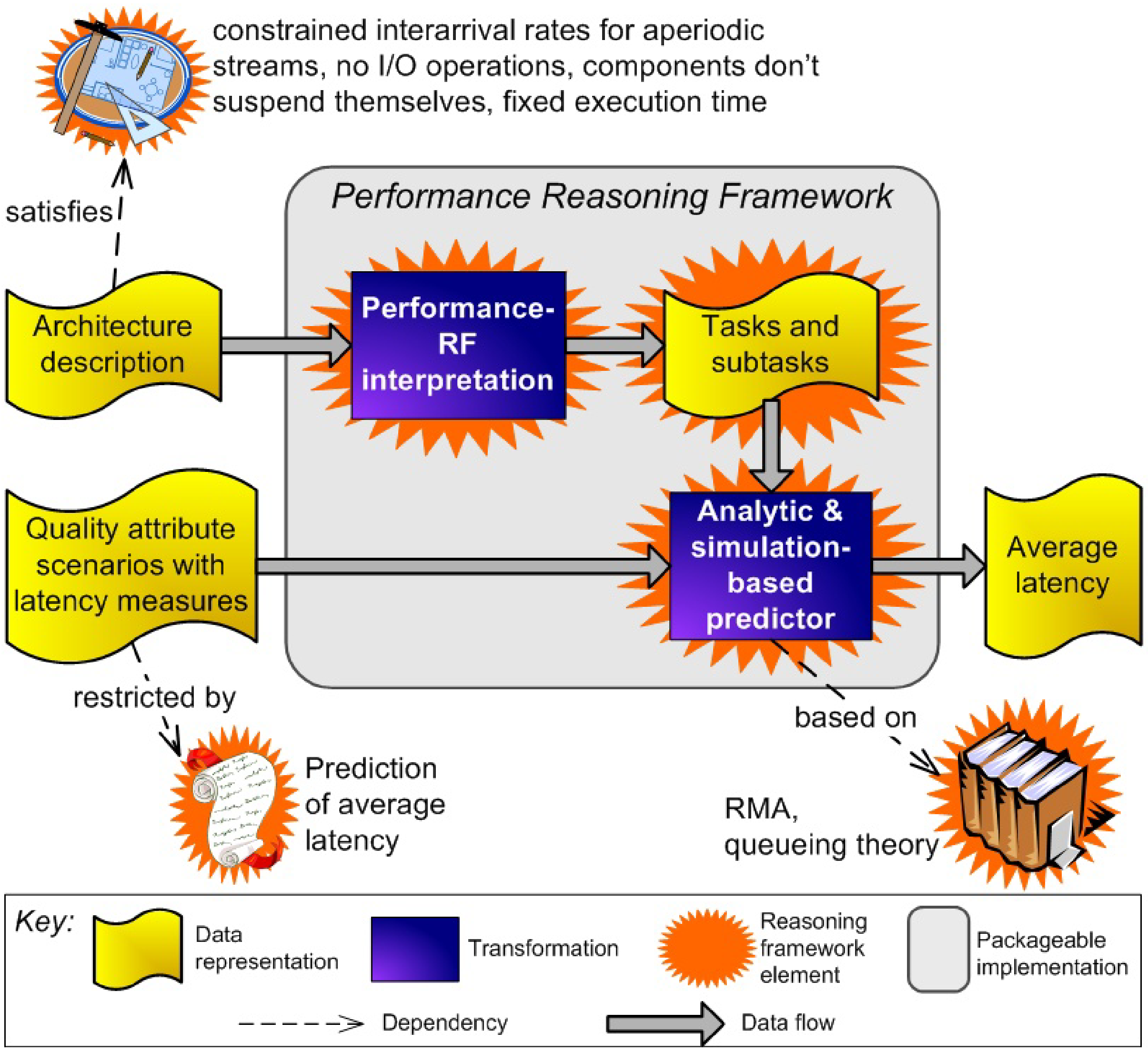
Reason for encapsulating quality attribute knowledge is to enable incorporation of quality attribute knowledge in ArchE without requiring quality attributes to know about each other.

Elements of a Reasoning Framework

1. Problem description
2. Analytic theory
3. Analytic constraints
4. Model representation
5. Interpretation
6. Evaluation procedure



Example: Performance Reasoning Framework


constrained interarrival rates for aperiodic streams, no I/O operations, components don't suspend themselves, fixed execution time
satisfies
Performance Reasoning Framework
Architecture description
Performance-RF interpretation
Tasks and subtasks
Quality attribute scenarios with latency measures
Analytic & simulation-based predictor
Average latency
restricted by
Prediction of average latency
based on
RMA, queueing theory
Key:
Data representation
Transformation
Reasoning framework element
Packageable implementation
Dependency
Data flow

Modifiability Reasoning Framework - 1

Based on coupling and cohesion concepts.

Modules are coupled to each other:

- Tightly (high probability of change propagating)
- Medium (medium probability of change propagating)
- Low (low probability of change propagating)

Responsibilities are assigned to modules.

Cost of change is assigned to each responsibility.

A change to one responsibility in a module is assumed to propagate to other responsibilities in the module.

Modifiability Reasoning Framework - 2

Modifiability scenario is tied to the modification of several responsibilities

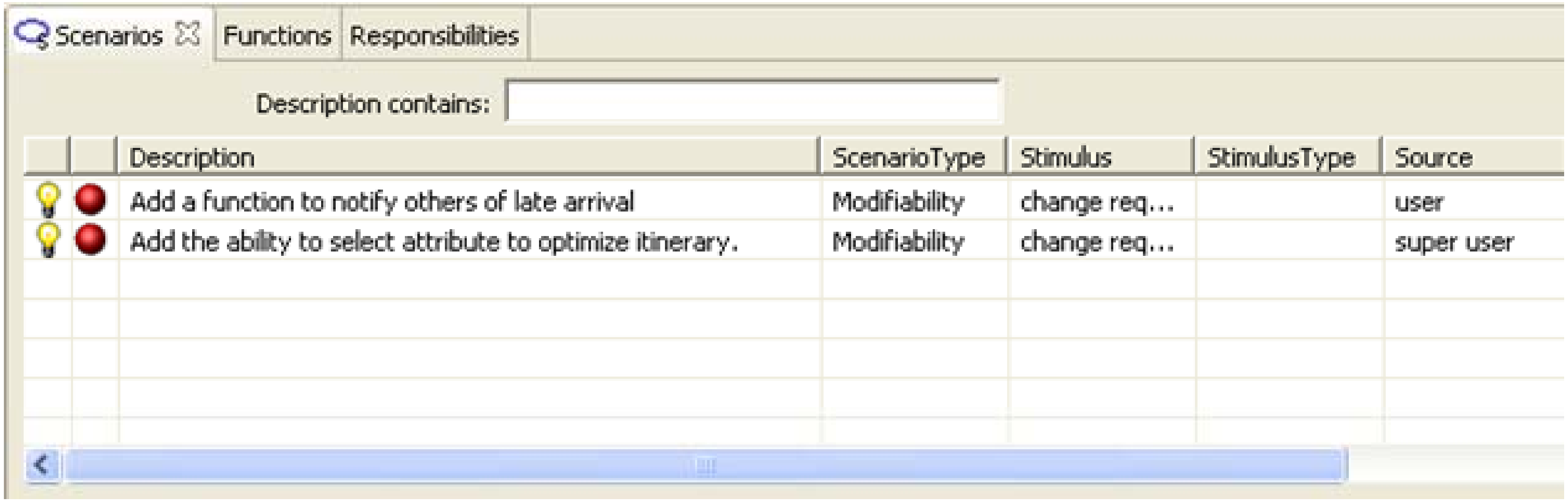
Each responsibility has a cost of change and a probability of propagating to other responsibilities. Each of the propagated to responsibilities, in turn, has a cost of change and a probability of propagating to additional responsibilities.

Sum the costs weighted by the probability of a responsibility being changed.

ArchE calculations for CTAS

ArchE calculates whether cost of change for particular scenario is within bounds.

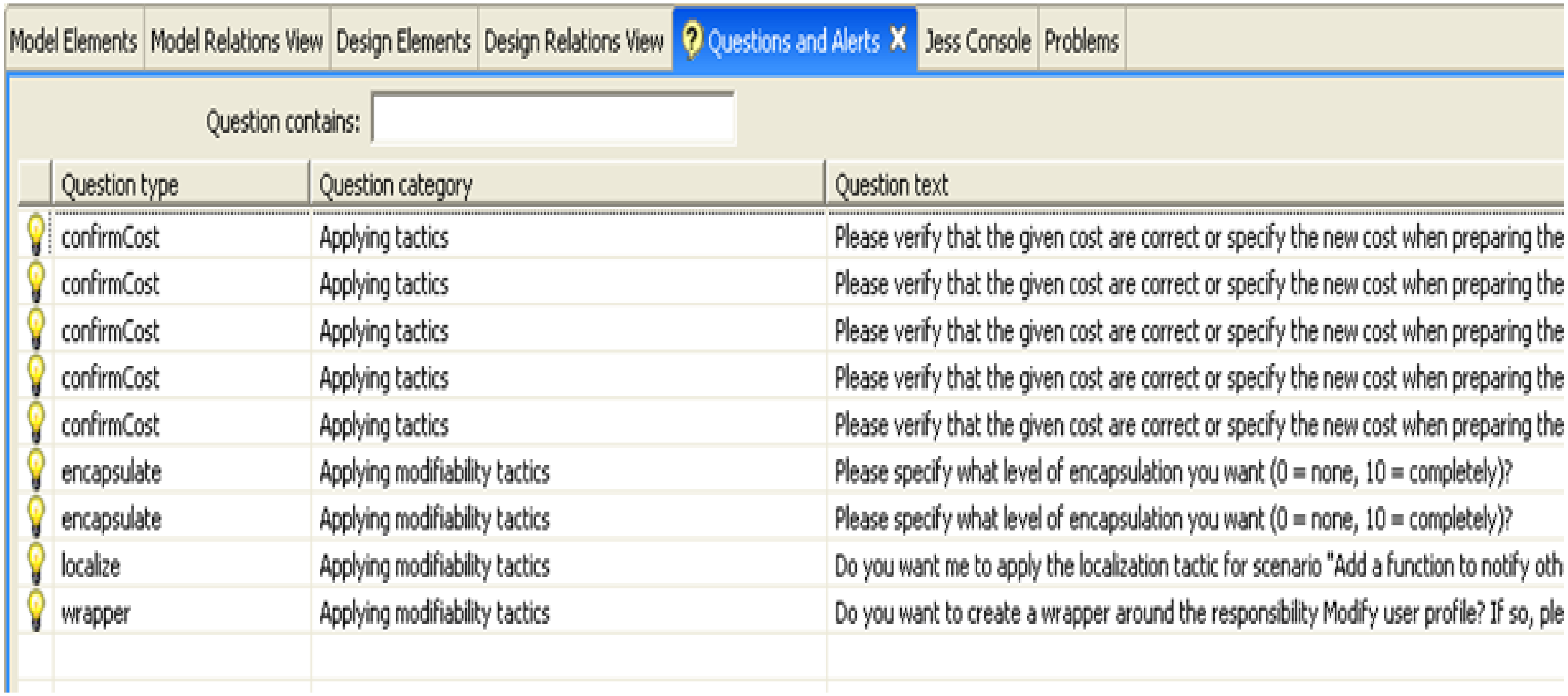
Out of bounds is indicated by red light.



		Description	ScenarioType	Stimulus	StimulusType	Source
💡	🔴	Add a function to notify others of late arrival	Modifiability	change req...		user
💡	🔴	Add the ability to select attribute to optimize itinerary.	Modifiability	change req...		super user

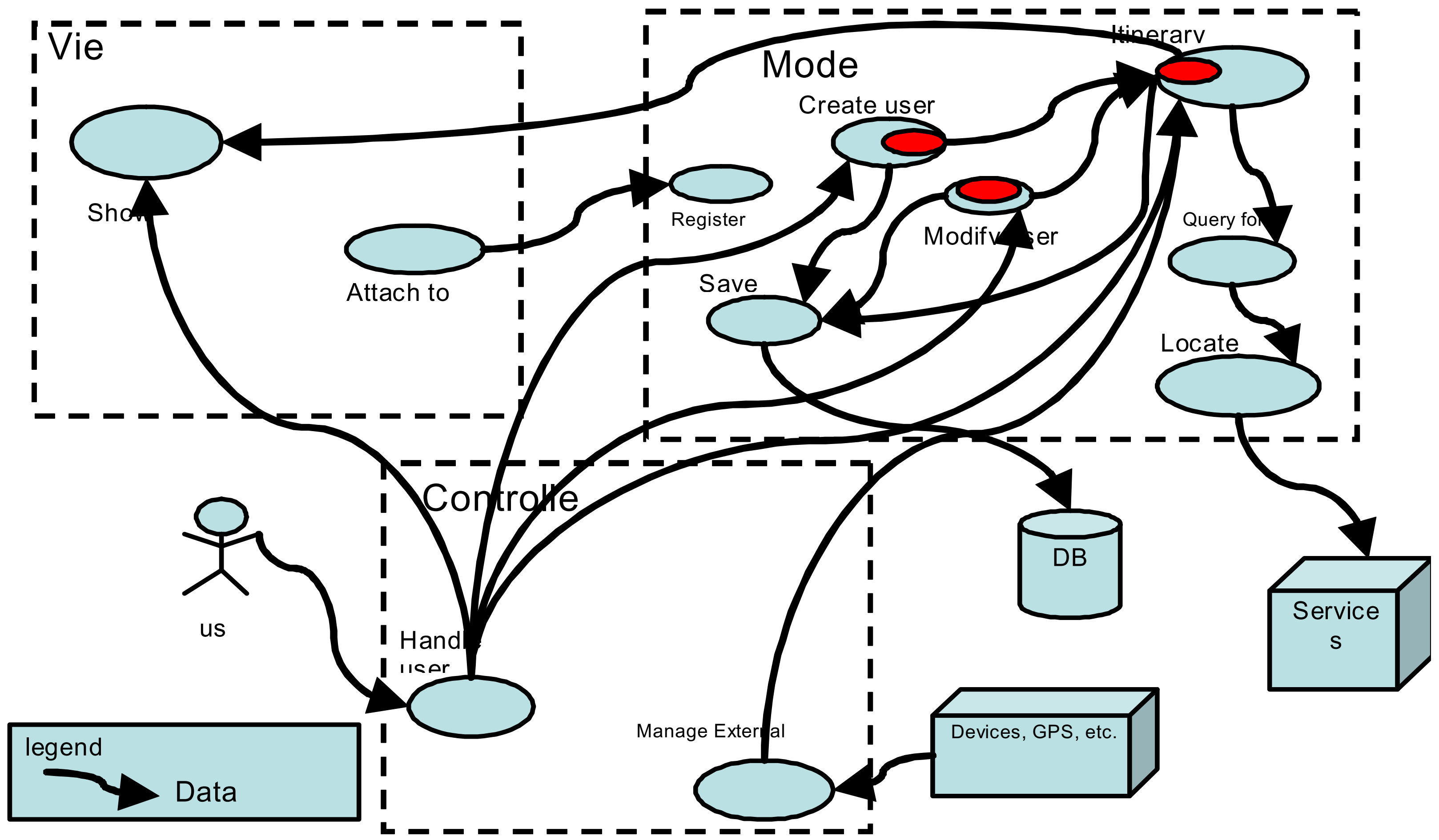
Scenario – Notify others of late arrival

ArchE suggests several tactics – encapsulate and localize:

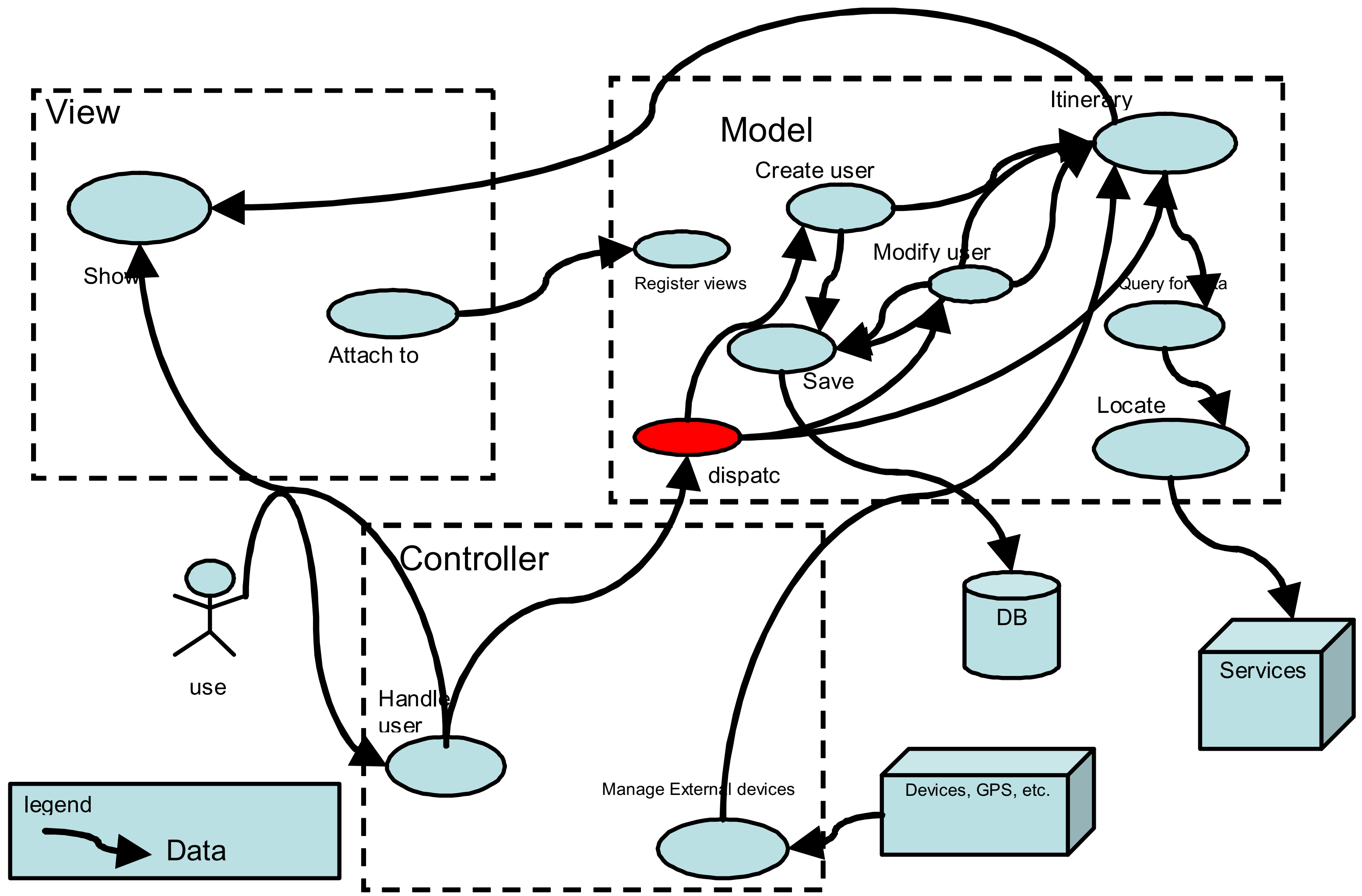


	Question type	Question category	Question text
💡	confirmCost	Applying tactics	Please verify that the given cost are correct or specify the new cost when preparing the
💡	confirmCost	Applying tactics	Please verify that the given cost are correct or specify the new cost when preparing the
💡	confirmCost	Applying tactics	Please verify that the given cost are correct or specify the new cost when preparing the
💡	confirmCost	Applying tactics	Please verify that the given cost are correct or specify the new cost when preparing the
💡	confirmCost	Applying tactics	Please verify that the given cost are correct or specify the new cost when preparing the
💡	encapsulate	Applying modifiability tactics	Please specify what level of encapsulation you want (0 = none, 10 = completely)?
💡	encapsulate	Applying modifiability tactics	Please specify what level of encapsulation you want (0 = none, 10 = completely)?
💡	localize	Applying modifiability tactics	Do you want me to apply the localization tactic for scenario "Add a function to notify oth
💡	wrapper	Applying modifiability tactics	Do you want to create a wrapper around the responsibility Modify user profile? If so, ple

Localize changes – before



Localize changes - after



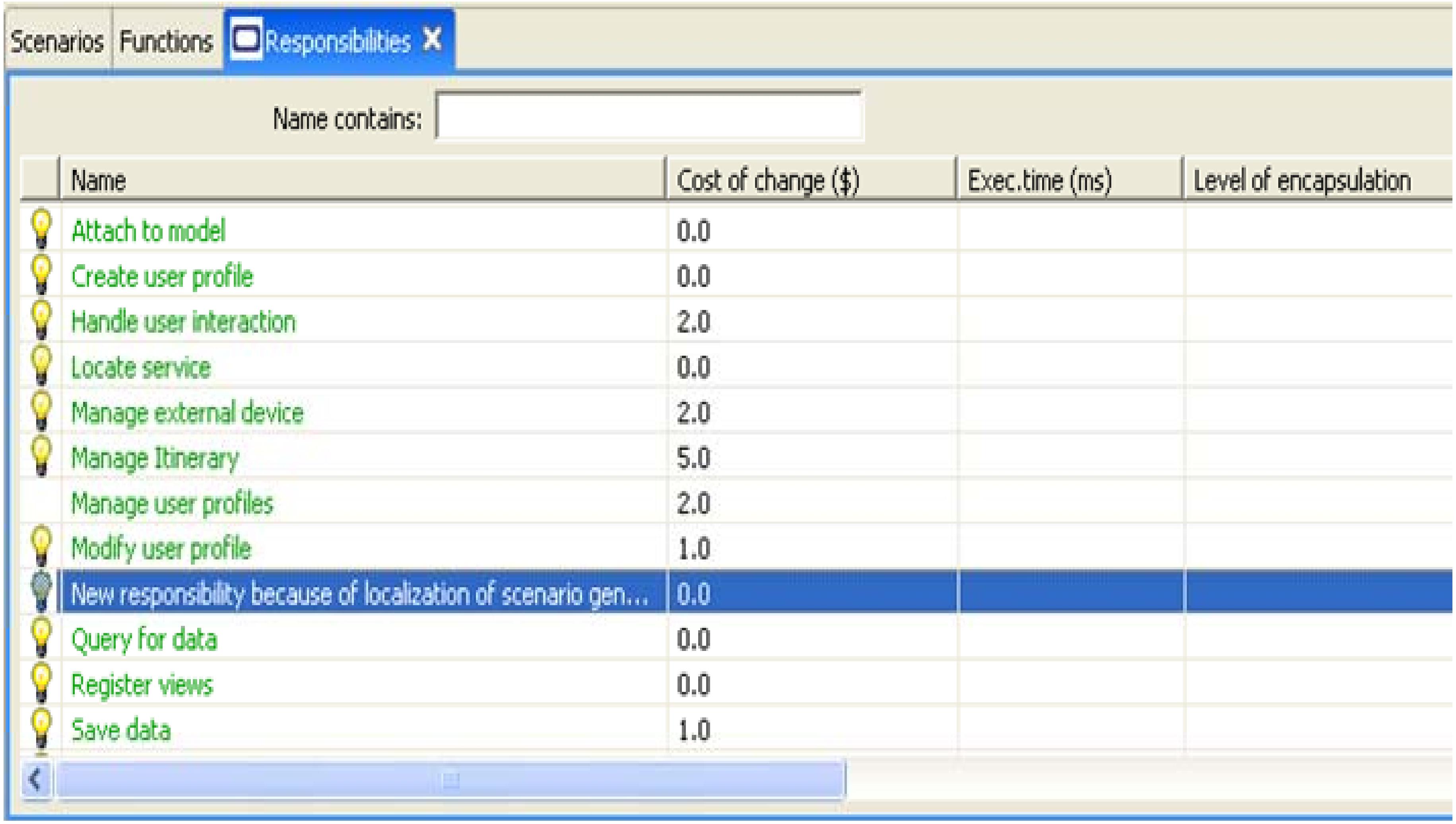
ArchE creates new responsibility

ArchE does not know semantics of application – architect must label new responsibility. In CTAS it is called “dispatch”

Cost of change must be entered for “dispatch”

New probabilities of propagation must be entered for “dispatch”

New responsibility in ArchE



Name contains:

	Name	Cost of change ($)	Exec.time (ms)	Level of encapsulation
💡	Attach to model	0.0		
💡	Create user profile	0.0		
💡	Handle user interaction	2.0		
💡	Locate service	0.0		
💡	Manage external device	2.0		
💡	Manage Itinerary	5.0		
	Manage user profiles	2.0		
💡	Modify user profile	1.0		
💡	New responsibility because of localization of scenario gen...	0.0		
💡	Query for data	0.0		
💡	Register views	0.0		
💡	Save data	1.0		

Continuing with ArchE

Architect continues choosing one tactic at a time.

ArchE has reasoning frameworks for modifiability and real time performance.

Architect interacts, choosing tactics until all of the scenarios have been satisfied.

The resulting design is then exported.

Use of ArchE

ArchE has been used to support a graduate class in software architecture at Clemson University

Student feedback:.

- The overall concept is very convincing… with a little refining the software should be great.
- The good thing about ArchE during the architecture design process is that it automatically computes the effort of changing one quality attribute on the whole architecture
- The scenario based approach makes it easier to think about how architectural decisions will impact the required quality attributes of a system.

ArchE now and in the future

ArchE and the ArchE Users’ Guide can be downloaded from

http://www.sei.cmu.edu/architecture/arche.html

The available version of ArchE has reasoning frameworks for modifiability and real time performance

Toward the end of this year, we will distribute a version of ArchE that is extensible in reasoning frameworks.

- A researcher in quality attributes generates a reasoning framework embodying their theory
- ArchE will manage trade offs with other quality attributes
- ArchE will enable a comparison of a particular theoretical approach to other approaches for the for the same quality attribute.

DEMO